Handling Concept Drift in Predictive Process Monitoring

Marco Maisenbacher
Humboldt-Universität zu Berlin
maisenbm@hu-berlin.de

Matthias Weidlich
Humboldt-Universität zu Berlin
matthias.weidlich@hu-berlin.de

***Abstract*—Predictive process monitoring emerged as a technique to anticipate the outcome of a running instance of a business process. To this end, it first constructs a forecast model based on an encoding of traces of past process executions that are labelled with the prediction target. This model is then used to predict the outcome of a running process instance. However, existing approaches neglect that real-world processes are subject to continuous change, so that prediction models need to adapt to concept drift. In this paper, we take up ideas on incremental learning from general data mining and present a paradigm for predictive process monitoring under concept drift. It is grounded in a systematic experimental study that answers the questions of which encoding of process traces and which incremental learning strategies are particularly suited for predictive monitoring of continuously evolving processes.**

I. Introduction

An increasing number of business processes is supported by information systems [13]. Such systems control and monitor how the execution of business activities is coordinated to reach a specific goal. Such activities are often implemented by services [23], so that a business process may be realised by a service orchestration. However, even if activities relate to manual processing steps, they are commonly tracked by an information system [29]. Hence, data on the conduct of activities is commonly available in the form of event logs, collections of traces of past process executions. In recent years, process mining [1] emerged as a discipline that strives for the extraction of knowledge from such event logs.

Most process mining techniques target post-mortem analysis, i.e., the execution of a process is assessed after the fact, to identify operational issues such as compliance violations or bottlenecks. As such, analysis is limited to retrospective compensation once operational issues have been detected.

Predictive process monitoring (PPM), in turn, aims to anticipate operational issues by predicting properties of running process instances [25]. Examples for prediction targets are remaining execution times [2, 28] or the outcome of a process instance [24, 25]. By anticipating operational issues, PPM enables the early implementation of mitigation strategies that lead to massive time and cost savings. PPM commonly features an offline learning phase and an online application phase. During the former, a forecast model is constructed based on an event log that has been labelled with the prediction target. During the latter, a running process instance is assessed based on the learned model.

Since they rely on offline learning, existing PPM methods assume the underlying process to be stable, i.e., stationary over time. However, the context of business processes, i.e., organisational structures, legal regulations, and technological infrastructures, are subject to continuous change, and hence are the processes of a company [10, 14, 31]. As such, a so called *concept drift* is likely to occur in real-life business processes: over time, the execution of a process evolves in terms of control-flow dependencies and data handling.

While the presence of concept drift in business processes has been acknowledged in the process mining literature, existing work focuses on drift detection and localisation. Most prominently, temporal windows may be utilised to detect statistically significant changes in the control-flow of a business process, see [8, 9]. However, these methods do not allow for conclusions on whether the detected control-flow changes actually influence the prediction of properties of running process instances. They may detect changes that are irrelevant for PPM and thus lead to unnecessary invalidation of a forecast model and superfluous learning efforts. Yet, they may also miss changes that influence prediction of process instance properties, but do not materialise in the control-flow.

Against this background, this paper presents a paradigm *to handle concept drift in predictive process monitoring*. We take up ideas from incremental learning in general data mining and integrate them into a method for PPM, in which the forecast model adapts to concept drift automatically. More specifically, this paper makes the following contributions:

- We provide a conceptualisation of PPM under concept drift in statistical terms, thereby providing a precise definition of the problem at hand.
- We present a paradigm to handle concept drift in PPM, which is grounded in incremental learning and applies to drifts in the control-flow as well as data dimension of a business process. We further identify open research questions that need to be addressed when instantiating this paradigm.
- We present a systematic experimental study that provides empirical answers to these open questions. It features controlled drifts in the execution of an insurance claim handling process and outlines which encodings of traces and which incremental learning strategies are particular suited for PPM under concept drift.

The remainder of this paper is structured as follows. We first give an overview of essential concepts and related work on predictive process monitoring (Section II). Subsequently, we introduce our approach to PPM under concept drift and discuss open research questions (Section III). We then explore these questions in an experimental study (Section IV), before we conclude the paper (Section V).

II. Background and Related Work

This section gives background for our work. It reviews the state-of-the-art in predictive monitoring, notions of concept drift, and concept drift in process mining.

A. Related Work on Predictive Monitoring

The general idea of predictive process monitoring (PPM) is to predict the outcome of a running process instance based on past process executions. As such, PPM relies on an offline, supervised learning phase as illustrated in Figure 1. From the information systems that support the conduct of a business process, an event log is extracted in a first step. The log comprises a set of finite sequences of events, each representing a trace, i.e., a past execution of the process.

