Online Change Detection for Energy-Efficient Mobile Crowdsensing

Viet-Duc Le, Hans Scholten, and P.J.M Havinga

Pervasive Systems, University of Twente
7522 NB Enschede, The Netherlands
{v.d.le,hans.scholten,p.j.m.havinga}@utwente.nl

Abstract. Mobile crowdsensing is power hungry since it requires continuously and simultaneously sensing, processing and uploading fused data from various sensor types including motion sensors and environment sensors. Realizing that being able to pinpoint change points of contexts enables energy-efficient mobile crowdsensing, we modify histogram-based techniques to efficiently detect changes, which has less computational complexity and performs better than the conventional techniques. To evaluate our proposed technique, we conducted experiments on real audio databases comprising 200 sound tracks. We also compare our change detection with multivariate normal distribution and one-class support vector machine. The results show that our proposed technique is more practical for mobile crowdsensing. For example, we show that it is possible to save 80% resource compared to standard continuous sensing while remaining detection sensitivity above 95%. This work enables energy-efficient mobile crowdsensing applications by adapting to contexts.

Keywords: Mobile Crowdsensing, Change Detection, Energy Efficiency, Resource Constraints, Computational Complexity, Adaptive Sensing.

1 Introduction

Recently mobile devices have a great improvement in both technology and popularity. For examples, Samsung Galaxy S5 is equipped with Quad-core 2.5 GHz Krait 400 and various sensors (e.g., accelerometer, gyro, proximity, compass, barometer, temperature, humidity, gesture and heart rate). That sensors are everywhere and are being designed into mobile devices offers researchers a rich and powerful computing platform to develop applications that leverage the sensing capability of these mobile devices. These applications are generally categorized into individual sensing and community sensing. Individual sensing pertains to the context of a particular device user, such as, running, walking, sleeping and health conditions. Community sensing attaches to environmental context which can be enhanced by combining sensory data gathered from individual devices. These contexts include air pollution, crowd density, road condition, and social events. In this word, we consider change detection for community sensing.

Community sensing has been known as mobile crowdsending [7], which is composed of participatory sensing and opportunistic sensing. Participatory sensing requires the contribution of device users to sense contexts (e.g. report a road

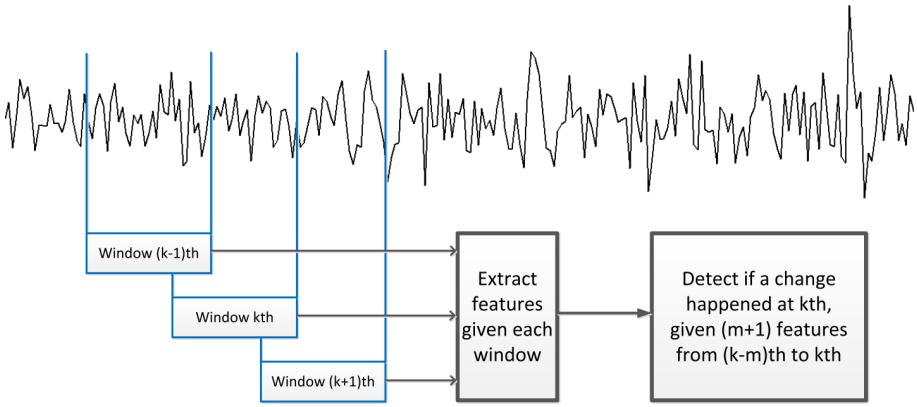


Fig. 1. Online change detection framework.

bump, taking a picture of fire). Conversely, opportunistic sensing autonomously senses contexts with minimum user involvement (e.g. continuously monitoring ambient temperature, device's locations). Technically, participatory sensing consumes less battery since it does not require continuously sensing like opportunistic sensing does. However, it is difficult to persuade numerous users to participate in sensing even with some incentives. In addition, participatory sensing may lead to biased results because the participants may provide incorrect information. Therefore, we focus on opportunistic sensing, which is more feasible to be deployed in large-scale networks.

Crowdsensing applications have exploited the microphone, GPS, accelerometer, gyroscope and pressure sensors. With the significantly developing of mobile devices, it is likely that more sophisticated and power-hungry sensors will be integrated in mobile devices in the future, such as, dust particle and gas sensors. Moreover, continuously uploading data to cloud servers is energy and bandwidth consumption. It also causes a considerable delay for real-time applications. Meanwhile, the capacity of battery and bandwidth are extremely limited. To avoid constantly invoking sensing, processing and uploading sensory data, the applications should be able to detect whether a context has changed. The power-hungry sensing, data processing and uploading just need to be executed around change points. Since contexts do not change very often throughout the day, change detection is a fundamental technique in energy-efficient mobile crowdsensing.

