An Accurate Two-Tier Classifier for Efficient Duty-Cycling of Smartphone Activity Recognition Systems

Vijay Srinivasan
Samsung R&D Center
San Jose, CA
v.srinivasan@sisa.samsung.com

Thomas Phan
Samsung R&D Center
San Jose, CA
thomas.phan@samsung.com

ABSTRACT

Our vision is a smartphone service that accurately tracks users' physical activities and transportation modes using accelerometer-based activity recognition carried out on the phone. A key challenge that needs to be addressed to realize this vision is the high power consumption of keeping the smartphone *awake* in order to perform sensor sampling, feature extraction, and activity classification. In this paper, we propose a two-tier classifier for reducing the *wake time percentage* of the activity recognition system, which we define as the percentage of time the activity recognition system keeps the smartphone awake. We compare our two-tier classifier to (i) a conventional single-tier classifier, and (ii) a confidence-based multi-tier classifier designed to reduce wake time. We evaluate our approaches using activity-labeled smartphone accelerometer data traces from 2 subjects over a total of 60 hours performing 7 different physical activities. Given an accuracy lower bound of 91%, our two-tier approach achieves the lowest wake times and is able to reduce the wake time percentage of the best single-tier classifier by 93.0% and 70.1% respectively for the 2 subjects.

Categories and Subject Descriptors

H.4.m [**Information Systems Applications**]: Miscellaneous

General Terms

Algorithms, Experimentation

Keywords

Smartphone sensing, Power efficiency, Activity recognition

1. INTRODUCTION AND MOTIVATION

A smartphone service that automatically tracks users' physical activities and transportation modes using only in-phone sensors has numerous exciting applications. Popular commercial products such as the Fitbit [1] have already demonstrated the benefits of long term physical activity tracking and reporting; positive results on behavior change are also observed in the Ubifit study [6]. Additionally, understanding transportation modes could help users assess their environmental impact and make green choices about their transportation habits [13, 7]. Widely available smartphones with inexpensive sensors such as the accelerometer provides an ideal platform for beneficially affecting user behavior on a large scale.

Smartphone platform and operating system	Power consumption due to wake lock on CPU	Accelerometer sampling at 32 Hz	Feature extraction and classification every 4 seconds
Android 2.3.4 on Galaxy SII	184.34 mW	35.53 mW	5.05 mW
Android 2.3.4 on Nexus S	167.15 mW	18.03 mW	4.49 mW

Figure 1: Power consumption break-down of the activity recognition service for two commercial smartphone platforms

Smartphone activity recognition systems need to be highly accurate in order for users to make intelligent lifestyle choices for the application scenarios mentioned above. For example, Reddy et al [13] place a lower bound of 90% accuracy in inferring transportation modes and observe that higher error rates compromise the user's ability to make effective transportation choices. In addition to accuracy, another key constraint is ensuring low power consumption of the activity recognition service. Figure 1 shows the break-down of the power consumption of our activity recognition system (which we will discuss fully in section 3.1) on the Samsung Nexus S and Galaxy SII platforms running the Android operating system, measured using a Monsoon Solutions power monitor [2]. From figure 1, the major bottleneck in continuous accelerometer-based activity recognition in modern smartphones is keeping the main CPU awake; sensor sampling, time domain and frequency domain feature extraction, and decision tree-based activity classification are a relatively small overhead over the baseline CPU power consumption. Recent work has highlighted the significant energy leaks possible due to misbehaving smartphone apps that keep the smartphone continuously awake [10].

In this work, we focus on the problem of accurately recognizing user activities while reducing the amount of time the activity recognition system as a whole is awake. We propose a two-tier classifier for reducing the **wake time percentage** of the activity recognition system, which we define as the percentage of time the activity recognition system keeps

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
PhoneSense'12, November 6, 2012, Toronto, ON, Canada.
Copyright 2012 ACM 978-1-4503-1778-8 ...$15.00.

the smartphone CPU awake. Our two-tier classifier uses a small sampling window to first identify if the user is idle or not, and then acquires more sensor samples to identify the user's fine-grained physical activity. We compare our two-tier classifier to (i) a conventional single-tier classifier, and (ii) a confidence-based multi-tier classifier we designed to reduce wake time. We evaluate our approaches using activity-labeled smartphone accelerometer data traces from 2 subjects over 60 hours performing 7 different physical activities, and perform an exhaustive analysis of parameter settings for the three classifier approaches considered. Given an accuracy lower bound of 91% (which was empirically close to the maximum accuracy achieved by our single-tier classifier on our two datasets), we observe that our two-tier approach achieves the lowest wake times and is able to reduce the wake time of the best single-tier classifier by 93.0% and 70.1% respectively for the 2 subjects.

