LIPS: Lifestyle Learning via Mobile Phone Sensing

Xiang Sheng, Jian Tang, Jing Wang, Teng Li, Guoliang Xue and Dejun Yang

***Abstract*—In this paper, we propose to learn LIfestyles of mobile users via mobile Phone Sensing (LIPS), and we develop a system and algorithms to realize this idea. First, we present the workflow and architecture of our system, LIPS. Combining both unsupervised and supervised learning, we propose a hybrid scheme for lifestyle learning, which consists of two parts: characterization and prediction. Specifically, we present a two-stage algorithm to characterize the lifestyle of a mobile user using Places of Interest (PoIs), which leverages two different algorithms for coarse-grained and fine-grained clustering in two stages respectively. Based on discovered PoIs, we present a method to build a model to predict his/her future activities using a supervised classification algorithm. In addition, we present an adaptive sampling algorithm for improving energy efficiency, which leverages both the discovered PoIs and the lifestyle model for adaptively controlling the sampling rate. We implemented the proposed system and algorithms based on the Android platform. We have validated and evaluated LIPS via extensive field tests carried out for over 1.5 months in 6 cities of USA. The experimental results show that LIPS can 1) well discover PoIs of mobile users, 2) precisely predict their future activities, and 3) achieve significant energy savings (compared to periodic sampling).**

***Index Items:* Mobile Computing, Mobile Phone Sensing, Human-Centric Sensing, Energy Efficiency**

I. Introduction

A smartphone is usually equipped with a rich set of embedded sensors such as camera, GPS, accelerometer, digital compass, gyroscope, and so on. External sensor(such as Google Glass, Smart Watch, Fitbit, Sensordrone [19], etc.) can be connected to the phone via its network interface (such as Bluetooth). The sensors of a smartphone can easily detect the context (such as location, local weather, activities, etc.) of its mobile user.

In this paper, we propose to learn LIfestyles of mobile users via mobile Phone Sensing (LIPS). According to businessdictionary.com, "*Lifestyle is expressed in both work and leisure behavior patterns and (on an individual basis) in activities, attitudes, interests, opinions, values, and allocation of income.*" Our idea is to leverage multiple sensors on a smartphone for obtaining a comprehensive view of the context (such as location, local weather, activities, etc.) of a mobile user over a long period, and to find out what a mobile user likes to do (characterization) and what he/she will do next (prediction) based on the collected sensor data. Such a lifestyle learning system can be used to support a large variety of applications for improving life quality. For example, a major application is to recommend local businesses to mobile users based on not only his/her location but also his/her lifestyle. This work represents one of the first efforts along this line, which is focused on lifestyle learning, while leaving lifestyle-aware recommendation or lifestyle-based applications for future research.

Xiang Sheng, Jian Tang, Jing Wang, Teng Li are with the Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY 13244. Email: {xsheng,jtang02,jwang93,tli01}@syr.edu. Guoliang Xue is with the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85287. Email: xue@asu.edu. Dejun Yang is with the Department of Electrical Engineering and Computer Science, Colorado School of Mines, Golden, CO 80401. Email: djyang@mines.edu. This research was supported in part by NSF grants 1218203, 1525920, 1461886 and 1444059. The information reported here does not reflect the position or the policy of the federal government.

We build a system, LIPS, to realize our idea. LIPS consists of a mobile frontend and a learning server on the backend. The mobile frontend can be implemented as a mobile app that reports the context information collected by sensors of the mobile phone to the learning server periodically. Based on this information, the learning server builds models for lifestyles of mobile users. Combining both unsupervised and supervised learning, we propose a hybrid scheme for lifestyle learning, which consists of two parts: characterization and prediction. Specifically, we present a two-stage algorithm to characterize the lifestyle of a mobile user using Places of Interest (PoIs), which leverages two different algorithms for coarse-grained and fine-grained clustering in two stages respectively. Based on discovered PoIs, we present a supervised learning based algorithm to build a model for predicting the future activities of a mobile user.

In addition, operating smartphone sensors (such as GPS) could be energy consuming. Even though some sensors (such as accelerometer) are always active, a thread needs to be spawned to collect its readings, which consumes energy too. To enable green lifestyle learning, we present an adaptive sampling algorithm, which adaptively controls the sampling rate according to discovered PoIs and the lifestyle model.

We propose practical and effective solutions to fundamental problems of lifestyle learning (learning and energy-efficient sampling). Specifically, we summarize our contributions in the following:

- We present an effective hybrid scheme for lifestyle learning, which combines both unsupervised and supervised learning.
- We present an energy-efficient sampling algorithm, which leverages the discovered PoIs and the lifestyle model for adaptively controlling the sample rate.
- We performed extensive field tests to validate and evaluate LIPS. The experimental results well justify the effectiveness and efficiency of LIPS on lifestyle learning.

II. Overview of LIPS

LIPS consists of two parts: mobile frontend and learning server, as illustrated by Fig. 1. The mobile frontend is implemented as a mobile app that runs on each mobile user's smartphone. The learning server, however, runs in the cloud.