In this work, given sensory data, we introduce a new online change detection technique, called Frequency Likelihood Estimation (FLE). FLE provides low false positive while maintaining high true positive compared to current techniques. In other words, FLE enables researcher to develop energy-efficient applications in mobile crowdsensing including data sampling, data processing and data gathering with minimum resource consumption: computation, storage and battery. Figure 1 illustrates an overview of our scheme. In a nut shell, sensory data are split into predefined overlapping time windows. Data features are extracted from the measurements within a window. FLE estimates the change

probability by modifying histogram-based outlier detection. Unlike the conventional techniques that build the histogram based on only historical features, FLE includes the current test feature. This solves the bin width problem, which the conventional techniques cannot. That conventional techniques use the KL divergence to detect changes also faces unstable determination of the threshold. FLE overcomes this issue by simply taking the absolute value of the samples. This method makes it easy to find a robust threshold to detect changes based on the frequency of the most left bin or the most right bin. The threshold can be defined by tuning the F-score.

To investigate the proposed technique, we first extracted data features given recorded audio data with various sounds, such as, footsteps, gun shots, sirens, etc. Then, we compare performance of our proposed technique FLE with one-class support vector machine and multivariate normal distribution, which are well-known in anomaly detection. The results show that the detecting change points using FLE is the most appropriate technique for online mobile crowdsensing applications in terms of energy efficiency.

The paper is organized as follows. Section 2 summarizes related work and discusses in more detail the novelty of this paper. Section 3 introduces the proposed detection through four subsections: descriptions, mathematical formulation, frequency likelihood estimation, complexity and evaluation metrics. Section 4 presents experiment setup and results. Finally, we conclude this work with Section 5.

2 Related Work

Online change detection has been known for various applications in sensing applications, such as, environment monitoring, remote sensing, intruder detection, fault detection, medical diagnostics, etc. It also has been used to automatically segment signals for postprocessing in data mining. However, to the best of our knowledge, change detection has not been widely used to save resource in mobile crowdsensing applications [2,14,15], especially to sampling adaptively, processing and uploading sensory data. Only several very recent work tackle the energy-efficiency problem by predicting the error based on data distribution. However, they focus on sampling framework rather then change detection techniques. For example, EmotionSense [14] and the Jigsaw [10] take a considerable delay from several seconds up to minutes because they require ruling feature extractions and classifiers to detect changes. Another example, EEMSS [16] recognizes user states as well as detects state transitions using hierarchial sensor management strategy. Since the sampling schedule is fixed after training phase, EEMSS is only work well with routine contexts that were trained. Moreover, none of the previous work clearly define the change detection concept and its roles in mobile crowdsensing.

In this paper, we formally introduce the benefit of using change detection techniques in resource-constrained mobile crowdsensing. To save the resources, devices should switch on power-hungry sensor, sensory data processing and uploading if and only if a change is detected. Since the main aim is to save resources, we propose a lightweight and efficient change detection technique, a variant of

histogram-based anomaly detection using nonparametric statistical method to construct a profile of historical samples.

Histogram has been used in various applications [5,6,8]. In general, the techniques comprise two steps. The first step builds a profile given normal data. The second step checks whether the test sample is anomalous by measuring the frequency of the bin in which it falls. A key challenge for the conventional techniques is to determine an optimal size of the bins, which can be very dynamic under mobile crowdsensing circumstance. The test sample usually falls out of the learning profile even it is normal. Even if the anomalous sample falls into the constructed histogram, the frequency of the bin in which it falls might be as high as other bins.

There are also numerous algorithms for change detection, which are well reported in two surveys of Chandola et al. [3,4]. However, classification-based techniques are most suitable for online change detection in dynamic contexts because of the lack of normal samples. As the page limit, only one-class support vector machine (OC-SVM) for outlier detection using Hamming distances [11] and multivariate gaussian distribution (MVN) in the Machine Learning course [12] are discussed in this work. OC-SVM is well known for its sensitivity and MVN is well known for its computational complexity. In particular, OC-SVM can deal with a small number of samples but requires high computation. Conversely, MVN is less effective but lightweight. In general, detecting change points from an online streaming data given short historical samples (say from 10 to 50 samples) is still an intricate research topic for mobile sensing platforms.

3 Change Detection in Opportunistic Sensing

In this section, we describe the concept and problem of change detection in the opportunistic sensing context. Given the problem definitions, we present FLE approach and evaluation metrics, based on which we study our approach in Section 4.

3.1 Description

As changes might be misclassified with conventional anomalies or outliers in data, again, we emphasize that their definitions are different in the context of mobile crowdsensing. An anomaly in data is defined as an unexpected pattern in a dataset, which typically indicates some kind of problem, such as, a road bump, a gun shoot or a defect of an engine. Changes not only include such anomalies but also are referred to as transitions of contexts, such as the sequential sounds in Figure 2. Since data next to transitions like in Figure 2 do not have significant dissimilarity in pattern with previous ones, a good anomaly detection technique might be ineffective under such circumstance. Assume that such changes can be detected and do not happen very often throughout the day, we describe how they would be used to reduce energy and bandwidth consumption in mobile crowdsensing through sensing, processing and uploading sensory data.

