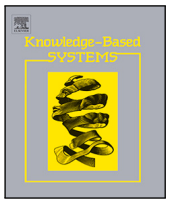
Knowledge-Based Systems 310 (2025) 112970

Contents lists available at ScienceDirect

Knowledge-Based Systems

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



Control-flow anomaly detection by process mining-based feature extraction and dimensionality reduction

Francesco Vitale [a,*], Marco Pegoraro [b], Wil M.P. van der Aalst [b], Nicola Mazzocca [a]

[a] University of Naples Federico II, Via Claudio, 21, Naples, 80125, Campania, Italy
[b] RWTH Aachen University, Ahornstr. 55, Aachen, 52074, Nordrhein-Westfalen, Germany

ARTICLE INFO

Keywords:
Event data
Process discovery
Conformance checking
Feature extraction
Dimensionality reduction

ABSTRACT

The business processes of organizations may deviate from normal control flow due to disruptive anomalies, including unknown, skipped, and wrongly-ordered activities. To identify these control-flow anomalies, process mining can check control-flow correctness against a reference process model through conformance checking, an explainable set of algorithms that allows linking any deviations with model elements. However, the effectiveness of conformance checking-based techniques is negatively affected by noisy event data and low-quality process models. To address these shortcomings and support the development of competitive and explainable conformance checking-based techniques for control-flow anomaly detection, we propose a novel process mining-based feature extraction approach with alignment-based conformance checking. This variant aligns the deviating control flow with a reference process model; the resulting alignment can be inspected to extract additional statistics such as the number of times a given activity caused mismatches. We integrate this approach into a flexible and explainable framework for developing techniques for control-flow anomaly detection. The framework combines process mining-based feature extraction and dimensionality reduction to handle high-dimensional feature sets, achieve detection effectiveness, and support explainability. The results show that the framework techniques implementing our approach outperform the baseline conformance checking-based techniques while maintaining the explainable nature of conformance checking. We also provide an explanation of why existing conformance checking-based techniques may be ineffective. Finally, the results indicate that detection effectiveness is not solely dependent on the specific framework technique used, as no one-size-fits-all process mining-based feature extraction approach is suitable for all the synthetic and real-world datasets.

1. Introduction

The business processes of organizations are experiencing ever-increasing complexity due to the large amount of data, high number of users, and high-tech devices involved [1,2]. This complexity may cause business processes to deviate from normal control flow due to unforeseen and disruptive anomalies [3]. These control-flow anomalies manifest as unknown, skipped, and wrongly-ordered activities in the traces of event logs monitored from the execution of business processes [4]. For the sake of clarity, let us consider an illustrative example of such anomalies. Fig. 1 shows a so-called event log footprint, which captures the control flow relations of four activities of a hypothetical event log. In particular, this footprint captures the control-flow relations between activities a, b, c and d. These are the causal (→) relation, concurrent (||) relation, and other (#) relations such as exclusivity or non-local dependency [5]. In addition, on the right are six traces, of which five exhibit skipped, wrongly-ordered and unknown control-flow anomalies. For example, ⟨a b d⟩ has a skipped activity, which is c. Because of this skipped activity, the control-flow relation b # d is violated, since d directly follows b in the anomalous trace.

1.1. Control-flow anomaly detection

Control-flow anomaly detection techniques aim to characterize the normal control flow from event logs and verify whether anomalies occur in new event logs [4]. To develop control-flow anomaly detection techniques, process mining has seen widespread adoption thanks to process discovery and conformance checking. On the one hand, process discovery is a set of algorithms that encode control-flow relations as a set of model elements and constraints according to a given modeling formalism [5]; hereafter, we refer to the Petri net, a widespread modeling formalism. On the other hand, conformance

* Corresponding author.
E-mail address: francesco.vitale@unina.it (F. Vitale).

https://doi.org/10.1016/j.knosys.2025.112970
Received 7 June 2024; Received in revised form 27 October 2024; Accepted 4 January 2025
Available online 17 January 2025

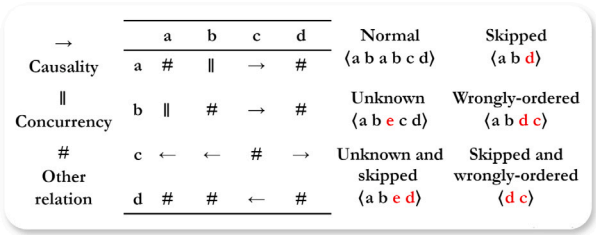


Fig. 1. An example event log footprint with six traces, of which five exhibit control-flow anomalies.

checking is an explainable set of algorithms that allows linking any deviations with the reference Petri net and providing the fitness measure, namely a measure of how much the Petri net fits the new event log [5]. Many control-flow anomaly detection techniques based on conformance checking (hereafter, conformance checking-based techniques) use the fitness measure to determine whether an event log is anomalous [6–9].

