UNIVERSITY OF CALIFORNIA
RIVERSIDE

Energy Efficient Sensing for Unsupervised Event Detection in Real-Time

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Electrical Engineering

by

Abdulrahman Bukhari

September 2019

The Thesis of Abdulrahman Bukhari is approved:

Committee Chairperson

University of California, Riverside

Acknowledgement

I would like to express my thanks to my thesis advisor, Professor Hyoseung Kim of Electrical and Computer Engineering Department at University of California – Riverside for his boundless guidance and support during all my research stages.

I would also like to thank my committee members Professor Amit K Roy Chowdhury and Professor Shaolei Ren for their valuable discussion and suggestions that provides guidance to my PhD research plan in the near future.

I also wish to thank the Electrical and Computer Engineering Graduate Students Advisor for her guidance and kind encouragement. She relentlessly answers all my questions and concerns throughout my degree.

I also thank my lab mates for their continues help: Yecheng Xiang, Yidi Wang, Mehdi Hossseini, Hyunjong Choi, and especially Mohsen Karimi. I would also like to thank my colleagues in the Electrical Engineering Department: Abdel-Rahman Al-Makdah, Yahya Sattar, Rakib Hyder, Sudipta Paul and Akbar Razah,

I would like to express my gratitude to the Saudi government and Umm Al-Qura University for sponsoring my Master of Science Degree in Electrical Engineering at University of California – Riverside and funding me that allowed me to pursue my graduate studies

Finally, I would like to express my warmest thanks and appreciation to my parents, especially my mother, for providing me with endless support and encouragement throughout my degree. I also thank my all my friends and family-in-law, especially my parents-in-law for their kind-hearted support. I also want to thank my baby son, Aws, who showed me that nothing is impossible. Lastly, I would like to give my heartiest thanks to the special person that stands by me, supports me throughout my entire journey and never failed to believe in me, my dearest wife, Aliyah.

ABSTRACT OF THE THESIS

Energy Efficient Sensing for Unsupervised Event Detection in Real-Time

by

Abdulrahman Bukhari

Master of Science, Graduate Program in Electrical Engineering
University of California, Riverside, September 2019
Dr. Hyoseung Kim, Chairperson

General-purpose sensing offers a flexible usage and a wide range of Internet of Things (IoT) applications deployment. In order to achieve a general-purpose sensing system that is suitable for IoT applications, several design aspects such as performance, efficiency and usability, must be taken into consideration. The work of this thesis is focusing on implementing an energy efficient general-purpose sensing system that is based on unsupervised learning techniques for events labeling and classification. The system clusters raw data collected from a variety of events, like microwave, kettle and faucet running, etc., for classification. During the training phase, the system computes sensing polling periods, based on the rate of change in classes, that are then feed into a dynamic scheduler implemented on the sensor board in order to reduce energy consumption. The system is deployed in a one-bedroom apartment for raw data collection, and system evaluation. The results show that the mean accuracy of event classification is 83%, and sensor data polling is reduced in average by 95%, which translates to 90% energy saving, compared to the fixed polling period in the state-of-art approach.

Table of Contents

List of Figures

I. Introduction

The Internet of Things (IoT) promises interaction of a variety of objects with each other enabling remote monitoring and control. One of many of IoT's essential aspects is sensing, wherein detecting changes in environmental facets enable tracking of the states of a certain event. For example, a change in light intensity in a room can determine the state of lighting as turned on or off. In addition to that, one of the most significant design considerations in IoT is energy consumption because most IoT devices are portable embedded electronic hardware requiring a long-lasting battery life.

In order to achieve sensing of various elements within an environment, several approaches have been studied. One approach is using single sensor for each sensing element. This approach can be robust in application with single objective such as detecting the state of lighting. However, in higher dimension applications that require sensing multiple facets, using single sensor approach is a complex and costly due to the fact that several sensors must be installed and integrated together to sense all the elements in the environment. Another alternative is using a single sensor for multiple events detection. This technique, however, requires that all events must be detected using one facet by wearing a sensor [2], or install it on a fixed place in a closed environment [3].

Similarly, a network of the same sensor can be installed around an environment to recognize activities related to one sensing element [4]. Although these approaches enable detecting a variety of events, they can be only deployed in single element sensing applications. Meaning, these techniques cannot be implemented in an environment that different events are represented by different sensing facets. General-Purpose sensing approach can be considered as the most effective approach for detecting states of different events that are sensed through different facets. An example of general-purpose sensing is a sensor board packed with various sensors that capture many sensing elements as raw data and then classify them into different categories [1].

As for power consumption reduction, two approaches can be considered. The first approach is by using a prediction model that can estimate sensor values, and the system uses these values instead of inquiring new readings periodically [5, 6]. As a result, the system will only request reading when there is a significant change in its value leading to lesser energy consumption. This approach can be only sufficient in case sensors values in a batch from a sensor network are correlated both temporally and spatially. The second approach is by changing sensor polling periods dynamically. In order to do so, the framework can maximize the polling period [7], or find an optimized period [8], for the scheduler without affecting the quality of the data. This approach is efficient when data changes periodically.

The thesis interduces an energy efficient sensing for event recognition using unsupervised learning techniques for labeling and classification. The design of the proposed framework mainly consists of two parts. First, the sensing part, motivated by the work done by Gierad et al. [1], wherein the sensor board collects raw data of three different environmental facets, instead of sensing twelve facets as purposed by [1], through a combination of three sensors. The raw data is then preprocessed and prepared for feature extractions. These features are fed into K-Means clustering algorithm [9] for labeling, analysis and event classification.

The second part of the system, which is the focus of this thesis, is reducing the energy consumption of the system by designing a new scheduler, motivated by Seonyeong's and his colleagues' work [8], that computes a dynamic polling periods for the sensor board. The scheduler assigns a different period for each event based on the rate of change in classes during the training phase. In case an event is detected, the system will enter idle mood for the duration of the assigned period since the recognized event is expected to remain unchanged until the period is expired. Therefore, the system will be in idle mood more frequently during events classification leading to reduction of the overall power consumption of the system.

The contributions of the thesis are:

- Implementation of a new dynamic scheduler based on the rate of change in classes for real-time event detection
- Two algorithms proposed to compute the polling period of the dynamic scheduler with latency guarantee
- Using unsupervised technique simplifies the labeling of raw data into labels without significant classification accuracy loss

The thesis has seven sections. The next section will review recent literatures related to the proposed framework, focusing on events sensing and sensors scheduling approaches. In section III and IV, the system framework is introduced and discussed thoroughly, specifically data acquisition, data preprocessing, feature extraction and classification, in addition to, the algorithms to find appropriate polling period for dynamic scheduler. Both classifier and scheduler are evaluated in section V. In the later section, discussion of findings, challenges and possible future work is presented, and then conclude.

II. Related Work

The work in this thesis combines two areas of research, sensing and energy consumption. These areas of research have been widely explored previously. Thus, the following subsections will discuss recent literatures on activity sensing and data acquisition scheduling.

A. Sensing and Classification

There have been many significant works done related to activity recognition based on different sensing approaches. One approach is using a single sensor is to detect several events. For example, an accelerometer has been used to recognize body activities, such as standing, walking and running [2]. This work is based on meta-level classifier, which is then compared to other base-level classifiers' performance. The meta-level classifier achieves high accuracy on average. However, inconsistent results showed across different experimentation set-ups, and some activities were hard to recognized and confused with other ones. This could be related the features selection, where only statistical features, such as mean, standard deviation, energy, etc., are used. Using advanced signal analysis tools, like the Fast Fourier Transform (FFT) could provide richer signal information improving the classification accuracy.

Similarly, the work done by Chen et al. [3] uses the sound detected by using a microphone to detect activities in the bathroom. Targeted activities include showering, toilet flushing, teeth brushing and hand washing. The system also provides additional information such as the date and time of use and the duration of each activities. The system evaluated in continuous real-time and achieved an average accuracy of 83%. The paper only considered Mel-Frequency Cepstral Coefficient (MFCC) method for features extraction and Hidden Markov Model (HMM) for classification, whereas there may exist better approaches, such as Support Vector Machine (SVM) as in [1] to achieve higher accuracy for supervised learning approach.

