2015 Seventh International Conference on Knowledge and Systems Engineering

Concept Drift Detection with Clustering via Statistical Change Detection Methods

Yusuke Sakamoto, Ken-ichi Fukui
The Institute of Scientific and Industrial Research, Osaka University, Japan
Email: fukui@ai.sanken.osaka-u.ac.jp

João Gama
University of Porto, Portugal

Daniela Nicklas
University of Bamberg, Germany

Koichi Moriyama, Masayuki Numao
The Institute of Scientific and Industrial Research, Osaka University, Japan

***Abstract*—We propose a concept drift detection method utilizing statistical change detection in which a drift detection method and the Page-Hinkley test are employed. Our method enables users to annotate clustering results without constructing a model of drift detection for every input. In our experiments using synthetic data, we evaluated our proposed method on the basis of detection delay and false detection, also revealed relations between the degree of drift and parameters of the method.**

I. Introduction

Over the last decade, electronic signal data have been produced and rapidly stored. A data stream is a series of data continuously produced at high speed. Because of this nature of streaming data, algorithms have to use limited computational resources in terms of computational power, memory, communication, and processing time [1].

In this work, we aim at developing a monitoring system for damage within fuel cells. Such damage is observed via ultrasonic waves of acoustic emission (AE) produced by cracks and/or delamination of the materials. Fukui et al. validated that major types of damage can be clustered on AE events via Kernelized self-organizing maps (SOM) [2]. Here it is impractical to collect all possible damage types in advance of monitoring; therefore, it is important to renew the cluster model while adapting to changes in characteristics and types of damage, i.e., concept drift [1], [3].

One way to adapt to concept drifting stream data is an online learning approach wherein the model is updated regardless of whether a concept drift exists [4]; however, since online learning updates frequently and changes the model, it is difficult for a user to intervene. Another way to adapt is to first detect a concept drift, and then update the model via a batch approach [5]–[10]. Using this methodology, a user can intervene since the frequency of model updates can be restrained. We employed the latter methodology because of the labeling (i.e., annotation) process for SOM clustering in our monitoring system.

Until now, research focused on concept drift detection has primarily comprised a binary classification problem or supervised learning [6]–[8]. In our application, AE events are not always easy to categorize (i.e., label); hence, unsupervised learning is preferable in our case. As for concept drift detection methods with clustering, density-based approaches have been proposed [9], [10]; however, a density-based method generally has high computational cost. Morevover, the model must be constructed for every data point or every window to compare to past models. Model construction for concept drift detection may cause detection delays of abnormalities during monitoring.

In this work, we therefore propose a concept drift detection method with clustering by adopting a statistical change detection method, namely the drift detection method (DDM) [6] used for classification problems or supervised learning, and a signal change detection method [11] in which the PageHinkley test (PHT) is used. Our method allows a user to intervene and is a non-density-based drift detection method for low computational cost, since there is no model construction for a drift detection. Here, though the SOM model has to be reconstructed when a drift is detected, in this paper, we focus on concept drift detection. Our experiments using synthetic data validated the combined DDM and PHT-based proposed method in terms of delays and false detection; furthermore, it revealed relations between parameters in DDM/PHT and the drift degree.

II. Monitoring Architecture

To handle high-frequency sensor data streams and to cluster AE events together with annotations, we propose the architecture shown in Fig. 1, which combines a data stream management system (DSMS) [12], [13] and a data mining method [2] in cooperation with a user. The figure shows the two types of processing. The low-frequency process for clustering and labeling, which involves a domain expert, must only be done when data changes on the left side of the figure, whereas the high-frequency cluster assignment and complex event processing are performed when data on the right side delivers information in near real-time to the monitoring application.