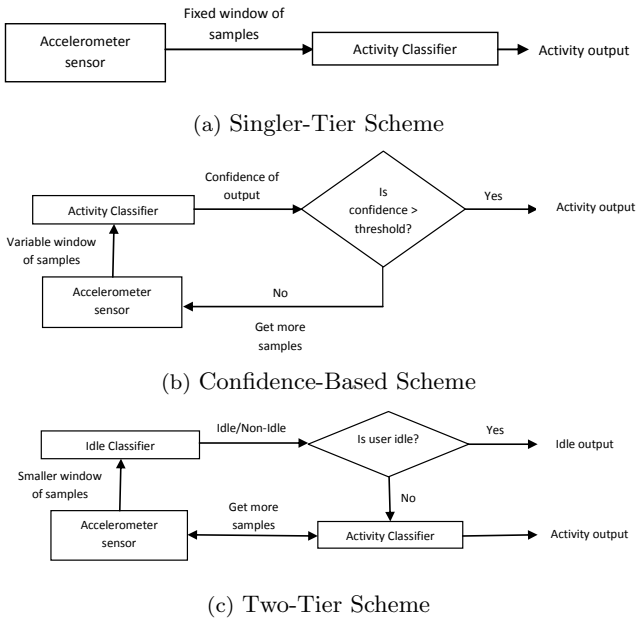
2. RELATED WORK

There is significant existing work on reducing power consumption for smartphone activity recognition. Nath [9] focuses on using intelligent sensor caching and activity correlation to reduce power consumption by using already inferred attributes or choosing less power hungry sensors to infer correlated activities. Other smartphone activity recognition works have explored using data admission control [8], classifier profiling and optimization [5], or triggering high power sensors based on low power sensors to save power [8]. Yan et al [14] focus on adaptively varying the sampling rate and set of features used to achieve the optimal energy consumption above the baseline CPU power consumption.

In contrast to the above existing works, we focus on reducing the high power cost of keeping the smartphone CPU awake to perform activity recognition. Thus, we address the complementary problem of duty cycling the activity recognition system as a whole. Rachuri et al [12] also focus on duty cycling the activity recognition system as a whole; however, their approach adaptively varies the sleep periods between classifier invocations, and treats the activity recognition system as a black box that returns either idle or non-idle events. In contrast, we assume constant sleep periods fixed by the client app, and focus on reducing the amount of time the activity recognition system stays awake to identify the current activity. In the future, we plan to integrate these two approaches to achieve better energy savings than is possible from either approach alone. Finally, there are several proposals for using low power cores in a smartphone dedicated to sensor sampling and data processing [11]. Our two-tier scheme is general enough to be integrated with such low-power cores as well to reduce the inherent wake time percentage and power consumption of activity recognition.

3. ACTIVITY CLASSIFIER DUTY-CYCLING SCHEMES

Figure 2 shows an overview of our three classifier schemes. We first present our single-tier activity classifier, including the feature extraction module and the decision tree classifier used. We then present two extensions to the single-tier classifier to reduce the wake time percentage of the activity recognition system, namely the confidence-based multi-tier classifier, and our proposed two-tier classifier.



(a) Singler-Tier Scheme

(b) Confidence-Based Scheme

(c) Two-Tier Scheme

Figure 2: Three schemes for duty-cycling smartphone activity classifiers

3.1 Scheme I: Single-Tier Classifier

Figure 2a shows the operation of our conventional single-tier classifier scheme. Our assumption is that the single-tier classifier is invoked by a client smartphone app between regular sleep periods of length T seconds, decided by the app. When invoked, the single tier classifier first obtains a fixed size window of N tri-axial accelerometer samples from the smartphone at a fixed sampling rate of f hz, and uses our core **activity classifier** to output a physical activity label such as idling, walking, or biking. We next present the design and implementation of the feature extraction and classification components of our activity classifier.

