Encoding Resource Experience for Predictive Process Monitoring

Jongchan Kim[a], Marco Comuzzi[1a], Marlon Dumas[b], Fabrizio Maria Maggi[c], Irene Teinemaa[d]

[a]*Department of Industrial Engineering, Ulsan National Institute of Science and Technology, Ulsan, Republic of Korea*
[b]*Institute of Computer Science, University of Tartu, Tartu, Estonia*
[c]*Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy*
[d]*Booking.com, Amsterdam, The Netherlands*

Abstract

Events recorded during the execution of a business process can be used to train models to predict, at run-time, the outcome of each execution of the process (a.k.a. case). In this setting, the outcome of a case may refer to whether a given case led to a customer complaint or not, or to a product return or other claims, or whether a case was completed on time or not. Existing approaches to train such predictive models do not take into account information about the prior experience of the (human) resources assigned to each task in the process. Instead, these approaches simply encode the resource who performs each task as a categorical (possibly one-hot encoded) feature. Yet, the experience of the resources involved in the execution of a case may clearly have an impact on the case outcome. For example, specialized resources or resources who are familiar with a given type of case, are more likely to execute the tasks in a case faster and more effectively, leading to a higher probability of a positive outcome. Motivated by this observation, this article proposes and evaluates a framework to extract features from event logs that capture the experience of the resources involved in a business process. The framework exploits traditional principles from the literature to capture resource experience, such as experiential learning and social ties on the workplace. The proposed framework is evaluated by comparing the performance of state-of-the-art predictive models trained with and without the proposed resource experience features, using publicly available event logs. The results show that the proposed resource experience features may improve the accuracy of predictive models, but that depends on the process execution context, such as the type of process generating an event log or the type of label that is predicted.

Keywords: Process Mining, Predictive Process Monitoring, Resource Experience

1. Introduction

Business processes are often supported by enterprise software systems, such as Customer Relationship Management (CRM) systems or Enterprise Resource Planning (ERP) systems. Such systems keep detailed records of relevant events that punctuate the execution of a process, such as the start or completion of tasks, the receipt of messages, etc. These records can be extracted from the databases of these systems and packaged in the form of *event logs*. In this context, an event log is a collection of *(execution) traces*, each one capturing the trail of events that occurred during one particular execution of a process (a.k.a. case). Each trace, in turn, consists of a sequence of events. Events are ordered in time, usually because they contain at least one timestamp. Moreover,

[1] Corresponding author - mcomuzzi@unist.ac.kr

an event contains a reference to a task, an identifier of the resource that performed the task, and possibly other domain-specific attributes.

Predictive process monitoring is a family of techniques that exploits event logs of business processes in order to generate predictions of future states or properties of each ongoing case of a process. Predictive monitoring methods differ depending on the prediction target. For example, a subset of predictive monitoring methods focus on predicting the remaining time of a case [53], while other methods focus on predicting the next events [9] or the outcome of a case [50]. In this paper, we focus on the latter "outcome-oriented" methods, i.e., we aim at predicting a case outcome expressed as a binary property of a case that is known when the case completes. For instance, in an order-to-cash process, the outcome of a case may be that the customer is satisfied with the delivered product (or, conversely, that the customer is unsatisfied), whereas in a helpdesk process the outcome may be that the reported issue is resolved on time (versus late) [50]. Predictive models allow decision makers to trigger actions pro-actively in order to prevent undesirable situations in a process.

In order to train machine learning models for predictive process monitoring, each prefix of a trace in the log (representing a partial case execution) is transformed into a feature vector, which is then labeled by a prediction target (e.g., the case outcome). The resulting set of labeled feature vectors is then used to train a classifier, a regressor or other types of predictive models (e.g., structured predictors). The resulting model is then fed with feature-encoded incomplete cases, at runtime, to generate the desired predictions.

The accuracy of a predictive monitoring method largely depends on the richness of information encoded in the feature vectors representing the incomplete cases. A wide range of methods for encoding incomplete traces as feature vectors have been proposed and evaluated in the literature [50]. These methods differ in terms of how they encode the temporal information (timestamps), the sequences of tasks, the (human) resources that perform each task, and other domain-specific attributes. As far as resources are concerned, existing approaches treat the resource as any ordinary attribute and in doing so, they neglect the fact that the outcome of a case (or the remaining time or other properties of a case) largely depend on the assignment of resources to tasks. In particular, existing predictive process monitoring approaches do not take into account the prior experience of the resources assigned to each task in a case.

Yet, resource experience plays an important role in the way a process case will unfold. In both clerical and knowledge-based tasks, users tend to become more efficient and effective as they gain experience on a certain task [29]. The more recent the experience, the stronger this effect may be. Also, users may be more efficient or effective when working with other specific users, for instance, because of their good personal relations or fit of personalities [12]. Based on these considerations, we hypothesize that resource information in an event log can be exploited to derive an additional set of features to enhance the performance of the trained predictive models.

This article proposes a framework for deriving features capturing resource experience from the raw data available in an event log. To this end, the paper identifies four dimensions of resource experience in business processes, which can be combined to obtain a wide set of resource experience features. For example, the experience of a resource may be captured by measures related to task familiarity, such as the number of times the resource has executed the same task in previous cases of the same process.

The proposed resource experience features are then used to extend the feature vectors used for training the predictive models. The article then investigates the effect of these resource experience features on the performance of the models used for predicting case outcomes. Specifically, the paper reports on an evaluation of the proposed framework designed to address two research questions: Do the new features capturing resource experience help to improve the performance of outcome-oriented predictive models? And, among the different types of features capturing resource experience, which one(s) are the most important in determining the behavior of outcome-oriented predictive models?

The paper is organized as follows. Related work and preliminaries are discussed in Section 2 . The research problem is introduced in Section 3. Section 4 presents the framework for generating resource experience features, while Section 5 presents the experimental evaluation. Concluding remarks are finally drawn in Section 6.

2. Background and Related Work

This section reviews related work about predictive process monitoring and resource information in event logs (Section 2.1) and about management insights regarding business process outcomes and the development of resource experience in the workplace (Section 2.2).

2.1. Predictive Process Monitoring and resource data in event logs

Predictive process monitoring [34] concerns various prediction tasks such as predicting the outcome of a process [32, 50], the next event of a running case [46, 48], or a time-related measure, e.g., the remaining time until the termination of a running case [47]. In outcome-oriented predictive monitoring, the outcome of a case is usually a binary variable. Approaches in the literature often define outcomes as the satisfaction of service level agreements or the satisfaction of temporal constraints defined on the order and the occurrence of tasks in a case [35, 50]. Extensive efforts have been devoted to enhancing the performance of predictive monitoring models from both the pre-processing and the learning sides. During pre-processing, trace clustering techniques [13, 14, 32] and sequence encoding techniques [28, 50] have been used to efficiently extract features from the input data. In learning, cutting-edge classification and regression algorithms, such as deep learning, have been applied for the purpose of predicting various targets of interests in business processes [19, 27, 39, 47].

All the aforementioned approaches consider intra-case features, generated only using events within an individual case. These are opposed to inter-case features, which are generated using events across multiple cases in the event log [44]. For instance, given an event occurring at a certain time instant, the number of cases active in a business process at that instant or the number of cases waiting to execute a certain activity in the process at that time instant are typical examples of inter-case features.

While in practice different cases are correlated from the perspectives of resource involvement, remaining time and other attributes, inter-case features have not been considered extensively in predictive process monitoring. The importance of incorporating inter-case features has been pointed out in different prediction tasks. In predicting the risk associated with a case, Conforti et al. [11] considers not only the local risk predictors intended for a single instance, but also the interplay between risks associated with multiple instances of the same process. This is motivated by the fact that the same resource can be employed by multiple instances running simultaneously, which may lead to multiple instances sharing some risks. Similarly, Senderovich et al. [43] developed an inter-case encoding method that considers concurrently active cases for generating inter-case features (e.g. the number of acute patients in the emergency department). More recent research has focused on the modelling of interactions among cases to improve process analysis and forecasting. Klijn and Fahland [23] have proposed an approach to model inter-case features for remaining time prediction. The features are used to identify contexts of high remaining time prediction errors, such as when cases are batched. Fahland et al. [17] have proposed an approach for reconstructing missing timestamps when cases share physical resources. Brunk at al. [8] have proposed an approach for predicting unexpected events that considers also the case execution context. Generally, we argue that inter-case features in predictive monitoring have been considered mainly from a system load perspective. In this paper, we propose a new class of inter-case features, that is, resource-aware features, which capture the experience of resources by referring to their historical involvement in the execution of a process.

Nakatumba and van der Aalst [37] developed a technique to analyse the impact of resource workload on service times using event log data. From the perspective of resource allocation, Arias et al. [3] proposed a resource allocation recommendation framework based on metrics such as frequency, performance, quality, cost, expertise and workload. Zhao et al. [55] propose a resource allocation optimisation model that considers the constraints of process execution time, cost and resource availability. Bidar et al. [5] consider resource preferences to solve the problem of resource-task allocation in business process automation. Erasmus et al. [16] also aim at improving the resource allocation in business processes by considering the resource ability, specified using the Fleishman's taxonomy. As far as performance monitoring is concerned, Senderovich et al. [45] use data mining classification and heuristic methods based on queuing theory to show how the performance of a process can be affected by the scheduling of resources. While focusing on different objectives, these works characterise resources in terms of their expertise or ability, i.e., the set of tasks that they usually perform, workload, i.e., the number of tasks or cases in which they are currently involved in, and the type of outcomes of cases in which they are involved in.

Pika et al. [41] devised a method to mine resource profiles from event logs. These profiles characterize resources in terms of skills, utilization, preferences, productivity and collaboration. Even though they are not intended to serve as features for predictive process monitoring, the rationale behind the design of resource profiles is in several ways similar to the rationale behind the framework that we propose. For instance, the skills profile is defined by the number of tasks in which a resource participates, while the productivity profile is linked to the outcome of the cases in which resources participate. Similarly, the number of tasks executed by a resource and the ratio of cases involving a resource with positive outcomes are features defined in our framework. A deeper comparison between resource profiles and our proposed features is provided later in Section 4.4.

2.2. Business process outcomes and resource experience

Business process outcomes concern the improvement of the operational efficiency, effectiveness and flexibility of business processes [36]. They are normally evaluated using process performance indicators [1]. While research has scarcely investigated the direct relation between resource experience and business process outcomes, a large body of research has focused on the positive relation between IT implementation and capabilities, in particular ERP systems, and business process outcomes [22, 54]. In this context, the skills of the IT resources and the actual usage of IT systems normally is positively related with positive process outcomes. In the healthcare sector, the Donabedian conceptual model [15] assumes that the *structure* of a healthcare organisation, including its human resources, influences the healthcare *outcome* through the *process*, i.e., the set of actions that make up the provisioning of healthcare services. Finally, the reflective perspective on business process management [4] suggests that the human resource individual experience can play a major positive role in the design, enactment, monitoring and improvement of organisational business processes.

A number of classic theories in management have highlighted how repeating experience and developing social ties on the workplace can positively influence invidual job performance. The commonly understood concept of the learning curve, which has also found empirical evidence [29, 51], explains the idea that performance on the workplace improves with experience at repeating the same or similar tasks. Similar learning dynamics are posited by the theories of experiential learning [20, 25] and absorptive capacity at the individual level [10]. Experiential learning affirms that learning occurs mainly through experience. In the workplace, experience is gained by repeating the same or similar tasks several times. Absorptive capacity is the ability to recognize the value of new information, assimilate it, and apply it for commercial ends [10], which has a positive impact on performance. At the individual level, this translates into the ability of individuals to assimilate and apply knowledge about executing tasks in order to improve their ability to execute them in the future [40]. There is also empirical evidence that experience learned in the workplace can be forgotten [21], which supports arguments in favor of periodically refreshing knowledge and skills.

While most evidence of learning through experience applies to clerical and manual work, individual performance in knowledge-intensive work is often associated with properties of both networks and ties [12]. Networks refer to whom a resource interacts with while performing their work, whereas ties refer to the nature of such relationships, e.g., whether resources have a personality similar to the one of the other resources with whom they have to work. Generally, performance and creativity in knowledge-intensive work is positively influenced by employees having a central position in organizational networks with strong ties, e.g., strong personality fit or matching work attitudes with other employees [7, 12].

3. Problem definition

An *event* in an event log records the execution of a particular *work item* in an individual execution of a business process, i.e., a process *case*. Each work item is an instance of a *task*.

An event log EL contains events. An event e is a tuple $e = (c, a, t, r, (d_1, v_1), \ldots, (d_m, v_m))$, where c is the case id, a is the task of which the work item[2] recorded by this event is an instance, t is the timestamp at which the event has been recorded, r is the resource that executed the work item and $(d_1, v_1), \ldots, (d_m, v_m)$, with $m \geq 0$, are other domain specific attributes and their values. For instance, the event $e = (45, \mathit{assess}, 2020.1.2, \mathit{Alice}, \mathit{amount} = 1000, \mathit{type} = \mathit{deep})$ captures the fact that, in a process case associated with loan request number 45, the resource Alice has executed a deep assessment of a loan request of 1000 USD on January 2nd, 2020. Note that, strictly speaking, the approach proposed in this paper requires, for each event, only the attributes c, a and r. The timestamps, in particular, are not required as long as the events in EL are ordered in time.

The universes of all events, tasks, and resources are denoted by $\mathcal{E}$, $\mathcal{A}$, and $\mathcal{R}$, respectively. We use a dotted notation to identify attributes of events, e.g., $e.c$ to identify the case id of event e. We refer to $A_{EL} \subset \mathcal{A}$ as the set of tasks that have at least one event in EL, i.e. $A_{EL} = \{\bar{a} \in \mathcal{A} : \exists e \in EL, e.a = \bar{a}\}$

The sequence of events generated in a given case form a trace $\sigma = [e_1, \ldots, e_n]$, where $\forall i \in [1, n]$, $e_i \in \mathcal{E}$, and $\forall i, j \in [1, n]$, $e_i.c = e_j.c$, i.e., all events belong to the same case; events in a trace must be ordered in time, using the timestamps when these are available. The universe of all traces is denoted by $\mathcal{S}$. The function $\mathit{trace} : \mathcal{E} \longrightarrow \mathcal{S}$ returns the trace σ to which an event e belongs, i.e., $\mathit{trace}(e) = \sigma$. Note that attributes of events e_i belonging to a trace σ may be the same $\forall e_i \in \sigma$. We refer to these attributes as case-level attributes. For instance, the amount requested in a loan request process is a case-level attribute. Attributes that can change for different events are called event-level attributes.