Most mobile crowdsensing applications require continuous sensing, which is the burden of battery. One of the solutions is that these applications just activate

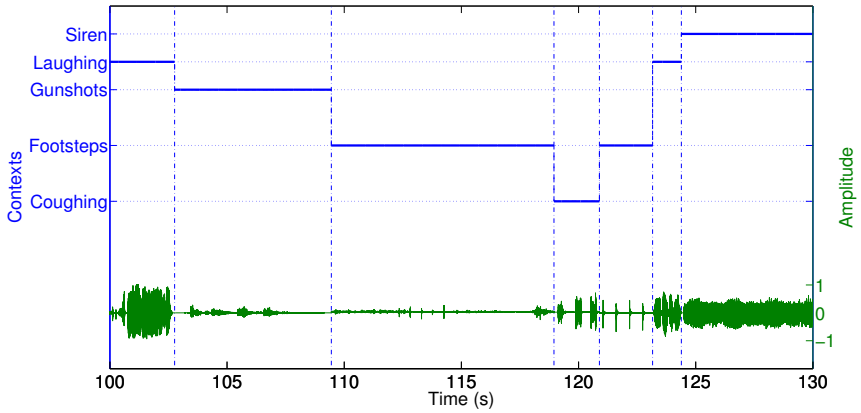


Fig. 2. A random sequence of sounds.

a minimum set of sensors to monitor the contexts. The rest can be idle to save the resource. A detected change can be used to invoke idle sensors to enhance information of collected data for better context inference. For examples, sound sensors can be use to detect if there is a sounds. If so, a camera is turned on to record the scene. By doing this, most power-hungry sensors stay in the sleep mode most of the time to prolong battery life.

Current approaches mine sensory data at individual devices to trigger other sensors in sensing or to avoid the overhead of uploading data to cloud servers. However, constantly processing data and data mining is also power hungry for mobile devices. The solution is detecting changes by some lightweight technique such as FLE in this paper. The reason is that change detection in our approach does not aim know what the context is by using data mining like in other approaches. Indeed, knowing that the current context has changed is fairly enough to trigger other sensors, data mining and data uploading processes.

Change detection also plays a key role in raw data gathering among mobile devices for high level context awareness. Most applications just need to know the transitions of contexts. For instance, knowing when a user changes from walking to running and then to sitting is barely enough to calculate how many calories have been burned. Therefore, storing, carrying and forwarding only sensory data sampled around change points significantly save bandwidth of networks.

In resume, change detection can be used to avoid constantly waking up sensors, data mining blocks and communication modules in mobile devices, which consume considerable power and bandwidth. This advantage facilitates mobile crowdsensing applications to operate in the long term.

3.2 Mathematical Formulation

We consider a mobile sensing system with multiple mobile devices that are equipped with sensors and daily carried by users. The idea beyond our proposal is that the devices should be able to trigger sampling, processing and uploading sensory data around the change points in order to save sensing power and bandwidth. In other words, the proposed technique should be featherweight

and able to quickly detect the change points on-fly with minimum false positive. To present the problem of online change detection more clearly, we first describe following definitions.

Definition 1. Let $\mathbb{T}$ denote an infinite set of discrete instants $t \in \mathbb{T}$. Therefore, the timestamp value, which is assigned to each consecutive sample, can be regarded as natural numbers from $\mathbb{N}$. We define time intervals $(t_1, t_2] = \{t \in \mathbb{T},\ t_1 < t \leq t_2\}$, $[t_1, t_2) = \{t \in \mathbb{T},\ t_1 \leq t < t_2\}$ and $[t_1, t_2] = \{t \in \mathbb{T},\ t_1 \leq t \leq t_2\}$.

Definition 2. At an arbitrary t, a data stream X is considered as an ordered sequence of the sample x_t, obtained with timestamp t. Without loss of generality, we define window W_t is a set of temporal samples spanning ω units backwards from time t, $W_t = \{x_i,\ i \in (t - \omega, t]\}$, where ω is called as the window size.

Definition 3. To control the transition of successive windows W, we define sliding step of ϑ time units. Upon sliding, window W_t^k at sliding step k^{th} subject to $t = k\vartheta + \omega$, $W_t^k = \{x_i,\ i \in (k\vartheta, k\vartheta + \omega]\}$ includes not only fresh ϑ samples but also a $\omega - \vartheta$ of samples from the previous window. For notational simplicity, we let $W^k = W_t^k$ be the window at sliding step k^{th}.

Definition 4. For better representing information of data, we define data features or window features are features extracted by mechanism $\mathcal{H}(\cdot)$. At any sliding step k^{th}, a feature is computed by $F^k = \mathcal{H}(W^k)$ or $F^k = \mathcal{H}(x_i),\ i \in (k\vartheta, k\vartheta + \omega]$. Note that F^k is a feature vector with p numerical attributes that represent data W^k. Therefore, the feature F^k can be also expressed as a row vector of p elements, $F^k = (a_1^k, a_2^k, \ldots, a_p^k)$.

