Energy-Efficient Data Collection for Context Recognition

Vito Janko
Jožef Stefan Institute
Jožef Stefan International
Postgraduate School
Ljubljana, Slovenia
vito.janko@ijs.si

Mitja Luštrek
Jožef Stefan Institute
Jožef Stefan International
Postgraduate School
Ljubljana, Slovenia
mitja.lustrek@ijs.si

UbiComp/ISWC'17 Adjunct , September 11–15, 2017, Maui, HI, USA
ACM 978-1-4503-5190-4/17/09.
https://doi.org/10.1145/3123024.3124430

Abstract
Detection of the user's context with mobile sensing systems is a common problem in ubiquitous computing. However, the typically small battery of such systems is often making continuous detection impractical. The strain on the battery can be reduced if the sensor setting is adapted to each context. We propose a method that efficiently finds near-optimal sensor settings. It uses Markov chains to simulate the behaviour of the system in different configurations, and multi-objective genetic algorithm to find a set of good non-dominated configurations.

Author Keywords
Context recognition; energy efficiency; wearable sensors; smartphone; Markov chains

ACM Classification Keywords
H.2.8 [Database Applications]: Data mining; G.3 [Probability and statistics]: Markov processes

Introduction
Widespread accessibility of wearable sensing devices allows for many possibilities for tracking the users who wear them. Data collected is often used to detect the user's context. For example, we could recognize if the user is walking, running, resting or performing similar activities using accelerometer data. This task was made easier and more

practical with the increased use of smartphones, which have many sensors built in. Sensing with multiple sensors, possibly at once, opens additional options for context recognition: detecting one's location or contexts such as shopping, traveling or work.

A major limitation of continuous data collection and context recognition is its heavy toll on the sensing device's battery life. This is especially relevant for smartphones, which have a very limited battery that must be shared between many applications. This issue is often neglected when discussing the design of context-recognition systems, however it is an essential component if such systems are to be used in practice.

Some solutions deal with this issue by optimizing the sampling rate or sensor duty cycle for the particular recognition task [3]. This, however, might be suboptimal. We might for example want to have the GPS active at a high sampling rate while the user is driving, but at a very low rate when they are working in the office. This calls for dynamic changes in the settings for both the sensors and for the subsequent processing. While many adaptive approaches already exist [5] they have a problem in that they are designed for a specific problem setting. If we were to recognize different contexts using different sensors we would have to adapt these methods, requiring either a lot of experimentation or expert knowledge. It would be useful to have a method that can be provided with a sensor-rich dataset, and is able to automatically tell which sensors or which sensor setting to use in each context.

A relatively simple solution of the above problem was proposed by Zhixian Yan, et al. [6]. They select the sensor settings and subsequent processing method based on the last classified activity. For each activity, they find the setting that best recognizes it and then use it when that activity is detected. However, since the selection is made for each activity in isolation, the effect on the whole system can be unpredictable. To illustrate: an accelerometer is very good at recognizing walking and resting, while a GPS is very good at recognizing driving. However, if we have only the accelerometer active while walking, driving will never be detected and the sensor switch will never occur.

Our goal is to build upon this approach in two ways. First we propose a Markov chain model to simulate runtime setting switching and predict what would happen if we used a particular setting for a particular context, taking into account above described interactions. Similar models were successfully used in related domains [4]. Second, we use a genetic multi-objective optimization - objectives being accuracy of the system and its energy consumption - to decide which assignments of settings to contexts to evaluate.

Problem description and methods

Suppose we have a sensing system that works as follows: the user is in one of the predefined contexts $c \in C$ and each context is associated with some setting $s \in S$. Settings can be which sensors are in use, or what sampling frequency or duty-cycle policy is active and so on. A setting is used as long as the system believes that user is in a context that corresponds to that setting. When a context change is detected by the system, the setting is changed accordingly. For example: if we are sitting we might use a low sensor sampling frequency, but when walking is detected, a higher sampling frequency can be turned on.

The main issue of this paper is to efficiently find an assignment of settings to contexts that generates good trade-offs between system energy consumption and system quality (for example: accuracy of the context recognition). What trade-off is considered best is of course up to the system's

$p(c)$ – the probability of context c, estimated by its proportion in the dataset

$Acc(c|s)$ – the accuracy for recognizing context c, given the setting s is active.

$s(c)$ – the setting corresponding to context c.

