Wi-Fi Hotspot Auto-Discovery: A Practical & Energy-Aware System for Smart Objects using Cellular Signals

Nithyananthan Poosamani
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
npoosam@ncsu.edu

Injong Rhee
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
rhee@ncsu.edu

ABSTRACT

The Internet of Things (IoT) paradigm aims to interconnect a variety of heterogeneous Smart Objects (SO) using energy-efficient methodologies and standard communication protocols. A majority of consumer devices sold today come equipped with wireless LAN and cellular technology to connect with the world-wide network. To discover Wi-Fi hot spots, there is a need for constant scanning of Wi-Fi radio in these devices and results in significant battery drain. We present PRiSM, a practical system to automatically locate Wi-Fi hotspots while Wi-Fi radio is turned off, by using the statistical characteristics of cellular signals. Cellular signals are received at zero extra cost in mobile devices and hence PRiSM is highly energy-efficient. It is a lightweight client-side only implementation and needs no prior knowledge on floor plans or wireless infrastructure. We implement PRiSM on Android-based devices and show up to 96% of energy savings in Wi-Fi sensing operations which is equivalent to saving up to 16% of total battery capacity, together with an average prediction accuracy of up to 98%.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; H.3.4 [**Information Storage and Retrieval**]: Systems and Software

General Terms

Design, Experimentation, Algorithm, Performance

Keywords

Wi-Fi Sensing, Cellular Signals, Smart Objects, Location Fingerprinting, Energy Efficiency

1. INTRODUCTION

The Internet of Things (IoT) paradigm aims to interconnect a variety of heterogeneous Smart Objects (e.g., sensors, smart devices, home automation equipment) using Machine-to-Machine communications. A majority of consumer devices sold today come equipped with a variety of wireless communication technology (e.g., Bluetooth Low Energy, Wireless LAN, Near Field Communication, Cellular) to connect with the world-wide network. With billions of such devices predicted to connect with the future internet, bringing the physical data from these devices to the digital world requires energy-efficient methodologies and standard communication protocols. Devices that provide health monitoring, smart home and workplace, enterprise data security, and many others need to constantly sense their context and communicate with the network to collaborate with others. But to connect to a hotspot, there is a *need for constant scanning of Wi-Fi access points (APs)* in these devices and results in undesired battery drain. To design an accurate and an energy-efficient Wi-Fi sensing system is (still) a very non-trivial task. The reasons include: almost 60% of battery drain in smart devices result from Wi-Fi [2], not all public Wi-Fi hotspots offer good connectivity and leads to poor user experience [5], frequent disconnection and re-association events with APs incur high energy costs than normal.

Prior works used optimal scanning intervals for Wi-Fi to identify hotspots [16,30]. The scanning intervals are increased or decreased based on parameters like AP inter-arrival time, AP density and user velocity. Since the Wi-Fi connectivity times and movement patterns vary among users, these methods do not adapt well for all users. Wi-Fi signal fingerprinting techniques [7, 33] use extensive offline pre-processing stage to construct signal strength models and to calibrate the radio maps. Also, the techniques take more time to converge. Multi-modal sensing techniques (e.g., Accelerometers [16], GPS [11, 13, 21], Bluetooth [6], Zigbee [35]) are also developed to identify context. Few others use average received signal strengths from connected cellular base stations to predict user location [12, 27, 31]. However, averaging the signal strength values results in loss of granularity and use of additional sensors consume significant extra battery energy (e.g., Accelerometers consume close to $0.667\,mWh$ every $30\,sec$ [22]). Some require infrastructural changes and extensive war-driving efforts to obtain feature-rich data sets. Also, most commercial systems (e.g., WiFi Sense [4], Place Lab [18]) turn on the radio interfaces continuously to identify context which results in excessive battery drain where Wi-Fi scan/association is observed to have high initial costs [9,24]. Upon observing the existing solutions, we sense a need for a new system which is light-weight and does not require extensive data pre-processing. It should consume minimal battery energy, provide ways to continuously accommodate the signal fluctuations, and be easily deployable in real world. Thus the question we ask ourselves is, *"How can we maximally discover Wi-Fi APs in a practical and energy-efficient way with zero extra sensing costs?"*. Given that the Wi-Fi scanning and transmission incur the same energy [31], this question draws more attention.

MOBIQUITOUS 2015, July 22-24, Coimbra, Portugal

DOI 10.4108/eai.22-7-2015.2260071

We develop a new Wi-Fi detection system, PRiSM (Practical and Resource-aware Information Sensing Methodology), which utilizes the freely available cellular signal information to statistically map the Wi-Fi APs with a logical location information. The signal strengths from these base stations are recorded however, the geographical coordinate location of these base stations is not required. A Wi-Fi *signature* is defined as the set of probability density functions (PDFs) of signal strengths from all connected and neighbor Base Stations (BS) when the smart phone is associated with that unique Wi-Fi AP. PRiSM runs in the background and reads cellular signals based on a scheduling policy and hence consumes minimal energy overhead. We use a novel technique to dynamically build and update the signature clusters in near real-time and thus avoid the need for an extensive training phase. We develop a specialized statistical matching algorithm which uses a likelihood estimation technique to automatically tune the decision thresholds for every signature. The threshold values are tuned by connecting to APs and comparing against the ground truth values (i.e., AP available, unavailable). We also implement a novel selective-channel Wi-Fi scanning framework to automatically connect to the APs without scanning or association by utilizing their stored frequency channel information. The empirically constructed signal distributions and decision thresholds for a Wi-Fi location can be adapted or learned as time evolves. We implement PRiSM on Android devices and perform both trace-based simulation and practical evaluation. PRiSM obtains up to 96% of energy savings in Wi-Fi sensing operations equivalent to saving up to 16% of total battery capacity, together with an average prediction accuracy of up to 98%. We discuss design considerations in § 2, implementation in § 3, evaluation results in § 4, related research works in § 5, possible improvements in § 6 and finally conclude in § 7.

2. DESIGN

2.1 Wi-Fi Power Consumption

In a smart phone, a Wi-Fi scan is initiated in response to two actions: by turning on the screen or when an application specifically requests for a scan. When an AP is available to connect, the Wi-Fi driver scans the available channels and connects to the pre-configured AP as shown in Figure 1 (a). If no such AP is found in the pre-configured list, it periodically scans until the device is successfully connected to an AP or until a connection time-out occurs in the Wi-Fi driver after 15 mins. The default time interval for consecutive scans vary between 5-30 sec in various *wpa_supplicant* implementations. Upon screen off, the Wi-Fi radio chipset is turned off after a delay of 2 mins to avoid race conditions in the driver. CPU Wake locks are obtained for operations during screen off. While in connected state, if the link quality deteriorates, the Wi-Fi radio driver is kept in high power state constantly due to repeated scan and association requests. Also to avoid packet loss, the driver operates at lower modulation rates. Our measurements using a power monitor show the repeated scan/association operations in Figure 2. When there is no AP available to connect, the Wi-Fi radio driver scans continuously and results in energy wastage (Figure 1 (b)). The energy consumed by the Wi-Fi radio under various screen conditions and AP availability conditions is shown in Figure 3. Thus, PRiSM can save substantial energy by intelligently avoiding poor and no Wi-Fi conditions in an accurate manner.