Another work that based on a single environmental facet sensing is ElectriSense [4] that can detect and classify the usage of electronic devices, which uses switch mode power supplies (SMPS), to analyze the electromagnetic interference (EMI) signals. By observing a power line, a new device operation will interduce a new EMI signal, and by subtracting the new signal with the baseline noise, features are extracted. Features are then fed into a K-Nearest Neighbor classifier for event detection. ElctriSense is able to classify the usage of electrical devices with an average accuracy of 93.82% from a single point at home. The single point application gives the user the freedom to place the sensor anywhere and still able to detect the usage of electrical devices across home. Although single sensor approach as in [2,3,4] achieves the application requirement, none qualifies as a general-purpose sensor because all events detected can be sense with a shared sensing element.

The sensor tag proposed in [1] achieves general-purpose sensing by combining nine different sensors that sense 12 environmental facets into a sensor tag. The collected raw data are classified into a wide range of events. The system framework extract features on board, leading to extensive computational cost, using FFT for high sampling rate sensors, such as the accelerometer, and statistical features for the low sampling rate ones, like temperature. The features are then sent to a server for more processing and assembling feature vector, which is fed into a machine learning model (SVM) for training and classification. The system supports two learning modules, manual learning (supervised learning) and automatic learning (unsupervised learning).

The sensor tag accuracy was evaluated based on 38 different events for seven days by collecting data on one day and test the classifier on the other. The average accuracy for supervised learning model was 96% while ranging from 88% to 30% for unsupervised learning. Synthetic Sensors shows promising results toward achieving general-purpose sensing with the ability to recognize a variety of events accurately. However, power consumption has not been studied in this work, nor other related sensing designs previously discussed, which motivated the sensor proposed in this thesis to explore energy efficient general-purpose sensing framework.

B. Data Acquisition Scheduling

Scheduling data aqusition in order to lower power consumption has been broadly discussed in IoT and real-time frameworks. Gedik et la. [5] introduced a dynamical data collection approach in a sensor network. By setting a subset of sensors as sampler for data collection where the values of the other subset of sensors (non-sampler) are predicted using probabilistic models. This way, the energy consumption in the sensor network is reduced and the quality of the data collection can remain high. Another proposed approach is integrating an adaptive enable/disable prediction method to reduce the energy consumption of sensor networks [6]. The framework can use several advanced features such sleep/awake scheduling. These two approaches are based on the predicting the value of the sensor using probability models. In other words, they require the data to be spatially and temporally correlated, which is not the case when in a sensing system that has few sensors or a single sensor.

The work by Han and his colleagues [7] suggests a new adaptive data collection method for different sensor networks models. Changing the state of the sensor from sleep mode to active requires more energy. So, changing the state rapidly consumes excessive energy in addition to unnecessary delay. Therefore, the transition time between states must be selected such that energy consumption and latency are optimized. This approach motivates the algorithm proposed to find an appropriate polling period such that the classification latency is limited.

The framework proposed in [8] analyses the sensor values, conditions and constraints to find a flexible polling interval for each sensor to achieve efficient sensing. One of the RT-IFTTT component is the sensor polling scheduler, which calculate the polling period for each sensor. It changes the period based on the rate of change in data and the probability of whether a triggering condition will occur soon or not. However, this framework is inefficient with events sensing applications because environmental facets of interest in this work changes slowly, e.g., temperature, humidity, etc., compared to vibration and sound captured by an accelerometer and a microphone running at high frequencies.

III. System Framework

The high-level architecture of the sensing system consists of two stages; data acquisition, and classification. There are sub-processes within these two stages and a data transfer medium in between. This section will interduce the system framework in details, followed by the data acquisition stage and the communication protocol. The server side will then be explored in depth, where raw data are prepared for classification. The proposed scheduler computes the dynamic polling periods based on the speed of change in classes.

A. Sensor Architecture

The system framework consists of a sensor board and server with a Bluetooth Low Energy (BLE) layer in between as shown in figure 1. The main microcontroller of the sensor board is from Texas Instrument, the CC2640R2 LAUNCHXL Board [10], that is packed with an ARM Cortex-M3 CPU supporting up to 48MHz. The MCU features a low-power sensor controller, which support off-board data acquisition reducing the load on the main CPU. It is also BLE 4.2 [11] compatible supporting one of the design considerations in achieving lower energy consumption.

There are two digital sensing components attached to the MCU and an analog one. The digital sensors, a 3-axis accelerometer and an illumination sensor, are connected through I2C protocol at 400KHz, while the analog sensor, the microphone, is connected to

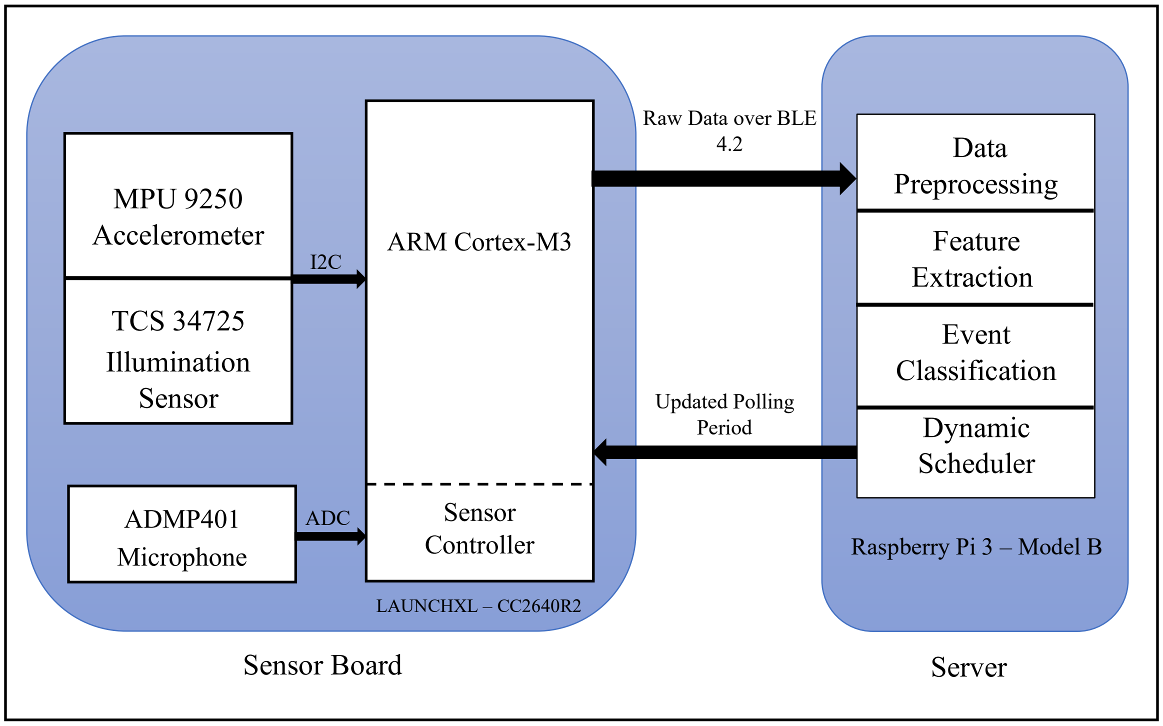


Figure 1: The sensing architecture consists of the sensor board for data acquisition and the server for featurization and classification

the sensor controller that support different sampling rates. Raw data are collected in the sensor board and are sent periodically to the server for further processing.

