Journal of Manufacturing Systems 45 (2017) 97–108

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

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

Technical Paper

On-line self-adaptive framework for tailoring a neural-agent learning model addressing dynamic real-time scheduling problems

Zeineb Hammami [a,*], Wiem Mouelhi [b], Lamjed Ben Said [a]

[a] SMART Lab, Computer Science Department, University of Tunis, ISG, 41 Avenue of freedom, Bardo 2000, Tunis, Tunisia
[b] SMART Lab, Computer Science Department, University of Manouba, ISAMM, University Campus of Manouba, Manouba 2010, Tunisia

ARTICLE INFO

Article history:
Received 7 October 2016
Received in revised form 8 August 2017
Accepted 10 August 2017

Keywords:
Neural-agent learning model
Dynamic scheduling
Real time adaptation
Data stream
Concept drift

ABSTRACT

The dynamic nature and time-varying behavior of actual environments provide serious challenges for learning models. Thus, changes may deteriorate the constructed control policy over time, which requires permanent adaptation strategies. Changes usually appear as an evolution in the relationship between instance variables composing stream data, known in machine learning under the term *concept drift*. Several adaptation strategies have been performed to tackle concept drifting data streams, always assuming that arrived instances are labeled, either completely or partially. However, this assumption is violated in many application areas, especially in the manufacturing field. We propose, in this paper, a new framework called Labeling Extraction from the current Model (LEM). LEM is adapted to retrieve learning labels, relying uniquely on unlabeled received instances and without any external supervision, which has never been previously addressed. Hence, to the best of our knowledge, there has been no effort addressing scheduling manufacturing problems for adaptation to data streams with concept drifts. Experiments are conducted to show the effectiveness of LEM. The obtained results demonstrate the ability of LEM to maintain the stability and efficiency of the control policy approximated by the learning model, by significantly improving its prediction performance, compared to its use without adaptation.

1. Introduction

In today's actual production environments dominated by dynamic conditions [1], the need to quick and almost immediate decisions has become challenging [2]. Hence, advanced tools and methods have increasingly been required to create intelligent decision making systems.

Over the last decades, predictive models have attracted many researchers that have relied in a first generation on the off-line modeling based on recorded historical data [3]. Machine learning was the most commonly used approach in research and development contexts. The focus has traditionally been on using a set of learning data assumed to be sufficient and representative. Indeed, it is this assumption that paved the way for the development of elegant learning algorithms, working on-line in a better way [4]. Nevertheless, with the appearance of environmental changes, models constructed making such assumption might become outdated. In manufacturing, the causes of defects are usually uncontrollable, such as machining variance or change of raw material nature [5]. Such causes may lead to an evolution in the relationship between variables composing stream data, in response to changes in the underlying data distributions. In machine learning, the change is known as "concept drift".

Learning models are updated using newly-arrived streaming data, representing potential changes, and always assuming that the arrived instances are labeled (either completely or partially) [6]. However, this assumption is violated in many application areas, especially in the manufacturing field. Indeed, determining if a manufacturing decision made at a given time will be good or not on the long run, given all the subsequent decisions, is a major obstacle [7]. The task will become even more complicated when the decisions (labels) associated to incoming instances are required to be available on-line. According to whether the instance is labeled or not, we can note (X, y) or X, respectively, where $X=\{x_1, x_2, \ldots, x_n\}$ represents a vector of n input variables and y is the corresponding output variable or label. We try, in this approach, to collect and analyze the incoming instances for tracking eventual drifts concurrently with process monitoring. The instances arrive entirely unlabeled to the model. We introduce, in this paper, a new framework – called Labeling Extraction from the current Model (LEM) – to address such circumstances. Hence, the approach presents the first attempt to deal with concept drifting data streams for manufacturing production systems. We show the effectiveness of the approach through illustrative examples and the performed experiments.