2.2 Cellular Signal Signatures

We investigate the feasibility of constructing a database using the statistical information of cellular signals for each Wi-Fi AP and the ability to distinctly identify the APs in the database based on their signatures. Cellular signals are ubiquitous in nature and are received continuously by the phones. A smart phone can receive signals from more than ten base stations (BSs) in dense urban areas [17]. GSM based Android phones can overhear signals from up to seven (six neighbouring and one connected) BSs in ASU (Active Set Updates) units at any time instant. The linear equation between dBm and ASU values for GSM networks is $dBm = 2ASU - 113$.

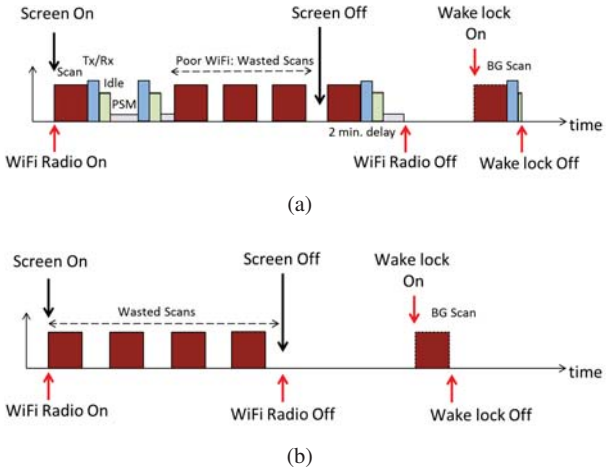


Figure 1: Working of default Wi-Fi when (a) an AP is available to connect with, and (b) an AP is not available.

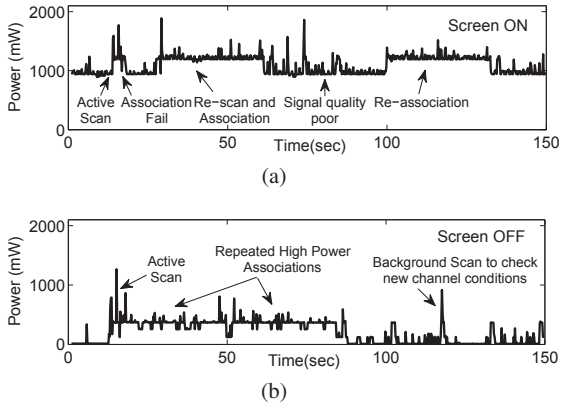


Figure 2: Repeated scan/association events under poor AP signal when the device screen is (a) ON, (b) OFF.

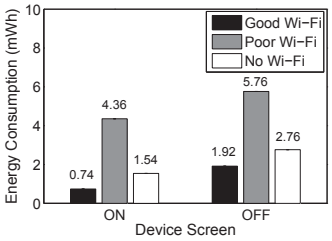
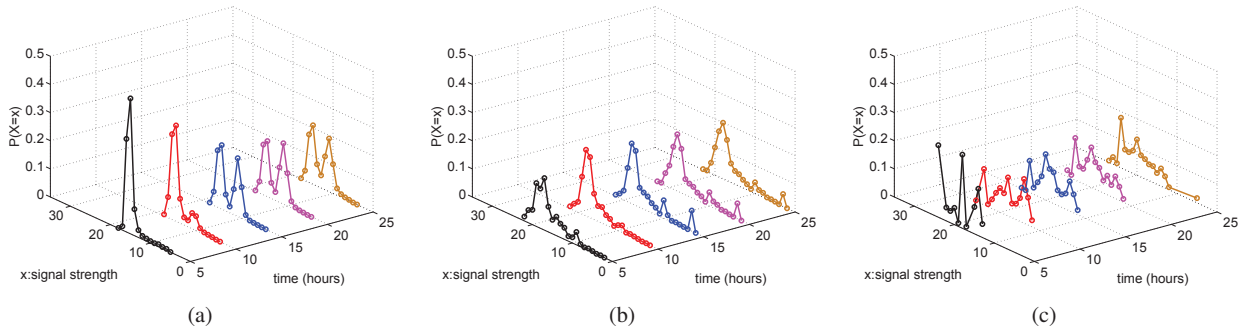
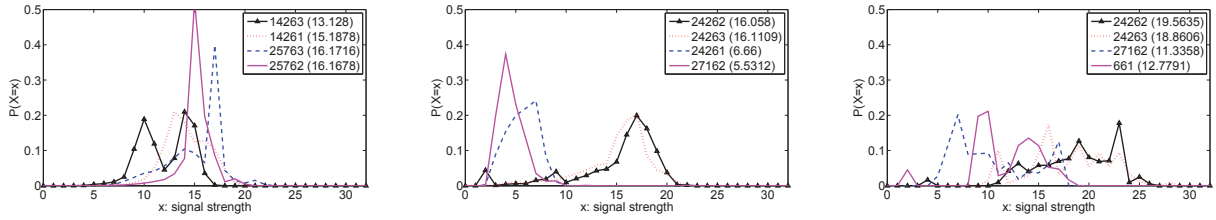


Figure 3: Default Wi-Fi energy consumption for one minute under various screen activation conditions.



(a) (b) (c)

Figure 4: The evolution of signal strength distributions from the most frequently connected base station for 3 different APs are depicted in (a), (b), and (c). For each AP, the data is aggregated over time whenever connected with the AP.



(a) (b) (c)

Figure 5: The personalized signatures for three APs: (a) AP_X (b) AP_Y, and (c) AP_Z. The distance between AP_X and AP_Y is about 7 km, AP_Y and AP_Z is about 30 $meters$. AP_Y and AP_Z are located in the same building. The observed base station IDs and their average signal strengths are given in the legend.

ASU values range from 0 to 31 and 99, which indicates unknown signal strength. The total time interval of observation of every base station within the signature differs and depends both on the total time spent by the user while connected to the particular Wi-Fi and also on the occurrence pattern of the base station.

To capture the entire signal characteristics that a user uniquely experiences for an AP, we propose to build cellular signal signatures using "probability distributions" of signal strengths from all observable connected and neighbor base stations rather than using abstracted information (e.g., "average signal strengths"). We performed the statistical measurements for all users in our dataset, but for explanation purposes, we will take random users to show the following results. Figure 4 shows the evolution of signatures recorded by a user over time for three Wi-Fi APs to which the user has connected most frequently. For better readability, we plotted only the signal strength distribution from the most frequently connected BS per Wi-Fi AP. The figure shows the PDF of signal strengths received from the connected BS at different intervals of time. Simply put, the distribution shown after 10 hrs includes the data used for the distribution shown at 5 hrs plus five more hours. Note that the signal strength distributions do not converge to a Gaussian distribution even after 25 hrs of signal accumulation. Hence, we develop a *non-parametric algorithm* which does not assume anything about the underlying data distribution. The correlation coefficient (ρ_{X_1,X_2}) between probability distributions accumulating signals for different amounts of time clarifies the existence of characteristic patterns in the signatures. High value of correlation coefficients for signatures after 25 hrs of signal accumulation and low cross-correlation values indicate that our statistical technique is likely to provide good performance in matching accuracy.

