An Automatic Wi-Fi-Based Approach for Extraction of User Places and Their Context

Luka Vidmar, 1 Mitja Štular, 1 Andrej Kos, 2 and Matevž Pogačnik 2

1 Telekom Slovenije, d.d., Cigaletova ulica 15, SI-1000 Ljubljana, Slovenia
2 Faculty of Electrical Engineering, University of Ljubljana, Tržaška cesta 25, SI-1000 Ljubljana, Slovenia

Correspondence should be addressed to Luka Vidmar; luka.vidmar@telekom.si

Received 29 October 2014; Revised 26 February 2015; Accepted 3 March 2015

Academic Editor: Rung Ching Chen

Copyright © 2015 Luka Vidmar et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the analysis of various sensor data from the mobile devices, it is possible to extract user situations, so-called user context. This is needed for the development of modern, user-friendly services. Therefore, we developed a simple, nonintrusive, and automatic method based on the Wi-Fi fingerprints and GPS. The method finds user stay points, aggregates them into meaningful stay regions, and assigns them four general user contexts: home, work, transit, and free time. We evaluated its performance on the real traces of six different users who annotated their contexts over eight days. The method determined the stay mode of the users with accuracy, precision, and recall of above 96%. In combination with the novel approach for aggregation, all regions relevant to the users were determined. Among the tested aggregation schemes, the fingerprint similarity approach worked the best. The context of the determined stay regions was on average accurately inferred in 98% of the time. For the contexts home, work, and free time, the precision and recall exceeded 86%. The results indicate that the method is robust and can be deployed in various fields where context awareness is desired.

1. Introduction

Ubiquitous or pervasive systems, as imagined by Weiser, are becoming reality with advancements in mobile computing, wireless connectivity, and sensing of the environment [1]. Mobile communications have become the most readily available and widespread technology. Widely used smartphones [2] contain an increasing number of communication technologies, such as Wi-Fi, infrared (IR), Bluetooth, radio-frequency identification (RFID), and global positioning system (GPS), as well as sensor modules, such as an accelerometer, compass, microphone, and camera. Measurements collected by these sensor probes can help extract current user location and other circumstances, so-called user context. User context is any information that can be used to characterise the situation of entities, such as a person, place, or object, that are considered relevant to the interaction between the user and the application, including the user and the application themselves [3]. While some services, for example, navigation, require accurate physical location, other services prefer locales that carry a particular semantic meaning, such as a shopping mall or restaurant, and user context, such as home or work. Such locales and user context can be used, for example, in the multimedia applications, in the recommender systems, and in the fleet management. In multimedia applications, the applications can become more user-friendly. In recommender systems, only relevant and personalised information, applications, and services can be suggested to the user [4]. In fleet management, automatic control of fleet can be achieved.

To achieve good performance, semantics of locales and user context used to be annotated manually, but this kind of intervention is not practical for the user. Therefore, automatic and nonintrusive solutions are preferred. However, inference without user intervention is challenging due to the unstable mobile environment and the high density of locales with semantic meaning.

The method for automatic inference of user context needs to trustworthy determine locales, movements between them and revisits. The problem is that the locales with a specific meaning to a user often considerably differ in size. For example, the user context work consists of both, a small locale
representing an office and a large locale representing the entire company premises with several floors of offices. The open research question is how to automatically extract small detailed user locales and at the same time determine the bigger units with the same semantic meaning to a user. In the literature there is very little focus on this issue. For example, in PlaceSense [5] the size of locales is arbitrary, because it is limited by the visibility of at least one representative Wi-Fi beacon: inside the building a locale can encompass a few rooms nearby in the same or neighbouring floors, while outdoors it covers a much larger area. The SensLoc method [6] is better than PlaceSense in the recognition of close-by detailed locales but does not anticipate the possibility that several inferred places can constitute a more general locale with the same context to a user, for example, several shops in a mall as shopping or offices and meeting rooms in the company premises as work. In general, the existing place discovery algorithms, such as PlaceSense [5] and SensLoc [6], do not reason about the actual semantics of discovered places. On the other hand, the algorithms that do assign semantic meaning to the recognized places require labelled data and do not infer more detailed user contexts within the general context categories.

In this paper, we propose a method that automatically, without user input, infers various-sized locales with the same semantic meaning and assigns them four different user contexts: home, work, free time, and transit. The outline of the method is as follows: the method (i) uses Wi-Fi fingerprints and physical locations, collected by sensors on the smartphones; (ii) determines user trajectories, that is, stay points (in the related literature, locales with semantic meaning are named stay points when they are inferred from the GPS data and places when they are inferred from the Wi-Fi data; in our paper, we use the term stay points for both) and paths between them; (iii) aggregates the determined stay points into meaningful stay regions (e.g., different rooms are aggregated into an apartment and floors into a building); and (iv) infers user context based on spatiotemporal features of determined stay regions.

Our main innovations are the following. (i) We introduce five optimization parameters to the best existing method, SensLoc [6]. These parameters improve the robustness of stay points determination, as well as inference of user static versus dynamic mode. (ii) We develop a novel approach that aggregates nearby stay points with the same semantic meaning to a user into meaningful stay regions. Thus, we determine more general locales of the same user context while maintaining the details of belonging stay points. (iii) Our method automatically, without labelled data, assigns one of four general user contexts to each determined stay region. For stay points within stay regions, we demonstrate how the more detailed user contexts could be extracted.

We evaluated our method by comparing it to the best-known existing methods. We use two annotated datasets from mobile users: the first was real life traces collected by six users over eight days and the second dataset was a scripted tour of one user within one apartment. With these datasets, we evaluate the determination of user static versus dynamic mode, the determination of stay points and their room-level accuracy, and the correctness of aggregation and of context inference. On our datasets, the proposed method outperforms, to our knowledge, the best method for determination of stay points. Moreover, the method is able to infer the user context comparably well as other similar methods.

The paper is organised as follows. Section 2 presents the related work. Section 3 provides the details of the proposed method. Section 4 deals with the evaluation of our method and Section 5 discusses the results. Section 6 draws the conclusions.

2. Related Work

2.1. Determination of Stay Points and Stay Regions. GPS is the most widely used technology for inference of stay points and user context [7–14], followed by Wi-Fi [5, 15–17] and a combination of several technologies, including cell IDs (CIDs), Bluetooth, accelerometer, sound, and camera [6, 18–20]. The main advantage of GPS is its high positioning accuracy and good coverage in the open outdoor environment, while Wi-Fi is mostly available in indoor and urban environments. Despite the fact that these two technologies are complementary, they are rarely combined for inference of stay points [18].

There are three major approaches for discovering stay points: geometric, fingerprinting, and inertial sensor-based approach.

The geometric approach relies on distances among physical locations. It can be further divided into clustering-based [9, 21], differential-based [12, 13], and probabilistic methods [8, 11]. The geometric stay points are mostly determined with GPS, but positioning based on Wi-Fi can be used also [15]. This Wi-Fi-based positioning requires a database, such as Skyhook [22], PlaceEngine [23], or Google location API [24], in which the fingerprints are linked with the geometric locations.

The fingerprinting approach relies on finding similar patterns. A new pattern is compared to previously learned patterns by, for example, comparing the number of beacons the fingerprints have in common [5] or calculating a similarity parameter, such as Tanimoto [25] or Sørensen’s similarity index [26]. However, the learned patterns lack a reference to physical location, but this can be obtained if needed, for example, with GPS [19] or from any of the abovementioned databases.

The inertial sensor-based approach most commonly relies on accelerometer data. In this approach, the stay points are determined as places without movements. Inertial sensors consume less energy than the other two technologies, but they can only be used as a complementary approach because they lack the physical location. However, the location can be determined in the same way as in the fingerprinting approach [26].

In conclusion, the fingerprinting approach proved to be better than the clustering-based and differential-based geometric approaches [6]. Again, both approaches are complementary but were rarely used together [18]. Moreover, a room-level accuracy of determined stay points was promised if the fingerprinting approach was combined with the inertial sensor-based approach [26]. However, it is not clear if
the room-level accuracy can be achieved only with the fingerprinting approach without prior field measurements.