The scientific literature also includes many conformance checking-independent techniques for control-flow anomaly detection that combine specific types of trace encodings with machine/deep learning [4,10]. Whereas these techniques are very effective, their explainability is challenging due to both the type of trace encoding employed and the machine/deep learning model used [11,12]. Hence, in the following, we focus on the shortcomings of conformance checking-based techniques to investigate whether it is possible to support the development of competitive control-flow anomaly detection techniques while maintaining the explainable nature of conformance checking.

1.2. Shortcomings of conformance checking-based techniques

Unfortunately, the detection effectiveness of conformance checking-based techniques is affected by noisy data and low-quality Petri nets, which may be due to human errors in the modeling process or the representational bias of process discovery algorithms [7,9,13]. Specifically, on the one hand, noisy data may introduce infrequent and deceptive control-flow relations that result in inconsistent fitness measures, whereas, on the other hand, checking event logs against a low-quality Petri net could lead to an unreliable distribution of fitness measures. Nonetheless, such Petri nets can still be used as references to obtain insightful information for process mining-based feature extraction, supporting the development of competitive and explainable conformance checking-based techniques for control-flow anomaly detection despite the problems above. For example, a few works outline that token-based conformance checking can be used for process mining-based feature extraction to build tabular data and develop effective conformance checking-based techniques for control-flow anomaly detection [14,15]. However, to the best of our knowledge, the scientific literature lacks a structured proposal for process mining-based feature extraction using the state-of-the-art conformance checking variant, namely alignment-based conformance checking.

1.3. Contributions

We propose a novel process mining-based feature extraction approach with alignment-based conformance checking. This variant aligns the deviating control flow with a reference Petri net; the resulting alignment can be inspected to extract additional statistics such as the number of times a given activity caused mismatches [5]. We integrate this approach into a flexible and explainable framework for developing techniques for control-flow anomaly detection. The framework combines process mining-based feature extraction and dimensionality reduction to handle high-dimensional feature sets, achieve detection effectiveness, and support explainability. Notably, in addition to our proposed process mining-based feature extraction approach, the framework allows employing other approaches, enabling a fair comparison of multiple conformance checking-based and conformance checking-independent techniques for control-flow anomaly detection. Fig. 2 shows a high-level view of the framework. Business processes are monitored, and event logs obtained from the database of information systems. Subsequently, process mining-based feature extraction is applied to these event logs and tabular data input to dimensionality reduction to identify control-flow anomalies. We apply several conformance checking-based and conformance checking-independent framework techniques to publicly available datasets, simulated data of a case study from railways, and real-world data of a case study from healthcare. We show that the framework techniques implementing our approach outperform the baseline conformance checking-based techniques while maintaining the explainable nature of conformance checking.

In summary, the contributions of this paper are as follows.

- A novel process mining-based feature extraction approach to support the development of competitive and explainable conformance checking-based techniques for control-flow anomaly detection.
- A flexible and explainable framework for developing techniques for control-flow anomaly detection using process mining-based feature extraction and dimensionality reduction.
- Application to synthetic and real-world datasets of several conformance checking-based and conformance checking-independent framework techniques, evaluating their detection effectiveness and explainability.

The rest of the paper is organized as follows.

- Section 2 reviews the existing techniques for control-flow anomaly detection, categorizing them into conformance checking-based and conformance checking-independent techniques.
- Section 3 provides the preliminaries of process mining to establish the notation used throughout the paper, and delves into the details of the proposed process mining-based feature extraction approach with alignment-based conformance checking.
- Section 4 describes the framework for developing conformance checking-based and conformance checking-independent techniques for control-flow anomaly detection that combine process mining-based feature extraction and dimensionality reduction.
- Section 5 presents the experiments conducted with multiple framework and baseline techniques using data from publicly available datasets and case studies.
- Section 6 draws the conclusions and presents future work.

2. Related work

The large body of research on control-flow anomaly detection in event logs sees a wide range of data-driven approaches [4]. In particular, the scientific literature developed techniques for control-flow anomaly detection that may or may not use conformance checking. To distinguish between those that use conformance checking and those that do not, we refer to them as conformance checking-based techniques and conformance checking-independent techniques.

2.1. Conformance checking-independent techniques

Conformance checking-independent techniques use specific types of trace encodings to extract features from event logs, such as one-hot encoding and word2vec [4]. In particular, trace encodings may extract process-based statistics, such as N-grams or directly-follows relationships [10]. We refer to these types of trace encodings as *statistical* process mining-based feature extraction.

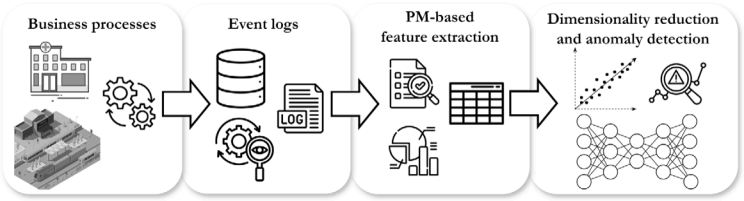


Fig. 2. A high-level view of the proposed framework for combining process mining-based feature extraction with dimensionality reduction for control-flow anomaly detection.