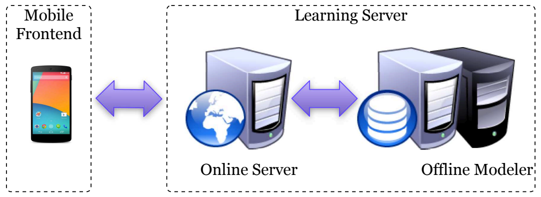


Fig. 1. The LIPS system

We can simply deploy multiple learning servers and load balancers if we need to serve a large number of mobile users from different locations.

In our design, a learning server is composed of an online server and an offline modeler. The online server supports a set of services for the mobile frontend, including login, raw data processing, messaging, notification, etc. The offline modeler, however, deals with lifestyle learning in the backend, which includes running the clustering-based characterization algorithm (Section III-A) and the supervised learning based prediction algorithm (Section III-B) on collected sensor data. The online server is designed to be light-weighted, which provides immediate online responses to the mobile frontend. However, lifestyle learning involves compute-intensive and time-consuming workload, which can only be done in an offline manner on powerful servers. This design ensures that online requests from the mobile frontend are not delayed by the time-consuming learning process.

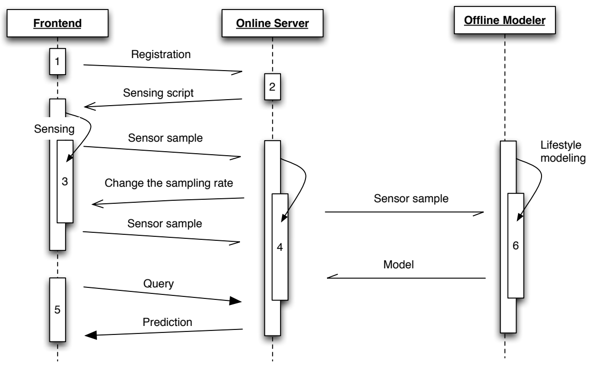


Fig. 2. The workflow of LIPS

We illustrate how the mobile frontend, the online server and the offline modeler interact with each other and how LIPS works in Fig. 2, which are further described in the following:

1) A mobile user registers for the lifestyle learning service by sending his/her personal information along with his/her preferences (e.g. only allowing coarse locations rather than fine locations) to the online server.
2) The online server accepts (or rejects) the registration request and sends sensing scripts according to user preferences to the mobile frontend.
3) The mobile frontend collects the mobile user's context information (Section III) using mobile phone sensors periodically and sends sensor data to the online server, which then stores them into a database. The mobile frontend adaptively adjusts its sampling rate to trade off energy efficiency and learning performance (Section IV).
4) The offline modeler periodically pulls sensor samples from the database.
5) The offline modeler discovers PoIs and builds the lifestyle model for activity prediction on a daily basis according to received sensor samples. And, it updates the online server with PoIs and lifestyle model.
6) The mobile frontend periodically sends a query with the current context information (Section III) to the online server. This query can also be sent in an ad-hoc manner.
7) The online server replies with the prediction results (Section III-B).

We implemented the mobile frontend and the online server for sensor data collection based on the SOR system [18] we built before. Due to space limitation and similarity, we omit details about them, which can be found at [18]. The offline modeler consists of four modules: Data Pre-processor, PoI Discover, Lifestyle Modeler and Place Information Provider. The Data Pre-processor decodes binary raw data, based on which numerical values (a.k.a feature data) will be generated (Section III) and stored into the database as input for the PoI Discover and the Lifestyle Modeler. The PoI Discover analyzes each user's feature data and discovers his/her Places of Interest (PoIs), which will be described in details in Section III-A). The Lifestyle Modeler builds a model for predicting future activities of a mobile user based on discovered PoIs using a supervised learning algorithm, which will be introduced in details in Section III-B). The Place Information Provider is used to retrieve the actual place (such as restaurant, coffee shop, etc.) information given the location of a PoI, which will be used for building the lifestyle model. In our system, we used Google's Place API [8] to obtain such information.

In addition, "sensor" has a much broader meaning in LIPS, which refers to data source that can provide context information of a mobile user. Therefore a sensor could be: 1) an embedded sensor (such as GPS, accelerometer, digital compass, etc.) on a mobile phone; 2) a service that can provide context information (such as local weather) to mobile users via APIs; or 3) an external sensor (such as Fitbit and Sensordrone [19]) that can be connected to a mobile phone via its network interface (such as Bluetooth).

III. Lifestyle Learning

As described above, the goal of lifestyle learning is to find out what a mobile user likes to do (characterization) and what he/she will do next (prediction). Mobile phones are usually carried by their users almost all the time, which make them perfect devices for providing useful context information to learn the lifestyle of mobile users. In LIPS, *sensor samples* are collected periodically by the mobile frontend for lifestyle learning. A sensor sample $\mathbf{s}$ is defined by a 3-tuple $(t, l, \mathbf{D})$, where t is the timestamp, l is the location, and $\mathbf{D}$ is a set of raw sensor readings that are used to produce feature data (described below).