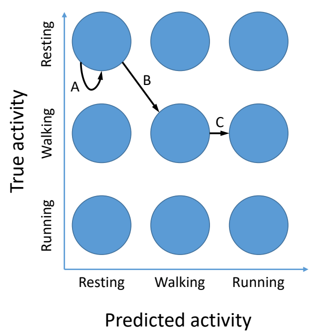

Figure 1: Markov chain, where each state represents a (true, predicted) activity of the user. Transition A, for example, models the situation where the user is continually resting, and their activity is recognized as such. For transition B, the user started walking, which was correctly recognized. Finally, transition C implies incorrect classification from walking to running.

designer, but we should strive to present him Pareto optimal solutions to choose from.

The most straightforward way to determine the optimal assignment of settings to contexts for such a system is to simply try out every assignment. This approach has two obvious problems that make it infeasible for a large number of settings and/or contexts. The first is the exponential number of assignments; if we have $|C|$ contexts and $|S|$ settings, we would have $|S|^{|C|}$ different assignments. Given that sensing system can have many sensors, each having many ways to be configured, we can expect the set of system settings to frequently be large. The second is the time of evaluation of one assignment; one must create appropriate machine learning models and then run through all the data in the dataset, classifying instances while simulating different settings and switching between them accordingly – a process that can be prohibitively slow (lasting seconds or minutes), if we plan on evaluating a large number of assignments.

The often used approach is to use expert knowledge to determine sensible assignments and test only those. However, such knowledge is often either not available or not good enough to isolate all good potential assignments, which calls for an alternative, automatic method for solving the task.

Reducing the time of evaluation

To evaluate as many assignments of settings to contexts as possible, we strive to reduce the time of each evaluation by mathematically estimating the energy consumption of the system and its classification quality and avoid actual experiments. One experiment is considered to be classifying all instances in the dataset using appropriate settings, and creating new classification models if so needed for the task.

Naive evaluation model

A simple approach, similar in spirit to the work of Zhixian Yan, et al. [6], is to first determine the accuracy of the system using each setting, by experimental evaluation. In these experiments the same setting is kept throughout, without switching depending on the context. This requires $|S|$ experiments, exponentially fewer than $|S|^{|C|}$. Then, to evaluate a specific assignment, the accuracies of each context c, using the corresponding setting $s(c)$, are summed up, weighted by the proportion of that context in the dataset.

$$\text{Accuracy} = \sum_{c \in C} p(c) Acc(c|s(c))$$

We can make a similar prediction about the energy consumption, given the information about the energy requirement of each setting.

This evaluation, while easy to compute, models only how the system behaves when the correct setting is active. However, it does not model what happens when context changes or when the classifier makes a classification mistake and the system subsequently switches to a setting inappropriate for the current context.

Markov-chain evaluation model

To improve upon the naive model, we propose a Markov-chain model. We create a Markov chain that has $|C|^2$ states. Each state represents a pair (current context, context the system believes we are in), marked (c, c') in short (example on Figure 1). Our goal will be to calculate the "steady-state" of this Markov chain, which gives us the proportion of the time we would spend in each of those states, given infinite or at least large enough time. Using this information, we can then make various predictions about the system performance that are much better than the ones using the above described naive approach.

$P((a,b) \rightarrow (c,d))$ – the probability of a transition from state (a,b) to state (c,d)

$T(a,c)$ – the probability in the dataset that the next context will be c given that the current one is a.

$C_s(d|c)$ – the probability that the classifier that works with setting s will classify an instance to d, if the true context is c.

$s(m)$ – the setting corresponding to state m.

$c(m)$ – the context corresponding to state m.

$acc(c,s)$ – the accuracy of the classifier that works with setting s, if the true context is c.

M – the set of all states in the Markov chain.

$t(m)$ – the predicted proportion of time spent in state m.

$e(s)$ – the energy requirement of a setting s per time unit.

Conf$_{i,j}$ – the value of the confusion matrix, for correct context i and predicted context j. Note: all the values are normalized to sum to 1.

We begin by making experiments where for each setting, a confusion matrix is calculated. Again this is done without sensor switching and requires $|S|$ experiments. Additionally, we need the transition probability $T(c_i, c_j)$ from each context to each other; that can easily be inferred from the dataset. Finally, we must make energy consumption estimation per time unit for each setting $e(s)$.

