A Supervised Approach for Change Detection in Data Streams

A. Bondu, M. Boullé

***Abstract*— In recent years, the amount of data to process has increased in many application areas such as network monitoring, web click and sensor data analysis. Data stream mining answers to the challenge of massive data processing, this paradigm allows for treating pieces of data on the fly and overcomes exhaustive data storage. The detection of changes in a data stream distribution is an important issue which application area is wide. In this article, change detection problem is turned into a supervised learning task. We chose to exploit the supervised discretization method "MODL" given its interesting properties. Our approach is favorably compared with an alternative method on artificial data streams, and is applied on real data streams.**

I. Introduction

The amount of data to process has increased in many application areas such as network monitoring, web click and sensor data analysis. Conventional learning approaches hardly scale on such amounts of data. Data streams analysis constitutes a possible answer to the processing of massive data sets. The paradigm of "Data streams" takes into account several constrains : i) the order of tuples [1] is not controlled ; ii) the emission rates of data streams are not controlled ; iii) the available hardware resources are limited (i.e. RAM and CPU). The tuples cannot be exhaustively stored considering these constrains : on the fly algorithms are required. A data stream analysis provides a result that continuously changes over time. The objective is to maximize the quality of the result, given the constrains imposed by the input streams, and considering the available hardware resources. An important issue in data streams processing is the change detection in the underlying distribution of tuples. The application area of change detection is generally related to the monitoring of systems :

- In the **industry** field, the observed system can be a production equipment provided with sensors. In this case, change detection allows early detection of dysfunctions, and ultimately the maintenance of the equipment can be improved.
- In the **computer science** field, the observed system can be the whole data transmitted over a network. Changes in distribution provide a support to the detection of possible attacks.
- In the **martketing** field, the observed system can be related to a service used by customers. The detection of emerging behaviors gives relevant indications to improve the service.

The designing of general, scalable and statistically relevant change detection methods is a great challenge. Change detection consists in comparing the underlying distribution of tuples observed at different times. Two temporal windows are defined : the "reference" and the "current". An overview of the main change detection approaches is given by A. Dries [1] : Change detection in the distribution of tuples can be considered as a statistical hypothesis test which involves two samples of multidimensional tuples. Such problems are studied in the statistical literature. The Wald-Wolfowitz and Smirnov tests was generalized to multidimensional data sets in [2]. Later, approaches based on nearest-neighbor analyses [3] or distance between density estimates [4, 5] have been developed. Most recently, statistics based on maximum mean discrepancy for universal kernels have become popular [6]. A range of statistical work on abrupt change detection have been done [7, 8] .

In this article, Section II presents an original approach which turns the change detection into a supervised learning problem. Our approach is favorably compared with an alternative method on artificial data streams, in a first experimental validation carried out in Section III. In Section IV, the proposed approach is applied on a real data stream emitted from an industrial equipment provided with sensors. At last, perspectives and future works are discussed in Section V.

II. A supervised approach for change detection

Generally, change detection methods handle two temporal windows [2] defined by an expert (*see* step 1 of Figure 1) : i) the *reference* window represents the normal operation of the observed system ; ii) the *current* window characterizes the present state of the system. Two types of detection can be distinguished : a) the detection of changes from a normal operation which involves a fixed reference window ; b) the detection of ruptures which involves a sliding reference window. In this article, we aim at detecting changes from a normal operation. The reference window is **fixed** and the current window **slides** over time. Defining these two windows is the only one required adjustment from the user.

Then, tuples are labeled according to their window belonging (*see* step 2 of Figure 1). The tuples belonging to the reference window [*respectively* to the current window]

Alexis Bondu is with EDF R&D, 1 avenue du Général de Gaulle 92140 Clamart France. http ://alexisbondu.free.fr

Marc Boullé is with Orange Labs, 2 avenue Pierre Marzin, 22300 Lannion France. http ://perso.rd.francetelecom.fr/boulle/

1. A tuple is an elementary piece of data that is emitted in a data stream.

2. A temporal window is defined by a "start date" and a "end date", and includes the tuples emitted during this time interval.

are labelled by the class '0'" [*respectively* by the class "+"]. The quality of the classifier quantifies the change in distribution. The following situations give the intuition of our approach :

1) Assuming the distribution of tuples has not changed between the two windows, classes are completely "mixed". In this case, any robust classifier is not able to discriminate the two classes.
2) Assuming the distribution of tuples has changed, the tuples of class "0" and "+" do not have the same distribution in the space $\mathbb{R}^K$. In this case, a relevant classifier is able to discriminate the classes.