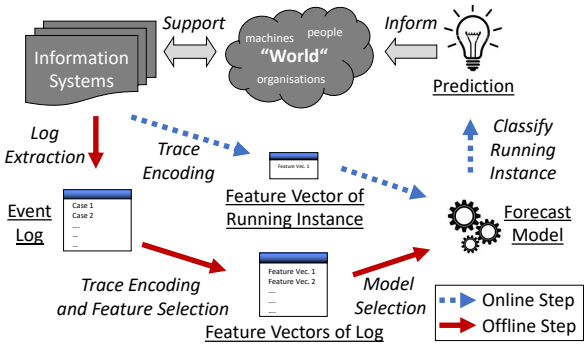


Figure 1: Predictive process monitoring with offline learning

Starting from the event log, the next step in PPM entails encoding the traces to learn a forecast model. A common encoding yields a feature vector per trace and encodes behavioural properties, such as the executed activities, see [27], as well as the values of processed data. Here, encoding behavioural properties of traces is typically approached as a sequence encoding problem [33]. According to a recent study [24], an index-based encoding of executed activities and values of processed data is particularly effective for PPM. Note that the encoding step concerns a predefined prefix of the traces, which determines at which point in time the derived forecast model can be used for prediction for a running instance of the process.

In this work, we focus on satisfaction of a temporal logic (TL) formula as the prediction target. In this case, feature vectors are labelled with truth values of an TL formula. A forecast model, i.e., a classifier, is trained based on the labelled feature vectors using data mining methods [15].

The derived classifier is then applied in an online setting: a predefined prefix of the trace of a running process instance is encoded as in the offline learning phase. It is then classified to be likely to satisfy or violate the TL formula in the future. In practice, this knowledge enables process owners to take pro-active measures to avoid an undesired outcome of this process instance, such as the violation of compliance rules.

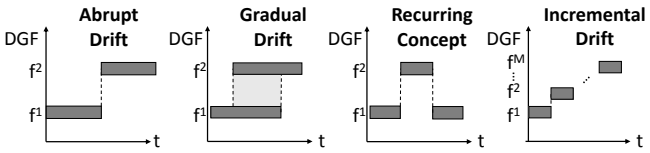


Figure 2: Types of Concept Drift

As an example, consider the process of handling insurance claims. A meaningful prediction target is the question whether a claim will be accepted. To this end, a forecast model is learned from feature vectors that encode the control-flow (*have detailed checks been conducted?*) and processed data (*what is the claim value?*) of past process executions. For a running instance of the process, a feature vector is derived from its trace (prefix) and classified according to this model.

B. Notions of Concept Drift

PPM as introduced above describes a supervised learning problem tailored to the specific context of business processes. In the general field of supervised learning, notions of concept drift have been described as follows [16].

Let C be a set of class labels and let $X_i \in \mathcal{R}^p$ be a p-dimensional data instance of some domain $\mathcal{R}$ at time point i that is labelled with some $c \in C$. Then, supervised learning aims to build a classifier $\mathcal{L}$, given a training set $H = (X_1, X_2, \ldots, X_t)$, to predict the label of a target instance X_{t+1}. In general, any X_i is assumed to stem from a data generating function (DGF) f_i. The DGF can be seen as the joint distribution of the random variables that generated the values of the instance and the label. Classical supervised learning relies on the assumption of a time-invariant data generating process, i.e., $f_i = f_j$ for all time points i and j.

Offline (aka batch) learning means that $\mathcal{L}$ is trained based on H, without any inbuilt capabilities to update $\mathcal{L}$ in terms of new instances. This strategy works as long as the data generating process of the training set coincides with that of the target instances, i.e. $f_1 = f_2 = \ldots = f_t = f_{t+1}$. *Concept drift* occurs if $f_i \neq f_j$ for some i and j, which can severely bias the learning process.

Four types of concept drift have been described in the data mining literature, see Figure 2. Changes in the DGF may be abrupt (f^1 is substituted by f^2) or gradual (f^1 is replaced by f^2 over time). In the latter case, the drift speed determines how quickly the probability of an instance to be produced by f^1 diminishes over time and converges to zero. Furthermore, recurring concept drift describes situations where the initial DGF re-emerges at some point in time. Finally, incremental drift relates to a sequence of changes in the DGF, from f^1 to f^M, in incremental steps, until a steady state is reached.

C. Related Work on Concept Drift in Process Mining

While concept drift is often neglected in process mining, a notable discussion of this topic has been presented by Bose et al. [8]. The authors raise the awareness for concept drift relating to the control-flow, data, and resource dimension of business processes. Here, changes in control-flow are closely related to patterns for behavioural and structural changes in business processes [30]. Bose et al. [8] further highlight the relevance of the general types of concept drift outlined in Figure 2 also for process mining applications. Yet, the solutions provided by [8] relate to the mere detection of abrupt concept drift: based on temporal windows, significant changes in the control-flow of a process are identified. Similar approaches, also with a control-flow focus, have been put forward in [9] to detect abrupt drifts. These ideas have been tailored to detect gradual and recurring drift in [26].

Existing techniques for PPM, however, do not account for drifts in the underlying business processes. A notable exception has been presented by Berti [3] for cycle time prediction. Since the work assumes a-priori knowledge of the concept drift points, it is not generally applicable, though.

III. Predictive Process Monitoring under Concept Drift

This section introduces a paradigm for predictive process monitoring under concept drift. Building upon the general notion of concept drift introduced above, we first provide a conceptualisation of PPM under concept drift in statistical terms. Subsequently, we explain how incremental learning can be applied to produce stable forecasts under such conditions. Finally, we identify questions that need to be addressed to instantiate the proposed approach.

