A Framework for Unsupervised Change Detection in Activity Recognition

March 27, 2017

1 Introduction

The basic procedure for activity recognition involves: i) collection of labelled data from the subjects that perform sample activities to be recognised ii) classification model generation by using collected data to train and test classification algorithms iii) a model deployment stage where the learnt model is transferred to the mobile device for identifying new unseen activities data. This approach for activity recognition performed the model generation phase on remote systems and pushed the generated model to the system to recognise new user activities. The drawback of this approach is that the model is static and does not reflect possible changes in the distribution of new evolving data. Another approach that aims to eliminates this induced the model by using the user self-annotated data from the device so that the model can be tuned to the individual user.

However, the two approaches are still not immune from changes that may occur in the underlying distribution of the unseen incoming data due to differences in user characteristics. This usually results in decreasing performance in the accuracy of the model as new users with different characteristics begin to use it. For example, a model that is trained to recognise walking activity given a specific data may take a new data from another slightly different distribution that correspond to say jogging for another user and classify it as walking. At this point, the model has suffered from the phenomenon call concept change. The source of changes can be known or unknown. But for activity recognition problem, it has been shown to be caused by some factors such as dissimilarities in the users profiles used during training to the users using the model during recognition Lane et al. (2011); Kwapisz et al. (2011). It may also be caused by the displacement of the sensors and orientation effect on the sensor readings Banos et al. (2014); Ustev et al. (2013).

Hence, various approaches have been developed for model adaptation during their online operation. All of these approaches Zhao et al. (2011); Lane et al. (2011); Abdallah et al. (2015) are blind in the sense that they do not identify concept changes before they start the adaptation process. In the field of stream mining, a related problem exists and two approaches are identified for handling concept drift. We have the informed and uninformed handling techniques Gama

et al. (2014). The informed concept drift adaptation approach attempts to react to the occurrence of concept drift by ensuring that the drift point is detected before taking any action. In contrast to this, uninformed adaptation incrementally and continuously updates the model at each time step a sample or set of samples are integrated into the model. The drawback of uninformed adaptation is that they react slowly to concept drift and consumes system resources as it continuously adapts the old concept which may be required to be replaced out-rightly or maintained without adaptation Gama et al. (2014). Hence an informed adaptation scheme is better and required for better management of concept drift.

Furthermore, many of the approaches Gama et al. (2004); Sobhani and Beigy (2011); Harel et al. (2014) for the detection of concept drift require the presence of labels to detect the drifts. However, in the domain of activity recognition labels are not easy to come by during online recognition. This is because the user will be required to provide a label for each activity being performed. This is impractical and tedious to do. Hence our approach based on unsupervised detection eliminates the need for ground truth to detect changes that caused a decrease in the accuracy of the recognition model.

This paper presents a novel framework called Unsupervised Change Detection for Activity Recognition (UDetect) which extends the Shewart Control Charts used in statistical process control domain for change detection in activity recognition. The approach aims to detect the variability between reference users data and the predicted data from the same or another user. The main contributions of this approach are:

- It can detect changes in the performance of the activity recognition model without using ground truth for error monitoring. Instead, it used a data discrimination method and a base classifier to detect the changes by using the parameters computed from the reference data of each class to discriminate outliers in the new data being classified to the same class.
- The approach is the first method to the best of our knowledge that addresses the problem of detecting within-user and cross-user variations that lead to concept drift in activity recognition.
- The approach is the first to employ statistical process control method for change detection in activity recognition with a robust integrated framework that seamlessly detects variations in the underlying model performance.

The rest of this paper is organised as follows. Section 2 examines the related work in change detection. Section 3 presents the concept change detection framework. Section 4 and 4.2 present the experimental evaluation results and discussions. The paper is concluded in Section 5.

2 Related Work

Concept drift or change is a phenomenon in classification problem where a classifier built to recognize certain concepts from set of training data becomes inaccurate over time because the distribution of the data being classified has changed from the initial distribution known to the model Gama et al. (2014). The changes in the data can manifest either as changes in the class label or changes in the attributes of the new unobserved samples. Changes in the class labels can occur while the attributes themselves remain unchanged. That is, given a sample with a class label say '0', when changes occur the same sample now has label '1'. On the other hand, the attributes of the data may change while the class labels remain unchanged. The two parts of the data can also change simultaneously. In the first and third situations, the classifier will need to be updated with the new emerging distribution of the data while the second situation may or may not affect the decision boundary and hence may not require model update. Another possible but infrequent change is the change in the prior probabilities of classes termed concept evolution that result in emergence of new concepts or merging of existing concept Masud et al. (2011).

More formally, concept drift arises as a result of differences in the relationship between input variable x any target variable y between two points in time t_0 and t_1 i.e. $P(x,y)_{t_0} \neq P(x,y)_{t_1}$ where $x \in R^n$ are the input attributes and $y \in \{y_i : i = 1...c \;\; number\ of\ classes\}$. The changes in this relationship can manifest in the form of changes in the class conditional probability $P(x/y)$ where the attributes values changes for given y_i but the class label y remains unaffected or it may result in posterior probability $p(y/x)$ changes which means the attributes remain unchanged but the class labels changed or there could be a simultaneous changes in posterior and class conditional probability. It is also possible to have prior probability changes leading to emergence of new concepts.

Change that arises from $p(y/x)$ is regarded as real concept drift, while that of $p(x/y)$ is referred to as virtual drift. The virtual drift can also occurred when both $p(x/y)$ and $p(y/x)$ changes simultaneously. The changes in $p(y)$ is referred to as concept evolution.

The following are some of the existing change detection methods in the literature: Online Cumulative Sum Test Page (1954): It is a sequential test that can be applied on stream of numerical data to detect change point. The test monitors the cumulative sum of the attribute of the data stream such as the mean or the error rate of a classification model and alert a change when the value exceeds a pre-set threshold γ. Specifically, the test begins by initializing the sum of the target value to 0. i.e. $S_0 = 0$, then computes the cumulative sum after receiving each observation x_i as $S_{i+1} = max(0, S_i + x_i - \xi)$ where ξ allowed magnitudes of change. The efficacy of this approach depends on the choice of the parameters of the test.