To evaluate a particular assignment, we have to calculate the transition probability from one Markov state to another. They can be calculated from the transition probabilities of the contexts and data from the previously computed confusion matrices. Intuitively: we get in a state (c,c') if the context really changes to c and if the system classifies this instance into c'.

$$P((a,b) \rightarrow (c,d)) = T(a,c)C_{s(b)}(d|c)$$

Having all the transition probabilities, we can use the basic Markov chain calculus to calculate the steady state of the Markov chain. This gives us the amount of time the system will be in each of the Markov states. Since we know how much time any setting is active and how much energy this setting consumes per time unit, the energy consumption of the whole system can be estimated. Additionally, since the confusion matrices give us the accuracy for each state, we can calculate the accuracy of the whole system.

$$\text{Energy estimation} = \sum_{m \in M} t(m)e(s(m))$$

$$\text{Accuracy estimation} = \sum_{m \in M} t(m)acc(c(m), s(m))$$

It should be noted that many other metrics can be determined from such a model. Example: precision, recall, F-score or the latency of activity-change detection. They can be used instead of the accuracy when evaluating the performance of the system using a chosen assignment. Most of those metrics can be calculated directly from the confusion matrix for the whole system, which in turn is trivially calculable from the steady state.

$$\text{Conf}_{i,j} = t(m(i,j))$$

Multi-objective optimization

An individual assignment can be accurately and quickly evaluated using a Markov-chain model. Their number however, scales rapidly with the increasing number of settings and contexts. For large $|S|$ and/or $|C|$ a more efficient search is required. Since we are essentially solving a problem of multi-objective optimization, we can use methodology from that research field.

We used the NSGA-II [2], a genetic multi-objective optimization algorithm, with good results. We assume that some other similar algorithm could be used in its place. Assignments of settings to contexts are used as the inputs to be optimized. If the settings themselves have a sensible structure – and if we have a huge amount of settings, they are almost bound to have it – we can use this structure to encode the inputs, to make algorithm operations (permutations, mutations) more natural. Example: if our setting is a combination of sensors that can be turned on or off, it can be written as a binary string, where 1 represents an active sensor. We can then concatenate all those strings for each setting that is associated to one of the contexts to get the final input for the algorithm.

Name	I [mA]
/	20
A	46
B	45
W	30
S	55
A,B	52
B,W	75
B,S	68
A,W	70
A,S	61
W,S	84
A,B,W	74
A,B,S	70
B,W,S	90
A,W,S	85
A,B,W,S	100

Table 1: Energy cost of different sensor combinations measured in mA. Sensors that were active were labeled A – accelerometer, S – sound, B – Bluetooth, W – WiFi.

	Naive	Markov
Acc. [%]	5.2	3.0
F-score	0.04	0.02
Energy[mA]	4.7	1.4

Table 2: The average prediction mistake of 300 random assignments, made on energy consumption, F-score and accuracy for the naive and Markov-chain model.

Dataset

The method was evaluated on the Commodity12 dataset [1]. In the Commodity12 project we were collecting data of diabetic patients that were wearing a smartphone and a chest heart-rate monitor. The data came from ten different physical and virtual sensors: accelerometer, barometer, light sensor, GPS location, a list of visible WiFi networks, location description by Foursquare webservice, sound, time, heart rate, and respiration rate. From it, we were interested in recognizing the following contexts: sleep, work, home, eating, transport, exercise, out (out of house, but not in any of the previous contexts). While our classification accuracy was reasonably high, sensors drained the phone's battery in less than a day, which called for energy optimization.

We started by choosing the sensor settings. Every sensor can be either active or not. We also included the option of the sensor in a "duty-cycling" mode, meaning that it is active some set amount of time (1 minute in our case) and then inactive for some time (14 minutes in our case). Using this list we have 10 sensors with 3 options giving us 3^{10} settings for any given context. Since we have 7 different contexts, and a setting must be chosen for each one, this corresponds to $(3^{10})^7 \approx 10^{33}$ different system configurations. Given this number, it is easy to see that any brute-force approach is bound to fail, and any handpicked configuration is likely to be sub-optimal.

Energy consumption

We must estimate the energy requirements for different combinations of sensors being active. Since the individual-sensor energy consumptions do not add up linearly, we had to try every combination. Fortunately this process was simplified by the following factors: 1) for both the heart rate and respiration rate only the Bluetooth connection is required and its power consumption is the same with one or two data streams; 2) having one, two or all of the following sensors – accelerometer, light and pressure – resulted in a very similar energy consumption, perhaps because they share many resources to operate; 3) Foursquare is "free" if WiFi and GPS sensors are active and it costs as those two otherwise (it needs both to work); 4) time "sensor" is free; 5) GPS added an extra 40 mA on average regardless of other sensors. This narrows the number of required combinations to test down to 16. The results are listed in Table 1.

