AdaSense: Adapting Sampling Rates for Activity Recognition in Body Sensor Networks

Xin Qi, Matthew Keally, Gang Zhou, Yantao Li, Zhen Ren
Department of Computer Science
College of William and Mary
Williamsburg, VA, USA
{*xqi, makeal, gzhou, yantaoli, renzh*}*@cs.wm.edu*

***Abstract*—In a Body Sensor Network (BSN) activity recognition system, sensor sampling and communication quickly deplete battery reserves. While reducing sampling and communication saves energy, this energy savings usually comes at the cost of reduced recognition accuracy. To address this challenge, we propose AdaSense, a framework that reduces the BSN sensors sampling rate while meeting a user-specified accuracy requirement. AdaSense utilizes a classifier set to do either multi-activity classification that requires a high sampling rate or single activity event detection that demands a very low sampling rate. AdaSense aims to utilize lower power single activity event detection most of the time. It only resorts to higher power multi-activity classification to find out the new activity when it is confident that the activity changes. Furthermore, AdaSense is able to determine the optimal sampling rates using a novel Genetic Programming algorithm. Through this Genetic Programming approach, AdaSense reduces sampling rates for both lower power single activity event detection and higher power multi-activity classification. With an existing BSN dataset and a smartphone dataset we collect from eight subjects, we demonstrate that AdaSense effectively reduces BSN sensors sampling rate and outperforms a state-of-the-art solution in terms of energy savings.**

***Keywords*-Activity Recognition, Sampling Rate Reduction, Body Sensor Network.**

I. Introduction

Body Sensor Networks (BSNs) consist of a group of wireless sensors with various monitoring capacities and different modalities. Data collected by body sensors is transmitted to an aggregator (e.g., a PC or a smartphone) and then is either analyzed by the aggregator or reliably delivered to a data center (e.g., a hospital) for future analysis. The most classic application based on a BSN is activity recognition, upon which more complex human-centered real-time applications, such as personal health monitoring [7], assisted living [5] and physical fitness assessment [3], can be built. To provide both high accuracy and real-time recognition, the activity recognition system must continuously fetch sensor data from sensor nodes. However, the intensive sensing and communication workload can quickly deplete the energy of sensor nodes, which is usually battery powered. Consequently, energy efficiency concerns are paramount in BSN system design.

One widely used method to reduce the energy consumed by sensing and communication is to decrease the number of raw samplings the sensors generate per unit time, i.e., sampling rate. Although reducing the sampling rate can prolong the BSN lifetime, this reduction is usually achieved at the expense of sacrificing recognition accuracy [16]. In this paper, we propose AdaSense, a framework that reduces BSN sensors sampling rate while maintaining high accuracy (or meeting a user-specified accuracy requirement). The three main contributions of AdaSense are:

First, through experiments and analysis, we demonstrate that to fulfill any accuracy requirement, detecting whether a specific activity happens or not (***single activity event detection*** through activity binary classification) requires a lower sampling rate than classifying multiple activities (***multi-activity classification***). Moreover, we notice that a human being's activity (e.g., sitting, lying down or walking) is a continuous process that lasts for a time period, which enables us to pursue system optimization through activity-specific design. Specifically, in AdaSense, we design the *Efficient Activity Recognition* (EAR) that performs either lower power single activity event detection or higher power multi-activity classification. When the current activity does not change (comparing to the previous activity), EAR performs single activity event detection with a lower sampling rate. When an activity change is detected, EAR performs multi-activity classification with a higher sampling rate to identify the new activity. After identifying the new activity, it switches back to event detection of the new activity with a lower sampling rate. As a result of combining both lower power single activity event detection and higher power multi-activity classification in runtime, the sensors sampling rate is reduced on average compared to only using higher power multi-activity classification.

Second, to amplify the sampling rate reduction brought by EAR, we propose a second method, *Sampling Rate Optimization* (SRO), which aims to reduce the sensors sampling rates of both single activity event detection and multi-activity classification under any accuracy requirement. This approach uses a novel Genetic Programming based (GP-based) algorithm to explore the feature set space and attempts to find an optimal feature set that effectively reduces the optimal (or the minimally necessary) sampling rate of multi-activity classification under any accuracy requirement. Moreover, the corollary that we prove in this paper guarantees that the

978-1-4799-0187-6/13/$31.00 ©2013 IEEE

optimal sampling rate of single activity event detection is no larger than that of multi-activity classification under any accuracy requirement. Therefore, this GP-based algorithm not only reduces the optimal sampling rate of multi-activity classification, but also has the potential to minimize the optimal sampling rate of single activity event detection without extra effort. This potential is validated in our evaluation.

Third, we evaluate AdaSense with an existing BSN dataset and a smartphone dataset we collected from eight subjects. The results demonstrate that AdaSense is able to effectively reduce the optimal sampling rates of both multi-activity classification and single activity event detection. Through the energy emulation method in [27], we also demonstrate that AdaSense achieves energy saving, ranging from 39.4% to 51.0%, on smartphone compared to the most recent sampling rate reduction method [27].

In literature, many methods have been proposed to reduce the energy overhead of sensing. Some works [16] [21] [23] benchmark several system parameters (e.g., sampling rate and sampling length (how long each sampling takes)) to identify thresholds that balance computation cost with accuracy. However, none considers combining detection (binary classification) with multi-classification to further reduce sensors sampling rate. Many works [12] [13] [24] [26] pursue energy savings through either static or dynamic sensor cluster selection. These approaches rarely consider adjusting other system parameters such as sampling rate that we focus on in this paper.

The rest of the paper is organized as follows. Related work is given in Section II. The motivation and design of AdaSense are given in Sections III and IV, respectively. Section V presents the evaluation results. Finally, Section VI concludes the paper.