Page Hinkely Test: This is a sequential test for change point detection originally devised by Page in 1954 for change detection in signal processing Page (1954). The approach is similar to CUSUM but rather than computing cumulative sum, it computes two test statistics, the cumulative difference between

Table 1: Categories of Drift

Types of Drift	**Notation**	**Comment**
Real Drift	$p(y/x)_{t_0} \neq p(y/x))_{t_1}$	This drift affect the decision boundary
Virtual Drift	$p(x/y))_{t_0} \neq p(x/y))_{t_1}$	Does not affect the decision boundary
Virtual Drift with Decision Boundary Change	$p(x/y))_{t_0} \neq p(x/y))_{t_1}$ and $p(y/x)_{t_0} \neq p(y/x))_{t_1}$	Simultaneous drift in class conditional probability and posterior probability which affects the decision boundary.
Concept Evolution	$p(y)_{t_0} \neq p(y)_{t_1}$	Concept evolution results in emergence of new classes other than the known classes.

the observed values and their mean up till the moment of the test defined as $c_T = \sum_{t=1}^{T}(xt - \overline{x} - \theta)$, where $\overline{x} = \frac{1}{T}\sum_{1}^{T} x_t$ and θ is the accepted magnitude of tolerable changes and the minimum of c_t defined as $Ct = min(ct, t = 1...T)$. The two parameters are compared as $PHtest = c_t - C_T$ and if the result is greater than a threshold ζ a change is signalled.

Methods based on Statistical Process Control: These methods unlike other sequential approach consider the system being monitored as a process and try to monitors the normal operation of the process from unwanted variations. Once the variation of the process is beyond the acceptable threshold a drift alarm is signalled. Notable methods of process controls include P-charts, X-chart, R-chart, CUSUM chart and a host of other process control charts. An often cited work that considers learning as a process and applied the principles of process control to concept change detection include DDM Gama et al. (2004); Klinkenberg and Renz (1998) . The methods monitors the performance evolution of a classifier and relies on the availability of ground truth to determine when the classifier gives a correct or incorrect prediction. The method incrementally computes the proportion of errors produced by the current model with $p_i = p_{i-1} + (x - p_{i-1})/n$ with x =1 if the prediction is incorrect and x=0 if the prediction is correct. The average error is thus computed incrementally. The standard deviation of the error rate s_i at each time step of the learning process is also computed. Two register p_{min} and s_{min} are maintained and they are updated with pi and s_i respectively whenever $p_i + s_i < p_{min} + s_{min}$. DDM has two thresholds to take decision on the drift: if $p_i + s_i \geq p_{min} + 2 * s_{min}$ it implies a warning level. Subsequent examples after this point are stored in anticipation

of a possible change of concept. If $p_i + s_i \geq p_{min} + 3 * s_{min}$ a drift level is signalled after a series of warning state concept drift is declared, the model induced by the learning method is reset and a new model is learnt using the examples stored since the warning level triggered. The values for p_{min} and s_{min} are reset to 0. The intuition behind this method is that, in the absence of concept drift the error rate should decrease indicating a stationary distribution. However, if the error rate increases significantly it means the classifier is no more in tandem with the distribution of the data. Thus a concept drift has occurred and the model has to be rebuilt. Authors in Baena-Garcıa et al. (2006) extends DDM to account for the distance between error points while Bouchachia (2011) used DDM as a component for their adaptation algorithm to make them informed.

2.1 Methods Using Statistical Test to Detect Changes Between Two sample Distributions

Approaches based monitoring the changes in distribution between two windows can either monitor the performance evaluation of the stream or the parameters obtained from the data stream. These approaches employ statistical testing technique to determine change point between the reference window and a detection window. The following gives a brief description of the methods. Welchs t test: It is a parametric test adapted from the Students t test by Welsh Welch (1947). Given two samples n_1 and n_2 sampled from population N_1 and N_2, the test is used to statistically test the null hypothesis that the means of the population $\overline{N_1}$ and $\overline{N_2}$ with unequal sample variances ${s_1}^2$ and ${s_2}^2$ are equal. The null hypothesis can be rejected depending on the p-value given in Equation 1.

$$p - value = \frac{(\overline{N_1} - \overline{N_2})}{\sqrt{(\frac{{s_1}^2}{n_1}) + (\frac{{s_2}^2}{n_2})}} \tag{1}$$

Kolmogorov-Smirnovs test Chakravarti and Laha (1967): This a non-parametric that is often used to determine if a set of samples are consistent with a reference distribution or if two samples are consistent with the same distribution or not. This test is based on empirical cumulative distribution function (ECDF) computed from the test samples. To test if two samples are consistent with the same distribution, the test computes ECDF for each ordered number of points N $x_1, x_2, ...x_N$ in each sample of size N_1 andN_2 according to Equation 2.

$$ECDF(i) = \frac{\gamma(i)}{N} \tag{2}$$

where $\gamma(i)$ is the number of points less than x_i and the x_i are ordered from smallest to largest value. The Kolmogorov-Smirnov distance between the two test samples is computed as: $D = max_i(ECDF_1(x_i) - ECDF_2(x_i))$. The null hypothesis assuming that the two samples follow the same distribution is rejected with a confidence θ, if: $\sqrt{\frac{(N_1 N_2)}{(N_1+N_2)}} D > K_\theta$

3 Unsupervised Change Detection Framework

The proposed and implemented method of unsupervised change detection presented in this section is a technique that detects changes in the activity recognition model that manifests in the form of reduced model accuracy. The change detection, in this case, is viewed from the perspective of detecting incoherency of the data representing the original model of the activity and the new data that are classified to be the same activity. The central idea of the framework is that if a classifier is trained on a set of data, the model obtained will continue to be accurate if the new unseen data fit coherently with the training data with some level of deviation. But if the new data classified to a particular class varied widely from the pattern of the reference data, it means the data should belong to a different class rather than where it is classified. Therefore, the technique relies on motoring the variation between the refrence data and the new data classified to the class. The method does not assume the presence of ground truth with each arrival of new samples to be classified. Hence, it reflects a realistic scenario for detecting concept drift in activity recognition.

The method employs a classifier that is pre-trained on a set of target concept to classify new samples, and when the classifier is deployed for recognition of new samples, batches of samples classified to the same class are mainitnaed. Intuitively, if the classifier is not misclassifying samples to a class, the distribution of the attributes in the samples classified to the same class should be stable. A change in the distribution of the samples classified to the same class is signalled if the distribution parameters of the samples deviate significantly from the previous reference parameters. The method relies on this assumption and monitors the parameter computed from the batches of data that are classified to the same class. If this parameter is within a threshold no change is detected but if this parameter exceeds a threshold a change is signalled in the particular class of data.

3.1 Conceptual Framework of the Detection Components

The realisation of the technique is formulated as two level architectural framework comprising of the off-line phase and the online phase. The off-line phase is charged with the functionality of extraction and formation of change parameters from the training dataset. It also performs the function of converting the multidimensional datasets into uni-dimensional data. The online phase functions include the classification of new samples and the detection of the change of concepts in the each class of activity present in the datasets. The details of these two components are discussed further in the following sub-sections.