Definition 5. Since the accumulative number of features F in feature domain $\mathbb{R}^p$ can be very large and we only need to temporally keep a finite number of historical features, we define the historical data at sliding step k^{th} is a set of temporal features spanning m units before the current k^{th}, $\Gamma^k = \{F^i,\ i \in [k - m, k)\}$, where m is called as the history length. For better visualizing, we express Γ_k as a $m \times\ p$ matrix

$$\Gamma^k = \begin{bmatrix} a_1^{k-m} & a_2^{k-m} & \ldots & a_p^{k-m} \\ a_1^{k+1-m} & a_2^{k+1-m} & \ldots & a_p^{k+1-m} \\ \ldots & \ldots & \ldots & \ldots \\ a_1^{k-1} & a_2^{k-1} & \ldots & a_p^{k-1} \end{bmatrix}. \tag{1}$$

From above definitions, we have the mathematical formulation of change detection. Given $\Gamma^k = \{F^i,\ i \in [k - m, k)$ is the training set and F^k is the test sample, where $m \in \mathbb{N}$ is considered as the number of observations. By assuming that observed features belong to a single class, change detection detects if the current feature value F^k belongs to another class. Mathematically, the problem is to find the statistical model probability $p(F)$. Then if

$$\begin{cases} p(F^k) \geq \epsilon, & \text{context is remained} \\ p(F^k) < \epsilon, & \text{context has changed} \end{cases}, \tag{2}$$

where ϵ is a threshold, which can be chosen in advance or tuned by maximizing the F-measure in statistics.

In other words, given a streaming data X sampled from consecutive contexts that composes n change points and have the length $\mathcal{T}$ time units, window size ω, sliding step ϑ and current timestamp value t_c, the change detection needs to be able to:

- have lightweight computation.
- detect as many as real change points,
- detect changes quickly within a certain delay,
- keep false detection as less as possible.

3.3 Frequency Likelihood Estimation

Our hypothesis is that most change detections are whether too sensitive to outliers or too robust with statistical dispersion (underlying of statistical samples). This characteristic makes these change detections suitable in detecting outliers that are significantly different from the dispersion like normal distribution, but not suitable to detecting changes that cannot be seen clearly. To detect the changes among sounds efficiently, we propose the lightweight Frequency Likelihood Estimator FLE, a nonparametric-based technique preliminarily described in [9]. The technique is termed "frequency likelihood" since we modify the conventional frequency histogram of consecutive samples. We reemphasize that using histogram to detect anomalies has been used in previous work. However, histogram-based techniques have to endure the issues caused by the highly dynamic contexts. In particular, a small amount of samples measured in a dynamic environment typically is not normal distributed, even left-skewed or right-skewed. Determining a fixed threshold also struggled with the dynamic. Furthermore, the conventional histogram based change detection HBOS [8] constructs the bin width solely based on historical samples. However, the number of historical samples are quite short for online detection. Therefore, HBOS has poor performance since the test samples usually fall out of the predefined histogram. Indeed, our new method is able to deal with such issues.

Given temporary dataset $m+1$ samples including training set Γ^k and test sample F^k, which is expressed as a $(m+1) \times\ p$ matrix

$$\mathcal{D} = \begin{bmatrix} a_1^{k-m} & a_2^{k-m} & \dots & a_p^{k-m} \\ a_1^{k+1-m} & a_2^{k+1-m} & \dots & a_p^{k+1-m} \\ \dots & \dots & \dots & \dots \\ a_1^{k} & a_2^{k} & \dots & a_p^{k} \end{bmatrix}, \tag{3}$$

for each attribute column of $\mathcal{D}$, FLE first counts the amount of elements that fall into each disjoint category, also called bin. Let b denote the total number of bins, FLE for each attribute i, $i = 1, \dots, p$, is a function $p_{i,j}$ that must satisfy:

$$m + 1 = \sum_{j=1}^{b} p_{i,j}. \tag{4}$$

In general, b is set to $b < (m+1)$ to make use of the histogram. Unlike HBOS using only m samples, FLE includes the test sample F^k in constructing

the histogram. This solve the case test sample falling out of histogram. It also turns out that the frequency $p_{i,j}$ at bin j will have the highest value for attribute i if there exists a change. That the highest values of frequencies in the modified histogram are almost similar makes it feasible to chose a constant threshold.

Another issue is that the position of the maximum frequency is highly dynamic and depends on which bin most samples fall into. Consequently, repeatedly finding the location of the means consumes more mobile platform's resources. FLE can overcome this issue by taking the absolute values of given data as the data set $\mathcal{D}$. As a result, this technique always pushes the mean and outliers to the most left bin (called least significant bin LSB) and the most right bin (called most significant bin MSB). Therefore, we only need to simply count the frequencies of LSB and MSB. The change probability of test data $|F_i^k|$ of the i^{th} attribute, denoted by $\widetilde{p}(|F_i^k|)$, can be estimated by:

$$\widetilde{p}(|F_i^k|) = p_{i,1} + p_{i,b}, \tag{5}$$

where $p_{i,1}$ and $p_{i,b}$ are the frequencies of LSB and MSB regarding to the i^{th} attribute, respectively. Note that HBOS has to count frequencies of all bins, which requires more computation.

As our goal is to detect anomalies, we aggregate such individual change probability for all attributes of a feature by finding the maximum value. We remark that using join probability will results in limiting the possibility to detect outlier since $\widetilde{p}(|F_i^k|)$ may have very small values. For instance, a buzzer emits considerable amplitude in high frequency but not in low frequency. Choosing the max probability is a suitable solution to increase possibility of detecting the buzzer sound. Hence, the estimate change probability given current feature observation F^k is

$$\widetilde{p}(|F^k|) = max\{\widetilde{p}(|F_i^k|)\},\ i = 1, \ldots, p. \tag{6}$$

To match (6) with the problem definition (2), we take the complement of the change probability as the imaginary density probability:

$$p(|F^k|) \leftarrow 1 - \widetilde{p}(|F^k|). \tag{7}$$

Using above (7), we can detect changes using the condition described by (2).