The server side is based on a Raspberry Pi 3 – model B featuring with a quad-core 64-bit ARM Cortex A53 1.2 GHz providing adequate computational power and portability for IoT application, in addition to, supporting BLE 4.2 communication protocol. It preprocesses the raw data received at high sampling rate from the accelerometer and the microphone, before feature extraction. The extracted features are fed into a K-means model for clustering, during training phase, and classification, during employment phase. During the training stage, the duration of each event is used to select the polling periods for each class.

B. Data Acquisition

In this framework, there are a total of three different sensors with different sampling rate. Both the accelerometer and the microphone are sampled at high frequencies, 4kHz and 5.8 kHz respectively, to capture a wide range of vibration and sound spectrums, while illumination sensor is sampled at 10Hz since the change in this element is considerably slow. In order to achieve high sampling rate, the accelerometer data are buffered in a 512 bytes buffer supported in the MPU9250 chip [12] before the MCU reading. As the buffer approaches overflowing, the MCU reads it and store the data in an array. A Mean-Moving-Window (MMW) is passed on the accelerometer signal to reduce the noise and the packet size, by using fewer number of data points, to meet the limitation of BLE bandwidth. The microphone is connected the Sensor Controller [10] through an Analog-to-Digital Converter (ADC), which is also support the sound signal buffering. MMW is applied to the microphone signal to reduce the overhead of data transferring as well.

Overall, the sensor board send eleven packets per second to the server over BLE, transferring a total of 512 data points the accelerometer, 704 data points for the microphone, and a total of 10 data points for the illumination sensor.

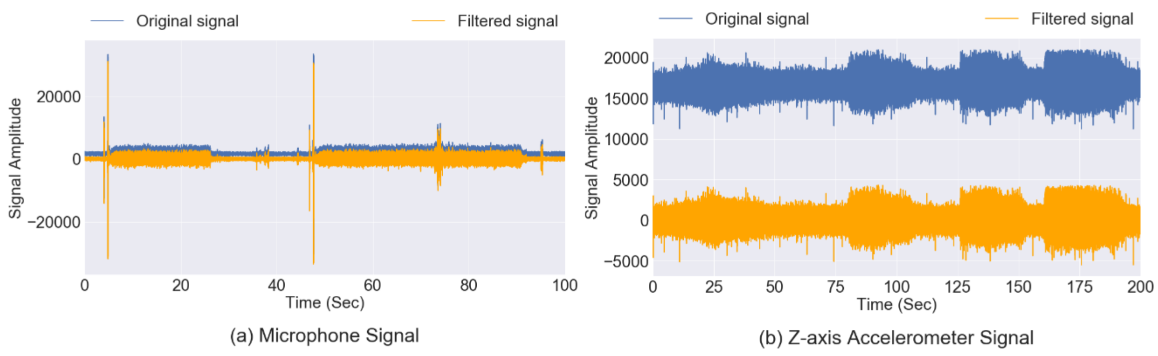


Figure 2: The microphone signal and Z-axis signal of the accelerometer before and after applying the high-pass filter

C. Signal Preprocessing

Since the data from Accelerometer and Microphone are collected at high sampling rate, it requires more processing than the ones from the illumination sensor. Therefore, a high-pass filter is applied first to remove the effects of the DC components. Signals from all sensors are sliced into different sized windows to prepare them for feature extraction. The following sub-sections will discuss these steps in details.

1. Filtering High-Frequency Signals:

The data collected from the accelerometer and the microphone contains noise and other factors, such as the DC components, that affect feature extraction. Therefore, filtering is a necessary step in order to reduce these effects before extracting features. In this case, an MMW filter is applied to both the Accelerometer and Microphone signals in the sensor board to reduce the noise before sending the data. After that, a high-pass filter is applied to cut low frequencies that will remove the DC gain effect.

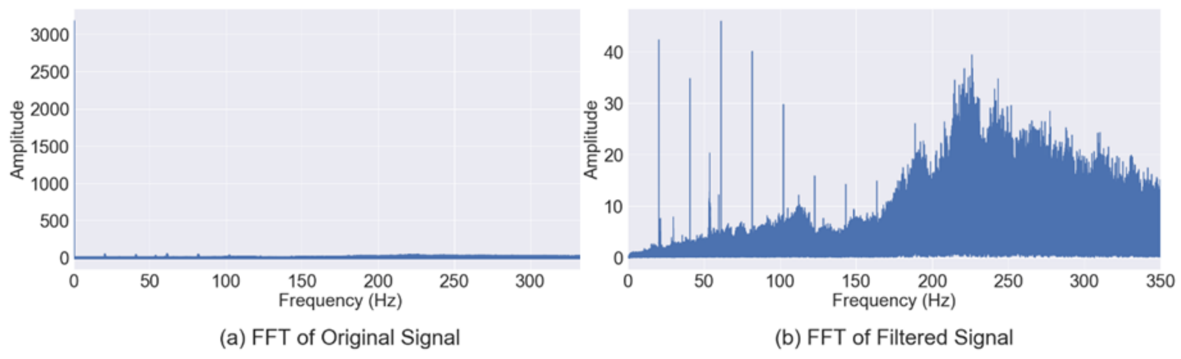


Figure 3: (a) shows the FFT of the microphone signal without any filter, while (b) is the same signal after applying a high-pass filter which improve peaks detection

It is clear from figure 2 that both signals have the same amplitudes and shifted to zero. This is because the DC components from both signals are removed. This is important in feature extraction because DC component has a very high amplitude and will take most of the energy after applying Fast Fourier Transform (FFT) [13] or Power Spectrum Density (PSD) [14] analysis on the signals. Figure 3 demonstrates the necessity of applying a high-band pass filter on the accelerometer and microphone signals.

2. Slicing Signals:

Signals collected for training from all sensors are continuous and need to be sliced into fixed windows. In high level, this is needed to make the signal periodic, especially in the case of applying FFT since it assumes that the periodic signals. Different window sizes are applied to each sensor based on the sampling frequency.

Signals coming from the accelerometer and microphone are sliced into 512-points and 704-points windows, respectively, with 25% overlap. Each window passes through a hanning function to avoid leakage that may be caused after applying FFT. Signals from

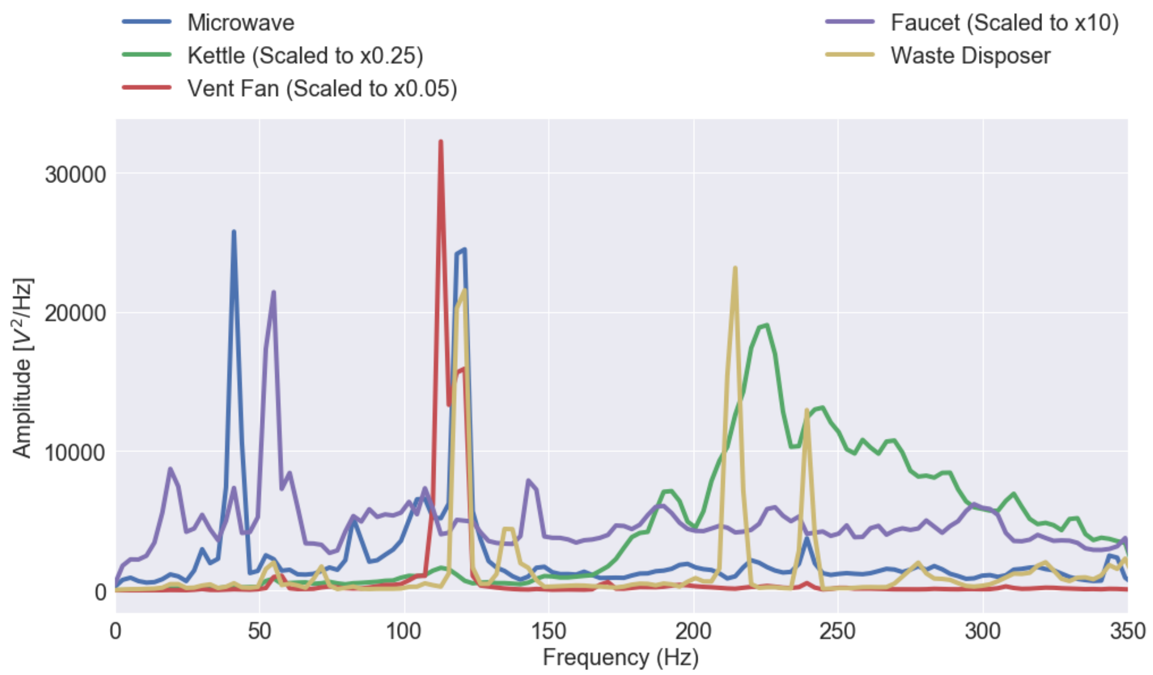


Figure 4: PSD of microphone signals for different events comparing their peaks

illumination sensors are slices into a 10-points windows with 25% overlap. After signals have been filtered and sliced into windows, there are ready for feature extraction.