Deep learning. A large class of conformance checking-independent techniques rely on deep learning by feed-forward and/or recurrent neural networks, such as the autoencoder [16–20], long-short term memory networks [21,22] and gated recurrent unit networks [23,24]. Some works combined these approaches, employing both the autoencoder and recurrent neural networks to exploit the ability of reconstruction-based and prediction-based anomaly detection [25–27]. Although conformance checking-independent techniques using deep learning are very effective, their performance could differ based on the type of trace encoding [28]. Moreover, deep learning is notoriously affected by explainability issues [11]. To address these issues, the scientific literature aimed to combine deep learning with process mining. Firstly, recurrent neural networks have been used to approximate the diagnostics that alignment-based conformance checking provides; arguably, these workarounds reduce the computational cost of alignment-based conformance checking while reaching comparable results [29,30]. Additionally, other approaches used deep learning to pre-process data so that process mining could provide better performance. For example, Krajsic and Franczyk [31] used the autoencoder to perform anomaly detection before process mining to obtain higher-quality process models through process discovery. Similarly, Wang et al. [26] applied process mining after conformance checking-independent control-flow anomaly detection to provide diagnostics and inspect the causes of deviating control flow. These works suggest that the literature recognizes the utility of process mining in improving the explainability of control-flow anomaly detection based on deep learning.

Statistics. Whereas many conformance checking-independent techniques rely on deep learning due to its detection effectiveness, several proposals set forth statistical approaches. Mostly, these approaches rely on evaluating the similarity of new event data to a normal and interpretable statistic. For example, Li and van der Aalst [32] encoded information from the directly follows and dependency relations, building "profiles" by which new event data are compared to evaluate whether they are anomalous. Ko and Comuzzi [33] extracted a statistical index termed leverage measure to use as a metric to calculate the "anomaly score" of traces in event logs; this measure can be used to either fit a statistical distribution or build a threshold to which new traces are compared. Similarly, Luftensteiner and Praher [34] presented an approach based on spectral analysis of the adjacency matrix built from a graph that encodes a given relation among events. The approach evaluates the difference between the second-highest eigenvalue of adjacency matrices — i.e., the spectral gap — of a normal event log and a new event log; if the spectral gap is too large, the new event log is anomalous. Finally, Mavroudopoulos and Gounaris [35] proposed a nearest-neighbor-based approach that evaluates the distance between reference "vector traces", i.e., trace encodings, and new vector traces using different metrics; if the distance exceeds a given threshold based on the set of normal trace encodings, the new vector trace is classified as anomalous.

The high heterogeneity of conformance checking-independent techniques for control-flow anomaly detection makes a comprehensive and fair comparison between them a challenging task. The framework that we propose in Section 4 aims to implement reconstruction-based anomaly detection, as done in many of the reviewed Refs. [16–20]. However, an overarching comparison of all conformance checking-independent techniques is out of the scope of this work. Instead, as discussed below, we focus on enhancing conformance checking-based techniques to both improve the detection performance and maintain the explainable nature of conformance checking.

2.2. Conformance checking-based techniques

Contrary to conformance checking-independent techniques, conformance checking-based techniques do not employ trace encodings such as one-hot encoding and word2vec. Instead, conformance checking-based techniques aim to seamlessly check new event logs with a reference Petri net, obtaining the fitness measure to indicate whether the event data is anomalous [6–9,36]. In addition, the fitness measure is tightly connected to the Petri net semantics, thus providing explainable ways to analyze the root cause of unfitting event logs [13]. However, relying on fitness thresholding has shown poor detection effectiveness mostly due to noisy data and/or low-quality process models [7,9]. In this regard, let us consider Fig. 3, which depicts a normal event log, namely an event log containing normal traces from one of the datasets we use during the evaluation, and a reference Petri net from the same dataset. The histogram shows that a high percentage of traces have a fitness measure below 0.75. It is therefore not trivial to set a threshold based on these results.

Solutions to the shortcomings. Whereas some works disregarded the issue of noisy event logs and/or low-quality Petri nets by deeming anomalous all traces that the reference Petri net does not perfectly fit [8,36], other works attempted to account for these shortcomings by exploiting other information related to either the reference Petri net or control-flow diagnostics obtained by conformance checking. For example, Bezerra et al. [6] combined process discovery with conformance checking to extract the "most appropriate" Petri net according to both a manually tuned fitness threshold and the simplicity of the Petri net obtained by process discovery. Similarly, Bezerra et al. [7] and Pecchia et al. [9] developed algorithms and methods to discover high-quality Petri nets with process discovery and automatically tune the fitness threshold for anomaly detection. In addition to these solutions, which somehow attempt to optimize the fitness threshold, other approaches combine conformance checking with machine learning to supply additional information about faulty control flow, such as the number of times an activity was not allowed to execute according to the model semantics. This process leads to conformance checking process mining-based feature extraction, which results in tabular data that can be processed by machine/deep learning algorithms for anomaly detection [14,15,37].

