entropy

MDPI

Article

Detecting Metachanges in Data Streams from the Viewpoint of the MDL Principle

Shintaro Fukushima * and Kenji Yamanishi

Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku 113-8656, Japan; yamanishi@mist.i.u-tokyo.ac.jp

* Correspondence: shintaro_fukushima@mist.i.u-tokyo.ac.jp

Received: 11 October 2019; Accepted: 16 November 2019; Published: 20 November 2019

Abstract: This paper addresses the issue of how we can detect changes of changes, which we call *metachanges*, in data streams. A metachange refers to a change in patterns of when and how changes occur, referred to as "metachanges along time" and "metachanges along state", respectively. Metachanges along time mean that the intervals between change points significantly vary, whereas metachanges along state mean that the magnitude of changes varies. It is practically important to detect metachanges because they may be early warning signals of important events. This paper introduces a novel notion of metachange statistics as a measure of the degree of a metachange. The key idea is to integrate metachanges along both time and state in terms of "code length" according to the minimum description length (MDL) principle. We develop an online metachange detection algorithm (MCD) based on the statistics to apply it to a data stream. With synthetic datasets, we demonstrated that MCD detects metachanges earlier and more accurately than existing methods. With real datasets, we demonstrated that MCD can lead to the discovery of important events that might be overlooked by conventional change detection methods.

Keywords: change detection; change of change; data stream; minimum description length principle; code length

1. Introduction

1.1. Purpose of This Paper

In this study, we are concerned with detecting changes in data streams. The goal of *change detection* is to detect the time points at which the nature of the data-generating mechanism significantly changes.

Thus far, many algorithms have been proposed to detect change points in data streams (e.g., [1–11]), and several studies addressed or have been related to the issue of changes of changes [12–18]. In this paper, we refer to the changes of changes as *metachanges*. A metachange refers to a change in the pattern of when or how changes occur. It is practically important to detect metachanges because they may be early warning signals of important events [12,13]. Metachanges have been treated from a viewpoint of *metachanges along time*. Metachanges along time indicate that the interval significantly varies between the change points. Such metachanges were called *burstiness* [12] and *volatility* [13] in previous studies. The detection of metachanges along time provides users with useful information from data streams. For example, in a machine in a manufacturing factory, a decrease in the interval between change points might be a sign of a serious failure.

There is also another type of metachange: *metachanges along state*. Here, "state" refers to the parameter value of the probability density function of a distribution. We consider a situation where change points $t_1, \ldots$ are detected for a data stream $y_1, y_2, \ldots$, and y_t is drawn from $p_y(y_t; \eta)$. Here, p_y is a probability density function of distributions, and η is the associated parameter. Note that η

is called *state* in this paper, and it varies before and after a change point. A metachange along state means a change of how significantly η varies before and after a change point. Metachanges along state might provide information such as changes of magnitude and velocity, which indicate an important change in the underlying data-generating mechanism. For example, in a machine in a manufacturing factory, a shift to an abrupt (sudden) change from a gradual (incremental) change [19], or its inverse shift, might be a sign of serious events.

A conceptual illustration of metachanges is shown in Figure 1, where the upper graph shows a data stream $y_1, \dots$ and change points $\{t_i\}_{i=1}^{8}$ on the horizontal axis. The lower left graph shows intervals between change points $\Delta t = t_i - t_{i-1}$ on the vertical axis. Metachanges along time occur at t_4, t_5, t_6, t_7: for example, $t_4 - t_3$ is different from $t_3 - t_2$ and $t_2 - t_1$. The lower right graph shows the states estimated piecewisely between the change points. Here, we assume y_t is drawn from the univariate normal distribution $p_y(y_t; \mu, \sigma)$, where μ is the mean and σ is the standard deviation. In this case, (μ, σ) is a state. In Figure 1, because there is no significant change in the magnitude of state change between t_1 and t_2, a metachange along state does not occur at t_2. However, there is a significant change in the magnitude of change of μ between t_2 and t_3: thus, a metachange along state occurs at t_3. Because the magnitudes of the changes of μ and σ are almost the same between t_3 and t_4, a metachange along state does not occur at t_4. Using the same procedure, we conclude that metachanges along state occur at t_3 and t_7 with respect to μ. Moreover, metachanges along state occur at t_6 and t_8 with respect to σ: the magnitude of the change of standard deviations around t_6 (t_8) is greater than those around t_5 (t_7). As a result, metachanges along state occur at t_3, t_6, t_7, and t_8. We can infer that metachanges along both time and state occur at t_6 and t_8, by combining the metachanges along time and state.

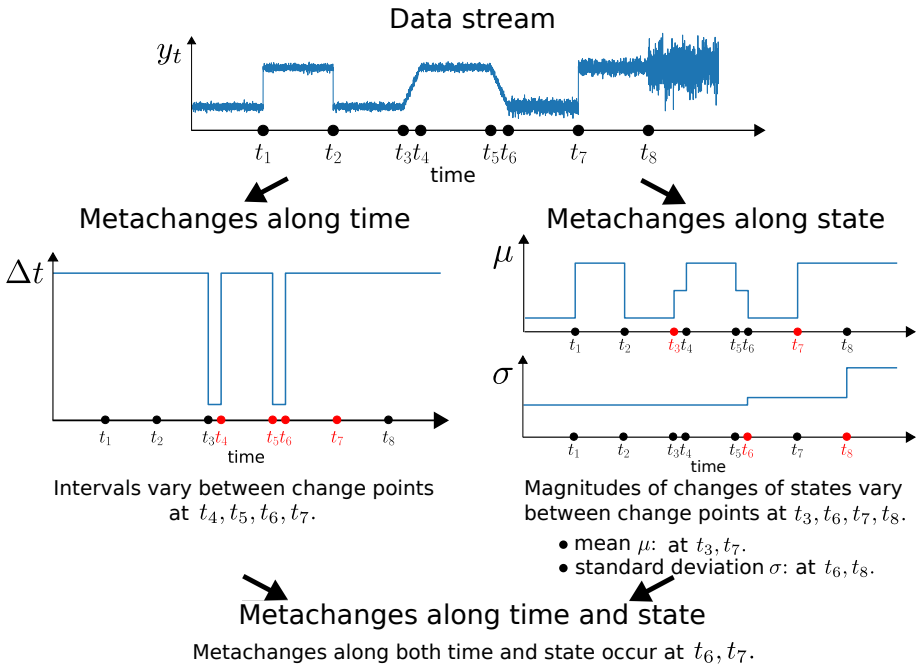

Figure 1. Conceptual illustration of metachanges.

Metachanges along time have been investigated in previous studies [12,13], and, although there have been several studies related to metachanges along state [14–18], the focus of these studies was not on metachanges along state in particular. The purpose of this paper is to propose a framework and an approach to detect metachanges along time and state from a unified view with the minimum description length (MDL) [20]. Therefore, our framework and approach not only include previous notions such as *burstiness* [12] and *volatility* [13] but also extend these notions to metachanges along state. MDL asserts that the best statistical decision strategy is the one that compresses the data best.

Description and coding with MDL are suitable for quantifying changes, and they enable us to easily integrate the code lengths of time and state.

1.2. Related Work

Change detection has been extensively explored in the area of data mining. Thus far, several methods have been proposed to detect metachanges in data streams [12,13], and there have been several studies related to metachanges along state [14–18].

Kleinberg [12] and Huang et al. [13] proposed algorithms for detecting metachanges along time. Kleinberg [12] proposed an algorithm to detect bursts in a time series. This algorithm assumes that intervals between successive events are drawn from an exponential distribution. The discretized values of the parameters of the exponential distribution are regarded as states. For intervals between successive events, states are estimated with dynamic programming. Changes of state indicate changes of intervals between the successive events. Huang et al. [13] proposed an algorithm, called the *volatility detector*, which detects changes of rates of change. The volatility detector prepares two buckets, called the *buffer* and the *reservoir*, to store intervals between change points. The intervals are put into the buffer sequentially. When the buffer is full, an interval is dropped from the buffer and moved to the reservoir in a first-in-first-out fashion. The reservoir stores the dropped interval by randomly replacing one of its stored intervals. If the ratio of variances of the buffer and the reservoir is over or under the specified threshold, the algorithm judges that the intervals change between change points. The authors called this event *volatility shift*. Both the burst detector and volatility detector are assumed to be used in two steps. That is, change points are detected with other change detection algorithms, and then changes of intervals between the change points are detected. While the burst detector works in an offline fashion, the volatility detector works in an online fashion.

Moreover, there have been several studies related to metachanges along state [14–18]. Aggarwal [15] introduced *velocity density estimation* to understand, visualize, and determine trends in the evolution of fast data streams. Spiliopoulou et al. [16,17] proposed an algorithm, called MONIC, to model and track cluster transitions. Ntoutsi et al. [18] proposed an algorithm, called FINGERPRINT, to summarize cluster evolution. Huang et al. [14] proposed a change type detector, intended to categorize change types into three relative types, some of which correspond to concept drifts proposed in [19]. Although their algorithms [14–18] are related to metachanges along state, they are not intended to characterize and detect metachanges directly. In addition, many change detection algorithms have been proposed based on detecting changes of state (e.g., [6–9,21,22]). The dynamic model selection [6,7] is the seminal work to apply MDL to the task of dynamic model selection and change detection. The MDL change statistics [8], SCAW [9], and STREAMKRIMP [22] are change detection algorithms with MDL. However, these algorithms are not intended to characterize and detect metachanges along state directly.

1.3. Significance of This Paper

In the context of Sections 1.1 and 1.2, the contributions of this paper are summarized in the following subsections.

1.3.1. Proposal of Concept of Metachange

To detect changes of changes in data streams, we define a concept of *metachanges* along both time and state. Previous studies [12,13] considered metachanges along time only. In this paper, we deal with metachanges along both time and state. Metachanges along time include the notions proposed in previous studies such as burstiness [12] and volatility [13]. Metachange along state could capture changes of changes of the parameters of distribution between change points.

Our concept of *metachange* can detect the potential change of changes in data streams, which was overlooked by previous studies.

1.3.2. Novel Algorithm for Detection of Metachanges

We define *metachange statistics* along both time and state. There is a challenge to combining the metachange statistics along time and those along state. In this paper, these statistics are defined based on the MDL principle. Metachange statistics along time (MCAT) is defined as the code length of an interval between the change points, whereas metachange statistics along state (MCAS) is defined as the difference between the predictive code length and the normalized maximum likelihood (NML) code length [23] after a change. It is possible to simply add these statistics because they are defined as code lengths, which enables us to detect metachanges along both time and state in a unified manner.

2. Theoretical Background of Metachange Statistics

In this section, we consider how to encode both intervals between change points and states around the change points. We assume that for a data stream $y_1, y_2, \ldots$ change points t_1, ... are detected and that the intervals between change points $x_i = t_i - t_{i-1}$ and y_t are drawn, respectively, from

$$x_i \sim p_x(x_i; \xi),\ y_t \sim p_y(y_t; \eta),$$

where p_x and p_y are probability density functions of distributions and ξ and η are the associated parameters. Finally, η is the *state* whose metachanges are addressed in this paper.

2.1. Definitions of Metachanges

In this subsection, we give definitions of metachanges.

Definition 1. *(Metachange along time) For intervals between change points $x_1, x_2, \ldots$, we say that a metachange along time occurs at a change point t_i for a threshold parameter $\delta_\mathrm{t} > 0$ if and only if*

$$\begin{aligned} &q_1 \to q_2 \text{ at } t = t_i, \\ &d(q_1, q_2) > \delta_\mathrm{t},\ q_1, q_2 \in \mathcal{F}_\mathrm{t},\ \mathcal{F}_\mathrm{t} = \{p_x(x; \xi)\}, \end{aligned} \tag{1}$$

where q_1 and q_2 are distributions of intervals. $q_1 \to q_2$ means that $x_t \sim q_1$ at $t = t_{i-1}$ and $x_t \sim q_2$ at $t = t_i$. d is a distance function between the probability density functions.