In LIPS, a list of features are extracted from a sensor sample, which are then used as input for discovering of PoIs of a mobile user (Section III-A) and for predicting his/her activities (Section III-B):

1) *Day and Time:* the day (Monday, Tuesday, etc.) and the time at the sampling instant. Note that both features are very important since the period of many people's lifestyles is one week and usually their activities in a day are highly time-dependent.
2) *Location and Speed:* the location and the moving speed at the sampling instant, which are obtained via either GPS (fine) or Google's Location Services (coarse) according to user's preferences.
3) *Moving State*: {On-foot, Driving, Bicycling, Still, Unknown}, which can be obtained by calling the activity recognition API [1] in the Google Play Services.
4) *Step Frequency*: the numbers of steps per second (if on-foot). Each sensor sample includes 10 continuous accelerometer readings, which are then used to estimate the step frequency using the method introduced in [2].
5) *Weather Condition*: {Sunny, Cloudy, Raining, Snowing}, which can be obtained by calling the REST Weather Channel API [21].
6) *Local Outdoor Temperature:* the outdoor temperature at the sampling instant and location, which can be obtained by calling the REST Weather Channel API [21] too.
7) *User State*: {Active, Inactive}, which shows whether or not the user is actively using the mobile phone. This can be obtained by using the Android system API to check if any app is launched in the past sampling period.

We select these features to build the lifestyle model because we believe they may all have impacts on a mobile user's activities. For example, a mobile user usually goes to a restaurant on Saturday night, however, if the weather happens to be bad (e.g., snowing), he/she may decide not to go out.

A. Lifestyle Characterization with Places of Interests (PoIs)

In order to learn lifestyle of a mobile user, we first need to know which places he/she likes to go, which, however, is hard to tell simply based on a set of collected sensor samples since some of them may be taken when he/she moves from one place to another. We characterize the lifestyle of a user using PoIs. A PoI is a place that a mobile user has visited, which could be a grocery store, a shopping mall, a restaurant, etc. Discovering PoIs for a mobile user is the first step of lifestyle learning.

From our field-tests, we find that sensor samples have the following two properties: 1) The set of collected sensor samples contains both samples related to PoIs, and samples corresponding to movements between PoIs, which may not be relevant. 2) The number of PoIs can not be determined beforehand.

Intuitively, we can apply a clustering algorithm to find clusters based on collected samples, which can then be used to identify desired PoIs. However, we find most existing clustering algorithms are not suitable for our problem due

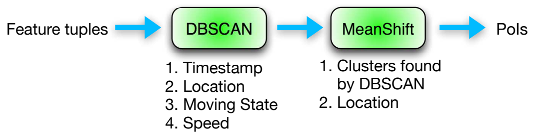


Fig. 3. The PoI discovery algorithm

to the following reasons: 1) Those clustering algorithms that require the number of clusters as input, such K-means [9], are not suitable here, since the number of PoIs are not known beforehand. 2) A naive approach, in which clusters are determined simply based on the amount of time a user stays in an area, is not applicable since a user may stay on certain part of a road for a long time due to traffic jam. 3) Clustering simply based on locations without taking time into consideration, may lead to many false PoIs due to overlapping samples collected over multiple days. 4) It is not reasonable to determine whether a place is a PoI or not simply based on a moving speed threshold. For example, a mobile user may jog around a place (such as a park or a lake) with samples evenly distributed around it. By just setting up a fixed speed threshold for clustering, it may not be discovered as a PoI. In summary, clustering should be done according to multiple relevant features rather than a single feature.

Based on our observations, we design a two-stage algorithm to discover PoIs from a set of sensor samples, which is illustrated in Fig. 3. As described above, instead of directly using raw sensor data, we extract useful information from collected sensor samples to produce feature data as input. We choose the DBSCAN [7] and MeanShift [5] algorithms for coarse-grained and fine-grained clustering in the first and second stage respectively. In the first stage, our main goal is to filter out those samples related to movements between two PoIs rather than actual PoIs. For each collected sample $\mathbf{s} \in \mathbf{S}$, we extract its *timestamp, location, moving state and moving speed* to build a feature tuple $\mathbf{f} = (t, l, a, v) \in \mathbf{F}$. The tuple values are normalized in each feature dimension.