1.1. The problem

Here, we address dynamic real time scheduling problems, where we seek to determine which allocation decision should be taken, when it should be taken, and for which resource. Therefore, we designate Y as a set of z candidate decisions to use for resources allocation, $Y=\{y_1, y_2, \ldots, y_z\}$. Examples of allocation decisions include selecting a given dispatching rule or assigning a given operator to a given machine. The objective is to select the most suitable decision among the available candidates as soon as a triggering event occurs so as to help optimizing the assigned performance criterion. Since the scheduling is dynamic, the performance is measured over a sufficiently long period of time, which corresponds to the period of study, using such classical

* Corresponding author.
E-mail address: zayneb.hammami@gmail.com (Z. Hammami).

http://dx.doi.org/10.1016/j.jmsy.2017.08.003
0278-6125/

measures as mean tardiness of jobs and mean flow time. The decision selection is based on the system state, characterized by the vector X related to the system variables that represent the system's dynamic during the period of study. Examples of state variables include the number of jobs in a given queue and the operations' slack times. Hence, the learning system tries to represent a relationship between the variables and to map states X to control decisions $y_l \in Y, (l \leq z)$ that optimize the objective function. The term concept refers to such a relationship that the learning system tries to model along a period of time.

When the target concept changes, adapting to occurring drifts will be immediately required. Therefore, the problem to be addressed is the possibility of updating the current learning model in real time, so as to maintain the right sequence of decisions during the production system's period of study.

1.2. Purpose

The aim of the LEM algorithm application is to help maintain the effectiveness of decisions regarding the expectation of the performance function. The algorithm is based on the current model, and relies uniquely on unlabeled received instances, without any external supervision, to extract labels and adjust training data when adaptation is necessary.

1.3. Organization

The paper is organized as follows. We present related studies on concept drifting data streams in Section 2. Subsequently, we present a brief description of the learning model we constructed off-line, which forms the subject of earlier work in Section 3, followed by a description of our proposed LEM algorithm in Section 4. Section 5 provides the experimental studies and finally, Section 6 presents the conclusion and perspectives.

2. Related research

Many approaches were adopted to collect and analyze data in order to track concept drifting simultaneously to the monitoring process. Such approaches mainly encompassed ensemble learning algorithms and single-model learning algorithms. Ensemble learning consists in maintaining a bank of models created over time. A final prediction is decided by combining the models' outputs, or by choosing the appropriate output from candidates [8]. Brzezinski and Stefanowski [9] put forward block-based ensembles where a new classifier is added in a batch frequency. Similarly, a new classifier model is built on sequential chunks of training points with Street and Kim [10]. Constructed classifiers are then combined into a fixed size ensemble using a heuristic replacement strategy. Tsymbal et al. [8] use a sliding window method to train and add a new model at a fixed frequency. Many other researchers fell into this strategy of models addition/removal over time [11–18]. The removal can occur to maintain only the models that approximate the current process state or when the number of models exceeds a given threshold [3]. Learn^{++}.NSE, an incremental learning algorithm, was introduced by Elwell and Polikar [19]. Learn^{++}.NSE is indeed an ensemble of classifiers approach, training a new classifier on each consecutive batch of data that have become available, and combining them using a weighted majority voting principle. Also, in accordance with the weighting strategy, Soares and Arajo [20] advance an On-line Weighted Ensemble (OWE). Indeed, OWE adds new regressor models if the system's accuracy is decreasing, and takes into account the models' errors in the past and current windows to exclude inaccurate models over time. In the approach proposed by Shrestha and Solomatine [21], new models are added when detecting deterioration on the ensemble performance, and the models representing the least contributions are removed using Boosting theory. In this approach, the models' weights are adapted, but the models are not retrained. Grbovic and Vucetic [22] also focus on the Boosting theory. The proposed approach was implemented in an incremental way (IBoost), and was inspired from the Sliding Window (SW) concept. Soares and Arajo [3] tried to retrain all regression models for each new incoming instance, aiming at keeping the models updated on the current scenario. Despite its effectiveness in tackling concept drifting data streams, the use of ensemble learning approaches presents some issues. The main shortcoming resides in the computational resources that are hugely increasing [3] while only a portion of the ensemble may contain information about the current system state. Single models that adapt to the system changes may be satisfactory.