Definition 2. *(Metachange along state) For a data stream $y_1, y_2, \ldots$, we say that a* metachange along state *occurs at a change point t_i for a threshold parameter $\delta_\mathrm{s} > 0$ if and only if*

$$\begin{aligned} &q_1 \to q_2 \text{ at } t = t_{i-1},\ q_2 \to q_3 \text{ at } t = t_i, \\ &|d(q_2, q_3) - d(q_1, q_2)| > \delta_\mathrm{s},\ q_1, q_2, q_3 \in \mathcal{F}_\mathrm{s},\ \mathcal{F}_\mathrm{s} = \{p_y(y; \eta)\}, \end{aligned} \tag{2}$$

where q_1, q_2, and q_3 are distributions of values of the data stream. Equation (2) means that $y_t \sim q_1$ at $t = t_{i-2}, \ldots, t_{i-1} - 1$, $y_t \sim q_2$ at $t = t_{i-1}, \ldots, t_i - 1$, and $y_t \sim q_3$ at $t = t_i \ldots, t_{i+1} - 1$. Here, d is the same as that in Definition 1.

Definition 3. *(Integrated metachange) For a change point t_i, we say that an* integrated metachange *occurs at t_i if and only if Equation (1) or Equation (2) holds.*

2.2. Problem Setting

In this subsection, we consider a situation where $(m+1)$ change points $t_1, \ldots, t_{m+1}$ are given. We consider how to encode x_i and y_t as shortly as possible. The ideal code length required for encoding

x_i is given by what we call the predictive code length, which is the sum of the negative logarithm of its predictive density p_x at each time point, defined as follows:

$$\min_{\{\hat{\xi}_{x^{i-1}}\}_{i=1}^{m}} \sum_{i=1}^{m} -\log p_x(x_i; \hat{\xi}_{x^{i-1}}), \tag{3}$$

where $\hat{\xi}_{x^{i-1}}$ are estimated at each change point. Similarly, the ideal code length required for encoding y_t around change points is given by the predictive code length as follows:

$$\min_{\{\hat{\eta}_{y^{t-1}} | t \in \text{neighbor}(t_i)\}_{i=1}^{m}} \sum_{i=1}^{m} \sum_{t \in \text{Neighbor}(t_i)} -\log p_y(y_t; \hat{\eta}_{y^{t-1}}), \tag{4}$$

where $\text{Neighbor}(t_i)$ indicates the neighborhood of a change point t_i. In practice, as explained in Section 3.3, $\text{Neighbor}(t_i) = [t_i - h, t_i + h], h \in \mathbb{N}$. $\hat{\xi}_{x^{i-1}}$ and $\hat{\eta}_{y^{t-1}}$ are estimated using $x^{i-1} = x_1 \dots x_{i-1}$ and $y^{t-1} = y_1 \dots y_{t-1}$, respectively. A change of $\hat{\eta}_{y^{t-1}}$ indicates a change of state. Detection of a metachange along time is asserted as a problem of detection of a change of $\hat{\xi}_{x^{i-1}}$ in Equation (3). On the other hand, detection of a metachange along state is asserted as a problem of detection of a change of how $\hat{\eta}_{y^{t-1}}$ in Equation (4) changes around a change point between change points.

3. Metachange Detection Algorithm

In this section, we present our online algorithm called metachange detection algorithm (MCD) for detecting metachanges along both time and state. We consider how to achieve Equations (3) and (4) in an online fashion. A schematic description of MCD is shown in Figure 2.

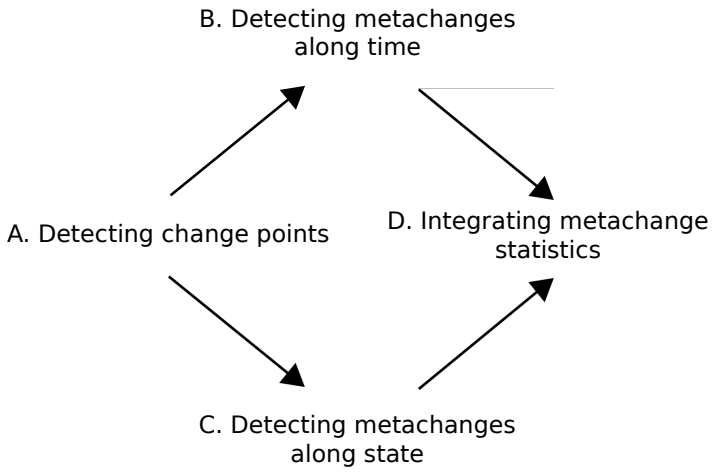

Figure 2. Schematic of the proposed metachange detection algorithm (MCD) algorithm.

First, we detect change points from data stream (A). Next, we concurrently detect metachanges along time (B) and along state (C). We introduce metachange statistics to quantify these metachanges. Finally, we integrate the metachange statistics along time and state into a statistics (D).

The key challenge of detecting metachanges along time and state is how to describe and integrate them. Our approach describes both metachanges as code lengths with MDL; therefore, it is easy to combine them.

3.1. Detecting Change Points

First, we detect change points $t_1, t_2, \dots$. As our proposed algorithm MCD works in an online fashion, it is necessary for the change detection algorithm to work in an online fashion (e.g., [1–4,8,9]).

In general, MCD is prone to errors by the change detection algorithm and its threshold parameter. We empirically investigate and discuss this point in detail in Section 4.

3.2. Detecting Metachanges along Time

For the detected change points $t_1 \ldots$, let us consider intervals between the successive change points $I_i = [t_{i-1}, t_i - 1]$, with length $x_i = t_i - t_{i-1}$. For an interval sequence $x^i = x_1 \ldots x_i$, we consider how to achieve Equation (3) in an online fashion. We define metachange along time (MCAT) a_{t_i} as the predictive code length

$$a_{t_i} \stackrel{\text{def}}{=} -\log p_x(x_i; \hat{\xi}_{x^{i-1}}), \tag{5}$$

where $p_x \in \mathcal{F}_\mathrm{t}$, $\mathcal{F}_\mathrm{t} = \{p_x(x;\xi)\}$ is a parametric class of probability distribution, and $\hat{\xi}_{x^{i-1}}$ is estimated using $x^{i-1} = x_1 \ldots x_{i-1}$. For example, we can estimate $\hat{\xi}_{x^{i-1}}$ as the maximum likelihood estimator. To deal with nonstationary data streams, we use the online discounting maximum likelihood estimator [24]

$$\hat{\xi}_{x^{i-1}} = \operatorname*{argmax}_{\xi} \sum_{t=1}^{i-1} r(1-r)^{i-1-t} \log p_x(x_t;\xi), \tag{6}$$

where $0 < r < 1$ is a discounting parameter. An increase in r has a greater effect on forgetting past data.

In this paper, we introduce a parametric class of the exponential distribution

$$\mathcal{F}_\mathrm{t} = \{p_x(x;\xi) = \xi \exp(-\xi x),\ \xi > 0\}. \tag{7}$$

By substituting Equation (7) into Equation (6), we get

$$\begin{aligned}\hat{\xi}_{x^{i-1}} &= \operatorname*{argmax}_{\xi} \sum_{t=1}^{i-1} r(1-r)^{i-1-t} \log(\xi \exp(-\xi x_t)) \\ &= \operatorname*{argmax}_{\xi} \sum_{t=1}^{i-1} r(1-r)^{i-1-t} (\log \xi - \xi x_t).\end{aligned} \tag{8}$$

The inside of argmax in the right-hand side of Equation (8) is expanded as

$$\begin{aligned}\sum_{t=1}^{i-1} r(1-r)^{i-1-t}(\log\xi - \xi x_t) &= r\log\xi \sum_{t=1}^{i-1}(1-r)^{i-1-t} - r\xi \sum_{t=1}^{i-1}(1-r)^{i-1-t}x_t \\ &= r\log\xi \frac{1-(1-r)^{i-1}}{r} - r\xi \sum_{t=1}^{i-1}(1-r)^{i-1-t}x_t \\ &= \log\xi(1-(1-r)^{i-1}) - r\xi \sum_{t=1}^{i-1}(1-r)^{i-1-t}x_t.\end{aligned} \tag{9}$$

The right-hand side of Equation (9) is maximized by deriving it with respect to ξ. As a result, we obtain the following optimal solution:

$$\hat{\xi}_{x^{i-1}} = \frac{1-(1-r)^{i-1}}{r\sum_{t=1}^{i-1}(1-r)^{i-1-t}x_t}. \tag{10}$$

Thus, by substituting Equation (10) into Equation (5), MCAT at t_i is

$$a_{t_i} = -\log p_x(x_i; \hat{\xi}_{x^{i-1}}) = -\log \hat{\xi}_{x^{i-1}} + \hat{\xi}_{x^{i-1}} x_i. \tag{11}$$

In practice, we judge that a metachange occurs along time when MCAT changes greatly between the change points. Technically, we use the change rate of MCAT: a metachange occurs along time when

$|(a_{t_i} - a_{t_{i-1}})/a_{t_{i-1}}| > \epsilon_t$ holds, where $\epsilon_t > 0$ is a threshold parameter. We call the algorithm described above as the *metachange detection along time algorithm* (MCD-T).

As for computational cost of MCAT, Equation (10) is written as

$$\hat{\xi}_{x^{i-1}} = \frac{1-(1-r)^{i-1}}{r\, s_{i-1}},$$

where

$$s_{i-1} \stackrel{\text{def}}{=} \sum_{j=1}^{i-1} (1-r)^{i-1-j} x_j.$$

s_i and s_{i-1} satisfy the following relation:

$$s_i = (1-r)s_{i-1} + x_i.$$

Therefore, the computational cost of MCAT a_{t_i} is $O(i)$.

Example:

We consider a data stream with a length of 200 time intervals between change points: $x_i = 100$ $(i = 1, \ldots, 100)$ and $x_i = 500$ $(i = 101, \ldots, 200)$. This means that there are 201 change points $\{t_i\}_{i=1}^{201}$. If we assume $t_1 = 100$, then $t_2 = 200, \ldots, t_{101} = 10{,}100$, $t_{102} = 10{,}600$, $t_{103} = 11{,}100$, $\ldots$, $t_{201} = 60{,}100$. Then, x_i is calculated as $x_i = t_i - t_{i-1}$. Figure 3 shows the time intervals at the change points (Figure 3, top), MCATs a_{t_i} (Figure 3, second graph), the change rate of MCATs $|(a_{t_i} - a_{t_{i-1}})/a_{t_{i-1}}|$ (Figure 3, third graph), and $\hat{\xi}_{x^{i-1}}$ (Figure 3, bottom). We observe in Figure 3 that we can detect the metachange along time when we choose a suitable threshold ϵ_t. Here, the discounting parameter is set to $r = 0.5$.

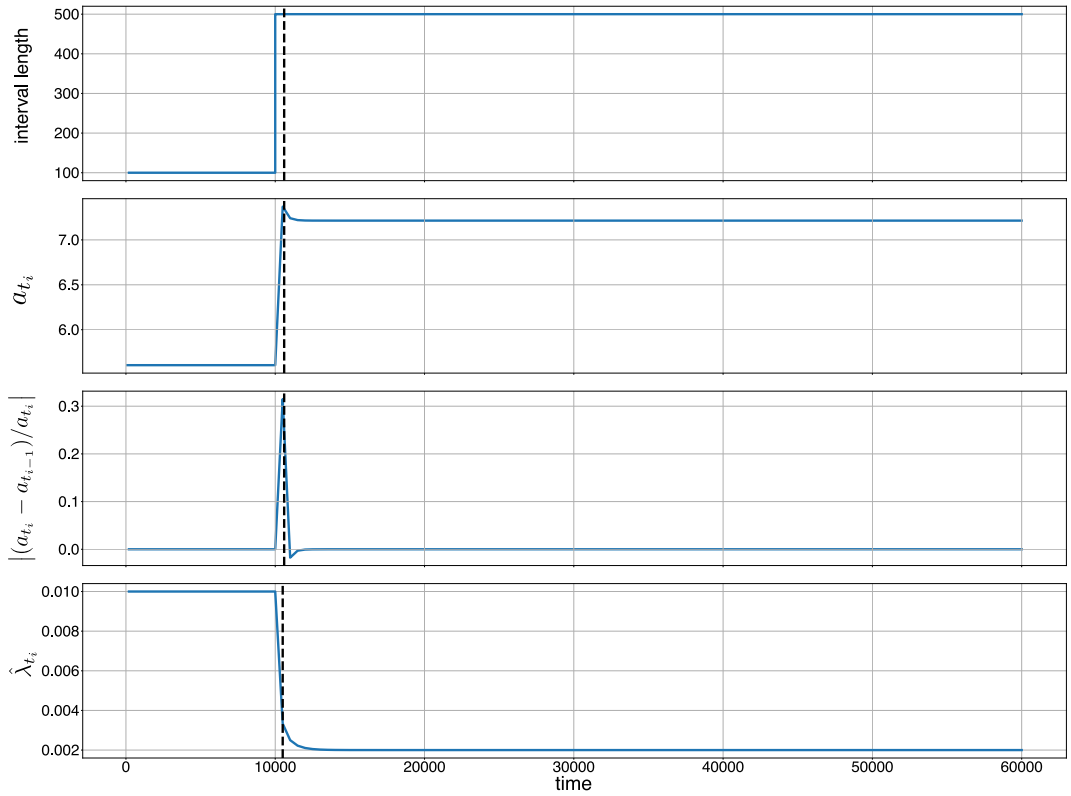

Figure 3. Metachange statistics along time (MCAT): (**top**) time interval at each change point; (**second**) MCAT a_{t_i}; (**third**) change rate of MCAT $|(a_{t_i} - a_{t_{i-1}})/a_{t_{i-1}}|$; and (**bottom**) the estimated parameter of the exponential distribution $\hat{\lambda}$. The discounting parameter $r = 0.5$.