II. Related Work

Reducing sampling rates to save energy is a well studied research topic. Some works such as [11] point out that to achieve high classification accuracy, it is not necessary for sensors to work at full sampling rate. The paper [16] statically addresses the tradeoff between accuracy and energy consumption. For both time and frequency domain features, it reports that in the sampling rate dimension an accuracy knee exists. Below the knee, there is a significant accuracy degradation. It suggests using the sampling rate at the knee to save energy while enjoying a relative high accuracy. In addition, Kobe [8] tries to achieve an optimal energy-latency-accuracy tradeoff for mobile sensor data classification. In contrast, AdaSense proposed in this paper considers the situation in which users wearing on-body sensors have an explicit accuracy requirement and uses single activity event detection sampling rate to reduce system energy overheads without violating user accuracy requirements. Compared to A3R [27], an independent and very recent work that also uses a lower sampling rate of single activity event detection to reduce system energy overheads, AdaSense explores the feature set space by Genetic Programming techniques and finds the optimal feature set that effectively reduces both the classification and detection sampling rates. In addition, AdaSense achieves more fine-grained sampling rate adaption instead of only utilizing the 4 fixed sampling rates ($5Hz$, $16Hz$, $50Hz$, and $100Hz$) exploited by A3R.

Some works have been proposed to adjust sensors sampling rate in different contexts. EmotionSense [23] uses declarative rules to adapt the sampling length of phone sensors. However, it does not consider reducing sampling rate. SpeakerSense [19] uses a low sampling rate to detect whether a speaker exists. It switches to a high sampling rate when a speaker is detected. Similarly, SociableSense [22] lets the sensors operate at a high sampling rate only when interesting events happen. If there are no interesting events, the sensors are put to operate at a low sampling rate. Although both SpeakerSense and SociableSense use different sensors sampling rates for different contexts, AdaSense is different from the following two aspects: first, instead of binary context categories such as interesting/non-interesting events, AdaSense uses more fine-grained context categories such as walking or lying down and hence enjoys more fine-grained sampling rate adaption; second, AdaSense quantifies the optimal sampling rate for each context category when considering user accuracy requirements.

Instead of controlling the sampling rate, many works achieve energy savings through sensor cluster selection. WolfPack [13] implements an online sensor clustering protocol that pursues high sensing confidence. Works such as [24] express the accuracy of context estimation through a quality of inference (QoINF) function that captures the dependence of estimation accuracy on the selected sensors. With QoINF, they develop a heuristic algorithm to calculate the sensor set which maximizes the estimation accuracy.

Some works save energy through optimizing sensor duty cycles. Systems using either state-related (EEMSS [26]) or context-based (SeeMon [12]) sensor subset are representatives of this group. Mercury [18] reduces energy consumption by disabling and enabling sensors dynamically. In terms of saving energy, AdaSense is orthogonal to both the duty cycling methods and the aforementioned sensor cluster selection methods.

III. Motivation

In this section, we demonstrate how to reduce sensors sampling rate without sacrificing user accuracy requirements. We illustrate how accuracy changes as the sampling rate varies. For this purpose, we use the Opportunity dataset [4] that contains naturalistic human activities recorded in a sensor rich environment. It comprises 72 sensors of 10 modalities, deployed in physical environments and on human bodies. It also includes an annotated dataset of complex, interleaved, and hierarchical naturalistic activities, with a par-

ticularly large number of atomic activities (around 30,000). Data is collected from 3 subjects with 4 locomotion activities (standing, walking, sitting, and lying down), labeled when being recorded and later reviewed by at least two different persons. Since this paper aims to reduce sampling rate for on-body sensing, we only use the data recorded by on-body sensors in the Opportunity dataset. We select 8 sensors (4 triaxial accelerometers and 4 triaxial gyroscopes) with the maximal sampling rate of $30Hz$.

In experiment, all sensor data is grouped by one second window, and statistical features (such as mean, variance, etc.) are extracted at the full sampling rate ($30Hz$). Then, the feature selection algorithm based on the Sequential Forward Selection (SFS) strategy [6] is applied to these features. The selected features are used to train a classifier. We use the Support Vector Machine (SVM) with RBF Kernel [6] as the classifier. It is worth of emphasizing that which classifier and which feature selection algorithm we use are not the novel parts of this paper. We have tried several other classifiers (such as the decision tree and Artificial Neural Network) and several other feature selection algorithms (such as the Sequential Backward Selection and Random Selection algorithms). The combination of SVM and SFS strategy based feature selection algorithm achieves the highest accuracy among all the combinations of classifier and feature selection algorithm we have tried. In addition, we choose SVM because [6]: (i) it is able to achieve high accuracy with a relatively small number of training examples; (ii) it scales well with data dimensionality; (iii) it is in a simple form and hence fast to execute in runtime; (iv) it is one of the best classifiers and has been successfully applied in activity recognition [16]. We choose RBF kernel since it achieves the highest accuracy compared to other kernels we have tried, such as polynomial kernel.

The selected features are extracted at each sampling rate from the data grouped by one second window and used as training and testing data at that sampling rate. We obtain the accuracy at each sampling rate following the 10-fold cross-validation routine [6]. In each round of cross validation, all data is divided into 10 subsets, 9 of which are used for training and the remaining 1 is used for testing, so that the testing data is different from the training data. We balance the number of instances for different activities in the training data. This process is repeated 10 times and each of the 10 subsets is used exactly once as the testing data. The estimated accuracy at each sampling rate is the average testing accuracy over 10 rounds. We calculate the accuracy at each sampling rate for single activity event detection by (true positive + true negative)/(true positive + false positive+ true negative + false negative). Figure 1 illustrates the accuracies of both multi-activity classification and single activity event detection for subject one with different sampling rates. Here, we only give the detection accuracies of sitting and lying down. The results for the other two activities (walking and standing) are similar.

In Figure 1, the curve with circle points summarizes the accuracies of doing multi-activity classification for the four activities with different sensors sampling rates. We first observe that the accuracy decreases as the sampling rate decreases and is also stable until the sampling rate drops below a threshold. For example, in Figure 1 the accuracy is above 93% until the sampling rate decreases to $24Hz$. This observation is consistent with what has been reported in [16]. Second, we observe that different accuracy requirements have different optimal (or the minimal necessary) sampling rates for multi-activity classification. In Figure 1, for instance, the optimal sampling rates for achieving 93% and 90% accuracies are $25Hz$ and $6Hz$, respectively. This observation tells us that, to save energy, different sampling rates can be used to meet different user-specified accuracy requirements.

