

Efficient Location Sensing in Longitudinal Cohort Studies

Afzal Hossain and Christian Poellabauer
 Department of Computer Science and Engineering
 University of Notre Dame, IN 46556, USA
 {ahossain, cpoellab}@nd.edu

Abstract—A longitudinal cohort study is a popular research method to observe a group of people over a prolonged period of time, e.g., to learn about their health, wellness, and social habits. Smartphones have become a very popular tool to perform such studies at a large scale. Location is an essential form of sensor data that can not only be used to monitor users' mobility and social interaction patterns, but also to identify places of personal significance, i.e., places where a user spends a significant amount of time, such as a user's home, workplace, and preferred social gathering places. However, continuously tracking a user's location can have significant impacts on the battery lifetime of a smartphone. Therefore, instead of frequent period location sensing, this paper identifies smartphone events that can be used to trigger location sensing at a much lower rate (and therefore more energy-efficiently), while still providing accurate location data. In this work, we demonstrate that this approach allows us to determine a user's significant places with an accuracy of 85%, while saving over 60% in computational and energy overheads.

I. INTRODUCTION

The location (GPS) sensor is the most important and most frequently used sensor in smartphone sensing systems, since a user's location adds an essential piece of context information to the user's activities. For example, using continuous traces of location data over extended periods of time, we can learn about a user's *personal places of significance*, i.e., locations a user frequently visits or where a user spends significant amounts of time [1], [2]. For many data collection scenarios, it is required to collect the location data in a continuous fashion to accurately determine these places of interest or to detect when a user moves from one location to another. This is the case for many longitudinal cohort studies [3], which represent a very popular form of research data collections in areas such as healthcare, social science, psychology, etc.

However, the location sensor of a smartphone is also one of the most energy consuming components [4], i.e., frequent or continuous use of the location sensor will impact the battery lifetime of the device [5]. If the battery of a smartphone expires too soon, this will lead to significant gaps in the smartphone data traces or, even worse, may lead a user to deactivate the phone's sensing tasks. One energy saving approach is to replace the period location sensing with a mechanism that triggers the sensor less frequently, while still collecting sufficient data for location analysis. This approach has been used by various tracking applications [6], [7], which use the accelerometer to detect motion and activates the location sensor only after a sufficient amount of movement has been

measured. In other scenarios, changes in Wi-Fi or cellular connectivity has also been used as the trigger for location sensor [8], often in combination with machine learning techniques to predict the need for sensor activation [9].

In day to day life, users take their smartphones everywhere and frequently check their devices for new messages, missed calls, news updates, or social media activity. A typical scenario is that users remove their phones from their pockets, unlock it, check for any activity, and then lock the phone again. That is, every time we use a phone, there will be an *unlock event*, followed by a *lock event*. There is an increasing amount of evidence [10], [11] of strong correlations between a user's behaviors, intentions, or activity patterns and phone events (such as screen lock/unlock events).

We believe that such *phone screen events* could be associated with changes in location and could be used to trigger location sensing. The energy costs for detecting phone screen events are nearly zero, and any smartphone app can register for such events and thereby use them as sensing triggers. In this paper, we explore the possibility of using screen events as reliable markers for location changes.

II. EXPERIMENTAL SETUP

We claim that screen events of a phone can be used to indicate when a user may be located at a place of significance to the user. Usually, *stay points* from a location trace are used to determine a place, where a location trace is a series of raw location data points in chronological order, each data point consisting of a timestamp, latitude, longitude, and an accuracy indicator. A stay point is then defined as a location where a user stayed for a certain minimum amount of time. The accuracy of determining a place using a location trace depends on the granularity (or *sampling rate*) of the data points, and we can use stay points computed using a very high sampling rate as *ground truth*. For example, let $A = \{L_{A1}, L_{A2}, \dots, L_{AN}\}$ be a series of location data points collected using a high sampling rate, thereby serving as ground truth or *reference log*. Another series $B = \{L_{B1}, L_{B2}, \dots, L_{BM}\}$ then represents location data points recorded at a much lower rate, e.g., triggered using phone screen events. The number of data points in series A will then be much higher than the number of data points in B. The goal of our work is then to demonstrate that, even with a much smaller number of data points, the accuracy will still be acceptable.