The potentially high rate of the input data stream entails processing tuples as quickly as possible. That is the reason why we have chosen to simplify the initial learning task into K univariate classification problems. The tuples are projected on each variable (*see* step 3 of Figure 1) and several classifiers are independently trained at the same time. While this simplification may seem drastic, this methodological choice provides a pragmatic response to the curse of dimensionality. In practice, few cases of change in distribution cannot be detected by examining each variable individually. The "Xor" problem, that is a textbook case, seems to be the only one change that cannot be detected.

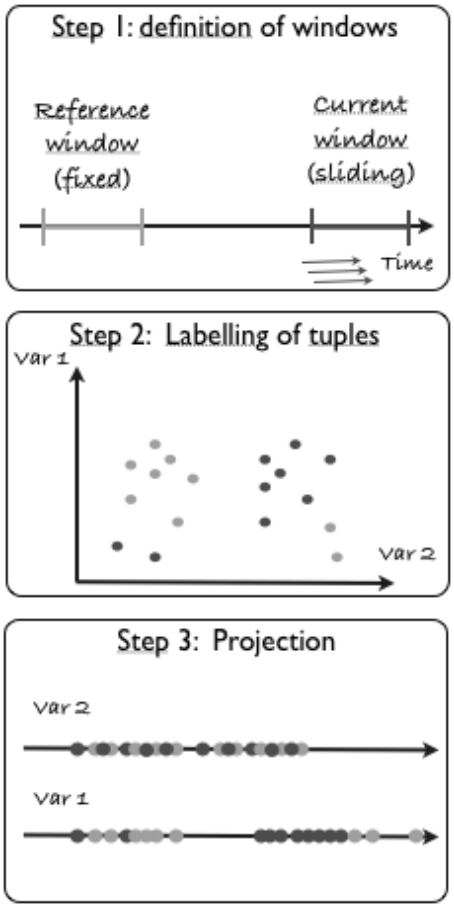

Fig. 1. Change detection as a supervised learning problems.

The choice of the classification method is critical and must meet the following criteria :

- ✓ the classifier needs to estimate the conditional density of the classes ;
- ✓ the classification method does not require prior knowledge ;
- ✓ the method does not involve user parameters to be adjusted ;
- ✓ the classifier needs to be resilient to outliers ;
- ✓ the method must be regularized in order to avoid over-fitting ;

Given these criteria, we chose to exploit the MODL supervised discretization method [9] that has been distinguished during challenges [10] :

- The assumptions on the distribution of tuples are low informative. It is a non-parametric approach [11] : the number of modeling parameters increases with the number of learning examples. In other words, this approach is able to estimate any distribution provided that the number of training examples is large enough.
- There is no user parameter to be adjusted.
- This classification method is based on rank statistics, that makes it insensitive to monotonic transformations of data. Difficulties related to data streams normalization and outliers [12] are solved elegantly.
- This classification method is regularized through a Bayesian approach. A compromise is naturally reached between the *complexity* of decision rules and their *generalization* ability.

A. MODL : a supervised discretization approach

The supervised discretization of a continuous variable consists in estimating the conditional distribution of classes owing to a piecewise constant estimator. The MODL approach [9] turns the discretization into a model selection problem. First, a family of discretization models is defined. The parameters of a discretization model are the following : the number of intervals, the bounds of intervals and the number of examples belonging to each class into each interval. A prior distribution $P(M)$ is defined over the family of models. This prior exploits the hierarchy of the model parameters : the number of intervals is first defined, then the bounds location and last, the conditional distribution are described in each interval. All possible values of model parameters are considered as equiprobable at each level of the hierarchy. A Bayesian approach is applied to select the best model, that is defined by maximizing the probability $P(M|D)$ of the model M given the data D. Exploiting the Bayes rule, and since the probability $P(D)$ is constant under varying the model, this amounts to maximizing $P(M)P(D|M)$. The number of training examples N and the number of classes