In Figure 1, the curves with cross points and square points illustrate the detection accuracies of the sitting and lying down activities, respectively. We observe that the detection accuracy is always greater than the classification accuracy at each sampling rate. This observation motivates us to present the following Lemma:

Lemma 1. *In an activity recognition system, suppose the set of features are fixed. At any sampling rate, a classifier is trained with the features extracted at this sampling rate. Then, the classifier accuracy of any activity event detection is always greater than or equal to that of multi-activity classification.*

Proof: Suppose N is the number of instances being classified and A is the activity set. We fix the sampling rate and set of features. A multi-activity classifier is trained with the features extracted at the current sampling rate. It is worth emphasizing that only one classifier exists at the current sampling rate in our problem setting and we use the classifier to do both multi-activity classification and single activity event detection. The classifier works as an *activity-specific classifier* when it is used to perform the event detection for any single activity in A. Thus, every activity in A has an activity-specific classifier at the current sampling rate.

We use TP, FP, TN, FN to denote the instance sets of True Positive, False Positive, True Negative, False Negative in the classification result, respectively. For the event detection of a single activity, a, the accuracy is:

$$\frac{(TP_a + TN_a)}{(TP_a + FP_a + TN_a + FN_a)} \tag{1}$$

For multi-activity classification, the total accuracy is [9]:

$$\frac{\sum_{a \in A} TP_a}{N} \tag{2}$$

For an activity a, $TN_a \geq \sum_{a' \neq a \&\& a' \in A} TP_{a'}$ since for every instance of the other activities $(A \setminus a)$ that is correctly

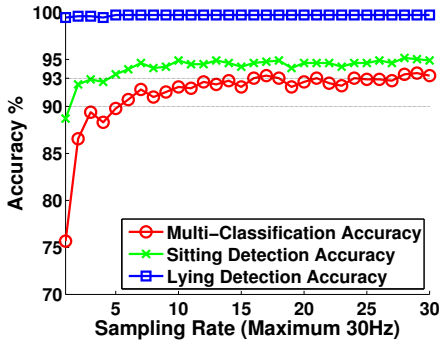

Figure 1. Accuracies of sitting and lying down event detections as well as multi-activity classification vs Sampling rate.

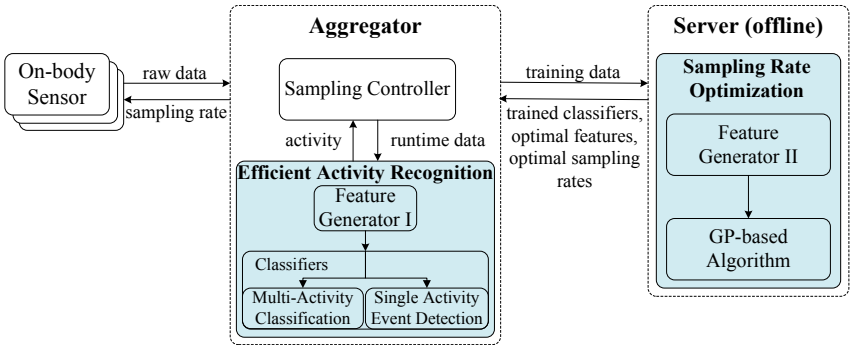

Figure 2. AdaSense architecture. **Note:** The sampling rate optimization and classifier training are performed on the server offine. No communication occurs between aggregator and server in runtime.

classified, it should be an instance in TN_a. In addition, TN_a also contains the misclassified instances among the other activities.

Thus, $(TP_a + TN_a) \geq \sum_{a \in A} TP_a$ (1).

Since $(TP_a + FN_a)$ is the number of instances of activity a and $(FP_a + TN_a)$ is the number of instances of the other activities.

Thus, $(TP_a + FP_a + TN_a + FN_a = N)$ (2).

With (1) and (2), the Lemma is proved. ■

Also, previous works such as [14] have already obtained experimental results that support (but not prove) this Lemma.

Henceforth, we refer to the optimal sampling rate of multi-activity classification under a certain accuracy requirement as ***classification sampling rate***. In Figure 1, we can see the classification sampling rate is $25Hz$ under 93% accuracy requirement. In the same way, we refer to the optimal sampling rate of single activity event detection under a certain accuracy requirement as ***detection sampling rate***. For example, the detection sampling rate for sitting is $5Hz$ under 93% accuracy requirement (See Figure 1). A ***corollary*** of Lemma 1 is:

Corollary 1. *Given a feature set, the classification sampling rate is no smaller than the detection sampling rate of any activity.*

The following questions arise from the corollary: (1) Can we combine multi-activity classification with single activity event detection in the activity recognition system to reduce sensors sampling rate? (2) If we can, to amplify the sampling rate reduction, can we develop a method that reduces both the classification sampling rate and all activities' detection sampling rates under any accuracy requirement? According to the corollary, the classification sampling rate is an upper bound of the detection sampling rates. Heuristically, when the classification sampling rate is reduced, the detection sampling rates could also be reduced *potentially*. Thus, it is reasonable to rewrite this question as: can we find a method that reduces the classification sampling rate?

To sum up, we need a design that exploits the above corollary and answers the above two questions. The BSN activity recognition system can benefit from such a design in terms of reducing sensing and communication workload while at the same time meeting user-specified accuracy.

IV. AdaSense Design

Motivated by the observations and questions discussed in Section III, we propose AdaSense, a framework that aims to reduce the sensors sampling rate in activity recognition systems. Its architecture is depicted in Figure 2, where the core components are highlighted in the shaded areas.

The *Efficient Activity Recognition* (EAR) is located on the aggregator (e.g., a smartphone). It exploits the corollary in Section III and performs either lower power single activity event detection or higher power multi-activity classification. In runtime, EAR periodically decides whether the current activity changes or not through single activity event detection with a lower sampling rate. When a change is detected, EAR classifies the new activity through multi-activity classification with a higher sampling rate. After classifying the new activity, EAR switches back to event detection of the new activity with a lower sampling rate.

To reduce the classification sampling rate as well as the detection sampling rates, EAR uses the optimal feature set and the classifier set trained on this feature set, which are generated by the *Sampling Rate Optimization* (SRO) on the server in the offline training phase. *In runtime, no communication occurs between the aggregator and server.*

In order to get the optimal feature set, SRO on the server obtains the training data from the aggregator and uses Feature Generator II to generate the basic statistical features. For energy concerns, the aggregator sends the training data to the server only when it is plugged into a power source. Then, based on these features, SRO utilizes a novel variant of Genetic Programming (GP) [15] to generate an optimal feature set, which effectively reduces the classification sampling rate under any accuracy requirement.