In the figure, the dotted line represents the system boundary of the DSMS. The raw sensor stream is first preprocessed to detect an AE event and extract features of the event (preprocessing), and then the event is assigned to a pre-constructed cluster map using the SOM (cluster assignment). Afterward, concept drift is checked based on the cluster assignment of the new event (concept drift detection). If a concept drift is detected, clustering is re-produced using recent data (update cluster map). Then, labels for clusters are updated, if necessary, with a user intervention (update labeling), i.e., a meaningful domain-based name such as “cracks in the electrode”, and then

978-1-4673-8013-3/15 $31.00 © 2015 IEEE
DOI 10.1109/KSE.2015.19

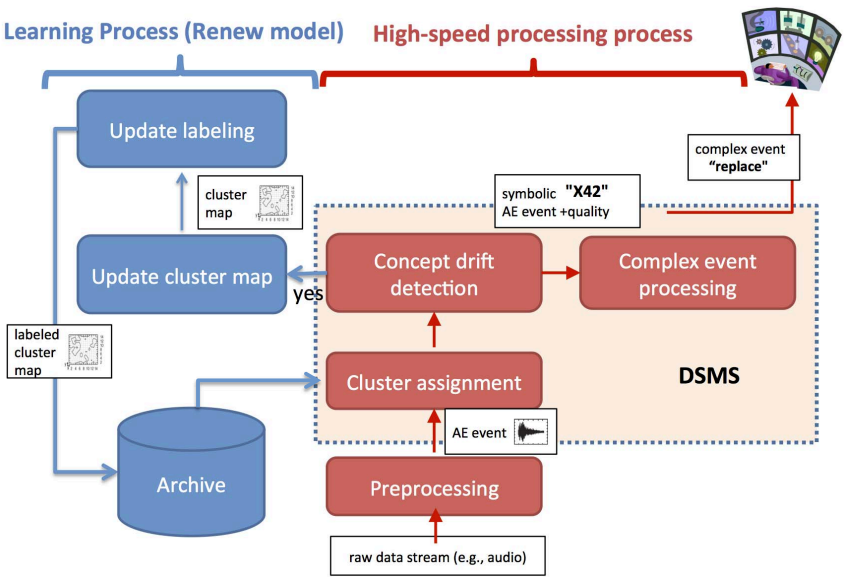


Fig. 1. Proposed monitoring architecture

archive the labeled cluster map (archive). While, if a concept drift does not exist, the system checks user pre-defined rules that combine events, count events, or detect temporal event patterns to provide alert to the user (complex event processing). These rules can be defined by symbolic rules, wherein the symbols are derived from labels of clusters. For example, if damaged events detected on the electrode are more than a predefined threshold, then the damage monitor application is informed or the damage control workflow is invoked.

Since the DSMS runs in memory, relevant information should be archived in a database, we recommend to store at least the complex and symbolic events for later analysis. For provenance, the labeled cluster maps or raw AE events may also be archived, but this may lead to excessive volumes of data, hence, using intelligent archiving techniques would be preferred.

In the next section, we mainly focused on the concept drift detection method used in our architecture.

III. Concept Drift

In streaming data, the nature of data often changes over time due to factors such as changes in situations, degradation of sensors, and accidents. "Concept drift" refers to changes of a "concept" or model that the algorithm must learn over time [1], [3].

A. Types of Concept Drift

There are several types of concept drift. Given conditional probability distribution $\Pr(\mathbf{X}|\mathbf{S})$ with variables $\mathbf{X}$ and information source $\mathbf{S}$, we next introduce some concept drift types that are related to our solid oxide fuel cell (SOFC) damage monitoring system.

1) Sudden Drift: Illustrated in Fig. 2a, sudden drift is a simple change in which the conditional probability distribution changes from $\Pr(\mathbf{X}|\mathbf{S}_1)$ to $\Pr(\mathbf{X}|\mathbf{S}_2)$ at time t due to changes of the information source from $\mathbf{S}_1$ to $\mathbf{S}_2$. Examples on the application of SOFC monitoring include the exchange of a cell to a new one, and sudden equipment failure.

2) Gradual Drift: Illustrated in Fig. 2b, gradual drift refers to a slow change of the information source from $\mathbf{S}_1$ to $\mathbf{S}_2$. Here, $\Pr(\mathbf{S}_1)$ gradually decreases and $\Pr(\mathbf{S}_2)$ increases over time. In SOFC monitoring, damage types change over time depending on the inner state [2], for example, damage gradually shifts from cracks in the electrolyte to cracks in the glass seal.