Figure 5 further shows that the signatures recorded by a user for different APs located far from or near to each other have significant dissimilarities. We again choose three Wi-Fi APs: AP_X, AP_Y, and AP_Z from a user's database, where distances between AP_X and AP_Y is about 7 km and between AP_Y and AP_Z is about 30 meters (AP_Y and AP_Z are in the same building). In the figures, base station IDs and their average signal strengths are given in the legend. As expected, the signatures for AP_X and AP_Y contain completely different sets of BSs and different patterns of signal distributions. On the other hand, the signatures for AP_Y and AP_Z show similar sets of BSs. However, they are still distinguishable because the signal distributions show unique patterns. Considering the possible differences in the environment and the behaviour of a user, observing dissimilar signal distributions even for nearby APs is not surprising and actually helps to identify the APs more reliably.

2.3 Proposed Algorithm: ATiS

We design an algorithm that can utilize detailed statistical properties of cellular signals instead of the averaged signal strength values. A simplified version of ATiS (Automatically Tuned Location Sensing) is explained in Algorithm 1. Since the entire signal distribution is available, ATiS predicts the location in near real-time. A higher level intuition of the algorithm is that if the probability of seeing a particular signal strength within the PDF of a base station (BS) is high and the probability of the BS observed when connected to an AP is high, the total joint distribution is maximized

Algorithm 1: **ATiS Signature Score Generation**

INPUT: Signature database for all Wi-Fi APs connected by the user
INPUT: Set of currently observed BSs and their corresponding signal strengths at time t
INPUT: Hashmap of unique Wi-Fi APs and reverse Hashmap of observed BS IDs to APs for fast lookup
OUTPUT: List of Wi-Fi APs in descending order of likelihood

Step 1. *For given input BS, look-up the reverse Hashmap to identify the signature cluster subset to reduce computation*
Step 2. *Calculate the score for the individual signatures*
for $all\ signatures\ in\ cluster\ subset$ **do**
  for $all\ Base\ Station\ ID's\ within\ signature$ **do**
    if $Base\ Station\ ID\ exists\ in\ input\ at\ time(t)$ **then**
      if $Requested\ signal\ strength\ bin\ is\ Empty$ **then**
        Normalize 'x' adjacent bins
      end if
      Evaluate likelihood of occurrence using expectation maximization technique
    end if
  end for
  $Accumulate\ final\ likelihood\ scores\ for\ all\ signatures$
end for
Step 3. *Apply the lower and upper bound thresholds ($[C_L, C_U]$) on generated scores*
Step 4. *Return Wi-Fi APs which satisfy the thresholds*
Step 5. *Check with the ground truth and update the signature thresholds if needed*

and we get a more accurate signature match. ATiS utilizes a set of signatures (P) each consisting of a set of base stations R_j and corresponding signal strength distributions $f_{k,j}(S)$, where $k \in R_j$ and $j \in P$. Note that j and k are signature ID's (e.g., Wi-Fi AP) and cellular base station ID's respectively. Each signature P has information pertaining to the number of occurrences made by its individual base stations in $n(k,j)$ and the total occurrences of all its base stations collectively in N_j. At any time interval $t \in [t_1, t_2]$ during the testing phase, the signal observed from a particular base station k is measured to be $s_k(t)$. For any signature j which has observed this particular unique base station over the course of its training time period, the likelihood of occurrence of the currently observed signals from the base station k is calculated as $v(k,j) = (\prod_{i=t_1}^{t_2} f_{k,j}(S = s_i(k)))$. Similarly, the likelihood is calculated for every base station k_1, k_2, ..k_n which is observed during the time frame of measurement t and which matches within the signature database. The overall maximum likelihood score of simultaneous occurrence of all such base stations within a particular signature j is then calculated as $s(j) = \left(\prod_{k \in P(j)} v(k,j)\right)$.

For any input BS, ATiS does a *local normalization* of signal strength values surrounding the target signal strength in the database and hence, performs well even under signal fluctuations. For example, assume in the signature database, a Wi-Fi location has recorded signal distribution for a BS having signal strength values only for *asu* values 17, 18, 20, and 21 out of the possible 0 to 31 values. During testing phase, if the input signal strength for the same BS is 19, ATiS does not mark the probability of finding the signal strength 19 as zero, instead, it normalizes the values of signal strength bins 18, 19 and 20. If all the requisite bins (here 18, 19 and 20) are empty, ATiS normalizes the expected value to be $1/|n(k,j)|$. ATiS pre-emptively calculates the value ahead of database update because after the current estimation time period t, the observed signals for this particular base station will be updated in the database. Hence, ATiS can perform well even under slight signal variation conditions. The closer the match of input base stations within a signature, the better is the score for the Wi-Fi. All signatures whose likelihood scores $s(j)$ satisfy the lower bound (C_L) and upper bound (C_U) thresholds are returned as output in descending order of their scores. Note that the values of $[C_L, C_U]$ are initialized with $[1, 0]$ initially and are decreased or increased over time to achieve a tight threshold range. The novel part of ATiS is that it auto-tunes thresholds within $0-1$ based on likelihood scores by checking the ground-truth (i.e. Wi-Fi AP (un)available) after each connection attempt and hence, does not overfit the data for any particular scenario. Also, by design, PRiSM utilizes a *cluster-reduction approach* to only compare the currently received signals with a small subset of the signatures in the database irrespective of the total database size and saves on computation time to compare from all the signatures otherwise.

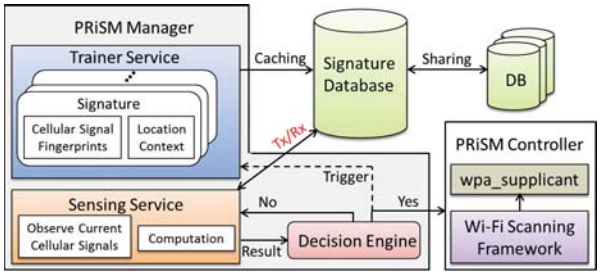


Figure 6: PRiSM system architecture.

3. IMPLEMENTATION

3.1 PRiSM Architecture

The primary modules of PRiSM as shown in Figure 6 include: *PRiSM Manager* at the application layer and *PRiSM Controller* at the platform layer of the Android stack. The manager runs in the system background and builds a list of unique signatures (inside the phone for privacy) for all connected Wi-Fi APs through the trainer service. The sensing service overhears the cellular signals at programmed time intervals to predict AP availability. The decision engine ranks the scores from the ATiS score generator algorithm and outputs the result. The controller implements a novel selective-channel Wi-Fi scanning framework to connect to APs directly without scanning or association via *wpa_supplicant* module in the phone system. It uses appropriate frequency channel information of APs stored in the database. The existing configuration file *wpa_supplicant.conf* is intelligently modified at runtime to provide access to the manager and the controller simultaneously. Hence, PRiSM can serve as a middleware for all Location Based Service (LBS) applications in the smart phone. PRiSM suppresses Wi-Fi connection to an AP in poor signal strength regions and when the user moves closer to the same AP, it automatically matches the good signature of the AP and connects to it.