With the optimal features, SRO linearly searches the classification sampling rate and the detection sampling rates

and sends them to the Sampling Controller on the aggregator. The *Linear Search* means: (i) starting from the full sampling rate and following its decreasing order; (ii) estimating the accuracy following the 10-fold cross validation routine described in Section III; (iii) selecting the sampling rate i as the optimal one if the resulting accuracy when using i can meet the requirement but the resulting accuracy when using $i-1$ cannot. The linear search stops after the optimal sampling rate is selected.

With the optimal features and sampling rates, SRO trains the classifier set, which contains a multi-activity classifier for multi-activity classification and activity-specific classifiers for each single activity event detection. The multi-activity classifier is trained with the optimal features extracted at the classification sampling rate. Each activity-specific classifier is trained with the optimal features extracted at the activity's detection sampling rate. SRO passes the optimal feature set and the trained classifier set to the EAR on the aggregator.

The Sampling Controller provides data for both the training and runtime phases. Under its control, sensors sample at the system default (usually full) rate in the training phase and at the optimal rates in the runtime phase. Feature Generator I extracts the optimal features from raw data and feeds them to the classifier that is currently used in runtime. Sampling Controller ensures that data is sampled at the correct rate. All sensors used by the optimal feature set should be active.

A. Efficient Activity Recognition

The *Efficient Activity Recognition* (EAR) aims to exploit the corollary of the proved Lemma by effectively combining lower power single activity event detection and higher power multi-activity classification to save energy.

At the very first time, EAR uses the multi-activity classifier to identify the current activity with a higher sampling rate and informs Sampling Controller of the current activity. When the current activity does not change, Sampling Controller controls sensors sampling at the current activity's detection sampling rate, i.e., the optimal sampling rate of current activity's event detection. The single activity event detection is performed by the current activity-specific classifier periodically with the optimal features, which are extracted from the most recent raw data by Feature Generator I and ensure that the detection meets user accuracy requirements. To minimize the number of false positive reports for activity change, EAR determines that the current activity has changed when at least four out of five (an experimentally chosen value) continuous event detection results of the current activity are negative.

When the current activity changes, EAR first informs Sampling Controller of the detected change. Then, Sampling Controller increases the sampling rate to be the classification sampling rate, i.e., the optimal sampling rate of multi-activity classification. After collecting raw data for adequate time (one second is enough for the datasets in our evaluation), Feature Generator I extracts the optimal features from the newly sampled data, with which the multi-activity classifier classifies the new activity. After classifying the new activity, EAR switches back to event detection of the new activity. Again, the optimal features ensure that the classification meets user accuracy requirements.

It is necessary to point out that EAR introduces a time delay for detecting an activity change and also classifying the new activity. This delay includes the data collection time for both detection and classification. Here, we argue that this time delay is acceptable. It is because we find that a short time period (e.g., 7.12 seconds on average for the smartphone dataset) is enough to achieve high accuracy (e.g., 87.38% on average for the smartphone dataset). This delay is shorter than that of 10 seconds in the previous work [14].

B. Sampling Rate Optimization

With EAR, activity recognition system works at a lower sampling rate most of the time while not violating user accuracy requirements. If we further reduce both the classification sampling rate and the detection sampling rates, we can amplify the sampling rate reduction brought by EAR. As stated in the corollary of Lemma 1, under the same accuracy requirement, the classification sampling rate is an upper bound of the detection sampling rates. Heuristically, if we effectively reduce the classification sampling rate, the detection sampling rates also have the *potential* of being reduced. To achieve that purpose, we design the *Sampling Rate Optimization* (SRO).

We notice that the classification sampling rate depends on the feature set, upon which the classifier is built. Under a user accuracy requirement, we call a feature set that has enough discriminative capacity an ***adequate feature set***. To reduce the classification sampling rate is equivalent to find an adequate feature set that has a low classification sampling rate. Therefore, the design challenge of SRO is transformed to exploring the possible feature set space, searching an adequate one that has a low classification sampling rate. To address the challenge, we design the following two components.

Feature Generator II is on the server side (Figure 2). From the aggregator, it receives training data that is sampled at full sampling rate and extracts the basic statistical features such as mean, variance, median, etc.

Optimization Algorithm Design. The design goal of SRO is to develop a method that explores the feature set space with the objective of minimizing the classification sampling rate. When it comes to exploring a feature space, Genetic Programming (GP) [15] is a good choice. It is an evolution-based approach and usually used to find approximate solutions for hard optimization problems. It starts from an initial generation composed of a certain number of individuals, and then stochastically transforms the current generation into a new and improved generation. Although it

cannot guarantee that the results are optimal, GP can escape traps which deterministic methods may be captured in [15].

Algorithm 1 GP-based Algorithm

Input: Initial generation I, a user specified accuracy requirement A, and the total number of generations G.
Output: The optimal feature set (OFS), the optimal sampling rates (OSR), and the classifier set C that is trained upon OFS.

1: /*$BEST$ stores the best individual set in each generation*/
   $BEST$ = the best individual set that is selected by the feature selection algorithm from the initial generation, I.
2: $OFS = BEST$ /*OFS stores the individual set with the minimal classification sampling rate so far.*/
3: Linearly search the classification sampling rate of the OFS under the accuracy requirement A
4: **if** A cannot be met **then**
5: /*Do best efforts when the required accuracy can not be met*/
   A = the maximal accuracy that OFS can achieve
6: **end if**
7: SR = the classification sampling rate of OFS under the accuracy requirement A /*SR stores the classification sampling rate of OFS*/
8: Assign the reciprocal of OFS's classification sampling rate as the fitness value of the individuals in $BEST$ and 0 as the fitness value of the individuals not in $BEST$
9: **for** g=2 to G **do**
10: Use the individuals with higher fitness values to create a new generation through recombination and mutation [10]
11: $BEST$ = the best individual set that is selected by the feature selection algorithm from the current generation
12: Linearly search the classification sampling rate of $BEST$ under the accuracy requirement A
13: **if** $SR > BEST$'s classification sampling rate **then**
14: $OFS = BEST$
15: $SR = OFS$'s classification sampling rate
16: **end if**
17: Assign the reciprocal of $BEST$'s classification sampling rate as the fitness value of the individuals in $BEST$ and 0 as the fitness value of the individuals not in $BEST$
18: **end for**
19: Linearly search the detection sampling rates, DSR, of all activities for OFS while the detection accuracy $\geq A$
20: $OSR = \{SR\} \cup \{DSR\}$;
21: Train the multi-activity classifier with OFS extracted at the classification sampling rate
22: Train each activity-specific classifier with OFS extracted at the activity's detection sampling rate
23: All classifiers form a classifier set C
24: return (OFS, OSR, C)