Given a trace σ of length n and an integer $l \leq n$, the prefix function returns the first l events of σ, that is, $\mathit{prefix}(\sigma, l) = [e_1, \ldots, e_l]$. A labeling function $y : \mathcal{S} \longrightarrow \mathcal{Y}$ is a function mapping a trace $\sigma \in \mathcal{S}$ to its class label $y(\sigma) \in \mathcal{Y}$, with $\mathcal{Y}$ being the domain of the class labels. For outcome predictions, $\mathcal{Y}$ is a finite set of categorical outcomes. In the context of this paper, we consider a binary outcome, i.e., $\mathcal{Y} = \{0, 1\}$. A label can also be associated to prefixes and all prefixes generated from a trace σ have the same class label (the one associated to σ). Similarly, we also define an event labeling function $\mathit{ye} : \mathcal{E} \longrightarrow \mathcal{Y}$ that associates to an event the label of the trace to which it belongs, i.e., $\mathit{ye}(e) = y(\mathit{trace}(e))$, $\forall e$.

In the specific case of outcome-based predictive monitoring, predictions are made using a classifier that takes as input a fixed number of independent variables (*features*) and learns a function to estimate the dependent variable (class *label*). This implies that, in order to use the data in an event log as input to a classifier, each trace in the log must be encoded as a feature vector. A sequence (or trace) encoder $f : \mathcal{S} \longrightarrow \mathcal{X}_1 \times \ldots \times \mathcal{X}_P$ is a function that takes a (partial) trace σ and transforms it into a feature vector in a P-dimensional vector space $\mathcal{X}_1 \times \ldots \times \mathcal{X}_P$ with $\mathcal{X}_p \subseteq \mathbb{R}, 1 \leq p \leq P$ being the domain of the p-th feature.

We divide features further into resource-aware and non-resource-aware. The former are generated using resource information in events, while the latter are generated without considering resource information. Let us assume that, among the P features generated from a (partial) trace σ, R features, with $R \leq P$, are non-resource-

[2] Note that, as defined by the lifecycle extension in the event log XES standard (`https://www.tf-pm.org/resources/xes-standard`), generally, in an event log, multiple events may refer to the same work item when each event captures a different state of the lifecycle of it (e.g., ready, in progress, terminated etc.). In this paper, and in the event logs that we considered for the evaluation, one event captures the execution of one work item in a process.

aware, whereas the remaining $(P-R)$ features are resource-aware. Given R non-resource-aware features with domains $\mathcal{X}_1, \ldots, \mathcal{X}_R$ and $(P-R)$ resource-aware features with domains $\mathcal{X}_{R+1}, \ldots, \mathcal{X}_P$, we distinguish between resource-aware and non-resource-aware classifiers for outcome-based predictive monitoring. A resource-aware classifier *rac* is a function that estimates the probability of a class label to be assigned to a feature vector including resource-aware features, i.e., $rac : \mathcal{X}_1 \times \ldots \times \mathcal{X}_R \times \mathcal{X}_{R+1} \times \ldots \mathcal{X}_P \longrightarrow \mathcal{Y}$. A non-resource-aware classifier *nrac* is a function that estimates the probability of a class label to be assigned to a feature vector that does not include resource-aware features, i.e., $nrac : \mathcal{X}_1 \times \ldots \mathcal{X}_R \longrightarrow \mathcal{Y}$.

One important element to model resource-aware features is the *handoff*, which captures the handover of work between resources executing consecutive events in a case. Formally, a handoff $h(r_1, r_2)$, with $r_1, r_2 \in \mathcal{R}$, is a relation that associates a resource r_2 that executed a given work item e_2 in a trace σ to the resource r_1 that executed the event e_1 preceding e_2 in σ, that is $h(r_1, r_2) \iff \exists e_1, e_2, \sigma : e_1, e_2 \in \sigma \wedge e_1.t < e_2.t \wedge e_1.r = r_1 \wedge e_2.r = r_2 \wedge \nexists e_3 \in \sigma : e_1.t < e_3.t < e_2.t$.

4. Approach

Section 4.1 introduces the framework for encoding resource experience, while Section 4.2 defines the dimensions of the framework. Section 4.3 formally defines the resource-aware features considered in this paper, whereas Section 4.4 finally draws a detailed comparison between the proposed framework with the Pika et al.'s one referenced earlier.

4.1. A conceptual framework for encoding resource-aware features

To identify relevant features for training machine learning models to predict case outcomes, we systematically asked the 5+1 "W" questions: Why, Who, Where, When, What, and How. We discarded the "Who" question because the "Who" is the resource itself. We also discarded the "Why" question ("Why has a resource been allocated to a work item?") because an event log does not directly contain information that would allow us to determine the reasons for a given allocation decision. In addition, in several settings, the allocation decision, as reflected in an event log, may be determined simply by the availability of resources.

The "When" question ("When has a resource performed a given task?") led us to identify `Recency` ("When was the most recent occasion when a resource performed a task?") as a source of features for capturing the notion of experience. The "Where" question ("Where or, more specifically, in which context did a resource perform a task?") led us to identify the `Context` as a possible source of features. By *context*, we mean the conditions under which a resource performed a given task. The "What" question led us to observe that resources gain experience in different ways, such as performing a given task or work item, or participating in the same case for which a prediction is being made. We introduce the `Target` dimension of resource experience to capture these multiple ways for resources to gain experience. Finally, the "How" question ("How much prior experience does a resource have?") led us to identify the `Aspect`, i.e., the measure used to quantify the resource experience in the scope identified using the previous dimensions. Accordingly, we retained four dimensions for encoding resource experience as a feature vector: `Recency`, `Context`, `Target`, and `Aspect` (cf. Fig. 1).

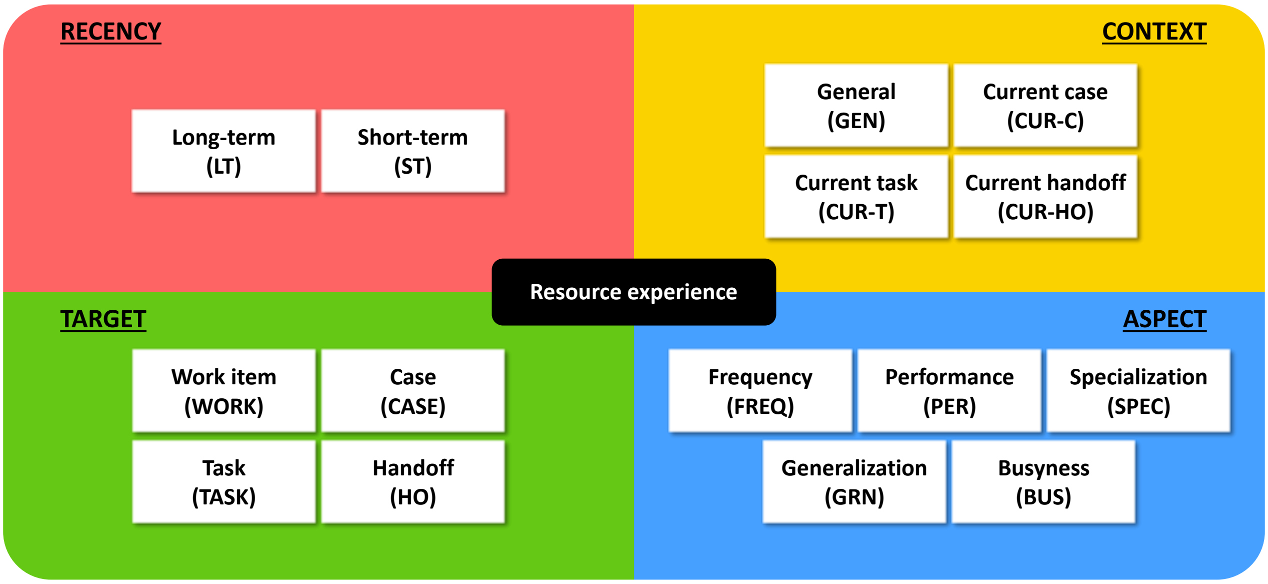


Figure 1: Resource experience framework

Note that the atomic actions that define experience in our framework are (i) execution of work items and (ii) involvement in handoffs. In other words, resources gain experience through executing work items and through interacting with other resources when receiving work from them. The former captures the experiential learning nature of resource experience, whereas the latter captures the importance of social ties in the development of the job experience.

Next, we discuss in detail the dimensions of the proposed framework. For each dimension, we identify the values that it can assume. Note that our objective is to define resource-aware features for each event, when processing a given prefix obtained from a trace.

4.2. Dimensions of resource-aware features

`Recency` (*rec*): Using the same skills and knowledge to execute work items over and over again keeps them fresh in our memory. At the same time, when skills are not practiced for some time, they start to fade away [21, 26]. For instance, a resource that performed a particular task two days ago is likely to perform the same task more efficiently today than another resource who last performed the same task a year ago. The `Recency` dimension of resource experience accounts for the time scope in which a resource has acquired a given experience. Regarding values, we distinguish between *long-term* recency, i.e., the considered time scope starts from the time of the earliest event in an event log, and *short-term* recency, i.e., the time scope is limited to a given time in the past until the timestamp of the current event. In principle, short-term recency may also be set considering other domain-specific criteria, such a recent number of terminated cases or a recent number of executed work items. In the evaluation, since we did not have any such domain-specific information for the event logs that we used, we set the value of recency to 30 days before the current event for all event logs.

`Context` (*con*): The management literature highlights that resources do not work in an organizational vacuum, but they are positioned within an organizational network, a certain technical environment, and specific organizational processes and policies [7, 12], all of which can influence the way in which they learn and perform their tasks. Based on this consideration, we propose that experience is gained by a resource when operating in the same *context*, such as repeating the same type of tasks within the same process case. For example, if a resource who has considerable experience in performing administrative tasks, such as checking the completeness of a loan application, is asked to compile a loan offer for the first time, she is likely to be less efficient in this task than another resource who has already compiled loan offers multiple times in the past. As far as the values of this dimension are concerned, context can capture the *general* experience of a resource, e.g., how many work items a resource has executed in a given process, but more importantly it can be restricted to a more specific domain, such as the *current case*, i.e., how much experience the resource has in the currently running case in executing work items, or the *current task*, i.e., how many times a resource has executed a work item corresponding to the same task as the current one, or the *current handoff*, i.e., how many times a resource has executed a work item following the same handoff as the current one.

Note that this list of values, which we consider in this paper, is not exhaustive, as other types of contexts can be relevant depending on the data available, such as the customer, the product type, the case type, or a cluster of (similar) tasks. Context can also be a combination of several attributes, e.g., to answer questions such as "How much experience does the resource have with the given task-handoff pair?". Furthermore, the context can be related to a (sub)sequence of work items or tasks, e.g., the sequence of tasks from the last three work items, or the tasks of a certain type, where task *type* is an attribute available in the event log.

`Target` (*tar*): This dimension concerns the type of atomic actions through which experience is gained by a resource. Based on the traditional theories of the learning curve [51] and experiential learning [20, 25], experience is gained by a resource by participating in some part of the process, i.e., executing a work item in it. Note that often resource experience and performance benefit also from executing a diverse range of tasks in a process [33, 42]. This is also implied by the theory of absorptive capacity at the individual level [40]. However, resources may also improve their experience by sharing insights with other resources during the execution of work. This can be fostered by a personality fit or similar work attitudes with other resources [12]. Since in an event log resource information is only associated with individual work items, we assume that resources interact when they execute consecutive work items, i.e., being involved in a handoff. Therefore, we define the following values for the target dimension:

- Work Item: experience is gained by executing any work item;
- Case: experience is gained by being involved in a case (executing at least one work item in it);
- Task: experience is gained by executing work items(s) of a given task;
- Handoff: experience is gained by interacting with other resources, i.e., being involved in a handoff.

`Aspect` (*asp*): While, in general, resource experience is positively correlated with resource performance, that may not always be the case in specific situations. For instance, the experiential learning theory highlights that, in some situations, repetitive tasks may put an excessive mental strain on a resource, which may result in a decrease of performance [20]. As far as workplace ties are concerned, the need to interact with other resources with a conflicting personality or a different work attitude may also lead to decreased performance [12]. Therefore, simply considering the frequency at which a resource has been involved in an atomic action (i.e., a target) through which experience is gained is not likely to capture the full extent of the impact of resource experience on resource performance and, consequently, process performance.

In the proposed framework, given a target of resource experience, evaluated in a given context and in a given time scope (recency), there can be multiple aspects that we want to encode into features. As mentioned above, a first aspect to be considered is the *frequency* at which a given target of experience occurs for a given resource, e.g., counting the number of times a resource executed a particular task, or counting the number of work items performed by a resource in a process case, in a given period of time. However, other aspects that link the resource experience to the type of experience and the level of process performance achieved may be relevant in a specific context. We, therefore, consider the following additional aspects:

- Performance: this aspect concerns the outcome achieved by the cases in which a resource has gained experience and can be calculated as the ratio between the number of cases with a positive outcome and the number of cases with a negative one;
- Specialization and Generalization: these aspects concern the extent to which experience gained by a resource is concentrated on a specific target, i.e., a specific case, task, or work item, or diluted across multiple targets;
- Busyness: this aspect concerns the extent to which the experience gained is concentrated in the time scope defined by the Recency dimension. While the Frequency aspect considers the absolute number of atomic actions occurred in a given a time scope for a resource, this aspect relates this number of occurrences to the duration of the considered time scope.