Aggregation of stay points into stay regions was studied only with stay points determined with the geometric approach. For this purpose, clustering was used, for example, K-means [9], density-based [14], and grid-based clustering [12, 19]. To the best of our knowledge, the aggregation was not explored with stay points determined with the fingerprinting approach.

2.2. Context Models and Context Inference of Stay Regions. There are mainly two types of context models in the domain of context awareness: (i) context models that refer to places, such as home, a restaurant, or a shop and (ii) context models that describe user activities, such as running or walking. Both can be combined to infer activities at various places, but we limit our scope to (i). Furthermore, user context models vary from very general ones, describing contexts like home, work, abroad, and in transit [16, 27], to very narrow ones, that is, domain-specific, such as for museums [28] or tour guides [29]. The most commonly used models distinguish several rather specific user daily contexts, such as visiting a friend, parking, food and drink, shopping, movie and shows, sport and exercise, tourism and amusement, travel and transport, outdoors, nightlife, and colleges and universities [10, 12, 13, 30]. None of the used context models tried to include all contexts that typically describe the everyday life of a user, nor did they organise them in a hierarchical structure.

The context of stay points can be automatically determined in an unsupervised or in a supervised manner. Contrary to the unsupervised approach, the supervised approach initially requires labelled data, which can be obtained manually by entering a label [16, 31], by using the check-in mechanism [30], or semiautomatically by confirming suggested labels [12, 20]. Automatic context inference typically includes rule-based approach, data mining techniques, and ontologies. These techniques were in practice combined with the following approaches for automatic context inference: location-based lookup [10, 13, 30], point of interest (POI) database and POI-activity matrix [7], popularity of nearby POIs [32], and temporal and image features extraction [30]. While the performance of location-based lookup was rather poor due to inaccuracy of positioning and density of POIs, the last two approaches that used social network data performed better [30]. Therefore, the current trend in the field of context inference favours these automatic and supervised approaches. However, there is a lack of further research into automatic and unsupervised context inference.

3. Methods

To address the identified challenges, we propose a hierarchical context model and a method for processing mobile sensor data in an automatic and unsupervised manner. The outline of our approach is as follows. The method first finds user stay points; then it aggregates them into stay regions and finally infers the context of determined stay regions. The details of each step are described below in the corresponding sections.

We evaluated the performance of the proposed method against user context annotations and compared its performance to the performance of SensLoc [6]. Both the sensor data and the annotations were collected with the smartphone application during two data collection campaigns.

3.1. Context Model. We formed a simple hierarchical taxonomy that is general enough to describe most of the possible user contexts and detailed enough to enable discrimination between various contexts but still not too interruptive for the users. It consists of four general contexts: work, free time, home, and transit. The first two contexts are further divided into more detailed subcontexts, as presented in Figure 1. Some of the subcontexts were divided even further.

3.2. Smartphone Application. We used the Android-based Funf open sensing framework [33] to build the application for collecting mobile sensor data and user context annotations. This framework supports periodic triggering of over 30 different sensor probes, but for our present work only location and visible Wi-Fi access points (APs) were used. The proposed hierarchical context model, presented in the previous paragraph, was integrated in the application. We adjusted the graphical user interface (GUI) of the application, as presented in Figure 2, so that there was one tab for manually choosing user context and another tab for editing all previously chosen contexts.

3.3. Data Collection Campaigns. For the data collection campaign, we activated all observed sensors and defined their sampling intervals and durations, where applicable. For location, the sampling interval was one minute, and duration, that is, maximal time the probe will scan before sending the best value, was five seconds; for Wi-Fi, the sampling interval was two minutes. The settings were the same for all users, and they could neither see nor change them. We recruited six middle-aged mobile users of various lifestyles. All were office workers, but five of them performed their work also outside the office premises. Two of them spent most of their time at home and at work, while others had more diverse lifestyles. Each of them had a different smartphone: Samsung Galaxy Ace, Samsung Galaxy Ace 2, Samsung Galaxy S Plus, Samsung Galaxy S II, Samsung Galaxy S III, and HTC One V. They were asked to use their phones as usual and to activate Wi-Fi and GPS modules on their smartphones. Each user participated in the campaign for eight days. The users were instructed to annotate their context whenever they carried their phone with them. Every time their context changed, they chose the appropriate context from the hierarchical context model, as presented in Figure 2. The collected data were uploaded to the dedicated server and processed offline. Through a simple web interface, the participants and researchers could see who was participating actively, track the amount of collected data, and observe trajectories on a map. With the map, all annotations were corrected, if necessary, to reflect the real contexts.
Additionally, a user followed a scripted tour within one apartment. She visited all rooms while collecting data with the smartphone application as described above.

3.4. The Proposed Method. First, we define some terms, such as GPS and Wi-Fi trajectory, GPS and Wi-Fi stay point, and stay region.

3.4.1. Definitions. A GPS trajectory is a sequence of time-stamped points \( p_{gps}(x_i, y_i, t_i) \), where \( t_i \) is a timestamp, \( p_{gps} \cdot t < p_{gps}_{i+1} \cdot t \), \( i = 0, 1, \ldots, k \), and \( (x_i, y_i) \) a pair of longitude and latitude values.

A Wi-Fi trajectory is a sequence of time-stamped points \( p_{wifi}(b, t_j) \), where \( t_j \) is a timestamp, \( p_{wifi} \cdot t < p_{wifi}_{i+1} \cdot t \), \( i = 0, 1, \ldots, k \), and \( b \) a list of beacon pairs (mac, rss), where \( j = 0, 1, \ldots, n \) and \( n \) is the number of visible APs at time \( t \).

A GPS stay point \( s_{gps} \) is a representation of a set of consecutive \( p_{gps} \), where \( i \) is between \( u \) and \( v \), distance \( (p_{gps_u}, p_{gps_v}) \) is smaller than a distance threshold \( D_{thr} \). \( p_{gps} \cdot t \) is bigger than a predefined time threshold \( T_{thr} \), and distance \( (p_{gps_u}, p_{gps_v}) > D_{thr} \). \( S_{gps} \) is represented as \( (x, y, t_e, t_d) \), where \( x \) and \( y \) stand for average longitude and latitude of the set, \( t_e \) is the entrance time equal to \( p_{gps_u} \cdot t \), and \( t_d \) is the departure time from stay point equal to \( p_{gps_v} \cdot t \).

A Wi-Fi stay point \( s_{wifi} \) is a representation of a set of consecutive \( p_{wifi} \), where \( i \) is between \( u \) and \( v \), \( p_{wifi} \cdot t - p_{wifi_u} \cdot t > T_{thr} \), similarity \( (b_u, b) \) is above threshold \( T_{sim} \), and similarity \( (b_u, b_{u+1}) < T_{sim} \). \( S_{wifi} \) is represented as \( (b, t_e, t_d) \), where \( b \) is a set of representatives of the stay point, \( t_e \) is the entrance time equal to \( p_{wifi_u} \cdot t \), and \( t_d \) is departure time from stay point equal to \( p_{wifi_v} \cdot t \).

A stay region is a cluster of several GPS and Wi-Fi stay points that are within 100 m or have a similar Wi-Fi fingerprint. They can be represented with average \( x \) and \( y \) coordinates or a common set of Wi-Fi beacons.