Our solution. To address the problem shown in Fig. 3 regarding the choice of a fitness threshold, the scientific literature proposes to either (a) tune an optimal fitness threshold by discovering the highest-quality process model through process discovery, or (b) provide additional control-flow information obtained by conformance checking to feed to machine learning. In this work, we follow approach (b) and develop a novel process mining-based feature extraction approach with alignment-based conformance checking, which aligns the deviating control flow with a reference Petri net; the resulting alignment can

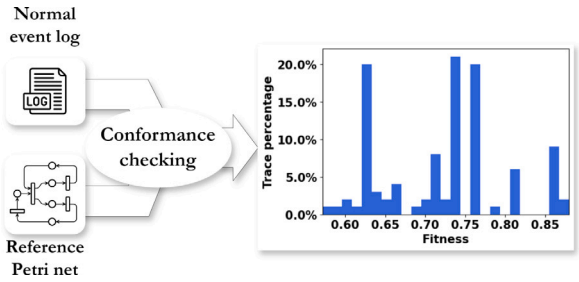


Fig. 3. The distribution of fitness values of case-study traces of a normal event log replayed on a reference Petri net. The inconsistent distribution of fitness values of the case-study traces makes fitness thresholding ineffective for control-flow anomaly detection.

be inspected to extract additional statistics such as the number of times a given activity caused mismatches. This allows the connection of these statistics to the reference Petri net for additional insights on the deviating control flow. We present this solution in the next section, followed by its integration into a flexible and explainable framework for developing conformance checking-based and conformance checking-independent techniques for control-flow anomaly detection.

3. The proposed process mining-based feature extraction

This section first presents the preliminaries of process mining to establish the notation used throughout the paper. In particular, we report the standard definitions from the process mining literature of the event log, sublog, trace, Petri net, fitness, and alignment and moves. After the necessary notation is established, the proposed process mining-based feature extraction with alignment-based conformance checking is developed, including the definitions of alignment-based fitness and per-activity cost. These two are used to extract alignment-based conformance checking diagnoses, i.e., the novel process mining-based feature extraction approach.

3.1. Preliminaries

Firstly, we report the definition of the event log and sublog, since they are the main inputs to process mining algorithms.

Definition 3.1 (*Event Log, Sublog and Trace*)**.** Let us denote $\mathcal{A}$ as the universe of activities and $\mathcal{A}^*$ the universe of possible sequences of activities in $\mathcal{A}$. Hereafter, we refer to a sequence of activities as a *trace* $\sigma \in \mathcal{A}^*$. An *event log* L is a multi-set of k traces, where a multi-set is a set such that there may be two equal yet distinct elements. Given $\mathcal{B}(\mathcal{A}^*)$ the set of multi-sets over $\mathcal{A}^*$, $L \in \mathcal{B}(\mathcal{A}^*)$. A *sublog* is a subset of L. Furthermore, we denote an *n-tuple of event logs* $\mathcal{L} = (\mathcal{L}(1), \mathcal{L}(2), \dots, \mathcal{L}(n))$ as $\mathcal{L} \in (\mathcal{B}(\mathcal{A}^*))^n$.

An event log can be processed by process discovery algorithms to build a process model. This process model captures a set of possible control flows based on the relationships among activities within the traces of the event log. There is a plethora of formalisms to capture these relationships. As mentioned in Section 1, in this work, we focus on the labeled Petri net (hereafter, Petri net).

Definition 3.2 (*Petri Net*)**.** Let P and Tr be two sets of nodes of a bipartite graph such that $P \cap Tr = \emptyset$, and $F \subseteq (P \times Tr) \cup (Tr \times P)$ a set of directed arcs. Furthermore, let $A \subseteq \mathcal{A}$ be a set of activities and $l : Tr \to A \cup \{\tau\}$ a function that associates to the elements of Tr either an activity of A or the "silent" label τ. A Petri net N is the tuple (P, Tr, F, A, l), where P are the places, Tr the transitions, F the arcs, A the activities, and l the labeling function. Finally, $M \in \mathcal{B}(P)$ is the (current) *marking* of the Petri net, i.e., a multi-set of tokens that determine which transition is allowed to *fire*. In the following, we denote $\mathcal{N}$ the universe of Petri nets.

Table 1
Alignment-based conformance checking diagnoses obtained from replaying the n-tuple of event logs $\mathcal{L}$ against a Petri net N.

Event log	c_1	c_2	$\cdots$	c_o	$F_{\mathcal{L}(i),N}$
$\mathcal{L}(1)$	$u_{c_1,\mathcal{L}(1)}$	$u_{c_2,\mathcal{L}(1)}$	$\cdots$	$u_{c_o,\mathcal{L}(1)}$	$F_{\mathcal{L}(1),N}$
$\mathcal{L}(2)$	$u_{c_1,\mathcal{L}(2)}$	$u_{c_2,\mathcal{L}(2)}$	$\cdots$	$u_{c_o,\mathcal{L}(2)}$	$F_{\mathcal{L}(2),N}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\mathcal{L}(n)$	$u_{c_1,\mathcal{L}(n)}$	$u_{c_2,\mathcal{L}(n)}$	$\cdots$	$u_{c_o,\mathcal{L}(n)}$	$F_{\mathcal{L}(n),N}$

Similar to process discovery, conformance checking algorithms also process event logs, but they use a reference Petri net and check the traces of event logs against it to verify if the execution fits the model [5]. The main output of conformance checking is the fitness, which quantifies how much the process model fits the event log.