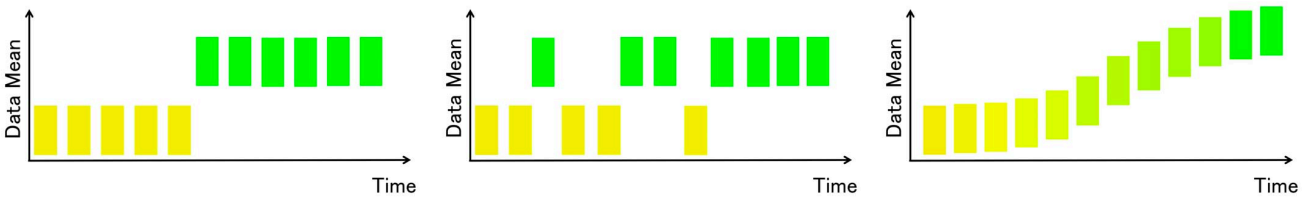
3) Incremental Drift: Illustrated in Fig. 2c, incremental drift is a gradual change of $\Pr(\mathbf{X}|\mathbf{S})$ that becomes a different distribution of variable $\mathbf{X}$ due to changes in the nature of $\mathbf{S}$. This type of drift is typically the most difficult to detect. In SOFC monitoring, the characteristic power spectrum of AE events, even from the same material, changes depending on the temperature and oxidation stage [2].

B. Statistical Concept Drift Detection Methods

In this section, we introduce drift detection methods using supervised classification and signal processing; furthermore, we propose an approach for applying unsupervised clustering.

1) Drift Detection Method (DDM): Gama et al. proposed DDM, which utilizes statistical process control (SPC) for binary classification problems [6]. We describe the details of DDM below.

Let data $(\vec{x_i}, y_i)$ be continuously provided, where $\vec{x_i}$ is a feature vector, y_i is its binary class, and i is the data number. Then, a trained decision model predicts a class $\widehat{y}$ for each input. Here, $y_i = \widehat{y}$ is positive and $y_i \neq \widehat{y}$ is false (i.e., error), i.e., an error is determined by a Bernoulli trial. Therefore, as false



(a) Sudden Drift (b) Gradual Drift (c) Incremental Drift

Fig. 2. Types of concept drifts

Algorithm 1 Drift Detection Method (DDM)

Input:
labeled dataset $x_1, x_2, \cdots, x_t$
warning threshold t_w (default $t_w = 2$)
detection threshold t_d (default $t_d = 3$)
warm-up window size w_0 (default $w_0 = 30$).

1. Initialize minimum classification error $p_{min} = \infty$ and corresponding standard deviation $s_{min} = \infty$. Set warning zone flag f_w to false and $w_1 = 0$.

2. For $j = 1$ to $t - 1$ (all observations)
if $j < w_0$ then **then**
$w_{j+1} = w_j + 1$(warm up, only grow the window)
else
i. Train a classifier on current window of size w_j.
ii. Classify observation w_{j+1}.
iii. Update the error rate over the current window. Let $\widehat{p}$ be the updated error rate and $\widehat{s} = \sqrt{p_i(1-p_i)/w_j}$ be the updated standard deviation.
iv. If $(\widehat{p}+\widehat{s}) < p_{min}+s_{min}$, then update minimum error via $p_{min} = \widehat{p}$ and $s_{min} = \widehat{s}$
v. If $(\widehat{p}+\widehat{s}) \geq (p_{min} + t_d * s_{min})$ and $f_w = true$ then a change is detected; set detection time as $t_d = j$
Obtain all observations on the new training and detection window since t_w (size of $w_{j+1} = j - t_w + 1$), and set $p_{min} = \infty$, $s_{min} = \infty$ and $t_w = \infty$.
ElseIf $(\widehat{p}+\widehat{s}) \geq (p_{min} + t_w * s_{min})$ then
If $f_w = false$ then
switch warning zone flag $f_w = true$ and set warning time $t_w = j$.
Else
set $f_w = false$ and update the window by adding x_{j+1} to it (size $w_{j+1} = w_j + 1$)
end if
$DT_{DDM} = t_w$
Output: detection time DT_{DDM}

probability p_i follows a binomial distribution, the standard deviation s_i of p_i can be given by

$$s_i = \sqrt{p_i(1-p_i)/i}. \tag{1}$$

In DDM, p_i and s_i are updated and maintained for successive inputs, then used for drift detection. These values are updated when $p_i + s_i < p_{min} + s_{min}$ is satisfied. Here, a binomial distribution approximates a normal distribution when the number of samples is sufficient. Therefore, confidence interval $1 - \delta/2$ can be $p_i \pm \alpha * s_i$, where α is a confidence coefficient. For example, $\alpha = 2$ and $\alpha = 3$ corresponds to approximately 95% and 99% confidence intervals, respectively.