4.3. Definition of resource-aware features

Figure 2 gives an overview of how events in an event log are encoded for predicting outcomes. In this section, our objective is to define the resource-aware features[3] generated when processing the current event e_i (event e_2 of case 3 in Figure 2), of a given trace σ, referred to as the *current case* in the remainder of this section (case 3 in Figure 2). To generate the features for e_i, in particular the resource-aware ones, which are inter-case, we must consider the events in the log with timestamps that come before the one of e_i (events in red on the top-right side of Figure 2). First, we identify the targets of the current event, i.e., its resource, case id, task, and handoff. Then, features are generated. Next, we define in detail how the resource-aware features are calculated based on the current case id, resource, task and handoff.

Note that, for simplicity, Figure 2 does not consider features generated from case-level attributes. Also, let us clarify that non-resource-aware features are calculated using traditional methods from the literature, e.g., [50], and include features encoding, for instance, the task label or any other domain specific attribute characterizing

[3] A detailed running example exemplifying the calculation of the features defined in this section is available at `https://github.com/jckim9203/resource-aware-predictive-monitoring`(DOI: 10.5281/zenodo.5081282)

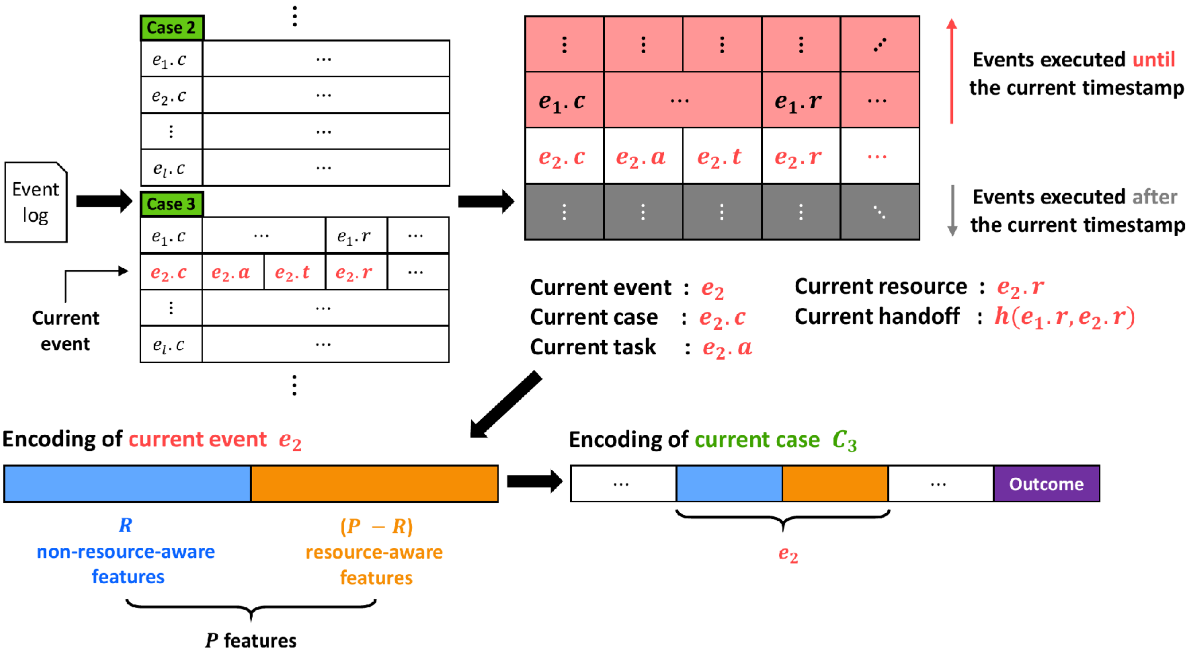


Figure 2: Extracting resource-aware features from event logs

e_i. The features associated with the events of a case are finally aggregated into a feature-label vector encoding the current case. The resource-aware features associated with e_i are described in detail next.

By combining the values of the four dimensions of resource experience, a number of resource-aware features can be generated. Note that not all possible combinations of values lead to meaningful features. In the following, we discuss in detail the ones that we consider in this paper.

Next, we group the definitions of resource-aware feature by the value of the *aspect* dimension. For the sake of conciseness, we only consider the value long-term for the Recency dimension. Feature definitions for $rec = short - term$ can be simply derived by reducing the time scope from the general one, i.e., from the time of the earliest event in an event log until the timestamp $e_i.t$ of the current event, to a given recent time window defined by a given value Δ_t, i.e., $[e_i.t - \Delta_t, e_i.t]$.

Frequency aspect. The resource-aware features considered for the frequency aspect ($asp = frequency$) are shown in Table 1. The ones for $con = general$ count the number of work items/cases/tasks/handoffs in which the resource $e_i.r$ has been involved until $e_i.t$. Additional features are obtained by restricting the context to the current case, task or handoff.

Performance aspect. Given that we consider binary class labels, i.e., $\mathcal{Y} = \{0, 1\}$, performance can be calculated using the polarity of the traces $S \subset \mathcal{S}$ (or events $E \subset \mathcal{E}$) obtained from the application of the other dimensions in our framework, that is:

$$polarity(S) = \frac{\sum_{\sigma \in S} y(\sigma)}{|S|}$$

Table 1: Resource-aware features for $asp = frequency$

Feature	Target	Context	Definition
n_work_items	work item	general	Number of work items executed by $e_i.r$ until $e_i.t$
n_cases	case	general	Number of cases in which $e_i.r$ executed at least one work item until $e_i.t$
n_tasks	task	general	Number of unique tasks executed by $e_i.r$ until $e_i.t$
n_ho	ho	general	Number of unique handoffs involving $e_i.r$ executed by $e_i.r$ until $e_i.t$
n_curr_case	work item	curr_case	Number of work items executed by $e_i.r$ until $e_i.t$ in the current case (i.e., having the same value of $e_i.c$)
n_curr_task	work item	curr_task	Number of work items executed by $e_i.r$ until $e_i.t$ having task equal to $e_i.a$
n_curr_ho	work item	curr_ho	Number of work items executed by $e_i.r$ until $e_i.t$ having handoff equal to $h(e_{i-1}.r, e_i.r)$

Table 2: Resource-aware features for $asp = performance$

Feature	Target	Context	Definition
per_case	case	general	Polarity of cases involving $e_i.r$ until $e_i.t$
per_work_item	work item	general	Polarity of work items involving $e_i.r$ until $e_i.t$
per-curr-task	work item	curr-task	Polarity of work items involving at least one work item of task $e_i.a$ executed by $e_i.r$ until $e_i.t$
per-curr-ho	work item	curr-ho	Polarity of work items involving one handoff equal to $h(e_{i-1}.r, e_i.r)$ until $e_i.t$

$$polarity(E) = \frac{\sum_{e \in E} ye(e)}{|E|}.$$

For instance, given E as the set of all work items executed from the earliest timestamp in an event log until the current timestamp, (*rec* = *long-term, con* = *general, tar* = *work item*), the polarity calculates the fraction of events in E having class label equal to 1.

The resource-aware features considered for the performance aspect ($asp = performance$) are shown in Table 2. Since polarity can only be defined at the level of set of cases or set of events, these features can only consider the values *tar* = *work item* and *tar* = *case* for the target dimension. Note that the value of polarity features for target *tar* = *case* differ from *tar* = *work item* if a resource executes more than one work item in at least one case.

Specialization and generalization aspects. Specialization and generalization concern assessing whether a resource is specialized in performing a particular task, or possesses more general experience, i.e., has performed several tasks with rather uniform frequencies in the same process. Specialization is calculated as the *ratio* between the experience of a resource related to a given target in a specific context and the experience of the resource related to the same target in a more general context, e.g., the ratio between the number of work items performed by

Table 3: Resource-aware features for $asp = specialization$

Feature	Target	Context	Definition
sp-work_item-case	work item	general	A/B, with A the number of work items performed by $e_i.r$, B the number of cases executed until $e_i.t$
sp-curr-case	work item	curr-case	A/B, with A the number of work items performed by $e_i.r$ in case $e_i.c$, B the number of work items executed by $e_i.r$ until $e_i.t$
sp-curr-task	work item	curr-task	A/B, with A the number of work items performed by $e_i.r$ corresponding to task $e_i.a$, B the number of work items executed by $e_i.r$ until $e_i.t$
sp-curr-ho	work item	curr-ho	A/B, with A the number of work items performed by $e_i.r$ with handoff equal to $h(e_{i-1}.r, e_i.r)$, B the number of work items executed by $e_i.r$ until $e_i.t$

Table 4: Resource-aware features for $asp = generalization$

Feature	Target	Context	Definition
gen-task	task	general	Entropy of distribution of tasks executed by $e_i.r$ until $e_i.t$
gen-case	case	general	Entropy of distribution of cases executed by $e_i.r$ until $e_i.t$
gen-ho	ho	general	Entropy of distribution of handoffs executed by $e_i.r$ until $e_i.t$

a resource in a given case and the total number of work items executed by that resource in the entire process determines the specialization of a resource related to the *target* work item in the *context* of the current case. Generalization is calculated as the entropy of the distribution of a given a target, e.g., the entropy of the distribution of all tasks executed by a resource until the current event. Given a target $T = \{t_k\}$, e.g., a list of tasks, the entropy of T is calculated as follows:

$$entropy(T) = -\sum_k p_T(t_k) \cdot log(p_T(t_k))$$

where $p_T(t_k)$ is the relative frequency of t_k in T.

The definitions of the resource-aware features considered for the specialization and generalization aspects are shown in Tables 3 and 4, respectively.

Busyness aspect. Busyness features are calculated by dividing a feature value (obtained as a frequency) by the time span in which that feature value was calculated. For instance, given the number of work items performed by a resource r, the busyness of r can be calculated as the ratio between this number and the difference between the latest and the earliest timestamp of the tasks executed by r.

As far as feature definitions are concerned, for busyness we consider one single feature *busyness* with $tar = work\ item$ and $con = general$, defined as the number of work items executed by $e_i.r$ until $e_i.t$ divided by the time, calculated in days, between the earliest timestamp in the event log and the current timestamp $e_i.t$.

To conclude, for a given event, there are a total of 38 resource-aware features, i.e., the 19 described in this section, plus the dual 19 features obtained by considering the recency value $rec = short - term$.

4.4. Detailed comparison with Pika et al.'s framework

To conclude this section, we can now compare more in depth the proposed resource-aware features with the resource profiles proposed by Pika et al. [41]. Many of the resource-aware features that we propose can be directly subsumed from resource profiles. In particular, this applies to all features involving the frequency of work item executions and resource involvement in cases. However, while resource profiles are calculated only using data of completed cases, the resource-aware features that we propose are designed to be calculated for each new event executed in a trace. In addition, the proposed features consider aspects of resource behavior that are not considered by resource profiles, such as the polarity of process outcomes, the entropy, the busyness and the number of work items having the same value of case/task/handoff as the current event. We also consider more detailed features involving work handoffs among resources, whereas resource profiles only consider the total number of handoffs involving a given resource.

In addition, some of the resource profiles are calculated using domain specific attributes that normally are unavailable in event logs, such as customer feedback or task creator, or are suitable only in specific scenarios. Conversely, the resource-aware features that we propose require attributes that are universally available in event logs, such as case id, activity and resource attributes. Examples of resource profiles that are suitable in specific scenarios are found in the *Preferences* category. For example, the *New attribute values* profile in the *Preferences* category refers to an attribute value never seen before in any prior event. In reality, an event log may fail to capture all the historic events pertaining to the corresponding process, for instance because it logs events only belonging to a specific time period. Therefore, it is uncertain whether an attribute value unseen before the current event is actually a completely new one. Also, the *Activity reassignments* profile is defined as the number of occurrences of an activity initiated by a given resource and eventually completed by a different one. This assumes that an event log contains information about a work item execution lifecycle, which is often not the case in reality. In addition, this situation does not necessarily model a resource reassignment as, in some cases, tasks may require multiple resources to sequentially or simultaneously work for their successful completion.

5. Evaluation

The first objective of the evaluation is to assess to what extent and/or in which situations the features capturing resource experience defined in this paper increase the performance of outcome-oriented predictive process monitoring. Then, a second objective is to analyse more in detail the explanatory power of resource-aware features. To do so, we conduct an explainability analysis, to understand the degree to which different types of resource-aware features contribute to the predictions made by a model.

We conducted a set of experiments[4] using different event logs publicly available. For each event log we compare the performance achieved by the model trained using only non-resource-aware features (that is, *nrac*) and the one that uses also resource-aware features (*rac*), all other conditions staying equal. As a performance

[4] The code used for the experiments is publicly available for testing and experiments repeatability at: `https://github.com/jckim9203/resource-aware-predictive-monitoring` (DOI: 10.5281/zenodo.5081282)

measure we consider the AUC. The AUC is the area under the Receiver Operating Characteristic (ROC) curve, which is constructed based on true positive and false positive rates. It has been considered extensively as a performance measure in existing research on predictive process monitoring [50, 52] because it tends to remain unbiased even with imbalanced class labels.

In all the experiments, we consider zero-bucketing of traces and index-based sequence encoding:

- Bucketing in predictive process monitoring [50] is the practice of dividing the encoded prefixes into buckets, i.e., groups, and then training a separate classifier using each bucket of prefixes. Zero-bucketing, which we choose in this work, refers to a default setting in which encoded prefixes are not divided into groups and, therefore, only one classification model is trained with all the encoded prefixes from traces in an event log. In this configuration, prefixes of different length are encoded into feature vectors of different lengths and zero-padding is applied when necessary to bring all feature vectors to the same length for training/testing the classifier.

- Prefixes in predictive process monitoring can be encoded in different ways. A typical solution, which we adopt in this work, is the index-based sequence encoding [28], whereby each event is encoded into a sequence of features derived from its attributes. Case-level attributes are encoded only once for each prefix. Index-based encoding is opposed to aggregation encodings, in which the features for encoding a case (or a prefix) can be derived by aggregating the values of the attributes of different events (for instance, the values of a numeric attribute across different events of a case may be aggregated into a single feature by using their average). Index-based encoding naturally fits the proposed framework to generate resource-aware features, since these features, by definition, characterize each event in a prefix.