3.2 Conceptual Framework of the Offline Component

This section describes the functional components of the off-line component of the framework. Figure 1 illustrates the off-line component of the entire general

framework. The main functions performed by the off-line component include the following:

1. Windowing.
2. Window summary computation.
3. Change parameter computation.
4. Model Generation.

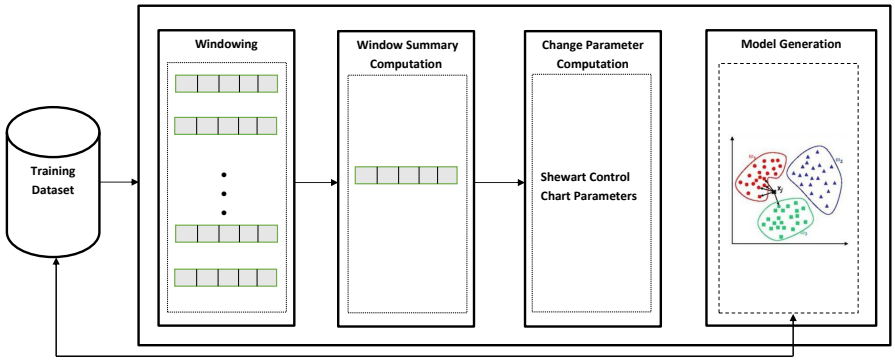

Figure 1: Off-line Sub-Framework for Change Detection

3.2.1 Windowing and Segmentation

The windowing sub-component is the first layer of the entire conglomeration of layers which, performs the function of partitioning and segmenting the incoming data into chunks of fixed sizes.

3.2.2 Window Summary Computation

The window summary computation is the second layer of the off-line component that is saddled with the function of converting the set of multidimensional instances in a fixed size window into a single summary value that will be used as input to the change parameter computation. After accumulating a batch of multidimensional data of a fixed size n from a window of activity data, the summary value of the data is computed as the average distance between each point in the batch of data and the mean of the batch of data. The average distance between each point and the centre (ADTC) is computed using Algorithm 8. The parameter computed is then accumulated and used by the next component to compute the change parameters.

Algorithm 1: Window Summary Computation

$X = getWindowData(n)$
foreach $x_i \in X$ **do**
  $x_d = x_d + x_i$
$\varpi_c = \frac{x_d}{n}$
// n is the size of the window
// $x_{(i)} \in R^n$ are the instances in the window and $\varpi_c \in R^n$ is defined as the centroid of the data in the window computed
foreach $x_i \in X$ **do**
  $E_d = E_d + ((x_i^1 - \varpi_c^1)^2 + (x_i^2 - \varpi_c^2)^2 + ... + (x_i^j - \varpi_c^j)^2)^{\frac{1}{2}}$
$\overline{E_d} = \frac{E_d}{n}$
return $\overline{E_d}$

3.2.3 Change Parameter Computation

The change parameter computation algorithm is presented in Algorithm 2. The algorithm is based on the statistical process control method by adapting the Shewhart individuals control chart parameters Wheeler (1993) to identify the significant variations of new incoming values. The main change parameters are the upper control limit for the range ($UCL_{\overline{R}}$), the lower control limit ($UCL_{\overline{E_d}}$) and upper control limit for the individual window summary values ($LCL_{\overline{E_d}}$). These parameters are used to monitor the new window summary statistics that are computed from the batches of classified activity data for concept change points during the online change detection.

Algorithm 2: Change Parameter Computation

Input: $X = \{\overline{E_{d1}}, \overline{E_{d2}}, ..., \overline{E_{dn}}\}$
// set of values obtained form the window summary
Output: $UCL_{\overline{R}}$, $UCL_{\overline{E_d}}$, $LCL_{\overline{E_d}}$
// change monitoring parameters
foreach $\overline{E_{di}} \in X$ **do**
  $R_{i+1} = \|\overline{E_{di+1}} - \overline{E_{di}}\|$
  $R = R + R_{i+1}$
$\overline{R} = \frac{R}{n}$
$\overline{\overline{E_d}} = \sum_{i=1}^{n} \overline{E_{di}}$
$UCL_{\overline{R}} = 3.27 \times \overline{R}$
$UCL_{\overline{E_d}} = \overline{\overline{E_d}} + 2.66\overline{R}$
$LCL_{\overline{E_d}} = \overline{\overline{E_d}} - 2.66\overline{R}$
return $UCL_{\overline{R}}$, $UCL_{\overline{E_d}}$, $LCL_{\overline{E_d}}$

3.2.4 Model Generation

The final layer of the off-line component is responsible for building a classifier prototype by making use of the training dataset which represents the current knowledge of the system. The main essence of the model generation is to train a learning model to be able to recognise the current pattern of activity that is present in the system. The classifier accuracy is then monitored when it is deployed to the online stage to recognise activity of the new set of data from the same set of users who performed the sample activities or entirely new persons whose characteristics may totally diverge from those present in the training set.

3.3 Conceptual Framework of the Online Components

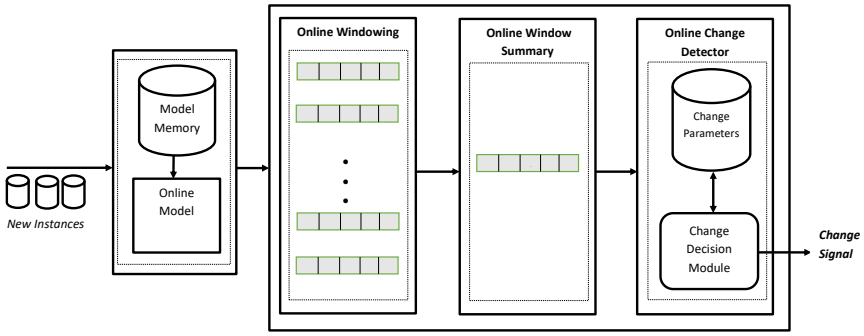

Figure 2: Online Change Detection Architectural Framework

Figure 2 shows the architecture of the online detection method. The components of the framework consists of:

- The Online Model
- Online Windowing and Segmentation Components
- Online Window Summary Computation
- Online Detector.

3.3.1 Online Model

The online model is a carry forward classification model that has been pre-trained during the off-line stage. The model is meant to classify new samples of activity data. As the data are classified they are passed to the dedicated temporary buffers from where they are sent to the appropriate window for the windowing and segmentation operation to be performed on the data.

3.3.2 Online Windowing and Segmentation Components

The online windowing and segmentation component is similar to the offline windowing module as they both perform the function of segmenting the data in the window into fixed size chunks. There are separate windows dedicated to each class of activity and the data in each of the window are processed when the pre-determined number of samples are available in the window. After forming a fixed size chunk of that from the window, the data is passed to the window summary computation component to convert the entire chunk into a single value summary.