Definition 3.3 (*Fitness*)**.** Let $L \in \mathcal{B}(\mathcal{A}^*)$ be an event log, $\sigma \in L$ a trace, and $N \in \mathcal{N}$ a Petri net. We denote the *fitness* computed by conformance checking when replaying σ on N as $F_{\sigma,N} \in [0, 1]$, whereas when replaying all traces of L we denote the *fitness* as $F_{L,N} \in [0, 1]$.

Although there are many conformance checking variants in the literature, the state-of-the-art is alignment-based conformance checking. This variant attempts to find the best path across the reference Petri net that most accurately approximates the traces in the event log that deviate from normal control flow. This is done by taking into account the *log moves* in an event log and their *alignment* to the reference Petri net by evaluating the corresponding *model moves*.

Definition 3.4 (*Alignment And Moves*)**.** Let us denote $\gg$ as the "skipped" activity. Let $N \in \mathcal{N}$ be a Petri net and $\sigma \in \mathcal{A}^*$ a trace. $\sigma_L = \langle a_1, \dots, a_o \rangle \in (\mathcal{A} \cup \{\gg\})^*$ is a (finite) sequence of log moves related to σ if and only if $\sigma_L \setminus \{\gg\} = \sigma$. Then, $\sigma_N = \langle b_1, \dots, b_o \rangle \in (\mathcal{A} \cup \{\tau\} \cup \{\gg\})^*$ is a (finite) sequence of model moves if and only if $\sigma_N \setminus \{\gg\}$ is a firing sequence of N, i.e. an allowed control flow. An *alignment* $\gamma_{\sigma,N}$ is the two-row matrix

$$\gamma_{\sigma,N} = \begin{array}{c|c|c|c} a_1 & a_2 & \cdots & a_o \\ \hline b_1 & b_2 & \cdots & b_o \end{array}, \tag{1}$$

if for all $1 \le i \le o$, $(a_i, b_i) \neq (\gg, \gg)$, where (a_i, b_i) is a *move*.

3.2. Alignment-based conformance checking diagnoses

Our proposal aims to collect information recorded throughout alignment-based conformance checking when replaying event logs on a reference Petri net; we term such information as *alignment-based conformance checking diagnoses*. These diagnoses involve computing the alignment-based fitness and the per-activity cost.

Definition 3.5 (*Alignment-Based Fitness*)**.** Let us denote $\Gamma_{\sigma,N}$ as the set of all alignments between a trace $\sigma \in \mathcal{A}^*$ and a Petri net $N \in \mathcal{N}$. Let δ be the unitary cost function for a pair of moves or an alignment. Finally, let $\gamma^w_{\sigma,N} \in \Gamma_{\sigma,N}$ be the worst-case alignment and $\gamma^*_{\sigma,N} \in \Gamma_{\sigma,N}$ the best-case alignment, i.e., the alignments with the least and most costs according to δ, respectively. The *alignment-based fitness* for σ is defined as

$$F_{\sigma,N} = 1 - \frac{\delta(\gamma^*_{\sigma,N})}{\delta(\gamma^w_{\sigma,N})}. \tag{2}$$

Let $L \in \mathcal{B}(\mathcal{A}^*)$ be an event log. The *alignment-based fitness* for L is defined as

$$F_{L,N} = \frac{\sum_{\sigma \in L} F_{\sigma,N}}{|L|}. \tag{3}$$

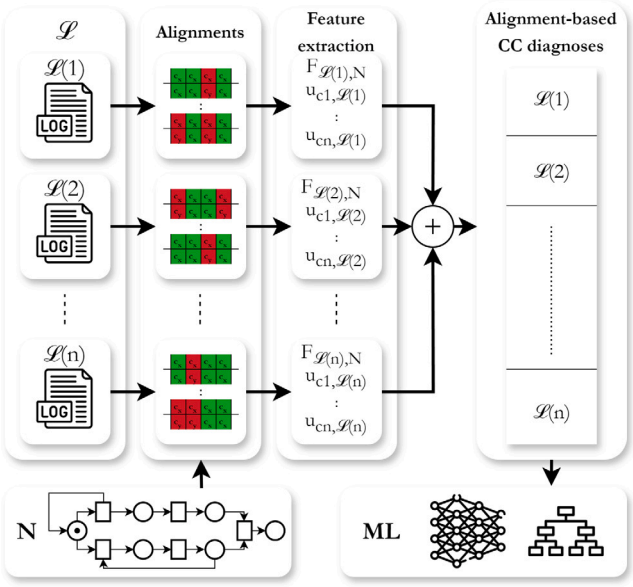


Fig. 4. Generation of alignment-based conformance checking diagnoses by replaying the n-tuple of event logs $\mathcal{L}$ against a Petri net N.