In GP, individuals (or computer programs) are the functions that extract features from the time series sensor data. They are encoded as tree structure with the non-leaf nodes being mathematical operations and the leaf nodes being the functions that extract basic features. The tree is calculated from the bottom up. At the bottom, the leaves extract the basic features. After that, the non-leaf nodes are executed level by level until the root is reached. Finally, the root's calculation result is the value of the feature represented by the tree (or the individual). The most important two steps of GP are the validation and survival steps [25]. In the validation step, GP evaluates individuals' fitness and assigns higher values to those with better fitness. In the survival step, GP ranks all individuals' fitness and selects those with higher ranks as the parents of the next generation.

However, our goal is slightly different from the traditional aim of GP. Instead of finding an optimal individual, we aim to develop an algorithm that can find an optimal feature set (or individual set). Thus, we design a variant of GP and describe it in Algorithm 1, which runs on the server.

In the algorithm, one set of individuals is said to be better than the other when it has more discriminative capacity. Different from traditional GP, in the validation step of our algorithm, feature selection [17] is used to select the best individual set for each generation. The algorithm assigns the reciprocal of the selected set's classification sampling rate (found by the linear search method stated in Section IV) as the fitness value of all individuals in the selected set. In addition, the algorithm assigns 0 as the fitness value of the individuals not in the selected set. In the survival step, individuals with higher fitness values are elected and used to create next generation through recombination and mutation [10]. SRO runs this algorithm with the following inputs: the initial generation (all the basic statistical features, I), user accuracy requirement (A), and a given number of total generations (G).

V. Evaluation

In this section, we evaluate AdaSense's performance for sampling rate reduction on the Opportunity dataset [4] and a smartphone dataset we collected from eight subjects. With the smartphone dataset, we also evaluate AdaSense's energy consumption and compare AdaSense's energy consumption with that of one state of the art sampling rate reduction methods [27] through power measurement and energy emulation. The classifier we use in the evaluation is the SVM with RBF kernel [6]. The classifier set is trained for each subject as indicated in Algorithm 1. As stated in previous sections, we follow the 10-fold cross-validation routine to obtain classifier accuracy. GP-based algorithm is implemented upon GPLAB [25]. To limit the complexity of each individual (or tree) in the algorithm, we set the height limit of each individual to be 3. We also set the number of individuals in each generation to be 100, which is large enough for the algorithm to obtain promising results.

A. Performance of On-body Sensing

In this subsection, we demonstrate the performance of AdaSense for sampling rate reduction on on-body sensors with the Opportunity dataset [4]. In the evaluation, we use the same accelerometers and gyroscopes mentioned in Section III and group the data with one second window. The basic statistical features (extracted by Feature Generator II in Figure 2) we consider are the max, min, max-min, mean, var, median, zero crossing rate, number of peaks, values of the histgram with 10 bins, and interquartile range. We select the best feature set from all these features and call it the Basic Feature Set (BFS), whose results will be compared with what AdaSense's optimal feature set achieves.

GP-based Algorithm Convergence. Figure 4 demonstrates how fast GP-based algorithm converges to the optimal classification sampling rate. In the experiments, we run the algorithm for three accuracy requirements on different

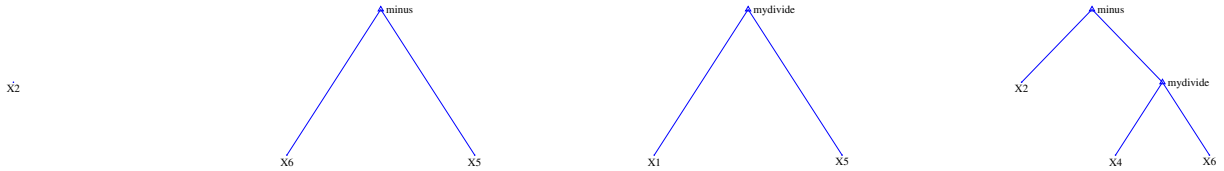

Figure 3. The optimal features for subject one. Xis are the basic statistical features. X1: the mean of Back-AccelerometerX (AccX); X2: the (max-min) of Back-AccY; X4: the zero crossing rate of LWR-AccY; X5: the Hist10(4) of LWR-AccY; X6: the Hist10(3) of LShoe-AccY. Hist10(i) represents the value of ith bin of the histgram with 10 bins.

Activities	Lying Down			Sitting			Standing			Walking		
Subjects	1	2	3	1	2	3	1	2	3	1	2	3
BFS Classi. SR (Hz)	24	30	25	24	30	25	24	30	25	24	30	25
BFS Detection SR (Hz)	1	4	1	5	4	2	3	6	11	9	8	6
SRO Optimal Classi. SR (Hz)	3	12	9	3	12	9	3	12	9	3	12	9
SRO Optimal Detection SR (Hz)	1	1	1	2	1	2	3	3	5	3	4	5

Table I
THIS TABLE SUMMARIZES THE SENSORS SAMPLING RATES FOR ACHIEVING BFS'S MAXIMAL ACCURACY WHEN DIFFERENT SAMPLING STRATEGIES ARE USED. THE BFS'S MAXIMAL ACCURACIES OF SUBJECT ONE, TWO, AND THREE ARE 93%, 91%, AND 93%. SR STANDS FOR SAMPLING RATE.

numbers of generations (20, 40, 60, 80, and 100). To obtain each data point on the curves, we run the algorithm 5 times on each subject's data and average the results over all runs and subjects. From this figure, we see that 60 generations are enough for GP-based algorithm to converge on the optimal classification sampling rates under different accuracy requirements. After 60 generations, no obvious improvement is found.