3.4.2. Finding Stay Points with Fingerprints. To find Wi-Fi stay points, we implemented the method from [6], found the optimal set of parameter values for our dataset, suggested five improvements, and evaluated them. The method from [6] uses sliding windows \( w_{slid} \) for a more accurate detection of stay points and omits locally administered APs, that is, local networks of laptops or smartphones, which are typically not at fixed locations. It uses a scan window \( w \), which helps in tolerating an unstable radio environment. Contrary to [6], we defined a scan window with a time value and not with the number of scans. The essence of the method is the Tanimoto coefficient [25], defined in (1), which measures similarity between two fingerprints \( F_1 \) and \( F_2 \):

\[
T(F_1, F_2) = \frac{F_1 \ast F_2}{(\|F_1\|^2 + (\|F_2\|^2) - F_1 \ast F_2).}
\]

In (1), \( F_1 \) and \( F_2 \) are vectors of compared scan windows, \( F_1 \ast F_2 \) a scalar product between both vectors, and \( |F_i| \) the Euclidean norm of the vector \( F_i \). As the scan window is used, each AP is represented with an average received signal strength (RSS) value and a response rate, that is, the ratio between detection count and number of scans in the scan window. In order to compute the Tanimoto coefficient, each vector is completed with APs that can be found in the other vector and are missing in the observed one. Such APs are assigned an average RSS value of zero.
Similar scan windows are those whose Tanimoto coefficient is above threshold $t_{\text{sim}}$. This notion of similarity is used to detect entrances and departures and to recognise familiar stay points. An entrance is detected when the number of consecutive similar scan windows exceeds the threshold $c_{\text{max}}$. For departures, the current scan window is compared to the reference fingerprint of the observed stay point, which consists of representative beacons, that is, APs with response rates above $r_{\text{rep}}$. A certainty value $c$, ranging from 0 to $c_{\text{max}}$, is also used. When a scan window is different from the reference fingerprint of the observed stay point, $c$ decreases; otherwise it increases. When $c$ reaches 0, a departure is detected. Familiar stay points are recognised by comparing representative beacons of entered stay point with the reference fingerprints of all known stay points. If comparison yields a Tanimoto coefficient above $t_{\text{sim}}$, a familiar stay point is recognised.

To find the optimal set of parameter values, we tested the performance of all possible combinations of $w$, $w_{\text{sl}}$, $c_{\text{max}}$, $t_{\text{sim}}$, and $r_{\text{rep}}$ and compared them with two SensLoc settings: both with adaptive $r_{\text{rep}}$ and $w$, but the base value of $w$ was 30 s (SensLoc 30 s) in one and 60 s (SensLoc 60 s) in the other. We tested the following window sizes: 15, 30, 45, 60, and 120 s. For the sliding window, we evaluated nonoverlapping windows as well as different values for $w_{\text{sl}}$: 15, 30, and 60 s. For $c_{\text{max}}$, we evaluated values 3 and 4, and for $t_{\text{sim}}$ and $r_{\text{rep}}$, the values from 0.5 to 0.8. We tested also the adaptive $r_{\text{rep}}$ and $w$. We sought a combination that gave a reasonably low number of stay points (as users typically visit a limited number of different places), not too many transitions among them and most time spent at stay points [34]. At the same time, we sought the highest values of accuracy, precision, and recall in the recognition of the static mode versus the moving mode. To obtain the accuracy, precision, and recall, we used the labels of the static user contexts, for example, home, at concert, dining, and visiting, as indicators of stay points, that is, the ground truth. For calculation, we considered all available time samples, and for each sample we assessed the correctness of the stay point determination.

We also proposed and evaluated five possible improvements.

1. We changed the units of RSS values. We tested the effect of values in mW instead of in dBm and thus favoured stronger beacons.

2. We improved the learning scheme of reference fingerprints. While SensLoc uses accumulated beacon statistics during the current stay for a better detection of departures, we tested an incremental learning of new representative beacons that was used not only for detection of departures, but also for updating the reference fingerprint of a stay point. By doing this, the reference fingerprint of each stay point continuously adapted to the changes of the Wi-Fi environment.

3. We used only unique MACs. Although MAC should be a unique identifier of AP, in reality several devices bear the same identifier. Therefore, we considered only unique MACs that were identified in our campaign.

4. We limited the number of considered beacons in the reference fingerprints. Due to the unstable radio environment, the set of beacons at the same location differs at various times. To improve the robustness, we calculated the Tanimoto coefficient only with the beacons contained in the current scan window.

5. We reduced the number of transitions between the same stay points. Due to the unstable radio environment, the transitions between the same stay points can often be falsely inferred. Therefore, we excluded the departures and returns to the same point if they occurred within a couple of minutes.

To evaluate all five improvements, we tested the performance of each improvement and of their combinations and compared it to the best performing set of parameter values without any improvements. The performance was again evaluated in terms of number of stay points and transitions among them, time spent at stay points, and accuracy, precision, and recall of finding stay points.

We also evaluated the effect of all combinations of parameters and of five proposed improvements on the room-level accuracy. For this test, we used a scripted dataset with our application while visiting all rooms of one apartment at specified times. The results of our method were compared with the diary of the LifeMap application [18].

Finally, the stay points were determined with settings and improvements that demonstrated the best overall performance. These stay points were assigned physical locations.
with GPS or Wi-Fi using the location-sensor data from the Google location API [24]. All stay points for which such location was not available were assigned approximate locations based on similarity of fingerprints. For a stay point $A$ without a location, another similar stay point $B$ with location information and with highest similarity value according to (1) was searched for. If it was found, the stay point $A$ was assigned an approximate location. If not, its location remained unknown.

3.4.3. Finding Stay Points with Geometric Approach. To find stay points from the GPS trajectories, we deployed the differential-based clustering. We used the threshold $D_{\text{thr}} = 50$ m as a maximum distance that a user can cover within the same stay point [19], $T_{\text{max}} = 10$ min as a maximum time difference between two consecutive measurements [19], and $T_{\text{thr}} = 3$ min as a minimum time that the user must spend in the same place to be considered a stay point.

3.4.4. Aggregation of Stay Points into Stay Regions. For aggregation of stay points with the same semantic meaning to a user into meaningful stay regions of adoptable size, we tested different combinations of the widely applied physical proximity as well as three other parameters: (i) fingerprint similarity, (ii) term frequency-inverse document frequency (TF-IDF), and (iii) SSID.

(i) The fingerprint similarity in the fingerprinting approach resembles the physical proximity in the geometric approach. If two fingerprints $F_1$ and $F_2$ of two stay points have some similarity ($T(F_1, F_2) > 0.1$), this means that these stay points are somewhat close together, because the range of Wi-Fi technology is rather short.

(ii) TF-IDF is used for measuring the importance of a specific POI category to a stay point in the collection of all POI categories and stay points. For this, we used 38 POI categories from the database of the local mobile operator. We sought all POI categories within 100 m of a stay point. A weight $w$ was assigned to each stay point $s$ in the collection of stay points of each user $u$ for each POI category $j$ according to (2):

$$w_{u,s,j} = \frac{n_{u,s,j}}{N_{u,j}} \log \left( \frac{|R_u|}{|M_{u,j}|} \right).$$  \hfill (2)

In the equation, $n_{u,s,j}$ equaled the number of POIs of category $j$ at a stay point $s$ for user $u$, $N_{u,j}$ equaled the total number of POIs at a stay point $s$ for user $u$, $|R_u|$ equaled the total number of stay points for user $u$, and $|M_{u,j}|$ equaled the number of stay points containing category $j$ for user $u$. For aggregation, we calculated the Euclidean distance between vectors of weights of two stay points $A$ and $B$ that are physically close. For this, each distance of weights was normalised with weights of the stay point $A$. If the resulting distance was below 2, we assumed stay points $A$ and $B$ had the same weights.

(iii) Stay regions, such as a shopping mall, company premises, or concert hall, are covered by several APs to provide coverage of the entire space. These APs typically belong to the same SSID. Therefore, the stay points with APs of the same SSID tend to belong to the same stay region. For each stay point, we extracted the most common SSID. The stay points $A$ and $B$ were aggregated if the most common SSID of the stay point $A$ was present in more than 60% of Wi-Fi scans of the stay point $B$.

We designed and evaluated six aggregation schemes. The aggregation scheme 1 used only the physical proximity of stay points with location information. The aggregation scheme 2 used the physical proximity of stay points with location information and stay points with approximate location information. The aggregation scheme 3 used only the fingerprint similarity. The aggregation scheme 4 aggregated stay points based only on SSID. The aggregation scheme 5 aggregated the stay points with location information firstly on the physical proximity. Then, all of the other stay points were added based on the fingerprint similarity. The aggregation scheme 6 firstly aggregated the stay points with location information based on proximity and TF-IDF. Then, all of the other stay points were added based on the fingerprint similarity.