Definition 3.6 (*Per-Activity Cost*)**.** Let $N \in \mathcal{N}$ be a Petri net, $\sigma \in \mathcal{A}^*$ a trace, $C_{\sigma,N} \subseteq \mathcal{A}$ the set of activities found either in σ or in labeled transitions of N, δ the cost function for a pair of moves or an alignment, $\gamma^*_{\sigma,N} \in \Gamma_{\sigma,N}$ the best-case alignment, and $\gamma^*_{c,\sigma,N}$ the set of moves of $\gamma^*_{\sigma,N}$ such that $\forall (a,b) \in \gamma^*_{c,\sigma,N}, c = a \vee c = b$. The *per-activity cost* for σ is defined as

$$u_{c,\sigma} = \sum_{(a,b)\in\gamma^*_{c,\sigma,N}} \delta((a,b)), \quad c \in C_{\sigma,N}. \tag{4}$$

Let $L \in \mathcal{B}(\mathcal{A}^*)$ be an event log and $C_{L,N} \subseteq \mathcal{A}$ the set of activities found either in traces of L or in labeled transitions of N. The per-activity cost for L is:

$$u_{c,L} = \sum_{\sigma\in L} u_{c,\sigma}, \quad c \in C_{L,N}. \tag{5}$$

Definition 3.7 (*Alignment-Based Conformance Checking Diagnoses*)**.** Let $\mathcal{L} \in (\mathcal{B}(\mathcal{A}^*))^n$ be an n-tuple of event logs, $N \in \mathcal{N}$ a Petri net, $C_{\mathcal{L},N} = \{c_1, \ldots, c_o\} \subseteq \mathcal{A}$ the set of activities found either in traces of event logs of $\mathcal{L}$ or in labeled transitions of N. *Alignment-based conformance checking diagnoses* for $\mathcal{L}$ is tabular data arranged as in Table 1.

Fig. 4 visualizes the generation of alignment-based conformance checking diagnoses. A starting set of event logs $\mathcal{L}$ is aligned to a Petri net N, generating a set of alignments per event log $\mathcal{L}(1) \ldots \mathcal{L}(n)$. Subsequently, alignment-based fitness $F_{\mathcal{L}(x),N}$ and per-activity costs $u_{c_1,\mathcal{L}(1)} \ldots u_{c_n,\mathcal{L}(1)}$ per event log $\mathcal{L}(1) \ldots \mathcal{L}(n)$ are calculated. The results are joined together to form alignment-based conformance checking diagnoses. Finally, machine learning can handle the resulting tabular data.

Illustrative example. Fig. 5 shows the connection between alignment-based conformance checking diagnoses of three traces and the reference Petri net of one of the datasets we use in Section 5. Specifically, the reference Petri net provides control-flow relations that alignment-based conformance checking uses to check whether traces σ_1, σ_2 and σ_3 deviate from such relations. For example, σ_1 exhibits duplicated (`t35`), wrongly-ordered (`t51`→`t62`), skipped (`t21`→`t32`, `t44`→`t54`, `t76`→`t82`), and unknown (`t75`) control-flow anomalies. As a result, this reflects in some mismatches in the corresponding alignment, highlighted by the moves (`t35`, ≫), (≫,`t32`), (`t62`, ≫), (≫,`t76`) and (`t75`, ≫). These mismatches can be projected onto the process model. For example, by summing all per-activity costs related to `t44`, this results in $\sum_\sigma u_{t44,\sigma} = 3$.

4. Framework

4.1. Problem statement

The framework combines process mining-based feature extraction and dimensionality reduction to allow the flexible development of several conformance checking-based and conformance checking-independent techniques for control-flow anomaly detection. This enables a fair comparison of multiple techniques, which may include our proposed process mining-based feature extraction approach using alignment-based conformance checking.

Framework guidelines. Following Chandola et al.'s guidelines [38] for developing anomaly detection techniques, which include defining the nature of data, type of anomalies, and the research areas employed, the framework:

- Handles the event logs of business processes (see Definition 3.1) and addresses control-flow anomalies such as unknown, wrongly-ordered, and skipped activities;
- Uses conformance checking and other types of trace encodings for process mining-based feature extraction and generates tabular data from event logs;
- Employs dimensionality reduction with the tabular data resulting from training and validation event logs to build a classifier, and subsequently uses the classifier to discriminate normal and anomalous event logs.

Scenario type. The type of data used during the training phase of anomaly detection techniques is highly dependent on the scenario type. For example, whereas supervised scenarios have plenty of data with accurate labels for normal and anomalous data, unsupervised scenarios involve unlabeled data only and are effective when anomalous data are infrequent [38]. There is an intermediate scenario, namely the weakly-supervised scenario. This scenario brings the benefit of having loads of labeled normal data and a limited amount of anomalous data, and is common in business process monitoring [27,39,40]. By using only normal data during the training phase, the technique is tuned to accurately discriminate those patterns that are consistent with normal behavior, and to detect *any* deviations from it. Due to weak supervision being a common scenario in business process monitoring, and the potential benefits of tailoring the anomaly detection technique to normal data only, the proposed framework deals with the weakly-supervised scenario. Hence, the framework includes only normal event logs during training and retains anomalous event logs in the test set. However, it is worth noting that weak supervision means the data may contain inaccuracies and incomplete information [41,42]. Consequently, noisy event logs and low-quality Petri nets can lead to issues such as those shown in Fig. 3.