J are given by the classification problem to be solved. A discretization model is defined by the following parameters : $\left\{I, \{N_i\}_{1\leq i\leq I}, \{N_{ij}\}_{1\leq i\leq I, 1\leq j\leq J}\right\}$, with I the number of intervals, N_i the number of examples located in the interval i, and N_{ij} the number of examples labelled by the class j located in the interval i. The prior distribution $P(M)$ and the likelihood $P(D|M)$ can be analytically calculated by exploiting the definition of the family of models and the hierarchical prior distribution : combinatorics technics are used. By applying the negative logarithm on the term $P(M)P(D|M)$, we obtain a criterion to minimize :

$$\mathcal{C}(M) = \underbrace{\log N + \log \binom{N+I-1}{I-1} + \sum_{i=1} \log \binom{N_i+J-1}{J-1}}_{-\log P(M)} + \underbrace{\sum_{i=1} \log \frac{N_i!}{N_{i1}!N_{i2}!\ldots N_{iJ}!}}_{-\log P(D|M)} \quad (1)$$

The optimization of this criterion defines the most probable model given the data, also called $\mathcal{M}ap$ *(Maximum A Posteriori)*. The first term of the criterion $\mathcal{C}$ corresponds to the choice of the number of intervals, and the second term corresponds to bounds location. The third term represents the choice of the conditional distribution in each interval. These three first terms penalize complex models including a large number of intervals. The last term corresponds to the likelihood of data given the model. This term favors complex models with a good fit of the conditional distribution. This approach is intrinsically regularized, a compromise is naturally reached between the complexity of the models and their generalization ability. Numerous comparative experiments indicate that this classification approach provides very good results in practice and avoids over-fitting [9].

B. Detection and diagnostic

The criterion $\mathcal{C}_k(M) = -\log[P(M|D_k)]$ corresponds to the probability that a discretization model M explains the type of stream regime *(reference or current)* given the data D_k, characterized by the explicative variable k. The negative logarithm of a probability represents a coding length in information theory [13]. This criterion can be interpreted as the ability of a discretization model to encode the type of stream regime given the data D_k. Let $M_\emptyset$ be the null model that discretizes the variable k into a single interval. $M_\emptyset$ represents the length coding of the type of stream regime without exploiting the variable k. In the case where the distribution of k does not change between the reference and the current window, it is impossible to discriminate the type of stream regime exploiting this variable : the null model is the best one. By contrast, if the distribution of k has significantly changed, this variable can be exploited to detect the type of stream regime : there is at least one discretization model that is more probable than the null model, which corresponds to a more compact length coding than $M_\emptyset$. The compression gain [14] is defined as follows :

$$\mathcal{G}ain_k(M) = 1 - C_k(M)/C_k(M_\emptyset)$$

Our approach exploits $\mathcal{G}ain_k(\mathcal{M}ap)$, the compression gain of the most probable model given the data. $\mathcal{G}ain_k(\mathcal{M}ap)$ is equal to 0 when the variable k does not allow to discriminate the type of stream regime. This value is strictly positive when there is a significant difference in the distribution of k, depending on the type of stream regime *(detectable by a discretization model)*. $\mathcal{G}ain_k(\mathcal{M}ap)$ is equal to 1 when the variable k allows to perfectly discriminate the two types of stream regime. This method is correctly regularized, which allows us to exploit $\mathcal{G}ain_k(\mathcal{M}ap)$ as a quality indicator of the discretization model. By this way, we avoid the assessment of $\mathcal{M}ap$ based on a test set and an evaluation criterion *(such as the AUC, the MSE or the BER ... etc)*. All tuples belonging to the temporal windows are used during the learning phase, we hope to obtain a better discretization model.

quantifying the change in distribution

In this paragraph, the proposed criterion aims at globally quantifying the change in distribution. The change is independently characterized on each variable $k \in K$ by $\mathcal{G}ain_k(\mathcal{M}ap)$. The following aggregating criterion is then defined :

$$\mathcal{C}hange = \frac{1}{K} \sum_{k\in[1,K]} \mathcal{G}ain_k(\mathcal{M}ap) \quad (2)$$

contribution of each variable

The sum of the contributions $\mathcal{C}ontrib(k)$ over all variables corresponds to the change evaluated by the criterion $\mathcal{C}hange$. The contribution of the variable k is defined as follows :

$$\mathcal{C}ontrib(k) = \frac{\mathcal{G}ain_k(\mathcal{M}ap)}{K}$$

III. Experimental validation on artificial data streams

This experimental validation involves two artificial data streams that share the same temporal structure. Every second, one tuple is drawn from an underlying distribution which changes over time. Figure 2 shows how the underlying distribution evolves. The first 2000 tuples are emitted from the *"initial distribution"* which represents the normal operation. The reference window is set up at this moment. Between 4000 and 6000 seconds, the underlying distribution progressively [3] moves from the *"initial"* to the *"modified"* distribution. Then 2000 tuples are emitted from the *"modified"* distribution. Between 8000 and 10000 seconds, the underlying distribution progressively returns to its *"initial"* state. At last, 2000 tuples are emitted from the *"initial"* distribution. In our experiments tuples are defined in $\mathbb{R}^2$. The

3. During transitions, the probability that a tuple is drawn from the "initial" or the "modified" distribution linearly varies over time.