A. Data Collection

The NetHealth project [12] is a social and behavioral health study cohort conducted by the University of Notre Dame. As part of this study, students use a smartphone app to collect data from a variety of sensors continuously. While the NetHealth database consists of records from more than 400 students for over two years so far, in this paper, we limit our analysis to the 130 individuals in our study and investigate their data collected over a two-month period.

The *location manager* of the app is responsible for acquiring periodic location updates, which are obtained every three minutes. Most students in the study walk between campus locations and this rate is typically sufficient to get relatively accurate location information. In our work, we use this location trace as ground truth (i.e., period location trace A). Further, the app also recognizes when the user locks and unlocks the device and records time of such an event and the type of event (lock versus unlock). In addition, for the purpose of this paper, we also record the location at those events, we refer to this a location trace B (which is event-triggered and not periodic, unlike trace A).

B. Preliminary Analysis

The number of screen events in a day will differ from individual to individual. Further, for an individual, phone use will vary on different days. Before we investigate the use of phone screen events to trigger location sensing, we take a closer look at trends in phone interactions.

The leftmost boxplot in Figure 1 shows the distribution of the average screen events per day for the 130 users investigated in this paper. The small circle in the box represents the mean of the distribution. The average number of screen events per day is 187, i.e., location trace B contains on average 187 location records (compared to 480 location records in location trace A). Further, very often screen events happen rather frequently after each other, e.g., a user may unlock the phone, check for an potential new messages, and immediately relock the phone. During such brief time intervals, the user's location will typically not have changed much. Therefore, we also introduce a *skip window*, which is a time interval that is triggered by a screen event and no additional location updates are recorded until the next event after the skip window has passed. The left most boxplot has a skip window of zero; the boxplots to the right of the graph then show the distribution for skip windows varying from 1 minute to 9 minutes in 1-minute intervals. This graph shows that with increasing skip windows, the average number of location records in location trace B reduce significantly, e.g., to less than 100 for skip windows of 2 minutes and to about 50 for windows of 8 minutes and larger. The tradeoff in the choice of the skip window is that a larger window will reduce the size of location trace B, but also to decreased location accuracy, as discussed in Section IV.

III. METHODOLOGY

In our analysis, location traces are considered on a daily basis, i.e., the location sequence of a user X on the first day

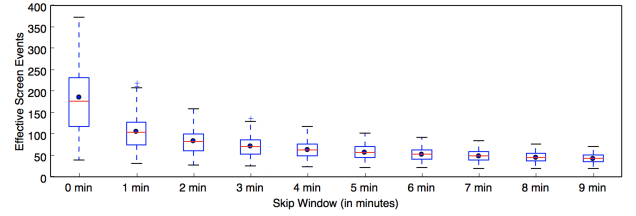


Figure 1: Phone screen event distributions per day as a function of the skip window

will be considered a separate entity from the location sequence of the same user for the second day, etc.

A. Stay Point Detection

A stay point is a place represented by a cluster of location points within a certain geographic boundary where a user resides longer than a certain amount of time, in our experiment it is 30 minutes (frequently used in literature [13]). Each stay point has a mean coordinate that is determined using the location points that belong to the stay point. Further, each stay point also has an *entry time* (timestamp of first location point) and *exit time* (timestamp of last location point). We use DBSCAN [14] to group location records into stay points. DBSCAN requires two parameters: one is the maximum distance between two points (ϵ) to determine if they are neighbors when forming a group, the other is the minimum number of data points (*min_pts*) needed to create a cluster. DBSCAN returns cluster labels associated with each data point, where a label indicates the cluster in which a data point belongs. Then, we scan location data points and their cluster labels in chronological sequence. We find the windows where all points within that window belong to the same cluster label in sequence. We only keep the windows that spanned over 30 minutes.