Explainability. In addition to Chandola et al.'s guidelines, the framework aims to provide an explanation of the anomalies according to Li et al.'s [12] definitions, which are the *oracle-definition*, *detection-definition*, and *explanation-definition*. The **oracle-definition** is domain-specific and reflects the above-mentioned unknown, wrongly-ordered and skipped activities. For example, wrong medical treatment could lead to a doctor skipping key analyses for their patient, thus skipping activities. The remaining two types are framework-dependent and strongly connected to process mining-based feature extraction and dimensionality reduction, which are the framework's main steps depicted in Fig. 6 and described in the following.

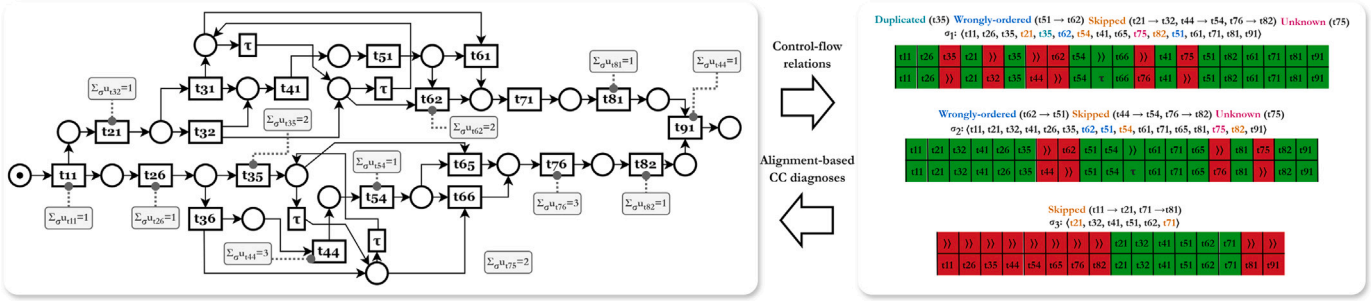


Fig. 5. The connection between alignment-based conformance checking diagnoses of three traces and the reference Petri net of one of the datasets we use in Section 5.

4.2. Process mining-based feature extraction

Firstly, event logs and a reference Petri net are obtained from the business process to check for deviations. On the one hand, the event logs are monitored while the business process is being executed and are labeled as normal or anomalous. On the other hand, the reference Petri net could be obtained either manually by domain experts or through the automatic generation from normal event logs, for example by using process discovery algorithms. The event logs are split into training, validation and test sets so that the normal control flow is characterized using the training set, the ability to generalize is evaluated with the validation set, and the performance is assessed with the test set, which is the only set including anomalous event logs. Process mining-based feature extraction applies to all three sets to obtain tabular data from event data.

Definition 4.1 (*Process Mining-Based Feature Extraction*). Let $\mathcal{L} \in (\mathcal{B}(\mathcal{A}^*))^n$ be an n-tuple of event logs and $N \in \mathcal{N}$ a Petri net. Statistical process mining-based feature extraction $\mathcal{F}_S$ is defined as

$$\mathcal{F}_S : (\mathcal{B}(\mathcal{A}^*))^n \to \mathbb{R}^{n \times f}, \tag{6}$$

such that $\mathcal{F}_S(\mathcal{L})$ is an $n \times f$ matrix, where the columns are obtained directly from statistics related to the event logs of $\mathcal{L}$. Conformance checking process mining-based feature extraction $\mathcal{F}_{CC}$ is defined as

$$\mathcal{F}_{CC} : (\mathcal{B}(\mathcal{A}^*))^n \times \mathcal{N} \to \mathbb{R}^{n \times f}, \tag{7}$$

such that $\mathcal{F}_{CC}(\mathcal{L}, N)$ is an $n \times f$ matrix, where the columns are obtained by checking the event logs of $\mathcal{L}$ against N by conformance checking.

Statistical process mining-based feature extraction has the advantage of being conformance checking-independent, as features are extracted from the statistical occurrence of certain patterns within the traces of event logs. However, the disadvantage is the absence of a reference Petri net, which allows an improved explanation of anomaly detection results. For example, Fig. 5 shows that conformance checking process mining-based feature extraction implemented with our proposed approach associates the per-activity cost with model elements, flagging those transitions that cause mismatches. In the following, we review the other process mining-based feature extraction approaches we use in Section 5. We denote $C_{\mathcal{L}} \subseteq \mathcal{A}$ the set of activities found in the traces of the event logs of $\mathcal{L}$. Besides, recall we denote $C_{\mathcal{L},N} \subseteq \mathcal{A}$ the set of activities found either in the traces of the event logs of $\mathcal{L}$ or in labeled transitions of a reference Petri net (Definition 3.7).