To perform feature extraction, we first transform the N tri-axial accelerometer samples consisting of three time series into one orientation-independent Cartesian magnitude time series. We extract two time-domain features, namely real-valued power and entropy. We also obtain four frequency-domain features by running an FFT on the magnitude time series: (i) the highest magnitude frequency, (ii) the magnitude of the highest magnitude frequency, (iii) the weighted mean of the top-5 highest magnitude frequencies weighted by magnitude, and (iv) the weighted variance of the top-5 highest magnitude frequencies weighted by magnitude. In section 4.2, we discuss which subset of our 6 features achieves highest accuracy given a window size parameter N. We use the C4.5 decision tree algorithm to perform supervised classification based on features extracted from the raw accelerometer data. For each user, we use the Weka implementation of the C4.5 decision tree builder [4] to train a decision tree model offline using an activity-labeled sequence of accelerometer samples. The trained decision tree model is exported as a JSON file to the smartphone, and is used to perform activity classification in real-time on the phone using native APIs for sampling the accelerometer and our feature extraction modules. Our **activity classifier** is implemented to run on Tizen [3] and Android smartphones, and is also extended to run in a desktop environment for test-

Subject ID	Number of hours	Smartphone platform	Percentage of idle labels	Non-idle activities performed
A	57.21	Tizen on Galaxy SII	94.85%	Walking, running, driving
B	3.13	Android 2.3.4 on Nexus S	29.88%	Walking, running, driving, stairs up, stairs down, biking

Figure 3: Details of smartphone accelerometer data collection from two subjects

ing purposes given a trace of collected tri-axial accelerometer samples from smartphones.

3.2 Scheme II: Confidence-Based Multi-Tier Classifier

A key drawback of the single-tier classifier is that it uses a fixed window of N accelerometer samples, irrespective of the activity being performed by the user. The intuition behind the confidence-based classifier is that we stay awake only for as long as is necessary to obtain a high confidence prediction of the current activity being performed; for example, a small window size might be sufficient to identify activities such as idling with high confidence, but a higher window size might be necessary to differentiate between similar activities such as walking and running slowly.

Figure 2b shows the operation our confidence-based classifier scheme. To implement the confidence-based classifier, we first train a series $A[1...m]$ of m different activity classifiers on a series $W[1...m]$ of m increasing window sizes respectively. The confidence-based multi-tier classifier performs classification repeatedly starting with the smallest window size $W[1]$ and the associated activity classifier $A[1]$, before obtaining $W[2] - W[1]$ samples to move on to the next window size and classifier. For some window size $W[i]$, where $i \in [1, m]$, the confidence-based classifier outputs the prediction result of $A[i]$ and goes back to sleep, if either (i) the confidence value c_i output by classifier $A[i]$ exceeds a **confidence threshold** denoted by $\hat{c}$, or (ii) $i = m$. The confidence threshold $\hat{c} \in [0, 1]$ is determined by the accuracy requirements of the application; higher values of $\hat{c}$ are expected to produce high accuracy results. We obtain the confidence value c_i from each decision tree output as the proportion of training instances that favor the current classifier output in the leaf node of the decision tree. Other classifiers such as Naive Bayes output a probability for the maximum likelihood activity given the sensor data, which could be used as a confidence measure.

3.3 Scheme III: Two-Tier Classifier

A potential drawback of our confidence-based classifier is that the confidence value returned by the activity classifier could be unreliable due to overfitting or differences between the training and test datasets. To overcome this problem, we propose the two-tier classifier scheme, which explicitly leverages the hypothesis that differentiating between the idle and non-idle activities requires a smaller window size compared to identifying the user's fine-grained physical activity. As shown in figure 2c, our two-tier classifier first uses a small window of accelerometer samples to determine if the user is idle or not; if the user is not idle, the two-tier classifier acquires more sensor samples to identify the user's fine-grained physical activity.

For each user, we pick exactly two window sizes w_{idle} and w_{active}, where $w_{idle} < w_{active}$. We then train two classifiers

Scheme	Parameter	Set of possible values searched
All schemes	Sleep period T in seconds	[5, 10, 30, 60, 90, 120]
Single-Tier	Fixed window size N in seconds	[0.125, 0.25, 0.5, 1, 2, 4, 8]
Confidence Based	Confidence threshold $\hat{c}$	[0, 0.3, 0.6, 0.7, 0.8, 0.9, 0.95, 0.98, 1]
Two-Tier	Idle window size w_{idle} in seconds	[0.125, 0.25, 0.5, 1]
	Active window size w_{active} in seconds such that $w_{active} > w_{idle}$	[0.25, 0.5, 1, 2, 4, 8]

Figure 4: All parameters used in the three classifier schemes, and the set of possible parameter values explored for each scheme