3.3.3 Online Window Summary Computation

The online window summary calculation is carried out on the formed data of size n. The computation is akin to the off-line format of the component because the same parameter as the off-line component is required for the detection module. Hence, Algorithm 8 is employed in computing the online window summary as it was done in the off-line computation of the same parameter.

3.3.4 Online Detector

Algorithm 3: Online Change Detector

Input: $UCL_{\overline{R}}$, $UCL_{\overline{E_d}}$, $LCL_{\overline{E_d}}$
`// change parameters computed in the offline phase`
Output: change signal
foreach *instance* x_k *classified to a stream of window activity* **do**
  $x_w = getWindowStep(n)$
  `// Form a window Size from the Stream`
  $w_{statistic} = computeWindowSummary(x_w)$
  `// Compute Window Summary`
  if $(w_{statistic} >= LCL_{\overline{E_d}})$ *or* $(w_{statistic} <= UCL_{\overline{E_d}})$ *and* $(w_{statistic} <= UCL_{\overline{R}}$ **then**
    No change
  else
    change detected
    initiate adaptation

The online detector module is the main component in the online detection framework that determines when a change occur in the new instances being classified. The change can manifest in different ways. One manifestation of change is the diminishing accuracy of the online model that classifies the samples, but this requires knowing the ground truth of the classified sample which is not realistic in this scenario of activity recognition. Therefore, an indirect

approach to detect the change is to determine if the data classified to a class is divergent widely from the original baseline data of the class. Base on this, the online detector uses the already computed change parameters from the training samples to decide whether the new window summary fits coherently within the existing data in the samples or whether it diverges widely from the baseline data. If the new summary value fits coherently with the change parameters no change is detected and the sample is said to be within control but if the value diverges widely then a change is signalled. The online detection algorithm is presented in Algorithm 3. In this algorithm, as new samples are classified into designated windows, and the window summary statistic is computed on a fixed amount of window data, the static is compared to the change parameters and a decision is taken to declare a change or no change.

4 Experimental Study

The objective of the experiments is to identify when the accuracy of the underlying model begins to degrade without having access to the ground truth. This change point is due to the differences between user data used for training and new unseen data during activity recognition. To simulate this scenario, we employed the data of one user for training and another subject data for evaluating the change point detection. In other words, a known amount of one user data is used as training data to create a bespoke up-to-date model and also used to compute change parameters. The training and the test data are then combined and passed to the model so that if there are differences between the distributions of the activity data between the users, the method should be able to identify the change points after the first user data. Hence, the first set of data to test is from the original user while the rest are from another user.

4.1 Experiment with HARS Dataset

This first experiment used the Human Activity Recognition Using Smartphone Dataset-HARS. The experiment evaluates our change detection method that detects the variation in between users data that leads to classifier errors. The results obtained show correlations between the chart of the output parameters and the level of accuracy between any two users. The data plotted are the window summary statistics computed from the chunks of data taking from the moving window designated for each class of activity being recognised. A batch size of 3 was utilised to compute the parameter for all the experiments in this part. Each of the activity types has its own dedicated window for detecting the variability in the data classified to a given class. This variability can indicate the rate of misclassification in the corresponding activity dedicated to the window. The charts show this by out of control points indicating non-uniformity in the data classified to the same window.

Sample results from the experiments are presented in Figures 3 to 8 and Figures 9 to 14. The charts in Figure 3 to 8 are obtained by setting the data

of the user with ID 19 as training set while the combination of user 19 and 14 are set as test data. The first 360 samples of the testing data belong to user 19 while the remaining 323 data points of the total 683 belongs to user 14. The demarcation point is noted to be able to identify the difference between the peak points in the sequences of the two users data.

As shown in the Figure 3, the chart of the sequence of the individual parameter obtained from the window of activity class 'walking' is shown in the upper part while the bottom of the figure indicates the moving range of the parameters. A change is detected at the time step 489 on the individual chart. This is the point where the parameter goes out of the upper control limit. This indicates the point where the distribution of the data changes from the initial data distribution and points to the deterioration in the accuracy of the model. It also indicates that the samples around these time steps are misclassified which makes their computed parameter goes out of control limit. The more the out of control points the more the proportions of the misclassified samples that are classified into this window. Table 2 shows the proportions of misclassification for each class. We can see that there is classification error in this class. Similarly, changes are detected in the activity class 'Walking-Upstairs', 'Sitting' and 'Standing' shown in Figures 4, 5, and 6 respectively. The change points are indicated by the out of control limit points in the individual chart and moving range chart of the change detection parameters. The changes detected show the variation in the activity of the initial user and the test user data.

The proportions of misclassified samples in these classes as shown in Table 2 corroborates the non-homogeneity of the data that are classified to the window dedicated to each class. Thus the approach is able to detect changes in the distribution of the initial user data that belongs to the original activity and those that comes from another user. It should be noted that points after the change points that are within control limits indicate instances from test data have the same and correct class as the initial training data. However, no change is detected in the activity class 'Walking-Downstairs' (Figure 5) and class 'Laying' (Figure 8). This is evident by the absence of out of control points in the two charts for the two classes. This is because there is no variability in the training data of the user and the test data from another user and hence the proportions of their misclassified samples are 0 for each of the two classes as shown in Table 2.

The charts in Figures 9 to 14 are obtained by setting the user 30 data as training set and the combination of users 30 and 2 as test data. The first 383 samples of the testing data belong to user 30 while the remaining 302 data points of the total 685 belongs to user 2. The same observations are recorded as in the user 19 against 14 experiment. The change detected correlates with the proportion of error in the misclassified samples in each of the window dedicated to each class of activity suggesting that the distribution of the test data has changed for some part of the test data. In this second the experiment, changes are detected in activities 'Walking', 'Walking-Downstairs', 'Sitting' and 'Standing'. While no change is detected in activities 'Walking' and 'Laying'. These are shown in Figures 9 - 14. It should be noted that the individual chart