Window-based single approaches have been widely used in the literature. Widmer [23] was among the first to use SW for choosing reformation block instances. Further works relied on the instance windowing mechanism [24–30]. Incremental approaches have also been raising attention to tackling concept drifting data streams. The first attempt was with Schlimmer et al. [31], who addressed concept drift for classification problems. Incremental learning approaches can ensure faster adaptability for dynamic environments compared to batch-based approaches, the latter requiring to wait for a batch to perform retraining. Nevertheless, they may be outperformed by the batch-based approaches that, despite containing some outliers, may always stabilize the system performance [3]. In another approach, Isazadeh et al. [2] considered all streaming data to ensure incremental learning, sample by sample, for a classification model. Other works of sample-based learning include [32,33]. Unfortunately, applying these approaches still requires high computational time due to the required constant update.

With the aim of reducing the passive model update limitations, many approaches employed drift detection strategies before undertaking the corrective actions. The challenge thus became to discern concept drifts from noise. Bifet and Gavald [34] focused on adapting sliding windows' size for classification problems, the fact that paved the way for inventing the ADaptive WINdowing (ADWIN) method. Rowcliffe [5] adapted existing concept drift detection algorithms to develop inspired methods of learning in the presence of concept drift. Kanoun et al. [35] tried to detect variations of the number of data in the stream input buffer in order to track eventual concept drifts. Khamassi and Sayed-Mouchaweh [36] proposed an Error DISTance-based approach for drift detection and monitoring (EDIST). EDIST monitors the error distance distributions of two data-windows, with a self-adaptive parameters adjustment. The authors proposed thereafter a new framework to deal with complex drifts, called EDIST2 [37]. The framework monitors the learner's performance through a self-adaptive window, autonomously adjusted through a statistical hypothesis test.

All the aforementioned approaches can be applied only on entirely labeled data streams. Indeed, to obtain labeled data, we usually require expert annotation, which makes supervised learning costly and time-consuming. Associated with gathering labeled data, semi-supervised learning (SSL) witnessed a huge popularity, particularly when dealing with environments where labeled data are scarce while unlabeled data are abundant. Several efforts have attempted to address unlabeled data streams. For instance, Masud et al. [6] used the K-Means algorithm to obtain an ensemble of micro-clusters. Then, instances were classified according to the K-nearest neighbor rule. With Ditzler and Polikar [38], classifiers were generated with labeled instances only. Thereafter, K-Means algorithm was used to adjust the weights of each classifier, using predicted labels. Bertini et al. [39] proposed a graph-based semi-supervised approach that extends the static classifier based on the K-associated optimal graph, across on-line semi-supervised monitoring. Instead of K-Means, a clustering algorithm based on K-Modes was introduced and developed with Wu et al. [40], to produce concept clusters at leaves in an incrementally-built decision tree. Labels were predicted in the method of majority-class using these concept clusters. All mentioned SSL algorithms still make the assumption that labeled instances are continuously available in a regular basis. However, such assumption is violated in many practical applications.

Zhang et al. [41] address an environment where labeled instances are received sporadically, in a random way. Received labeled instances are used to generate clusters and classifiers. Then, a label propagation method is used to infer clusters class label, by resorting to both class label information from classifiers, and internal structure information from clusters. Dyer et al. [42] addressed an initially labeled environment, where few labeled instances are received once, at initialization, and only unlabeled instances are received at subsequent time steps. In that approach, the authors try to create α-shapes from the received instances, compact α-shapes and extract instances from the compacted α-shapes, in order to serve as training data for future time steps.

In this paper, we present a new dynamic regression model with fast adaptation of a setting policy to deal with concept drifting data streams for manufacturing scheduling issues. Thereby, we apply the proposed LEM algorithm so as to update the neural-agent learning model, previously constructed off-line, into a model able to work with continuous data stream. Hence, with the application of LEM, we developed an underlying step towards updating parameters of the neural agents composing the decisional model concerned with the detected drifts. To the best of our knowledge, there has been no effort yet to address scheduling manufacturing problems for adaptation to data streams with concept drifts. LEM allows us to extract labels and adjust training data, using entirely unlabeled received instances and without any external supervision, a practice that has never been previously addressed. We show the effectiveness of LEM implementation through illustrative examples and the performed experiments.