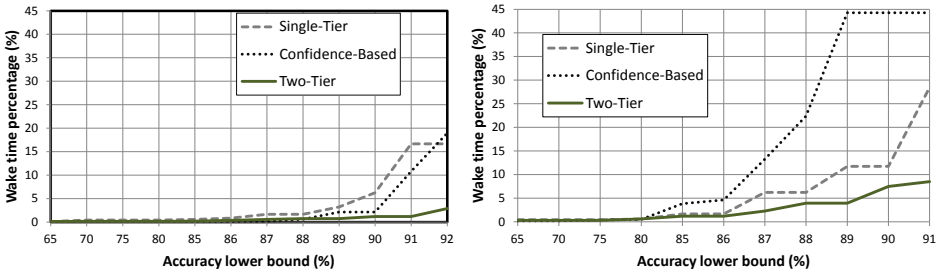
A_{idle} and A_{active} offline corresponding to the two window sizes. Our idle activity classifier A_{idle} classifies a window of w_{idle} accelerometer samples into idle or non-idle; in the training data for A_{idle}, we map all mobile activities such as walking, running and biking to a single **non-idle** predicate. We train our regular single-tier classifier offline using a window size of w_{active} to differentiate among all activities labeled by the end user. In online operation, Tier I of our two-tier classifier first obtains w_{idle} samples and uses classifier A_{idle} to determine if the user is idle or not. If the user is idle, the two-tier classifier goes back to sleep for a sleep period T determined by the client app. If the user is not idle, Tier II of our two-tier classifier acquires the remaining $w_{active} - w_{idle}$ samples from the accelerometer, and applies classifier A_{active} to infer the user's fine-grained physical activity. If the user is idle for a significant fraction of the time, and the idle classifier is accurate for a small window size w_{idle} relative to w_{active}, we expect our two-tier approach to reduce the wake time percentage of the activity recognition system without sacrificing accuracy significantly.

4. EVALUATION

In this section, we discuss the evaluation of our three approaches for duty-cycling the activity recognition system, including details of the data collection effort from two subjects over 60 hours to evaluate our three approaches. We observe that the two-tier approach outperforms the confidence-based approach and the single-tier approach significantly at high accuracy bounds. The two-tier reduces the wake time percentage of the single-tier approach by 93.0% and 70.1% respectively for the two subjects, given a classification accuracy lower bound of 91%.

4.1 Data Collection

We collected data from two subjects to evaluate our duty-cycling approaches. Table 3 shows the details of the data collection effort. For both subjects, tri-axial accelerometer data was collected at a fixed sampling rate of 32 Hz. Both subjects were male professionals employed full-time in a software research lab in California. Subject A used a Tizen app on the Galaxy SII that continuously logs accelerometer data in the background, and another Tizen app that lets the user select her current activity from a list displayed on the smartphone. Subject B used an Android app on the Nexus S that lets the user explicitly start and stop the accelerometer data collection, and specify the activity performed between the start and stop intervals. Subject A collected continuous labeled data over 2.5 days, while subject B collected chunks of labeled idle and non-idle activities over several weeks. Idle activities labeled by the subjects included activities such as sitting and standing with the phone in the pocket, or the



(a) Subject A (b) Subject B

Figure 5: Best wake time percentages achieved by the three duty-cycling schemes given a lower bound on classification accuracy

phone placed on a stationary surface. Each non-idle activity was generally performed with the phone in a fixed position relative to the user, such as driving with the phone in a bag, or walking with the phone in the front pocket. A promising split and merge approach has been proposed recently [8] to recognize activities performed with multiple phone positions.

4.2 Experimental Setup

We run our three duty-cycling schemes on the collected data traces in an offline desktop environment to evaluate the wake time percentage of each approach. We first explicitly divided the collected data into training and test datasets. For subject A, we use the first 25.8 hours of the dataset for training, and the remaining 31.4 hours of the dataset for testing our duty-cycling approaches. For subject B, we randomly select 1.52 hours of activity chunks containing all activities as the training dataset and use the remaining 1.61 hours as the test dataset. Figure 4 summarizes the parameters used in the three classifier schemes, and the possible set of values for each parameter explored in our evaluation. For ease of illustration, we depart from our previous convention and represent window sizes in seconds instead of the number of accelerometer samples. For each scheme, we perform a brute force search and compute the wake time percentage and classification accuracy for each point defined by the parameter space shown in figure 4.