A. The Problem of Concept Drift in PPM

Existing notions of concept drift in process mining, see [8], fall short in covering the forecast dimension of PPM. We therefore provide a definition of concept drift that is derived from the notion introduced in Section II for general supervised learning problems, but relates to the peculiarities of predictive process monitoring.

We model the context of PPM by a set S of events, which is extended over time. Each event $e_i^k \in S$ carries information on when it has been added to S, i.e., at time point i, and to which trace it belongs, i.e., process instance k. Let S_k denote the maximal set of events of trace k, i.e., $S_k = \{e_j^{k'} \in S \mid k' = k \ \wedge \forall\ e_l^{k''} \in (S \setminus S') : k'' \neq k\}$.

Furthermore, let $\mathcal{E}$ denote an encoding operation that assigns a p-dimensional feature vector $X \in \mathcal{R}^p$ to any set S_k that contains the events of a single trace. Such a feature vector X includes a label $c \in C$ for a discrete prediction target, i.e., the truth value of a TL formula.

Let t be the current time, i.e., $\forall\ e_i^k \in S$, it holds that $i \leq t$. Concept drift in predictive process monitoring occurs, if for some trace k and some point in time $j \leq t$, it holds that

$$P_t(\mathcal{E}(S_k)) \neq P_j(\mathcal{E}(S_k)), \tag{1}$$

where P_t and P_j denote the joint distributions of the labelled feature vectors at time t and j, respectively.

The above definition of concept drift in PPM highlights the importance of trace encodings. Defining concept drift as a change in the joint distribution of feature vectors, it further enables us to reason on the sources of concept drift based on Bayes' theorem. According to [21], given a classification problem, the theorem selects the prediction target label $c \in C$ with the highest posterior probability for the target instance:

$$P(c|X_{t+1}) = \frac{P(c) \times P(X_{t+1}|c)}{P(X_{t+1})}. \tag{2}$$

Consequently, concept drift may stem from changes in all these components, i.e., changes in the posterior probability $P(c|X_{t+1})$, the prior probability $P(c)$, and the class conditional probability $P(X_{t+1}|c)$.

Adopting this view to predictive monitoring, we consider the following drift examples in the above scenario of insurance claim handling. Let the prediction target be the acceptance of a claim and suppose that claims are processed differently depending on whether the claim value is high or low. Furthermore, most claims of a high value are rejected, whereas the opposite holds true for claims of a low value. Raising the threshold that categorises claims as high or low value will result in (1) a changed prior probability, since more claims are considered as low value and thus will be accepted; and (2) a changed class conditional probability, since the probability to observe an accepted claim with a value between the new and the old threshold value decreases. Both changes are likely to influence the posterior probability and hence the classification decision.

B. PPM with Online Learning

The main difference of our paradigm with respect to the state-of-the-art discussed in Section II is a switch to online learning. As illustrated in Figure 3, completed executions of a process arrive over time. Their traces are encoded, so that new feature vectors are continuously created and incorporated by an incremental learning algorithm. Running process instances are then classified with the most recent forecast model.

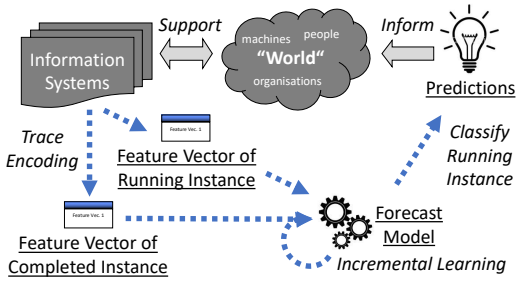


Figure 3: Predictive process monitoring with online learning

In essence, incremental learning works as follows: Suppose $\mathcal{L}_1$ is an initial learner (hypothesis) at time point t that describes all data seen so far. New data arrives one at a time. Hence, at every successive period $t+i$, with $i \geq 1$, a new labelled instance X_{t+i} arrives. A new learner $\mathcal{L}_2$ is produced as an updated version of $\mathcal{L}_1$, based solely on the set of data instances that arrived after t and on the outdated hypothesis $\mathcal{L}_1$. As such, learners are expected to keep track of sufficient statistics that mirror the dynamics of older data instances.

The frequency with which a new learner is constructed, i.e., the number of traces in our context, varies among learning algorithms. It also impacts the trade-off of incremental learning between robustness to irrelevant events (e.g., outliers) and adaptability to new data (e.g., changes in concepts).

For the setting of PPM, different types of incremental learning algorithms may be used: simple ones with no particular capabilities to handle concept drift and advanced ones that handle drift explicitly [16, 18].

1) Simple Incremental Learners: First and foremost, *naive Bayes* may be applied in an incremental manner [5], updating its hypothesis with each arriving trace. Specifically, the current estimates of prior and class conditional probabilities are updated by incrementing the respective counts based on the new data instance.

The *Online Perceptron* is another incremental learner that produces a linear hyperplane between instances of two classes by minimizing the classification error measured by an objective function. It relies on stochastic gradient descent and is configured by the learning rate [7].