The original DDM uses two thresholds, namely "Warning level" and "Drift level". The warning level uses $\alpha = 2$ as $p_i + s_i \geq p_{min} + 2 * s_{min}$, whereas the drift level uses $\alpha = 3$ as $p_i + s_i \geq p_{min} + 3 * s_{min}$. When input x_w exceeds the warning level, all inputs after x_w are stored. Then, when input x_d exceeds the drift level, the model is reconstructed using the stored data $(x_w, \cdots, w_d)$. In addition, p_i and s_i are reset. This DDM algorithm is presented in Algorithm 1.

2) Application to Clustering: As mentioned above, DDM was originally developed for binary classification problems. The false probability p_i is followed by the binomial distribution; hence its standard deviation s_i can be calculated by (1). This work utilizes the assignment error of data x_i to the SOM model as p_i as

$$p_i = \min_j ||x_i - W_j|| \ (j = 1, 2, ..., m) \tag{2}$$

where W_j denotes the reference vector of the j-th neuron node in SOM and m is the number of nodes.

Regarding s_i, by using window as $p_{i-w}, p_{i-w+1}, ..., p_i$ (where w is the window size) for p_i, s_i is calculated as

$$s_i = \sqrt{\frac{\sum_{n=i-w}^{i}(p_n - \mu)^2}{w+1}}, \tag{3}$$

where μ is an average of p_i in the window.

3) Page-Hinkley test (PHT): This test [14] is a sequential analysis method that has been used for change detection [11]. PHT assumes that a random variable follows a normal distribution and detects a change in its average. In PHT, cumulative error U_T from the beginning to the current time is calculated as

$$U_T = \sum_{t=1}^{T}(y_t - \overline{y}_T - \sigma), \tag{4}$$

where y_t is an observed value, $\overline{y}_T$ is an average observed value given by $\overline{y}_T = 1/T \sum_{t=1}^{T} y_t$, σ is a parameter to determine the degree to which noise is permitted, and m_T is maintained as the minimum value of U_T, i.e., $m_T = min(U_1, U_2, \cdots, U_T)$. Then, variable PH_T is given as $PH_T = U_T - m_T$. Concept drift is judged when PHT becomes greater than the predefined parameter λ, i.e.,

$$PH_T > \lambda. \tag{5}$$

Algorithm 2 Page-Hinkley test (PHT)

Input:
dataset $y_1, y_2, \cdots, y_t$
magnitude threshold σ
detection threshold λ
for $t > 0$ **do**
1 Compute
$\overline{y}_T = 1/T \sum_{t=1}^{t} x_t$
$U_T = \sum_{t=1}^{T} (x_t - \overline{y}_T - \sigma)$
$m_T = min(U_1, U_2, ..., U_T)$
If $PH_T = U_T - m_T > \lambda$
Return and report a change at time t_{PH}
else
Return to 1
end for
Output: detection time t_{PH}

The complete PHT algorithm is described in Algorithm 2. Here, in our situation, the assignment error eq. (2) is also used for y_i in PHT.

IV. Experiments

We validated the DDM and PHT-based concept drift detection methods for clustering using synthetic data, which we describe below.

A. Synthetic Data

The synthetic data is generated from two classes of two-dimensional normal distributions with different centers. Fig. 3 illustrates a data distribution in which the distance between classes is 1.5. The distance between classes varied in our experiments.