Each individual in our dataset has two location traces for each day: location trace A containing all periodic location sensor readings (ground truth) and location trace B containing the location readings triggered by the screen events. We apply the stay point detection algorithm described above on both traces, with $\epsilon = 50m$. Since the records in trace A are spread apart by 3 minutes, given this time threshold, at least 11 location data points are required to form a stay point. In contrast, location data points in trace B are sporadic and the minimum number of data points to form a stay point is therefore uncertain. However, a stay point must have at least an *entry time* and an *exit time*, i.e., we can set the value of *min_pts* in the algorithm to 2. The stay points detection algorithm is then executed with the following parameters:

$$SP_A = \text{Stay_Points_Detection}(A, 50m, 30min, 11)$$

$$SP_B = \text{Stay_Points_Detection}(B, 50m, 30min, 2)$$

B. Similarity Measurement

Figure 2 shows an example of two sets of stay points SP_A (for stay points determined from trace A) and SP_B (for stay points determined from trace B). The x-axis displays the time

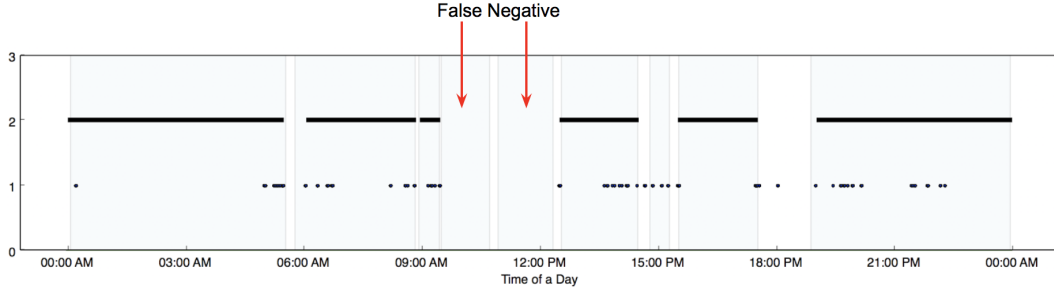


Figure 2: Stay points and screen events during a 24-hour span

from midnight to 23:59pm of one particular day. The shaded areas represent the stay points in SP_A , where the left edge of a shaded area indicates the entry time and right edge the exit time. The horizontal line segments at $y=2$ represent the stay points in SP_B , where the beginning of a segment is again the entry time and the end of a segment the exit time. The scatter plot at $y=1$ displays the phone screen events.

1) *Segment Overlap Score*: We calculate the amount of overlap between the two sets. To do so, we analyze the 24 hours span on a per-minute level, i.e., 1440 slots (the first slot being midnight and then each minute represented as a different slot). An array size of 1440 contains the value for each minute of the day, indicating the coordinate of the stay point. A null value in the array indicates that the person was not at any stay point on that particular minute.

Next, we compare one-to-one values of the two arrays generated from SP_A and SP_B . Since the stay points for SP_A and SP_B are determined using different sets of location points, the stay point coordinate will differ, even if they are approximately the same stay point. We therefore have to also define a distance threshold (in this paper we use a threshold of 200m), i.e., if the stay point coordinates are within this distance threshold they are considered to be the same stay point. In this approach, the comparison may lead to five possible scenarios. If the distance between two segment coordinates is less than the distance threshold, we then mark this segment as matched. We also count it as matched if both entries are null. Other scenarios are unmatched cases, i.e., where one entry has a coordinate, while the other is null or where the distance between the entries is greater than the threshold. We then count the number of matching entries (which in the ideal case is 1440) and divide this number by 1440 to obtain a percentage.