3.3. Detecting Metachanges Along State

For a change point t_i detected in Section 3.1, we consider how to achieve Equation (4) in an online fashion. We consider a subset of time around t_i for Neighbor(t_i) in Equation (4). The subset is denoted by $J_i = [t_i - h, t_i + h]$, where $h \in \mathbb{N}$ is a window size. Thus, we consider a sequence $y_{t_i-h}^{t_i+h} = y_{t_i-h} \dots y_{t_i+h}$, with length $n = 2h + 1$. We introduce a parametric class of probability distributions $\mathcal{F}_{\mathrm{s}} = \{p_y(Y;\eta); \eta \in H\}$. Here, Y is a random variable and η is a real-valued parameter. H is the associated parameter space.

Next, we define metachange statistics along state (MCAS) at change point t_i. First, two statistics, $b_{t_i}^+$ and $b_{t_i}^-$, are introduced. These are defined as the difference between two code lengths for $y_{t_i+1}^{t_i+h}$: one is the "expected" code length, estimated using the parameter change at t_{i-1} and the estimated parameter with $y_{t_i-h}^{t_i-1}$. The other is the code length with the parameter estimated in terms of $y_{t_i+1}^{t_i+h}$. Formally, $b_{t_i}^{\pm}$ is defined as the difference between the predictive code length and the NML code length [20] after the change point. The former is calculated as the predictive code length, which is the total code length for encoding $y_{t_i+1}^{t_i+h}$ in a predictive way, using the estimated parameter $\eta^{\pm}$ as follows:

$$\frac{1}{h} \sum_{t=t_i+1}^{t_i+h} -\log p_y(y_t; \hat{\eta}^{\pm}), \tag{12}$$

where $\hat{\eta}^{\pm}$ is defined as

$$\hat{\eta}^{\pm} \stackrel{\text{def}}{=} \hat{\eta}_{y_{t_i-h}^{t_i-1}} \pm \left(\hat{\eta}_{y_{t_{i-1}+1}^{t_{i-1}+h}} - \hat{\eta}_{y_{t_{i-1}-h}^{t_{i-1}-1}} \right), \tag{13}$$

which indicates the parameter change to *the same side* and *the opposite side* in the same way as the previous change point t_{i-1}. Here, $\hat{\eta}_{y_{\tau_1}^{\tau_2}}$ means the maximum likelihood estimator of η using $y_{\tau_1}^{\tau_2} = y_{\tau_1} \dots y_{\tau_2}$.

The latter is calculated as the NML code length, which is defined as the negative logarithm of the NML distribution [20]:

$$\frac{1}{h} \left(\sum_{t=t_i+1}^{t_i+h} -\log p_y(y_t; \hat{\eta}_{y_{t_i+1}^{t_i+h}}) + \log C_h \right). \tag{14}$$

The difference between Equation (12) and Equation (14) is given by

$$b_{t_i}^{\pm} \stackrel{\text{def}}{=} \frac{1}{h} \left\{ \sum_{t=t_i+1}^{t_i+h} \left(-\log p_y(y_t; \hat{\eta}^{\pm}) + \log p_y(y_t; \hat{\eta}_{y_{t_i+1}^{t_i+h}}) \right) - \log C_h \right\}, \tag{15}$$

where $C_h = \sum_{z_{t_i+1}^{t_i+h}} \max_{\eta} p_y(z_{t_i+1}^{t_i+h}; \eta)$ in Equation (15) is computed using Rissanen's approximation formula under some regularity conditions [23]:

$$\log C_h \approx \frac{k}{2} \log \frac{h}{2\pi} + \log \int \sqrt{|I(\theta)|}\, \mathrm{d}\theta,$$

where k is the dimension of H and $I(\theta) \stackrel{\text{def}}{=} \mathbb{E}_{\eta}[-\partial^2 \log p_y(Y;\eta)/\partial\eta_i \partial\eta_j]$ is the Fisher information matrix at the parameter value η. Intuitively, Equation (15) quantifies the redundant code length for coding $y_{t_i+1}^{t_i+h}$ with the parameters estimated in terms of the parameter change at t_{i-1} and the parameter values in the former part of t_i.

Finally, we define MCAS as

$$b_{t_i} \stackrel{\text{def}}{=} \min(b_{t_i}^+, b_{t_i}^-), \tag{16}$$

which means that metachanges along state are quantified by the relative magnitude of changes in the parameters in this paper. The computational cost of MCAS is $O(h) = O(1)$. We judge that a

metachange along state occurs at t_i when $b_{t_i} > \epsilon_s$ holds, where $\epsilon_s > 0$ is a threshold parameter. We call the algorithm described above as the *metachange detection along state algorithm* (MCD-S).

Example:

We generate a data stream with length 11,250:

$$y_t \sim \begin{cases} \mathcal{N}(0.0,\, 0.05) & (t = 1, \ldots, 1000) \\ \mathcal{N}(1.0,\, 0.05) & (t = 1001, \ldots, 2000) \\ \mathcal{N}(0.0,\, 0.05) & (t = 2001, \ldots, 3000) \\ \mathcal{N}(1.0,\, 0.05) & (t = 3001, \ldots, 4000) \\ \mathcal{N}((t-4001)/1000,\, 0.05) & (t = 4001, \ldots, 5000) \\ \mathcal{N}(0.0,\, 0.05) & (t = 5001, \ldots, 6000) \\ \mathcal{N}((t-6000)/1000,\, 0.05) & (t = 6001, \ldots, 7000) \\ \mathcal{N}(1.0,\, 0.05) & (t = 7001, \ldots, 8000) \\ \mathcal{N}(1-(t-8000)/250,\, 0.05) & (t = 8001, \ldots, 8250) \\ \mathcal{N}(0.0,\, 0.1) & (t = 8251, \ldots, 9250) \\ \mathcal{N}(1.0,\, 0.1) & (t = 9251, \ldots, 10,250) \\ \mathcal{N}(1.0,\, 0.3) & (t = 10,251, \ldots, 11,250) \end{cases},$$

where $\mathcal{N}(\mu,\, \sigma)$ denotes the probability density function of the univariate normal distribution with mean μ and standard deviation σ.

Figure 4 shows data stream $\{y_t\}$ (Figure 4, top) and statistics $\{b_{t_i}\}$ (Figure 4, bottom). The parameter is set to $h = 200$. True change points occur at 1001, 2001, 3001, 4001, 5101, 6001, 7001, 8001, 8251, 9251, and 10,251. Figure 4 shows that the statistics b_{t_i} increase when there is a change in how parameters behave around a change point between successive change points. At $t_2 = 2001$ and $t_3 = 3001$, b_{t_i} are relatively small, which shows that parameter changes (i.e., their magnitudes) do not differ much between $t_1 = 1001$ and $t_2 = 2001$ and between $t_2 = 2001$ and $t_3 = 3001$. However, b_{t_i} increases at $t_4 = 4001$ because the change shifts to a gradual change from an abrupt one. These results indicate that MCAS provides information regarding changes in the behavior around the change points.

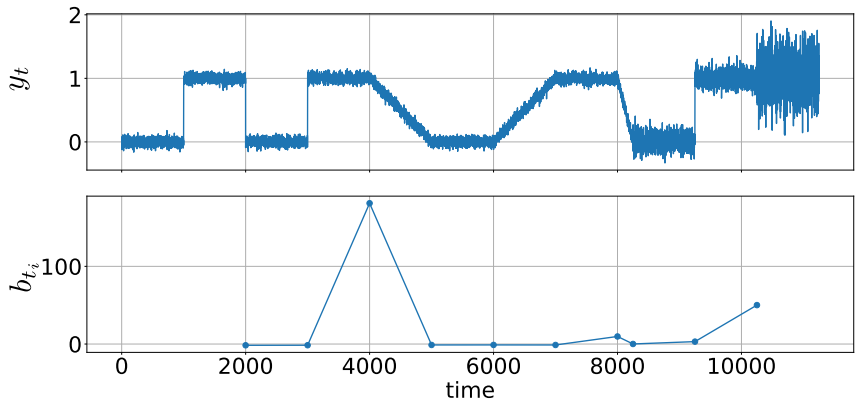

Figure 4. Metachange statistics along state (MCAS): (**top**) data stream y_t; and (**bottom**) MCAS b_{t_i}. Window size $h = 200$.

3.4. Integrating Metachange Statistics

Finally, we consider how to integrate MCAT a_{t_i} and MCAS b_{t_i} at a change point t_i. Because a_{t_i} and b_{t_i} are code lengths, they can be summed. Therefore, we propose adding a_{t_i} and b_{t_i} with weighting. Integrated metachange (MCI) s_{t_i} at t_i is defined as

$$s_{t_i} \stackrel{\text{def}}{=} a_{t_i} + \lambda b_{t_i}, \tag{17}$$

where $\lambda \in \mathbb{R}$ is a hyperparameter. We should carefully choose λ with data. In Section 4.3, λ is determined using a grid search.

In practice, we judge that a metachange along both time and state occur at t_i when MCI greatly changes between the change points. As in the case of metachanges along time in Section 3.2, we use the change rate of MCI: a metachange along both time and state occurs at t_i if $|(s_{t_i} - s_{t_{i-1}})/s_{t_{i-1}}| > \epsilon_{\mathrm{ts}}$, where $\epsilon_{\mathrm{ts}} > 0$ is a threshold parameter.

We call the overall algorithm described above *MCD*; it is summarized in Algorithm 1.

Algorithm 1 MCD.

Input: r: discounting parameter ($0 < r < 1$), h: window size, ϵ_{ts}: threshold parameter
Output: a_{t_i}: metachange statistics along time, b_{t_i}: metachange statistics along state, s_{t_i}: integrated metachange statistics.

$i = 1$
for $t = 1, \ldots$ **do**
  Input y_t.
  Detect change point with a change detection algorithm.
  if t is change point **then**
    $t_i \leftarrow t$.
    $x_i \leftarrow t_i - t_{i-1}$.
    Calculate metachange statistics along time a_{t_i} according to Equation (11).
    Calculate metachange statistics along state b_{t_i} according to Equation (16).
    Calculate integrated metachange statistics s_{t_i} according to Equation (17).
    Raise an alarm if and only if $|(s_{t_i} - s_{t_{i-1}})/s_{t_{i-1}}| > \epsilon_{\mathrm{ts}}$.
    $i \leftarrow i + 1$.
  end if
end for

4. Experiment

We conducted five experiments to confirm the effectiveness of the proposed algorithm MCD (https://github.com/s-fuku/metachange).