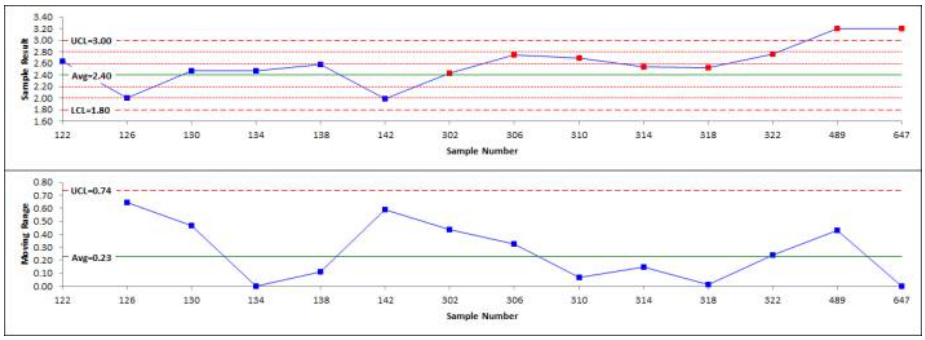

Figure 3: Parameter Chart of Walking Activity in User 19 Against 14

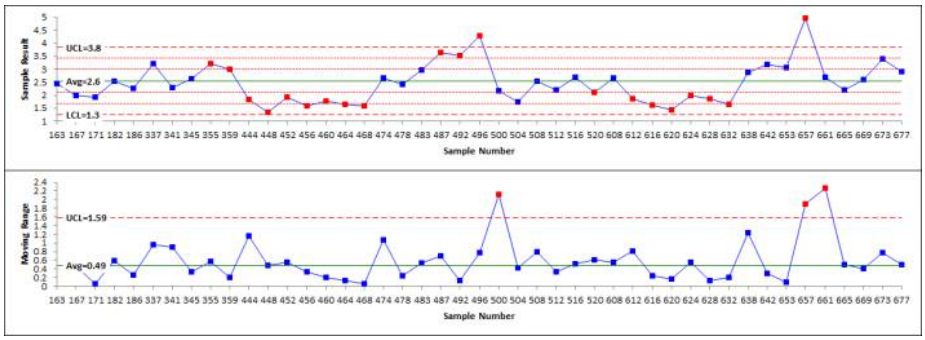

Figure 4: Parameter Chart of Walking-Upstairs Activity in User 19 Against 14

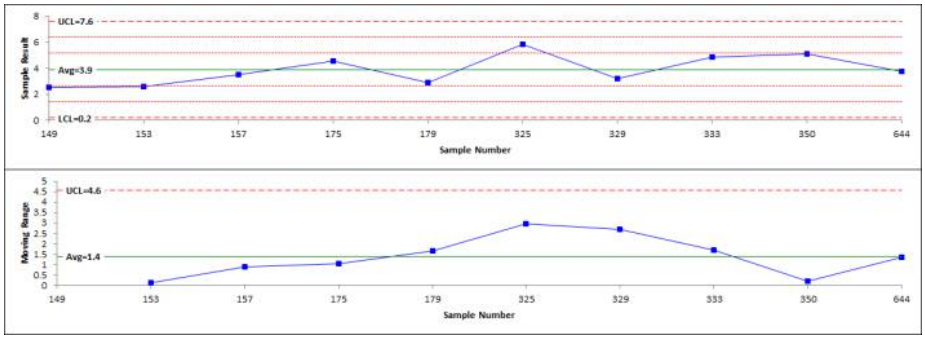

Figure 5: Parameter Chart of Walking-Downstairs Activity in User 19 Against 14

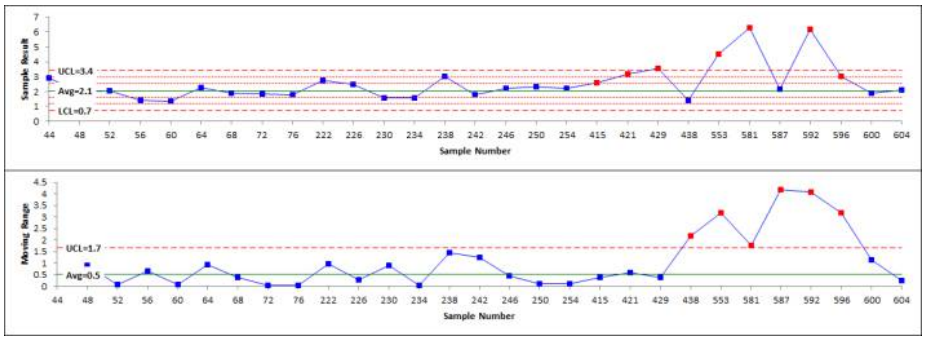

Figure 6: Parameter Chart of Sitting Activity in User 19 Against 14

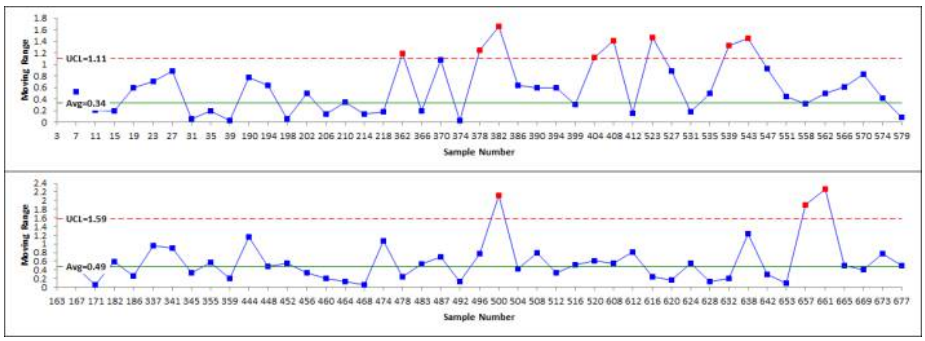

Figure 7: Parameter Chart of Standing Activity in User 19 Against 14

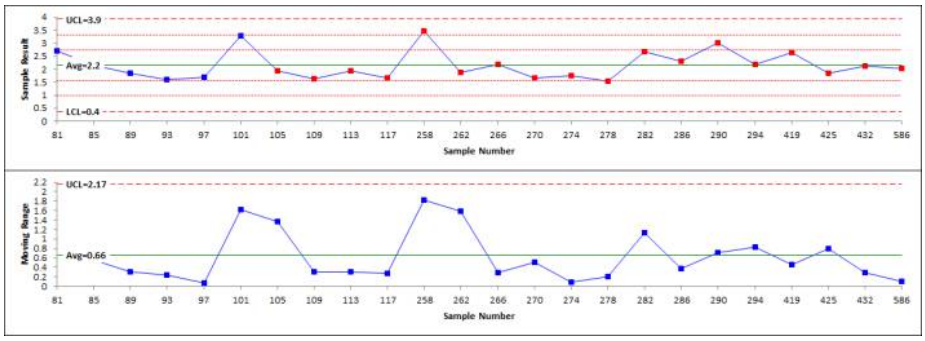

Figure 8: Parameter Chart of Laying Activity in User 19 Against 14

is the main chart that indicates changes. The moving range chart is meant to corroborate the detections. Once the individual chart detects the change there is no need to look at the range chart. But if the range chart detects the change we have to confirm from the individual chart before taken a decision on the admissibility of the change detected.