D. Feature Extraction

Since sensors data have different sampling frequencies, two feature extraction approaches have been used. For signals with high sampling rate from the accelerometer and the microphone, FFT and PSD tools are used to extract features, while statistical information, the mean, variance and range, are used as features for the illumination sensor signal. Both methods will be discussed in the coming subsections.

1. Features Based on FFT and PSD

FFT and PSD are very important tools to analyze signals, especially these with high frequency. Both techniques transform signals into frequency domain for better analysis.

FFT is used to compute the Discrete Fourier Transform (DFT) but with lower computational cost. The signal is characterized by its magnitude and phase, instead of amplitude and time. As a result, it is more efficient to recognize events by their frequency and magnitude. PSD is similar to FFT but distribute the signal's magnitudes into frequency bins instead of distributing them over all frequencies. Using these two powerful signal processing tools, the system can find the peaks in the frequency response and use these peaks as features as shown in figure 4.

In order to extract features from accelerometer and microphone signals, FFT and PSD are applied on the windows extracted from previous preprocessing stage and the peaks with highest magnitudes will be used as features, where for each peak only the frequency is selected due to the fact that the magnitude value changes depending on position of the sensor board. The code to extract these peaks is based on the repository provided by [15]. In this framework, the highest five peaks are chosen as features from each window. Since each feature consist of 1-point, each window will produce 10 feature points, 5 peaks from each FFT and PSD. Therefore, the dimension of the feature matrix of accelerometer and the microphone, in the training phase, will be of the number of windows by 10 features. During testing and deployment, the server will compute a 10-element feature vector for each sensor.

2. Statistical Features

Three statistical features, the mean, variance and range are selected for the illumination sensor. Each of these is extracted on a single window which consist of ten points. Therefore, the size of the feature matrix extracted from the illumination sensor is the number windows by three during training, and a 3-elements feature vector during deployment stage.

All collected features are normalized in a range between (0-100) since K-means algorithm is based on the Euclidian distance. If features are not normalized, those with large values will cause cluster centers draw to them, which will decrease the classification performance and interduces labeling errors. After all features are normalized, the feature vector is created by concatenating all the features. The feature vector, which contains 23 features is analyzed by an unsupervised learning technique, namely K-means, to cluster the data for labeling and classification.

E. Clustering and Classification

Collecting raw data from different sensors for a variety of events and manually label each event is a time-consuming procedure. Therefore, the framework is based on unsupervised learning techniques which reduces the complexity of the sensing problem by labeling clustered data as events, then using the same trained model to verify the correctness of the classes.

The procedure of data cluttering, events labeling during training phase, verification of classes, and system deployment is performed by two different systems. The training part is done on a personal computer featuring an i7 Core CPU and the deployment stage is performed on the server, the Raspberry Pi 3 model-B. Both systems support Python and sci-kit learning package [16], and they use the same signal preprocessing, feature extraction and K-means model. The following sections will explore both stages in details.

1. Training Stage

It is more efficient to use a PC since the training stage is offline and requires deep analysis dealing with large data sets. After collecting the training data on the system, these data sets are transferred to the PC for analysis. As described previously, the raw data are preprocessed for feature extraction. After the feature matrix is created, the K-means model implemented in sci-kit learn package is used to cluster features into different classes. The training data set is collected in the presence of five kitchen events; microwave, faucet, kettle, waste disposal, cooking vent, for 15 minutes for each event. The state of each event is whether it is running (on) or unused (off). During unused (off) state, the scheduler considers it as a no event period. Notice that each event affects different environmental facets. For example, a kettle running produces vibration and ambient sound, while the cooking vent change the light intensity since its lamps is turned on while being used. So, during K-means clustering, the expected number of clusters is six.

After the clustering is performed, the model is verified on a different data set collected before deploying the model to the server for real-time testing. To verify the model, different data sets are collected for each event. Features are extracted from each date set after being preprocessed and these features are fed into the trained model for labeling. The model is expected to classify each data set with a distinct label. It is expected to see a no-event label within each set.

The system is re-trained after verification but with adding the verification dataset and additional analysis is performed for the dynamic scheduling which will be further discussed in a later section in the thesis.

2. Deployment Stage

The model is transferred to the server for testing and deployment after verifying that each event is classified correctly. During this stage, the server will run in real-time unlike the offline training phase. The sensor board send raw data as described in the data acquisition section periodically every second to server. The server waits for all the packets to be received and buffer different signals into separate windows as expressed earlier.

After all packets are received and the windows are ready, accelerometer and microphone signals are filtered using a high-band filter then passed through FFT and PSD to find the highest five peak frequencies, and statistical features are extracted from the signal of the light sensor. After that, these features are concatenated to create a feature vector that is classified by the K-mean trained model. The accuracy of the classifier is evaluated at this stage without the proposed dynamic scheduler. More details are provided regarding this in the evaluation section.

IV. Dynamic Scheduling

In this work, the main objective of the system design is reducing energy consumption. This design consideration is important because the system can be deployed on a battery as a power source, which allow placing the sensor at a location close to the environmental facets of interest. There are three main components in the sensor board that consume energy, the CPU, the BLE, and the attached sensors. The best approach would be putting all these components to sleep when there is no change in classification. However,