4.1. Synthetic Dataset 1 (Metachanges along Time)

We defined six levels of time intervals between change points referring to the work in [13,25]. The interval lengths were 100,000, 50,000, 10,000, 5000, 1000, and 500. The change points were set using a Bernoulli distribution oscillating between $\mu = 0.2$ and $\mu = 0.8$. For each combination of two intervals, we generated the streams based on the scheme above. Each stream contained 100 change points. In what follows, L_1 and L_2 indicate the first and second interval lengths, respectively.

We confirmed the effectiveness of MCD by comparing it with a volatility detector (VD) [13]. We used the SEED algorithm [13] and the sequential MDL-change statistics algorithm (SMDL) [8] for change detection. SEED was based on ADWIN2 [21] and its parameters were set to $\delta = 0.05$, $\Gamma = 75$, $\hat{\epsilon} = 0.025$, and $\alpha = 0.025$, which are the same as those in [13]. The window size w of SMDL was set to $w = 0.2L_1$, and the threshold parameter ϵ was set to $\epsilon = 0.01$. For the Bernoulli distribution, the change score Ψ_t of SMDL at time t was calculated as

$$\Psi_t = -\hat{\mu}_0 \log \hat{\mu}_0 - (1 - \hat{\mu}_0) \log (1 - \hat{\mu}_0) - \frac{1}{2} \left(-\hat{\mu}_1 \log \hat{\mu}_1 - (1 - \hat{\mu}_1) \log (1 - \hat{\mu}_1)\right) - \frac{1}{2} \left(-\hat{\mu}_2 \log \hat{\mu}_2 - (1 - \hat{\mu}_2) \log (1 - \hat{\mu}_2)\right),$$

where $\hat{\mu}_0 = \sum_{i=t-w}^{t+w} y_i/(2w+1)$, $\hat{\mu}_1 = \sum_{i=t-w}^{t-1} y_i/w$, and $\hat{\mu}_2 = \sum_{i=t}^{t+w} y_i/(w+1)$. If $\Psi_t > \epsilon$, t is regarded as a change point. We determined that t was a change point if the change score Ψ_t was the maximum. The parameter of MCD-T was set to $r = 0.2$. Below, we discuss the dependency of MCD-T on r in Figure 5. For VD, buffer size $B = 32$ and reservoir size $R = 32$, which were the same as in [13]. We also

discuss the dependency of VD on B and R below in Figure 6. In running SEED [13], we used the Java source code provided by the authors (https://www.cs.auckland.ac.nz/research/groups/kmg/DavidHuang.html). We started to use change points when its number reached $B + R$ for MCD-T and VD because the buffer and the reservoir of VD are not full until $B + R$ intervals arrive.

We investigated the trade-off between detection delay and accuracy in terms of benefit and false alarm rate, defined as in [8,26]. For MCD-T, we first fixed the threshold parameter ϵ_{t} and converted MCAT $\{a_{t_i}\}$ in Equation (11) to binary alarms $\{\alpha_{t_i}\}$. That is, $\alpha_{t_i} = \mathbb{1}\left(|(a_{t_i} - a_{t_{i-1}})/a_{t_{i-1}}| > \epsilon_{\mathrm{t}}\right)$, where $\mathbb{1}(t)$ denotes the binary function that takes 1 if and only if t is true. We evaluated MCD-T by varying ϵ_{t}. We let τ be a maximum tolerant delay of metachange detection. When the metachange really started from t^*, we defined the *benefit* of an alarm at time t as

$$b(t; t^*) = \begin{cases} 1 - \frac{|t-t^*|}{\tau} & (0 \leq |t - t^*| < \tau), \\ 0 & (\text{otherwise}). \end{cases} \tag{18}$$

The number of *false alarms* was calculated as

$$n(\alpha_1^m) \stackrel{\text{def}}{=} \sum_{k=1}^{m} \alpha_{t_k} \mathbb{1}(b(t_k, t^*) = 0). \tag{19}$$

We visualized the performance by plotting the recall rate of the total benefit, b, against the false alarm rate, $n/\sup_{\epsilon_{\mathrm{t}}} n$, with ϵ_{t} varying. Likewise, for VD, α_{t_i} was calculated using the *relative volatility* between the variances of the buffer and the reservoir by varying the threshold parameter β. We evaluated all four combinations of change detectors SEED and SMDL and metachange detectors MCD-T and VD by calculating the average and standard deviation of the area under the curve (AUC) of the benefit vs. FAR curves. The AUC scores were calculated over 50 sequences. The delay parameter was set to $\tau = 5L_2$. Table 1 shows the average AUC scores. Table 1 shows that MCD-T with SEED or MCD-T with SMDL outperforms VD with SEED or VD with SMDL. This indicates the effectiveness of MCD-T.

Because MCD-T depends on discounting parameter r and the change detection algorithm used, we investigated these effects. First, we examined the dependency of AUC on r for all combinations of L_1 and L_2. We calculated AUC for 30 times with $r = 0.01, 0.05, 0.1, 0.2, 0.3, 0.4$, and 0.5. We used SEED [13] as the change detection algorithm, and its parameters were set to the same values as above. The dataset used was also the same as in the previous experiment. Figure 5 shows that, when L_1 is relatively small (e.g., $L_1 = 500, 1000, 5000, 10,000$), AUC is not heavily dependent on r. When L_1 is larger, however, we observe that the larger r is, the smaller AUC is. This is because, with an increase of L_1, the number of false alarms of SEED also increases. In such situations, MCD-T is more prone to the false alarms when r is larger.

Figure 6 shows the dependency of AUC of VD on the buffer size B and the reservoir size R ($B = R$) for comparison. We calculated AUC for 50 times. We observe from Figure 6 that AUC decreases as B increases. In addition, we also see that MCD-T outperforms VD for various combinations of r and $B(= R)$ by comparing Figure 5 with Figure 6.

Next, we investigated the effect of the change detection algorithm used. We used SEED by changing the parameter $\hat{\epsilon} = 0.0025, 0.005$, and 0.0075. Other conditions and the dataset were the same as in the previous experiment. Here, $\hat{\epsilon}$ is a hyperparameter that controls the threshold parameter [13]. Figure 7 shows that AUC does not heavily depend on $\hat{\epsilon}$ for all combinations of L_1 and L_2. In general, the threshold parameter of the change detection algorithm controls the performance of MCD-T. Hence, it should be carefully set.

Table 1. Average area under the curve (AUC) scores on Synthetic Dataset 1 ($r = 0.2, \tau = 5L_2$). Boldfaces describe best performances.

L_1	L_2	SEED		SMDL	
		MCD-T	VD	MCD-T	VD
100,000	50,000	**0.603 ± 0.180**	0.458 ± 0.199	0.500 ± 0.060	0.313 ± 0.160
100,000	10,000	0.621 ± 0.147	0.310 ± 0.167	**0.710 ± 0.254**	0.463 ± 0.113
100,000	5000	0.645 ± 0.129	0.328 ± 0.152	**0.668 ± 0.223**	0.416 ± 0.164
100,000	1000	**0.651 ± 0.110**	0.275 ± 0.135	0.512 ± 0.123	0.448 ± 0.107
100,000	500	**0.697 ± 0.140**	0.336 ± 0.140	0.660 ± 0.111	0.506 ± 0.138
50,000	100,000	**0.788 ± 0.093**	0.647 ± 0.107	0.729 ± 0.067	0.639 ± 0.107
50,000	10,000	**0.671 ± 0.103**	0.280 ± 0.130	0.605 ± 0.171	0.556 ± 0.060
50,000	5000	**0.708 ± 0.087**	0.293 ± 0.144	0.617 ± 0.183	0.546 ± 0.146
50,000	1000	**0.718 ± 0.067**	0.294 ± 0.140	0.655 ± 0.161	0.501 ± 0.144
50,000	500	**0.767 ± 0.110**	0.316 ± 0.133	0.686 ± 0.074	0.470 ± 0.157
10,000	100,000	0.863 ± 0.059	0.794 ± 0.058	**0.877 ± 0.068**	0.791 ± 0.015
10,000	50,000	0.834 ± 0.050	0.735 ± 0.050	**0.876 ± 0.066**	0.823 ± 0.026
10,000	5000	**0.723 ± 0.040**	0.344 ± 0.250	0.658 ± 0.159	0.498 ± 0.084
10,000	1000	**0.781 ± 0.014**	0.375 ± 0.260	0.689 ± 0.083	0.444 ± 0.077
10,000	500	**0.809 ± 0.063**	0.391 ± 0.256	0.671 ± 0.163	0.520 ± 0.070
5000	100,000	**0.856 ± 0.060**	0.796 ± 0.067	0.854 ± 0.071	0.798 ± 0.036
5000	50,000	0.825 ± 0.047	0.726 ± 0.062	**0.875 ± 0.032**	0.708 ± 0.043
5000	10,000	**0.777 ± 0.030**	0.575 ± 0.139	0.716 ± 0.060	0.630 ± 0.031
5000	1000	**0.783 ± 0.009**	0.436 ± 0.257	0.709 ± 0.009	0.353 ± 0.098
5000	500	0.816 ± 0.054	0.493 ± 0.269	**0.839 ± 0.191**	0.413 ± 0.097
1000	100,000	**0.872 ± 0.072**	0.814 ± 0.072	0.836 ± 0.036	0.812 ± 0.036
1000	50,000	0.844 ± 0.059	0.754 ± 0.061	**0.947 ± 0.037**	0.810 ± 0.027
1000	10,000	0.802 ± 0.022	0.668 ± 0.050	**0.873 ± 0.049**	0.805 ± 0.023
1000	5000	0.801 ± 0.014	0.648 ± 0.064	**0.895 ± 0.053**	0.812 ± 0.053
1000	500	**0.816 ± 0.048**	0.560 ± 0.242	0.711 ± 0.141	0.409 ± 0.108
500	100,000	**0.876 ± 0.068**	0.831 ± 0.063	0.830 ± 0.079	0.820 ± 0.023
500	50,000	**0.845 ± 0.062**	0.767 ± 0.062	0.836 ± 0.044	0.818 ± 0.010
500	10,000	0.827 ± 0.051	0.676 ± 0.047	**0.872 ± 0.023**	0.822 ± 0.016
500	5000	0.830 ± 0.047	0.663 ± 0.042	**0.864 ± 0.047**	0.819 ± 0.017
500	1000	0.830 ± 0.050	0.612 ± 0.100	**0.935 ± 0.022**	0.853 ± 0.095

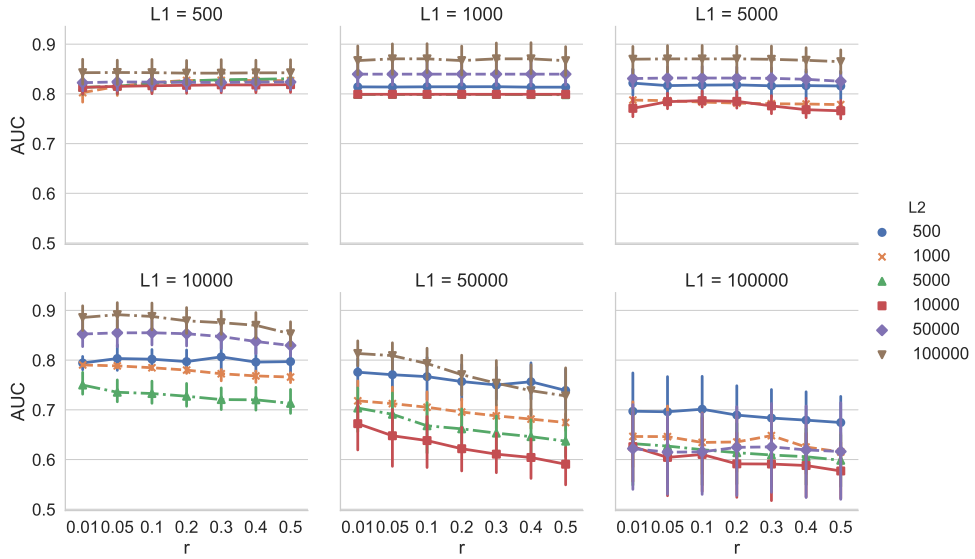

Figure 5. Dependency of AUC on discounting parameter r for MCD-T on Synthetic Dataset 1.