3.2 PRiSM Operation

The three important tasks performed by PRiSM is shown in Figure 7 and they include: bootstrapping, signature matching, and online training. *Bootstrapping* is the first process when a signature database is created for every user for the first time. Here, an *event* represents the process of connecting to a Wi-Fi AP. Since most people show regular movement patterns on a weekly basis [14], the signatures are continuously updated as time evolves but most signatures get stabilized quickly within a week. The process of computing the likelihood score for an AP from all matching signatures and

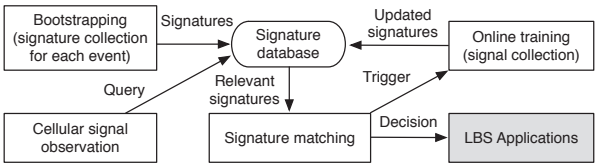


Figure 7: PRiSM operation includes three tasks: bootstrapping, signature matching, and online training.

threshold parameters is called as *Signature Matching*. The decision engine notifies the Wi-Fi on/off decision along with the AP channel information to the Wi-Fi controller within a sub-second time period. 'LBS' (Location Based Service) applications, though not a main part of PRiSM operation, is shown here (shaded in Figure 7) since PRiSM also can serve as a middleware for all such applications in the smart phone. Only upon successful connection to an AP, we enter *Online Training* through which the signature database is kept up-to-date. It is done to capture environmental changes such as configuration updates in an AP, changes in indoor signal propagation paths and behavioural changes in the user. PRiSM suppresses Wi-Fi connection to an AP in poor signal strength regions and when the user moves closer to the same AP, it automatically matches the good signature of the AP and connects to it.

When PRiSM predicts an AP, it tries to connect to the AP even without scanning. If the ground truth (checked by connecting to the AP after every prediction) has an AP (i.e., *true positive*), the connection attempt becomes successful and hence reduces the time to connect to an AP by 33.7%. If the ground truth has no AP (i.e., *false positive*), the connection attempt will be unsuccessful and it auto-tunes the threshold parameters. PRiSM predicts no AP under two conditions: *Zero Match* (i.e., overheard BS ID's do not match with any stored Wi-Fi signature) and *Threshold Mismatch* (i.e., overheard BS ID's matched with some Wi-Fi signature but failed to satisfy the threshold parameters). In the case of zero match, PRiSM assumes the user is in a new place and scans all channels once to provide the results to the user. Here, it simultaneously aids for user experience and reduces energy on repeated scans until the user decides to connect to any AP. In the case of threshold mismatch, it first scans only those channels associated with its known list of APs in the database. If the scan results match with an AP in the database (i.e., *false negative*), it connects with the AP and simultaneously tunes its threshold parameters and hence saves energy instead of scanning all channels. If no match is found (i.e., *true negative*), PRiSM stops further scans and turns off the Wi-Fi interface to save energy from excessive unnecessary scans.

3.3 PRiSM Cost

Cellular signals are received and processed all the time by the phone MODEM at no extra cost. PRiSM activates the CPU only to read cellular signal values from the MODEM and to compute using ATiS. At all other times, CPU is not activated by PRiSM and consumes negligible energy ($0.6 - 1.1\,\mu Wh$) on top of CPU base energy. The sampling policy is shown in Table 1. The overall energy costs for continuous Wi-Fi sensing using PRiSM is minimal when compared to normal Wi-Fi scan. Using the reverse hashmap, the signatures are computed only for the MACs with current observed BS IDs. Hence, PRiSM only compares the currently received signals with a small subset of signatures in the database irrespective of the total database size and saves on computation time. Thus the space and time complexity needed for computation is a function of the density of APs in the nearby environment and is almost constant. In our traces, the signature comparisons never exceeded 35 even though some users had up to 337 unique signatures stored in their database. Hence, PRiSM is more robust to handle database explosion.

Table 1: PRiSM cellular signal sampling policy.

Screen	Wi-Fi State	
	Disconnected	Connected
ON	1 sample every 20 sec	20 contiguous samples every 60 sec
OFF	1 sample every 20 sec	1 sample every 60 sec

Table 2: Dataset information.

Dataset	# of Volunteers	Total hours	Avg. Wi-Fi %
D1	24	2592	89.6
D2	16	1440	81.3

4. EVALUATION

4.1 Datasets

We obtained Institutional Review Board (IRB) approval from North Carolina State University to gather datasets (Table 2) from Android based devices running our customized monitoring application. Data was collected for over two weeks from graduate students (29), undergraduate students (6), and employees (5). Undergraduate students predominantly covered locations within the campus. Graduate students had both on-campus and off-campus locations. Each employee data is from a different urban city in the US. Dataset 'D1' is obtained from our lab Nexus One phones used by volunteers as their primary device. It includes timestamp, Wi-Fi signal statistics for connected and neighbor APs, screen unlock info, and cellular signal statistics for connected and neighbor BSs. Dataset 'D2' is obtained from personal phones of volunteers due to non availability of test phones in large numbers. It includes screen on/off information in addition to screen unlock information present in 'D1' but lacks neighbor BS information due to the closed nature of GSM API found in those phones. In both datasets, cellular signal and screen information are recorded at each second and Wi-Fi information at each minute. Since fine-grained screen activity information is required to accurately predict energy savings, we use 'D1' to analyze the algorithm accuracy and apply those parameters (false positives, false negatives, etc) to 'D2' to predict energy savings. The devices recorded up to 35 APs in some campus locations. Also, the students recorded higher number of signatures for unique APs (up to 337) than the employees due to their movement patterns and the number of unique locations visited throughout the data collection period.

4.2 Accuracy Measurements

A trace-driven simulator builds the signatures and evaluates the accuracy of the algorithms by checking with the ground truth values in the dataset. The robustness of an algorithm depends on the proportion of true positives and true negatives correctly identified. In Figures 8 (a) and (b), the diagonal line represents the random prediction of an algorithm, points above and below the diagonal represent good and bad prediction accuracy. ATiS obtains higher percentage of true positives and true negatives compared to other algorithms because ATiS uses the entire signal distribution from BSs and auto-tunes its threshold parameters as time evolves by adjusting itself to signal variations. However, other class of algo-

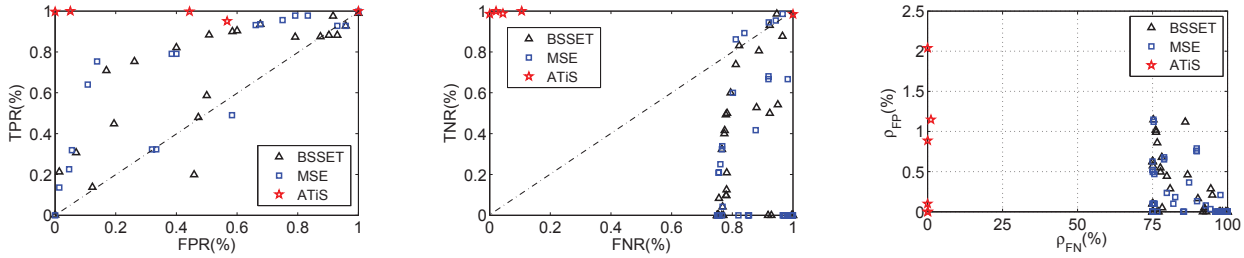


Figure 8: (a, b) ROC curves and (c) ρ_{FP} Vs. ρ_{FN} values for a randomly selected user for all algorithms in dataset 'D1'. ATiS achieves very high true positive and true negative values and very low ρ_{FP} and ρ_{FN} values simultaneously.

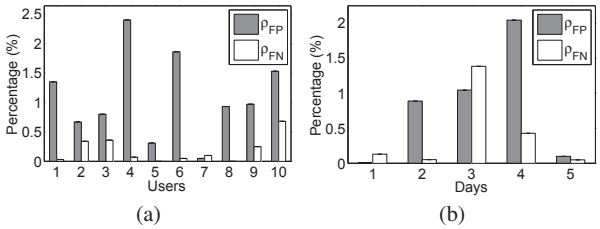


Figure 9: (a) Average ρ_{FP} and ρ_{FN} for users in dataset 'D1' and (b) ρ_{FP} and ρ_{FN} for 5 consecutive days for a user.