Non-resource-aware features, including the ones derived from the case-level attributes, are generated using the same pre-processing scripts of Teinemaa et al. [50]. A set of 38 resource-aware features are generated for each event in a prefix as discussed in the previous section. As classifiers, we consider random forest (RF) and Xgboost (XGB), which have emerged as the best performing classifiers across different event logs in the benchmark published by Teinemaa et al. [50].

We also present an in-depth feature contribution analysis using SHAP (Shapley Additive Explanations) [31]. Based on the coalitional game theory, SHAP is able to calculate the degree to which each feature contributes on average to a prediction. SHAP is a robust explainable AI technique commonly used in particular to interpret black box models, such as deep learning models. Since, in this work, we use RF and XGB, in the experiments, we have used TreeSHAP [30], a fast implementation of the SHAP method fit for tree-based classification models.

The metrics that we use in the feature contribution analysis are shown in Figure 3. After having ordered all the features in descending order by contribution (calculated using the absolute value of the TreeSHAP's feature importance), we first calculate the sum of all the feature contributions and then we consider the top-contributing ones in a given percentile. For instance, in Figure 3, the 75% percentile of feature contribution includes the features F_1, F_3, F_2, F_6. For a given percentile, in the evaluation we consider (i) the ratio of features belonging to it that are resource-aware (50% in the 75% percentile in the figure) and (ii) the ratio of the

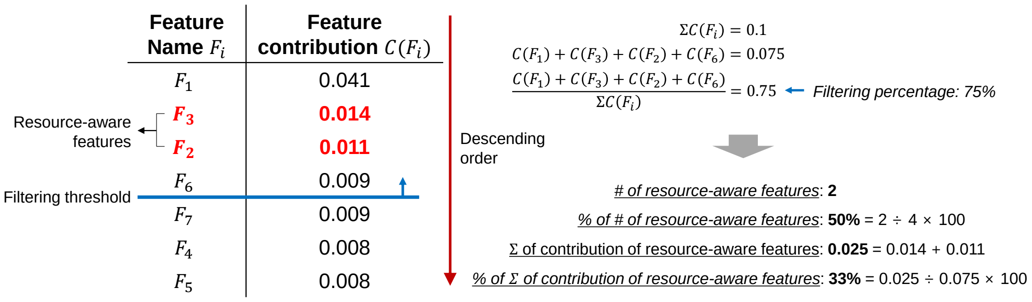

Feature Name F_i
Feature contribution $C(F_i)$
F_1 0.041
F_3 0.014
F_2 0.011
F_6 0.009
F_7 0.009
F_4 0.008
F_5 0.008
Resource-aware features
Filtering threshold
Descending order
$\Sigma C(F_i) = 0.1$
$C(F_1) + C(F_3) + C(F_2) + C(F_6) = 0.075$
$\frac{C(F_1) + C(F_3) + C(F_2) + C(F_6)}{\Sigma C(F_i)} = 0.75$
Filtering percentage: 75%
# of resource-aware features: 2
% of # of resource-aware features: 50% = 2 ÷ 4 × 100
Σ of contribution of resource-aware features: 0.025 = 0.014 + 0.011
% of Σ of contribution of resource-aware features: 33% = 0.025 ÷ 0.075 × 100

Figure 3: Illustration of the metrics considered in the feature contribution analysis

Log	Process description	Outcome description	# cases	# events	Examples of domain-specific features
BPIC2011_1	Treatment and diagnosis process in the Gynaecology department of a Dutch academic hospital	Temporal constraint satisfaction on the order of occurrence of tasks in a case	1,140	67,480	Diagnosis code, Treatment code, Specialism code, Age
BPIC2011_2			1,140	149,730	
BPIC2011_3			1,121	70,546	
BPIC2011_4			1,140	93,065	
BPIC2012_accepted	Application for a personal loan or overdraft at a global financing organization	Whether an application is (accepted, cancelled, or declined); or not	4,685	186,693	Amount requested, Lifecycle of an application
BPIC2012_cancelled			4,685	186,693	
BPIC2012_declined			4,685	186,693	
BPIC2015_1	Application for building permit at 5 Dutch municipalities	Temporal constraint satisfaction on the order of occurrence of tasks in a case	696	28,775	Application cost, Construction, Area protection, Entrance/Way out
BPIC2015_2			753	41,202	
BPIC2015_3			1,328	57,488	
BPIC2015_4			577	24,234	
BPIC2015_5			376	1,051	
Road Traffic Fines	Managing fines punishing road traffic infractions, at an Italian regional agency for traffic management	Whether a fine is repaid in full or sent for credit collection	129,615	460,556	Article, Vehicle class, Infraction type, Fine amount

Table 5: Event logs considered in the evaluation

cumulative contribution of the resource-aware features belonging to it (in the figure, 33%). The former provides the absolute number of resource-aware features that are contributing to the predictions. The latter gives a more detailed insight on the relative magnitude of the contribution of these features.

5.1. Datasets and experimental settings

For our experiments, we have considered all the datasets used in the benchmark published by Teinemaa et al. [50] and containing the *resource* attribute. In particular, we have used the event logs of the Business Process Intelligence Challenge 2011 (BPIC 2011, 1 dataset with 4 different labelings, which we treat as 4 different datasets in this section), the Business Process Intelligence Challenge 2012 (BPIC 2012, 1 dataset with 3 different labelings, which we treat as 3 different datasets) and the Business Process Intelligence Challenge 2015 (BPIC 2015, 5 event logs). Additionally, we have considered the Road Traffic Fines (RTF) event log. All the event logs are publicly available at `https://data.4tu.nl/`. Table 5 reports the descriptive statistics of these logs, briefly describing also the type of process generating the log, the outcome label to be predicted, and examples of domain-specific information from which non-resource-aware features are derived. Note that some event logs have more than one label (4 for BPIC 2011 and 3 for BPIC 2012). For these, we consider a different log for each label available. For BPIC 2011, 19 cases are missing the 3rd label. Also, the number of events for the BPIC 2011 event logs is different for each label because the cases are cut at the prefix in which the satisfaction/violation of the constraint determining the label becomes known [50]. More details about the labels used is given later while discussing the results.

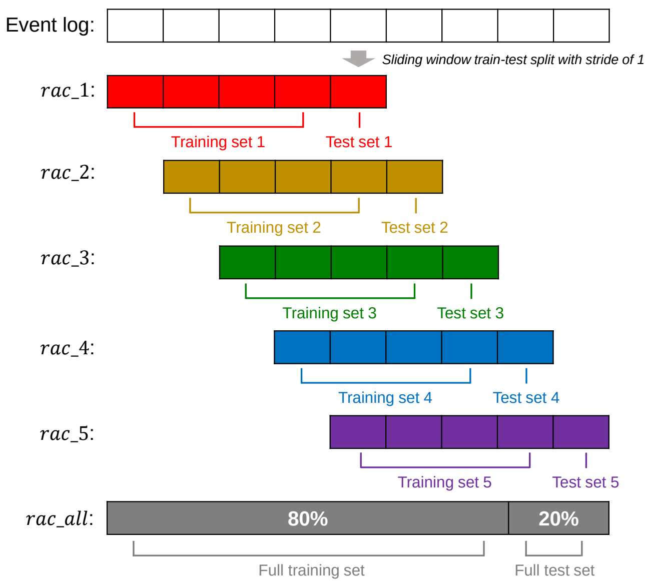


Figure 4: Temporal split-based cross-validation procedure

Outcome-oriented predictive process monitoring is an instance of the early time-series classification problem, whereby its aim is to predict the correct outcome of a case prefix as soon as possible after its beginning. Therefore, the performance of predictive models is usually evaluated on early (i.e., short) prefixes. In this work, similarly to others in the literature [52, 50], we consider prefixes generated using up to the first 20 events for the BPIC event logs and up to the first 10 events for the RTF log.

Because of the time-series nature of event log data, in the experiments we consider a temporal split of traces [49]. Given a split timestamp t_s, the training set contains the traces for which the first event occurs before the split timestamp ($e_1.t < t_s$), whereas the test set contains the traces for which $e_1.t > t_s$. The split timestamp t_s is chosen by imposing a given ratio between traces in the training/test sets (e.g., 80%/20%) The events e of the traces in the training set with $e.t > t_s$ are then discarded. Such a temporal split is required to ensure that a model is not trained using events that happen in the *future* with respect to the ones used for testing.

To account for the variability of traces across time and achieve robust results, we also implemented a temporal split-based cross-validation procedure, which is depicted in Figure 4. An event log is split into nine batches containing roughly the same number of prefixes using the temporal split mechanism described above. Then, we train and test five models (rac_1-rac_5). Each model uses four consecutive batches for training and the next batch for testing (i.e., using a *stride* of 1). An additional model (rac) is trained and tested using the whole event log, i.e., using 80% of traces for training and 20% for testing after applying the temporal split.

As classifiers, we consider the Python implementation of RF and XGB in the packages `scikit-learn` and `xgboost`, respectively. For the hyperparameter configurations, we use the Tree-structured Parzen Estimator (TPE). For RF, TPE has been applied to find the ideal value of the hyperparameter $max\_features$ in the interval $max\_features \in [0, 1]$. For XGB, TPE has been applied to find the 5 hyperparameters $learning\_rate$, sub_sample, max_depth, $colsample\_bytree$ and min_child_weight in the intervals $learning\_rate \in [0, 1]$, $subsample \in [0.5, 1]$, $max\_depth \in \{x|x \in \mathbb{N}, 4 \leq x \leq 30\}$, $colsample\_bytree \in [0.5, 1]$ and $min\_child\_weight \in \{1, 2, 3, 4, 5, 6\}$.

	BPIC 2015_1	BPIC 2015_2	BPIC 2015_3	BPIC 2015_4	BPIC 2015_5
Mean case duration	96 days	160 days	63 days	111 days	101 days
Median case duration	63 days	115 days	39 days	92 days	80 days
Number of cases	696	753	1,328	577	1,051
Number of resources	22	9	18	12	15
Mean number of events in a case	41	55	43	42	52

Table 6: Detailed descriptive statistics of BPIC 2015

The values obtained from the hyperparameter optimization are available in the github repository.

5.2. Results

The impact of the resource-aware features on the model performance and their contribution to the prediction with respect to the non-resource-aware ones differ for each event log analysed. Therefore, this section includes a separate discussion for each (set of) event log(s).

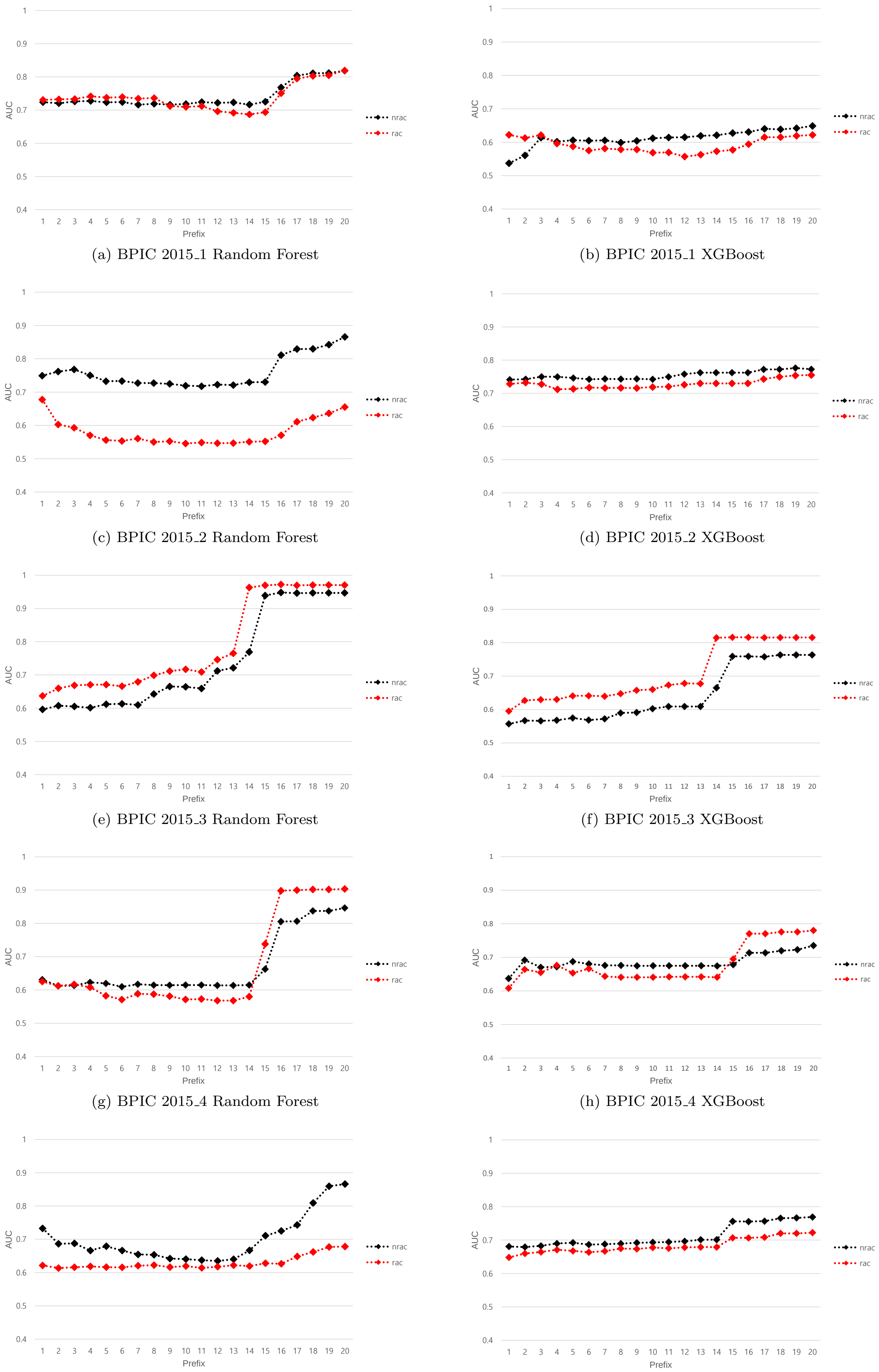
BPIC 2015 event logs

Figure 5 shows the performance obtained by the models that use (*rac*) or do not use (*nrac*) the resource-aware features for prefixes of different lengths using RF and XGB. The relative performance of the two models varies greatly depending on the event log considered.

