Online Change Detection for Timely Solicitation of User Interaction

Timothy Patterson[1], Sally McClean[2], Chris Nugent[1], Shuai Zhang[1], Leo Galway[1], Ian Cleland[1]

[1] School of Computing and Mathematics, University of Ulster at Jordanstown, BT37 0QB, UK {t.patterson, cd.nugent, s.zhang, l.galway, i.cleland}@ulster.ac.uk

[2] School of Computing and Information Engineering, University of Ulster at Coleraine, BT52 1SA, UK {si.mcclean@ulster.ac.uk}

Abstract. The accurate detection of changes has the potential to form a fundamental component of systems which autonomously solicit user interaction based on transitions within an input stream, for example accelerometry data obtained from a mobile device. This solicited interaction may be utilized for diverse scenarios such as responding to changes in a patient's vital signs within a medical domain or requesting activity labels for generating real-world labelled datasets. Within this paper a change detection algorithm is presented which does not require knowledge of the underlying distributions, can run in online scenarios and considers multivariate datastreams. Results are presented demonstrating practicable potential with 99.81% accuracy and 60% precision for real-world accelerometry data.

Keywords: Multivariate change detection, Online change detection, Soliciting user interaction

1 Introduction

The timely engagement between a system and the end user is a fundamental concept within domains ranging from healthcare, for example responding to changes in a patient's vital signs [1] to machine learning, for example engaging with users after the commencement of a new activity to solicit activity labels [2]. Nevertheless, such engagement may be expensive in terms of user time and, in the event of excessive requests for interaction may degrade the usability and relevance of the system. It is therefore necessary to have a method of change detection which can be utilized to solicit interaction from the user when a transition is detected.

Within [2] we present our previous work in the development of a mobile-based framework for the large-scale gathering and labelling of activity data. Such labels have the potential to provide an invaluable resource to the research community by facilitating the training of supervised algorithms using truly representative data collected in a free-living environment. The developed framework enables

the labelling of data via an Android based mobile application and currently contains two primary components: an activity recognition (AR) module and a labelling prompt module. The AR module is responsible for identifying the user's current action and contains both stationary activities, for example 'standing still' and non-stationary activities, for example 'running'. The AR module detects activities based on 3 second windows with a total of 3 consecutive windows (i.e. 9 seconds of data) being required before an activity is labelled. Upon detecting a transition from an activity to 'standing still' the AR module initiates the label prompting module. This module displays a screen to the user enabling them to click an icon representing the activity they have just transitioned from.

Whilst this approach enables data collection in a relatively free-living scenario there are two fundamental constraints imposed by the overall framework. Firstly, the requirement that a user transitions from an activity to 'standing still' results in potentially informative inter-activity data pertinent to real-world situations being lost. For example, the sequence {stand still - walk - jog - run - jog - walk - stand still} may be considered as a typical series of activities for running. Such inter-activity data could subsequently be utilized for training models which predict, in real-time the activity that a user is transitioning to, thus expediting the AR process by enabling the selection of appropriate classifiers. Secondly, the number of false prompts that a user receives is, to some extent controlled by requiring 3 consecutive 3 second windows containing the same activity. This results in a delay between the user finishing an activity and receiving a prompt. Furthermore, this approach to controlling the number of erroneous prompts results in activities which may have an inherently short duration, for example traversing a short flight of stairs remaining undetected by the AR module.

A popular technique within the literature for detecting changes is the Cumulative Sum Control Chart (CUSUM) which has been utilized in applications such as identifying changes in cardiovascular events [3] and detecting the progression of eye disease [4]. A particular criticism of CUSUM is that it may be inaccurate when identifying sudden shifts that are not from the same distribution [5] which may yield it ineffective when detecting changes based on accelerometry data.

In this paper we present an online multivariate change detection algorithm which may be utilized in a real-time mobile-based activity labelling framework to autonomously solicit user interaction upon detecting a change. We develop the univariate change detection algorithm by Jain and Wang [6] resulting in two main contributions. Firstly, the algorithm in [6] is extended to consider multivariate data streams thus enabling the incorporation of multiple sensors into the change detection process consequently enhancing the overall accuracy of the algorithm. Secondly, we compute the test statistic for *all* positions within a window as opposed to the *most likely point* proposed by Jain and Wang. This has the primary advantage of enabling covariance between sensor observations to be considered in the hypothesis stage.