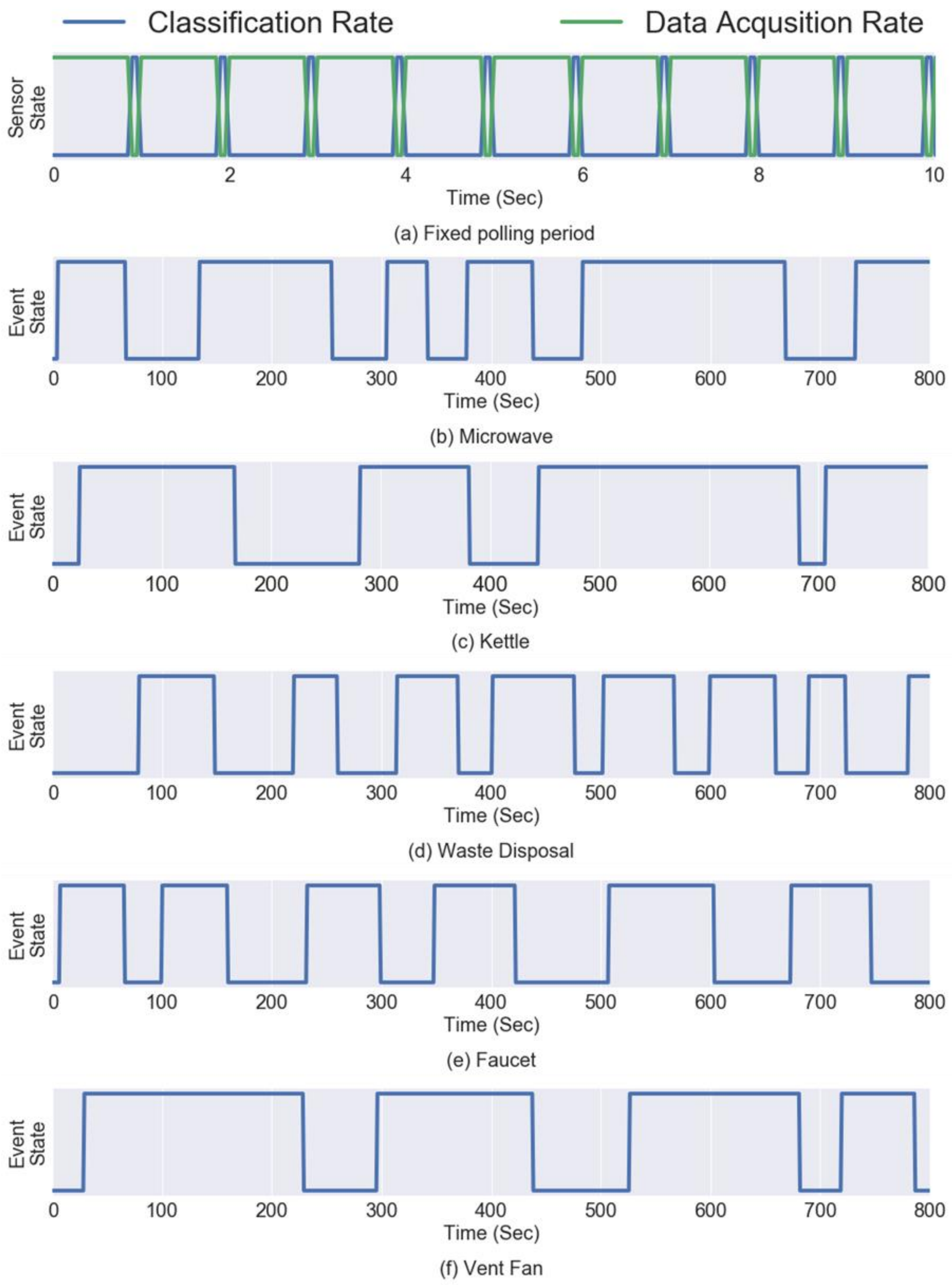


Figure 5: Sensor static scheduling and events operating frequencies

scheduling data transition through BLE is sufficient to illustrate this paper approach efficiency for energy reduction since the BLE consume a significant amount of current estimated at 120μA/Byte when transmitting, while approximated only 1μA in standby [17, 18].

A. Motivational Example

In sensors design, data acquisition scheduling is one of the most common research approaches as in [5-8]. Instead of inquiring sensor reading continuously at a high sampling rate, the sensor will only transmit data when needed. Determining the demand for new data depends on the type of application. In event detection application, using probabilities to predict the value of the sensor, like in [5, 6] is not viable because the data of different events are not temporally correlated, as events can occur at any time instance. Conversely, finding the appropriate polling period for the sensor based on the rate of change in event state is feasible.

As shown in figure 5 (b-f), each type of event runs for different time intervals. The sensor data will not change significantly during this time period since the event classified is still running. Hence, the sensor can be idle as it is not required to transmit data over this period. However, this period is not constant for each event, e.g., the microwave could run for 10, 30, 60, 120 seconds or more based on personal usage, while kettle could run for longer time based on the water level inside it. Therefore, it is necessary to determine a suitable idle sensor period, polling period, for each classified event.

B. Problem Statement and Solution

To meet the design requirement of reducing energy consumption, the framework must support a scheduler such that the sensor will be idle for longest possible period. The sensor will not transmit data during the polling period, which will lead to energy saving. Another important factor in choosing polling periods is worst-case classification latency. This latency occurs when the sensor is idle and the actual classified event changes during this period. Thus, the problem of this thesis can be expressed as.

Problem Statement: the problem is to find a polling period for each classified event such that the following condition is met.

Condition: the polling period of each event must not exceed the ***worst-case classification latency*** (WCCL) defined by the user.

$$N \times T_{pp} - T_e \leq WCCL$$

where T_{pp} is the polling period assigned by the scheduler for each event, T_e is the time when the event actually ends, and N is the number of times the polling period is repeated until the classifier recognize a new event. In order to solve this problem, a dynamic scheduler based on the rate of change in classified events is implemented in the framework. The scheduler will assign different polling period for each event, such that the sensor will be idle for that period.

C. Proposed Algorithms

To determine the polling periods, this work proposes two algorithms, the first one is simply choosing the minimum event time-interval as a polling period for each event, and

```
Algorithm 1: Minimum Event Time-Interval
    Input  : $T_e$: An event classifier-based periods
    Output: $T_{pp}$: Minimum Polling Period
1 $T_{pp} \leftarrow$ FindMinimum($T_e$);
2 return $T_{pp}$
```

the second algorithm is finding a base-period, such that the time between the actual end of an event and the classification changes is within the WCCL.

1. Minimum Event Time-Interval

As shown in Alg.1 pseudo-code, this algorithm simply chooses the minimum period of an event among all other periods found based on the classifier results. Although this approach may not satisfy the condition in the problem statement, its contribution to energy reduction is significant. Different events run for various time intervals, and each event has different rate of changes depending on personal usage.

As can be seen in figure 5 (b-f), for the events {microwave, kettle, waste disposer, faucet, vent fan, no event} the shortest running periods based on the training data are {30, 46, 24, 59, 67,24} seconds. The *no event* class is when the classifier does not detect any activity. The purpose of adding *no event* class period in the scheduler is to avoid excessive data transmission by fixing the polling period at one second. Accordingly, Alg.1 uses these shortest periods for each class as the sensor polling periods. Based on the event detected by the classifier, the server will request the sensor to halt transmission and enter idle mode for the period corresponds to the detected event.

The power saving due to this approach can be significant, as will be discussed in the evaluation section, since the sensor will be on idle mode for periods of 24 to 67 seconds

Algorithm 2: WCCL Constrained Polling Period Search

```
Input  : $T_e$: An event classifier-based periods
         $\varepsilon$: Worst Case Classifier Latency
Output: $T_{pp}$: WCCL Constrained Polling Period
1  $T_e \leftarrow$ AscendingSort($T_e$);
2  $T_{pp} \leftarrow$ FindMinimum($T_e$);
3  $L1 \leftarrow T_{pp}$;
4  $L2 \leftarrow$ Length($T_e$);
5  for $i \leftarrow 1$ to $L1$ do
6      for $j \leftarrow 0$ to $L2$ do
7          $N \leftarrow$ Ceil($\frac{T_e[j]}{T_{pp}}$);
8          $Threshold \leftarrow T_e[j] + \varepsilon$;
9          $nT_{pp} \leftarrow N \times T_{pp}$;
10         if $nT_{pp} > Threshold$ then
11             $T_{pp} \leftarrow T_{pp}$- 1;
12             Break;
13         end
14     end
15 end
16 if $T_{pp} > 1$ then
17     return $T_{pp}$ , the polling period for this event;
18     else Cound not find a suitable $T_{pp}$. Increase $\varepsilon$;
19 end
```

each cycle. In contrast, frameworks with static scheduling, as in [1], runs continuously at high sampling rate leading to excessive usage of energy. Of course, this algorithm will compromise WCCL, whereas it only considers the smallest period that is not necessarily an integer multiple for other periods of the same event. To clarify this with an example, if the polling period chosen is based on the pervious Alg.1, and the system detected that the microwave is running. The sensor will be idle for 30 seconds. Assuming that the microwave was running for 50 seconds, the sensor will be idle for additional 10 seconds before the system recognize that the current event ended. Consequently, a different algorithm is introduced that solves the stated problem while meeting the WCCL condition.

2. WCCL Constrained Polling Period

As stated in the previous scenario, the additional idle period that causes classification latency can be guaranteed if the multiple of the polling period (T_{pp}) does not exceed the event classifier-based periods by ε (WCCL), which is defined by the user. A naïve approach would be using Greatest Common Factor (GCD) of all occurrence intervals of an event, which would result in one second as polling period for most cases. Thus, a new algorithm is proposed that approximate the largest factor of all periods and guarantees the WCCL condition. Looking back at the microwave example, if the polling period was 8 seconds instead of 30 seconds with WCCL of 4 seconds, the sensor will be idle for 4 seconds after 9 cycles compares to 10 seconds.