We feed those feature tuples into the DBSCAN algorithm to produce a set of clusters. DBSCAN is a density-based clustering algorithm, which ensures that the output clusters are areas of high density while outside of the clusters are areas of low density [7]. This is desirable for our problem since DBSCAN can efficiently filter out those sparsely distributed samples related to movements between PoIs. Moreover, we perform clustering based on multiple features, which avoids the potential issues related to single feature based clustering described above. Specifically, we first perform DBSCAN based on the feature tuples to discover a set $\mathbf{C}$ of clusters $\mathbf{C} = \{\mathbf{C}_1, \cdots, \mathbf{C}_N\}$ from $\mathbf{F}$. Then for each cluster $\mathbf{C}_j \in \mathbf{C}$, apply DBSCAN again to further divide this cluster into a set of sub-clusters $\mathbf{C}_j = \{\mathbf{C}_{j,1}, \cdots, \mathbf{C}_{j,M}\}$. Note that for most cases, one round of DBSCAN is sufficient, i.e., the second round of DBSCAN will not be able to divide each $\mathbf{C}_j \in \mathbf{C}$ further into multiple smaller clusters. However, the second round of DBSCAN is necessary in some cases. For example, if a mobile user visits places in two different cities, only two clusters (each corresponds to a city) will be returned after the first round because DBSCAN does clustering based on the density of samples. This is obviously too coarse so clustering needs to be done again to improve granularity. After clustering using the DBSCAN algorithm, we can have a set of clusters of feature tuples. However, since DBSCAN is not a centroid-based clustering algorithm, it does not return cluster centers, which is what we need. In addition, we find that if multiple PoIs are close to each other (e.g., multiple PoIs in a plaza), the related feature tuples may be put into the same cluster. So we still need to do fine-grained clustering following the first stage.

In the second stage, we use the MeanShift algorithm on each cluster found in the first stage. In this stage, only locations are used for clustering. The MeanShift algorithm uses a similar idea for clustering but can return cluster centers (if each cluster has a convex shape) [5]. These cluster centers will then be returned as the set of PoIs.

We formally present our PoI discovery algorithm as Algorithm 1, in which the first stage starts from Step 3, and the second stage starts from Step 8. We use c'_i and r'_i to denote the center and the corresponding radius of cluster $\mathbf{C}'_i$ respectively.

Algorithm 1 The PoI Discovery Algorithm

Input: The set of feature tuples $\mathbf{F}$;
Output: The set of PoIs $\mathbf{P}$;

```
1:  $\mathbf{P} \leftarrow \emptyset$;
2:  $\mathbf{C} \leftarrow \emptyset$;
3:  $\{\mathbf{C}_1, \cdots, \mathbf{C}_N\} \leftarrow \texttt{DBSCAN}(\mathbf{F})$;
4:  for $\mathbf{C}_j \in \{\mathbf{C}_1, \cdots, \mathbf{C}_N\}$ do
5:      $\{\mathbf{C}_{j,1}, \cdots, \mathbf{C}_{j,M}\} \leftarrow \texttt{DBSCAN}(\mathbf{C}_j)$;
6:      $\mathbf{C} \leftarrow \mathbf{C} \bigcup \{\mathbf{C}_{j,1}, \cdots, \mathbf{C}_{j,M}\}$;
7:  end for
8:  for $\mathbf{C}_{i,j} \in \mathbf{C}$ do
9:      $\mathbf{F}' \leftarrow \emptyset$;
10:     for $\mathbf{f} = (t, l, a, v) \in \mathbf{C}_{i,j}$ do
11:         $\mathbf{F}' \leftarrow \mathbf{F}' \bigcup \{l\}$;
12:     end for
13:     $\{\mathbf{C}'_1, \cdots, \mathbf{C}'_n\} \leftarrow \texttt{MeanShift}(\mathbf{F}')$;
14:     $\mathbf{P} \leftarrow \mathbf{P} \bigcup \{(c'_1, r'_1), \cdots, (c'_n, r'_n)\}$;
15: end for
16: return $\mathbf{P}$
```

The output of this algorithm is a set $\mathbf{P}$ of PoIs (with center locations and radii). Obviously, these locations cannot be directly used to predict activities of the mobile users. In LIPS, the Place Information Provider uses the Google Place API [8] to find the actual places according to these locations.

B. Lifestyle Modeling for Activity Prediction

In this section, we describe how to predict a mobile user's activities in the next T hours according to the discovered PoIs. Note that PoIs tell us exactly which places the mobile user visited. For a mobile user, PoIs $\mathbf{p}$ and $\mathbf{p}'$ may be two different restaurants he/she usually goes to, but they both correspond to the same activity "dining". So we are actually interested in knowing what kind of activities a mobile user will do in the near future (rather than exactly which place he/she will visit) such that we can provide related and useful information (such as recommendation) to him/her.

In LIPS, we define a set of activities: mall shopping, dining, grocery shopping, outdoor recreation, indoor recreation, movie, gas station, car wash, exercise, laundry, library, and schooling. This set can certainly be expanded according to the new PoIs and needs. In addition, we need to map PoIs to activities. We again use Google Place API to find the type of each PoI. For example, given a hiking trail, it will return its type as "park". In LIPS, We then create a table to map each type to certain activity. For example, if the type of a PoI is "park", its corresponding activity is "outdoor recreation".