Table 3: Fine-grained measurements for Wi-Fi sensing.

Item	Energy Consumption (mWh)			
	HTC Nexus One		Samsung Galaxy S5	
Screen	On	Off	On	Off
Wi-Fi Radio Up	0.0943	0.1181	0.2528	0.3164
Wi-Fi Radio Down	0.0405	0.0606	0.0510	0.2993
Scan	0.1376	0.1955	0.5333	0.5811
Auth/Assoc	0.1588	0.2711	0.2570	1.4481
PRiSM Active	0.0019	0.0173	0.0015	0.0012
Wakelock	NA	0.0241	NA	0.0527
CPU Normal	0.2706	0.0059	0.0871	0.0032

rithms (BSSET and MSE are discussed in § 5.3) use average signal strength values from BS and persistent threshold values which either overfit or underfit the data. Though the results are shown from a single user for clarity, we observed a similar pattern across all users in the dataset.

False positive ratio (ρ_{FP}) is defined as the number of cases that an algorithm detects an AP when there is no such AP in the ground truth divided by the total number of cases. Similarly, false negative ratio (ρ_{FN}) is defined as the number of cases that an algorithm detects no AP when there is an AP in the ground truth divided by the total number of cases. Higher ρ_{FP} indicates losing more chances for energy saving and higher ρ_{FN} indicates losing more connection opportunities. Figure 8 (c) shows that BSSET and MSE class of algorithms require very high threshold values to achieve lower ρ_{FP} values, which results in undesired higher ρ_{FN} values. ATiS achieves lower ρ_{FP} and ρ_{FN} values simultaneously and hence results in minimum lost opportunities for connection with maximum energy saving. Figure 9 (a) shows the variation in mean ρ_{FP} and ρ_{FN} values for individual users over their entire evaluation period suggesting the difference in their mobility patterns and the places they visit. The overall ρ_{FP} and ρ_{FN} values for all the users in the dataset 'D1' averaged to $1.10\,\%$ and $0.19\,\%$, which is very close to *zero* (ideal value). Since PRiSM starts predictions from day-1, we show the variation in the values after each day in Figure 9 (b). Hence, even with small number of samples in acquired data during the initial few days, ATiS keeps the false positives and false negatives low and it further improves as time evolves.

4.3 Energy Measurements

4.3.1 *Measurement setup and energy calculations*

Energy measurements are obtained from Monsoon power monitor [3] with values recorded every $200\,\mu s$. For practical purposes, we avoid using mobile power monitors as in [27]. Since PRiSM modifies default Wi-Fi connection framework, we obtain fine-grained energy information for important Wi-Fi processes as shown in Table 3. The Wi-Fi measurements are obtained by subtracting the background energy (which includes CPU, LCD, and backlight) from total consumed energy. Extensive trials are performed using an automated program to avoid finger touch events on the LCD screen and to avoid sensitive fluctuations in power consumption. We also remove all background processes and turn off other sensors not associated with the Wi-Fi to avoid energy variations. For trace-based simulation, we first extract Wi-Fi event information (e.g., radio-enable, scan, authentication) for various screen activity conditions recorded in the dataset and combine with practical usage values in Table 3 to accurately calculate the total energy consumption by Wi-Fi usage specific to each particular user for each day.

4.3.2 *Default Wi-Fi Vs. Footprint Vs. PRiSM*

In Figure 10, we compare the energy consumed by PRiSM with three other Wi-Fi sensing systems: *Default Wi-Fi* refers to Wi-Fi in off-the-shelf phones, *Footprint* refers to the Wi-Fi sensing system in [31], and *Ideal* refers to the imaginary oracle sensing system introduced for user clarity. We define the characteristics of an ideal system as: uses zero system/CPU energy to identify Wi-Fi APs, connects automatically to the APs without scanning, and shuts down Wi-Fi radio immediately in places where Wi-Fi is absent. PRiSM implements a full version of a sub-optimal algorithm, *PRiSM-SubOpt*, which scans for APs before connection and a prototype version of an optimal algorithm, *PRiSM-Opt*, which knows AP channel information and connects directly without scanning.

Footprint triggers scan based on distance moved by the user (more than $10\,m$ indoors or $20\,m$ outdoors). In no Wi-Fi areas, Foot-

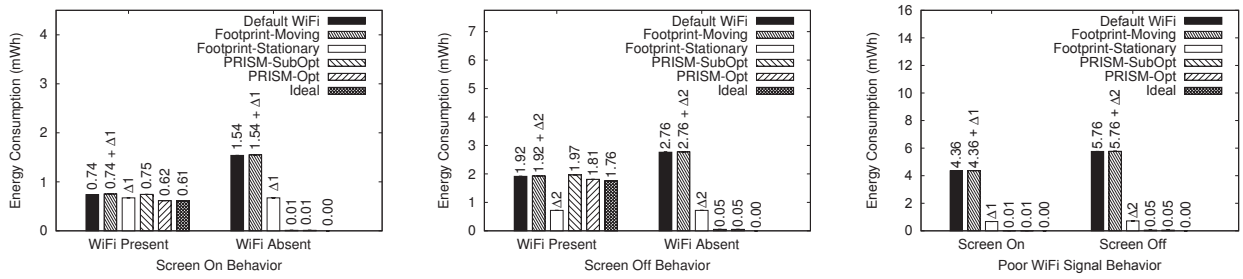


Figure 10: Wi-Fi energy consumed every minute for (a) screen ON, (b) screen OFF, and (c) under poor Wi-Fi signals. For Footprint, $\Delta 1$ is estimated to be $0.673\,mWh$ for screen on and $\Delta 2$ is estimated to be $0.719\,mWh$ for screen off conditions.

print scans for Wi-Fi first and later records all places which do not have Wi-Fi bloating its history list. Even in areas with Wi-Fi, unless the user moves, it does not scan even if the Wi-Fi radio is turned off after screen off delay. PRiSM, however, checks for Wi-Fi availability every sampling period and connects to Wi-Fi if needed, else, it maintains the radio in off state. Also, it connects directly to the AP without scanning and avoids turning on Wi-Fi in poor signal strength areas and hence saves energy intelligently. Since energy measurements for Footprint is not available and implementing the entire system is out of scope in our experiments, we combine the accelerometer energy value ($0.667\,mWh$) obtained in [22] and our own test measurements to sample cellular signals thrice ($0.006\,mWh$ for screen-on and $0.052\,mWh$ for screen-off) to calculate the additional cost incurred by Footprint in both screen on ($\Delta 1$) and off ($\Delta 2$) conditions to be $0.673\,mWh$ and $0.719\,mWh$ per minute. For a *stationary user*, Footprint effectively suppresses Wi-Fi scans in no Wi-Fi areas, but still incurs the overhead energy from accelerometer usage, which is significantly high compared to PRiSM. For a *moving user*, Footprint consumes more energy than default Wi-Fi and PRiSM. When Wi-Fi is available, PRiSM-SubOpt consumes slightly higher energy than default Wi-Fi since it uses extra energy for cellular overhearing. However, PRiSM-Opt always consumes less energy than default Wi-Fi by design. The Ideal system always consumes the lowest possible energy and provides a baseline to compare for the maximum amount of energy that can be saved by any Wi-Fi sensing system.