Results

We used the NSGA-II algorithm to find a set of nondominated solutions, approximating the Pareto front, and used Markov chains for the evaluation of the points that the algorithm was iteratively selecting. The resulting set can be found in Figure 2.

We started by estimating the mean absolute error of predicted values compared to those we got in real experiments with runtime sensor data switching. We tested both on random 300 assignments and on points on the resulting Pareto front. Results are listed in Table 2 and Table 3. The F-score was added to demonstrate that other metrics of classification quality can be predicted and used instead of the accuracy. In all cases, the Markov-chain model outperformed the naive one and its predictions were very close to the actual values.

Point A on Figure 2 shows an assignment that performs as well as the original system with all the sensors turned fully on. However, it consumes almost half less energy as the original (which consumed 140 mA). Point B has only the sensor "time" active. As this data is essentially free, the sensing system consumes as little energy as the baseline case. Even so, it performs at a 65% accuracy, much higher then the majority classifier would (the majority class

	Naive	Markov
Acc. [%]	5.8	2.3
F-score	0.09	0.03
Energy[mA]	4.9	1.1

Table 3: The average prediction mistake of assignments in the nondominated set, made on energy consumption, F-score and classification error for the naive and Markov-chain model.

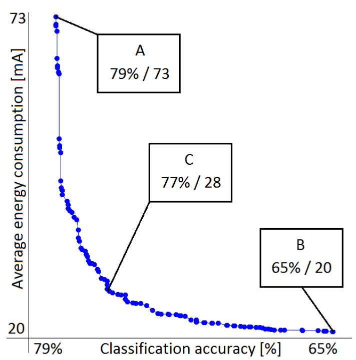

Figure 2: Nondominated set of assignments evaluated in the Commodity12 experiment. Three points of interest were labeled, and their value is displayed as classification accuracy [%] / energy consumption [mA].

occurs in in 46% cases). Since the training and test data came from the same person, it is easy to explain why the time of the day might be an informative feature. Lastly we have the point C, which looks like a reasonable trade-off. It consumes little energy, but loses negligible accuracy compared to the best case. It accomplishes this by discarding the sound entirely and uses the GPS sparsely. All the other sensors are almost always in the "duty-cycle" mode, which can be considered reasonable as the daily activities in this dataset do not change often. Two most frequent activities "home" and "work" can also be retrieved solely by using the list of visible WiFi networks.

The Markov-chain model performs on average roughly 37 000 times faster on the Commodity12 dataset then an average experiment, needing 1.5 ms for a single evaluation in comparison to 55 000 ms for an experiment (time of building a model excluded). Using multi-objective optimization we were able to find solutions shown in Figure 2 after evaluating 100 000 assignments, magnitudes less than the 10^{33} assignments in the search space.

Conclusion

The results in this paper show that using Markov chains, we can accurately and quickly predict the behaviour of a sensing system that is switching between its settings based on the current context. Markov-chain model, combined with an efficient search implemented using a multi-objective genetic algorithm, provides a powerful tool for optimizing the energy needs of a data collection system while maintaining its context classification accuracy.

Using this approach we found some good solutions for the Commodity12 system, one of them consuming 5 times less energy in exchange for 2% accuracy loss, better then any solution we found using expert knowledge alone.

REFERENCES

1. Božidara Cvetković, Vito Janko, Alfonso E Romero, Özgür Kafalı, Kostas Stathis, and Mitja Luštrek. 2016. Activity Recognition for Diabetic Patients Using a Smartphone. *Journal of medical systems* 40, 12 (2016), 256.

2. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6, 2 (2002), 182–197.

3. Aftab Khan, Nils Hammerla, Sebastian Mellor, and Thomas Plötz. 2016. Optimising sampling rates for accelerometer-based human activity recognition. *Pattern Recognition Letters* (2016).

4. Vassilis Kostakos, Denzil Ferreira, Jorge Goncalves, and Simo Hosio. 2016. Modelling smartphone usage: a markov state transition model. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 486–497.

5. Jeongyeup Paek, Joongheon Kim, and Ramesh Govindan. 2010. Energy-efficient rate-adaptive GPS-based positioning for smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 299–314.

6. Zhixian Yan, Vigneshwaran Subbaraju, Dipanjan Chakraborty, Archan Misra, and Karl Aberer. 2012. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *Wearable Computers (ISWC), 2012 16th International Symposium on*. Ieee, 17–24.