"initial" and *"modified"* distributions are defined on Table I for both artificial data streams. These normal distributions are denoted by $\mathcal{N}(m, v)$, where m is a two-dimensional vector corresponding to the mean and v is the covariance matrix.

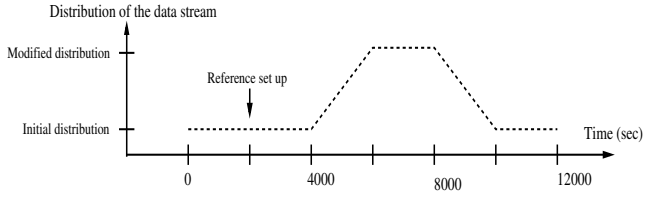

Fig. 2. Temporal structure of both artificial data streams.

	initial distribution	modified distribution
Data stream 1 :	$\mathcal{N}\left(\begin{smallmatrix}0\\0\end{smallmatrix}, \begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)$	$\mathcal{N}\left(\begin{smallmatrix}4\\8\end{smallmatrix}, \begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)$
Data stream 2 :	$\mathcal{N}\left(\begin{smallmatrix}0\\0\end{smallmatrix}, \begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)$	$\mathcal{N}\left(\begin{smallmatrix}0\\0\end{smallmatrix}, \begin{smallmatrix}4&0\\0&9\end{smallmatrix}\right)$

TABLE I
DEFINITION OF "INITIAL" AND "MODIFIED" DISTRIBUTIONS FOR BOTH ARTIFICIAL DATA STREAMS.

A. Experimental protocol

Defining the temporal windows constitutes the only one adjustment required by our change detection approach. As part of our experiments, the reference window is fixed and includes the first 2000 tuples of the data stream. These tuples are assumed to be representative of the normal operation of the observed system. The reference window is sliding [4] and includes, at every moment, the last 300 emitted tuples.

B. Alternative approach

Our detection strategy is compared with an alternative approach [5] that involves four successive steps :

1) data stream summary using the micro-clustering method "DenStream" based on a time-weighting of tuples [15] ;
2) current distribution estimate owing to a modified version of Parzen window, that handles micro-clusters instead of tuples ;
3) comparison of the current distribution with the reference distribution using the Kullback-Leibler divergence ;
4) assessment of variables contributions by a criterion that exploits the Kullback-Leibler divergence into a subspace.

In contrast with our approach, the alternative method requires the adjustment of several user parameters. The parameters related to the summary of the stream are the following : i) the fading parameter of a time-weighting function ; ii) the minimum weight of micro-clusters ; iii) the maximum variance allowed into micro-clusters. The estimate of the distribution exploits a modified Parzen window estimator that requires the tuning of a smoothing parameter δ (see Equation 3).

$$\hat{P}^*(x) = \frac{1}{C.W} \sum_{j=1}^{C} \frac{\omega_j}{\sqrt{2\pi \left(\delta^2 + r_j^2\right)^k}} \exp^{-\frac{d(x,c_j)^2}{2\left(\delta^2 + r_j^2\right)}} \quad (3)$$

Notation : W is the total weight of the data stream, C is the number of micro-clusters summarizing the stream, ω_j is the weight of the *jth* micro-cluster, c_j is the barycenter of the tuples belonging to the *jth* micro-cluster, r_j is the standard deviation of tuples belonging to the *jth* micro-cluster, δ is a smoothing parameter of the distribution estimate.

C. Comparative results

In this section we compare our change detection approach with the alternative method [5] presented in Section III-B. In the case of the alternative method, the presented results come from the original article [5]. Both approaches provide the same outputs : an overall change indicator and the contributions of each variable. Used indicators are describe in Sections III-B and II.