Furthermore, tree-based learners, as used for PPM with offline learning [25], may work incrementally. *Hoeffding Trees* [12] determine splitting rules per batch of data instances (i.e., traces in PPM), classifying them with the most recent model, before deriving new rules that extend the tree. The latter is controlled by the split confidence that bounds the potential bias introduced by incremental processing. A grace period also specifies the minimal amount of instances at a node necessary to allow for further splits.

2) Advanced Incremental Learners: Learners that handle concept drift explicitly rely on windowing and alerting. Sliding windows select which incoming data instances shall be used for learning. To avoid the need to specific a static window size, adaptive windowing (ADWIN) adapts windows by testing for differences in the distribution of sufficiently large sub-windows of a current window. Windowing can be combined with the simple incremental learners above. That is, the *Adaptive Hoeffding Tree* using ADWIN [4] relies on sliding windows of data instances to compute splitting rules that may replace existing rules, once their classification error becomes smaller. Keeping a sliding window per node, the window size is governed by ADWIN.

Furthermore, option trees with nodes that replace a single splitting rule by a set of rules allows for potentially multiple paths from the root to leaves of the tree for a single data instance. The actual decision is then taken by an additional rule over nodes with multiple options. Combined with the notion of a Hoeffding tree, this yields the model of *Adaptive Hoeffding Option Trees* [6], which tracks the classification error of each node estimated by an exponentially weighted moving average (EWMA). This estimate is used to decide among multiple options, so that outdated branches in the tree possess minimal influence on the classification decision.

Finally, the drift detection mechanism (DDM) [17] may alert and detect drifts based on the error rate of a corresponding classifier over time. Combining it with any simple incremental learner yields a Single Classifier Drift [17]. If DDM raises a drift alert, new data instances are stored. Upon drift detection, a new model is learned from these instances.

C. Open Research Questions

To instantiate the above paradigm, three main research questions need to be answered.

Research Question 1: What is a good encoding operation for PPM under concept drift?

PPM under concept drift is heavily influenced by the chosen encoding operation for traces. For offline learning, it has been shown that encodings that combine the control-flow and data dimensions of a business process outperform simpler approaches [24]. Yet, PPM with online learning may particularly suffer from the curse of dimensionality in the case of high dimensional encodings: Most incremental learners require a large increase in the volume of training data even for a small increase in dimensionality [22]. It is further known that the mean accuracy of classifiers decays after model complexity reaches some optimal value [19]. Thus, the question is whether the encodings presented in [24] for offline learning also work well with our online paradigm.

Research Question 2: Which incremental learners are suitable for PPM under abrupt concept drift?

Focusing on abrupt concept drift as the most commonly described drift type, there is neither theoretical nor empirical evidence for a globally superior incremental algorithm for classification problems [32]. More specifically, the performance of any learner depends on its alignment with the posterior probability of the specific learning problem. While there is no clear hypothesis for an answer to the above question, studies in stream mining suggest that advanced incremental learners shall outperform learners that do not handle concept drift explicitly. It is unclear whether these results carry over to the case of PPM.

Research Question 3: What are the consequences of different drift speeds in PPM?

Beyond abrupt concept drift, gradual drifts and recurring concepts are particularly important for business processes. Gradual drifts occur when process changes are incorporated by applying them solely to instances that are started after a particular point in time. Recurring concepts, in turn, may

be caused by seasonal effects (e.g., peaks in the number of running instances) on the execution of business processes. For gradual drifts, the question is how the drift speed impacts the adaptivity of the forecast model, given that the learning is disturbed occasionally by traces still related to the old concept. For drifts with recurring concepts, the question is whether learners adopt quicker to a concept seen earlier.

IV. Experimental Study

To highlight how the proposed paradigm handles concept drift in predictive process monitoring and to answer the above research questions, this section presents a systematic experimental study. Below, we clarify the scenario and setup, before turning to the experimental results with respect to these questions.

A. Scenario

The scenario is grounded in a health insurance claim handling process at a travel agency, which was put forward as a benchmark for the detection of concept drift in [8] and used in several other studies [26]. The process is visualised as a BPMN model in Figure 4. In essence, there is either a basic or a comprehensive check of claims, depending on their claim value. Afterwards a decision is taken and the claimant is notified, potentially via different communication means. In parallel, a questionnaire is posted to the claimant.

Extending the benchmark towards PPM, the process was enriched with variables describing characteristics of the claims and the corresponding claimants, see Table I.

Table I: Data in the claim handling process

Attribute	Attribute-Type	Domain
Age	Numeric	Normal Distribution
Claim Value	Numeric	Normal Distribution
Customer Status	Character	4 Categories
Diagnosis Code	Character	5 Categories
Prior Claims	Binary	Levels: Yes, No

We considered two prediction targets for this process, also indicated in Figure 4. First, the agency is interested in predicting whether a claim will be accepted or rejected in a running process instance. The second prediction target is cost related: whether notification will involve phone and post.

As a baseline concept, we considered the following grounding of these predictions. A claim is of high value, if its value is above 1000 Euro. Those claims are rejected in 80% of the cases. The most important predictor for the rejection or acceptance of these claims is the existence of prior claims. On the other hand, claims with low value are accepted in 80% of the cases. The most important predictor for these claims is a particular diagnosis code. Regardless of the claim value, though, the execution of business activities plays a role, since the decision point at which claims with high and low value are separated is directly visible in the control-flow. The second prediction target is mostly influenced by the two data attributes age and value, as specified in Figure 4.