2) *Measure of Relevance*: We can also use precision and recall values to measure the similarity between two sets of stay points. If a stay point in SP_A overlaps with a stay point in SP_B by more than 30 minutes (i.e., the threshold time for a stay point), then it is positively identified (true positive); otherwise, we consider it as a false negative. In Figure 2, the vertical arrows indicate two false negatives, where there is no corresponding stay point using trace B. False positive is the case where screen events form a stay point which is not part of the ground truth. This case is unlikely since typically trace

A will have many more location points.

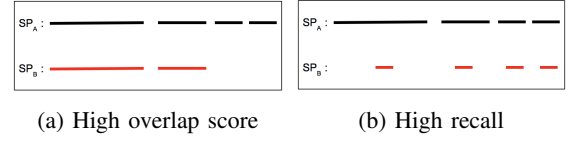


Figure 3: Two different cases of similarity measurement

The outcome of the segment overlap score depends on the width of the stay points (i.e., entry and exit times), including true negative regions. On the other hand, precision and recall values focus the identification of stay points (i.e., the number of stay points) from the perspective of the secondary data source (i.e., the phone screen events). Figure 3 compares these two approaches to measure similarity, i.e., in the case of Figure 3a, we have a high segment overlap score, but low recall, while the situation is the opposite in Figure 3b. Therefore, if two traces A and B produce a high score using both metrics, then it can be said that trace B is very similar to trace A.

C. Efficiency Measurement

Trace B (generated by phone screen events) typically contains much fewer data points than trace A (periodic location sensing). Since this data has to be stored first locally on the mobile device and then transmitted to a remote server for storage and processing, we also consider the amount of data being produced and communicated. Since each screen event generates exactly one data entry, we can express the daily savings in the following manner:

$$savings = 1 - \frac{\text{average number of events}}{\text{number of records in ground truth}} \quad (1)$$

Equation 1 also applies to measuring the energy savings. The energy consumption caused by the location sensor is not precisely linear to the number of API calls, i.e., it may vary depending on the positional fix, current location, interferences, etc. But the number of API calls provides an indicator of the potential energy consumption.

IV. RESULTS AND ANALYSIS

Our experimental dataset contains location traces of 130 students collected from mid November to mid January. During

this time period, students first spent most of their time on campus (at the end of fall semester), then traveled to their respective homes for winter break, and then again moved back to campus to start the spring semester. So, their location sequences will have a lot of variety. We remove the location traces on those days when students did not leave a stay point at all (e.g., spending the entire day in their dorm rooms).

A. Accuracy

We measure the accuracy of our hypothesis by calculating similarity scores between stay points of traces A and B. Table I shows the segment overlap scores for different values of *skip_window*. We see that we achieve a maximum of 88.2% similarity with a window of size 0. The score decreases as *skip_window* grows.

Table I: Similarity Measurement

Skip Window	Overlap Score	Precision	Recall	F1 Score
0 min	88.2%	99.1%	82.8%	90.2%
1 min	87.8%	99.1%	81.8%	89.6%
2 min	87.25%	99.2%	80.4%	88.8%
3 min	86.37%	99.2%	78.7%	87.8%
4 min	85.4%	99.2%	76.8%	86.7%
5 min	84.34%	99.3%	75%	85.5%

True stay points can be in non-overlapping segments. Therefore, we calculate precision, recall, and F1 score to evaluate accuracy from another angle. Table I also summarizes these scores for different values of *skip_window*. We see that the precision is always more than 99%, which means that very few stay points in trace B are a false positive. And that is reasonable, because if the locations from trace B form a cluster, then during that period, there must be sufficient location data in trace A to also form a cluster. The maximum recall value is around 83% and it decreases as *skip_window* increases. False negative indicates that the participant generated either zero or an insufficient number of events to form a cluster. F1 score is above 85% in all the cases.