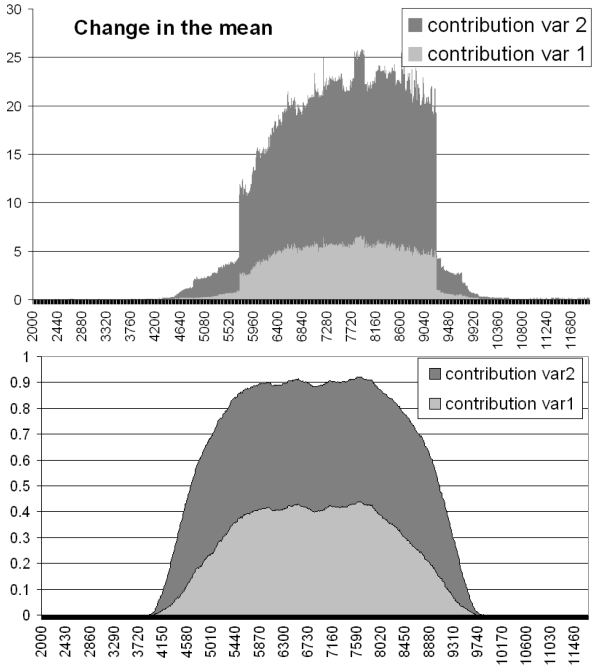

Fig. 3. Comparative results on the first data stream, where a change of mean occurs. The vertical axis corresponds to the detected changes, and the horizontal axis represents the order of emitted tuples.

Both detection approaches are evaluated regarding three criteria : i) the earliness of the detection ; ii) the sharpness of the detection ; iii) the ability to assess the contributions. Figure 3 presents the results on the first data stream, where a change of mean occurs. The top chart corresponds to the alternative method and the bottom chart to our approach. In both cases, the horizontal axis represents the number

4. In order to reduce the computational time of our experiments, we chose to carry out a measurement point at every ten tuples.

of emitted tuples from the beginning of the experiment. The vertical axis characterizes the level of the detected change, either by the Kullback-Leibler (on the top) or by the $\mathcal{C}hange$ criterion (on the bottom). On each chart, the sum of contributions represents the overall level of detection. As the Figure 3 shows, our approach begins to detect changes very early (from 4100 tuples, against 4500 tuples for the alternative method). Secondly, our approach dominates the alternative method in terms of sharpness detection : the curves are less noisy. Between 6000 and 8000 tuples, both contributions have substantially the same value and are nearly constant. During this period of time, both classes are perfectly discriminated for each variable. The two variables provide equivalent quantities of information relatively to the discrimination of classes.

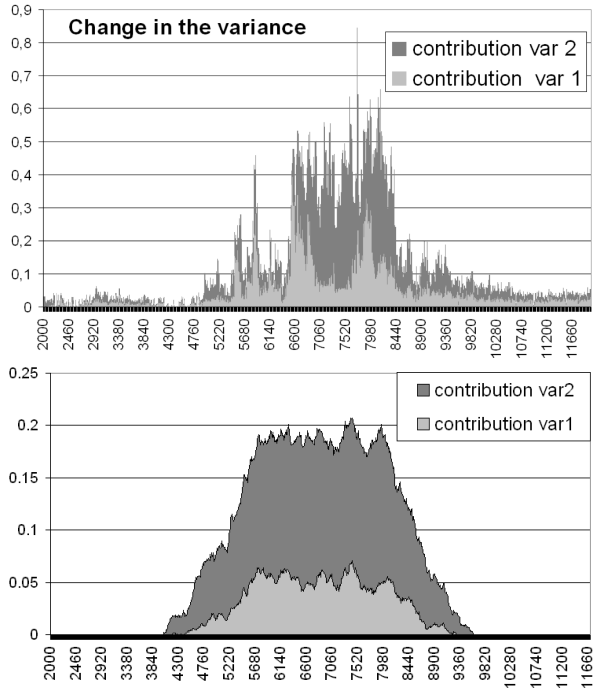

Fig. 4. Comparative results on the second data stream, where a change of variance occurs. The vertical axis corresponds to the detected changes, and the horizontal axis represents the order of emitted tuples.

As previously, Figure 4 presents the results of both change detection methods on the second data stream, where a change of variance occurs. Once again, our approach detects changes very early (from 4050 tuples against 4800 for the alternative method), and the sharpness of detection curves provided by our approach is much better. In contrast with the previous data stream, the threshold effect on the contributions does not appear here : classes are more difficult to discriminate on this data stream. To conclude, the comparative experiments presented in this section show that our approach dominates the alternative method. Moreover, our approach is easier to implement because it does not involves user parameters.