3.4 Complexity

Given a temporary dataset $\mathcal{D}$ including training set Γ^k and test sample F^k. The total number of entities in $\mathcal{D}$ is $p \times (m+1)$. For each attribute of feature F_i, FLE uses $m+1$ elements. Therefore, it requires $m+1$ operations to find the min and max values. It also needs another $m+1$ operations to count the frequency density of LSB and MSB. Therefore, the complexity, or big-O of FLE is:

$$O(2(m+1)p) \simeq O(mp), \tag{8}$$

where m is the history length and p is the dimension of the data feature. We remark that the conventional histogram HBOS requires b, the number of bins,

iterations to compute frequency density for each element. Therefore, the complexity of FLE is b times less than that of HBOS, which is $O(mbp)$. Moreover, the complexity of FLE is much less than that of OC-SVM $O(m^3p^3)$ and even the well-known lightweight MVN $O(m^2p^2 + p^3)$

Regarding to the framework in Figure 1, the total computational complexity, denoted by C, computed by summing feature extraction and detection technique costs. For example, the computational cost of FLE with the MedianX feature extraction is

$$C = O(q(mp + \omega)), \tag{9}$$

where q is the total quantity of sliding steps.

3.5 Evaluation Metrics

Since detecting changes from streaming data always has a delay, we leverage true positive (TP) and false negative (FP) [13] by adding latency parameter ζ in milliseconds. For better understanding, we interpret these definitions in context change detections. Assume that we expect there is no change of context, null hypothesis (H_0). Let H_a denote the alternative hypothesis, there is a change happened within previous ζ ms. The error types then can be redefined:

- TP_ζ is the total number of sliding steps when real changes are detected within the acceptable latency ζ. The number of correctly detecting a change when H_a is true.
- FP is the total number of sliding steps when the technique wrongly alarms that there is a change. The number of falsely detecting a change when H_0 is true.

The value of ζ is set based on application requirements. However, it should not be less than the period of the shortest context. For example, we set ζ to 1000 ms in our experiment. Base on these errors, we define following metrics to evaluate the performance of change detection: Sensitivity S and Efficiency E.

$$S = 100\frac{TP_\zeta}{n} \tag{10}$$

and

$$E = 100(1 - \frac{FP}{q}). \tag{11}$$

We remark that sensitivity does not count the changes detected later than 1 s. To evaluate how fast changes can be detected, we investigate the required latency L_S to obtain the expected sensitivity S. Together with the aforementioned computational complexity C, these three metrics perfectly represent the change detection problem defined in Section 3.2. High sensitivity S means high detected change points. High efficiency E means less false detection. Remark that execution time of algorithms does not really represent their computational costs since it heavily depends on the implementation optimization. Therefore, we prefer the computational complexity over the execution time.

Feature spaces	Notations	Description	p	Complexity
spectral	PSDP	Power spectral density peak	1	$O(\omega \log_2 \omega)$
	SBC	Spectral subband centroids	4	$O(\omega \log_2 \omega)$
	SBCR	Spectral subband centroid ratio	4	$O(\omega \log_2 \omega)$
cepstral	MFCC	Mel-frequency cepstral coefficients	20	$O(2\omega \log_2 \omega + \omega)$
principal	PCA	Principal component analysis	20	$O(\omega \log_2 \omega + \frac{\omega^2}{4})$
temporal-spectral	DWT	Discrete wavelet transform	4	$O(\omega \log_2 \omega)$
temporal	ZCR	Zero-corssing rate	1	$O(\omega)$
	MinX	Minimum amplitude	1	$O(\omega)$
	MaxX	Maximum amplitude	1	$O(\omega)$
	IqrX	Interquartile range	1	$O(\omega)$
	MedianX	Median amplitude	1	$O(\omega)$
	MeanX	Mean amplitude	1	$O(\omega)$
	StdX	Standard variance amplitude	1	$O(\omega)$

Table 1. Features used in the experiments.

4 Experiments

In this section, we first describe the experiment setup and then analyze the results. The following experiments are aimed to evaluate (i) FLE together with several change detection techniques, that are OC-SVM and MVN and (ii) thirteen types of feature extractions shown in Table 1. In particular, we focus on the performances of such techniques when combined with different feature types, especially light computational features.

4.1 Experiment Setup

Without loss of generality, we generated a database by randomly mixing 200 sound tracks of 10 common sounds (20 tracks per sound), which we may encounter in daily life: babies crying, bells ringing, cars horning, humans coughing, dogs barking, footsteps, glass breaking, gun shoots, human laughing and siren. Sound tracks are downloaded from the free database [1], and have various lengths from 1 second to 10 seconds. Through our experiment on Nexus 7, we experienced realtime processing takes considerable and unpredictable extra delay for each window, up to 500 ms. The major cause comes from microphone hardware, thread interrupts and sound echoes. Therefore, we process offline the database as an online stream of audio with 20 ms windows and 10 ms sliding steps. We extract thirteen different features described in Table 1, which cover most common feature spaces. Remark that the Blackman - Harris window is used to reduce leakage within a Fourier Transform analysis.