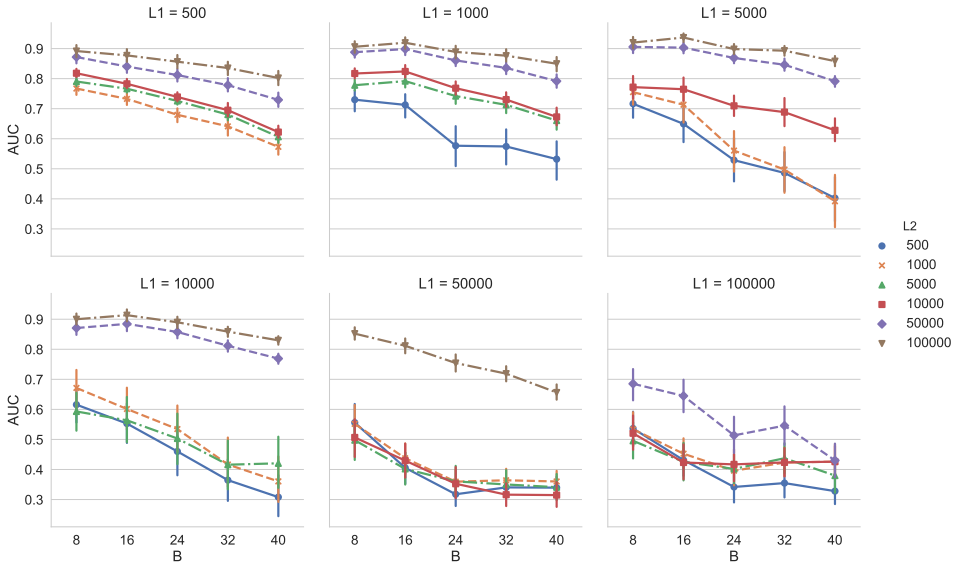

Figure 6. Dependency of AUC on the buffer size B (= the reservoir size R) for VD on Synthetic Dataset 1.

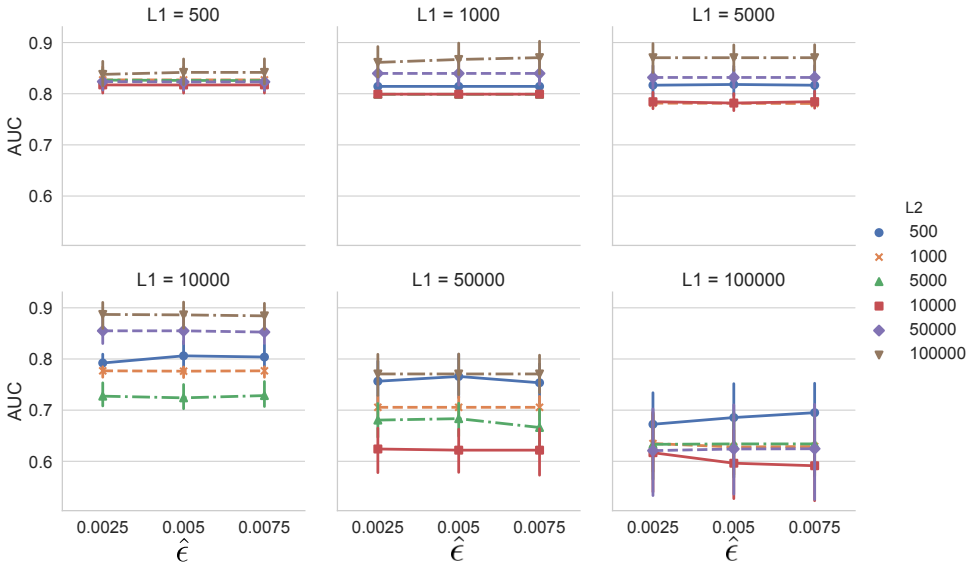

Figure 7. Dependency of AUC on threshold controlling parameter $\hat{e}$ of SEED [13] on Synthetic Dataset 1.

4.2. *Synthetic Dataset 2 (Metachanges along State)*

We generated a data stream with length $24L$, where $L = 500, 1000, 2000$. The generated data stream contained a metachange along state. In the former part, each datum was drawn from

$$y_t \sim \begin{cases} \mathcal{N}(0, 0.1) & (t = 1, \ldots, L), \\ \mathcal{N}(0.5, 0.1) & (t = L+1, \ldots, 2L). \end{cases} \tag{20}$$

After we repeated the procedure 10 times, we obtained a subsequence with length $20L$. In the latter part, each datum was drawn from

$$y_t \sim \begin{cases} \mathcal{N}((t - 20L)/2L, 0.1) & (t = 20L+1, \ldots, 21L), \\ \mathcal{N}(0, 0.1) & (t = 21L+1, \ldots, 24L). \end{cases}$$

A metachange along state occurred at $t = 20L + 1$. For change detection, we employed four algorithms for comparison: (1) SMDL [8], a semi-instant method with the MDL change statistics;

(2) ChangeFinder (CF) [1,2,4], a state-of-the-art method of abrupt change detection; (3) Bayesian online change point detection (BOCPD) [3], a retrospective online change point detection with a Bayesian scheme; and (4) ADWIN2 [21], adaptive windowing methods. As we assumed a situation where change and metachange mechanisms do not vary significantly, we decided to choose the best combinations of parameters of each change detection algorithm by grid search, as in [8,27]. We generated 10 sequences with the scheme above and calculated the F-scores for each combination of the following parameters:

- SMDL: Window size $w = 50, 100$ ($L = 500$), $w = 100, 200$ ($L = 1000$), $w = 200, 400$ ($L = 2000$). Threshold parameter $\epsilon = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7$.
- CF: Discounting rate $r = 0.003, 0.005, 0.01, 0.03, 0.1$. Threshold parameter $\delta = 0, 0.5, 1.0, 1.5, 2.0$ (regression orders $k_1, k_2 = 3$, smoothing parameters $T_1, T_2 = 5$).
- BOCPD: Parameter related to change intervals $\alpha = 100, 300, 600$. Threshold parameter $\epsilon = 0.1, 0.3$.
- ADWIN2: Confidence parameter $\delta = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$.

F-score is defined as the harmonic mean of *precision* and *recall*, which are calculated using the number of *true positives* (TP), *false positives* (FP), and *false negatives* (FN) as follows [9]: *TP* is the number of true change points that are τ-neighbors of estimated change points. Thus, *FP* and *FN* are calculated as $FP = \ell - TP$ and $FN = m - TP$, where ℓ and m are calculated as $FP = \ell - TP$ and $FN = m - TP$, where ℓ and m denotes the total number of estimated and true change points, respectively. Finally, we calculated $recall = TP/(TP + FN)$ and $precision = TP/(TP + FP)$ for each method. In this experiment, we set τ to 100.

After optimizing the parameters of each change detection algorithm, we generated 30 data streams with the scheme above and detected change points and the metachange. In the metachange detection, we compared MCD-S with SMDL. We chose SMDL for comparison because it calculates a change score at each time based on changes of parameters with MDL. Hence, a change rate of scores between change points is regarded as the degree of metachange along state. Hereafter, we refer to SMDL for metachange detection as SMDL metachange (SMDL-MC) and the window parameter as w_{mc}. We calculated MCAS in Equation (16) for MCD-S and the change rate $|(\Psi_{t_i} - \Psi_{t_{i-1}})/\Psi_{t_{i-1}}|$ for SMDL-MC. Ψ_t is the *change score* at time t for a univariate normal distribution [8]:

$$\Psi_t = \frac{1}{2} \log \frac{\hat{\sigma}_0^2}{\hat{\sigma}_1 \hat{\sigma}_2} + \log \frac{C_{2w_{\mathrm{mc}}}}{C_{w_{\mathrm{mc}}}^2},$$

where $\hat{\sigma}_0$, $\hat{\sigma}_1$, and $\hat{\sigma}_2$ are the maximum likelihood estimators of standard deviations calculated for $y_{t-w_{\mathrm{mc}}+1}^{t+w_{\mathrm{mc}}}$, $y_{t-w_{\mathrm{mc}}+1}^{t-1}$ and $y_t^{t+w_{\mathrm{mc}}}$, respectively. C_k is the normalizer of the normalized maximum likelihood code length [20]

$$\log C_k = \frac{1}{2} \log \frac{16\mu_{\max}}{\pi \sigma_{\min}^2} + \frac{k}{2} \log \frac{k}{2e} - \log \Gamma \left(\frac{k-1}{2} \right),$$

where Γ is the gamma function. In this paper, $\mu_{\max} = 2$ and $\sigma_{\min} = 0.005$. The window parameters h of MCD-S and w_{mc} of SMDL-MC were set to $h, w_{\mathrm{mc}} = 100$ ($L = 500$), $h, w_{\mathrm{mc}} = 200$ ($L = 1000$), and $h, w_{\mathrm{mc}} = 400$ ($L = 2000$). In calculating the F-scores, the maximum tolerant delay was set to $\tau = 0.5L$.

Table 2 shows the average AUC values of MCD-S and SMDL-MC for the detection of metachanges along state at $t = 20L + 1$. The first and second rows in the header represent change detection and metachange detection algorithms, respectively. The best parameters for each combination of change detection and metachange detection algorithms are $\epsilon = 0.7, w = 100$ ($L = 500$), $\epsilon = 0.7, w = 200$ ($L = 1000$), and $\epsilon = 0.7, w = 400$ ($L = 2000$). Table 2 shows that MCD-S outperforms SMDL-MC overall because MCD-S deals with metachanges along state directly in terms of MCAS, whereas SMDL-MC only quantifies the difference in code lengths between situations where there is a change and where there is no change.

Table 2. Average AUC scores on Synthetic Dataset 2. The first and second headers represent change detection and metachange detection algorithms, respectively. Boldfaces describe best performances.

L	SMDL		CF		BOCPD		ADWIN2	
	MCD-S	SMDL-MC	MCD-S	SMDL-MC	MCD-S	SMDL-MC	MCD-S	SMDL-MC
500	$\mathbf{0.887 \pm 0.100}$	0.795 ± 0.156	0.874 ± 0.111	0.851 ± 0.170	0.701 ± 0.318	0.572 ± 0.332	0.797 ± 0.186	0.853 ± 0.114
1000	$\mathbf{0.921 \pm 0.018}$	0.905 ± 0.012	0.912 ± 0.042	0.830 ± 0.052	0.751 ± 0.323	0.743 ± 0.291	0.834 ± 0.094	0.847 ± 0.048
2000	$\mathbf{0.970 \pm 0.010}$	0.953 ± 0.011	0.912 ± 0.033	0.843 ± 0.022	0.829 ± 0.124	0.821 ± 0.138	0.951 ± 0.032	0.887 ± 0.046

We further investigated the effects of window size h and threshold parameters of the change detection algorithms. We chose SMDL [8] for change detection. Figure 8 shows the dependency of AUC on h and threshold parameter ϵ of SMDL. The interval length was set to $L = 500$, threshold parameter was set to $\epsilon = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7$, and $h = w = 50, 100, 150$, where w is the window parameter of SMDL. Figure 8 (top and bottom) shows the dependency of AUC of MCD-S on the threshold parameter ϵ of SMDL and the dependency of F-score of SMDL on ϵ, respectively. We observe in Figure 8 (top) that AUC of MCD-S decreases between $\epsilon = 0.2$ and 0.4, but, when ϵ exceeds 0.4, AUC begins to increase for $h = 50, 100, 150$. This reflects the fact that there are many local maximum points of the change scores of SMDL, leading to false alarms of change points around $\epsilon = 0.2$–0.4. It is noticeable that F-scores of SMDL decrease for $\epsilon = 0.1$ $(h = 100)$, and for $\epsilon = 0.2$ $(h = 150)$, but AUCs of MCD-S do not do so much. This is because SMDL detects many false positive change points, but it detects the metachange point accurately.

As for the dependency of AUC on window size h, we observe that AUC generally increases as h increases for the same ϵ.