IV. Experiments on real data streams

This section presents experiments realized on real data streams. We show only the results obtained by our approach, because we were not able to operate the alternative method. Tuples are not normalized and there are four explicative variables, which makes the step of micro-clustering difficult to carry out. In practice, the required user parameters are very difficult to tune, especially in an unpredictable environment. This makes the alternative method unsuitable for real data streams.

The considered real data streams are generated by four sensors on an industrial equipment. Tuples are emitted at regular intervals and include the average values of sensor measurements on this time period. These experiments involve three real data streams with malfunctions actually observed on the equipment. The three charts on the top of Figures 5, 6 and 7 represent the sensor measurements *(vertical axis)*, according to the order of tuples *(horizontal axis)*. For each data stream, the reference window is defined by an expert and represents the normal operation. The ends of reference windows are symbolized by a vertical line on the charts. Reference windows include the first 400, 1000 and 1000 emitted tuples, respectively for the data streams No. 1, No. 2 and No. 3. For all data streams, the current window includes the last 150 emitted tuples. This choice is motivated by an applicative constraint : the width of the current window determines the latency before the first measurement.

Results on the first data stream (Figure 5) :

The malfunction that occurs in the first data stream is a simultaneous drift of all sensors in favor of higher values. This is a progressive drift and the variance of sensor measurements tends to increase over time. As the bottom chart shows, our approach early detects the drift. Between the 550th and the 1000th tuple, the contributions of each variable progressively increase and have nearly the same values *(the contribution of the fourth variable is slightly higher than the others)*. After the 1000th emitted tuple, the contributions are almost constant, that means that the classes are perfectly discriminated for each variable. This first application gives consistent results and contributes to validate our approach. However, this type of malfunction is easily detectable by observing original data and does not underline the practical benefits of our method.

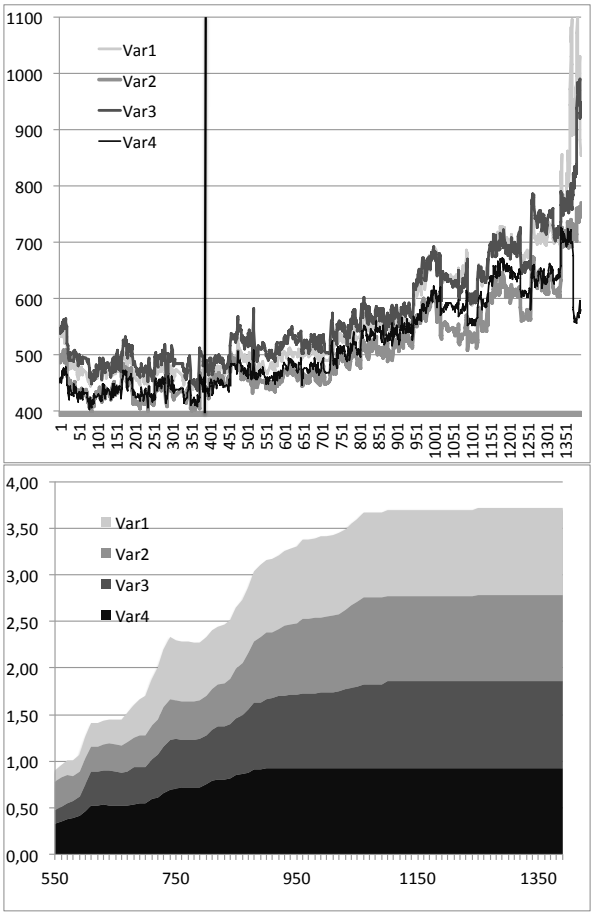

Fig. 5. Results on the first real data stream. The top chart plots the values of explicative variables over time. The bottom chart plots the detected change and the contributions of each variable.