The remainder of this paper is structured as follows: in Section 2 the Multivariate Change-Detection algorithm is presented. In Section 3 we provide an

overview of the experimental setup with subsequent results presented in Section 4. Finally, Conclusions and Future Work are discussed in Section 5.

2 Multivariate Change-Detection Algorithm

Consider a data stream of length q consisting of data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_q$. Each data point $\mathbf{x}_q$ is a p element vector where p is the number of sensor observations for each variable. The data stream may contain points from multiple distributions, for example $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{k-1}$ may have distribution D_1 whilst $\mathbf{x}_k, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_q$ may have distribution D_2. It is therefore the overall aim of the algorithm to identify the position in the data stream of change points k.

The change-detection algorithm follows an hypothesis-and-verification principle. In the hypothesis step a point is detected within the window under consideration which maximizes the test statistic. In the second stage the hypothesis that a detected change point is significant is verified.

2.1 Hypothesis Generation

In the hypothesis generation stage we pass an analysis window of length n over the datastream assuming that there is a maximum of one change point per window. The movement of the window over the datastream may be either distinct in which case the start of a new window (other than the first) is at position $m+cn+1$ where m is the padding size and c is the number of previous windows. Alternatively, a sliding window version of the algorithm may be executed with the start position incremented by a predetermined number of data points. For ease of notation we denote the data points within a window as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ regardless of their actual position within the data stream. Following Jain and Wang [6] we pad either side of the window with m points such that the analysis window contains data points $\mathbf{x}_{1-m}, \ldots, \mathbf{x}_{n+m}$ therefore containing a total of $n+2m$ data points. This padding is necessary to accurately detect change points which occur at the extremities of the window and is particularly crucial when executing a distinct window version of the algorithm.

Within each window we slide an index variable, l, $1 < l \leq n$ subsequently computing summary statistics of the component distributions separated at l. Specifically, we compute the means, $\bar{\mathbf{f}}_1(l)$ and $\bar{\mathbf{f}}_2(l)$, which contain the mean of observations, in addition to variance-covariance matrices, $\mathbf{S}_1(l)$ and $\mathbf{S}_2(l)$, which contain the variance of observations in the diagonals and their covariance in the off-diagonals. To ensure that the change detection algorithm can operate in online scenarios we compute $\bar{\mathbf{f}}_1(l)$, $\bar{\mathbf{f}}_2(l)$ and $\mathbf{S}_1(l)$, $\mathbf{S}_2(l)$ recursively. Thus as index l increments to position $l+1$ the summary statistics are calculated as follows:

$$\bar{\mathbf{f}}_1(l+1) = \frac{m+l-1}{m+l}\bar{\mathbf{f}}_1(l) + \frac{f(\mathbf{x}_{l+1})}{m+l}, \tag{1}$$

$$\bar{\mathbf{f}}_2(l+1) = \frac{n+m-l+1}{n+m-l}\bar{\mathbf{f}}_2(l) - \frac{f(\mathbf{x}_{l+1})}{n+m-l}, \tag{2}$$

$$\mathbf{S}_1(l+1) = \frac{m+l-1}{m+l}\mathbf{S}_1(l) + \frac{1}{m+l-1} \times [\mathbf{x}_{l+1} - \bar{\mathbf{f}}_1(l+1)]^{'}[\mathbf{x}_{l+1} - \bar{\mathbf{f}}_1(l+1)], \tag{3}$$

$$\mathbf{S}_2(l+1) = \frac{n+m-l+1}{n+m-l}\mathbf{S}_2(l) - \frac{1}{n+m-l} \times [\mathbf{x}_{l+1} - \bar{\mathbf{f}}_2(l+1)]^{'}[\mathbf{x}_{l+1} - \bar{\mathbf{f}}_2(l+1)]. \tag{4}$$

Having calculated summary statistics before and after l we proceed to compute the F statistic at position l, F_l as follows [7]:

$$F_l = \frac{n_1 + n_2 - p - 1}{p(n_1 + n_2 - 2)}T^2, \tag{5}$$

where $n_1 = m + l - 1$, $n_2 = n + m - l + 1$, p is the number of variables and T^2 is the Hotelling T-squared statistic calculated as [7],

$$T^2 = (\bar{\mathbf{f}}_1 - \bar{\mathbf{f}}_2)^{'}\left\{\mathbf{S}_p\left(\frac{1}{n_1} + \frac{1}{n_2}\right)\right\}^{-1}(\bar{\mathbf{f}}_1 - \bar{\mathbf{f}}_2), \tag{6}$$

where $\mathbf{S}_p$ is the pooled variance-covariance matrix,

$$\mathbf{S}_p = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}. \tag{7}$$

Under the null hypothesis (i.e. equal distributions) and assuming Gaussian distributions this has an F distribution [7]. We choose the point l which maximizes F_l as the most likely change point within a window and proceed to the hypothesis verification phase.