In particular, there are two cases worth noticing: the second log, in which *rac* is consistently worse than *nrac*, and the third log, in which *rac* is better than *nrac*. The reason behind these different behaviors could be due to the fact that the two event logs refer to process execution contexts that are completely different. According to van der Ham [18], who presented an in-depth analysis of the different logs used for the BPIC 2015, the second log of this set refers to a municipality executing the process in a very inefficient way, whereas the third log refers to a municipality executing the same process efficiently. This performance difference is also acknowledged by the data shown in Table 6, where we can see that the mean and median case duration of the process in the third log are considerably lower than the ones in the second log. Based on these results, we can conclude that, in this experiment, using the resource-aware features appears to be effective when the process is executed efficiently, whereas it can be detrimental to the predictive performance when the process is executed inefficiently. Specifically, modeling resource experience in this particular context helps building better predictors when resources are working efficiently.

Figure 6 shows the performance obtained using the models trained on the temporally-split batches. The results show a consistent difference in terms of performance obtained by these models. This may highlight that the impact of the resource-aware features varies over time since also the experience of the resources varies.

Figure 7 shows the ratio of the top-contributing resource-aware features and the ratio of their cumulative contribution for different percentiles for the *rac* model. From the plots, we can notice that, for all percentiles, the resource-aware features contribute more to the predictions in the case of third event log in the BPIC 2015 set of logs, which is the one for which the *rac* model performs better than *nrac*. In addition, for the second and the third event logs, the contribution magnitudes of the resource-aware features tend to converge to a value of 40% for the 90% percentile. This means that, when almost all the features are considered, the contribution of



(a) BPIC 2015_1 Random Forest

(b) BPIC 2015_1 XGBoost

(c) BPIC 2015_2 Random Forest

(d) BPIC 2015_2 XGBoost

(e) BPIC 2015_3 Random Forest

(f) BPIC 2015_3 XGBoost

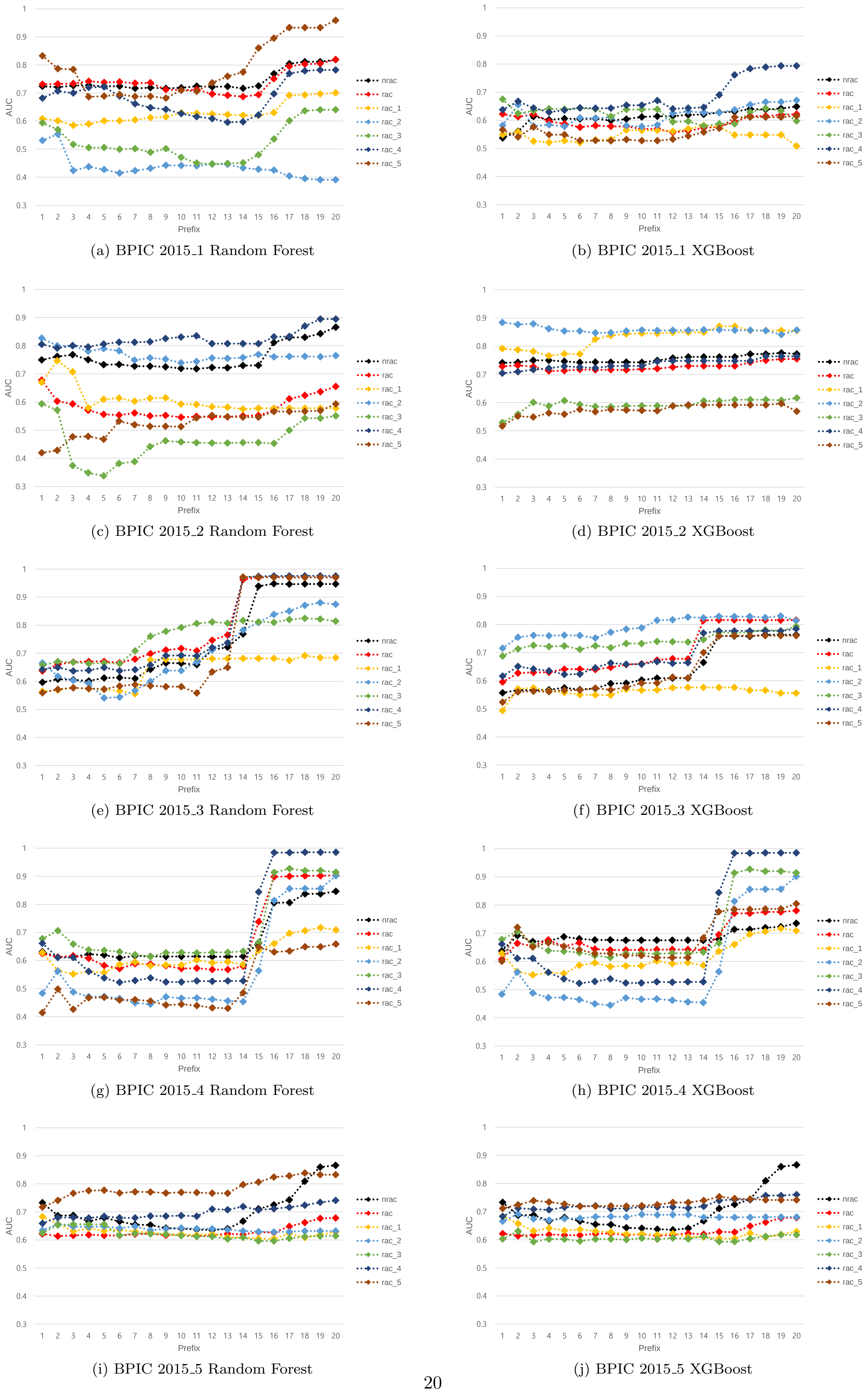
(g) BPIC 2015_4 Random Forest

(h) BPIC 2015_4 XGBoost

(i) BPIC 2015_5 Random Forest

(j) BPIC 2015_5 XGBoost

Figure 5: AUC for different prefix lengths for the BPIC 2015 event logs



(a) BPIC 2015_1 Random Forest

(b) BPIC 2015_1 XGBoost

(c) BPIC 2015_2 Random Forest

(d) BPIC 2015_2 XGBoost

(e) BPIC 2015_3 Random Forest

(f) BPIC 2015_3 XGBoost

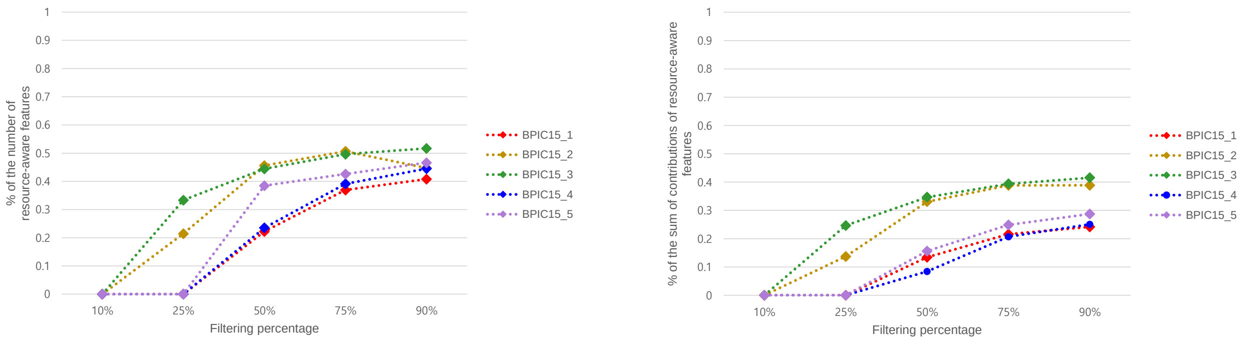
(g) BPIC 2015_4 Random Forest

(h) BPIC 2015_4 XGBoost

(i) BPIC 2015_5 Random Forest

(j) BPIC 2015_5 XGBoost

Figure 6: AUC for different prefix lengths for the BPIC 2015 event logs using the temporally-split batches

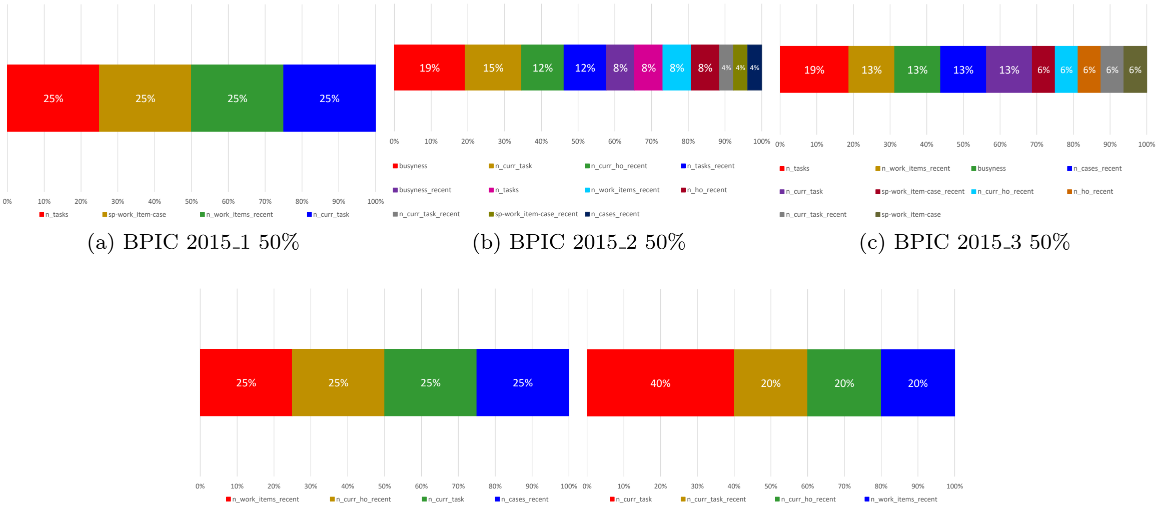


(a) Ratio of the top-contributing resource-aware features

(b) Ratio of the cumulative contribution of the top-contributing resource-aware features

Figure 7: Ratio of the number and cumulative contribution of the top-contributing resource-aware features in the BPIC 2015 logs

the resource-aware features to the prediction becomes rather high for these two logs. However, this contribution leads to wrong predictions in the case of the second log, thus decreasing the performance of the model.



(a) BPIC 2015_1 50%

(b) BPIC 2015_2 50%

(c) BPIC 2015_3 50%

(d) BPIC 2015_4 50%

(e) BPIC 2015_5 50%

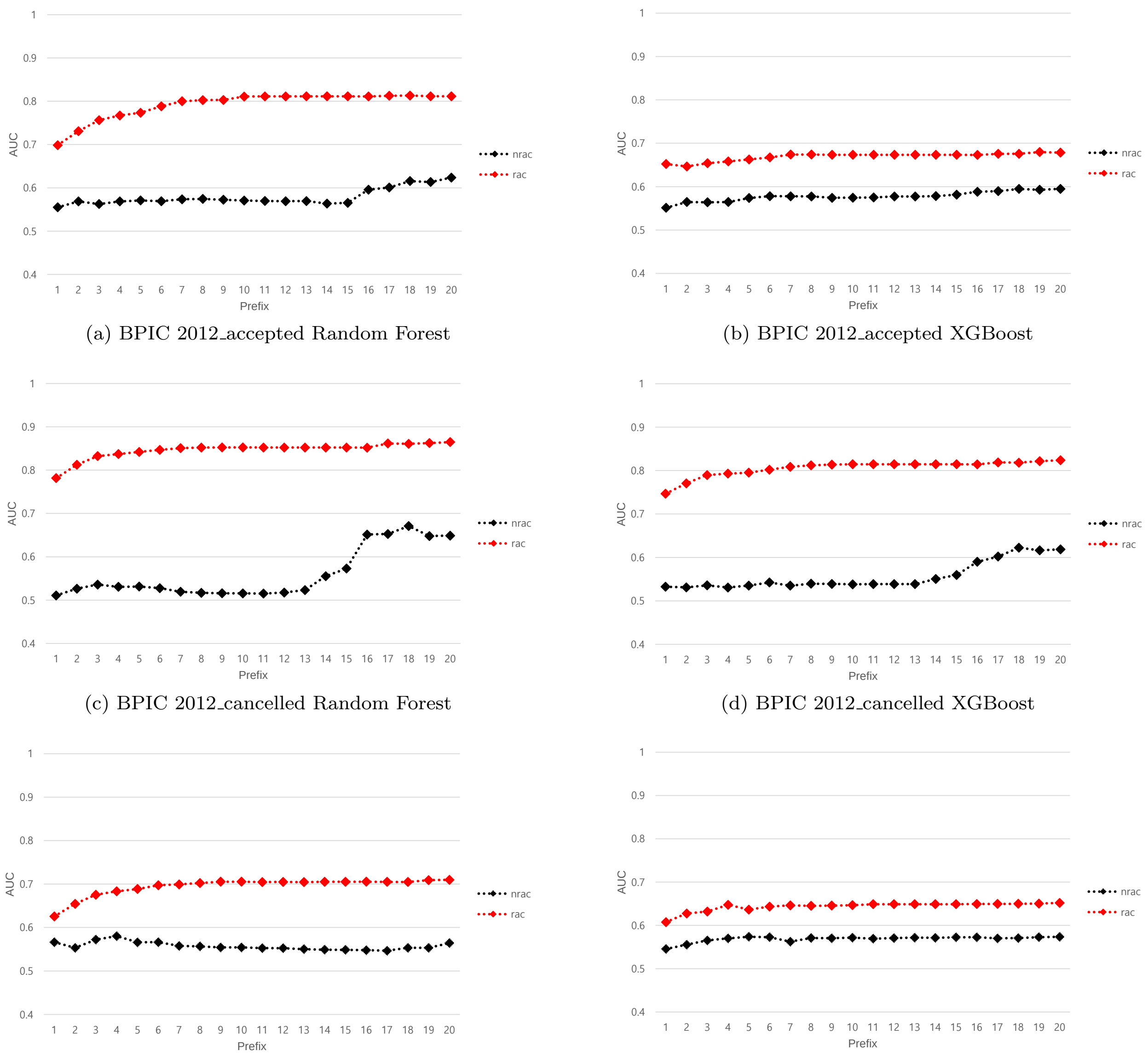
Figure 8: Distribution of the top-contributing resource-aware features for the 50% percentile in the BPIC 2015 logs

Figure 8 shows the distribution of the contributions of different resource-aware features within the 50% percentile across the dimensions defined in Section 4. Note that most of the features shown in the figure refer to the *frequency* aspect of the framework, e.g., counting the number of cases or work items in which the resource of the current event has been involved. Since this type of features mainly refer to the *experiential learning* of the resource behavior, we can conclude that, in this particular context, the experience accumulated by the resources is highly contributing to explain the predictions made by the models.