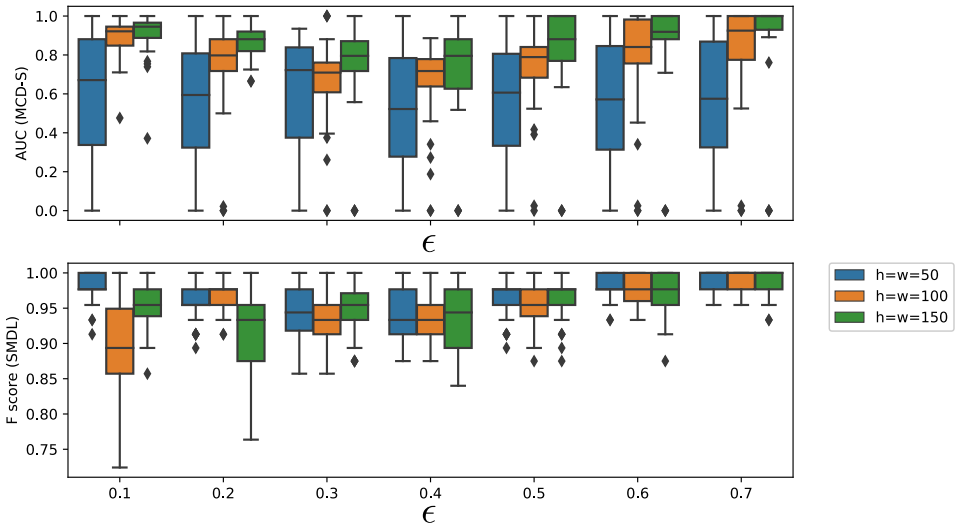

Figure 8. Dependency of AUC on threshold parameter ϵ for SMDL [8] and window size h of MCD-S on Synthetic Dataset 2.

4.3. *Synthetic Dataset 3 (Metachanges Along Time and State)*

We generated a data stream that contained metachanges along both time and state. The stream consisted of two subsequences. The former part repeated changes of mean. Each instance was drawn from Equation (20) with $L = L_1$. We repeated the procedure for 50 times and obtained a subsequence with length $100L_1$. The latter part comprised the following four parts, each with length L_2:

$$y_t \sim \begin{cases} \mathcal{N}(0, 0.1) & (t = 100L_1 + 1, \ldots, 100L_1 + L_2), \\ \mathcal{N}(0.45, 0.1) & (t = 100L_1 + L_2 + 1, \ldots, 100L_1 + 4L_2). \end{cases}$$

In total, we obtained a data stream with length $100L_1 + 4L_2$. A metachange along both time and state occurred at $t = 100L_1 + L_2 + 1$. We chose lengths L_1 and L_2 among 400, 450, and 500.

We detected the metachange in the following three ways: we first detected change points with the same algorithms as in Section 4.2, and then detected the metachanges with MCD-T, MCD-S, and MCD. The parameters of the change detection algorithms were tuned as in Section 4.2. The ranges of parameters were the same as those in Section 4.2. except that, for SMDL, the threshold parameter $\epsilon = 0.05, 0.1, 0.15$ for all combinations of L_1 and L_2. The parameter of MCD-T was selected among $r = 0.1, 0.2, 0.3$ and MCD-S was among $h = 0.1L_1, 0.2L_1$. The window size of SMDL were selected among $w = h$, and the maximum tolerant delay was $\tau = L_2$. We chose the weight parameter λ in Equation (17) among $\lambda = 0.001, 0.01, 0.1, 1, 5, 10$. For VD, the buffer and reservoir sizes (B and R) were selected among $16, 24, 32$. All the parameters were selected with grid search for the AUCs of metachange detection to be maximum.

Table 3 shows the average AUC values. Table 3a–c show average AUC values with MCD-T, MCD-S, and MCD. Table 3a shows that MCD combined with SMDL as the change detection algorithm outperforms MCD-S and MCD-T.

Table 3. Average AUC scores of metachange detection on Synthetic Dataset 3. The first and second headers represent change detection and metachange detection algorithms, respectively. Boldfaces describe best performances.

(**a**) Metachange detection along time.

L_1	L_2	SMDL		CF		BOCPD		ADWIN2	
		MCD-T	VD	MCD-T	VD	MCD-T	VD	MCD-T	VD
400	450	**0.867 ± 0.022**	0.845 ± 0.013	0.818 ± 0.031	0.815 ± 0.025	0.843 ± 0.053	0.825 ± 0.038	0.839 ± 0.048	0.806 ± 0.043
400	500	**0.871 ± 0.021**	0.867 ± 0.021	0.815 ± 0.040	0.812 ± 0.041	0.831 ± 0.049	0.814 ± 0.033	0.823 ± 0.048	0.826 ± 0.038
450	400	**0.813 ± 0.024**	0.804 ± 0.017	0.795 ± 0.044	0.784 ± 0.029	0.810 ± 0.038	0.805 ± 0.031	0.805 ± 0.052	0.812 ± 0.035
450	500	**0.872 ± 0.014**	0.863 ± 0.019	0.822 ± 0.039	0.819 ± 0.047	0.847 ± 0.032	0.829 ± 0.042	0.816 ± 0.044	0.815 ± 0.034
500	400	**0.874 ± 0.024**	0.867 ± 0.024	0.837 ± 0.016	0.813 ± 0.045	0.822 ± 0.019	0.797 ± 0.029	0.815 ± 0.011	0.802 ± 0.031
500	450	**0.893 ± 0.015**	0.873 ± 0.019	0.829 ± 0.013	0.859 ± 0.039	0.833 ± 0.011	0.823 ± 0.049	0.819 ± 0.021	0.875 ± 0.031

(**b**) Metachange detection along state.

L_1	L_2	SMDL		CF		BOCPD		ADWIN2	
		MCD-S	SMDC-MC	MCD-S	SMDC-MC	MCD-S	SMDC-MC	MCD-S	SMDC-MC
400	450	**0.901 ± 0.012**	0.857 ± 0.014	0.823 ± 0.013	0.833 ± 0.024	0.855 ± 0.021	0.858 ± 0.031	0.809 ± 0.015	0.867 ± 0.011
400	500	**0.923 ± 0.016**	0.911 ± 0.023	0.813 ± 0.011	0.812 ± 0.014	0.852 ± 0.034	0.851 ± 0.028	0.805 ± 0.036	0.798 ± 0.024
450	400	**0.895 ± 0.022**	0.875 ± 0.011	0.835 ± 0.021	0.809 ± 0.033	0.855 ± 0.034	0.853 ± 0.025	0.809 ± 0.033	0.892 ± 0.031
450	500	**0.917 ± 0.017**	0.905 ± 0.023	0.842 ± 0.039	0.825 ± 0.047	0.837 ± 0.051	0.819 ± 0.042	0.838 ± 0.044	0.615 ± 0.034
500	400	**0.875 ± 0.024**	0.863 ± 0.022	0.822 ± 0.032	0.813 ± 0.045	0.810 ± 0.026	0.797 ± 0.022	0.729 ± 0.024	0.702 ± 0.023
500	450	**0.865 ± 0.021**	0.823 ± 0.028	0.715 ± 0.038	0.723 ± 0.049	0.728 ± 0.045	0.706 ± 0.038	0.694 ± 0.042	0.675 ± 0.031

(**c**) Metachange detection along both time and state.

L_1	L_2	SMDL	CF	BOCPD	ADWIN2
		MCD	MCD	MCD	MCD
400	450	**0.985 ± 0.011**	0.971 ± 0.023	0.968 ± 0.033	0.967 ± 0.029
400	500	**0.989 ± 0.007**	0.975 ± 0.016	0.971 ± 0.005	0.969 ± 0.031
450	400	**0.983 ± 0.016**	0.981 ± 0.013	0.968 ± 0.035	0.966 ± 0.014
450	500	**0.987 ± 0.010**	0.982 ± 0.014	0.975 ± 0.025	0.970 ± 0.029
500	400	**0.979 ± 0.015**	0.973 ± 0.011	0.969 ± 0.012	0.964 ± 0.013
500	450	**0.975 ± 0.012**	0.969 ± 0.010	0.967 ± 0.018	0.954 ± 0.021

Table 4 shows the best parameters for each combination of intervals. We observe that the more intensive a metachange along time is, the bigger r is and the less λ becomes. These results reflect the

fact that it is necessary to adapt to recent data, and MCAT increases in such a situation, leading to the decrease of λ.

Table 4. Best parameters for each combination of intervals.

L_1	L_2	r	w	h	λ
400	450	0.2	$0.2L_1$	$0.2L_1$	0.1
400	500	0.3	$0.2L_1$	$0.2L_1$	0.01
450	400	0.1	$0.2L_1$	$0.2L_1$	0.1
450	500	0.2	$0.2L_1$	$0.2L_1$	0.1
500	400	0.3	$0.2L_1$	$0.2L_1$	0.01
500	450	0.1	$0.2L_1$	$0.2L_1$	0.1

4.4. Real Dataset: Human Action Recognition Data

We applied MCD to the detection of metachanges in human action recognition data called HASC-PAC2016 dataset [28] (HASC-PAC2016 dataset is publicly available at http://hub.hasc.jp/). The data were collected from the Human Activity Sensing Consortium (HASC, http://hasc.jp/). HASC-PAC2016 dataset contains sequences of acceleration data for three axes, and each sequence is segmented into one of six action labels: "stay", "walk", "jog", "skip", "stair up", (go upstairs) and "stair down" (go downstairs). For this experiment, we aimed to evaluate the effectiveness of our proposed algorithm MCD by using a data stream with ground truth of "changes of action changes" and "changes of intervals of actions". The former corresponds to metachanges along state, and the latter to metachanges along time. We combined each action into a data stream as follows: first, we repeated "stay" and "walk" alternately for 15 times; then "jog" and "skip" for 15 times; and, finally, "stair up" and "stair down" for 15 times. We repeated each pair of actions for 15 times because "stair up" and "stair down" have only 15 files, which are the fewest in all the six actions. We obtained a data stream of length 89,324. Table 5 shows the files used for a participant named Person06023. We read the files sequentially in alphabetical order for each action. Figure 9 shows the data stream we obtained. Here, acc_X, acc_Y, and acc_Z represent accelerations for x-, y-, and z-axes, respectively.

Table 5. Files for generating a sequence of Person06023.

Action Label	Files
stay	HASC N-acc.csv (N = 0605581–0605595)
walk	HASC N-acc.csv (N = 0608420–0608434)
jog	HASC N-acc.csv (N = 0611173–0611187)
skip	HASC N-acc.csv (N = 0613411–0613425)
stair up	HASC N-acc.csv (N = 0615620–0615634)
stair down	HASC N-acc.csv (N = 0614162–0614166)

First, we detected change points with SMDL [8]. It was a challenge to determine the hyperparameters of SMDL—window size w and threshold parameter ϵ—in an online change detection. We tuned w and ϵ with the remaining dataset for Person06023, which alternated "stay" and "walk" four times, and "jog" and "skip" likewise. Although this dataset lacked "stair up" and "stair down", we thought that it was enough to estimate the best configuration of w and ϵ. We calculated F-score as described in Section 4.2 for the change points between different action labels. We selected $w = 900$ and $\epsilon = 0.75$ among $w \in \{500, 600, 700, 800, 900, 1000\}$ and $\epsilon \in \{0, 0.25, 0.5, 0.75, 1\}$. Figure 10 shows histograms of intervals for each action label. We observe in Figure 10 that most of the intervals are around 960–970 for "jog", "walk", and "skip", whereas, for "stay", "stair up", and "stair down", the intervals are around 1020. We can see that $w = 900$ was enough to detect changes.