2.2 Hypothesis Verification

An hypothesis verification stage is executed to prove or disprove the null hypothesis that a significant change did not occur at point l. Firstly, we compute the probability of finding an F value lower than that calculated in Equation 5 resulting in b. The F Cumulative Distribution Function is utilized for this phase with p and $n_1 + n_2 - p$ degrees of freedom. As multiple statistical tests are being simultaneously performed within the window it is necessary to adjust our confidence value, α to reflect the confidence for the entire window and not a single, isolated value. We therefore use a Bonferroni correction [8] to compute a threshold t as:

$$t = \alpha/n. \tag{8}$$

Secondly, we reject the null hypothesis (i.e. a significant change did occur) if,

$$(1 - b) < t, \tag{9}$$

and the position $\mathbf{x}_l$ is subsequently labelled as a change point within the datastream.

When one examines the accelerometry data it can be seen that the actual values at a change point increase or decrease over a range of data points. Thus when executing a sliding window version of the algorithm change points are detected which are adjacent as the datapoints become increasingly indicative of a 'significant' change. We therefore define a further parameter, a which indicates the number of adjacent detected change points required before the algorithm would alert the user to a 'real' change occurring. Furthermore, once a change point satisfying this criteria is detected, sequential adjacent change points are considered as indicative of the same event and would therefore not be dispatched to the user.

3 Experimental setup

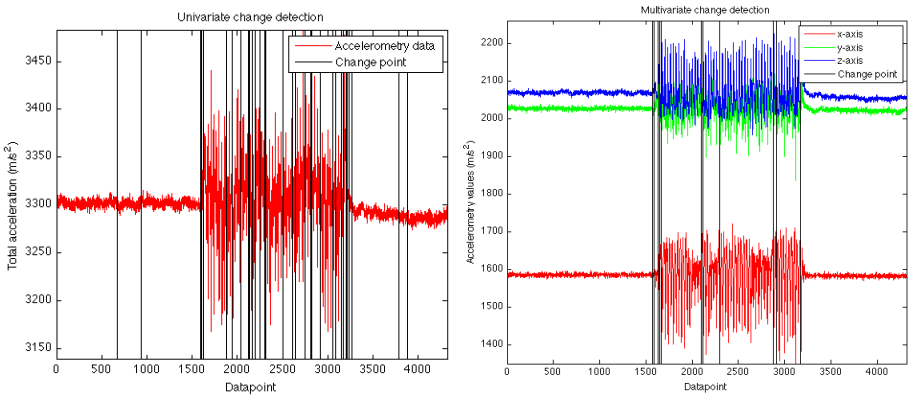
To facilitate the evaluation of change detection algorithms, accelerometry data was captured from a healthy participant wearing two Shimmer wireless sensing platforms [9]. The Shimmers were placed in the middle of the participant's left pectoral and at mid-point between the thigh and knee on the anterior of the participant's right leg. These Shimmer positions enabled anterior-posterior and lateral movements of the subject to be effectively captured [10].

The participant performed multiple tests with each containing two high-level scenarios: 'arrive home' which comprised the subset of activities {ascend stairs, walk, sit down} and 'leave home' which consisted of {stand up, walk, descend stairs} [10]. For the stationary activities 'sit' and 'stand' the participant sat or stood for approximately 30 seconds and then transitioned to the opposite stationary activity. When measuring accelerometry data for non-stationary activities the participant stood for approximately 30 seconds, proceeded to perform the activity for approximately 30 seconds and then transitioned to standing. Throughout activity execution accelerometry data was wirelessly streamed to a receiving computer via the IEEE 802.15.1 Bluetooth communications protocol.