	Walking	**Walking-Upstairs**	**Walking-Downstairs**	**Sitting**	**Standing**	**Laying**
User 19 Against 4	0.15	0.49	0.00	0.32	0.26	0.00
User30 Against 2	0.10	0.00	0.45	0.19	0.11	0.00

Table 2: Proportion of Error Per Class

4.2 Experiment with Opportunity Dataset

This second experiment used the Opportunity Activity Recognition Dataset. The dataset has more data points and is obtained from more inertia sensors than the first dataset used in the first experiment. Thus, the efficacy of the change detection method is more rigorously tested using this dataset.

Sample results from the experiments are presented in Figures 15 to 18. The charts in the Figures are obtained by setting the data of ADL session 2 of user 3 as training set while the combination of this data and that of ADL session 2 of user 1 are set as test data. The first 27825 samples of the testing data belong to ADL session 1 of user 4 while the remaining 32224 data points of the total 60049 belongs to ADL session 1 of user 1. We noted this demarcation points to be able to identify the points where the window statistic parameters go beyond the change detection parameters as the sequence of parameters are plotted for detecting the change points.

Figure 15 shows the chart of the window summary parameter obtained from the window designated for 'Standing' activity. The upper part of the chart shows the individual parameter value enmeshed with change detection parameter limits while the lower part of the figure indicates the moving range of the parameters.

A significant change is detected at the time step 35571 on the individual chart. This is the point where the parameter goes out of the upper control limit. This indicates the point where the distribution of the data changes from the initial data distribution and points to the deterioration in the accuracy of the model. It also indicates that the samples around these time steps are misclassified which makes their computed parameter goes out of control limit. The more the out of control points the more the proportions of the misclassified samples that are classified into this window.

Similarly, changes are detected in the activity class 'Walking', and 'Lying' as shown in Figures 16 and 17 respectively. The change points are indicated by the out of control limit points in the individual chart and moving range chart of the change detection parameters. The changes detected show the variation in the activity of the initial user and the test user data. However, there is no change detected in the 'Sitting ' activity as shown in Figure 18. This is especially so as