We further define two concepts to which the process may drift from the baseline. First, concept C1 differs from the baseline in the logic of clustering claims into those that need basic or comprehensive checking. Comprehensive claim checking is now done, if the claimant's age is >50. These claims are accepted or rejected based on the customer status instead of the existence of prior claims. For claims with basic checking, the decisive factor is still the internal diagnosis code, yet the condition changed. A drift to concept C1 is expected to change the posterior probabilities and the class conditional probability for the first prediction target.

Concept C2 differs from the baseline in the process structure: For cost purposes, hospitals are no longer contacted as part of comprehensive claim checking. Also, the data distribution of incoming claims differs: patients are older (the baseline mean age, 50, increased by four years) and claim values are higher (the baseline mean value, 1000, increased by 70). Considering the second prediction target, a drift to this concept is expected to change the prior probabilities. However, changes in the posterior probability are likely to be less accentuated compared to a drift to C1.

B. Experimental Setup

To simulate the benchmark process, we adapted its formalisation as a Coloured Petri net (CPN), provided by the authors of [8]. Using the CPN Tools [20] engine, the traces of 100,000 process instances have been derived for each concept (baseline, C1, and C2). The results were split into 25 event logs per concept, which enabled us to conduct sensitivity analysis by means of confidence intervals.

We note that online classification requires tailored evaluation measures (standard measures for offline learning such as ROC do not account for the temporal dimension). In the remainder, we measure *accuracy*, i.e., the proportion of both true positives and true negatives among the total number of considered instances, and follow a *interleaved-test-then-train* (ITTT) approach [5]. That is, each data instance is used to test the current model before it is utilised for training, so that the classification accuracy is updated incrementally. We used ITTT with a sliding window of 100. That is, the value for the k-th instance is the mean accuracy of the ITTT results for instances $k - k/2$ until $k + k/2$. Furthermore, we will always illustrate the 95% confidence intervals based on the 25 trials, computed with a non-parametric bootstrap method with 100 bootstrap replicates [11].

Based on the baseline concept and concepts C1 and C2, we define two evaluation scenarios. Scenario S1 focuses on the first prediction target (claim acceptance), when the process drifts from the baseline to concept C1. Scenario 2 is about the second prediction target (notification) for a process that drifts from the baseline to concept C2. In the trace encoding, S1 and S2 used a prefix length of two and four, respectively.

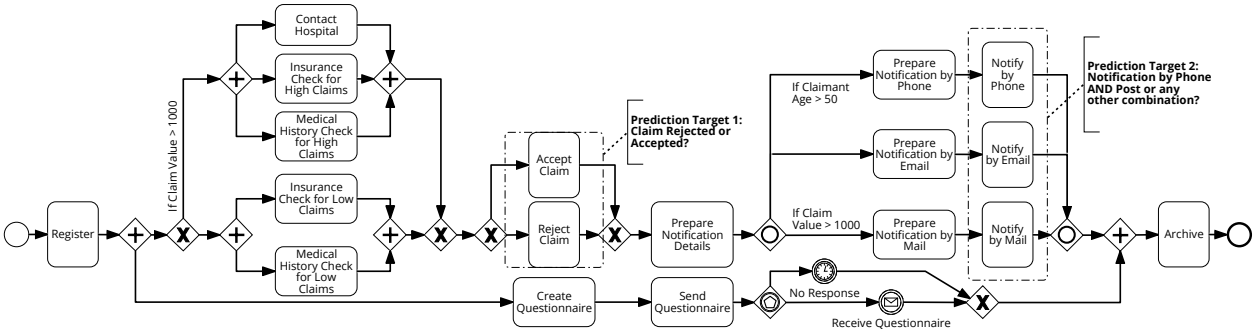
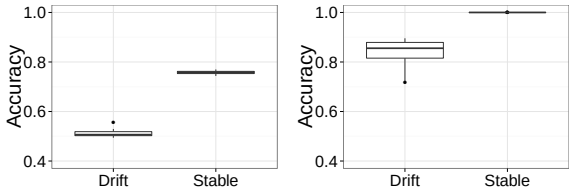


Figure 4: Insurance claim handling benchmark, adopted from [8], used as the baseline in the experimental study

C. Experimental Results

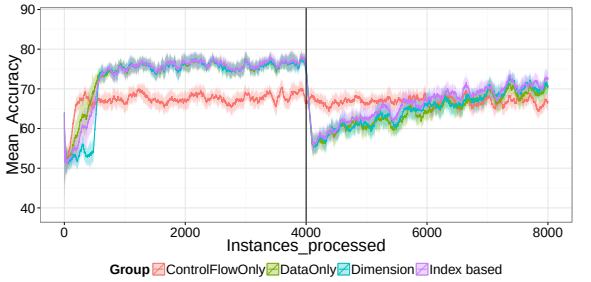
Our experiments are steered by the research questions raised in Section III-C. However, before turning to these questions, we demonstrate the shortcomings of PPM with offline learning (see Section II) under concept drift. To that end, we apply the methodology of [24] for both evaluation scenarios, using the index-based trace encoding and a random forest model. The loss in accuracy of the forecast model after the drift happened is illustrated in Figure 5 with boxplots for the statistical moments. In both scenarios, the results witness a dramatic loss of accuracy, if the classifier learned before the drift, is applied afterwards. For scenario S1, e.g., the median accuracy drops by 25%, from 0.75 to 0.50. This highlights the importance of handling concept drift in PPM.