4 Results

Within this section we present results comparing the classification performance of the proposed multivariate change detection algorithm with the univariate approach proposed by Jain and Wang [6]. To enable the evaluation of the univariate approach the magnitude of acceleration was calculated from the x,y and z axes of the captured data and used as input to the univariate algorithm. In order to quantitatively evaluate the algorithms, the start and end data points of each activity for two tests was manually labelled.

We define the positive and negative detection cases as follows: a true positive (TP) is a correctly identified change. When determining true positives a quarter second buffer was included at either side of the manually labelled change point to accommodate subjectivity errors inherent in manual labelling. Thus,



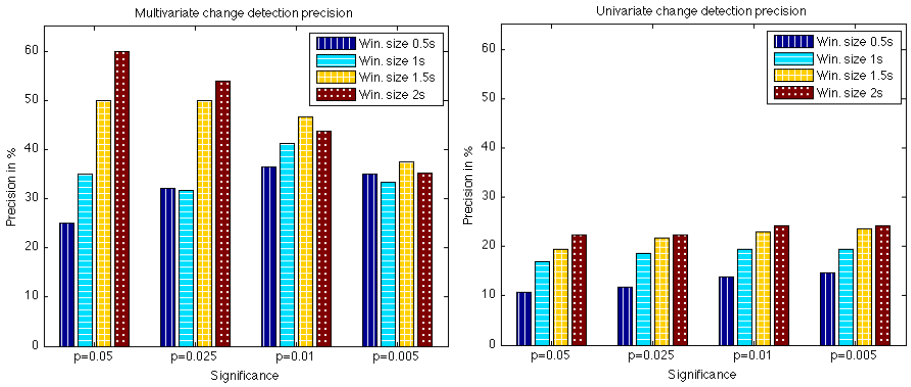
(A) Univariate change detection as proposed by Jain and Wang

(B) Presented multivariate change detection algorithm

Fig. 1: Example sliding window change detection results for the activity 'stand still - descend stairs - stand still'. The window size was 1 second with confidence $p = 0.025$. The number of neighbours required, a was 1.

a detected change point was considered true if its index in the datastream, $l \in \{z - (f/4) \dots z + (f/4)\}$ where z is the index in the datastream of the manually labelled change point and f is the sampling frequency in Hz. A further consideration when interpreting true positive results is the level of granularity required by the host application. For example in Figure 1 the graph of accelerometry values with detected change points are displayed for the activity 'descend stairs'. There are three possible levels of change for this activity: firstly, there is the transition from standing still to descending stairs; secondly, there are transitions present at each individual step as the subject traverses the stairs; thirdly there are 11 landings present for a total of 102 stairs resulting in the dataset containing multiple transitions between traversing stairs and walking for a short duration (approximately 1 - 4 steps). Bearing in mind the target application of soliciting user interaction after a transition we only consider the primary transitions between high-level activities, for example 'standing still - descend stairs - standing still'. It is useful, however, to note that the level of granularity required can be readily modified by choosing the type of algorithm used, i.e. sliding or distinct, the size of the consideration window and by adjusting the significance level in Equation 8. We define a true negative (TN) as a non-transitional point which is not labelled as a change.

A false positive (FP) is a non-transitional point which is highlighted by the algorithm as a change. In terms of user experience this type of error is likely to be the most detrimental as it will result in them receiving unintuitive requests for interaction. A false negative (FN) occurs when the algorithm fails to detect a change in the user's activity. Bearing in mind our target application this type of error would primarily impact upon the quality of the dataset labels as as the labelling program would not request user interaction.



(A) Multivariate (B) Univariate

Fig. 2: Comparison of change detection precision results for the sliding window version of the algorithm

Due to a disproportionately high number of true negatives in the data, accuracy defined as $\frac{TP+TN}{TP+TN+FP+FN}$ was relatively high ranging from 99.25% to 99.81% for the multivariate approach and 97.92% to 99.22% for the univariate approach. Thus, when evaluating the algorithm we focus on precision defined as $\frac{TP}{TP+FP}$. This is primarily due to our application where unintuitive requests for interaction may degrade user experience.