Feature selection is an important concern for classification algorithms with limited training data. For the single-tier classifier in scheme I, we only use the magnitude series and time-domain features for window sizes $N \leq 0.5$ seconds, based on empirical evidence that additional features did not improve average classification accuracy significantly for small window sizes. For window sizes larger than 0.5 seconds, we use the magnitude time series and both time-domain and frequency-domain features. For the confidence-based classifier in scheme II, we use the same sequence of fixed window sizes from figure 4 used for scheme I, and the corresponding classifier models that were built for our scheme I evaluation. Finally for the two-tier classifier in scheme III, we use the same feature set configuration used in scheme I for the active window sizes w_{active}. For all the idle window sizes w_{idle}, we limit the feature set to only the magnitude time series and the time-domain features, based on empirical evidence that this limited feature set is sufficient to differentiate between idle and non-idle activities with high accuracy.

We use two main evaluation metrics for our three duty-cycling schemes: **classification accuracy** and **wake time percentage**. Classification accuracy is measured as *time slice accuracy*, i.e. the percentage of time slices (one sample in our evaluation) for which the ground truth label assigned by the user and the classifier output match. Since our classifiers are invoked only once every T seconds, we interpolate the latest classifier result from the last classifier invocation over the sleep period T. For subject A, since more than 94% of the time slices contain the idle activity, we report classification accuracy as the average of the individual time-slice accuracies of the four activities from figure 3 performed by the user; we do this averaging to ensure that the accuracy does not reflect the classifier's ability to detect the idle activity alone. For subject B, we report the overall time slice accuracy without any averaging. Since we run our three classifiers *offline* on the collected data traces, we measure **wake time percentage** as the percentage of always awake accelerometer samples used by each scheme, asssuming zero lag time between when the wake lock is released and the CPU goes to sleep; we also ignore the negligble wall time required on modern smartphones to compute our FFT and time-domain features for the magnitude time series with at most 256 samples (8 second windows). In the future, we plan to measure wake time percentage on a smartphone taking into account the possibly small platform-dependent lag time.

Accuracy lower bound (%)	Average parameters across 3 subjects for scheme #:						
	I		II		III		
	T	N	T	$\hat{c}$	T	w_{idle}	w_{active}
65	90	0.1875	120	0.35	120	0.125	0.375
70	75	0.3125	120	0.8	120	0.125	0.5
75	120	0.5	105	0.85	120	0.125	0.5
80	105	0.5	105	0.9	120	0.125	0.75
85	75	0.75	65	0.9	75	0.1875	0.75
86	90	1	65	0.925	90	0.25	1.5
87	30	1.25	75	0.95	45	0.25	1
88	30	1.25	17.5	0.94	60	0.25	3
89	30	2.5	7.5	0.975	60	0.25	3
90	45	4	7.5	0.975	45	0.375	4
91	7.5	2.5	7.5	1	45	0.75	4

Figure 6: Average parameter settings for best instances of each scheme given a lower bound on accuracy

4.3 Classifier Results

Figure 5 shows the results of our parameter search over all the parameters shown in figure 4 to find the best wake time percentages for schemes I, II, and III, given a classification accuracy lower bound. As seen in figure 5, for subject A, given an accuracy bound of 91%, our two-tier classifier reduces the wake time percentage of the single-tier classifier from 16.67% to 1.16%, resulting in a reduction

of 93%; for subject B, given an accuracy bound of 91%, our two-tier classifier reduces the wake time percentage of the single-tier classifier from 28.43% to 8.5%, a reduction of 70.1%. The two-tier approach achieves a lower reduction in wake time percentage for subject B compared to subject A, due to the relatively small proportion of idle time from subject B's dataset (29.88%) compared to subject A's dataset (94.85%), as seen in figure 3. Figure 6 shows the average values of the best parameters for each scheme and for each lower bound accuracy value across the two subjects (corresponding to the wake time percentage results shown in figure 5). The two-tier approach combines idle window sizes less than 1 second with activity window sizes of around 4 seconds to achieve a significantly better wake time percentage compared to the single-tier classifier scheme. *We limit our exploration of classification accuracy lower bounds to 92% for subject A and 91% for subject B, because these are the highest integer accuracy lower bounds satisfied by both schemes I and II for both subjects for the parameters shown in figure 4*; for subject A, scheme I alone achieves a higher 93.29% accuracy for a wake time percentage of 16.67%, but scheme III achieves 93.07% accuracy for a lower wake time percentage of 6.24%. We found that the best *always on* classifier with 100% wake time percentage only achieves slightly better accuracies, namely 94.6% and 93.4% for subjects A and B respectively.