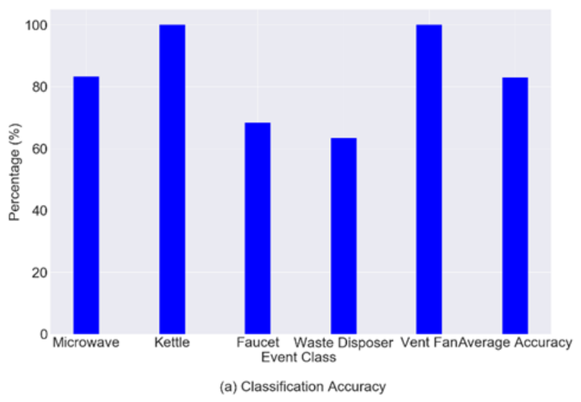
The algorithm proposed is an exhaustive search-based algorithm that will select the polling period found by the alg.1 as a base period. This period is compared to all other event occurrences periods such that the difference after N cycles does not exceed the WCCL condition. If T_{pp} failed the test, it will be decremented by one, and the search will be repeated until the polling period is found. In case T_{pp} reaches 2 seconds, it will be selected since the sensor will be idle for half the total operation time in worst-case, assuming the user will select ε greater than one. The user has the freedom to choose a consistent ε for all events or specify a unique one for different events based on the importance and tolerated latency.

V. Evaluation

To show that the framework meets the design consideration, classification accuracy and energy efficiency, four experiments are performed. The purpose of the first experiment is to evaluate the accuracy of the classifier based on K-means, an unsupervised learning technique, without the dynamic scheduler. The second part of the evolution is finding the effects of ε on the polling periods based on Alg.2 and selecting T_{pp} randomly. These selected polling periods, in addition to the ones computed by Alg.1, are used in the next experiment to explore the latency introduced by the dynamic scheduler and compared it to the static scheduler, similar to previous sensing work [1-4]. Lastly, the power consumption is studied, which shows significant energy reduction compared to [8]

A. Classification Accuracy

To evaluate the performance of the classifier, the trained model is deployed to the server, the Raspberry Pi 3 – model B, which is connected to the sensor tag, based on the CC2640R2 LAUNCHXL Board, via BLE transition medium. In this experiment, the dynamic scheduler is not integrated into the system, since the main objective is to evaluate the accuracy of the classifier. The system is deployed to the same location, a kitchen in a one-bedroom apartment, where training data has been collected with similar conditions. Events occurred at different time instances, and the classes is decided based on the detected event every second.



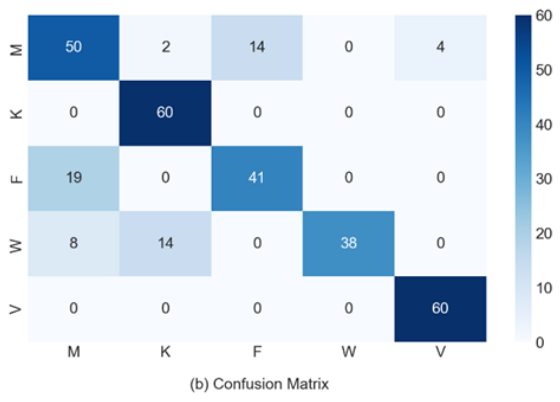


Figure 6: Classification accuracy and confusion matrix based on the result of 5 different classes deployed at real-time experiments

For each class, the test is repeated ten times, one minute for each event, and the classification is recorded every second. If the classification matches the actual event, it will be considered as correctly classified. Otherwise, it is a misclassification. The total number of correct classifications is divided by the 60 for each test per minute. The results are averaged as shown in figure 6, whereas kettle and vent fan events are detected with a 100% accuracy. The microwave event accuracy comes next at 83.3%, while faucet and waste disposer events are classified with relatively low accuracy at 68.3% and 63%, respectively.

The confusion matrix shows 14 false negatives and 19 false positives between the microwave and the faucet. This miss classification is because these two classes shares common frequencies as shown in the FFT and PSD analysis previously discussed. The waste disposer is confused with the kettle with 14 false positives as both emits similar vibration that is captured by the accelerometer. The overall Miss Rate = 0.20 which is computed by dividing the total number of false detections by the number of classifications.

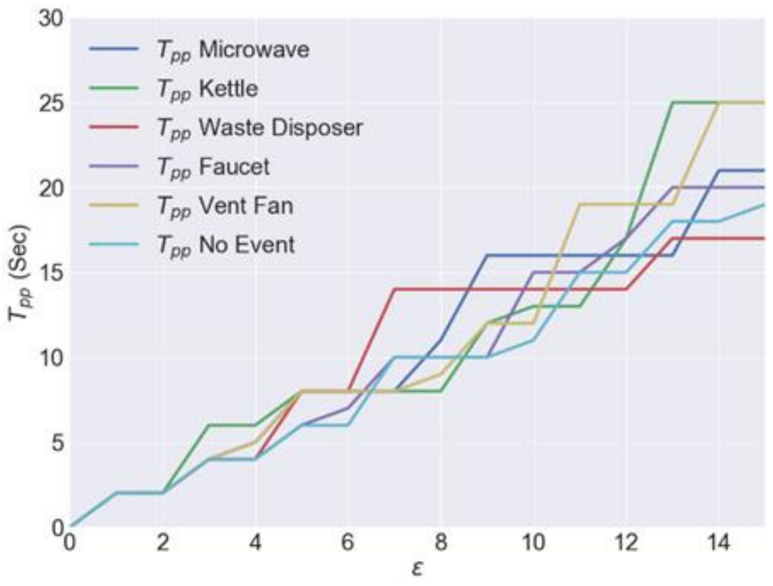


Figure 7: Experimenting Alg.2with different ε values and explore its effects on T_{pp}

B. Dynamic Scheduler Polling Periods

The objective of this experiment is to test a range of ε, from 1 to 15 as the worst-case classification latencies, and explore its effects on T_{pp}. Obviously, T_{pp} increases as the user chooses larger values ε but the change is not linear for all events, e.g., the polling periods of the kettle and vent fan reaches 25 seconds when ε is 13 and 14 seconds, respectively.

As for the next two experimentations, the polling periods of each class is selected randomly with different ε values to study the classification latency and energy consumption. The highest polling periods of 25 seconds is chosen for kettle and vent fan classes with WCCL of 13 and 14 seconds, respectively. For microwave, waste disposer, faucet and no events, the polling periods selected are 16, 14, 20 and 15 seconds with ε of 9, 7, 13 and 11 seconds, respectively. These periods are relatively lower than the ones