For evaluation, we first mapped the locations of the user-annotated static subcontexts to see how the locations were grouped together. From this, we manually extracted meaningful stay regions for every user, that is, the groups of locations with the same semantic meaning, such as her home building, company premises, a shopping mall, and a fitness club nearby. These regions were named the “user-annotated stay regions.” For each of them, the containing stay points were extracted from the user trajectories. Then, we ran all aggregation schemes, but only on the stay points where the users stayed at least one minute. Finally, we manually ranked the schemes according to how well their stay regions matched the user-annotated stay regions. Then, we used the best two schemes to aggregate three sets of stay points separately: the first set was determined with our method and the other two with SensLoc of two different settings. For each set and aggregation scheme, we compared the number of correctly determined stay regions, merged stay regions, divided stay regions, missed stay regions, falsely determined stay regions, and interesting stay regions, as well as precision and recall of stay region determination, as defined in (3):

$$\text{precision} = \frac{\#\text{correct} + \#\text{interesting}}{\#\text{correct} + \#\text{interesting} + \#\text{merged} + \#\text{divided} + \#\text{false}},$$

$$\text{recall} = \frac{\#\text{correct}}{\#\text{correct} + \#\text{merged} + \#\text{divided} + \#\text{missed}}. \hfill (3)$$

Similarly as in [5, 6], we only considered the regions where a user stayed for more than five minutes. For all users, we also evaluated how revisits to familiar stay points were recognized for two most visited stay regions: home and work. At each
Table 1: An example of five different settings for stay point determination chosen from all possible combinations of window size ($w$), sliding window ($w_{sl}$), certainty threshold ($c_{max}$), similarity threshold ($r_{sim}$), and response rate ($r_{rep}$) and compared to the original settings of the SensLoc method with the base value of $w = 30$ s (SensLoc 30 s) [6] and SensLoc with the base value of $w = 60$ s (SensLoc 60 s). For $c_{max} = 3$ and different values of parameters $w$, $w_{sl}$, $r_{sim}$, and $r_{rep}$, we show the number of determined different Wi-Fi stay points ($s_{wifi}$), number of transitions among them ($t$), time spent at stay points ($t$), and accuracy ($A$), precision ($P$), and recall ($R$) of recognition of the static mode versus the moving mode. $S_{wifi}$, tran, and $t$ are sums, while $A$, $P$, and $R$ are average values for six users over eight days. Bold Settings 2 yielded the best overall results.

| Settings | $w$ [s] | $w_{sl}$ [s] | $r_{sim}$ | $r_{rep}$ | $s_{wifi}$ | $t$ [h:min] | $A$ [%] | $P$ [%] | $R$ [%] |
|----------|---------|-------------|-----------|-----------|-----------|----------------|---------|---------|---------|
| SensLoc 30 s | ad. | 15 | 0.7 | ad. | 944 | 3,292 | 283:22 | 81.0 | 98.8 | 80.0 |
| SensLoc 60 s | ad. | 15 | 0.7 | ad. | 1,164 | 5,882 | 474:58 | 77.1 | 99.3 | 76.0 |
| Settings 1 | 60 | 15 | 0.7 | 0.7 | 967 | 5,586 | 444:06 | 76.6 | 99.2 | 75.4 |
| Settings 2 | 120 | 15 | 0.6 | 0.6 | 408 | 2,075 | 573:15 | 91.1 | 99.3 | 90.9 |
| Settings 3 | 120 | 15 | 0.7 | 0.7 | 766 | 4,159 | 524:28 | 83.3 | 99.5 | 82.4 |
| Settings 4 | 120 | 15 | 0.8 | 0.8 | 1,418 | 7,603 | 424:36 | 68.0 | 99.5 | 65.7 |
| Settings 5 | 120 | 30 | 0.7 | 0.7 | 581 | 2,619 | 516:54 | 81.7 | 99.5 | 80.6 |

subsequent visit, we assessed the number of newly recognized stay points determined with our method and with SensLoc of both settings. Finally, we evaluated the accuracy of detected entrance and departure times to and from stay regions. We measured the time offset between the annotated ground truth and detected entrance and departure times by our method and by SensLoc of both settings.

3.4.5. Context Reasoning. We designed a method for automatic and unsupervised assignment of four general user contexts to the identified stay regions, evaluated its performance, and proposed an approach for in-depth reasoning about stay points within each general context. Inference of user context was based on a spatiotemporal clustering. It extracted work as the stay region denoting the office building where a user spent most of her time during typical working hours between 8:00 and 17:00. Home was inferred as the stay region where a user spent most of her time in the evening, during the night, and at weekends, while transit was obtained as a moving mode from inference of stay points. All other stay regions were annotated with the context free time.

For evaluation, we compared the results of our method to the user-annotated data. All user-annotated contexts that represented a moving mode, such as cycling or walking, were treated as the context transit. Each of the four contexts was evaluated with accuracy, precision, and recall for the best two aggregation schemes. To do this, we considered all available time samples, and for each sample we assessed the correctness of automatic context inference. For easier comparison with the related literature, we also used accuracy, precision, and recall to evaluate if the context of each stay point was inferred correctly. Although the context was assigned to the stay regions, we did not evaluate their contexts because the false aggregation into regions and false context assignment would lead to the accumulated error.

For the in-depth reasoning about stay points within each context, we plotted the temporal patterns of the stay points’ visits for each user and examined their characteristics. By doing this, we tried to extract various subcontexts of stay points, such as a living area, a dining area, and a sleeping area, all within the context home. Because we did not have such detailed user annotations, we unfortunately were not able to evaluate the proposed approach.

4. Results

4.1. Finding Stay Points. For the inference of stay points with the original SensLoc method, the optimal set of parameter values for our dataset turned out to be the following: $w = 120$ s, $w_{sl} = 15$ s, $r_{sim} = 0.6$, $c_{max} = 3$, and $r_{rep} = 0.6$. It yielded the highest accuracy, precision, and recall in the recognition of the static mode versus the moving mode, and the time spent at stay points approached the user-annotated 690 h 41 min, that is, 91.8% of the time when the application was running. At the same time, these values gave a reasonably low number of stay points and not too many transitions among them, as highlighted in Table 1. For $w$ shorter than 60 s, the time spent at stay points was too small, while bigger $r_{sim}$, $c_{max}$, and $r_{rep}$ resulted in too many inferred stay points, too frequent transitions among them, and too little time spent at them. Also, with the adaptive $w$ and $r_{rep}$ of SensLoc, all six observed parameters were worse, and therefore all versions that considered it were omitted from the table. In comparison, as shown in the first two rows of Table 1, the original set of values of SensLoc method with the base value of $w = 30$ s (SensLoc 30 s) and SensLoc with the base value of $w = 60$ s (SensLoc 60 s) resulted in an enormous amount of various stay points and transitions between them. Both versions of SensLoc had at least 10% lower values of precision and recall and determined at least 14.2% less time spent at stay points compared to Settings 2.

The accuracy, precision, and recall of inferred stay points were very high for all six users except for User 1. For the highlighted combination in Table 1, the standard deviation of accuracy was 4.6%, of precision 0.4%, and of recall 5.3%. For SensLoc 60 s, the standard deviation of accuracy was 6.8%, of precision 0.5%, and of recall 7.5%, while they were higher for SensLoc 30 s.