We compared the change detection using FLE against OC-SVM and MVN. OC-SVM is one of the best anomaly detection technique in term of sensitivity and MVN is one of the best in term of computational complexity, which are used most for anomaly detection. We set the bin length b to 10 ($b \leq m$) for FLE, the radius of radial basis function kernel γ to $1/p$ for OC-SVM and the significant level of t-distribution α to 0.05 for MVN as common setting values used in other work. The metrics described in Section 3.5 are used to evaluate performance in terms of sensitivity, latency and efficiency. The detection algorithms were repeatedly run with different length of historical sliding steps, from 10 to 50

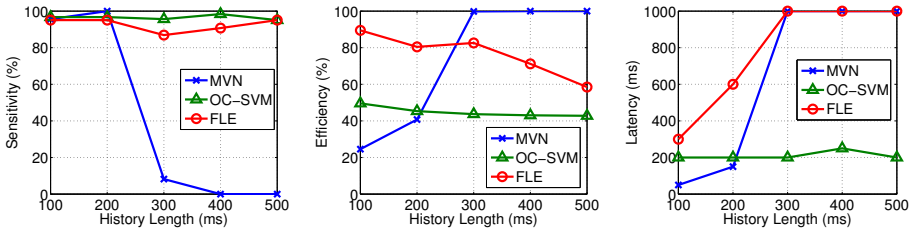


Fig. 3. Change detection performance with the MedianX feature (a) Sensitivity, (b) Efficiency and (c) Latency.

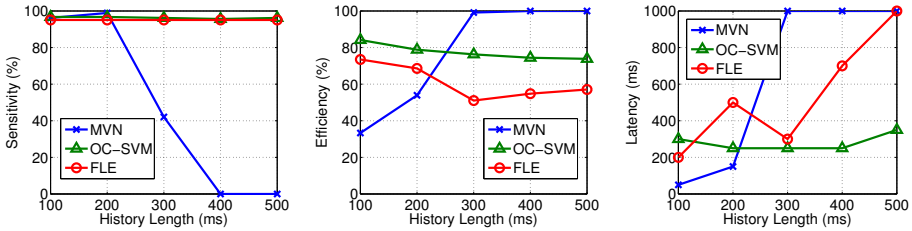


Fig. 4. Change detection performance with the ZCR feature (a) Sensitivity, (b) Efficiency and (c) Latency.

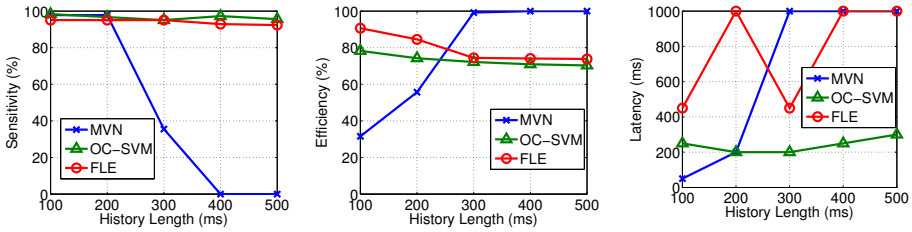
steps, or 100 ms to 500 ms. More than 500 ms is not practical since the time between two consecutive change points can be 1 s.

4.2 Experiment Results

Since the length of a sound track in our database can be 1 s, we only count detected changes with the delay less than 1 s. In addition, the sensitivity S is very critical in most applications. Therefore, the runs having sensitivity below 95% will be ignored in our analysis.

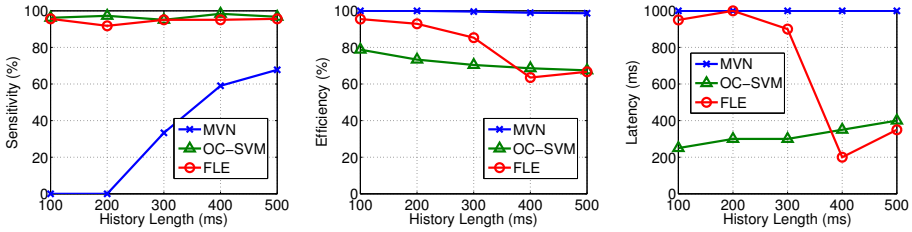
Although we conducted the experiment with various features, we only show results of the features best presenting for each feature space. They are MedianX, SCR, PSDP, MFCC, PCA, and DWT. Figure 3 shows the performance of change detection algorithms with the MedianX feature, the numerical value separating the higher half of window W_t from the lower half. By looking at both Figure 3(a) and (b), we observe that FLE scores best in term of efficiency as shown in Figure 3(b), while being able to detect real changes at high rate as shown in Figure 3(a) as we expected. Conversely, MVN fails to detect change points when the history length is over 200 ms as shown in Figure 3(a). Therefore, the high efficiency of MVN as shown in Figure 3(b) is meaningless. In fact, MVN is not be able to detect changes when the history length is above 200 ms. OC-SVM performance is more suitable when latency is important.