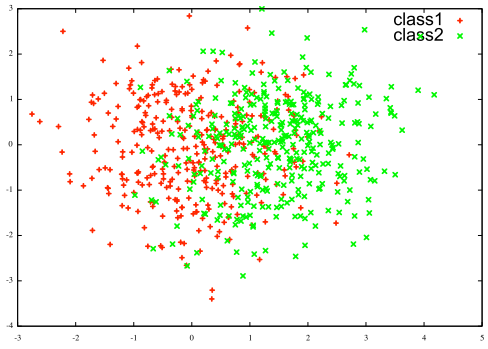


Fig. 3. Data distribution of synthetic data

B. Experimental Procedure

We conducted our experiments as follows:

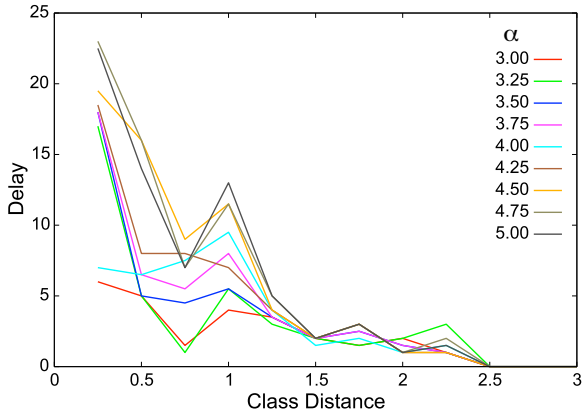


Fig. 4. Relation between the drift degree and detection delay in DDM

Experimental Procedure

1) Generate data for class 1 (350 points) and for class 2 (350 points) with different centers.
2) Construct a SOM model for class 1 (using 300 points).
3) Successively assign class 1 (50 points) followed by class 2 (350 points) to the SOM model. Judge a concept drift via DDM or PHT for every assignment.
4) Repeat Steps 1 to 3 several times with different random values for the data distributions.

For SOM settings, we used a regular grid of 10×10 nodes, a Gaussian function for the neighborhood function, and randomly generated initial reference vectors.

C. Experiment 1. Detection Delay and Drift Degree

We first validated the effect of detection delay by varying the distance between classes of data distributions as $0.25, 0.50, 0.75, \cdots, 2.00$, which corresponds to drift degree. The smaller the distance, the smaller the drift, thus making the drift degree more difficult to detect. For each distance, we tested 30 trials with different random values for the data distributions. In addition, the earliest correct detection was data number 51; therefore, we set this moment as zero delay.

1) DDM: Detection parameter α were varied as $2.00, 2.25, 2.50, \cdots, 5.00$ with fixed window size $w = 10$. Note that as mentioned above, since p_i was not guaranteed to follow a binomial distribution in our method, α does not correspond to a confidence interval. Therefore, we varied α as a parameter to check its relevance to detection. Fig. 4 shows our results with the distance between classes represented on the x-axis and medians of detection delay on the y-axis.

2) PHT: Detection parameter λ were varied as $0.2, 0.4, 0.6, \cdots, 2.0$ with the other parameter fixed at $\sigma = 0.05$, which has been commonly used in previous research [11]. Results shown in Fig. 5 are the same as in DDM.

3) Discussion: From Figs. 4 and 5, when the distance between classes is large, the detection delay becomes small in both methods. From this result, we conclude that our

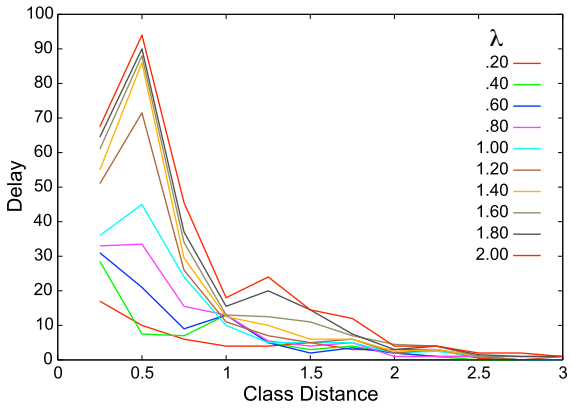


Fig. 5. Relation between the drift degree and detection delay in PHT

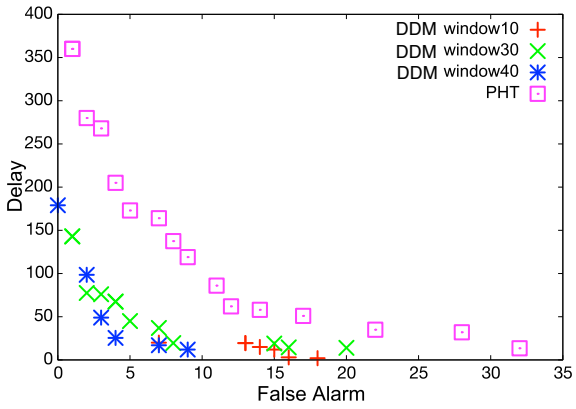


Fig. 6. Pareto solutions of the detection delay (median) and the number of false detection ($d = 0.5$)