In figure 5, for both subjects, we note that at lower accuracy bounds, the difference in wake time percentages among the 3 schemes is small enough that it is likely to become negligible given delays in the smartphone going to sleep once the wake lock is released. For both subjects, at high accuracy bounds (greater than 90% for subject A and 86% for subject B), the confidence-based approach requires very high wake time percentages. This is because any leaves with very high confidence were limited in number and were seldom reached in the decision trees trained on smaller training datasets; as a result, the algorithm defers to the highest window size at the high confidence thresholds shown in figure 4. We propose to explore confidence-based approaches in the future with classifiers such as Naive Bayes classifiers that return a probability distribution over activity labels.

Figure 7 shows the average accuracies and wake time percentages across the two subjects, given *both* sleep period constraints and sample accuracy constraints. We picked the sample accuracy constraints to be the highest integer accuracy satisfiable by either scheme I or II for each sleep period and for each subject; we show the results for scheme III, and pick the best results among either scheme I or scheme II for comparison. We observe that our scheme III achieves similar accuracy as the other schemes, but achieves at least a 50% reduction in wake time percentage for each sleep period up to 30 seconds, compared to the best from schemes I and II. At very high sleep periods, we observe that all schemes achieve a similar, low wake time percentage. Thus, from figures 5 and 7, we note that our two-tier approach is most effective for applications with high accuracy and/or low sleep period requirements, such as applications that immediately push notifications or make media recommendations to end users based on a change in physical status.

5. CONCLUSIONS AND FUTURE WORK

In this work, we present and evaluate a two-tier classifier for reducing the wake time percentage of smartphone activity recognition systems. Based on accelerometer data traces collected from 2 subjects over 60 hours, we show that our two-tier classifier reduces the wake time percentage by 93.0% and 70.1% respectively for the 2 subjects, given a 91% accuracy lower bound. As part of future work, we propose to evaluate how our approach could be integrated with adaptive sleep periods [12] to reduce wake time percentage even further. We also propose to evaluate and minimize the adverse effect of any CPU lag time on waking up and going to sleep. Finally, we propose to integrate our two-tier scheme on low power cores; an interesting direction is to run the light-weight sensor sampling and Tier I algorithm on the low power core, and wake up the high power core if necessary for Tier II computation.

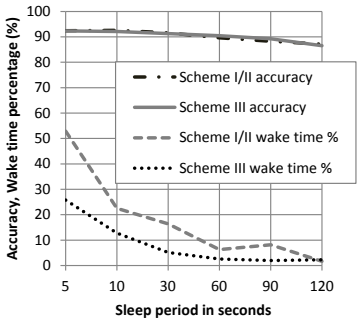


Figure 7: Scheme III outperforms schems I and II, given sample accuracy requirements and sleep period constraints

6. REFERENCES

[1] Fitbit wireless activity tracker. `http://www.fitbit.com/`.

[2] Monsoon Power Monitor. `http://www.msoon.com/LabEquipment/PowerMonitor/`.

[3] Tizen Operating System. `https://www.tizen.org/`.

[4] Weka - Machine Learning Software in Java. `http://sourceforge.net/projects/weka/`.

[5] D. Chu et al. Balancing energy, latency and accuracy for mobile sensor data classification. In *Sensys*. ACM, 2011.

[6] S. Consolvo et al. Activity sensing in the wild: a field trial of ubifit garden. In *CHI*. ACM, 2008.

[7] J. Froehlich et al. Ubigreen: investigating a mobile tool for tracking and supporting green transportation habits. In *CHI*. ACM, 2009.

[8] H. Lu et al. The jigsaw continuous sensing engine for mobile phone applications. In *Sensys*. ACM, 2010.

[9] S. Nath. Ace: exploiting correlation for energy-efficient and continuous context sensing. In *Mobisys*. ACM, 2012.

[10] A. Pathak, A. Jindal, Y.C. Hu, and S.P. Midkiff. What is keeping my phone awake?: characterizing and detecting no-sleep energy bugs in smartphone apps. In *Mobisys*. ACM, 2012.

[11] N. Priyantha et al. Eers: Energy efficient responsive sleeping on mobile phones. In *PhoneSense*, 2010.

[12] K.K. Rachuri et al. Sociablesense: exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In *Mobisys*. ACM, 2011.

[13] S. Reddy et al. Using mobile phones to determine transportation modes. *ACM Transactions on Sensor Networks (TOSN)*, 2010.

[14] Z. Yan et al. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *International Symposium on Wearable Computers*. IEEE, 2012.