In order to build a lifestyle model for future activity prediction. We need to have a training set, in which each item is a *feature-activity* tuple $(\mathbf{f}; \pi)$. $\mathbf{f} = (d, t, l, a, v, w, p, u)$, where d is the day (Monday, Tuesday, etc.), t is the time, l is the location, a is the moving state, v is the moving speed, w is the weather condition, p is the outdoor temperature, u is the user state (active or not); and π is the associated activity. Note that here the feature tuple includes all features discussed in the beginning of this section, which is different from that introduced in the previous section. In addition, we aim to predict the activities in the next T_s to T_e hours so when building the training set, the activities need to be the activities discovered in that future period. For example, suppose a feature tuple of a mobile user, Alice, at 11:00AM is $\mathbf{f}$, and we want to predict her activities in the next 1 to 2 hours (i.e., between 12:00PM and 1:00PM), and her activity during that period turned out be "dining" according to some collected samples, then we will add a feature-activity tuple $(\mathbf{f}, \text{“dining”})$ into the training set. Of course, if there were more than one activities, say "dining" and "indoor recreation" during that period, we will add both $(\mathbf{f}, \text{“dining”})$ and $(\mathbf{f}, \text{“indoor-recreation”})$ into the training set.

After the training set is built, we can apply a supervised classification algorithm [9] to make predictions. We tested a few widely-used algorithms with the collected sensor data and found that Support Vector Machines (SVM) [9] turns out to be the most effective one. Moreover, it is known that SVM is usually very effective in the high-dimensional spaces (many features), fast and memory-efficient. So in LIPS, we employ SVM to predict future activities of a mobile user. We chose to use the Gaussian Radial Basis Function (RBF) kernel [9] for high accuracy in our implementation. SVM can return a model such that when given a feature tuple (as shown above) of a mobile user, it can return the probability of each possible activity he/she may perform in the future.

IV. Lifestyle-aware Adaptive Sampling

On one hand, if the sampling rate is reduced (i.e., sampling period is increased), energy spent for sensor data acquisition and communications can certainly be reduced. Moreover, the mobile phone system will have a much higher chance to enter the sleep mode, which is known to consume much less energy than the active mode does. On the other hand, reducing the sampling rate may lead to less samples, which will have a negative impact on the performance of PoI discovery and activity prediction. Hence, we need to develop an effective algorithm that adaptively adjusts the sampling rate to trade off energy efficiency and learning performance.

We consider the following three cases when we try to make a decision on whether or not to reduce the sampling rate: 1) If the user is at some place, which is not one of discovered PoIs, the sampling rate should not be reduced since otherwise there may not be sufficient samples for discovering this possibly new PoI. 2) If the user is at one of discovered PoIs and the activity prediction is *stable* (explained below), the sampling rate can be reduced. 3) If the user is at one of discovered PoIs, but the prediction result is not stable, the sampling rate should not be reduced.

In our adaptive sampling algorithm, we make sure that the sampling period falls in the range of $[T_{\min}, T_{\max}]$. We set the initial sampling period to $T_{\min}$. $T_{\min}$ and $T_{\max}$ were set to 5min and 20min respectively in our implementation. Every time (say at time t) when a new feature tuple is collected, the algorithm checks whether or not its location l_t falls in the radius of any discovered PoI. If so, the algorithm further employs the developed activity prediction model $\mathcal{M}(\cdot)$ (described above) to predict his/her future activities and stores the results to $\mathbf{\Pi}_t$. Then the algorithm compares $\mathbf{\Pi}_t$ with the previous results to see if there is any significant change. If no, the sampling period is doubled, otherwise it remains the same as before. Here $\pi_t^{\max}$ ($\pi_{t-T}^{\max}$) and $p_t^{\max}$ ($p_{t-T}^{\max}$) denote the most likely activity and the corresponding probability predicted according to the current sample $\mathbf{f}_t$ (the previous sample $\mathbf{f}_{t-T}$), respectively. α is a threshold, which is used to define the condition that triggers adjustment of the sampling period. The larger the α is, the more likely the sampling period will be increased. For all the other cases, the algorithm stays with the minimum sampling period (i.e. 5min in our implementation). We formally present our adaptively sampling algorithm as Algorithm 2. Note that the feature tuple (sample) $\mathbf{f}_{t-T}$ collected at the last sampling instant $t-T$ and the corresponding prediction results $\mathbf{\Pi}_{t-T}$ are given as input.

Note that since the PoI discovery algorithm depends on the density of sensor samples, we need to duplicate sensor samples to maintain the sample density if the sampling rate is reduced by the adaptive sampling algorithm. We only duplicate samples associated with PoIs already discovered therefore the duplication will not affect the discovery of PoIs.

V. Validation and Performance Evaluation

The field tests were conducted with a group of volunteers for over 1.5 months from 6 cities in USA. During the experiments, all the volunteers used Android-based Nexus 4 or Nexus 5 phones. To protect their identities, we use a single capital letter as their names in the following. Sensors were first sampled every 5 minutes. In the last three days of experiments, we started to apply the proposed lifestyle-aware adaptive sampling algorithm to adaptively adjust the sampling rate, and in the meanwhile, we still collected sensor samples every 5 minutes for comparisons.