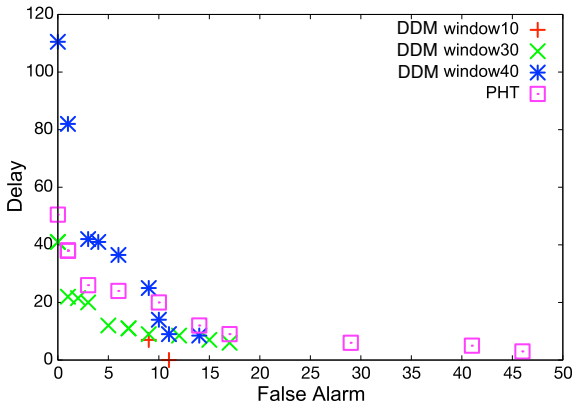


Fig. 7. Pareto solutions of the detection delay (median) and the number of false detection ($d = 1.0$)

proposed method works as a concept drift detection method for unsupervised clustering when the drift is sufficiently large.

Furthermore, the delay increases after distance 1.0 for both DDM (Fig. 4) and PHT (Fig. 5), in which distance 1.0 is equal to 1σ of the normal distribution in the synthetic data. For values larger than 1σ, though smaller detection delays can be obtained depending on the parameter, there is a trade-off between delay and false detection, as presented in the next section.

D. Experiment 2. Effect of Window Size

In the second experiment, we varied the window size of DDM to evaluate the effect window size would have on detection performance. Window size were varied as 10, 30, and 40. The distance between classes were varied as 0.5, 1.0, and 1.5 since the detection delays changed around 1σ in Experiment 1. Furthermore, we tested 50 trials for each distance.

Medians of detection delay and the number of false detection among 50 trials are shown in Figs. 6, 7, and 8. Here, false detection is considered when an algorithm detects drift before the predefined correct drift of data number 51. Since there is a trade-off between the detection delay and false detection, Pareto solutions were calculated in the graphs; therefore, the closer to the origin of the graph, the better performance of the algorithm.

From these results, we first confirm that DDM performed better than PHT for all distances when an appropriate window size was set. Furthermore, when a small window size was used ($w = 10$), a lower variety of Pareto solutions were obtained. This variety of Pareto solutions indicates that we have an applicable range for balance between detection delay and false detection.

Next, in DDM, detection delay and false detection depend on window size w and detection parameter α. When the distance between classes is small (Fig. 6), a larger window size is better as the Pareto solutions are closer to the origin. When the distance is large (Fig. 8), performance does not depend on window size since they are approximately equal when adjusting the detection parameter.

Standard deviation s_i becomes stable and asymptotically close to the true value when a larger window size is used; however, since sensitivity for error and for immediate response are trade-offs, the detection delay increases with the larger window size. Therefore, when the distance between classes is small, i.e., a smaller drift, it is efficient to use the larger window size to reduce false detection and adjust the detection parameter to balance the detection delay and false detection.

V. Future Work

In this work, we validated the statistical concept drift detection method with clustering as a first step; however, other detection methods are also available that use the same framework, including an early drift detection method (EDDM) [7], an exponentially weighted moving average (EWMA) [15], and a statistical test of equal proportions to detect concept drift (STPED) [8], all of which should be investigated. In addition, though only sudden drift was tested in this study, other drift types, including gradual and incremental drifts, should be tested. Moreover, the presented algorithms must be evaluated on real-world data with non-Gaussian distributions.