In the next step, we evaluated the impact of the five proposed improvements and their combinations on the performance. As shown in Table 2, a combination of improvements
Table 2: The performance of the method with the proposed improvements (Imp) compared to the performance without improvements. Only some of the improvements and their combinations are shown. The improvements were (1) mW instead of dBm; (2) incremental update of representative beacons; (4) selection of used beacons for calculation of Tanimoto; and (5) interpolation of transitions between the same stay points. Bold combination gave the best overall results. The performance was evaluated as the number of determined different Wi-Fi stay points ($s_{wifi}$), number of determined different GPS stay points ($s_{gps}$), number of transitions (tran), time spent at stay points ($t$), and accuracy ($A$), precision ($P$), and recall ($R$) of recognition of the static mode versus the moving mode. $s_{wifi}$, $s_{gps}$, tran, and $t$ were sums, while $A$, $P$, and $R$ were average values for six users over eight days.

|                  | $s_{wifi}$ | $s_{gps}$ | tran   | $t$ [h:min] | $A$ [%] | $P$ [%] | $R$ [%] |
|------------------|------------|-----------|--------|-------------|---------|---------|---------|
| Without any Imp  | 408        | 0         | 2,075  | 573:15      | 91.1    | 99.3    | 90.9    |
| Imp 1            | 258        | 0         | 1,063  | 593:28      | 98.3    | 94.9    | 93.8    |
| Imp 2            | 341        | 0         | 1,638  | 581:38      | 98.3    | 92.2    | 91.4    |
| Imp 4            | 199        | 0         | 925    | 606:21      | 97.8    | 97.1    | 95.3    |
| Imp 1, 2, and 4  | 254        | 0         | 718    | 631:37      | 96.4    | 99.1    | 97.0    |
| Imp 1, 2, 4, and 5| 254        | 0         | 535    | 635:47      | 96.6    | 99.1    | 97.2    |
| Imp 1, 2, 4, and 5 + GPS | 254 | 2 | 537 | 636:06 | 96.6 | 99.1  | 97.2 |

Table 3: The comparison of correctly aggregated (Cor), missed (Mis), merged (Mer), divided (Div), interesting (Int), and false (Fal) stay regions for both aggregation schemes and three observed methods for finding stay points: SensLoc 30 s, SensLoc 60 s, and our method. The precision ($P$) and recall ($R$) of stay region determination are shown as well. Only stay regions where a user stayed for more than five minutes were considered. Bold combination gave the best overall results.

| Method                  | Cor | Mis | Mer | Div | Int | Fal | $P$ [%] | $R$ [%] |
|-------------------------|-----|-----|-----|-----|-----|-----|---------|---------|
| SensLoc 30 s—aggregation 3 | 56  | 18  | 3   | 10  | 0   | 0   | 81.2    | 64.4    |
| SensLoc 30 s—aggregation 5 | 48  | 18  | 8   | 13  | 0   | 0   | 69.6    | 55.2    |
| SensLoc 60 s—aggregation 3 | 59  | 11  | 4   | 13  | 0   | 0   | 77.6    | 67.8    |
| SensLoc 60 s—aggregation 5 | 52  | 11  | 6   | 18  | 0   | 0   | 68.4    | 59.8    |
| Our method—aggregation 3  | 66  | 0   | 9   | 12  | 0   | 0   | 75.9    | 75.9    |
| Our method—aggregation 5  | 64  | 0   | 7   | 16  | 0   | 0   | 73.6    | 73.6    |

1, 2, 4, and 5 resulted in the amount of time spent at stay points that even better resembled the user-annotated time, in the highest recall and in the lowest number of transitions. Improvement of considering only unique MACs (Imp 3) did not affect the performance, so we omitted it from the table. With GPS, only two additional stay points were recognised, adding less than 20 min to the overall stay time and not affecting accuracy, recall, and precision.

The proposed improvements levelled up accuracy, precision, and recall so that they were high for all users. They increased the accuracy and recall compared to Settings 2 from Table 1 for more than 5% and determined an additional 9.1% of time spent at stay points. For the highlighted combination in Table 2, the standard deviation of accuracy was 1.2%, of precision 0.5%, and of recall 1.1%.

The highlighted combination of settings and improvements demonstrated also the best performance at the room level. Of all settings and improvements, it reflected the reality the most with respect to inferred rooms, transitions between them, and time spent in them. As with LifeMap, two out of four rooms were identified.

4.2 Aggregation of Stay Points. In this section we present how well the stay regions determined with different aggregation schemes reflected the “user-annotated stay regions,” as defined in Section 3.4.4. Table 3 shows the results of the two most promising aggregation schemes. The accuracy of aggregation is compared for three different methods for finding stay points: our method, SensLoc 30 s, and SensLoc 60 s. The aggregation scheme 3 was based only on the fingerprint similarity, and the aggregation scheme 5 was based on the physical proximity and the fingerprint similarity. A combination of our method with the aggregation scheme 3, highlighted in Table 3, yielded the best overall performance. It was better than the combination of our method and the aggregation scheme 5 in precision, recall, and number of correctly determined stay regions. The aggregation scheme 3 outperformed the aggregation scheme 5 also on the other two observed methods for finding stay points. The highlighted combination, compared to the results of the other two observed methods for finding stay points, had at least 8.1% higher recall and 7 correctly determined stay regions more without missing any user-annotated stay region.

Other aggregation schemes that we tested did not perform well. The aggregation schemes based solely on the physical location or on the physical location with TF-IDF suffered from two facts. Firstly, not all stay points had the location information, for example, because it was not provided by the Google API. And secondly, the location information was not accurate due to positioning error. However, the aggregation scheme 4 based on SSID worked better, but it aggregated also the stay points that were far apart.
An evaluation of revisits to two most visited stay regions revealed very different behaviour between the three observed methods for finding stay points. While our method normally determined new stay points during the first visit to a stay region and then accurately recognized revisits, SensLoc 30 s and SensLoc 60 s determined several (up to 17) new stay points during each revisit.

Figure 3 compares the time offset between the detected departure (a) and entrance (b) times and the ground truth. The time offset for departure times for all three methods was negative for at least 78% of the visits. With our method, the time offset was within −120 to 0 for 71% of the visits which is for 26% better than with SensLoc 30 s and for 9% better than with SensLoc 60 s. With our method, the inferred departure times were more than 120 s earlier than the actual departure times in 13% of the visits. This is for 20% less than with SensLoc 30 s and for 15% less than with SensLoc 60 s. The time offset of entrance times for our method was within 0 to 120 seconds for about 62% of the visits, while the percentage of this time offset was for at least 11% lower with both versions of SensLoc.

### 4.3. Context Inference

For 90.1% of all the minutes when the application was running, our method was able to automatically and in an unsupervised manner determine user context. The results presented in Table 4 show that, with our logic, the user context was inferred with high accuracy, precision, and recall. The contexts *home* and *work* were inferred very correctly, while the contexts *transit* and *free time* were inferred noticeably less correctly. The average inference accuracy in time over all users was 97.5% with a standard deviation of 1.5%.

The accuracy, precision, and recall of the inferred contexts were different from user to user. For the context *home*, the standard deviations were the smallest; they were higher for *work* and highest for *transit* and *free time*.

Figure 4 presents the correctness of context inference per stay points in comparison with context inference in time, already presented in Table 4. When evaluating context per stay points, the context *home* was, as before, inferred very accurately with precision of 90.5% and recall of 95.0%, whereas the results for the context *work* were noticeably worse with precision of 83.3% and recall of 53.6%. The overall accuracy of stay point labelling was 89.7%.

For three users, we observed that the context *home* consisted of two to six stay points. Figure 5 shows an example of four different stay points within the context *home* for User 3. From the time pattern of stay point 1, we can reason that it represents a sleeping area. Similarly, we can reason that stay point 2 represents a room used during the day.

### 5. Discussion

With the proposed method, we were able to trustworthily find user stay points, aggregate them into stay regions, and infer...
their contexts. Our method differs from the related work in several ways. Firstly, it is entirely automatic and nonintrusive, because it does not require user input at any stage. Secondly, the method is simple and universal because it uses only Wi-Fi and GPS, but no other smartphone sensors, such as accelerometer [16, 18], sound [16], and camera [30]. This enables its wide deployment, for example, on laptops, simpler mobile phones, and M2M devices. Thirdly, the proposed method relies mostly on Wi-Fi fingerprints, while most of the related work used GPS for stay points inference [7–14]. Wi-Fi is preferable because it spends significantly less energy [6], works instantly, is enabled by the users more often than GPS, and covers most urban and indoor environments. Due to all of the characteristics, the method can potentially be deployed in various fields, such as in adjustment of multimedia applications and services, in recommender systems, kids and elderly care, studies on user mobility and their context-dependent behaviour, and in fleet management.