4.3.3 Sensing intervals (δ) and threshold (τ) effects

The energy consumed by PRiSM and an Ideal system for various sensing intervals (δ) and Wi-Fi thresholds (τ) is discussed here. We do not compare Footprint because of the non-availability of accelerometer values in our dataset. Also, PRiSM does not measure distance from APs or discriminate between indoor and outdoor locations. The energy savings vary between users and depends on their individual mobility patterns and Wi-Fi availability (e.g., Users who often experience poor and no Wi-Fi situations save more energy than users who experience good Wi-Fi. The reason is in good Wi-Fi areas, the only avenue to save energy is to avoid scan costs). We define *battery capacity* as the maximum amount of energy that can be extracted from a smart phone battery and is assumed to be $5000\,mWh$ in our energy calculations.

Sensing interval of δ = 1 sec is equivalent to keeping the Wi-Fi interface continuously ON. When δ increases, the average battery saving for all users combined decreases steadily as shown in Figure 11 (a). The decrease in energy saving from that of 1 sec scanning is because of following reasons: scan is not performed continuously and during the time slots (e.g., δ = 30 sec, 45 sec.. 5 min), only 20 sec of time slot is utilized for sensing operations and the Wi-Fi radio is turned OFF for remaining time. Email sync applications are shown to take close to 18.54 sec [32]. Hence, we assume a constant time of 20 sec for the purpose of evaluation and can be varied if necessary. We see that PRiSM-Opt achieves close to 96% in average battery savings to that of an ideal system. The variation in average battery savings for all users for different thresholding values (τ) is shown in Figure 11 (b). The decrease in savings with smaller thresholds ($\tau = -90$) is due to increased energy usage to connect to Wi-Fi in poor signal areas. Even under no thresholding (τ = $None$), ATiS achieves close to 90% of that achieved by an ideal system. This shows that the huge energy savings of PRiSM are mainly due to the better performance of ATiS algorithm and not just the RSSI thresholding parameter. However, to provide better user experience and also to save on battery energy, PRiSM as a system, uses a default value of $\tau = -80\,dBm$.

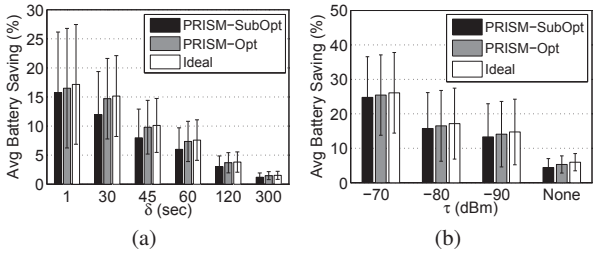


Figure 11: Mean battery savings for all users in the dataset with 95 % confidence interval. (a) vary δ given $\tau = -80$ dBm, (b) vary τ given $\delta = 1$ sec.

4.3.4 Overall energy impact of PRiSM

Wi-Fi sensing measurements in Table 3 show that latest Samsung Galaxy S5 phones consume more energy compared to older Google Nexus One phones because of powerful Wi-Fi chipsets. We infer that in spite of all the commercial advancements made in recent times to reduce the power utilization in Wi-Fi radio's (e.g., better sleep cycles, reduced idle times), scanning for Wi-Fi APs still requires substantial energy. Hence, PRiSM in general can save substantial battery energy in all phones without discrimination. From Table 4, users spend about 30% of battery energy on average for Wi-Fi sensing operations. We observed that about 11.24% of that energy is wasted for Wi-Fi sensing in regions with poor/no Wi-Fi

Table 4: Total Wi-Fi usage and battery savings for users in dataset 'D2' for Wi-Fi offloading with $\tau = -80\,dBm$.

User	Wi-Fi Avail (%)			Wi-Fi Battery	Battery Savings (%)		
	Good	Poor	No	Usage (%)	PRiSM-SubOpt	PRiSM-Opt	Ideal
1	82.92	6.69	10.40	21.79	6.45	7.24	7.79
2	94.62	0.49	4.90	21.26	1.96	2.91	3.57
3	48.60	6.03	45.37	20.15	14.52	14.83	15.24
4	74.56	2.56	22.87	19.95	6.37	7.11	7.63
5	11.73	74.69	13.58	60.08	57.04	57.20	57.83
6	71.12	25.29	3.59	42.24	23.23	24.22	25.00
7	61.82	10.92	27.26	36.11	23.18	23.86	24.62
8	71.52	9.54	18.95	34.65	18.97	19.77	20.57
9	97.71	0.80	1.50	15.29	0.34	1.10	1.57
10	91.36	4.37	4.27	33.39	5.42	6.82	7.74
Avg	**70.59**	**14.14**	**15.27**	**30.49**	**15.75**	**16.51**	**17.16**
% of Ideal achieved					**91.79**	**96.20**	-

combined, which is very significant. On average, PRiSM saves about 16.51% of total battery energy, which is equivalent to saving almost $825.5\,mWh$ worth of energy spent on Wi-Fi if we assume the battery capacity to be $5000\,mWh$. [10] estimates the average battery lifetime[1] of a smart phone to be 40 hrs and 27 hrs for casual and regular usage respectively. Using this result, we observe that PRiSM on average can extend the battery lifetime by 6.6 hrs and 4.5 hrs for casual and regular phone usage respectively. Given that about 70% of users in our dataset travelled in 'Good Wi-Fi' areas, energy savings for PRiSM will be much higher if users had high mobility patterns in 'No Wi-Fi' and 'Poor Wi-Fi' areas, which happens more often in practice.

4.3.5 *Practical verification of energy savings*

We identified test phones with similar battery aging by comparing the amount of time it took them for a full battery discharge with bare-essential Android system processes. Practical verification of energy savings shown in Table 5 involved two phones: one normal phone and other running PRiSM-SubOpt. A mock application was installed on both phones to check for Wi-Fi at different sensing intervals. The application just connects to the AP and no data transmission is done since energy consumption may change with different data transfer rates even to same AP's at a particular time instant. For an RSSI threshold setting of $\tau = None$, the sensing intervals for $30\,sec$, $60\,sec$, and $120\,sec$ saw average Wi-Fi availability of $55.20\,\%$, $77.32\,\%$, and $0\,\%$ respectively. This Wi-Fi availability is calculated by comparing the Wi-Fi connectivity information recorded from user logs and the signature database file given to the users for test. For $\delta = 30\,sec$, PRiSM obtained huge energy savings in no-WiFi areas and incurred minimal energy overhead in areas with Wi-Fi. For $\delta = 60\,sec$, PRiSM should have had more battery left at the end since the sensing intervals are less frequent but recall that PRiSM only saves energy on scan costs in areas with good Wi-Fi. Given that users saw an average of $77.32\,\%$ Wi-Fi, only about $9\,\%$ of battery remained. For $\delta = 120\,sec$, we specifically tested the scenario where user visits totally new places (i.e., with zero stored information about the location in database and hence $0\,\%$ Wi-Fi availability to connect). Hence, the energy savings by PRiSM is more in this scenario than previous sensing intervals even though the Wi-Fi sensing operations are less frequent at every $120\,sec$. The results show that in all cases, PRiSM-SubOpt had substantial amount of battery left when the normal phone is completely discharged. We believe that PRiSM-Opt will save even more energy in these situations.