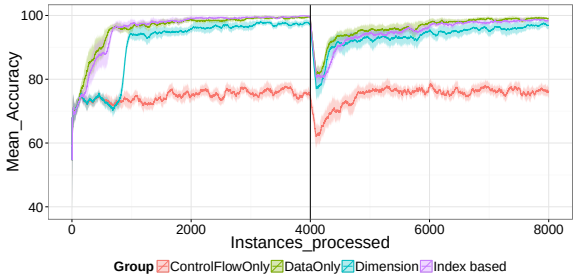
(a) S1: Acceptance (b) S2: Notification

Figure 5: Impact of drift for both prediction scenarios

To answer research question RQ1 on the choice of the trace encoding, we consider four encoding strategies: the *control-flow only* approach are feature vectors that contain solely the executed activities; the *data only* strategy are feature vectors that contain solely the values of processed data; the *index-based* encoding combines both, control-flow and data, yet reduces the dimensionality by separating static data from data that changes over time; the *dimension* encoding is the full combination of control-flow and data features. Using Hoeffding trees as the model, the impact of abrupt concept drift in both evaluation scenarios is shown in Figure 6. While the drift leads to a drop in accuracy for all encoding strategies, the index-based encoding performs best overall. It combines the behavioural and data perspective of business processes (improving over control-flow only or data only encodings in scenario S1), and avoids the curse of dimensionality (improving over the dimension encoding in S2).



(a) S1: Acceptance

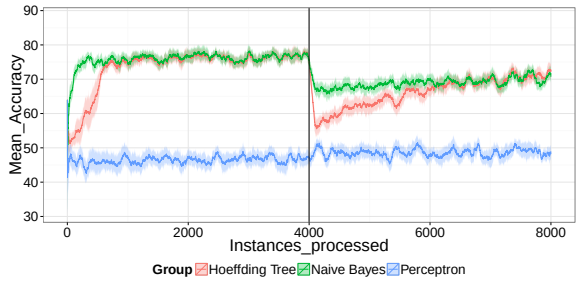


(b) S2: Notification

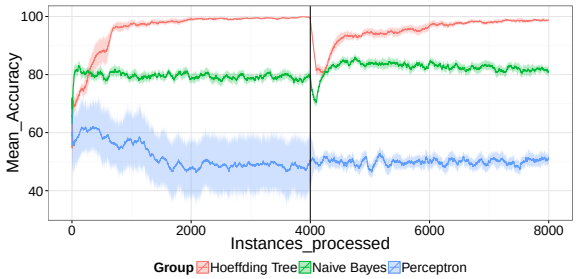
Figure 6: Influence of the trace encoding strategy

Focussing on RQ2, the suitability of specific learners, we start by exploring the accuracy obtained incremental learners that do not handle concept drift explicitly. We consider the same drift scenarios as above (index-based encoding), using the online Naive Bayes (NB), Hoeffding tree (HT), and Online Perceptron (OP) classifiers, see Section III-B. We use HT with a split confidence of 0.1 and OP with a learning rate of 1, which turned out yield the best results. The results in Figure 7 illustrate that the OP model yields very weak results for both prediction targets. We attribute this to the OP being a linear classifier, whereas our scenarios require non-linear classification. The NB and HT models drop in accuracy at the drift point and then adapt to the new concept. The HT is hit stronger by the drift, since its adaptivity is limited by the grace period and the split confidence. Though, it may achieve better overall accuracy, as in scenario S2.

Next, we explore the advanced incremental learners that explicitly handle concept drift: the adaptive Hoeffding tree (AHT), the adaptive Hoeffding option tree (AHOT) and