5.1. Finding Stay Points. We improved the method for inference of stay points from [6] by finding a better set of parameter values for our dataset and by proposing and evaluating five improvements. The chosen set of parameter values worked better on our dataset because it optimised the determination of moving versus static mode, indicated by high values of accuracy, precision, and recall parameters. It also approached the user-annotated time spent at stay points and decreased the number of stay points and transitions between them. One possible reason why the optimal set of parameter values from [6] with the base value of \( w = 30 \) s did not work on our dataset could be a different sampling of the Wi-Fi fingerprints. While Kim et al. used a fixed number of scans per scan window, our method dealt with irregularly sampled scans. Therefore, we increased the base value of \( w \) to 60 s which improved the results of SensLoc, but they remained worse than the results of our method. However, for both settings of SensLoc a large amount of different stay points was likely a consequence of a higher similarity threshold which split very similar stay points. A combination of our four proposed improvements further improved the performance of the original SensLoc method. The largest impact was caused by improvement 4 that limited the number of considered beacons in the reference fingerprints when calculating the Tanimoto coefficient. This compensated for the unstable radio environment but potentially at the cost of not inferring all the moves. Without this improvement, the beacons from the current scan window were compared to a bigger set of beacons from the reference fingerprint of a stay point, which resulted in a lower value of the Tanimoto coefficient. This was interpreted as a move from the stay point, although a lower number of beacons could have been just a consequence of a change in the radio environment. However, if only the beacons contained in the current scan window were considered for calculation, this favoured the stay mode and reduced the number of transitions. The resulting increase in accuracy, precision, and recall values indicated that this approach was reasonable because its outcomes resembled the user-annotated data well. After adding improvements 1 and 2, the resemblance with the user-annotated data was even greater. Improvement 1 was a simple change of units, which favoured the contribution of stronger, that is, closer, beacons in the calculation of the Tanimoto coefficient. This was interpreted as a move from the stay point, which resulted in a lower value of the Tanimoto coefficient. Improvement 2 introduced accumulated statistics of the representative beacons, averaging not only current but also all previous visits of a particular stay point. This approach together with improvement 4 was more robust in the ever-changing radio environment, while it maintained the ability of adapting to sudden changes. Improvement 5 omitted all transitions where a departure from a stay point was followed by a return to the same stay point soon thereafter. Some such transitions indeed occurred, but we did not consider them relevant for our work. With GPS, we identified only two additional stay points because the GPS data were rarely available. This was the consequence of our users spending most of their time in urban and indoor environments, where GPS was often not available.
With our set of parameter values and improvements, we ran into a risk of losing some information about the actual user trajectories due to merging of the adjacent stay points and eliminating transitions between them. However, our short test on the scripted trajectory showed that with our set of parameter values and improvements the results most closely approximated the real number of rooms, the time spent in them, and transitions between them. The results were as good as those of the LifeMap application [18] which, besides the Wi-Fi fingerprints, relied also on the accelerometer data. The LifeMap application promised room-level accuracy but did not achieve it in our case. Some of the most likely reasons for not achieving room-level accuracy with our method were its dependence on the number and placement of APs in the environment, the layout of the rooms, and the construction materials used. The room-level accuracy was not achieved with other methods [6]; therefore we did not compare our method to them. For example, PlaceSense [5] merged the nearby locales indoors, while, outdoors, a single locale covered a large area, such as a long street or a park. Similarly, the geometric approaches, such as in [15], did not distinguish different locales within the same building or close-by, because they considered only two spatial dimensions which were not sufficiently accurate.

The methods for finding stay points are usually evaluated as the number of correct, divided, merged, false, interesting, and missed stay points [5, 6]. Such evaluation was not possible with annotations used in our dataset, because the users did not annotate the detailed locales, such as rooms, but only larger locales that contained several stay points, such as an apartment or a shopping mall. Therefore, we performed such evaluation only on larger locales, that is, stay regions.

When evaluating the method for finding stay points through comparison with the user-annotated data, some caution is needed. Although the user-annotated data is a good indicator of the ground truth, it caused some systematic error in our calculation of the accuracy, precision, and recall, and in the estimation of the time spent at stay points. On the one hand, the user annotations were not entirely correct due to the different understanding of the labels among the users and due to the imprecise timings of the context changes. We minimised this type of error by manually correcting all annotations so that they reflected the real contexts well. On the other hand, our context categories could not have been uniformly assigned to the dynamic or static modes. Some labels of the static user contexts, for example, home, at concert, and dining, were better indicators of the stay points than others, for example, shopping. Similarly, the dynamic user contexts, such as waiting for a bus in the context transit, often included some elements of the static mode. Therefore, the total user-annotated time spent at the stay points was never reached, and the values of accuracy, precision, and recall were never 100%.

5.2 Aggregation of Stay Points into Stay Regions. We proposed and evaluated six approaches for the aggregation of fingerprint and geometric stay points into stay regions. While the aggregation of the geometric stay points has been well explored [8, 9, 11, 14, 15, 21], we are not aware of any solutions for the aggregation of the fingerprint stay points and for joining them with the geometric ones.

Among the proposed aggregation schemes, the results of aggregation scheme 3, which was based on the fingerprint similarity, resembled the user annotations the most. The number of correctly determined stay regions together with the precision and recall was consistently the highest compared to the second best aggregation scheme 5. The logic of the fingerprint similarity used in the aggregation scheme 3 resembles the logic of the physical proximity, but the fingerprints are more sensitive to small changes of the user location and are also available for the majority of stay points. However, the scheme was limited, as it could not aggregate the GPS stay points unless they had the Wi-Fi fingerprints. Moreover, this scheme failed to aggregate the stay points with different fingerprints even though they belonged to the same stay region. Therefore, in the future, the use of the location information and SSID could be considered to account for both of these issues. The aggregation scheme 5, which was based on the physical proximity and on the fingerprint similarity, performed similarly well as the aggregation scheme 3. However, all aggregation schemes that were based on the physical proximity did not yield such good results because some stay points lacked the location information or because this information was inaccurate. The latter was problematic due to the urban environment, where the density of stay points and stay regions was high. The aggregation scheme 4, which was based on SSID, successfully aggregated the stay points within a large shopping mall, which offered its own WLAN. However, it also aggregated the stay points of gas stations of the same provider, even though they were located in different cities. A similar issue could occur also with the stay points that included the municipality WLAN in their fingerprints. With the further development of our context inference method, we foresee two possible enhancements of the current work: crowdsensing [30] and the use of additional sensors [6, 19, 20, 30, 35]. Thus, we could obtain more characteristics of each stay point, such as the temporal patterns of visits, the user activity, light, and sound, and use them for the aggregation.

After selecting the best aggregation scheme, we used the matching between the automatically determined and the user-annotated stay regions as an additional performance indicator of our method for finding stay points. In comparison to both versions of the original SensLoc method, our method for finding stay points had the highest number of correctly determined stay regions and highest recall and did not miss any regions. Its precision was third highest due to the high number of merged and divided stay regions. The number of merged and divided stay regions was in all cases a result of the incorrect aggregation and not of the incorrect stay point determination. However, the opposite was true for the number of missed stay regions.

The merged stay regions occurred when the stay points of the neighbouring stay regions were similar; that is, the Tanimoto coefficient was higher than 0.1. For example, an open concert area, a coffeehouse terrace, and a bar terrace were merged into one stay region. A possible solution to keep
such stay regions separate would be increasing the threshold value of Tanimoto coefficient.

The divided stay regions were determined when the stay points from the same stay region were not similar, for example, an indoor and outdoor area of the same restaurant or different shops within a big shopping mall. We foresaw that the aggregation based on SSID could be a solution for such cases. However, our current approach (the aggregation scheme 4) did not lower the number of divided stay regions on our dataset, because we were not able to find the right SSID for such aggregation.