In Figure 2 the precision of the sliding window version of the algorithm with increments of one data point is presented. The developed multivariate change detection algorithm consistently achieved higher precision than the univariate approach. A maximum precision of 60% was achieved with a window size of two seconds and significance $p = 0.05$. As the required confidence increased the precision decreased for the multivariate approach; this was caused by true change points not satisfying the hypothesis verification stage and therefore being incorrectly labelled as a non-transitional point. The threshold used in the hypothesis verification is directly related to window size (Equation 8). Thus as the window increases in size the threshold decreases resulting in a hypothesized change point requiring a higher F value to reject the null hypothesis. In the multivariate results the accuracy generally increased with window size for $p = 0.05$ and $p = 0.025$. As the confidence increased ($p = 0.01$ and $p = 0.005$) the window size had less impact upon precision with similar results achieved for all window sizes where $p = 0.005$. The minimum precision achieved for the sliding window version of the univariate approach was 11% with a maximum of 25%. The precision of the univariate approach increased with required significance and window size. This was due to a reduced number of false positives caused by a hypothesized change point requiring a higher test statistic value to reject the null hypothesis.

5 Conclusions and Future Work

Within this paper we have presented an approach to change detection which may operate in real-time scenarios, does not require knowledge of the underlying distribution(s) and can incorporate multivariate datastreams. The developed

algorithm outperformed the real-time univariate approach by Jain and Wang [6] for both accuracy and precision metrics.

Whilst we have evaluated our approach using real sensor data a key part of future work will be to generate a synthetic dataset thus providing a resource for quantitatively determining the impact of parameter choices. In particular we wish to measure the trade-off between accuracy and computational performance of window size, confidence values and the type of algorithm used. Additionally, bearing in mind the controlled nature of the presented experiments a further part of future work will be to incorporate our multivariate approach to change detection into real-world systems such as mobile-based applications for gathering and labelling activity data. This will enable us to evaluate the algorithm using data collected from multiple individuals within a free-living scenario.

6 Acknowledgements

The authors acknowledge support from the EPSRC through the MATCH programme (EP/F063822/1 and EP/G012393/1) and to Invest N.I. under R and D grant RD0513844.

References

1. D. Clifton, D. Wong, L. Clifton, S. Wilson, R. Way, R. Pullinger, and L. Tarassenko. A Large-Scale Clinical Validation of an Integrated Monitoring System in the Emergency Department. *IEEE Journal of Biomedical and Health Informatics*, 17(4):835–842, July 2013.
2. I. Cleland, M. Han, C. Nugent, H. Lee, S. Zhang, S. McClean, and S. Lee. Mobile based prompted labeling of large scale activity data. In *The 7th International Conference on Ubiquitous Computing and Ambient Intelligence (UCAmi)*, pages 9–17, Guancaste, Costa Rica, 2013. Springer International Publishing.
3. S. Zhang, S. McClean, B. Scotney, L. Galway, and C. Nugent. A framework for context-aware online physiological monitoring. In *IEEE International Symposium on Computer-Based Medical systems*, pages 1–6, Bristol, UK, 2011. IEEE.
4. J. Ledolter and R. Kardon. Detecting the Progression of Eye Disease: CUSUM Charts for Assessing the Visual Field and Retinal Nerve Fiber Layer Thickness. *Translational Vision Science & Technology*, 2(6):2, September 2013.
5. D.R. Prajapati and P.B. Mahapatra. A new X chart comparable to CUSUM and EWMA charts. *International Journal of Productivity and Quality Management*, 4(1):103–128, 2009.
6. A. Jain and Yuan-fang Wang. A New Framework for On-Line Change Detection (Unpublished), Accessed September 2014. Online, `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.62.5929`.
7. Alvin C. Rencher. *Methods of Multivariate Analysis*. John Wiley & Sons, New York, USA, 2nd edition, 2002.
8. C. E. Bonferroni. Il Calcolo delle Assicurazioni su Gruppi di Teste. In *Studii in Onore del Profesor S. O. Carboni Roma*, 1936.
9. Shimmer. Shimmer 2 Specification and User Manual, Accessed September 2014. Online, `http://www.shimmersensing.com/images/uploads/docs/Shimmer_User_Manual_rev2Rk.pdf`.
10. S. Zhang, L. Galway, S. McClean, B. Scotney, D. Finlay, and C. Nugent. Deriving Relationships between Physiological Change and Activities of Daily Living using Wearable Sensors. In *Sensor Systems and Software*, pages 235–250, Miami, Florida, USA, 2011. Springer Berlin Heidelberg.