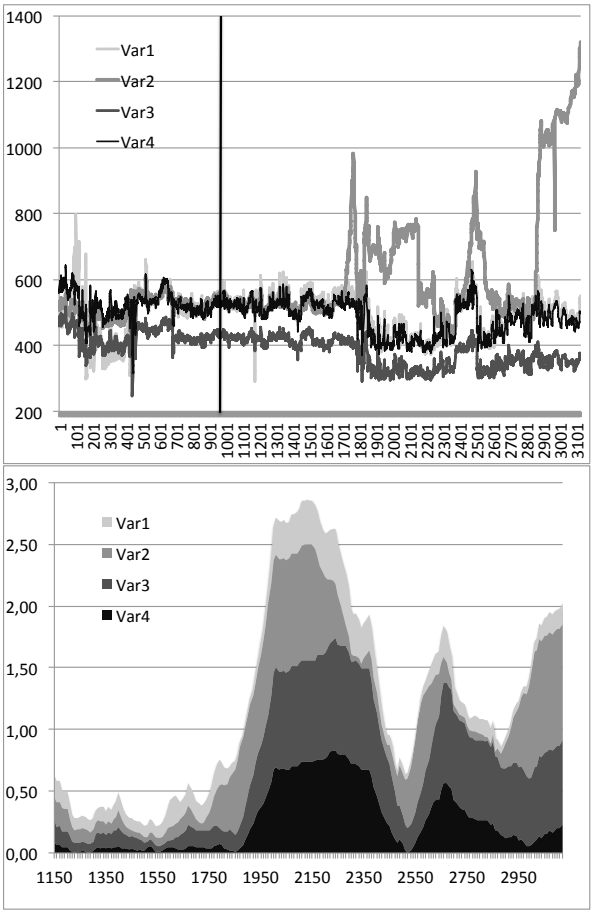

Fig. 6. Results on the second real data stream. The top chart plots the values of explicative variables over time. The bottom chart plots the detected change and the contributions of each variable.

Results on the second data stream (Figure 6) :

The malfunction in the second data stream is more difficult to describe than the previous one. The fluctuations of the first variable seem to be low. The second variable drifts chaotically to higher values. Variables 3 et 4 present two ruptures in favor of lower values *(the intervals [1750-2350] and [2500-3000] on the horizontal axis)*. The bottom chart presents the results of our change detection method. The level of detection fluctuates over time with a peak between 1850 and 2550 tuples. This chart shows that the variable 3 most often contributes to the detected changes, which is difficult to appreciate by observing the original data. The readability of this kind of chart constitutes the main added value of our approach.

Results on the third data stream (Figure 7) :

The malfunction in the third data stream mainly occurs on variables 3 and 4. From the beginning of the monitoring process, the variable 3 presents a rupture in favor of higher values. A malfunction appears on this variable when the 2300th tuple is emitted : this is a drift of sensor measurements to higher values. Our approach captures these changes in a proper way. Once again, our approach provides a synthetic result that is easy to interpret *(bottom chart)*.

To conclude, the experiments presented in this section illustrate the practical interest of our approach applied to the monitoring of operating equipments. Our method provides a synthetic view of detected changes in near real time. In some application cases, this information can be exploited to carry out a preventive maintenance.

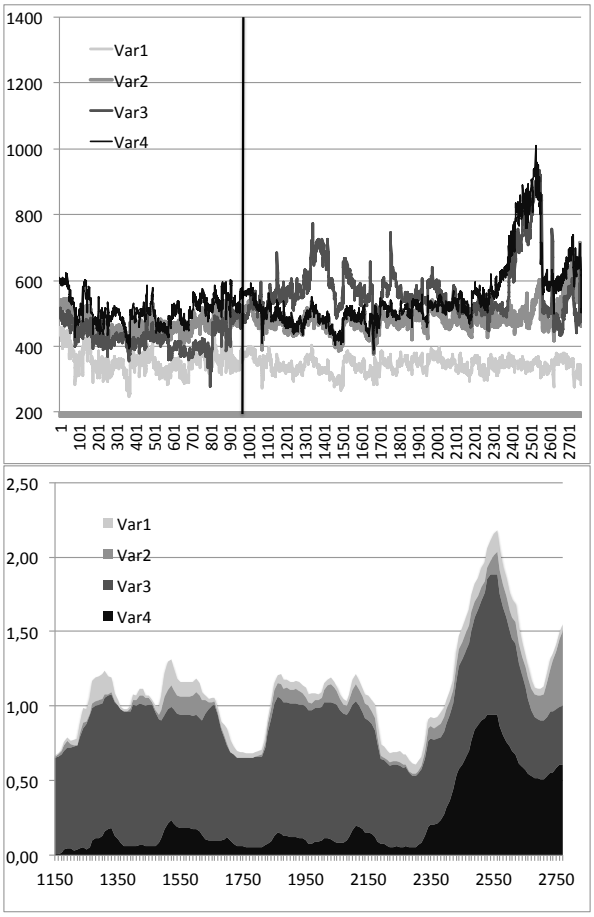

Fig. 7. Results on the third real data stream. The top chart plots the values of explicative variables over time. The bottom chart plots the detected change and the contributions of each variable.