All missed stay regions occurred, because the methods for finding stay points were unable to determine all of the relevant stay points. Our method did not miss any stay regions, which means that it was more successful compared to both versions of the original SensLoc method not only in the parameters discussed in Section 5.1, but also in finding stay points that were relevant to the user. For example, the original method failed to determine the locales, such as bakery, stores, shopping malls, and restaurants, where the users spent less time compared to the most visited locales.

The evaluation of revisits showed that, with our method, the number of newly determined stay points within the most visited stay regions approached zero after a few revisits. However, with SensLoc many new stay points within such stay regions were determined at each subsequent revisit. Which method was better in reflecting every detail in the user trajectories cannot be evaluated with our dataset due to too general annotations for such purpose. However, such behaviour of our method could indicate a lower ability to distinguish the stay points close-by or the resistance to the changing radio environment.

The comparison of the time offsets showed that with our method the results more closely approached the ground truth than with both SensLoc settings on our dataset. Our method detected earlier entrances and later departures which further confirms that the chosen values of parameters and the improvements added to the original SensLoc method were reasonable.

5.3. Context Inference. We demonstrated that, with our automatic and unsupervised spatiotemporal reasoning about the user context, high precision, recall, and accuracy can be achieved. The context inference was possible more than 90% of the time, which could be attributed to the use of the widely available Wi-Fi fingerprints and the sliding windows for the inference of stay points. However, the context inference was not possible all of the time because the Wi-Fi fingerprints and GPS were not always available and because the sampling of the sensor data behaved inconsistently.

We introduced a hierarchical context model consisting of all contexts, static and dynamic, that typically describe the everyday life of a user. At the moment, the hierarchical structure enables the correct determination of four general contexts that can be partitioned later into more detailed subcontexts with the use of more logic and more sensor data. The four general contexts of our model resemble [16, 27] the most. We have used several subcontexts suggested by others [10, 12, 13, 30], but we combined them into a broader category named free time. Moreover, our context inference logic resembles [27], which inferred user context based on CID transitions in an unsupervised manner. A somewhat similar, but essentially supervised, approach was used in [30], where logic for personal naming of places was based on time of the day, day of the week, stay duration, three features from image recognition, and familiarity. In [10, 13], the entirely supervised approach of user context inference used nearby POIs, time of the day, the week, and stay durations. We used the last three features in our method for automatic and unsupervised inference of personal names of stay regions.

We evaluated the proposed method for context inference from two perspectives. Firstly, we focused on the percentage of time when the inference worked correctly; and, secondly, we assessed the correctness of context inference for each stay region. We believe that the first option is currently more relevant for the inferred four general contexts because we did not focus on the recognition of subcontexts. Moreover, we could not directly compare the performance of our method for context inference with related work because different context models, sensors, and context reasoning have been used. Therefore, we compared our results with the related work that inferred similar contexts. Our unsupervised method worked accurately on average 97.5% of the time, which was 17% more than reported for the GPS-based supervised method for inferring contexts home, work, shopping, dining, visiting, and other [13]. The accuracy, precision, and recall of our method were in general high because all of our users were very typical and easily predictable. For example, they had only one home, and they spent most of the working time in the main office. This reflects the reality of most users well. Nevertheless, some additional logic should be added to account for the atypical users. However, for the contexts free time and transit, the precision and recall were lower due to the discrepancy between the user labels and the automatically inferred contexts. Namely, our method misclassified the contexts that were mostly static but contained some elements of dynamism, such as shopping. Similarly, it misclassified the contexts that were in essence dynamic but contained some stops, such as resting while cycling or waiting for a bus during transit.

When assessing the correctness of the inferred contexts of stay regions, our method on average performed slightly less accurately than the supervised method based on GPS [10] but performed better than the essentially supervised approach in [30], which in addition to our sensors used a camera. In [10], the overall average accuracy for five contexts was 90.6%, for our three static contexts it was 89.7%, and for ten different contexts in [30] it was 56%. In [10], the contexts home and work were inferred with 100% accuracy, precision, and recall, while in our case the values for home were over 90%, but lower for work. Compared to [30], our recall for home and the precision for work were better, while the precision for home and the recall for work were worse. The good performance of our method could be a consequence of the simple context model and the simple logic for inferring two major contexts home and work, where the users spend most of their time. All our users had only one home, so the logic for this context worked very well. However, half of our users did not work only in the main office as predicted by our logic, which explains the low
recall for this context in comparison to [30]. Therefore, some improvements would be needed to account for other offices and field work. The context free time, which we defined as everything that was not home, work, or transit, accumulated the errors of all other contexts. With improved inference of the contexts work and transit, the inference of the context free time will improve as well. This could be achieved with, for example, better logic, additional sensors, or the overlay layer for control of transitions among various contexts, such as the Markov chain or the dynamic Bayesian network.

Nevertheless, some systematic errors will always remain because the entirely unambiguous context model, where each user situation would be attributed to only one context, is not realistic. For example, when a user walks up the stairs to her flat, should this be labelled as home or transit? This depends on the user perception. Thus, there will always be some error when the user annotations are used for evaluation.

We have shown that after the general context of a stay region had been determined, we could infer the subcontexts of its stay points. By doing this, we could approach the room-level context recognition, which is desired for some services. In the case of home, the bedroom could potentially be distinguished from places where a user spends most of her time during the day. Similarly, for the context work, the main office could be distinguished from other rooms. However, with the existing annotations we were not able to evaluate if such reasoning worked, so this remains an open question.

5.4. Other Findings and Comments. Despite the various characteristics of smartphones and the differences in the user lifestyles, our method proved to work robustly. We noticed that the Wi-Fi and location sensors on different smartphones behaved differently and most of the time the data were not sampled with the preset sampling frequency. This might be attributed to the Android peculiarities or the overload of CPU. On the other hand, the user lifestyles differed in the amount of time spent at home, number of work places, and diversity of free time activities. Despite all of these differences, the standard deviations of accuracy, precision, and recall were low, especially when finding the stay points. Nevertheless, the differences in the accuracy and precision among the users decreased after the proposed improvements, which resulted in the lower respective standard deviations. Thus, we believe that our method for finding stay points together with the proposed improvements is robust. Its robustness should be further investigated with more users. When evaluating the context inference, the standard deviations of accuracy, precision, and recall increased due to the very different user lifestyles and limited reasoning of our method. The context inference worked better for the users who spent most of their time at home and in the main office and worse for the users with more dynamic lifestyles. Although all of our conclusions were based solely on six users, we believe that the rather simple logic of our method would work for the majority of users.

Last, but not least, the application in its current form represented a huge burden on the battery life. As already demonstrated in [6, 16], the energy consumption can be optimised without losing much accuracy by dynamically adapting the sampling frequencies and using less power-consuming sensors when possible.

6. Conclusions

In the paper, we presented and evaluated a method that extracts the trajectories from the mobile sensor data and analyses them. The method is able to successfully determine user stay points, aggregate them into stay regions, and assign them four different user contexts: home, work, transit, and free time. In the part of analysing the user trajectories, our method outperformed the current best performing method SensLoc on our dataset in determining static versus dynamic mode, extracting locales that were relevant to the users, estimating the time spent at such locales, and detecting the accurate entrance and departure times. In the part of context inference, our method works similarly well as all the related work. The advantages of our method are as follows. It is entirely automatic and nonintrusive because it does not require user input at any stage. It is simple and universal because it uses only Wi-Fi and GPS but no other smartphone sensors, which enables its wide deployment. Contrary to most of the related work, the method relies primarily on the Wi-Fi fingerprints, which is preferable because of smaller energy consumption, instant functioning, and good coverage in urban and indoor environments.