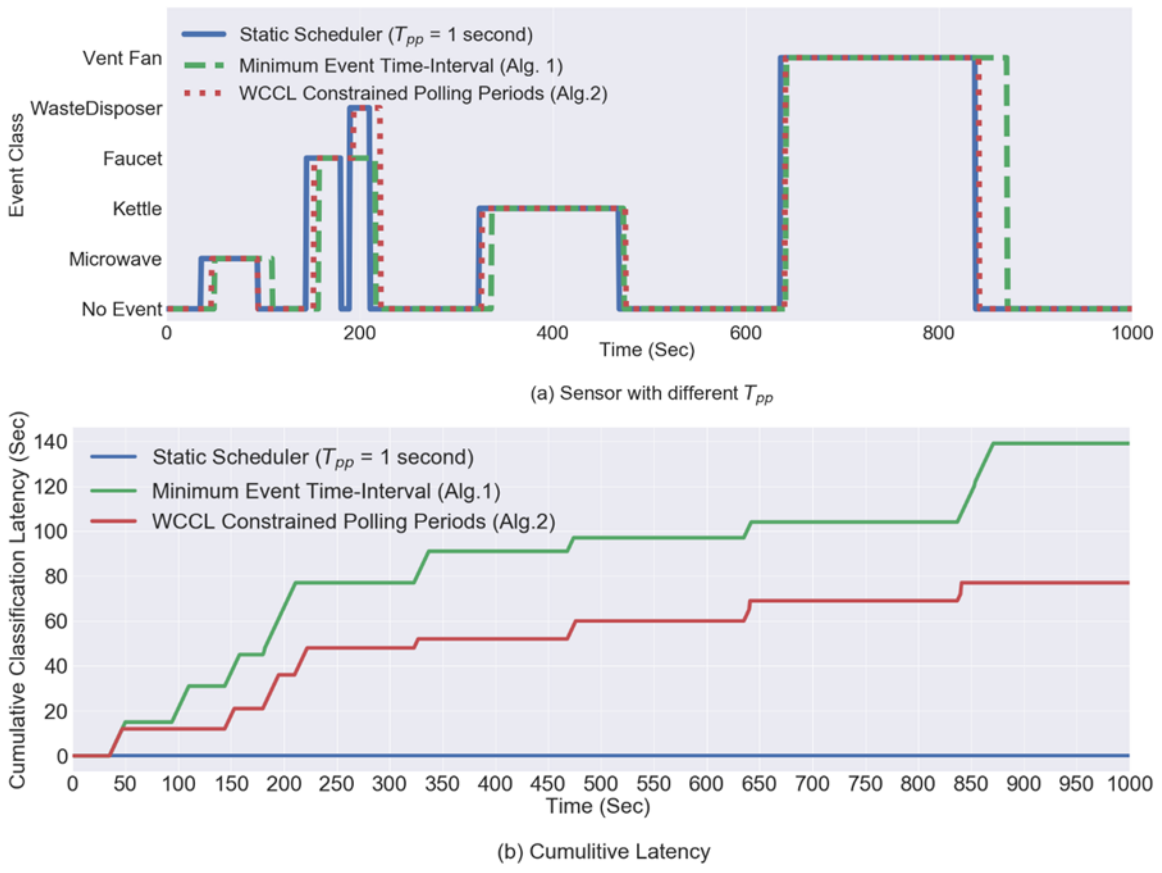


Figure 8: Latency evaluation based on the dynamic scheduler against the state-of-art approach

selected by Alg.1, as shown in the Algorithms subsection. All selected T_{pp} are feed into the implemented dynamic scheduler to simulate real-time deployment.

C. Dynamic Scheduler Latency Evaluation

The dynamic scheduler is deployed with the selected polling periods for each class from the previous experiment based on Alg.2 and Alg.1. The test is simulated three times, where the first one the framework runs with static scheduler, at a fixed polling period of 1 second, to recognize events at real-time. The duration of each event, randomly performed, is captured using a timer in order to repeat the exact pattern of events for the experiments with the scheduler.

The scheduler based on the WCCL constrained polling periods algorithm latency performance significantly exceeds the minimum event time-interval algorithm. Alg.2 scheduler closely matches the static scheduler pattern with a slight delay at detecting the first event as clearly seen in figure 8 (a). However, since the polling periods are marginally large for both algorithms, there is a two event state changes missed by the minimum event time-interval algorithm after faucet detection, while one change is missed by the WCCL constrained polling periods based scheduler as the actual time between no event and waste disposer event was only 9 seconds.

It is evident that the state-of-art, with static scheduler, has a superior performance when it comes to latency since event classification occurs continuously meeting real-time applications requirement with almost zero latency. Still, many applications are latency tolerant, such as sensing home appliances states which does not result in unwanted consequences when the latency is constrained. According to figure 7 (b), the dynamic scheduler with T_{pp} based on Alg.2 has an overall cumulative latency of 80 seconds over a deployment of one thousand seconds (8%), while Alg.1 based T_{pp} has 140 seconds overall latency over the same span, making it 175% more than WCCL constrained polling periods approach.

The maximum classification latency due the WCCL constrained polling periods-based scheduler is 13 seconds, the minimum is 5 seconds, and the average latency is 9.25 seconds. On the other hand, the minimum event time-interval based approach has a significantly high delay reaching 35 seconds, while the least latency captured was 7

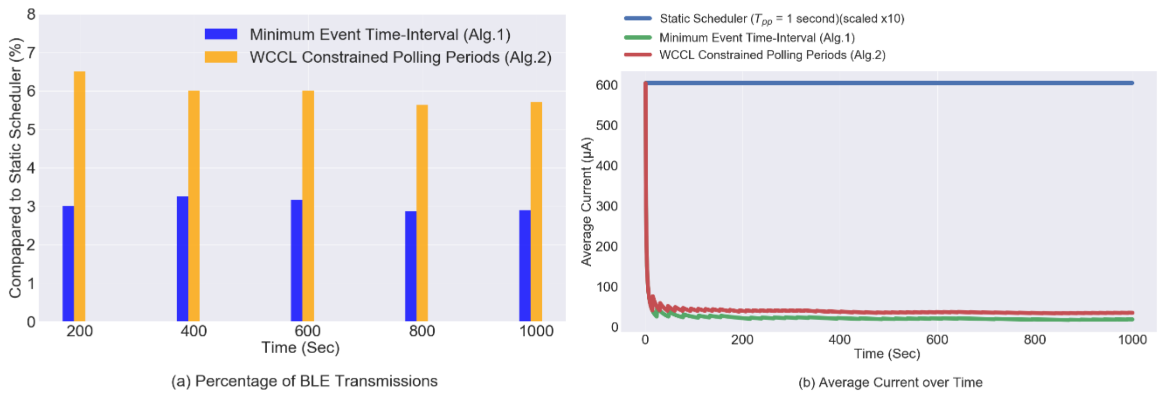


Figure 9: Power consumption evaluation based on the number of data polling request and the average current of the BLE packet transmission

seconds. The average latency in this case is 17.87 seconds, which is almost double the average latency compared when the scheduler is based on Alg.2

D. Dynamic Scheduler Energy Reduction

The most advantage of integrating a dynamic scheduler in a sensing framework is to reduce the number of data polling requests, leading to more idle (inactive) time on the main sensor which signify the average current reduction. This experiment is based on this the simulation results from the previous section, which indicates remarkable efficiency. Figure 9 (a) compares the number of BLE transmissions based on a dynamic scheduler compared to the static scheduling approach. The transmitted BLE packets using WCCL constrained polling periods-based scheduler is approximately 6% of the total number of transmissions made by the static approach, and 3% for minimum event time-interval based scheduler. With respect to the latency performance achieved by Alg.2 as discussed previously, 3% polling requests more compared to Alg.1 is relatively neglectable.

To approximate the power consumption, according to [17], each transmitted byte leads to a 120µA consumption. However, another important factor in power consumption is the BLE connection interval and based on the energy calculator tool provided by Texas Instruments [18], the active current of the sensor is approximated to 603.9 µA. Therefore, the average current can be computes as follows.

$$I_{average} = I_{active}\ x\ (Time_{active}/Time_{total})$$

Figure 9 (b) demonstrates a significantly low average current for dynamic scheduler approaches, and a high linear one for the static scheduler mechanism. The state-of-art shows a significant active current 600 µA that is 33× and 17× more than dynamic scheduling with minimum event time-interval and WCCL constrained polling periods, respectively. Although the Alg.1 achieves notable energy reduction, its latency performance limits its benefits. Subsequently, a dynamic scheduler based on Alg.2 is a potential approach for realizing energy efficient sensing framework.

VI. Discussion

The framework passed through several development stages to complete, from designing and implementation to experimentation and deployment. During these stages several findings and challenges were recorded. This section will focus on these that are mainly related to the thesis work, namely: classification accuracy, environmental facets, and dynamic scheduling algorithms, in addition to, possible future work.