As a last remark, we have to consider that, for the BPIC 2015 event logs, the label to be predicted is the satisfaction of the following temporal constraint: the activity "send confirmation receipt" must not be eventually followed by "retrieve missing data" (that is, a confirmation should not be sent to the client if additional information to close a case is still needed). This constraint appears to refer to standard working procedures that could easily be learnt by a human resource participating in the process and may not be directly linked to the efficacy and effectiveness of the resources in executing their job. This aspect may also justify the fluctuating performance achieved with these logs.

BPIC 2012 event logs

Figure 9 shows the performance obtained by *rac* and *nrac* for different prefix lengths with RF and XGB. Figure 10 shows the performance of the models trained using the temporally-split batches.



(a) BPIC 2012_accepted Random Forest

(b) BPIC 2012_accepted XGBoost

(c) BPIC 2012_cancelled Random Forest

(d) BPIC 2012_cancelled XGBoost

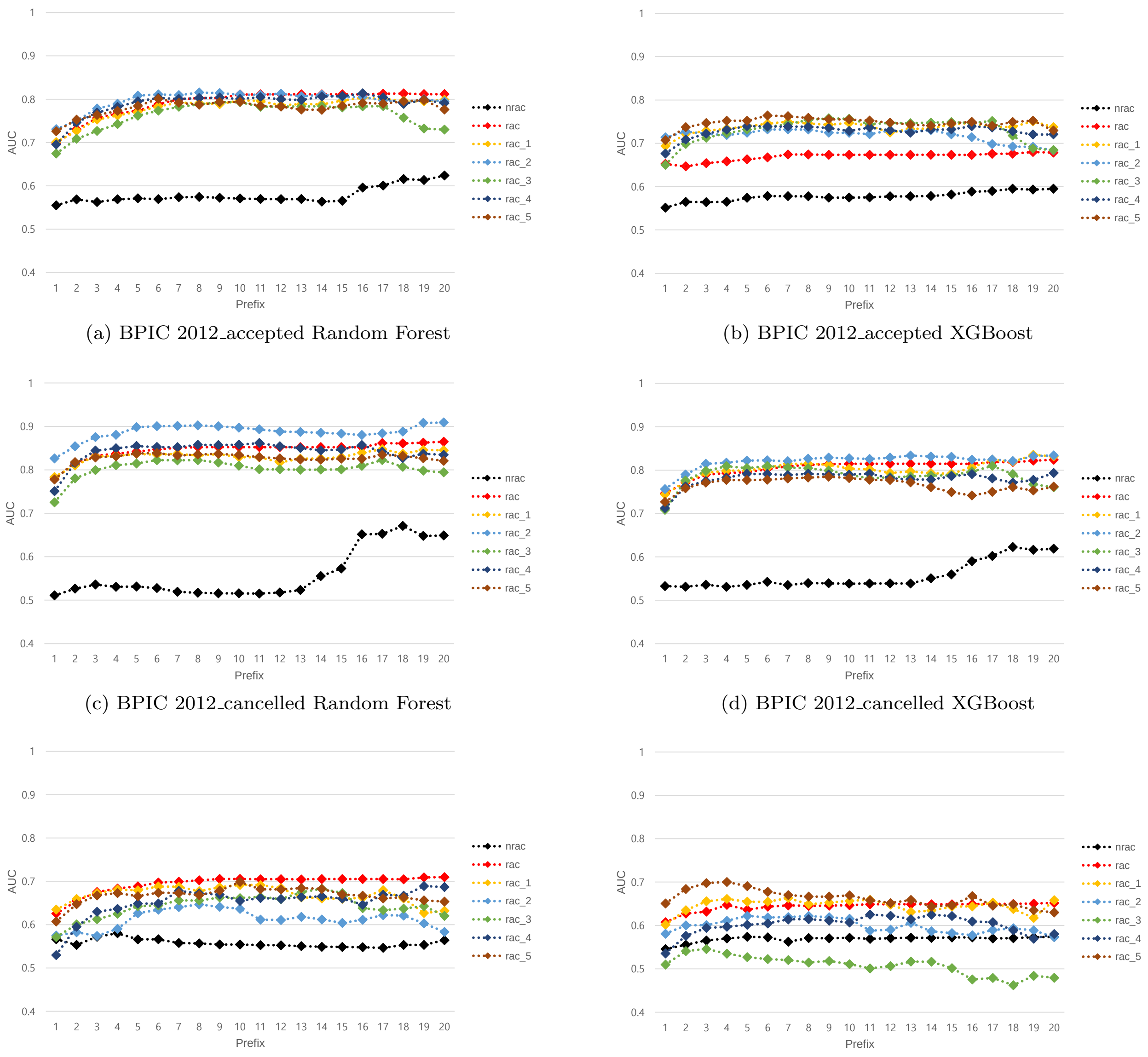
(e) BPIC 2012_declined Random Forest

(f) BPIC 2012_declined XGBoost

Figure 9: AUC for different prefix lengths for the BPIC 2012 event logs

Two insights emerge clearly: with the BPIC 2012 event logs, the models trained using the resource-aware features are more accurate and, particularly in the case of the *accepted* and *cancelled* event logs, there is not much variability in the performance of the models trained using traces belonging to different time batches.

The positive influence of the resource-aware features on the performance of the classifiers, in this case, may be due to the fact that the label to be predicted explicitly captures the outcome of the process, i.e., whether a request will be accepted, cancelled, or declined, and therefore may be directly influenced by the experience of the resources in the process. This conclusion is supported by the fact that according to Adriansyah and Bujis



(a) BPIC 2012_accepted Random Forest

(b) BPIC 2012_accepted XGBoost

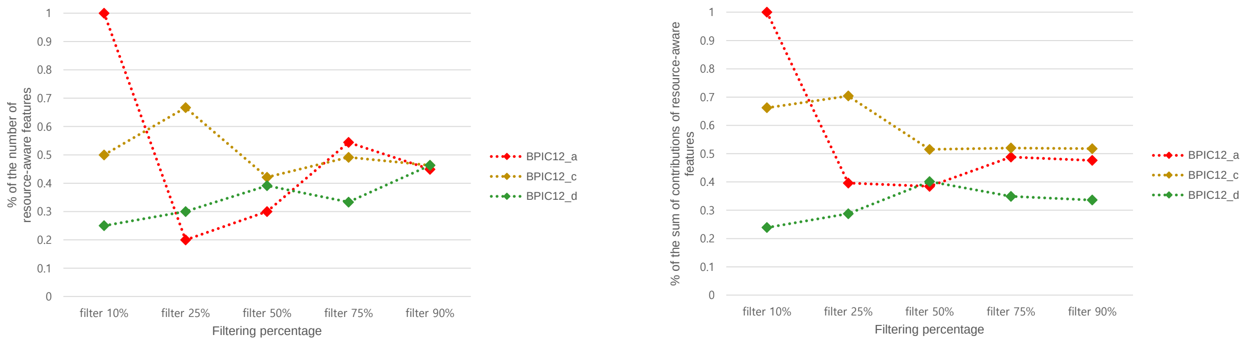
(c) BPIC 2012_cancelled Random Forest

(d) BPIC 2012_cancelled XGBoost

(e) BPIC 2012_declined Random Forest

(f) BPIC 2012_declined XGBoost

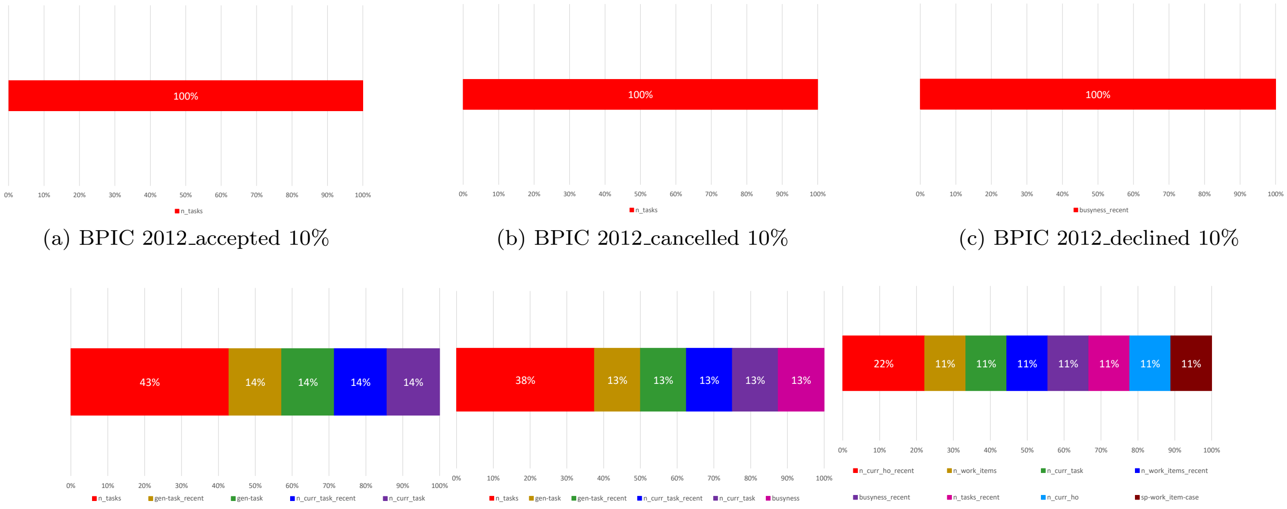
Figure 10: AUC for different prefix lengths for the BPIC 2012 event logs using the temporally-split batches



(a) Ratio of the top-contributing resource-aware features

(b) Ratio of the cumulative contribution of the top-contributing resource-aware features

Figure 11: Ratio of the number and cumulative contribution of the top-contributing resource-aware features in the BPIC 2012 logs



(a) BPIC 2012_accepted 10%

(b) BPIC 2012_cancelled 10%

(c) BPIC 2012_declined 10%

(d) BPIC 2012_accepted 50%

(e) BPIC 2012_cancelled 50%

(f) BPIC 2012_declined 50%

Figure 12: Distribution of the top-contributing resource-aware features for the 10% and 50% percentiles in the BPIC 2012 logs

[2], and Bose and van der Aalst [6], who have analyzed these datasets in detail, the resources in this process tend to work as *specialists*: in particular, some resources are specialized in approving applications, while others are specialized in dealing with more problematic applications. This level of specialization may signal the fact that the experience developed in the workplace by these resources is important to correctly predict the outcome of a process case.

Figure 11 shows the analysis of the feature contribution for *rac*. From the plots, it can be clearly seen that, in this case, the resource-aware features contribute massively to the predictions: in the case of the *accepted* log, for instance, all the features in the top 10% percentile are resource-aware, while for the *cancelled* event log the number and cumulative contribution of the resource-aware features in the top 10% percentile is also very high (50% and 65%, respectively). These results show the importance of the resource-aware features for these logs.

Figure 12 shows the distribution of different top-contributing resource-aware features within the 10% and 50% percentiles. It can be noticed that, particularly for the *accepted* and *cancelled* event logs, the number of tasks executed by a resource is the most frequent type of highly contributing resource-aware features (specifically, the only one appearing in the 10% percentile). This can be seen as an indication that the experiential nature of learning is important in this context: resources get more efficient at what they do as they execute more tasks in the process.

BPIC 2011 event logs

Figure 13 shows the performance obtained by *rac* and *nrac* for different prefix lengths with RF and XGB. Figures 14 shows the performance of the models trained using the temporally-split batches.

In this case, it can be seen that the resource-aware features do not seem to have an influence, either positive or negative, on the performance of the models. This may be due to the fact that the label to be predicted, in this case, is the satisfaction of a temporal constraint, which is not likely to indicate whether a case was executed efficiently or not.

Figure 15 shows the ratio of the top-contributing resource-aware features and the ratio of their cumulative contribution for different percentiles for *rac*. These results show that the resource-aware features do not contribute particularly to the prediction (specifically, there are no resource-aware features within the 10% and 25% percentiles). Note that the labels to be predicted in this event log are defined by temporal constraints, e.g., either one of the two activities "tumor marker CA-19.9" and "ca-125 using meia" must be executed in a case, which capture the execution of standard exams dictated by clinical pathways. In this case, human resources should simply interpret correctly the guidelines prescribing (or not) correctly the execution of the exams. The human resource experience and/or social ties probably bear a limited impact on their behaviour in the process. This confirms that resource-aware features are not important in this particular context. For this reason, we omit the analysis of the distribution of the type of important resource-aware features for these event logs.