Algorithm 2 Lifestyle-aware Adaptive Sampling Algorithm

Input: $\mathbf{f}_t$, $\mathbf{f}_{t-T}$, $\mathbf{\Pi}_{t-T}$, T;
Output: T';

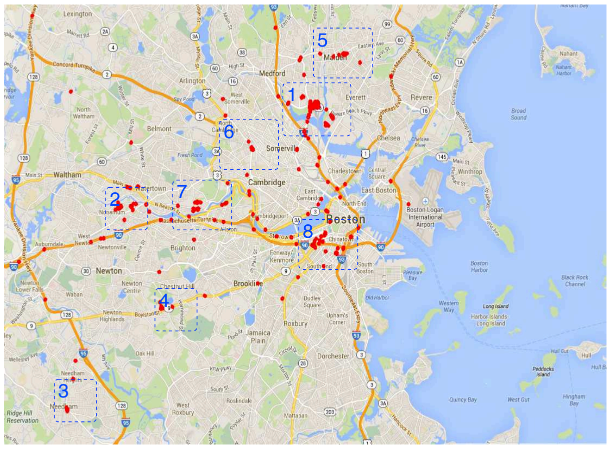
```
1:  if $\mathbf{f}_{t-T} = \text{nil}$ then
2:      return $T_{\min}$;
3:  end if
4:  $T' \leftarrow T_{\min}$;
5:  if $\exists \mathbf{p} = (c, r) \in \mathbf{P}$ s.t. $\|l_t - c\| \leq r$ then
6:      $\mathbf{\Pi}_{t-T} \leftarrow \mathcal{M}(\mathbf{f}_{t-T})$;
7:      $\mathbf{\Pi}_t \leftarrow \mathcal{M}(\mathbf{f}_t)$;
8:      if $\pi_{t-T}^{\max} = \pi_t^{\max}$ and $|p_{t-T}^{\max} - p_t^{\max}| < \alpha * p_{t-T}^{\max}$ then
9:          if $2 * T \leq T_{\max}$ then
10:             $T' \leftarrow 2 * T$;
11:         else
12:             $T' \leftarrow T$;
13:         end if
14:     end if
15: end if
16: return $T'$;
```

A. Validation and Evaluation of Lifestyle Learning

First of all, we present experimental results to validate the proposed PoI discovery algorithm. We used the Google Maps to show the sensor samples and the corresponding PoIs. We conducted interviews with the volunteers and used their descriptions about their lifestyles as ground truths for comparisons. Due to the space limitation, we only present and analyze results of Ms. A. Similar observations can be made for other volunteers.

Ms. A is a businesswoman living in the Great Boston area. Most of her activities happen in the region shown in Fig. 4(a). According to her description, her home is located in area 1. Across the river is a grocery store, where she usually goes for grocery shopping. Area 2 is a small commercial district, where her company and several restaurants are located. On weekdays, she usually leaves home and goes to work in the morning. But occasionally, she needs to meet customers in areas 3 and 4. She likes shopping very much. On weekends, sometimes, she meets her friends and has breakfast together in area 5; sometimes, she meets her friends in Harvard University in area 6. Then they go shopping in areas 7 and 8. There is a large mall in area 7, and area 8 is the downtown of Boston, where a lot of shops and restaurants are located. On weekdays, she usually has lunch in the restaurants close to her company. On weekends, she usually goes to some restaurants in the downtown area.

The periodically collected sensor samples and the PoIs discovered by our algorithm are shown in Fig. 4(a). From


(a) Sensor samples

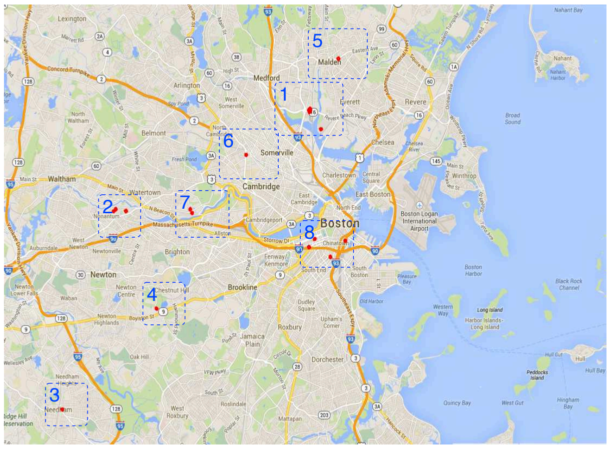

(b) Discovered PoIs

Fig. 4. PoI discovery for Ms. A

Fig. 4(b), we observe that the following places are discovered as PoIs: 1) Ms. A's home and the grocery store near her home in area 1; 2) her workplace and the restaurants that she likes to go to in area 2. 3) the customers' sites in area 3 and 4; 4) Harvard University in area 6; and 5) the shopping malls and restaurants in areas 7 and 8. We can also see that sensor samples related to movements between PoIs are successfully filtered out by the proposed algorithm.