[1]In this paper, 'battery lifetime' refers to the operating time of the battery from one full charge to full discharge.

Table 5: Nexus One practical energy evaluation.

System	Wi-Fi Sensing Interval (δ)		
	$30\,sec$	$60\,sec$	$120\,sec$
Normal	lasted $14.5\,hrs$	lasted $24.3\,hrs$	lasted $33.0\,hrs$
PRiSM	had $54\,\%$ left	had $9\,\%$ left	had $65\,\%$ left

5. RELATED WORK

5.1 Wi-Fi Sensing and Power Reduction

Wi-Fi hotspots are scarcely distributed when compared to cellular networks [25]. Hence, prior works developed optimal scanning intervals for Wi-Fi to identify hotspots [16, 30]. The scanning intervals are increased or decreased based on parameters like AP inter-arrival time, AP density and user velocity. Since the Wi-Fi connectivity times and movement patterns vary among users, these algorithms do not adapt well for all users. Wi-Fi signal fingerprinting techniques [7, 33] use extensive offline pre-processing stage to construct signal strength models and to calibrate the radio maps. Also, these techniques take more time to converge. PRiSM does not assume anything about the underlying data model or distribution and hence takes a non-parametric approach.

Few others use multi-modal sensors (e.g., Accelerometers [16], GPS [11, 13, 21], Bluetooth [6], Zigbee [35]) in addition to identify context. Unlike PRiSM, the sensors may not be available always and they consume extra battery energy (e.g., Accelerometers consume close to $0.667\,mWh$ every $30\,sec$ [22]). Some require infrastructural changes and extensive war-driving efforts to obtain feature-rich data sets. Also, most commercial systems (e.g., WiFi Sense [4], Place Lab [18]) turn on the radio interfaces continuously to identify context which results in battery drain. Previous works also captured the high initial costs for Wi-Fi scan/association [9, 24]. Various other techniques reduced idle state power consumption [28, 34] and idle state time periods [20, 26]. Others reduce cellular data transfer costs by introducing delayed transfers [8, 19] or data offloading [24] through Wi-Fi. However, PRiSM predicts Wi-Fi availability at a location in real-time, without turning on the Wi-Fi interface and reduces sensing costs (i.e., radio power up/down, scan, association and DHCP) through use of cellular signals.

5.2 Location Prediction

Prior works use average received signal strengths from connected cellular base stations to predict user location [12, 27, 31]. However, averaging the signal strength values results in loss of granularity. Footprint [31] triggers Wi-Fi scans based on user movement (greater then $10\,m$ indoors or $20\,m$ outdoors) irrespective of the direction. The cellular signal information is used only to measure the distance moved by the user based on signal changes. All locations with no available Wi-Fi are recorded in the history and used to suppress scans at those locations later. As time grows, the list size may increase exponentially and become practically infeasible to maintain. Moreover, even in locations with Wi-Fi, a full scan is performed before connecting to the AP.

PRiSM only shares the key idea of using cellular signals for Wi-Fi prediction with Footprint but is fundamentally different in its design, working and practical aspects. Unlike Footprint, PRiSM

efficiently stores the entire signal distributions of cellular signals clustered into logical locations that have Wi-Fi availability. It dynamically builds and update the signature clusters in near real-time and thus avoids the need for a specialized training phase. It compares only a subset of clusters for signal match at runtime and reduces computation complexity. It uses a non-parametric statistical matching algorithm to decide to either connect to the Wi-Fi directly without scanning or to suppress scans in no Wi-Fi locations. Bartendr [27] predicts future cellular signal strengths based on current data set and direction of travel to schedule cellular data transfers. SmartDC [12] predicts meaningful locations by observing a variety of radio signals with 80% accuracy and a long computational delay of $160\ sec$. But Wi-Fi decisions need to be taken much quicker (order of few seconds) to maintain user experience and additional sensors consume significant battery energy.

5.3 Existing Localization Algorithms

A class of algorithms (referred as *BSSET*) uses the set of cellular BS ID's to evaluate the likelihood of matching a fingerprint in the database. It can simply count the number of common BSs or can sum up the weight values of common BSs, where the weight is assigned to each BS based on its frequency of observation. Another set of algorithms (referred as *MSE*) use mean squared error for matching [23], [29]. An error is defined as the difference between the signal strength in current observation and the average signal strength recorded in the fingerprint for the same BS.

Most Artificial Intelligence (AI) techniques typically identify the top k fingerprints showing the smallest MSE values and then calculate the center from the locations paired with k fingerprints. This extension is called kNN (k-nearest neighbor) but they have the following problems: minimal training phase but costly testing phase including both time and memory, and assumes that data is in feature/metric space which means it is associated with some distance. PRiSM requires entire signal distribution clusters of the training data for quicker prediction and hence, uses a specialized hybrid algorithm which includes lazy learning techniques and statistical likelihood estimation. Both BSSET and MSE algorithms need their own hard-coded threshold value (C) but PRiSM auto-tunes its threshold parameters regularly. Some others [33] use a model-based approach to build radio signal maps. They take more time to converge and require extensive war-driving to generate the data set. PRiSM does not assume anything about the underlying data model or distribution and hence takes a non-parametric approach.

5.4 Screen Activation

Huge energy savings are reported for power measurements for cellular radio/LTE traffic obtained during screen-off conditions [15], Wi-Fi [9, 16, 24, 25]. In PRiSM, we perform measurements under both screen on and off conditions, show that screen off energy is more compared to screen on due to use of CPU wakelocks, and use appropriate energy values for user logs. Hence, the final energy figures of PRiSM more accurately match actual Wi-Fi power consumptions.

6. DISCUSSION

PRiSM places no restrictions to the users in the way they hold their phones or the places they visit. Each user accumulated dynamic cellular signal variations from distinct antenna gains and antenna placement in their devices. PRiSM constructs unique signatures in every device and hence, operates on the assumption that a user connected to the logical Wi-Fi location previously before evaluation phase. As discussed in § 3.2, for Zero-match case, PRiSM creates signatures for the new location only after manual user input. Hence, it only creates signatures for user specified logical locations and reduces meaningless signatures but requires manual supervision. To reduce this limitation, a centralized signature database can be implemented and shared using crowd sourced data. Initially, we only used Nexus One phones to collect data since we get neighbour cell information in addition to connected cell information from the Android software stack. In recent months, many popular devices from Motorola, HTC, Sony and LG started providing the neighbor values. Hence, the scope and impact of PRiSM is wide.

Load balancing or *Cell breathing* techniques are used in CDMA systems where the Base Station (BS) output power is split among active users. Hence, coverage range of cellular towers is shrinked based on user load. In GSM and current LTE systems, each BS usually transmits with full transmission power in the downlink. Based on the location of the mobile within the coverage area, it receives a percentage of the transmitted power which PRiSM utilizes to logically localize locations. PRiSM is not heavily impacted by this because it uses both connected and neighboring cell towers for estimation and hence, absence of one or two BSs does not affect its working. Also note that PRiSM periodically updates the signal strengths for the signatures whenever the user visits the logical locations and hence, provides a robust method to update the database.