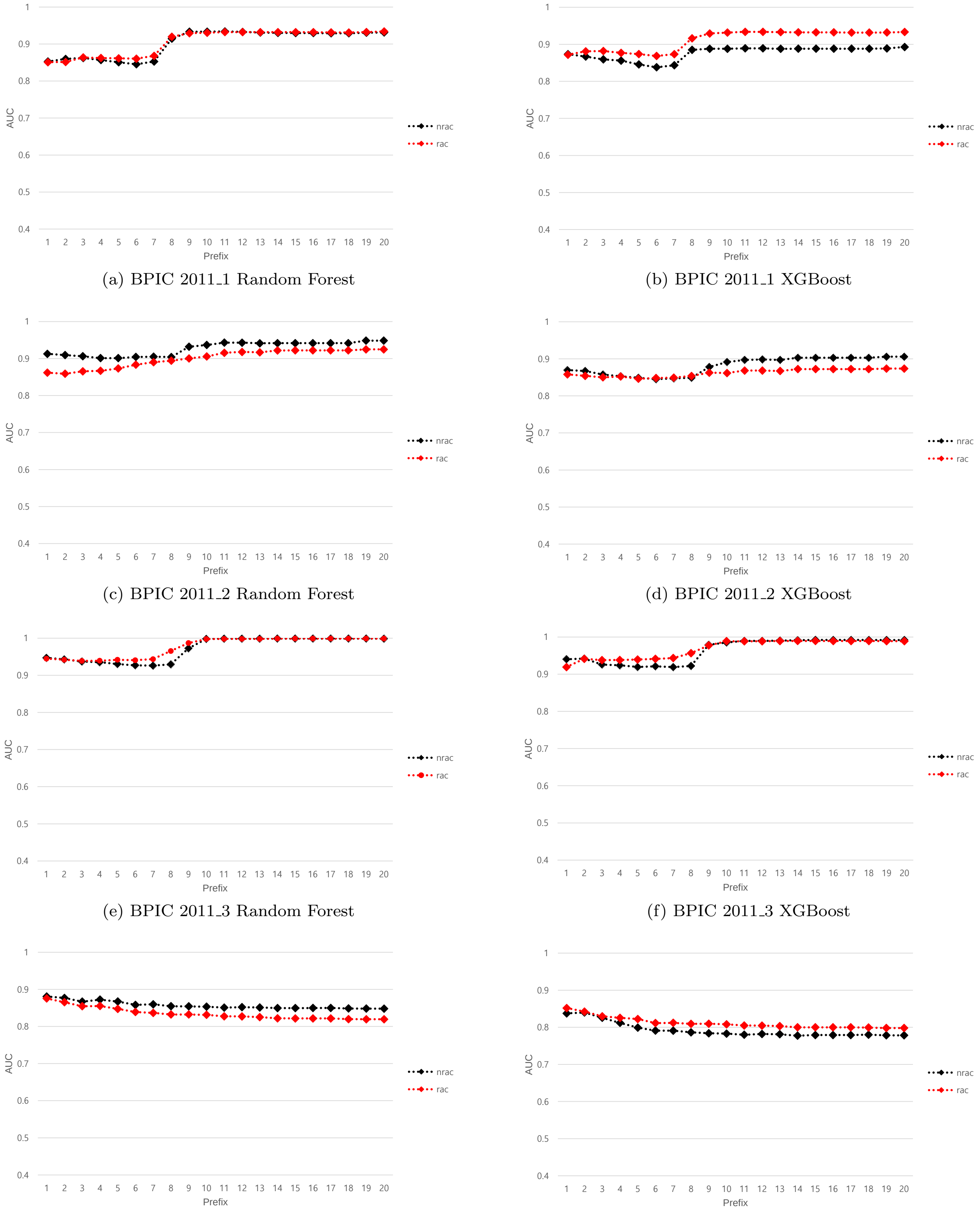
Road traffic fines event log

Figure 16 shows the performance obtained by *rac* and *nrac* for different prefix lengths with RF and XGB. Figure 17 shows the performance of the models trained using the temporally-split batches.

Similarly to the case of the BPIC 2011 event logs, also in this case using the resource-aware features proposed neither improves nor decreases substantially the performance of the model. This may be due to the fact that the label to be predicted, i.e., whether a fine is repaid in full or sent for credit collection (see Table 5), concerns the behaviour of the customer of the process (i.e., the motorist supposed to pay the fine), rather than the behavior of the resources handling the fine.

5.3. Threats to validity

To conclude the discussion on the experimental results, we briefly discuss here the threats to validity of the presented evaluation. Concerning the internal validity, a general problem in predictive process monitoring is that considering prefixes as the basis for encoding the feature vectors may introduce irrelevant features and/or spurious correlations between the feature vectors and the labels that improve the performance of a model without having discovered higher level concepts [24]. The experiments that we present here may also suffer from the same problem, which could be mitigated by using different predictive models, like the ones based on LSTM that use a different type of encoding, or focusing on global predictions, i.e., considering only fully-completed traces



(a) BPIC 2011_1 Random Forest

(b) BPIC 2011_1 XGBoost

(c) BPIC 2011_2 Random Forest

(d) BPIC 2011_2 XGBoost

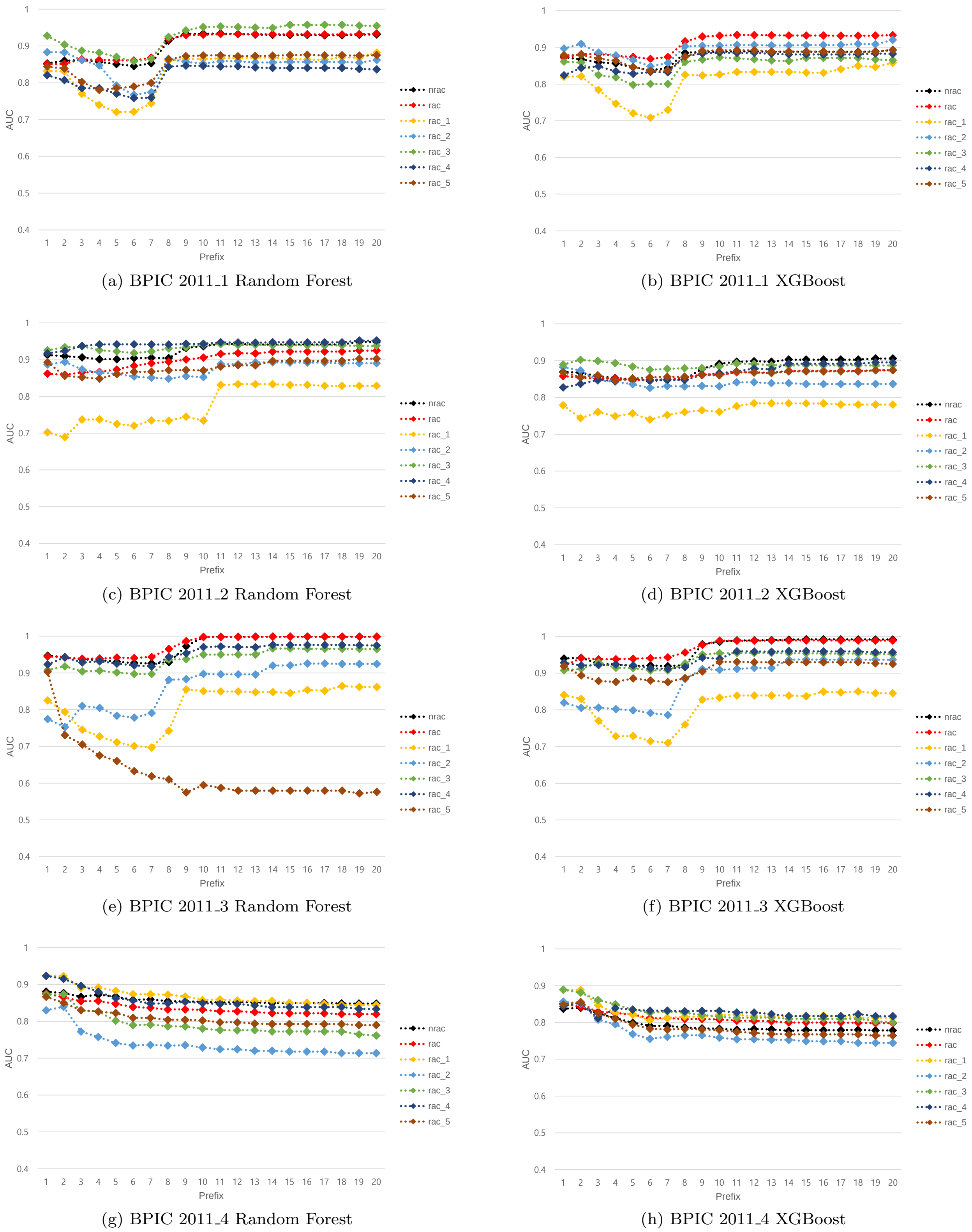
(e) BPIC 2011_3 Random Forest

(f) BPIC 2011_3 XGBoost

(g) BPIC 2011_4 Random Forest

(h) BPIC 2011_4 XGBoost

Figure 13: AUC for different prefix lengths for the BPIC 2011 event logs



(a) BPIC 2011_1 Random Forest

(b) BPIC 2011_1 XGBoost

(c) BPIC 2011_2 Random Forest

(d) BPIC 2011_2 XGBoost

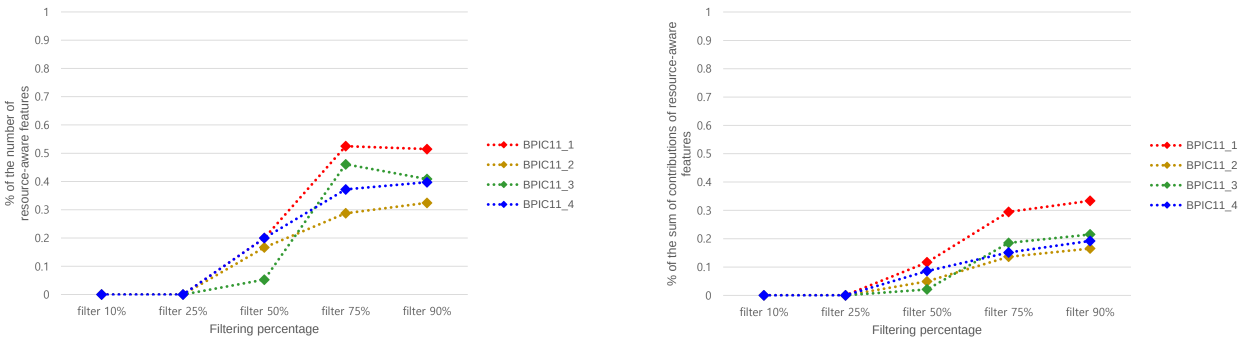
(e) BPIC 2011_3 Random Forest

(f) BPIC 2011_3 XGBoost

(g) BPIC 2011_4 Random Forest

(h) BPIC 2011_4 XGBoost

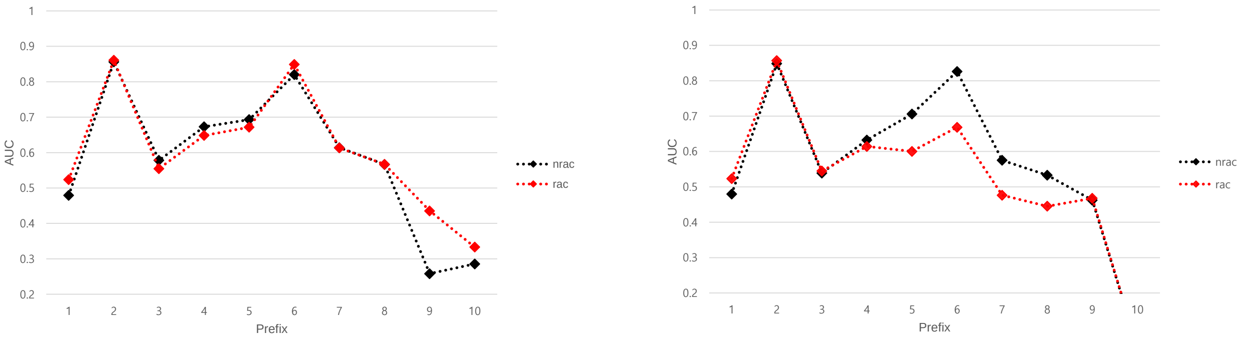
Figure 14: AUC for different prefix lengths for the BPIC 2011 event logs using the temporally-split batches



(a) Ratio of the top-contributing resource-aware features

(b) Ratio of the cumulative contribution of the top-contributing resource-aware features

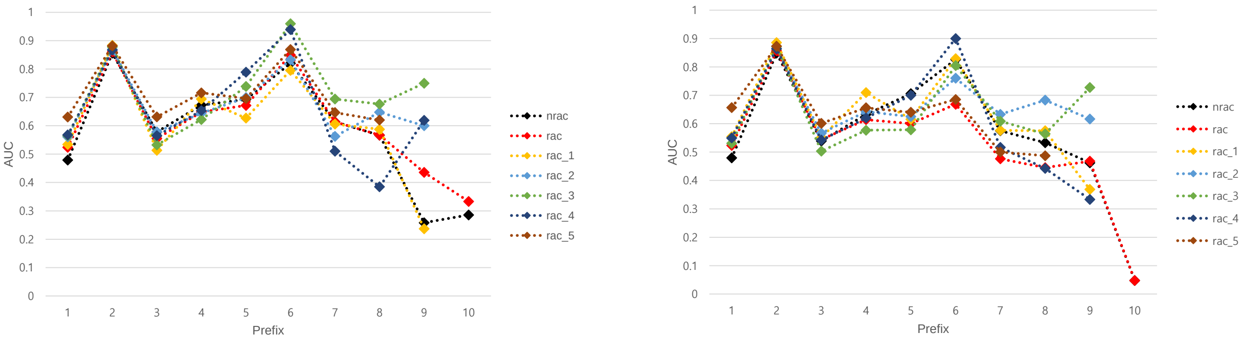
Figure 15: Ratio of the number and cumulative contribution of the top-contributing resource-aware features in the BPIC 2011 logs



(a) RTF Random Forest

(b) RTF XGBoost

Figure 16: AUC for different prefix lengths for the RTF event log



(a) RTF Random Forest

(b) RTF XGBoost

Figure 17: AUC for different prefix lengths for the RTF event log using the temporally-split batches

for training. In this paper, we also considered only a limited number of combinations of bucketing and encoding methods. This was done to maintain the number of experiments manageable and prioritizing the choice to consider a large set of event logs. As we discussed while presenting the results of each group of event logs, the results may also be influenced greatly by the type of label to be predicted. In some cases, the label represents a process outcome that is most likely linked to the resource experience, while, in other cases, it represents the satisfaction of a constraint, which may not be directly influenced by the resources, or a process outcome driven by the behavior of the customer, rather than by the behavior of the resources involved in the process.