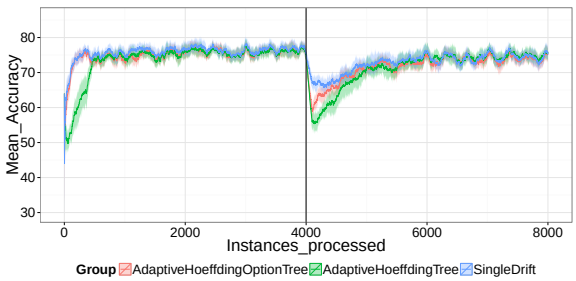


(a) S1: Acceptance

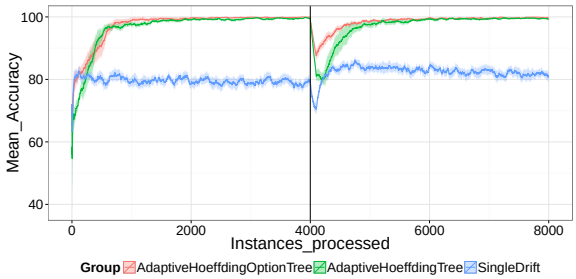


(b) S2: Notification

Figure 7: Simple incremental learners



(a) S1: Acceptance



(b) S2: Notification

Figure 8: Advanced incremental learners

the single drift (SD) classifiers. Again, the optimal split confidence for the tree-based models is determined as explained above. In the same vein, the optimal number of options is set for AHOT. For SD, naive Bayes is used as a baseline classifier. As illustrated in Figure 8, in scenario S1, the classifiers show similar accuracy, with SD being least hit by the concept drift. The latter is due to the underlying naive Bayes being less affected in this scenario. In S2, in turn, SD has weak overall accuracy. The tree-based models, AHT and AHOT, generally also quickly adapt to the new concept. AHOT further outperforms AHT, especially in the drift phase, which indicates that the combination of tree options with an EWMA estimation of the current error delivers a faster drift recognition than the ADWIN-based approach in the AHT. Also, AHOT performs significantly better then the simple learners considered in Figure 7. For instance, in scenario S2, NB and HT yield a mean accuracy of 75% and 81%, respectively, for the 200 traces directly following the drift. For these traces, AHOT achieves a mean accuracy of 89%.

Turning to RQ3, we test the two best classifiers from the above experiments in terms of stability under gradual drifts and drift with recurring concepts. To implement a gradual drift, following [5], we consider the weighted combination of two pure distributions that characterizes the target concepts before and after the drift. That is, there is a probability according to which the traces are drawn from either scenario and which diminishes over time. As suggested by [5], this probability is modelled by the sigmoid function. Figure 9 shows the results for scenario S2 under two drift speeds, i.e., convergence to the new concept within 100 traces or 1000 traces, respectively. We observe that the drop in accuracy is partly mitigated under a slow drift speed. In addition,

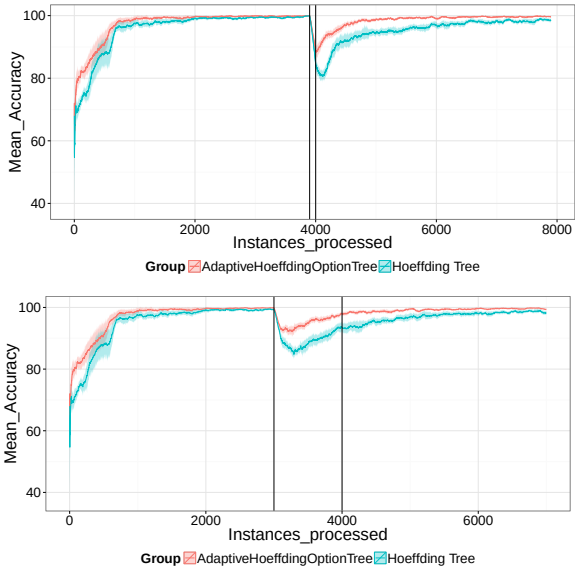


Figure 9: Impact of drift speed (scenario S2)

adjustment to the new concept happens slower with a slow drift speed. However, we note that the overall accuracy of the tested models is comparable to the case of abrupt drift for the prior and post drift periods.

Finally, we turn to recurring concepts. We consider a scenario where 2000 traces come from the baseline, followed by an abrupt drift to the new concept for 4000 traces. Afterwards, traces come from the baseline again. For scenario S2, the results are shown in Figure 10 (S1 yields similar trends). While AHOT is more accurate overall, the drop in its accuracy is also smaller after the second drift point, since information on the baseline concept is still part of the model.

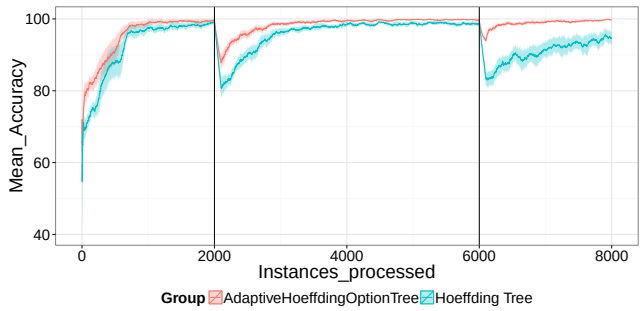


Figure 10: Drift with recurring concept (scenario S2)

V. Conclusion

This paper proposed a paradigm for online predictive process monitoring. Addressing the need for adaptivity due to continuous changes in business processes, it employs incremental learning to adapt the forecast model. We further argued that an instantiation of this paradigm raises research questions on the appropriateness of trace encodings, the selection of incremental learners, and the impact of drift speeds and recurring concepts. We therefore presented a systematic experimental study based on the well-established benchmark that provides empirical evidence to answer these questions. In future work, we intend to focus in more detail on the sensitivity of our paradigm with respect to other evaluation scenarios, e.g., structural changes in business processes and multi-dimensional prediction targets.

Acknowledgements. We are grateful to J.C. Bose for providing us with the CPN formalisation of the benchmark process. This work has been supported by the German Research Foundation (DFG), grant WE 4891/1-1.