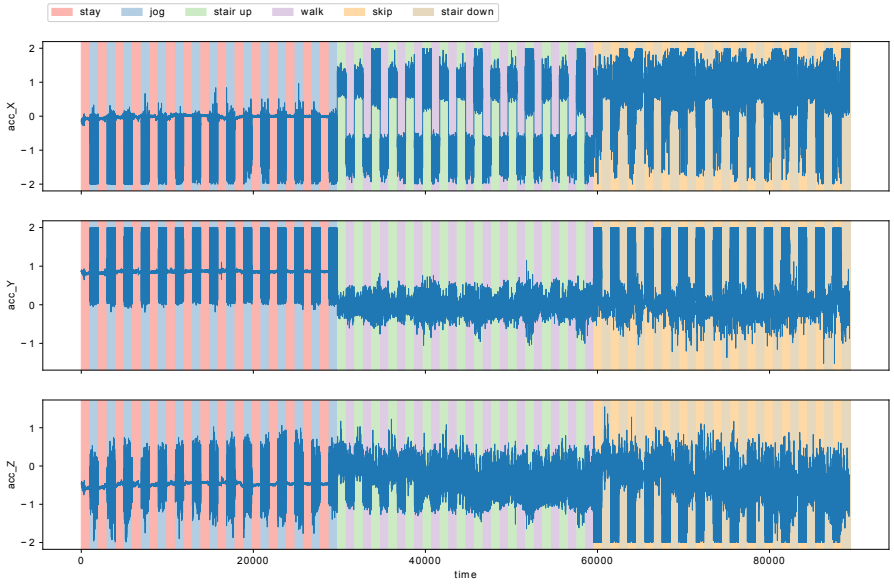

Figure 9. Human action recognition data for Person06023. Each row represents accelerations for x-, y-, and z-axes, respectively.

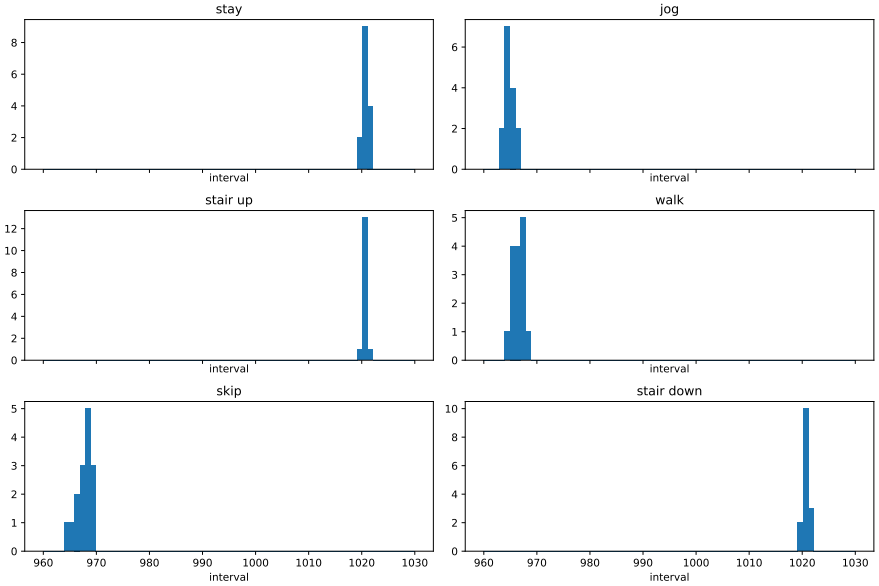

Figure 10. Histograms of intervals for each action label.

We applied SMDL to the stream and obtained the estimated change scores $\{\Psi_t\}$ at each time point. We calculated Ψ_t with the multivariate normal distribution. Specifically, Ψ_t is calculated as

$$\begin{aligned}\Psi_t = &\frac{1}{2}\log\frac{|\hat{\Sigma}_0|^2}{|\hat{\Sigma}_1||\hat{\Sigma}_2|} + \frac{1}{2w}\log\frac{C_{2w}}{C_w^2} \\ &+ \frac{1}{2w}\left\{\sum_{i=t-w}^{t+w}(y_i-\hat{\mu}_0)^\top\hat{\Sigma}_0^{-1}(y_i-\hat{\mu}_0)\right. \\ &\left. - \sum_{i=t-w}^{t-1}(y_i-\hat{\mu}_1)^\top\hat{\Sigma}_1^{-1}(y_i-\hat{\mu}_1) - \sum_{i=t}^{t+w}(y_i-\hat{\mu}_2)^\top\hat{\Sigma}_2^{-1}(y_i-\hat{\mu}_2)\right\},\end{aligned} \tag{21}$$

where $\hat{\mu}_0 = 1/(2w+1)\sum_{i=t-w}^{t+w} y_i$, $\hat{\mu}_1 = 1/w\sum_{i=t-w}^{t-1} y_i$, and $\hat{\mu}_2 = 1/(w+1)\sum_{i=t+1}^{t+w} y_i$. $\hat{\Sigma}_0 = \frac{1}{2w}\sum_{i=t-w}^{t+w}(y_i-\hat{\mu}_0)(y_i-\hat{\mu}_0)^\top$, $\hat{\Sigma}_1 = \frac{1}{w}\sum_{i=t-w}^{t-1}(y_i-\hat{\mu}_1)(y_i-\hat{\mu}_1)^\top$, and $\hat{\Sigma}_1 = \frac{1}{w+1}\sum_{i=t}^{t+w}(y_i-\hat{\mu}_2)(y_i-\hat{\mu}_2)^\top$.

Note that C_w in Equation (21) is the normalizer of the NML code length [29,30]:

$$\log C_w = -(m+1)\log\frac{m}{2} + \frac{m}{2}\log\mu_{\max} - \frac{m^2}{2}\log\sigma_{\min} + \frac{mw}{2}\log\frac{w}{2e} - \log\Gamma\left(\frac{m}{2}\right) - \log\Gamma_m\left(\frac{w-1}{2}\right), \tag{22}$$

where m is the dimension of the data stream, Γ is the gamma function, and Γ_m is calculated as

$$\Gamma_m(x) = \pi^{\frac{m(m-1)}{4}} \prod_{j=1}^{m} \Gamma\left(x + \frac{1-j}{2}\right).$$

We set $\mu_{\max} = 50$ and $\sigma_{\min} = 0.005$.

Next, we defined the ground truths for metachanges along state at two time points where the changes of action label changes occurred: t =29,752 from "jog" to "stair up", and t =59,588 from "walk" to "skip". Moreover, we also defined the ground truths for metachanges along time at time points where the changes of intervals occurred. We see in Figure 10 that the distributions are significantly different between four types of "changes of action changes": from "stay" to "jog", from "jog" to "stair up", from "stair up" to "walk", and from "skip" to "stair down".

We detected metachanges along time with MCD-T and volatility detector (VD) [13], and compared them. Figure 11 shows the estimated MCAT with MCD-T and the relative volatility with VD. The parameter of MCD-T was set to $r = 0.1, 0.2, 0.3$, whereas one of VD was $B = R = 10, 15, 20$. Figure 11 shows the results.

We observe in Figure 11 that MCAT detects the metachanges along time between the four action pairs, respectively, for $r = 0.1, 0.2$, and 0.3. However, the relative volatility fails to detect some of these metachanges along time.

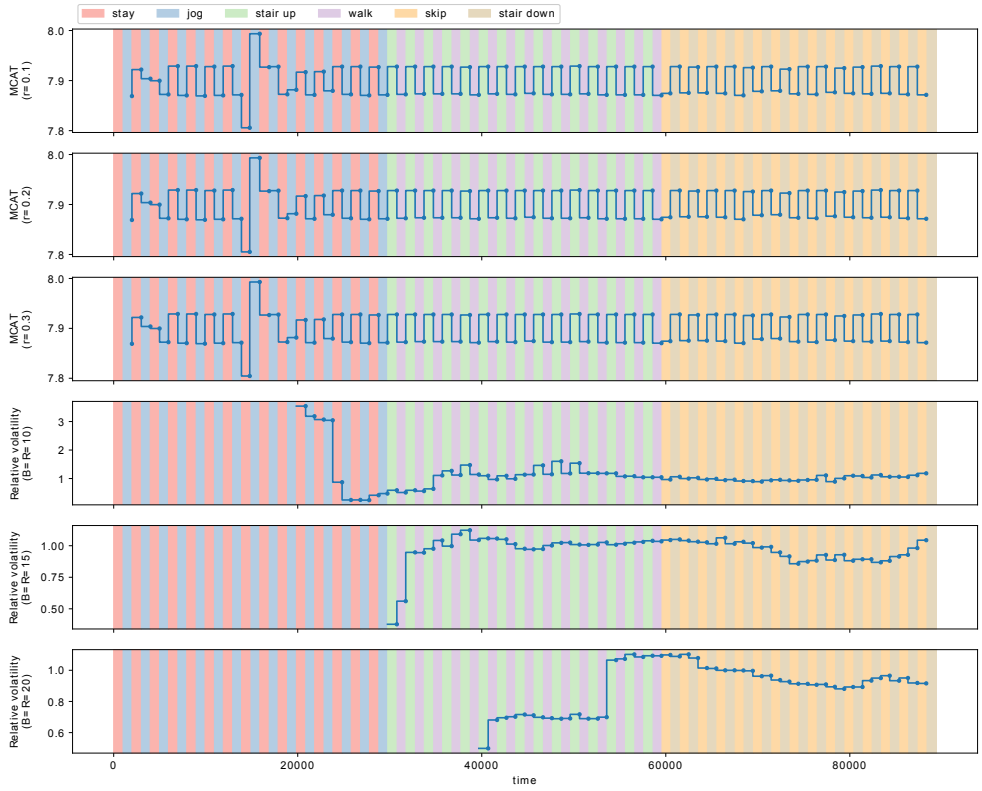

Figure 11. MCAT with MCD-T ($r = 0.1, 0.2, 0.3$) and the relative volatility with the volatility detector [13] ($B = R = 10, 15, 20$).

We detected metachanges along state with MCD-S and the change rate of the MDL change statistics [8]. Figure 12 shows the estimated MCAS with MCD-S and the MDL change statistics. We observe in Figure 12 that both MCD-S and the MDL change statistics detect a time point around t =29,752 from "jog" to "stair up". However, the MDL change statistics do not change significantly at a time point around t =59,588, where a metachange along state happened from "walk" to "skip". It indicates that the change rate of the MDL change statistics failed to detect the metachange along state around t =59,588, whereas MCD-S detected it successfully.

Figure 12. MCAS of MCD-S ($h = 900$) and the MDL change statistics ($w = 900$).

In summary, the proposed algorithm MCD detected metachanges along both time and state more accurately than other methods.

4.5. Real Dataset: Production Condition Data

We applied MCD to the detection of metachanges in the production condition data. The data were collected from a factory of a manufacturing company. Each datum comprised eight attributes, and the length of the stream was 26,450. The factory reported that important events occurred 10 times during the study period, at t =668, 2634, 2635, 9663, 13,230, 13,231, 17,372, 17,832, 20,131, and 25,441. Figure 13 shows the attributes from the stream. The dashed line indicates the time points where important events occurred. We investigated whether the detected metachanges were signs of important events, and we finally concluded that it might be true. The details are as follows.

Figure 13 shows that the scales of attributes were different. Hence, we normalized each attribute X to $(X - \mu)/\sigma$, where μ and σ are the sample mean and standard deviation, respectively, which were calculated with the first 250 time points. First, we applied SMDL [8] to the stream and obtained the estimated change scores $\{\Psi_t\}$ at each time. We calculated Ψ_t with the multivariate normal distribution in Equation (21). The window sizes w of SMDL and h of MCD were set to $w = h = 250$ by field knowledge that it roughly represents a unit of production. Moreover, $\mu_{\max}$ and $\sigma_{\min}$ in Equation (22) were set to 60 and 0.001, respectively. Next, we detected change points t_1, t_2, ... as time points where the change scores Ψ_{t_i} were locally maximum within an interval where $\Psi_t > \epsilon$. We set $\epsilon = 0.3$ when the total change points detected was less than 0.5% of the total length. It is a business demand by a factory, and so there were not many alarms. The number of detected change points was 97 (0.37%). Finally, we determined the discounting parameter r and the weight parameter λ of MCD in Equation (17) with the first 5000 time points. We selected $r = 0.1$ and $\lambda = 0.2$ so that the AUC score at $t = 2634$ and $t = 2635$ would be the maximum. The AUC score was calculated using Equations (18) and (19).