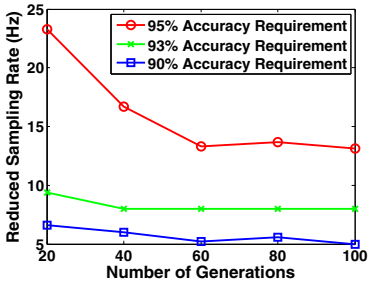

Figure 4. Optimal sampling rate vs. Number of generations. Different accuracy requirements are considered.

From Figure 4, we also observe that a higher accuracy requirement converges at a higher classification sampling rate. For example, for the accuracy requirements of 90%, 93%, and 95%, the average classification sampling rates converge at $5Hz$, $8Hz$, and $13Hz$, respectively. It is because activity recognition is harder when the accuracy requirement is higher. To meet the increased accuracy requirement, it is necessary to collect more information by increasing sampling rates.

Sampling Rate Reduction. To obtain the sampling rate reduction results for each subject in the dataset, we set each subject's accuracy requirement to the maximal accuracies (93%, 91%, and 93%) achieved by BFSs at high sampling rates. This requirement demonstrates how our GP-based algorithm can generate a feature set that achieves the maximal accuracy of BFS while at the same time effectively reduces both the classification sampling rate and the detection sampling rates. We run our GP-based algorithm for 60 generations for each subject and illustrate subject one's optimal feature set in Figure 3.

There are four features in the set. All the features are in the form of trees whose heights are equal to or smaller than 3. The leaves of each tree are the basic statistical features extracted from on-body sensors. The non-leaf nodes are the arithmatic operations applied between their child nodes. The optimal features depend on three on-body sensors (Back, LWR, LShoe), whose deployment positions are listed in [2]. The optimal feature sets of the other two subjects have similar number of features, whose heights are also equal to or smaller than 3. Since the number of optimal features is small and the structure of them is not complex, they can be easily extracted in real-time with limited computational cost.

We summarize the sampling rate reduction results of each subject in Table I. From Table I, we first observe that single activity event detection requires a lower sampling rate than multi-activity classification. For instance, in order to determine the sitting activity with 93% accuracy for subject one using BFS, the classification sampling rate is $24Hz$ while the detection sampling rate is only $5Hz$. The experimental results agree with the corollary of Lemma 1. Compared to the method that only uses the classification sampling rate [16], the results demonstrate that EAR reduces the sampling rate on average in activity recognition systems by combining lower power single activity event detection and higher power multi-activity classification.

From Table I, we also observe that SRO further reduces both the classification sampling rate and detection sampling rates for each subject. For example, with BFS, the detection sampling rates of sitting and walking for subject one are $5Hz$ and $9Hz$, respectively. With the optimal feature set, the detection sampling rates are further reduced to $2Hz$ and $3Hz$, respectively. Thus, the sampling rate reduction brought by EAR is amplified by SRO.

Therefore, from Table I, we conclude that the optimal feature set generated by GP-based algorithm enjoys both lower classification and detection sampling rates, which result in lower sensing and communication workloads for on-body sensor nodes and hence reduce energy overheads. Moreover, the optimal feature sets guarantee the system is

Activities	Lying Down				Sitting				Standing				Running				Walking				Cycling			
Subjects	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
BFS Classi. SR (Hz)	18	15	19	15	18	15	19	15	18	15	19	15	18	15	19	15	18	15	19	15	18	15	19	15
BFS Detection SR (Hz)	4	3	4	1	6	9	9	7	10	7	7	6	7	6	7	5	8	5	6	7	6	6	8	7
SRO Optimal Classi. SR (Hz)	9	10	12	8	9	10	12	8	9	10	12	8	9	10	12	8	9	10	12	8	9	10	12	8
SRO Optimal Detection SR (Hz)	2	1	2	1	3	5	6	3	5	4	4	3	4	6	3	2	4	2	4	5	3	4	5	5
Subjects	**5**	**6**	**7**	**8**	**5**	**6**	**7**	**8**	**5**	**6**	**7**	**8**	**5**	**6**	**7**	**8**	**5**	**6**	**7**	**8**	**5**	**6**	**7**	**8**
BFS Classi. SR (Hz)	22	20	17	19	22	20	17	19	22	20	17	19	22	20	17	19	22	20	17	19	22	20	17	19
BFS Detection SR (Hz)	2	1	3	2	8	9	8	6	11	10	9	8	7	9	8	6	10	8	7	6	12	6	11	7
SRO Optimal Classi. SR (Hz)	11	10	9	7	11	10	9	7	11	10	9	7	11	10	9	7	11	10	9	7	11	10	9	7
SRO Optimal Detection SR (Hz)	1	1	3	2	4	3	4	3	6	5	4	5	3	3	4	2	4	5	2	4	5	6	5	5

Table II
THIS TABLE SUMMARIZES THE SENSORS SAMPLING RATES FOR ACHIEVING BFS'S MAXIMAL ACCURACY WHEN DIFFERENT SAMPLING STRATEGIES ARE USED. THE BFS'S MAXIMAL ACCURACIES OF SUBJECT 1, 2, 3, 4, 5, 6, 7, AND 8 ARE 85%, 82%, 88%, 85%, 91%, 89%, 87%, AND 92%. SR STANDS FOR SAMPLING RATE.

accurate, since they are generated under the accuracy requirements of the maximal accuracies (93%, 91%, and 93%) achieved by BFS. Although we cannot directly measure the energy savings of AdaSense on the Opportunity dataset, we will explicitly demonstrate how AdaSense saves energy on smartphone sensing in next subsection.

B. Performance of Smartphone Sensing

Up to now, we have been presenting how AdaSense reduces sensors sampling rate for on-body sensors in BSNs. In this subsection, we emphasize that it also potentially applies to smartphone sensors, which are used for activity recognition. We evaluate the performance of AdaSense on reducing smartphone sensors sampling rate and sensing energy consumption. In this setting that only includes smartphone, there is no communication between on-body nodes and aggregator (See Figure 2).