V. Conclusion and perspectives

This article proposes to turn the change detection into a supervised classification problem. Two temporal windows are defined on the input data stream : i) the reference window represents the normal operation of the observed system ; ii) the current window characterizes the present state of the system. The quality of the classifier trained on this data quantifies the change in the distribution observed between the two windows. We chose to exploit the MODL supervised discretization method [9] given its interesting statistical properties.

Our approach implies as many univariate classifiers as the input space includes variables. This is a pragmatic modeling choice that simply answers to the curse of dimensionality [16]. According to our point of view, the important algorithmic cost of a multivariate analysis is not justified, given the few cases of change that cannot be detected by examining each variable individually.

In Section III, our approach is favorably compared with an alternative method [5] on artificial data streams. Our approach is distinguished by its ability to early detect the changes, and by the sharpness of its detections. Our approach is also applied on real data streams in Section IV, these experiments highlight the practical interest of our approach for monitoring industrial equipments.

Hardware constrains *(ex : available RAM, CPU)* and applicative constrains *(ex : tolerated detection delay)* could be explicitly taken into account by our approach. This future improvement will consist in automatically tuning the size of the reference window and the rate of the computation of the change detection indicators. At last, an incremental version of the algorithm used to optimize the criterion $\mathcal{C}(M)$ could be proposed.

Références

[1] A. Dries and U. Rückert. Adaptive Concept Drift Detection. In *SIAM Conference on Data Mining*, pages 233–244, 2009.

[2] J.H. Friedman and L.C Rafsky. Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests. *Annals of Statistic*, 7(4) :697–717, 2006.

[3] P. Hall. Permutation tests for equality of distributions in high-dimensional settings. *Biometrika*, 89(2) :359–374, June 2002.

[4] N. H. Anderson, P. Hall, and D. M. Titterington. Two-sample test statistics for measuring discrepancies between two multivariate probability density functions using kernel-based density estimates. *Journal of Multivariate Analysis*, 50(1) :41–54, 1994.

[5] A. Bondu, B. Grossin, and ML Picard. Density estimation on data streams : an application to Change Detection. In *EGC (Extraction et Gestion de l'Information)*, 2010.

[6] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and Smola A. J. A Kernel Method for the Two-Sample-Problem. In *NIPS*, pages 513–520. MIT Press, 2006.

[7] M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes : Theory and Application*. Prentice-Hall, November 1993.

[8] F. Desobry and M. Davy. Support Vector-Based Online Detection of Abrupt Changes. In *Proc. IEEE ICASSP, Hong Kong*, pages 872–875, 2003.

[9] M. Boullé. MODL : A bayes optimal discretization method for continuous attributes. *Machine Learning*, 65(1) :131–165, 2006.

[10] I. Guyon, A.R. Saffari, G. Dror, and J.M. Bumann. Performance prediction challenge. In *International Joint Conference on Neural Networks*, pages 2958–2965, 2006. http ://www.modelselect.inf.ethz.ch/index.php.

[11] C.P. Robert. *The Bayesian Choice : From Decision Theoretic Foundations to Computational Implementation*. Springer, 2007.

[12] E. Ogasawara, LC. Martinez, D. Oliveira, G. Zimbrao, GL. Pappa, and M. Mattoso. Adaptive Normalization : A Novel Data Normalization Approach for Non-Stationary Time Series. In *IJCNN (International Joint Conference on Neural Network)*, Barcelonne, 2010.

[13] C.E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3) :379–423, 1948.

[14] M. Boullé. Optimum simultaneous discretization with data grid models in supervised classification : a bayesian model selection approach. *Advances in Data Analysis and Classification*, 3(1) :39–61, 2009.

[15] F. Feng Cao, M. Ester, W. Qian, and A. Zhou. Density-based clustering over an evolving data stream with noise. In *SIAM Conference on Data Mining*, pages 328–339, 2006.

[16] R.E. Bellman. *Adaptive Control Processes*. Princeton University Press, Princeton, NJ, USA, 1961.