Table II: Measurement of Data and Energy Efficiency

Skip Window	Number of Location Records (Screen Events)	Number of Location Records (Ground Truth)	Savings (%)
0 min	187	480	60.9
1 min	107		77.5
2 min	85		82.1
3 min	73		84.6
4 min	65		86.4
5 min	59		87.7

B. Efficiency

Table II shows the savings using trace B over trace A in terms of the amount of data generated and energy consumed. Trace A (ground truth) has location data in 3 minutes interval, therefore it has total 480 records per day. We see as the number of effective screen events drop with growing values of *skip_window*, the savings grow. Combining the scores from both tables, we suggest that setting a *skip_window* of 2 or

3 minutes provides a good trade-off in terms of accuracy and efficiency. While the accuracy does not drop significantly, the savings go above 80%. Note that this of course depends on the periodicity of trace A too, i.e., the choice of *skip_window* is likely to change if the periodicity of trace A changes.

V. CONCLUSIONS

Accurately determining significant places of personal interest has found a lot of interest in a variety of applications. The “traditional” approach is to use energy-consuming sensors to identify location changes or to implement complex adaptive learning approaches that may require a long time to learn or may fail to detect sudden events. In this paper, we explore the possibility of using screen events to identify place changes.

REFERENCES

- [1] S. Vhaduri and C. Poellabauer, “Cooperative discovery of personal places from location traces,” in *Proceedings of the 25th International Conference on Computer Communication and Networks (ICCCN)*, Waikoloa, Hawaii, August 2016.
- [2] S. Vhaduri and C. Poellabauer, “Hierarchical cooperative discovery of personal places from location traces,” *IEEE Transactions on Mobile Computing*, vol. PP, November 2017.
- [3] A. Hossain and C. Poellabauer, “Challenges in building continuous smartphone sensing applications,” in *Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2016 IEEE 12th International Conference on, pp. 1–8, IEEE, 2016.
- [4] K. Katevas, H. Haddadi, and L. Tokarchuk, “Sensingkit: Evaluating the sensor power consumption in ios devices,” in *Intelligent Environments (IE)*, 2016 12th International Conference on, pp. 222–225, IEEE, 2016.
- [5] C. Miller, S. Chasins, C. Farris, J. Varner, C. Carmony, C. Poellabauer, and A. Striegel, “An integrated monitoring system for mobile phones,” in *Proceedings of the 1st International Workshop on Sensing for App Phones (PhoneSense)*, Zurich, Switzerland, November 2010.
- [6] J. Paek, J. Kim, and R. Govindan, “Energy-efficient rate-adaptive gps-based positioning for smartphones,” in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pp. 299–314, ACM, 2010.
- [7] L. Zhang, J. Liu, H. Jiang, and Y. Guan, “Senstrack: Energy-efficient location tracking with smartphone sensors,” *IEEE sensors journal*, vol. 13, no. 10, pp. 3775–3784, 2013.
- [8] 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*, pp. 43–56, ACM, 2010.
- [9] A. Khan, S. K. A. Imon, and S. K. Das, “An energy efficient framework for localization and coverage in participatory urban sensing,” in *Local Computer Networks (LCN)*, 2014 IEEE 39th Conference on, pp. 193–201, IEEE, 2014.
- [10] P. Chow, H. Xiong, K. Fua, W. Bonelli, B. A. Teachman, and L. E. Barnes, “Sad: Social anxiety and depression monitoring system for college students,” 2016.
- [11] S. Abdullah, M. Matthews, E. L. Murnane, G. Gay, and T. Choudhury, “Towards circadian computing: early to bed and early to rise makes some of us unhealthy and sleep deprived,” in *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing*, pp. 673–684, ACM, 2014.
- [12] R. Purta, S. Mattingly, L. Song, O. Lizardo, D. Hachen, C. Poellabauer, and A. Striegel, “Experiences measuring sleep and physical activity patterns across a large college cohort with fitbits,” in *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, pp. 28–35, ACM, 2016.
- [13] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W.-Y. Ma, “Mining user similarity based on location history,” in *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, p. 34, ACM, 2008.
- [14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Kdd*, vol. 96, pp. 226–231, 1996.