Computers in Industry 83 (2016) 78–91

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

Reconfiguration process for neuronal classification models: Application to a quality monitoring problem

Mélanie Noyel[a,b], Philippe Thomas[a,*], André Thomas[a], Patrick Charpentier[a]

[a] Université de Lorraine, CRAN, UMR 7039 CNRS, Campus Sciences, BP 70239, 54506 Vandœuvre-lès-Nancy cedex, France
[b] ACTA Mobilier Parc d'activité Macherin Auxerre Nord, 89470 Monéteau, France

ARTICLE INFO

Article history:
Received 4 May 2016
Received in revised form 14 September 2016
Accepted 21 September 2016
Available online xxx

Keywords:
Classification
Control chart
Learning
Multilayer perceptron
Neural network
Quality
Relearning

ABSTRACT

In the context of smart industries, learning machines currently have various uses such as self-reconfiguration or self-quality improvement, which can be classification forecasting problems. In this case, learning machines are tools that facilitate the modeling of the physical system. Thus, it is obvious that the model must evolve with changes in the physical system, thereby leading to adaptability/reconfigurability problems. Among the various tools reported previously, real-time systems seem to be the best solution because they can evolve autonomously according to the behavior of the physical system. In the present study, we propose a method for using learning machines efficiently in an evolving context. This method is divided into two components: (1) model conception by defining the objective function and influential factors, setting up data collection, and learning using multilayer perceptrons; and (2) monitoring system conception with the aim of tracking the misclassification rate, determining whether the physical system is drifting, and reacting by model adaptation based on the control charts. This paper focuses on the model monitoring procedure because the model conception procedure is quite classical. The proposed method was applied to a benchmark derived from previous research and then to an industrial case of defect prevention on a robotic coating line for which other methods have proved unsuccessful.

1. Introduction

In the smart industry domain, the elements of the physical system share information with each other and with a real time decision-making system. These decision systems must evolve and adapt according to the physical state of the environment. For example, in the quality control domain, we may consider the impact of environmental factors (such as temperature and humidity) on a die-casting process [49] or a lacquering process [33], as well as the impact of tool wear on the finished surface in a machining process [45]. Other examples of real-world change detection problems include modeling in bio-medicine monitoring and industrial processes, as described by [4].

These different contexts lead to classification or supervised classification problems, which can be resolved using machine learning algorithms, where various techniques can be applied, including logic-based algorithms (such as decision trees and rule-based classifiers), neural networks (NNs such as multilayer perceptrons (MLPs) and radial basis function networks), statistical learning (such as naive Bayes classifiers and Bayesian networks), and support vector machines (SVMs). The present study focuses on classification problems where continuous data are mainly considered. In this case, Kotsiantis [22] has highlighted that the most suitable approaches are logic-based algorithms, NNs, and SVMs.

To extract a classifier from data using machine learning, the complete learning dataset is provided to the learning machine, which obtains descriptions for the underlying concepts in the dataset [24]. This type of learning is called batch learning. However, the target concept may be non-stationary and it can change over time, thereby leading to concept modification [29]. Gama [14] separated this conceptual evolution into concept drift when the evolution is gentle and concept shift when the evolution is sudden. For example, a concept drift may result from tool wear or filter clogging, whereas a concept shift may occur after the replacement of parts or a system modification. When a concept shift or concept drift occurs, the model obtained using batch learning may no longer be accurate. Therefore, incremental algorithms, online algorithms, and anytime algorithms have been

* Corresponding author.
E-mail addresses: mnoyel@acta-mobilier.fr (M. Noyel), philippe.thomas@univ-lorraine.fr (P. Thomas), andre.thomas@univ-lorraine.fr (A. Thomas), patrick.charpentier@univ-lorraine.fr (P. Charpentier).

http://dx.doi.org/10.1016/j.compind.2016.09.004
0166-3615/

proposed to respond to this problem. Incremental algorithms consider new data in order to adapt the model without restarting the complete learning process Salperwick et al., 2009. Online algorithms are used when the stream of data is continuous and the data are used individually and sequentially Salperwick et al., 2009. In these two approaches, the learning must be faster and the data are generally presented only once to the algorithm. Anytime learning is defined as learning the best model (considering a given criterion) until a break occurs (such as new data arrival) [9]. In these three approaches, the learning and exploitation of the model must be performed simultaneously, so the computational time required may be prohibitive in real-time applications, even if anytime approaches include resource constraints. These learning approaches may exhibit slower convergence than batch approaches, thereby leading to an inaccurate model. Moreover, because the data are used individually, outliers may have a deleterious impact on the model obtained. Finally, incremental learning must address the plasticity-stability dilemma [6], which demands a compromise between the stability and reconfigurability of the model.