Since the ZCR feature has been used heavily in sound analysis, we consider ZCR in our study too besides the MeanX, even both of them are in temporal space. The results are shown in Figure 4. Efficiency of FLE is around 60%, a bit below our expectation. However, the efficiency is still good enough for most



(a) (b) (c)

Fig. 5. Change detection performance with the PSDP feature (a) Sensitivity, (b) Efficiency and (c) Latency.



(a) (b) (c)

Fig. 6. Change detection performance with the MFCC feature (a) Sensitivity, (b) Efficiency and (c) Latency.

sensing applications. OC-SVM scores best when using the ZCR feature: high sensitivity, high efficiency and short latency. This is due to that OC-SVM is originally designed for classification and ZCR is more informative than MeanX, especially when applying on sounds. Contrary to FLE and OC-SVM, MVN with the ZCR feature performs as poor as with the MeanX feature.

Figure 5 shows the performance metrics of change detection with the PSDP feature when varying the history length parameter. Both FLE and OC-SVM yield good results: high sensitivity ratios, high efficiency ratios and acceptable latency. FLE is better in term of the efficiency and OC-SVM is better in term of the latency. Since the sensitivity ratios of MVN are very poor when the history length is greater than 20 steps, its efficiency and latency results in Figure 5(b) and Figure 5(c) are meaningless.

By looking at performance results with MFCC shown in Figure 6, we draw the same conclusions as with PSDP. The similarity is due to that both PSDP and MFCC represent the frequency aspect.

Figure 7 shows the results when apply changes detection using the PCA feature. And again, both FLE and OC-SVM perform well in terms of sensitivity, efficiency and latency. Contrary to FLE and OC-SVM, MVN is not able to detect more than 95% of real changes. In addition, we observe that there using PCA gives shorter latency than using MFCC. However, using PCA gives low efficiency, which means more falsely detections.

We have investigated change detection methods with features extracted in time and frequency domains. We going on with a feature in time-frequency

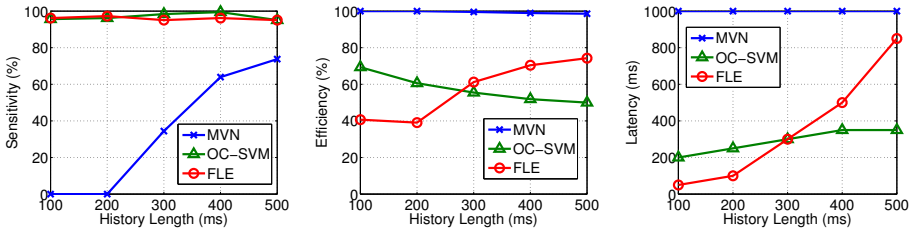


Fig. 7. Change detection performance with the PCA feature (a) Sensitivity, (b) Efficiency and (c) Latency.

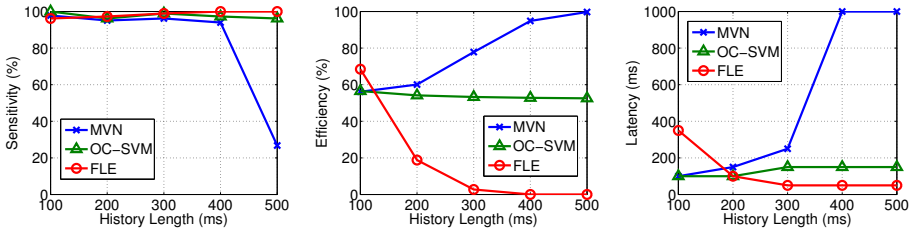


Fig. 8. Change detection performance with the DWT feature (a) Sensitivity, (b) Efficiency and (c) Latency.

domain, the DWT feature. The experimental results are shown in Figure 8. Surprisingly, FLE has very low efficiency. In other words, FLE is not suitable for energy-efficiency sensing with DWT. This is not what we expected. The fact is that DWT provides higher time resolution of high frequencies and lower time resolutions of lower frequencies. A small change in sensory data will results in a significant change in the historical pattern. This leads to very high sensitivity and short latency, but low efficiency. Due to the high resolution, MVN is able to detect well real changes with acceptable efficiency.

Above conclusive results can be summarized as in Table 2 by averaging performance values of history lengths. In the table, the hyphen "-" indicates the value is meaningless. MVN is unsuitable to change detection in our assumption because it is not sensitive to context changes. Overall, OC-SVM performs best in terms of sensitivity and latency. OC-SVM is able to detect more than 97% changes points within approximate 300 ms. We also observed that FLE is more efficiency than others, especially with the featherweight feature MedianX. In addition, the computational cost of the combination of FLE and MedianX is the least, see the numerical example calculated by (9) in Table 3. Since the objective of our problem is lightweight computing and high detection efficiency, FLE with MedianX is a suitable solution for the energy-efficient problem. Moreover, FLE also performs well when using more complex features such as PSDP and MFCC. This is more than what we expected. In addition, the results clearly lead to the conclusion that OC-SVM is suitable for applications of which high sensitivity and fast respond are more important than resource saving.