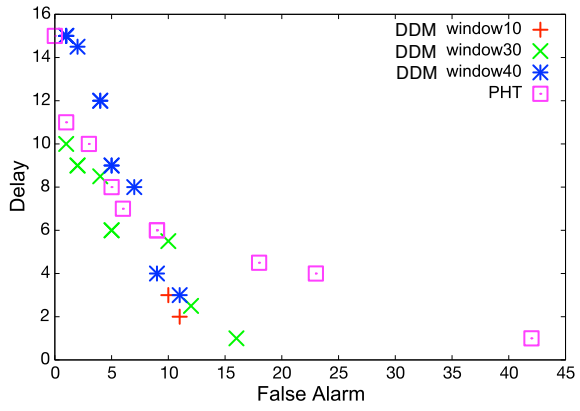


Fig. 8. Pareto solutions of the detection delay (median) and the number of false detection ($d = 1.5$)

VI. Conclusion

In this paper, we proposed a concept drift detection method for unsupervised clustering utilizing two kinds of statistical change detection methods, namely DDM and PHT. Our experiments using synthetic data revealed that DDM performed better than PHT in terms of detection delay and false detection as long as the appropriate window size was used. Moreover, the smaller the degree of drift, the greater effect the window size had on DDM. Since detection delay and false detection are trade-offs, in practice, using a larger window size and adjusting detection parameter to balance delay and false detection is an efficient approach.

Acknowledgments

This work was partially supported by the JGFoS Connect Follow-Up program at Alexander von Humboldt-Foundation, by the internship program at the Graduate School of Information Science and Technology, Osaka University, and by the overseas dispatch program of the president discretion expenses at Osaka University. Also thanks to the European Commission through the project MAESTRA (Grant number ICT-2013-612944).

References

[1] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, pp. 44:1–44:37, 2014.

[2] K. Fukui, S. Akasaki, K. Sato, J. Mizusaki, K. Moriyama, S. Kurihara, and M. Numao, "Visualization of damage progress in solid oxide fuel cells," *Journal of Environment and Engineering*, vol. 6, no. 3, pp. 499–511, 2011.

[3] I. Zliobaite, "Learning under concept drift: an overview," http://arxiv.org/abs/1010.4784, 2010.

[4] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. D. Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Computing Surveys (CSUR)*, vol. 46, no. 1, pp. 1–31, 2013.

[5] D. Kifer, S. Ben-David, and J. Gehrke, "Detecting change in data streams," in *Proceedings of the 30th Very Large Data Bases Conference (VLDB)*, 2004, pp. 180–191.

[6] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Proc. Brazilian Symposium on Artificial Intelligence*, 2004, pp. 286–295.

[7] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno, "Early drift detection method," in *Proc. ECML/PKDD 2006 Workshop on Knowledge Discovery form Streams*, 2006, pp. 77–86.

[8] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," in *Proc. 10th International Conference on Discovery Science (DS-10)*, 2007, pp. 264–269.

[9] G. Cabanes and Y. Bennani, "Change detection in data streams through unsupervised learning," in *Proc. The 2012 International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1–6.

[10] R. M. Vallim, J. A. A. Filho, R. F. de Mello, A. C. P. L. F. de Carvalhol, and J. Gama, "Unsupervised density-based behavior change detection in data streams," *Intelligent Data Analysis*, vol. 18, no. 2, pp. 181–201, 2014.

[11] H. Mouss, D. Mouss, N. Mouss, and L. Sefouhi, "Test of page-hinckley, an approach for fault detection in an agro-alimentary production system," in *5th Asian Control Conference*, vol. 2, 2004, pp. 815–818.

[12] H.-J. Appelrath, D. Geesen, M. Grawunder, T. Michelsen, and D. Nicklas, "Data stream clustering: Odysseus: a highly customizable framework for creating efficient event stream management systems," in *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, 2012, pp. 367–368.

[13] D. Geesen, M. Brell, M. Grawunder, D. Nicklas, and H.-J. Appelrath, "Data stream management in the AAL: Universal and flexible preprocessing of continuous sensor data," in *Proceedings of 5th AAL Kongress*, 2012, pp. 213–228.

[14] D. Hinkley, "Inference about the change point in a sequence of random variables," *Biometrika*, vol. 57, no. 1, pp. 1–17, 1969.

[15] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters*, vol. 33, pp. 191–198, 2012.