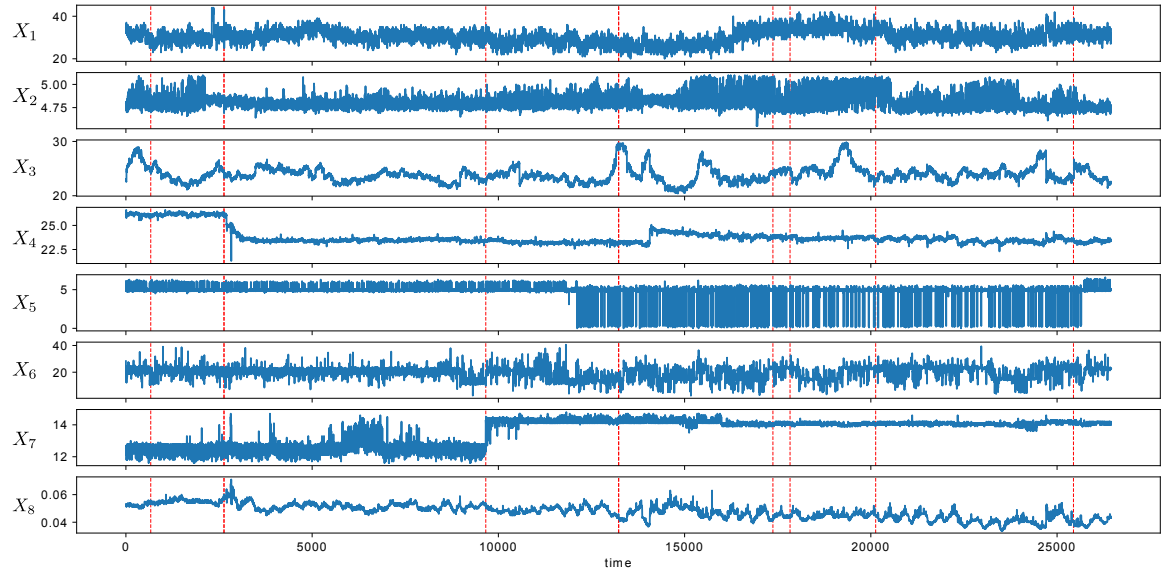

Figure 13. Data stream of the production condition data. Red dashed line indicates the time points where the important events occurred.

Figure 14 shows the MDL change statistics $\{\Psi_t\}$ calculated with SMDL [8] (Figure 14, top), the estimated MCAT a_{t_i} (Figure 14, second), logarithm of the estimated MCAS $\log_{10} b_{t_i}$ (Figure 14, third), and logarithm of the estimated MCI $\log_{10} s_{t_i}$ (Figure 14, fourth). We also estimated the relative volatility with VD [13,25] (Figure 14, fifth) and the change rate of the MDL change statistics $|(\Psi_{t_i} - \Psi_{t_{i-1}})/\Psi_{t_{i-1}}|$ (Figure 14, bottom) for comparison in detecting metachanges along both time and state. For VD, the buffer size B and the reservoir size R were both set to 10. In Figure 14 (top), the red points indicate the detected change points.

We summarize what can be seen for metachange statistics in Figure 14 as follows:

- $t = 9663$: The trend of MCI increases roughly after $t = 5000$, which can be interpreted as a combination of MCAT and MCAS in Figure 14. The relative volatility and the change rate of the MDL change statistics do not show such a significant sign.
- $t = 13{,}230$, 13,231, 17,372, 17,832: For time points between $t = 10{,}000$ and $t = 15{,}000$, the trend of MCI increases. It is also due to the combination of MCAT and MCAS, but is more influenced by MCAS. It might also be a sign of important events at $t = 17{,}372$ and 17,832 as well as $t = 13{,}230$ and $t = 13{,}231$. The relative volatility increases after $t = 13{,}231$, which might be a sign of the important event at $t = 17{,}372$. However, the change rate of the MDL change statistics does not show such a significant sign.
- $t = 25{,}440$: For time points between $t = 20{,}000$ and $t = 25{,}000$, the trend of MCI increases with large fluctuations. It is also more influenced by MCAS. It might also be a sign of important events at $t = 25{,}440$. The relative volatility increases for the time points, but the change rate of the MDL change statistics does not show such a significant sign.

In summary, we can observe a sign of metachange for each important event. We therefore infer that there might have been some symptoms that should be analyzed using field knowledge.

Figure 14. Metachange statistics of the production condition data: (**top**) the MDL change statistics $\{\Psi_{t_i}\}$. Blue dots show change points $\{t_i\}$, where $\Psi_{t_i} > \epsilon$; (**second**) estimated MCAT a_{t_i}; (**third**) estimated logarithm of MCAS $\log_{10} b_{t_i}$; (**fourth**) estimated logarithm of integrated metachange statistics (MCI) $\log_{10} s_{t_i}$; (**fifth**) relative volatility [13]; and (**bottom**) change rate of the MDL change statistics $|(\Psi_{t_i} - \Psi_{t_{i-1}})/\Psi_{t_{i-1}}|$. $h = w = 250, \epsilon = 0.3, B = R = 10, \lambda = 0.2$.

5. Conclusions

We propose the concept of *metachanges* along time and state in data streams, and we introduce *metachange statistics* to quantify metachanges from a unified view with MDL. The key idea of our proposed method is to encode the time intervals and change of states with code lengths in the same fashion. Next, we introduce the novel methodology of MCD. Using synthetic datasets, we empirically demonstrated that the proposed algorithm was highly effective in detecting metachanges along time and state. Using a real dataset, we demonstrated that the proposed algorithm could detect metachanges in both time and state, some of which were overlooked by VD [13] and the MDL change statistics [8]. The estimated metachange statistics might have been a sign of important events.

Future work will be directed toward the theoretical guarantee of metachange statistics, especially integrated metachange statistics. We will also consider how to adapt to a non-stationary data stream by updating the weight parameter λ in Equation (17). Other research directions might lie in the extension of metachange statistics to transient periods between change points. Furthermore, metachange detection of model structure change and its change sign is another interesting line of research.

Author Contributions: Conceptualization, S.F. and K.Y.; methodology, S.F. and K.Y.; software, S.F.; validation, S.F.; formal analysis, S.F.; investigation, S.F.; resources, K.Y.; data curation, S.F.; writing–original draft preparation, S.F.; writing–review and editing, S.F. and K.Y.; visualization, S.F.; supervision, K.Y.; project administration, K.Y.; funding acquisition, K.Y.

Funding: This work was partially supported by JST KAKENHI 19H01114 and JST-AIP JPMJCR19U4.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Yamanishi, K.; Takeuchi, J. A unifying framework for detecting outliers and change points from non-stationary time series data. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), Edmonton, AB, Canada, 23–25 July 2002; pp. 676–681.
2. Takeuchi, J.; Yamanishi, K. A unifying framework for detecting outliers and change-points from time series. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 482–492. [CrossRef]
3. Adams, R.; MacKay, D. Bayesian online changepoint detection. *arXiv* **2007**, arXiv:0710.3742.
4. Takahashi, T.; Tomioka, R.; Yamanishi, K. Discovering emerging topics in social streams via link anomaly detection. *IEEE Trans. Knowl. Data Eng.* **2014**, *26*, 120–130. [CrossRef]
5. Miyaguchi, K.; Yamanishi, K. On-line detection of continuous changes in stochastic processes. In Proceedings of the 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Paris, France, 19–21 October 2015; pp. 1–9.
6. Yamanishi, K.; Maruyama, Y. Dynamic syslog mining for network failure monitoring. In Proceedings of the eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD), Chicago, IL, USA, 21–24 August 2005; pp. 499–508.
7. Yamanishi, K.; Maruyama, Y. Dynamic model selection with its applications to novelty detection. *IEEE Trans. Inform. Theory* **2007**, *53*, 2180–2189. [CrossRef]
8. Yamanishi, K.; Miyaguchi, K. Detecting gradual changes from data stream using MDL-change statistics. In Proceedings of the 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, USA, 5–8 December 2016; pp. 156–163.
9. Kaneko, R.; Miyaguchi, K.; Yamanishi, K. Detecting changes in streaming data with information-theoretic windowing. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017; pp. 646–655.
10. Yamanishi, K.; Fukushima, S. Model change detection with the MDL Principle. *IEEE Trans. Inform. Theory* **2018**, *64*, 6115–6126. [CrossRef]
11. Aminikhanghahi, S.; Cook, D.J. A survey of methods for time series change point detection. *Knowl. Inf. Syst.* **2017**, *51*, 339–367. [CrossRef] [PubMed]

12. Kleinberg, J. Bursty and hierarchical structure in streams. *Data Min. Knowl. Discov.* **2003**, *7*, 373–397. [CrossRef]
13. Huang, D.; Koh, Y.S.; Dobbie, G.; Pears, R. Detecting volatility shift in data streams. In Proceedings of the 2014 IEEE International Conference on Data Mining (ICDM), Shenzhen, China, 14–17 December 2014; pp. 863–868.
14. Huang, D.; Koh, Y.S.; Dobbie, G.; Pears, R. Tracking drift types in changing data streams. In Proceedings of the International Conference on Advanced Data Mining and Applications, Hangzhou, China, 14–16 December 2013; pp. 72–83.
15. Aggarwal, C. A framework for diagnosing changes in evolving data streams. In Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (SIGMOD), San Diego, CA, USA, 9–13 June 2003; pp. 575–586.
16. Spiliopoulou, M.; Ntoutsi, I.; Theodoridis, Y.; Schult, R. MONIC: Modeling and monitoring cluster transitions. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), Philadelphia, PA, USA, 20–23 August 2006; pp. 706–711.
17. Spiliopoulou, M.; Ntoutsi, E.; Theodoridis, Y.; Schult, R. MONIC and followups on modeling and monitoring cluster transitions. In Proceedings of the joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Prague, Czech Republic, 23–27 September 2013; pp. 622–626.
18. Ntoutsi, I.; Spiliopoulou, M.; Theodoridis, Y. Summarizing cluster evolution in dynamic environments. In Proceedings of the International Conference on Computational Science and Its Applications, Santander, Spain, 20–23 June 2011; pp. 562–577.
19. Gama, J.; Žliobaitė, I.; Bifet, A.; Mykola, P.; Abdelhamid, B. A survey on concept drift adaptation. *ACM Comput. Surv.* **2014**, *46*, 44:1–44:37. [CrossRef]
20. Rissanen, J. *Optimal Estimation of Parameters*; Cambridge University Press: Cambridge, UK, 2012.
21. Bifet, A.; Gavaldá, R. Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM International Conference on Data Mining, Philadelphia, PA, USA, 26–28 April 2007; pp. 443–448.
22. van Leeuwen, M.; Siebes, A. StreamKrimp: Detecting change in data streams. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Antwerp, Belgium, 15–19 September 2008; pp. 672–687.
23. Rissanen, J. Stochastic complexity and modeling. *Ann. Stat.* **1986**, *14*, 1080–1100. [CrossRef]
24. Yamanishi, K.; Takeuchi, J.; Williams, G.; Milne, P. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Min. Knowl. J.* **2004**, *8*, 275–300. [CrossRef]
25. Huang, D. Change Mining and Analysis for Data Streams. Ph.D. Thesis, The University of Auckland, Auckland, New Zealand, 2015.
26. Fawcett, T.; Provost, F. Activity monitoring: noticing interesting changes in behavior. In Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), San Diego, CA, USA, 15–18 August 1999; pp. 53–62.
27. Liu, S.; Yamada, M.; Collier, N.; Sugiyama, M. Change-point detection in time-series data by relative density-ratio estimation. *Neural Netw.* **2013**, *43*, 72–83. [CrossRef] [PubMed]
28. Ichino, H.; Kaji, K.; Sakurada, K.; Horii, K.; Kawaguchi, N. HASC-PAC2016: Large scale human pedestrian activity corpus and its baseline recognition. In Proceedings of the UBICOMP/ISWC'16 adjunct, Heidelberg, Germany, 12–16 September 2016; pp. 705–714.
29. Hirai, S.; Yamanishi, K. Efficient computation of normalized maximum likelihood coding for Gaussian mixtures with its applications to optimal clustering. In Proceedings of the IEEE International Symposium on Information Theory, St. Petersburg, Russia, 31 July–5 August 2011; pp. 1031–1035.
30. Hirai, S.; Yamanishi, K. Efficient computation of normalized maximum likelihood coding for Gaussian mixtures with its applications to optimal clustering. *IEEE Trans. Inform. Theory* **2013**, *59*, 7718–7727. [CrossRef]