Performance	MedianX			ZCR			PSDP			MFCC			PDA			DWT		
	S	E	L_S	S	E	L_S	S	E	L_S	S	E	L_S	S	E	L_S	S	E	L_S
MVN	41	-	-	47	-	-	46	-	-	32	-	-	34	-	-	82	-	-
OC-SVM	**97**	45	**210**	**96**	**78**	**280**	**97**	73	**240**	**97**	72	**320**	**97**	**57**	**290**	**98**	**54**	**130**
FLE	93	**76**	780	95	61	540	94	**80**	780	95	**81**	680	96	**57**	360	98	18	120

Table 2. Average performance values of history lengths.

Big O	MedianX	ZCR	PSDP	MFCC	PDA	DWT
MVN	4.55E+07	4.55E+07	1.28E+08	1.38E+10	1.42E+10	6.19E+08
OC-SVM	6.62E+08	6.62E+08	7.44E+08	5.19E+12	5.19E+12	4.16E+10
FLE	***1.46E+07***	**1.46E+07**	**9.66E+07**	**2.35E+08**	**6.47E+08**	**1.02E+08**
HBOS	2.92E+07	2.92E+07	111E+08	5.27E+08	9.39E+08	1.60E+08

Table 3. Numeric computation complexity of change detections with historical length is 200 ms, given the experimental dataset comprised of 81105 sliding windows (160 samples per window).

5 Conclusion

In this paper, we propose the nonparametric-based change detection FLE to detect change points of contexts. The technique is featherweight, sensitive and efficient compared to existing ones such as MVN and OC-SVM. Particularly, FLE estimates the density probability of a test sample using the frequency sum of the least and the most significant bins of the modified histogram. The complexity of FLE is only $O(mp)$, which is much lighter than the complexity of MVN, OC-SVM and even HBOS. Although the proposed method can be applied on various sensory data types, we conducted the experiment on sound data since audio signal is more complex then others in terms of the environmental noises, the tempos and the dynamics. The experimental results are consistent with our analysis. The results show that FLE can detect more than 95% change points in limited time while saving 80% resource compared with standard continuous sensing. This work makes continuous sensing applicable for mobile crowdsensing applications. A testbed including sound recognition has been planned.

Acknowledgements

This work is supported by the SenSafety project in the Dutch Commit program, www.sensafety.nl.

References

1. Findsounds. http://www.findsounds.com
2. Beach, A., Gartrell, M., Akkala, S., Elston, J., Kelley, J., Nishimoto, K., Ray, B., Razgulin, S., Sundaresan, K., Surendar, B., Terada, M., Han, R.: Whozthat? evolving an ecosystem for context-aware mobile social networks. Netwrk. Mag. of Global Internetwkg. 22(4), 50–55 (Jul 2008)

3. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: A survey. Knowledge and Data Engineering, IEEE Transactions on 24(5), 823–839 (2012)
4. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Comput. Surv. 41(3), 15:1–15:58 (2009)
5. Eskin, E.: Modeling system calls for intrusion detection with dynamic window sizes. In: Proc. DISCEX (2001)
6. Fawcett, T., Provost, F.: Activity monitoring: Noticing interesting changes in behavior. In: Proc. SIGKDD. pp. 53–62 (1999)
7. Ganti, R., Ye, F., Lei, H.: Mobile crowdsensing: current state and future challenges. Communications Magazine, IEEE 49(11), 32–39 (November 2011)
8. Goldstein, M., Dengel., A.: Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. In: In Stefan Advances in Konwledge Discovery and Data Mining. pp. 577–593. Lecture Notes in Computer Science, Springer (206)
9. Le, V.D., Scholten, H., Havinga, P.: Flead: Online frequency likelihood estimation anomaly detection for mobile sensing. In: Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication. pp. 1159–1166. UbiComp '13 Adjunct, ACM, New York, NY, USA (2013)
10. Lu, H., Yang, J., Liu, Z., Lane, N.D., Choudhury, T., Campbell, A.T.: The jigsaw continuous sensing engine for mobile phone applications. In: Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems. pp. 71–84. SenSys '10, ACM, New York, NY, USA (2010)
11. Manevitz, L.M., Yousef, M.: One-class svms for document classification. J. Mach. Learn. Res. 2, 139–154 (2002)
12. Ng, A.: Machine learning. https://class.coursera.org/ml/lecture/97
13. Olson, D.L., Delen, D.: Advanced DataMining Techniques. Springer (2008)
14. Rachuri, K.K., Musolesi, M., Mascolo, C., Rentfrow, P.J., Longworth, C., Aucinas, A.: Emotionsense: A mobile phones based adaptive platform for experimental social psychology research. In: Proceedings of the 12th ACM International Conference on Ubiquitous Computing. pp. 281–290. Ubicomp '10, ACM, New York, NY, USA (2010)
15. Sherchan, W., Jayaraman, P.P., Krishnaswamy, S., Zaslavsky, A., Loke, S., Sinha, A.: Using on-the-move mining for mobile crowdsensing. In: Proceedings of the 2012 IEEE 13th International Conference on Mobile Data Management (Mdm 2012). pp. 115–124. MDM '12, IEEE Computer Society, Washington, DC, USA (2012)
16. Wang, Y., Lin, J., Annavaram, M., Jacobson, Q.A., Hong, J., Krishnamachari, B., Sadeh, N.: A framework of energy efficient mobile sensing for automatic user state recognition. In: Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services. pp. 179–192. MobiSys '09, ACM, New York, NY, USA (2009)