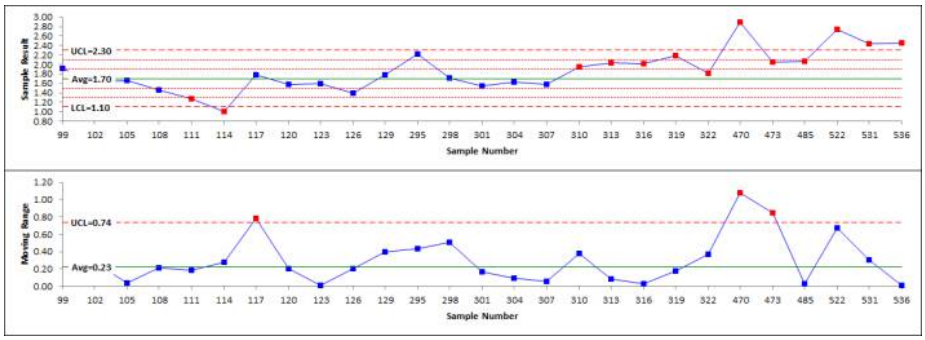

Figure 9: Parameter Chart of Walking Activity in User 30 Against 2

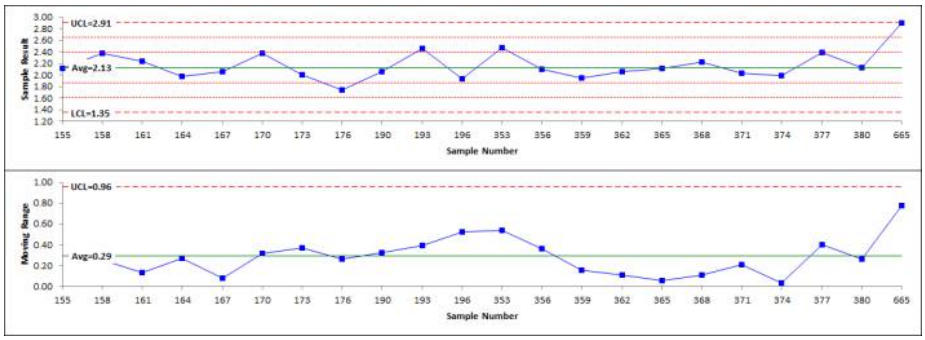

Figure 10: Parameter Chart of Walking-Upstairs in User 30 Against 2

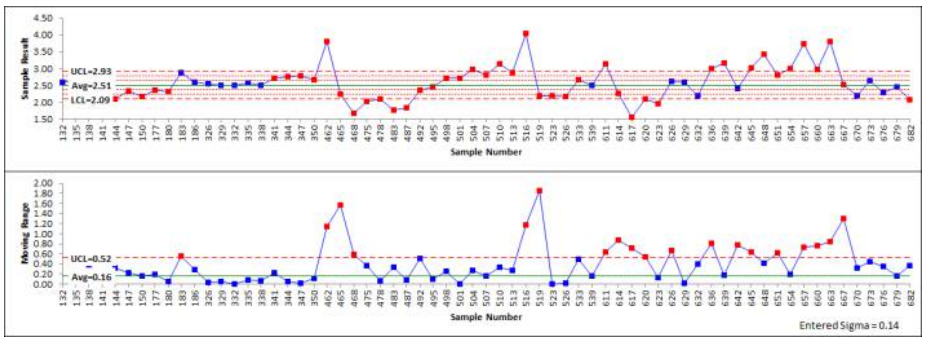

Figure 11: Parameter Chart of Walking-Downstairs Activity in User 30 Against 2

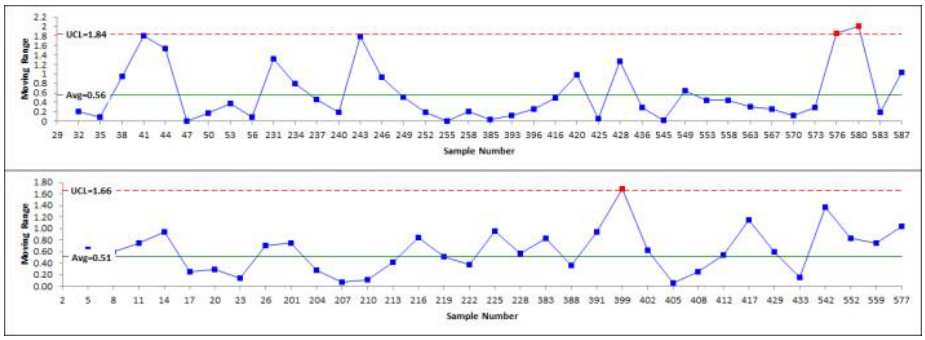

Figure 12: Parameter Chart of Sitting Activity in User 30 Against 2

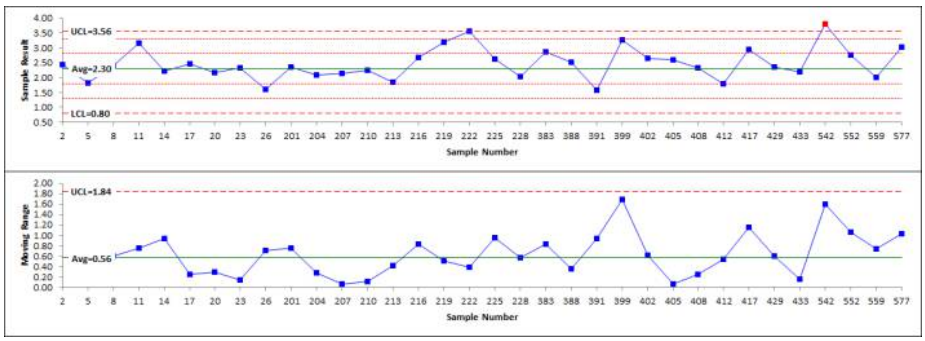

Figure 13: Parameter Chart of Standing Activity in User 30 Against 2

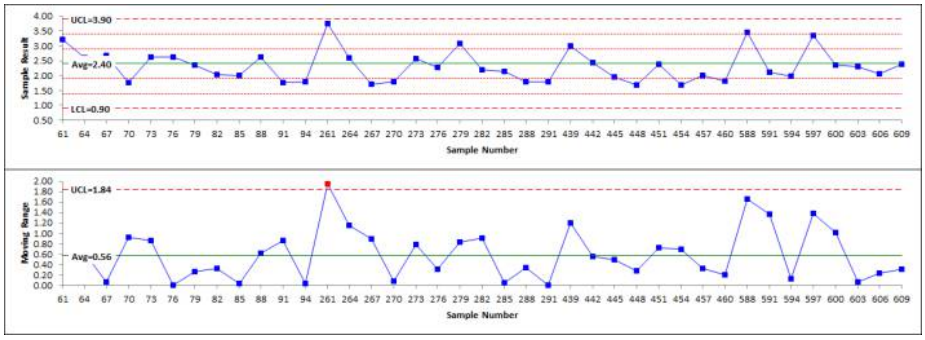

Figure 14: Parameter Chart of Laying Activity in User 30 Against 2

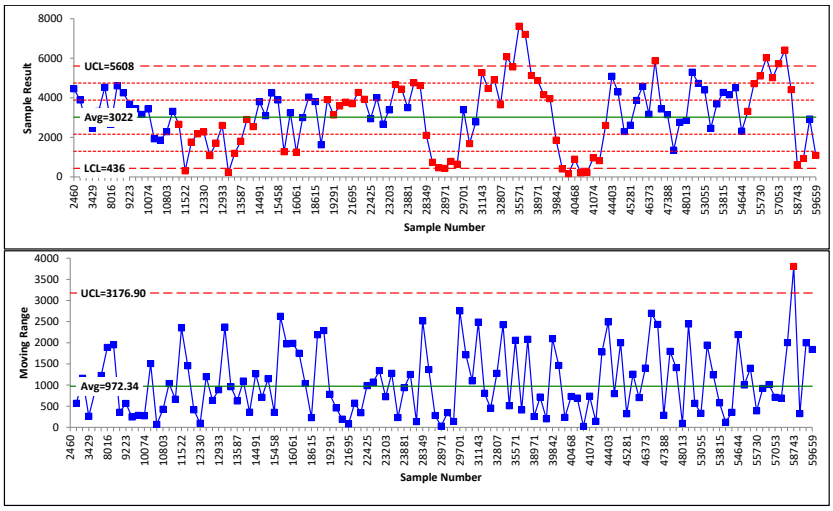

Figure 15: Parameter Chart of 'Standing' Activity in User 3 ADL 2 Against User1 ADL2

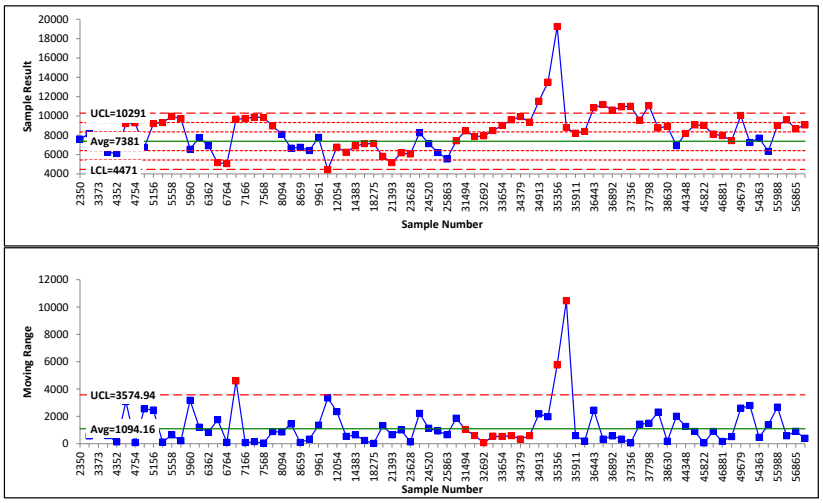

Figure 16: Parameter Chart of 'Walking' Activity in User 3 ADL 2 Against User1 ADL2

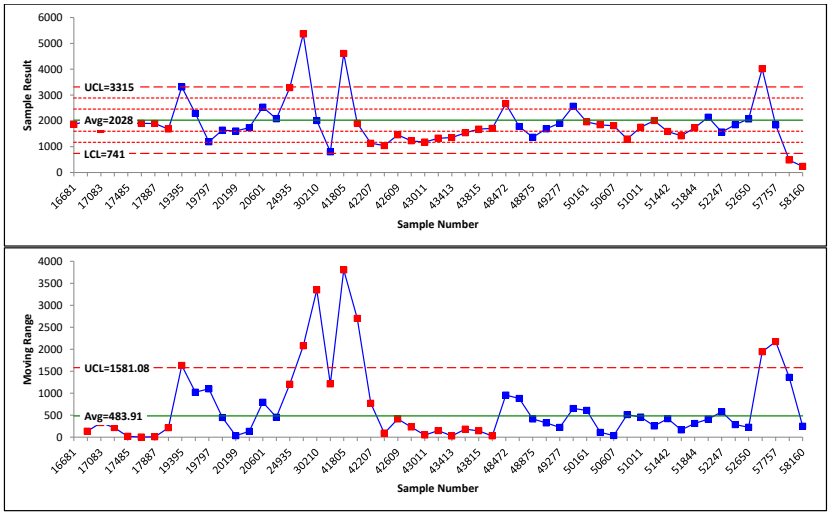

Figure 17: Parameter Chart of 'Lying' Activity in User 3 ADL 2 Against User1 ADL2