Improving the classification accuracy requires deep understand of the design limits. The factors that lead to the classifier error rate can be grouped into three categories. Firstly, FFT and PSD peaks contribute to this loss because some events share common frequency peaks as shown in figure 4. Although these peaks have different amplitude, the features are selected based on the frequency value of these peaks. Thus, to avoid classification confusion, the amplitude values could be used as features, but this will limit the deployment location of the sensor as it will become sensitive to amplitude changes.

Additionally, both signals proceed using similar tools to avoid additional computational cost on the server. So, different signal processing tools can be used for accelerometer and microphone signals, e.g., using MFCC for microphone signals similar to [3]. Furthermore, the accelerometer was expected to be more sensitive when running at 4kHz according to [1, 19] but only the kettle and waste disposer events vibration were captured. One explanation to this is due to the weight of the sensor, where the breakout board of each sensor are used on the tag, leading to heavier weight and lesser vibration sensitivity. The weight can be reduced in future work by designing a custom circuit board using the required sensors only.

Second potential improvement can made on data transmission medium. Since BLE has limitations on its actual bandwidth, signals have been averaged before transmission to reduce the overhead resulting in a decreased data quality. However, BLE was used in this framework since it provides the system with lower energy consumption compared to other mediums. Hence, by using another communication alternative in future design, such as Wi-Fi which has greater power consumption, the quality of the raw data can be improved allowing more accurate analysis and features extraction.

There are many environmental facets that can be used for activity recognition. In early framework developing stages, additional sensors were used, like temperature, pressure, humidity, and gas. Due to the fact that the rate of change in these elements are slow compared to the accelerometer and microphone, they were removed from the sensor tag. However, these sensors, and others, are useful when trying to detect appliances such as oven and stove, or in applications similar to [8]. So, the framework has the capability to be deployed to different environment, which can be tested in the future.

As for the dynamic scheduling, based on both algorithms, it showed exceptional reduction in the total number of data transited when implemented compared to the static scheduling approach. However, only Alg.2 performed with constraint latency most of the time. The miss classification due to the latency, as shown in figure 8, was because the period of "no event" was less than the polling period assigned by the scheduler. This can be relatively avoided by using different polling periods selection mechanism, instead of using the proposed method in the evaluation section. For example, selecting WCCL polling periods smaller than the one found for "no event". This way, the possibility of missing a change in class state is reduced.

The sensor scheduling can be improved in the future by extending it to change the sampling rate of each sensors. E.g., sounds that can be detected by lower frequency, the microphone can run at lower sampling rate leading to further energy reduction. It is also possible that multiple sensors are deployed in an open space environment to enable event detection in a wider range. However, a potential challenge can be imposed by this. Extending the dynamic scheduler to multiple sensor requires deep synchronization analysis since these sensors can detect the same event, or each one detects different events.

VII. Conclusion

The framework proposed in this thesis, to the author knowledge, is the first that combine unsupervised learning technique for labeling and real-time classification with a dynamic scheduler to achieve an energy efficient sensor. The sensor collects and sends raw data of three environmental facets to the server for preprocessing, feature extraction and classification using K-mean algorithm. The polling periods of the dynamic scheduler is selecting based on two proposed algorithms. The classifier is deployed in a kitchen of a one-bedroom apartment for performance evaluation, where the simulation shows promising results for the WCCL constrained polling periods based dynamic scheduler. Overall, the system shows promising classification accuracy for an unsupervised classifier, and a potential dynamic scheduling mechanism for energy efficient IoT sensing application.

References

1. Gierad Laput, Yang Zhang, and Chris Harrison, “Synthetic Sensors: Towards General-Purpose Sensing,” in Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17), ACM, New York, NY, USA, 3986-3999. DOI: https://doi.org/10.1145/3025453.3025773

2. Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L. Littman, “Activity recognition from accelerometer data,” in Proceedings of the 17th conference on Innovative applications of artificial intelligence - Volume 3 (IAAI’05), AAAI Press, 1541–1546. DOI: https://doi.org/10.5555/1620092.1620107

3. Jianfeng Chen, Alvin Harvey Kam, Jianmin Zhang, Ning Liu, and Louis Shue, “Bathroom activity monitoring based on sound,” in Proceedings of the Third international conference on Pervasive Computing (PERVASIVE’05), Springer-Verlag, Berlin, Heidelberg, 47–61. DOI: https://doi.org/10.1007/11428572_4

4. Sidhant Gupta, Matthew S. Reynolds, and Shwetak N. Patel, “ElectriSense: single-point sensing using EMI for electrical event detection and classification in the home,” in Proceedings of the 12th ACM international conference on Ubiquitous computing (UbiComp ’10), Association for Computing Machinery, New York, NY, USA, 139–148. DOI: https://doi.org/10.1145/1864349.1864375

5. B. Gedik, L. Liu and P. S. Yu, "ASAP: An Adaptive Sampling Approach to Data Collection in Sensor Networks," in IEEE Transactions on Parallel and Distributed Systems, vol. 18, no. 12, pp. 1766-1783, Dec. 2007.
DOI: https://doi.org/10.1109/TPDS.2007.1110

6. H. Jiang, S. Jin and C. Wang, "Prediction or Not? An Energy-Efficient Framework for Clustering-Based Data Collection in Wireless Sensor Networks," in IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 6, pp. 1064-1071, June 2011. DOI: https://doi.org/10.1109/TPDS.2010.174

7. Qi Han, Sharad Mehrotra, and Nalini Venkatasubramanian, “Energy Efficient Data Collection in Distributed Sensor Environments,” in Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS&apos;04) (ICDCS ’04), IEEE Computer Society, USA, 590–597. DOI: https://doi.org/10.5555/977400.978037

8. S. Heo, S. Song, J. Kim and H. Kim, "RT-IFTTT: Real-Time IoT Framework with Trigger Condition-Aware Flexible Polling Intervals," 2017 IEEE Real-Time Systems Symposium (RTSS), Paris, 2017, pp. 266-276.
DOI: https://doi.org/10.1109/RTSS.2017.00032

9. S. Lloyd, "Least squares quantization in PCM," in IEEE Transactions on Information Theory, vol. 28, no. 2, pp. 129-137, March 1982.
DOI: https://doi.org/10.1109/TIT.1982.1056489

10. Texas Instruments, "CC2640R2F SimpleLink™ Bluetooth® low energy Wireless MCU Datasheet Rev. A," SWRS204A –December 2016–Revised January 2017.
http://www.ti.com/lit/gpn/cc2640r2f

11. Bluetooth Technology Website, "Bluetooth Low Energy," March 10, 2017.
https://web.archive.org/web/20170310111443/https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/low-energy.

12. InvenSense, Inc., "MPU-9250 Product Specification Revision 1.1," June 20, 2016.
http://www.invensense.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf

13. E. O. Brigham and R. E. Morrow, "The fast Fourier transform," in IEEE Spectrum, vol. 4, no. 12, pp. 63-70, Dec. 1967.
DOI: https://doi.org/10.1109/MSPEC.1967.5217220

14. Stoica, Petre, and Randolph L. Moses, "Spectral analysis of signals," Pearson/Prentice Hall, 2005.

15. Ahmet Taspinar, "Machine Learning with Signal Processing Techniques," April 4, 2018, GitHub repository, https://github.com/taspinar/siml

16. Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, Vol. 12, November 2011, 2825-2830.

17. Texas Instruments, "Measuring CC13xx and CC26xx current consumption Rev. D," Application Report, SWRA478D–February 2015–Revised January 2019.
http://www.ti.com/lit/pdf/swra478

18. Texas Instruments, "Bluetooth Power Calculator Tool," Version 3.0.0, July 2019.
http://www.ti.com/tool/BT-POWER-CALC

19. Gierad Laput, Robert Xiao, and Chris Harrison, “ViBand: High-Fidelity Bio Acoustic Sensing Using Commodity Smartwatch Accelerometers,” In Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16), ACM, New York, NY, USA, 321-333.
DOI: https://doi.org/10.1145/2984511.2984582