Due to the limitations of the online approaches, batch learning is useful because it can be performed in real time whereas learning is performed offline. Batch learning can approximate every nonlinear function with the desired accuracy, as well as using a variety of algorithms to avoid bad local minima or the overfitting problem, and cross-validation can be performed based on the validation dataset. None of these processes can be performed with incremental learning, and Sarle [41] showed that it is generally more difficult and unreliable than batch learning. Considering the drift or shift concepts allows changes to be detected during batch learning, and change detection can be performed with different approaches Salperwick et al., 2009, as follows:

- Relearning the classifier from scratch,
- Adapting the classifier,
- Adapting the data summary used in the classifier (e.g., the kNN model),
- Using the sequence of classifiers that is learned over time in a classifier ensemble as example [5].

Using of classifier ensemble allows to improve the accuracy of the classification. However, many individual classifiers must be evaluated simultaneously and this fact may be time consuming during the exploitation phase of the model. Adapting the classifier by parameters relearning allows to limit the computational cost by considering that even if a drift occurs, the original model is not so far of the desired one. However, in case of major context change, the model structure itself must be corrected. In this case, the classifier must be relearned from scratch.

We propose an adaption of the classifier in order to limit the computational cost. We focus on the use of a MLP classification model due to its adaptability to change. In fact, the adaptation of a MLP classifier may be performed by a learning process based on the new dataset using the initial model parameters as the parameter set for initialization. It is assumed that even if a concept shift or concept drift occurs, the initial classifier is no more accurate but its parameter set will remain close to the optimal set.

The following different approaches may also be used for change detection [16]:

- Using sequential analysis (e.g., sequential probability ratio test [52], Page-Hinkley test (PHT) [35,18], and cumulative sum [35]);
- Using statistical process control (SPC) [21,15,6];
- Monitoring the distribution based on two different time windows [10,1];
- Contextual approaches [17,6].

Sequential analysis approaches are very sensitive to the choice of the detection threshold [38]. The main limitation of time window-based approaches is the possibility of high memory consumption [16]. Contextual approaches are used mainly in conjunction with incremental learning. Thus, in the present study, a SPC approach is used to determine when relearning is required. However, even if a change is detected and the need for relearning is evident, a question remains: What dataset should we use for relearning? PHT can estimate the time when a drift or shift concept starts [38]; thus, PHT may be used in conjunction with SPC to build the relearning dataset. Finally, industrial datasets are often affected by outliers so a robust learning algorithm is required in order to limit the impact of outliers on the classifier.

The main contribution of this paper is the proposition of a model monitoring procedure which associates SPC and PHT algorithms in order to adapt the model to change (by using relearning procedure). MLP is used as model and its adaptation is performed by using backpropagation algorithm. SPC is used to estimate the need of model's adaptation. PHT is used to determine the dataset which must be used during the relearning procedure.

In the following, after explaining the batch learning approach for designing the monitoring system, we discuss the impact of changing the context. The need for adaptation is highlighted and the proposed method is developed in two steps: determining when the system is finally drifting using control charts and evaluating how much data must be relearned. Finally, the proposed approach is tested on both a benchmark case and an industrial case, i.e., a quality monitoring problem for a lacquering company.

2. Description of the batch learning process

To predict the behavior of a real system, a classical approach involves the design of a forecasting model (Fig. 1). The behavior of this forecasting system, which is parallel to the physical one, is as similar as possible to that of the real system. It can measure different parameters in the physical system and compare its forecasts with reality. For classification problems, the considered forecasting system can predict the class of the output data; therefore, it can be used to evaluate the decision taken upstream.

The classical approach to the design of the forecasting model employs a learning machine in order to extract the forecasting model directly from the data using batch learning. This task is performed according to a classical knowledge discovery and data mining (KDD) process, which comprises two main steps: collecting the dataset (including identification and data collection and preprocessing) and the data mining task (which needs to define the training and validation datasets) [37]. This KDD process is summarized by Fig. 2 and the two main task will be more detailed in the two following subparts.

2.1. Dataset collection

To design the dataset collection, the first step is to define the objective function, which is the output of the forecasting model and it corresponds to the characteristics of the physical system that we aim to monitor.

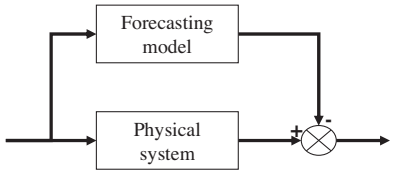

Fig. 1. Relationship between the forecasting model and physical system.