Concerning the construct validity of the proposed resource-aware features, in this work we only consider features that can be extracted from an individual event log, which contains events logged for one specific business process. As such, the proposed resource-aware features are constructed using resource task performance measures in one process as a proxy of resource experience. We acknowledge that an event log is limited both in time and scope and that more reliable features may be extracted by considering data spanning in time beyond an event log time scale and/or data gathered from other contexts in which resources may have gained experience, such as related business processes and on-the-job training. We also believe, however, to have partly mitigated this threat, at least as far as the time scale is concerned, by considering event logs with a large time scale (often spanning years of process execution). Therefore, we believe to have considered, for each process, a sufficient amount of data to get an indication of the effect of the proposed resource-aware features on the performance of a predictive model.

Concerning the external validity, for some of the event logs that we considered (BPIC 2011 and RTF) there is no domain-specific information available to provide an in-depth analysis of the experimental results. More generally, the experimental results show that it is not possible to draw general conclusions about the role and the impact of the resource-aware features in predictive process monitoring, and that each process execution context has to be considered independently in order to assess the effectiveness of the proposed framework.

6. Conclusions

This article proposes a framework for extracting features from event logs capturing the notion of resource experience for the purpose of training models for outcome-oriented predictive process monitoring. The proposed framework defines four dimensions of resource-aware features, i.e., `Recency`, `Context`, `Target`, and `Aspect`, which are derived from an analysis of the management theories pertaining to resource experience.

The experimental evaluation showed that, in some cases, the proposed features can help improving the performance of the outcome-oriented predictive models. However, the effectiveness of the proposed features strongly depends on the process execution context and the results differ based on the event logs considered. In particular, we found that models trained using resource-aware features are more likely to show higher performance in contexts where the process is executed efficiently or when the label to be predicted captures an outcome that clearly depends on the efficiency of the resources in executing their tasks. In other words, the proposed features appear to be effective when the outcome label measures an actual business process outcome [36] related with process efficiency, effectiveness or flexibility improvements.

The research presented in this paper can be extended along several lines. First, the experimental evaluation focused on predicting case outcomes. However, resource experience features can also be applied to other predictive monitoring use cases, such as predicting the remaining time of a case. Second, the proposed framework is designed to be generic, i.e., independent of a specific application domain. However, the results presented in this paper have shown that the application domain clearly influences the effectiveness of the proposed framework. Therefore, further refinements should be conceived for specific domains like, for example, the manufacturing domain, where the frequent execution of the same task by a resource may lead to fatigue effects, which may cancel out the benefits of task familiarity and recency. Moreover, since resources may be involved in several processes within the same domain, resource-aware features could also be extended to capture the fact that experience gained in one process may be reused in a different one. This links to the more general field of cross-process research in process mining, which is emerging only recently in the literature [38]. Finally, the approach presented in this article may be used as a starting point to design experiments to validate a range of disciplines and theories from a managerial perspective. For instance, comparing in different contexts the predictive power of features capturing the repetition of the same task and the execution of different tasks in a process could substantiate managerial findings about the impact on performance of specialised v. general skills.

Acknowledgment

This research was partly funded by the Estonian Research Council (grant PRG887) and by the 0000 Project Fund (Project Number 1.210079.01) of UNIST (Ulsan National Institute of Science & Technology).

References

[1] van der Aalst, W.M.P., 2016. Process mining: data science in action. Springer.

[2] Adriansyah, A., Buijs, J.C., 2012. Mining process performance from event logs, in: International Conference on Business Process Management, Springer. pp. 217–218.

[3] Arias, M., Rojas, E., Munoz-Gama, J., Sepúlveda, M., 2016. A framework for recommending resource allocation based on process mining, in: International Conference on Business Process Management, Springer. pp. 458–470.

[4] Balzert, S., Fettke, P., Loos, P., 2012. A framework for reflective business process management, in: 2012 45th Hawaii International Conference on System Sciences, IEEE. pp. 3642–3651.

[5] Bidar, R., ter Hofstede, A., Sindhgatta, R., Ouyang, C., 2019. Preference-based resource and task allocation in business process automation, in: OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Springer. pp. 404–421.

[6] Bose, R.J.C., van der Aalst, W.M., 2012. Process mining applied to the BPI challenge 2012: Divide and conquer while discerning resources, in: International Conference on Business Process Management, Springer. pp. 221–222.

[7] Brass, D.J., Galaskiewicz, J., Greve, H.R., Tsai, W., 2004. Taking stock of networks and organizations: A multilevel perspective. Academy of Management Journal 47, 795–817.

[8] Brunk, J., Stierle, M., Papke, L., Revoredo, K., Matzner, M., Becker, J., 2021. Cause vs. effect in context-sensitive prediction of business process instances. Information Systems 95, 101635.

[9] Ceci, M., Lanotte, P.F., Fumarola, F., Cavallo, D.P., Malerba, D., 2014. Completion time and next activity prediction of processes using sequential pattern mining, in: International Conference on Discovery Science, Springer. pp. 49–61.

[10] Cohen, W.M., Levinthal, D.A., 1990. Absorptive capacity: A new perspective on learning and innovation. Administrative Science Quarterly 35, 128–152.

[11] Conforti, R., de Leoni, M., La Rosa, M., van der Aalst, W.M., ter Hofstede, A.H., 2015. A recommendation system for predicting risks across multiple business process instances. Decision Support Systems 69, 1–19.

[12] Cross, R., Cummings, J.N., 2004. Tie and network correlates of individual performance in knowledge-intensive work. Academy of Management Journal 47, 928–937.

[13] De Leoni, M., van der Aalst, W.M., Dees, M., 2016. A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Information Systems 56, 235–257.

[14] Di Francescomarino, C., Dumas, M., Maggi, F.M., Teinemaa, I., 2016. Clustering-based predictive process monitoring. IEEE Transactions on Services Computing 12, 896–909.

[15] Donabedian, A., 1988. The quality of care: how can it be assessed? Jama 260, 1743–1748.

[16] Erasmus, J., Vanderfeesten, I., Traganos, K., Jie-A-Looi, X., Kleingeld, A., Grefen, P., 2018. A method to enable ability-based human resource allocation in business process management systems, in: IFIP Working Conference on the Practice of Enterprise Modeling, Springer. pp. 37–52.

[17] Fahland, D., Denisov, V., van der Aalst, W., 2021. Inferring unobserved events in systems with shared resources and queues. `arXiv:2103.00167`.

[18] van der Ham, U., 2015. Benchmarking of five dutch municipalities with process mining techniques reveals opportunities for improvement. Business Process Intelligence Challenge 2015 .

[19] Harane, N., Rathi, S., 2020. Comprehensive survey on deep learning approaches in predictive business process monitoring, in: Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough. Springer, pp. 115–128.

[20] Häusser, J.A., Schulz-Hardt, S., Schultze, T., Tomaschek, A., Mojzisch, A., 2014. Experimental evidence for the effects of task repetitiveness on mental strain and objective work performance. Journal of Organizational Behavior 35, 705–721.

[21] Jaber, M.Y., Sikström, S., 2004. A numerical comparison of three potential learning and forgetting models. International Journal of Production Economics 92, 281–294.

[22] Karimi, J., Somers, T.M., Bhattacherjee, A., 2007. The role of information systems resources in erp capability building and business process outcomes. Journal of Management Information Systems 24, 221–260.

[23] Klijn, E.L., Fahland, D., 2020. Identifying and reducing errors in remaining time prediction due to inter-case dynamics, in: 2020 2nd International Conference on Process Mining (ICPM), IEEE. pp. 25–32.

[24] Klinkmüller, C., van Beest, N.R., Weber, I., 2018. Towards reliable predictive process monitoring, in: International Conference on Advanced Information Systems Engineering, Springer. pp. 163–181.

[25] Kolb, D.A., 2014. Experiential learning: Experience as the source of learning and development. FT press.

[26] Korytkowski, P., 2017. Competences-based performance model of multi-skilled workers with learning and forgetting. Expert Systems with Applications 77, 226–235.

[27] Kratsch, W., Manderscheid, J., Röglinger, M., Seyfried, J., 2020. Machine learning in business process monitoring: A comparison of deep learning and classical approaches used for outcome prediction. Business & Information Systems Engineering , 1–16.

[28] Leontjeva, A., Conforti, R., Di Francescomarino, C., Dumas, M., Maggi, F.M., 2016. Complex symbolic sequence encodings for predictive monitoring of business processes, in: International Conference on Business Process Management, Springer. pp. 297–313.

[29] Lieberman, M.B., 1987. The learning curve, diffusion, and competitive strategy. Strategic Management Journal 8, 441–452.

[30] Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.I., 2020. From local explanations to global understanding with explainable AI for trees. Nature Machine Intelligence 2, 56–67.

[31] Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems 30, 4765–4774.

[32] Maggi, F.M., Di Francescomarino, C., Dumas, M., Ghidini, C., 2014. Predictive monitoring of business processes, in: International Conference on Advanced Information Systems Engineering, Springer. pp. 457–472.

[33] Małachowski, B., Korytkowski, P., 2016. Competence-based performance model of multi-skilled workers. Computers & Industrial Engineering 91, 165–177.

[34] Márquez-Chamorro, A.E., Resinas, M., Ruiz-Cortés, A., 2017a. Predictive monitoring of business processes: a survey. IEEE Transactions on Services Computing 11, 962–977.

[35] Márquez-Chamorro, A.E., Resinas, M., Ruiz-Cortés, A., Toro, M., 2017b. Run-time prediction of business process indicators using evolutionary decision rules. Expert Systems with Applications 87, 1–14.

[36] Mooney, J.G., Gurbaxani, V., Kraemer, K.L., 1996. A process oriented framework for assessing the business value of information technology. ACM SIGMIS Database: the DATABASE for Advances in Information Systems 27, 68–81.

[37] Nakatumba, J., van der Aalst, W.M., 2009. Analyzing resource behavior using process mining, in: International Conference on Business Process Management, Springer. pp. 69–80.

[38] Neu, D.A., Lahann, J., Fettke, P., 2021. A systematic literature review on state-of-the-art deep learning methods for process prediction. Artificial Intelligence Review , 1–27.

[39] Park, G., Song, M., 2020. Predicting performances in business processes using deep neural networks. Decision Support Systems 129, 113191.

[40] Park, J.H., Suh, H.J., Yang, H.D., 2007. Perceived absorptive capacity of individual users in performance of enterprise resource planning (ERP) usage: The case for korean firms. Information & Management 44, 300–312.

[41] Pika, A., Leyer, M., Wynn, M.T., Fidge, C.J., Hofstede, A.H.T., Aalst, W.M.V.D., 2017. Mining resource profiles from event logs. ACM Transactions on Management Information Systems (TMIS) 8, 1–30.

[42] Pika, A., Wynn, M.T., 2020. Workforce upskilling: A history-based approach for recommending unfamiliar process activities, in: International Conference on Advanced Information Systems Engineering, Springer. pp. 334–349.

[43] Senderovich, A., Di Francescomarino, C., Ghidini, C., Jorbina, K., Maggi, F.M., 2017. Intra and inter-case features in predictive process monitoring: A tale of two dimensions, in: International Conference on Business Process Management, Springer. pp. 306–323.

[44] Senderovich, A., Di Francescomarino, C., Maggi, F.M., 2019. From knowledge-driven to data-driven inter-case feature encoding in predictive process monitoring. Information Systems 84, 255–264.

[45] Senderovich, A., Weidlich, M., Gal, A., Mandelbaum, A., 2014. Mining resource scheduling protocols, in: International Conference on Business Process Management, Springer. pp. 200–216.

[46] Tama, B.A., Comuzzi, M., 2019. An empirical comparison of classification techniques for next event prediction using business process event logs. Expert Systems with Applications 129, 233–245.

[47] Tax, N., Verenich, I., La Rosa, M., Dumas, M., 2017. Predictive business process monitoring with LSTM neural networks, in: International Conference on Advanced Information Systems Engineering, Springer. pp. 477–492.

[48] Taymouri, F., La Rosa, M., Erfani, S., Bozorgi, Z.D., Verenich, I., 2020. Predictive business process monitoring via generative adversarial nets: The case of next event prediction, in: International Conference on Business Process Management, Springer. pp. 237–256.

[49] Teinemaa, I., Dumas, M., Leontjeva, A., Maggi, F.M., 2018. Temporal stability in predictive process monitoring. Data Mining and Knowledge Discovery 32, 1306–1338.

[50] Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M., 2019. Outcome-oriented predictive process monitoring: Review and benchmark. ACM Transactions on Knowledge Discovery from Data (TKDD) 13, 1–57.

[51] Terwiesch, C., Bohn, R.E., 2001. Learning and process improvement during production ramp-up. International Journal of Production Economics 70, 1–19.

[52] Verenich, I., Dumas, M., La Rosa, M., Maggi, F.M., Di Francescomarino, C., 2016. Complex symbolic sequence clustering and multiple classifiers for predictive process monitoring, in: International Conference on Business Process Management, Springer. pp. 218–229.

[53] Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I., 2019. Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM Transactions on Intelligent Systems Technology 10, 34:1–34:34.

[54] Wieder, B., Booth, P., Matolcsy, Z.P., Ossimitz, M.L., 2006. The impact of erp systems on firm and business process performance. Journal of Enterprise Information Management .

[55] Zhao, W., Yang, L., Liu, H., Wu, R., 2015. The optimization of resource allocation based on process mining, in: International Conference on Intelligent Computing, Springer. pp. 341–353.