An example rule (not given in XML format for the sake of clarity) input that can be implemented in the decision engine is "do not connect to a particular AP on weekdays but do connect on weekends". Hence, PRiSM filters its decision based on the day, even though the AP is available at all days. We see many potential applications for the decision engine in PRiSM (e.g., Cellular network carriers can dynamically modify data-offloading rules for their customers based on their real-time network congestion levels at a location. This can facilitate fast handover between Wi-Fi and cellular data usage and readily complement commercial ISP solutions like Hotspot 2.0 [1]. Note that Hotspot 2.0 promises seamless Wi-Fi authentication and handoff, but still needs to identify places with good Wi-Fi).

Some APs do not allow data transmission even after successful connection with the AP (i.e., closed). PRiSM currently does not handle this specific case, but works well for both open and password protected APs if the device has connected to those APs previously. This limitation can be rectified by first performing additional data connectivity check on each newly connected AP. Closed APs can then be added to a separate list to avoid future connections. Recently, Apple© blocked Wi-Fi scans initiated from any user application and made its Application Programming Interface (API) private. However, PRiSM does not initiate scans, instead, it records information such as AP name, MAC ID's, and the signal strengths after system initiated connection. Though PRiSM is implemented in Android, its functioning holds good for most mobile operating systems in general.

7. CONCLUSION AND FUTURE WORK

In this paper, we presented PRiSM to solve the Wi-Fi Access Point (AP) discovery problem for smart devices by utilizing received cellular signal information and the regularity in human movement patterns. We developed a novel technique to dynamically build and update the signature clusters in near real-time to avoid extensive training time periods. We also developed a statistical matching algorithm with auto-tuning capabilities to maximize Wi-Fi detection opportunities and to simultaneously minimize Wi-Fi sensing energy costs. To the best of our knowledge, we are the first to report the energy usage of Wi-Fi for various phone screen activation scenarios and quantified the energy wastage if connected to an

AP with poor link conditions. We implemented PRiSM on Android phones and show substantial energy savings for both trace-based simulation and practical evaluation. In our future work, we plan to use cellular multi-homing to achieve fine-grained location accuracy by using multiple radio signal combinations (UMTS, LTE) and also to develop a common Wi-Fi signature database for all users.

Acknowledgments

This work was supported in part by the National Science Foundation grants 1016216 and 0910868 and Samsung Electronics Co., Ltd, Korea. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Samsung Electronics Co., Ltd. The authors thank Dr. Kyunghan Lee (UNIST, Korea) and Arpit Gupta (Georgia Tech, USA) for insightful discussions during initial stages of this work. We also thank all anonymous reviewers for their comments and feedback on this work.

8. REFERENCES

[1] Hotspot2.0. `http://www.ruckuswireless.com/technology/hotspot2/`.

[2] How wi-fi drains your cell phone. `http://www.technologyreview.com`.

[3] Monsoon solutions. `www.msoon.com/LabEquipment/PowerMonitor/`.

[4] Wifi sense. `http://www.windowsphone.com/`.

[5] A.Gember, A. Akella, J. Pang, A. Varshavsky, and R. Caceres. Obtaining in-context measurements of cellular network performance. In *IMC*, 2012.

[6] G. Ananthanarayanan and I. Stoica. Blue-Fi: Enhancing wi-fi performance using bluetooth signals. In *ACM MobiSys*, 2009.

[7] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *IEEE INFOCOM*, 2000.

[8] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3G using WiFi. In *ACM MobiSys*, 2010.

[9] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: A measurement study and implications for network applications. In *IMC*, 2009.

[10] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *USENIX*, 2010.

[11] S. Chakraborty, Y. Dong, D. Yau, and J. Lui. On the effectiveness of movement prediction to reduce energy consumption in wireless communication. *IEEE Transactions on Networking*, 5:157–169, 2006.

[12] Y. Chon, E. Talipov, H. Shin, and H. Cha. Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *ACM Sensys*, 2011.

[13] P. Deshpande, A. Kashyap, C. Sung, and S. R. Das. Predictive methods for improved vehicular wifi access. In *ACM MobiSys*, 2009.

[14] M. C. Gonzalez, C. A. Hidalgo, and A. L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453:779–782, June 2008.

[15] J. Huang, F. Qian, Z. M. Mao, S. Sen, and O. Spatscheck. Screen-off traffic characterization and optimization in 3g/4g networks. In *IMC*, 2012.

[16] K. Kim, A. Min, D. Gupta, P. Mohapatra, and J. Singh. Improving energy efficiency of Wi-Fi sensing on smartphones. In *Infocom*, 2011.

[17] A. Kupper. *Location-Based Services: Fundamentals and Operation*. Wiley, 2005.

[18] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit. Place lab: Device positioning using radio beacons in the wild. In *Proceedings of Pervasive Computing*, 2005.

[19] K. Lee, I. Rhee, J. Lee, S. Chong, and Y. Yi. Mobile data offloading: How much can wifi deliver? In *ACM Conext*, 2010.

[20] J. Manweiler and R. R. Choudhary. Sleepwell: Avoiding the rush hours: WiFi energy management via traffic isolation. In *ACM MobiSys*, 2011.

[21] A. J. Nicholson and B. D. Noble. Breadcrumbs: forecasting mobile connectivity. In *ACM Mobicom*, 2008.

[22] J. Paek, J. Kim, and R. Govindan. Energy-efficient rate-adaptive gps-based positioning for smartphones. In *ACM Mobisys*, 2010.

[23] P. Prasithsangaree, P. Krishnamurthy, and P. Chrysanthis. On indoor position location with wireless lans. In *IEEE PIMRC*, 2002.

[24] M. R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. Energy-delay tradeoffs in smartphone applications. In *ACM Mobisys*, 2010.

[25] A. Rahmati and L. Zhong. Context-for-wireless: Context-sensitive energy-efficient wireless data transfer. In *ACM Mobisys*, 2007.

[26] E. Rozner, V. Navda, R. Ramjee, and S. Rayachu. Napman: Network-assisted power management for wifi devices. In *ACM MobiSys*, 2010.

[27] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan. Bartendr: A practical approach to energy-aware cellular data scheduling. In *Mobicom*, 2010.

[28] E. Shih, P. Bahl, and M. Sinclair. Wake on wireless: An event driven energy saving strategy for battery operated devices. In *ACM Mobicom*, 2002.

[29] A. Varshavsky, E. De Lara, J. Hightower, A. LaMarca, and V. Otsason. GSM indoor localization. *Pervasive Mobile Computing*, 3:698–720, December 2007.

[30] W. Wang, M. Motani, and V. Srinivasan. Opportunistic energy-efficient contact probing in delay-tolerant applications. *IEEE/ACM Transactions on Networking*, 17:1592–1605, 2009.

[31] H. Wu, K. Tan, and J. Liu. Footprint: Cellular assisted WiFi ap discovery on mobile phones for energy saving. In *WINTECH*, 2009.

[32] F. Xu, Y. Liu, T. Moscibroda, R. Chandra, L. Jin, Y. Zhang, and Q. Li. Optimizing background email sync on smartphones. In *Mobisys*, 2013.

[33] M. Youssef and A. Agrawala. The horus wlan location determination system. In *ACM Mobisys*, 2005.

[34] X. Zhang and K. G. Shin. E-mili: energy-minimizing idle listening in wireless networks. In *ACM Mobicom*, 2011.

[35] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma. Zifi: wireless lan discovery via zigbee interference signatures. In *ACM Mobicom*, 2010.