References

[1] W. M. P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.

[2] W. M. P. van der Aalst, M. H. Schonenberg, and M. Song. "Time prediction based on process mining". In: *Inf. Syst.* 36.2 (2011), pp. 450–475.

[3] A. Berti. "Improving Process Mining prediction results in processes that change over time." Data Analytics, IARIA. 2016.

[4] A. Bifet and R. Gavaldà. "Learning from Time-Changing Data with Adaptive Windowing". In: *SDM*. 2007, pp. 443–448.

[5] A. Bifet and R. Kirkby. *Data stream mining. A Practical Approach*. 2011.

[6] A. Bifet et al. "New ensemble methods for evolving data streams". In: *ACM SIGKDD*. 2009, pp. 139–148.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[8] R. P.J. C. Bose et al. "Dealing With Concept Drifts in Process Mining". In: *IEEE Trans. Neural Netw. Learning Syst.* 25.1 (2014), pp. 154–171.

[9] J. Carmona and R. Gavaldà. "Online Techniques for Dealing with Concept Drift in Process Mining". In: *IDA*. 2012, pp. 90–102.

[10] P. Châtel, J. Malenfant, and I. Truck. "QoS-based Late-Binding of Service Invocations in Adaptive Business Processes". In: *IEEE ICWS*. 2010, pp. 227–234.

[11] A. C. Davison and D. V. Hinkley. *Bootstrap Methods and their Application*. Cambridge University Press, 1997.

[12] P. M. Domingos and G. Hulten. "Mining high-speed data streams". In: *ACM SIGKDD*. 2000, pp. 71–80.

[13] M. Dumas et al. *Fundamentals of Business Process Management*. Springer, 2013.

[14] W. Fdhila et al. "Change and Compliance in Collaborative Processes". In: *IEEE SCC*. 2015, pp. 162–169.

[15] C. D. Francescomarino et al. "Predictive Business Process Monitoring Framework with Hyperparameter Optimization". In: *CAiSE*. 2016, pp. 361–376.

[16] J. Gama et al. "A survey on concept drift adaptation". In: *ACM Comput. Surv.* 46.4 (2014), 44:1–44:37.

[17] J. Gama et al. "Learning with Drift Detection". In: *SBIA*. 2004, pp. 286–295.

[18] R. T. Hoens, R. Polikar, and N. V. Chawla. "Learning from streaming data with concept drift and imbalance: an overview." In: *Progress in Artificial Intel.* (2012), pp. 89–101.

[19] G. Hughes. "On the mean accuracy of statistical pattern recognizers". In: *IEEE Trans. on Inf. Theory* 14.1 (1968), pp. 55–63.

[20] K. Jensen, L. M. Kristensen, and L. Wells. "Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems". In: *STTT* 9.3-4 (2007), pp. 213–254.

[21] M. G. Kelly, D. J. Hand, and N. M. Adams. "The Impact of Changing Populations on Classifier Performance". In: *ACM SIGKDD*. 1999, pp. 367–371.

[22] E. J. Keogh and A. Mueen. "Curse of Dimensionality". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by C. Sammut and G. I. Webb. Springer, 2017, pp. 314–315.

[23] K. Klai and S. Tata. "Formal Modeling of Elastic Service-Based Business Processes". In: *ICSOC*. 2013, pp. 424–431.

[24] A. Leontjeva et al. "Complex Symbolic Sequence Encodings for Predictive Monitoring of Business Processes". In: *BPM*. 2015, pp. 297–313.

[25] F. M. Maggi et al. "Predictive Monitoring of Business Processes". In: *CAiSE*. 2014, pp. 457–472.

[26] J. Martjushev, R. P.J. C. Bose, and W. M. P. van der Aalst. "Change Point Detection and Dealing with Gradual and Multi-order Dynamics in Process Mining". In: *BIR*. 2015, pp. 161–178.

[27] S. Oh et al. "Forecasting Workloads in Multi-step, Multi-route Business Processes". In: *IEEE SCC*. 2014, pp. 355–361.

[28] A. Rogge-Solti and M. Weske. "Prediction of Remaining Service Execution Time Using Stochastic Petri Nets with Arbitrary Firing Delays". In: *ICSOC*. 2013, pp. 389–403.

[29] D. Schulte. "Human Task Management for RESTful Services". In: *ICSOC Workshops*. 2011, pp. 207–212.

[30] B. Weber, M. Reichert, and S. Rinderle-Ma. "Change patterns and change support features - Enhancing flexibility in process-aware information systems". In: *Data Knowl. Eng.* 66.3 (2008), pp. 438–466.

[31] M. Weidlich, M. Weske, and J. Mendling. "Change Propagation in Process Models Using Behavioural Profiles". In: *IEEE SCC*. 2009, pp. 33–40.

[32] D. H. Wolpert. "The Lack of a Priori Distinctions Between Learning Algorithms". In: *Neural Comput.* 8.7 (Oct. 1996), pp. 1341–1390. ISSN: 0899-7667.

[33] Z. Xing, J. Pei, and E. J. Keogh. "A brief survey on sequence classification". In: *SIGKDD Explorations* 12.1 (2010), pp. 40–48.