Data Collection. In data collection, we use the accelerometer on HTC Google Nexus One Android smartphone and collect its readings when eight subjects (3 females, 5 males) performing different activities. The subjects put the phone at either their left or right pants pockets. The direction of phone head in each subject's pocket is random. The subjects are instructed to perform each of the six activities (sitting, lying down, standing, walking, cycling, and running) for 30 minutes. During each activity, each subject is allowed to perform other jobs such as reading papers and operating computers. All subjects perform walking, running, and cycling with the speeds they wish, there is no special speed requirement. We concatenate the data of each activity in the way that all activities interleave with each other with 5 minutes period. During each activity, we record the triaxial readings of the accelerometer, which operates at $30Hz$. There are three columns in the raw readings. We group the raw readings with one second window and use the basic statistical features (the max, min, max-min, mean, var, median, zero crossing rate, number of peaks, values of the histgram with 10 bins, and interquartile range), which are extracted by Feature Generator II in Figure 2, as the input to GP-based algorithm.

In data collection, we only consider the modality of accelerometer because the other method [27], which we plan to compare AdaSense with in terms of energy consumption, only uses accelerometer readings.

Sampling Rate Reduction. Similar to the evaluation of on-body sensing, we select the BFS for each subject and use the maximal accuracy each BFS achieves (85%, 82%, 88%, 85%, 91%, 89%, 87%, and 92%) at high sampling rates as the accuracy requirements input to GP-based algorithm. In addition, we use 60 as the input of the total generation to GP-based algorithm. Compared to the accuracy achieved by CenceMe [20] when the phone was deployed in subjects' pockets, BFS of smartphone sensors we obtain achieves slightly higher accuracy for more activities because: CenceMe only considers limited number of features (the mean, standard deviation, and number of peaks of the accelerometer readings), but BFS contains the best features selected from a larger feature pool (19 features in total). Compared to the results in the on-body sensing subsection, BFS of smartphone sensors achieves lower accuracy because: (i) on-body sensors are deployed at different positions of human body, capturing more information of human activities; (ii) we use two sensing modalities (the accelerometer and gyroscope) of on-body sensors, but we only use one sensing modality (the accelerometer) of Google Nexus One; (iii) smartphone sensors are loosely tied to human body compared to on-body sensors.

We summarize the classification sampling rate and each activity's detection sampling rate of BFS and SRO optimal feature set in Table II. From the table, we observe that with either BFS or SRO optimal feature set, the classification sampling rate is always larger than the detection sampling rates. Thus, we can conclude that EAR reduces the sampling rate for activity recognition on average by combining lower power single activity event detection and higher power multi-activity classification. From the results, we also observe that SRO further reduces both the classification sampling rate and detection sampling rates. For example, for subject one with BFS, the detection sampling rate of standing is $10Hz$. With the SRO optimal feature set, the detection sampling rate is $5Hz$, which is reduced by 50%. Thus, from both Table I and II, we can conclude the heuristic strategy taken by GP-based algorithm, which reduces detection sampling rates through reducing classification sampling rate is sound. Aside from sampling reduction, the SRO optimal feature set is generated under the accuracy requirement of BFS's maximal accuracy. Thus, the SRO optimal feature set also guarantees the system is accurate.

Energy Savings. With the data collected from eight subjects,

we compare the energy consumption of AdaSense with that of the most recent sampling rate reduction method in [27]. Our evaluation settings are described as follows: (i) The method in [27] uses the maximal sampling rate supported by Android ($100Hz$) with the time domain features listed in [27] for multi-activity classification. In addition, it uses one of four sampling rates ($5Hz$, $16Hz$, $50Hz$, and $100Hz$) with the time domain features listed in [27] for single activity event detection. For each subject, the sampling rate for each single activity event detection is obtained by the A3R framework in [27]. (ii) AdaSense uses the classification sampling rate of the SRO optimal feature set for multi-activity classification. In addition, AdaSense uses the detection sampling rates of the SRO optimal feature set for single activity event detections.

Figure 5. Energy Measurement Setup.

Sampling Rate (Hz)	Energy Consumed per hour (J)
1	1.90
2	3.01
3	5.76
4	6.78
5	10.62
6	14.16
7	19.85
8	22.22
9	26.48
10	32.22
16	51.16
20	53.10
50	81.20
100	327.42

Figure 6. Energy Consumption of Accelerometer per hour on Google Nexus One Under Different Sampling Rates.

To make the comparison fair, all the methods uses the same sensing modality (accelerometer) to recognize activities and the energy consumption of each method is obtained by the emulation method described in [27], which "*uses the sequence of system state transitions observed and multiplies each such activity duration by the corresponding energy power consumption per unit time*". AdaSense controls the sensors sampling rate through the Android API *android.hardware.SensorManager.registerListener()*. We use the Monsoon Power Monitor [1] to measure the energy consumption of accelerometers per hour on Google Nexus One under different sampling rates (Figure 5). In the energy consumption measurements, we turn off the network interfaces and keep the system all the same except for the sampling rate. We use the difference between the energy consumption when the accelerometer is on with a certain sampling rate and that without operating sensor as the energy consumption for that sampling rate. In Figure 6, we list measurement results for the sampling rates that are frequently used in the energy consumption comparison and do not explicitly list the results of other sampling rates to save space. With the measurement results, we calculate the energy consumption per second (i.e., power) of different sampling rates for emulation. In addition, we also measure that the average power consumption for extracting features and running classifier on Google Nexus One each time is 3.5mw and the average running time for each feature extraction and classification is 12.7ms.

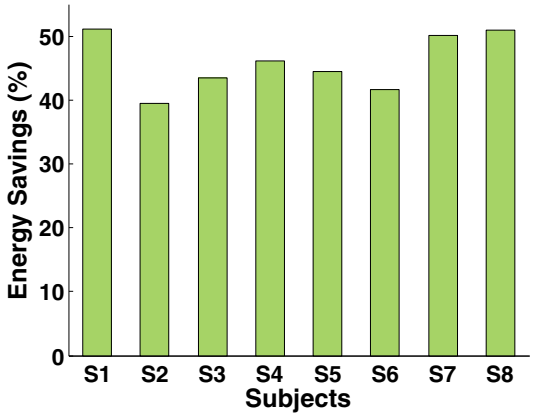

Figure 7. Energy savings compared to the method in [27].