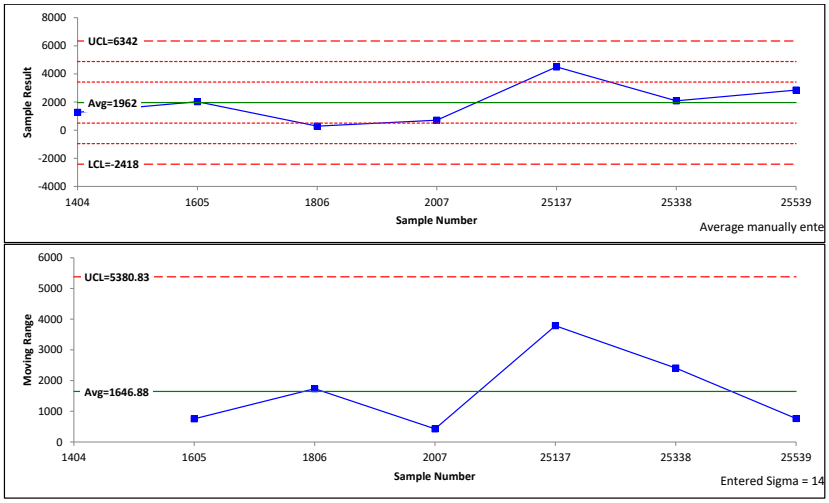

Figure 18: Parameter Chart of 'Sitting' Activity in User 3 ADL 2 Against User1 ADL2

there is no data from the other test users data are classified to this window of activity.

These results indicate the applicability and efficacy of the implemented method in detecting concept change point in activity recognition that involves diverse user characteristics and profiles.

4.3 Comparative Evaluation of UDetect with Drift Detection Method(DDM)

DDM relies on the error rate to detect a changes in the model accuracy. UDetect on the other hand, does not use error rate because it does not assume the presence of ground truth to determine the classification error. In other to compare this two methods, a user data is set, as the training set and series of other user data were streamed for classification with KNN. Table 3 shows the total

Test Data	UDetect	DDM
user 1	0	0
user 2	8	0
user 3	9	4
user 4	6	3

Table 3: Changes Detected By UDetect and DDM in Opportunity Dataset

number of changes detected by UDetect and DDM when user1 data was set as training set and user 1,2, 3, and 4 were used as a test set one after the other. A window size of 200 is used for all the data to compute the change statistics and the detection parameters. As expected no change was detected in user 1 by both UDetect and DDM. This is because the data came from the same user and there are no differing characteristics. However, UDetect detected 8 changes in user 2 data while DDM detected none. This shows that UDetect is more accurate since experiments have shown that the accuracy of using user 1 against other users' data is very low. Similarly, the number of change points detected by UDetect in user 3 and 4 is more than that of DDM. This implies that the proposed method is able to detect changes that reflect the level of accuracy of the underlying model.

5 Conclusion

This paper has presented a novel concept change detection method for activity recognition. The method is based on processing chunks of data classified to the same class and extract parameter that characterised each chunk. The average distance to centre parameter computed from each batch is monitored by using Shewart charts as the change point detector to identify outlier peaks that represent the drift point.

Such points indicate that the model is misclassifying the samples to the wrong class and thus need to be diagnosed to react to drift. The main benefits

of this method compared to the traditional drift detection approach in data stream domain is that it does not rely on the ground truth to detect drift in the data and thus is the more realistic approach for activity recognition. The method is evaluated using real activity recognition dataset obtained from mobile phones of diverse subjects and another large dataset that is obtained from more complex inertial sensors attached to users who perform the designated activity. The result indicates the method is able to identify the precise drift point in the data. Also, comparison of the method with DDM approach reveals that UDetect is able to detect more changes than DDM.

References

Z. S. Abdallah, M. M. Gaber, B. Srinivasan, and S. Krishnaswamy. Adaptive mobile activity recognition system with evolving data streams. *Neurocomputing*, 150:304–317, 2015.

M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno. Early drift detection method. In *Fourth international workshop on knowledge discovery from data streams*, volume 6, pages 77–86, 2006.

O. Banos, M. A. Toth, M. Damas, H. Pomares, and I. Rojas. Dealing with the effects of sensor displacement in wearable activity recognition. *Sensors*, 14 (6):9995–10023, 2014.

A. Bouchachia. Fuzzy classification in dynamic environments. *Soft Computing*, 15(5):1009–1022, 2011.

I. M. Chakravarti and R. G. Laha. Handbook of methods of applied statistics. In *Handbook of methods of applied statistics*. John Wiley & Sons, 1967.

J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Advances in artificial intelligence–SBIA 2004*, pages 286–295. Springer, 2004.

J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, 46(4):44, 2014.

M. Harel, S. Mannor, R. El-Yaniv, and K. Crammer. Concept drift detection through resampling. In *ICML*, pages 1009–1017, 2014.

R. Klinkenberg and I. Renz. Adaptive information filtering: Learning drifting concepts. In *Proc. of AAAI-98/ICML-98 workshop Learning for Text Categorization*, pages 33–40. Citeseer, 1998.

J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

N. D. Lane, Y. Xu, H. Lu, S. Hu, T. Choudhury, A. T. Campbell, and F. Zhao. Enabling large-scale human activity inference on smartphones using community similarity networks (csn). In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 355–364. ACM, 2011.

M. M. Masud, J. Gao, L. Khan, J. Han, and B. Thuraisingham. Classification and novel class detection in concept-drifting data streams under time constraints. *Knowledge and Data Engineering, IEEE Transactions on*, 23(6): 859–874, 2011.

E. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.

P. Sobhani and H. Beigy. *New drift detection method for data streams*. Springer, 2011.

Y. E. Ustev, O. Durmaz Incel, and C. Ersoy. User, device and orientation independent human activity recognition on mobile phones: Challenges and a proposal. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, pages 1427–1436. ACM, 2013.

B. L. Welch. The generalization ofstudent's' problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35, 1947.

D. J. Wheeler. *Understanding Variation: The Key to Managing Chaos*. SPC Press, second edition edition, 1993.

Z. Zhao, Y. Chen, J. Liu, Z. Shen, and M. Liu. Cross-people mobile-phone based activity recognition. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2545–2550. AAAI Press, 2011.