Next, we show the experimental results to justify the effectiveness of the proposed activity prediction algorithm. In the experiments, we aimed to predict activities in the next 1 to 2 hours. To evaluate the activity prediction algorithm, we chose to use the widely used cross-validation method [12]. We split all the training data randomly into 10 disjoint sets. In each test, 9 of these training sets were used for training, and the rest data set was used to test the accuracy of prediction. Hence, a total of 10 tests were performed for each volunteer. We show the results in Fig. 5.

From the figure, we can see the activity prediction algorithm works well. Among all the volunteers, it predicts with an average of accuracy of 72%, the lowest confidence at 56% and the highest at 89%. Moreover, we find out the following

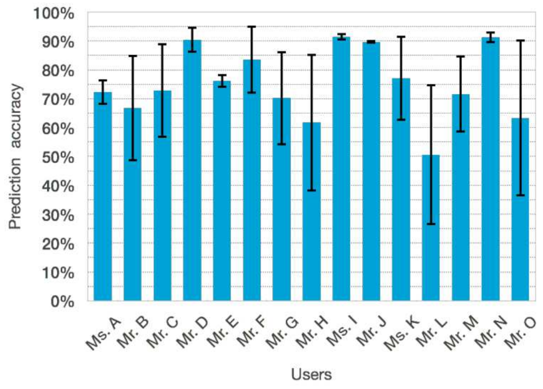


Fig. 5. Cross-validation for prediction accuracy

factors may affect the prediction accuracy: 1) If a mobile user has a very regular schedule, and his/her daily life follows a regular pattern, e.g., a college student, his/her activities can be predicted with high accuracy. Users D, I, J and N fall into this category. 2) If a mobile user has a quite flexible schedule in his/her daily life, it is hard to make accurate predictions. For example, Mr. M is a senior Ph.D student without any course work, so his schedule is quite flexible and his activities are relatively hard to predict. 3) It is hard to predict the activities of a mobile user who travels often. For example, Mr. L is an engineer, who often travels between cities for technical support. The accuracy of prediction for his activities is not as good as that for those who stay in a single city.

B. *Evaluation of Adaptive Sampling*

The proposed adaptive sampling algorithm was applied in the last three days of field tests. The threshold α was set to a relatively small value, 5%, during experiments. In this way, we can save sensing energy, while still preserving good performance of lifestyle learning. Suppose that the number of samples collected by our adaptive sampling algorithm and by periodical sampling (with the sampling period of 5min) are n' and n respectively. We chose to use the ratio $\frac{n-n'}{n}$ as the performance metric, which we call *energy saving ratio*. The corresponding results are shown in Fig. 6. From the figure, we can see that compared to periodic sampling, the proposed adaptive sampling algorithm achieves an energy saving of 52% on average, with the maximum saving at 63% and the minimum at 40%.

VI. Related Work

Comprehensive reviews for mobile phone sensing systems and applications can be founded in [14] and [17].

Research efforts have been made to analyze/presict mobility patterns of mobile users based on locations of their mobile phones. In an early work [10], the authors presented an algorithm called BeaconPrint, which uses WiFi and GSM radio fingerprints collected by someone's personal mobile devices to automatically learn the places they go and then detect when they return to those places. In [6], the authors built a model to identify the structure inherent in daily behaviors by finding

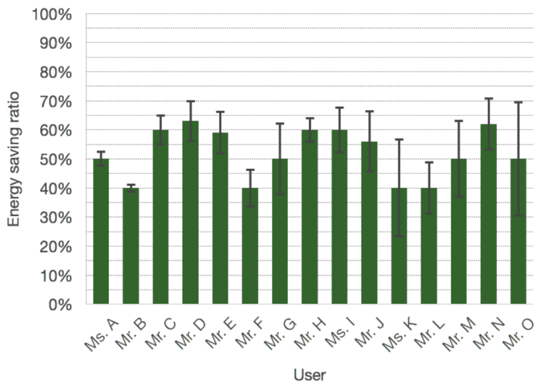


Fig. 6. Energy savings achieved by adaptive sampling

out the principal components (eigenbehaviors) in the data set and representing an individuals behavior over a specific day by a weighted sum of his/her primary eigenbehaviors. More related works along this line can be found in [11], [16], [3], [15]. Different from them, we aim to characterize and predict activities of a mobile user based on not only his/her locations but also many other features (such as weather and moving state) that can be captured by his/her mobile phone. Moreover, we address energy efficiency for data collection with model-based adaptive sampling, which has not been done by them.

Another line of related work is to detect/recognize the moving states of a mobile user using mobile phone sensing. In [20], the authors presented a framework for an Energy Efficient Mobile Sensing System (EEMSS). EEMSS uses hierarchical sensor management strategy to recognize user states as well as to detect state transitions. In [13], the authors described and evaluated a system that uses phone-based accelerometers to perform activity recognition. A rather comprehensive review on this topic can be found in a survey [4]. Unlike these works, we aim at lifestyle learning with moving states of mobile users as a feature, which we obtain using Google's Activity Recognition API [1].