With these measurement results, we emulate the energy consumption for each method. The emulation results indicate AdaSense saves energy compared to the method in [27]. The energy saving results are illustrated in Figure 7. Compared to the method in [27], the energy saving of AdaSense ranges from 39.4% to 51.0%. Although the method (A3R) in [27] similarly uses detection sampling rate of single activity to reduce system operating sampling rate, AdaSense outperforms it because: (i) for single activity event detection, AdaSense achieves more fine-grained sampling rate adaption instead of only utilizing the 4 fixed sampling rates ($5Hz$, $16Hz$, $50Hz$, and $100Hz$) exploited in [27]; (ii) A3R uses data sampled at $100Hz$, which consumes more energy compared to other sampling rates used by AdaSense, to perform multi-activity classification; (iii) AdaSense explores the feature set space by genetic programming techniques and finds the optimal feature set that effectively reduces both the classification and detection sampling rates.

False activity change detection and activity misclassification result in system instability and hence compromise energy savings and introduce extra delay for classifying new activity. However, AdaSense reduces the number of false activity change detection by the smoothing technique stated in Section IV. This technique determines the current activity having changed when at least four out of five continuous event detection results of the current activity are negative. Four out of five is an experimentally chosen value that keeps the activity change detection accuracy to be above 90%. From Figure 7, we can conclude that AdaSense achieves considerable energy savings with the help of this technique. For all eight subjects, the average time delay for classifying new activity is 7.12s, which is comparable to the delay (10s) in the most recent activity recognition system [14].

VI. Conclusion

In body sensor networks, sensor sampling and communication quickly deplete the limited battery energy. In this paper, we propose AdaSense that effectively reduces sensors sampling rate without violating user-defined accuracy requirements. Effective Activity Recognition in AdaSense reduces sensors sampling rate by turning multi-activity classification to single activity event detection most of the time. It only resorts to multi-activity classification when an activity change is detected. Sampling Rate Optimization in AdaSense further reduces the optimal sampling rates of both multi-activity classification and single activity event detection by utilizing a novel Genetic Programming based algorithm. With the Opportunity dataset and the smartphone dataset collected from eight subjects, AdaSense is demonstrated to effectively reduce both classification sampling rate and detection sampling rate. AdaSense is also demonstrated to save $39.4\%\sim51.0\%$ smartphone energy compared to one state-of-the-art solution.

References

[1] Monsoon power monitor. http://www.msoon.com/LabEquipment/PowerMonitor.

[2] Sensor positions. http://www.opportunity-project.eu/node/47.

[3] Fahd Albinali, Stephen Intille, and et al. Using wearable activity type detection to improve physical activity energy expenditure estimation. In *Proc. of Ubicomp '10*, pages 311–320.

[4] Michael Beigl and Francisco J. Cazorla-Almeida. Recording a complex, multi modal activity data set for context recognition. In *Workshop Proc. of ARCS '10*.

[5] Tessendorf Bernd, Bulling Andreas, and et al. Recognition of hearing needs from body and eye movements to improve hearing instruments. In *Proc. of Pervasive '11*.

[6] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[7] Octav Chipara, Chenyang Lu, and et al. Reliable clinical monitoring using wireless sensor networks: experiences in a step-down hospital unit. In *Proc. of Sensys '10*, pages 155–168.

[8] David Chu, Nicholas D.Lane, and et al. Balancing energy, latency and accuracy for mobile sensor data classification. In *Proc. of Sensys '11*.

[9] Russell G. Congalton. Review of assessing the accuracy of classifications of remotely data. *Remote sensing of the Environment*, pages 37:35–46, 1991.

[10] Kilian Förster, Pascal Brem, and et al. Evolving discriminative features robust to sensor displacement for activity recognition in body area sensor networks. In *Proc. of ISSNIP '09*, pages 43–48.

[11] Patrick S. Hamilton and E.P. Limited. *Open Source ECG Analysis Software Documentation*. 2002.

[12] Seungwoo Kang, Jinwon Lee, and et al. Seemon: scalable and energy-efficient context monitoring framework for sensor-rich mobile environments. In *Proc. of MobiSys '08*, pages 267–280.

[13] Matthew Keally, Gang Zhou, and et al. Exploiting sensing diversity for confident sensing in wireless sensor networks. In *Proc. of Infocom '11*, pages 1719–1727.

[14] Matthew Keally, Gang Zhou, and et al. Pbn: Towards practical activity recognition using smartphone-based body sensor networks. In *Proc. of Sensys '11*.

[15] J.R. Koza. *Genetic Programming: On the programming of computers by means of natural selection*. MIT Press, 1996.

[16] Andreas Krause, Matthias Ihmig, and et al. Trading off prediction accuracy and power consumption for context-aware wearable computing. In *Proc. of ISWC '05*, pages 20–26.

[17] Huan Liu and Hiroshi Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[18] Konrad Lorincz, Borrong Chen, and et al. Mercury: a wearable sensor network platform for high-fidelity motion analysis. In *Proc. of SenSys '09.*, pages 183–196.

[19] Hong Lu, A. J. Bernheim Brush, and et al. Speakersense: Energy efficient unobtrusive speaker identification on mobile phones. In *Proc. of Pervasive '11*.

[20] Emiliano Miluzzo, Nicholas D. Lane, and et al. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proc. of SenSys '08*, pages 337–350.

[21] Xin Qi, Gang Zhou, Yantao Li, and Ge Peng. Radiosense: Exploiting wireless communication patterns for body sensor network activity recognition. In *Proc. of RTSS '12*, 2012.

[22] Kiran K. Rachuri, Cecilia Mascolo, and et al. Sociablesense: Exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In *Proc. of MobiCom '11*.

[23] Kiran K. Rachuri, Mirco Musolesi, and et al. Emotionsense: a mobile phones based adaptive platform for experimental social psychology research. In *Proc. of Ubicomp '10*, pages 281–290.

[24] Nirmalya Roy, Archan Misra, and et al. An energy-efficient quality adaptive framework for multi-modal sensor context recognition. In *Proc. of PerCom '11*, pages 63–73.

[25] Sara Silva and Jonas Almeida. Gplab-a genetic programming toolbox for matlab. In *In Proc. of the Nordic MATLAB Conference (NMC-2003)*, pages 273–278, 2005.

[26] Yi Wang, Jialiu Lin, and et al. A framework of energy efficient mobile sensing for automatic user state recognition. In *Proc. of MobiSys '09*, pages 179–192.

[27] Zhixian Yan, Vigneshwaran Subbaraju, and et al. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *Proc. of ISWC '12*.