We improved the SensLoc method [6] for inference of stay points by finding a better set of parameter values for our dataset and by proposing and evaluating five improvements. Four of them further improved the performance of the original method without losing the vital information about the user trajectories. We introduced a novel approach which aggregated the nearby stay points with the same semantic meaning to a user into meaningful stay regions. By doing this, we were able to determine more general locales of the same user context but at the same time maintained the details of the belonging stay points. For such aggregation, we evaluated six approaches and found out that the aggregation scheme based on the fingerprint similarity resembled the real stay regions the most. Finally, we introduced a hierarchical context model and used a spatiotemporal logic to infer four most general user contexts. These were on average accurately inferred 97.5% of the time. Furthermore, we demonstrated how the more detailed user contexts could be extracted from stay points and stay regions that were determined in the previous steps of our method.

The results indicate that the method is robust and can be used with various users and on various smartphones. Due to its characteristics, the entire method or its modules can potentially be deployed in various fields, such as in adjustment of the multimedia applications and services, in the recommender systems, and in the fleet management. According to the needs in the specific field of deployment, the method can be further adjusted and expanded.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.
Acknowledgments

Special thanks are due to the European Union for the partial financing of a young researcher training programme from the European Regional Development Fund, European Social Fund, and Cohesion Fund and to the authors’ coworkers for their support.

References

[1] M. Weiser, “The computer for the 21st century,” ACM SIGMOBILE Mobile Computing and Communications Review, vol. 3, no. 3, pp. 3–11, 1999.

[2] Ericsson, “Ericsson mobility report,” 2014.

[3] A. K. Dey, G. D. Abowd, and D. Salber, “A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications,” Human-Computer Interaction, vol. 16, no. 2–4, pp. 97–166, 2001.

[4] I. Clarke and T. B. Flaherty, “Mobile portals: the development of m-commerce gateways,” in Mobile Commerce: Technology, Theory and Applications, B. E. Mennecke and T. J. Strader, Eds., pp. 185–201, IGI Publishing, Hershey, Pa, USA, 2003.

[5] D. H. Kim, J. Hightower, R. Govindan, and D. Estrin, “Discovering semantically meaningful places from pervasive RF-beacons,” in Proceedings of the 8th ACM International Conference on Ubiquitous Computing (UbiComp ’09), pp. 21–30, October 2009.

[6] 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 International Conference on Embedded Networked Sensor Systems (SenSys ’09), pp. 43–56, ACM, Zürich, Switzerland, November 2010.

[7] K. Xie, K. Deng, and X. Zhou, “From trajectories to activities: a spatio-temporal join approach,” in Proceedings of the International Workshop on Location Based Social Networks (LBSN ’09), pp. 25–32, November 2009.

[8] K. Zhang, H. Li, K. Torkkola, and M. Gardner, “Adaptive learning of semantic locations and routes,” in Proceedings of the 1st International ICST Conference on Autonomic Computing and Communication Systems, pp. 193–210, Rome, Italy, October 2007.

[9] D. Ashbrook and T. Starner, “Learning significant locations and predicting user movement with GPS,” in Proceedings of the 6th International Symposium on Wearable Computers (ISWC ’02), pp. 101–108, 2002.

[10] L. Liao, D. Fox, and H. Kautz, “Extracting places and activities from GPS traces using hierarchical conditional random fields,” International Journal of Robotics Research, vol. 26, no. 1, pp. 119–134, 2007.

[11] C. Lee, G. Yoon, and D. Han, “A probabilistic place extraction algorithm based on a superstate model,” IEEE Transactions on Mobile Computing, vol. 12, no. 5, pp. 945–956, 2013.

[12] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, “Collaborative location and activity recommendations with GPS history data,” in Proceedings of the 19th International World Wide Web Conference (WWW ’10), pp. 1029–1038, April 2010.

[13] L. Liao, D. J. Patterson, D. Fox, and H. Kautz, “Building personal maps from GPS data,” Annals of the New York Academy of Sciences, vol. 1093, no. 1, pp. 249–265, 2006.

[14] 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 World Wide Web Conference (WWW ’09), vol. 49, pp. 791–800, April 2009.

[15] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, “Extracting places from traces of locations,” in Proceedings of the 2nd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH ’04), pp. 110–118, October 2004.

[16] Y. Wang, J. Lin, M. Annavaram et al., “A framework of energy efficient mobile sensing for automatic user state recognition,” in Proceedings of the 7th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys ’09), pp. 179–192, ACM, Kraków, Poland, June 2009.

[17] J. Rekimoto, T. Miyaki, and T. Ishizawa, “LifeTag: Wi-Fi-based continuous location logging for life pattern analysis,” in Proceedings of the 3rd International Conference on Location- and Context-Awareness (LoCA ’07), pp. 35–49, 2007.

[18] Y. Chon and H. Cha, “LifeMap: a smartphone-based context provider for location-based services,” IEEE Pervasive Computing, vol. 10, no. 2, pp. 58–67, 2011.

[19] R. Montoliu and D. Gatica-Perez, “Discovering human places of interest from multimodal mobile phone data,” in Proceedings of the 9th International Conference on Mobile and Ubiquitous Multimedia (MUM ’10), vol. 12, pp. 1–10, 2010.

[20] T. Lovett and E. O’Neill, “Simulating user intervention for interactive semantic place recognition with mobile devices,” in Proceedings of the RecSys Workshop on Personalizing the Local Mobile Experience (LocalPeMA ’12), pp. 13–18, September 2012.

[21] M. Nanni and D. Pedreschi, “Time-focused clustering of trajectories of moving objects,” Journal of Intelligent Information Systems, vol. 27, no. 3, pp. 267–289, 2006.

[22] Skyhook, 2008, http://www.skyhookwireless.com/.

[23] J. Rekimoto, T. Miyaki, A. Shionozaki, and T. Sueyoshi, “PlaceEngine,” 2006, http://www.placeengine.com/.

[24] Google, “Android Location Package,” http://developer.android.com/reference/android/location/Location.html.

[25] Skyhook, 2008, http://www.skyhookwireless.com/.

[26] Y. Chon, E. Talipov, and H. Cha, “Autonomous management of everyday places for a personalized location provider,” IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, vol. 42, no. 4, pp. 518–531, 2012.

[27] H. Verkasalo and B. J. Salmeron, “Analysis of the contextual behaviour of mobile subscribers,” in Communications Infrastructure. Systems and Applications in Europe, R. Mehmoond, E. Cerqueira, R. Piesiewicz, E. Cerqueira, R. Piesiewicz, and I. Chlamtac, Eds., vol. 16, pp. 252–266, Springer, Berlin, Germany, 2009.

[28] D. Raptis, N. Tsleios, and N. Avouriis, “Context-based design of mobile applications for museums: a survey of existing practices,” in Proceedings of the 7th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI ’05), pp. 153–160, aut, September 2005.

[29] G. D. Abowd, C. G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton, “Cyberguide: a mobile context-aware tour guide,” Wireless Networks, vol. 3, no. 5, pp. 421–433, 1997.

[30] Y. Chon, Y. Kim, and H. Cha, “Autonomous place naming system using opportunistic crowdsensing and knowledge from crowdsourcing,” in Proceedings of the 12th International Conference on Information Processing in Sensor Networks (IPSN ’13), pp. 19–30, April 2013.
[31] R. Wind, C. S. Jensen, K. H. Pedersen, and K. Torp, “A testbed for the exploration of novel concepts in mobile service delivery,” in Proceedings of the 8th International Conference on Mobile Data Management (MDM’07), pp. 218–220, May 2007.

[32] 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 (GIS’11), pp. 112–121, ACM, Chicago, Ill, USA, November 2011.

[33] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland, “Social fMRI: investigating and shaping social mechanisms in the real world,” Pervasive and Mobile Computing, vol. 7, no. 6, pp. 643–659, 2011.

[34] M. C. González, C. A. Hidalgo, and A.-L. Barabási, “Understanding individual human mobility patterns,” Nature, vol. 453, no. 7196, pp. 779–782, 2008.

[35] P. Korpijärvi, M. Koskinen, I. Peltola, S.-M. Mäkelä, and T. Seppänen, “Bayesian approach to sensor-based context awareness,” Personal and Ubiquitous Computing, vol. 7, no. 2, pp. 113–124, 2003.