In short, to the best of our knowledge, we are the first to build a mobile phone sensing based system which energy-efficiently collects features to learn and analyze lifestyles of mobile users based on various context information (collected from mobile phones).

VII. Conclusions

In this paper, we presented a system, LIPS, and algorithms to learn lifestyle based on mobile phone sensing. First, we presented the workflow and architecture of LIPS. We proposed a hybrid scheme for lifestyle learning, which consists of two parts: characterization and prediction. Specifically, we presented a two-stage algorithm to characterize the lifestyle of a mobile user using PoIs, which leverages the DBSCAN and MeanShift algorithms for coarse-grained and fine-grained clustering in the first and second stages respectively. Based on discovered PoIs, we developed a method to build a model to predict his/her future activities using SVM. In addition, we presented a lifestyle-aware adaptive sampling algorithm for improving energy efficiency. We implemented the proposed system and algorithms based on the Android platform. We have validated and evaluated LIPS via extensive field tests carried out in 6 major cities of USA. The experimental results showed that LIPS can 1) well discovers PoIs of mobile users, 2) precisely predict their future activities with an average accuracy of 72%, and 3) achieve a significant energy saving of 52% on average (compared to periodic sampling).

References

[1] Recognizing the user's current activity, *http://developer.android.com/training/location/activity-recognition.html*

[2] I. Constandache, R. R. Choudhury, and I. Rhee, Towards mobile phone localization without war-driving, *Proceedings of IEEE Infocom'2010*.

[3] Y. Chon, N. D. Lane, F. Li, H. Cha, and F. Zhao, Automatically characterizing places with opportunistic crowdsensing using smartphones. *Proceedings of UbiComp'2012*, pp. 481–490.

[4] D. Choujaa and N. Dulay, Activity recognition from mobile phone data: state of the art, prospects and open problems, http://www.doc.ic.ac.uk/ nd/papers/MobilePhoneDataActivityRecognition.pdf

[5] D. Comaniciu and P. Meer, Mean shift: a robust approach toward feature space analysis, *IEEE Transactions Pattern Analysis and Machine Intelligence*, Vol.24, Vol. 5, 2002, pp. 603–619.

[6] N. Eagle and A. Pentland, Eigenbehaviors: Identifying structure in routine. *Behavioral Ecology and Sociobiology*, Vol. 63, No. 7, 2009, pp. 1057–1066.

[7] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *Proceedings of ACM SIGKDD'1996*, pp. 226–231.

[8] Google Places API, *https://developers.google.com/places/*

[9] J. Han, M. Kamber and J. Pei, Data Mining: Concepts and Techniques (3rd Ed.) *Morgan Kaufmann Publishers*, 2011.

[10] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, and J. Hughes, Learning and recognizing the places we go, *Proceedings of UbiComp'2005*, pp. 159–176.

[11] S. Isaacman, R. Becker, R. Caceres, S. Kobourov, M. Martonosi, J. Rowland, and A. Varshavsky, Identifying important places in peoples lives from cellular network data, *Proceedings of Pervasive Computing 2011*, pp. 133–151.

[12] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, *Proceedings of IJCAI'1995*, pp. 1137–1145.

[13] J. R. Kwapisz, G. M. Weiss and S. A. Moore, Activity recognition using cell phone accelerometers, *ACM SIGKDD Explorations*, Vo. 12, No.2, pp. 74–82.

[14] N. D. Lane *et al.* , A survey of mobile phone sensing, *IEEE Communications Magazine*, Vol. 48, No. 9, 2010, pp. 140–150.

[15] J. McInerney, S. Stein, A. Rogers and N. R. Jennings, Breaking the habit: measuring and predicting departures from routine in individual human mobility, *Pervasive and Mobile Computing*, Vol. 9, No. 6, 2013, pp. 808-822.

[16] S. Scellato, M. Musolesi, C. Mascolo, V. Latora and A.T. Campbell, Nextplace: a spatio-temporal prediction framework for pervasive systems, *Proceedings of Pervasive Computing 2011*, pp. 152–169.

[17] X. Sheng, J. Tang, X. Xiao and G. Xue, Sensing as a Service: challenges, solutions and future directions, *IEEE Sensors Journal*, Vol. 13, No. 10, pp. 3733–3741.

[18] X. Sheng, J. Tang, J. Wang, C. Gao and G. Xue, SOR: An Objective Ranking System Based on Mobile Phone Sensing, *IEEE ICDCS'2014*, pp. 114-123.

[19] Sensordrone, *http://sensorcon.com/sensordrone/*

[20] Yi Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari and N. Sadeh, A framework of energy efficient mobile sensing for automatic user state recognition. *Proceedings of ACM Mobisys'09*, pp. 179–192.

[21] Weather Channel API, *http://www.wunderground.com/weather/api/?ref=twc*