N-grams. The N-grams statistical process mining-based feature extraction involves counting the number of times an N-tuple of activities in the traces of the event logs of $\mathcal{L}$ occurs [10]. Let $C_{\mathcal{L}} \times \cdots \times C_{\mathcal{L}} = (C_{\mathcal{L}})^N$ be the set of N-grams and $|C_{\mathcal{L}}^N|$ the total number of N-grams. $\mathcal{F}_S : (\mathcal{B}(\mathcal{A}^*))^n \to \mathbb{N}^{n \times |C_{\mathcal{L}}^N|}$ associates each $L \in \mathcal{L}$ with a $|C_{\mathcal{L}}^N|$-tuple $(f_1, \ldots, f_{|C_{\mathcal{L}}^N|})$ such that $f_i \in \mathbb{N}$ counts the number of occurrences of the corresponding N-gram in L.

Directly-follows. The directly-follows statistical process mining-based feature extraction involves counting the number of times a directly-follows relation occurs in traces of $\mathcal{L}$, where a directly-follows relation is such that, given a trace $\sigma \in L$ and activities $a_1, a_2 \in \sigma$, either a_2 follows a_1 or vice versa [5]. Let $\mathcal{DF} = (C_{\mathcal{L}} \times C_{\mathcal{L}}) \setminus \{(c, c) : c \in C_{\mathcal{L}}\}$ be the set of directly-follows relations and $|\mathcal{DF}|$ the total number of directly-follows relations. $\mathcal{F}_S : (\mathcal{B}(\mathcal{A}^*))^n \to \mathbb{N}^{n \times |\mathcal{DF}|}$ associates each $L \in \mathcal{L}$ with a $|\mathcal{DF}|$-tuple $(f_1, \ldots, f_{|\mathcal{DF}|})$ such that $f_i \in \mathbb{N}$ counts the number of occurrences of a given directly-follows relation in L.

Token-based conformance checking. The token-based conformance checking process mining-based feature extraction [14,15] involves tracking the token dynamics while checking whether the traces of an event log meet the control-flow constraints of a reference Petri net. Specifically, when an activity of a trace is not allowed to execute according to the current marking of the reference Petri net, token-based conformance checking tracks the missing (m) tokens. In like manner, when the last activity of a trace executes, token-based conformance checking tracks the remaining (r) tokens in the places of the reference Petri net. Finally, token-based conformance checking tracks the consumed (c) and produced (r) tokens, and the total number of activities executed. Formally, $\mathcal{F}_{CC} : (\mathcal{B}(\mathcal{A}^*))^n \times \mathcal{N} \to \mathbb{R}^{n \times 3 + |C_{\mathcal{L},N}|}$ associates each $L \in \mathcal{L}$ with a $(3 + |C_{\mathcal{L},N}|)$-tuple $(F_{L,N}, f_m, f_r, f_{C_{\mathcal{L},N},1}, \ldots, f_{C_{\mathcal{L},N},|C_{\mathcal{L},N}|})$ such that $F_{L,N} = \frac{1}{|L|} \sum_{\sigma \in L} \frac{1}{2}(1 - \frac{m}{c}) + \frac{1}{2}(1 - \frac{r}{p})$ is the fitness obtained by token-based conformance checking [5], $f_m, f_r \in \mathbb{N}$ count the number of missing and remaining tokens, and $f_{C_{\mathcal{L},N},i}$ counts the number of times a given activity $c_i \in C_{\mathcal{L},N}$ executed.

4.3. Dimensionality reduction

Regardless of the type of process mining-based feature extraction used, the resulting tabular data are managed by a dimensionality reduction technique that reconstructs the data by encoding and decoding functions. This allows control-flow anomaly detection by the reconstruction-based approach, which has shown satisfying performance on account of the ability of dimensionality reduction to handle high-dimensional and possibly non-linear feature spaces.

Definition 4.2 (*Dimensionality Reduction Technique*). Let $T \in \mathbb{R}^{n \times f}$ be a matrix with n rows and f columns. A dimensionality reduction technique $\mathcal{DR}$ is a pair of functions defined as follows:

$$\begin{cases} \mathcal{DR}_E : \mathbb{R}^f \to \mathbb{R}^{f_R} \\ \mathcal{DR}_D : \mathbb{R}^f \to \mathbb{R}^f \end{cases} \qquad f_R < f, \tag{8}$$

where $\mathcal{DR}_E$ is the encoding function and $\mathcal{DR}_D$ the decoding function. $\hat{t}_i = \mathcal{DR}_D(t_i)$ is the reconstruction of the ith row of T by the decoding function of the technique after obtaining the encoding model upon the application of the encoding function. Let us define $\mathcal{E} : \mathbb{R}^f \to \mathbb{R}$ the function that calculates the reconstruction error. We denote $\mathcal{E}_{t_i} = \mathcal{E}(t_i) = \|t_i - \hat{t}_i\|$ the reconstruction error of t_i.

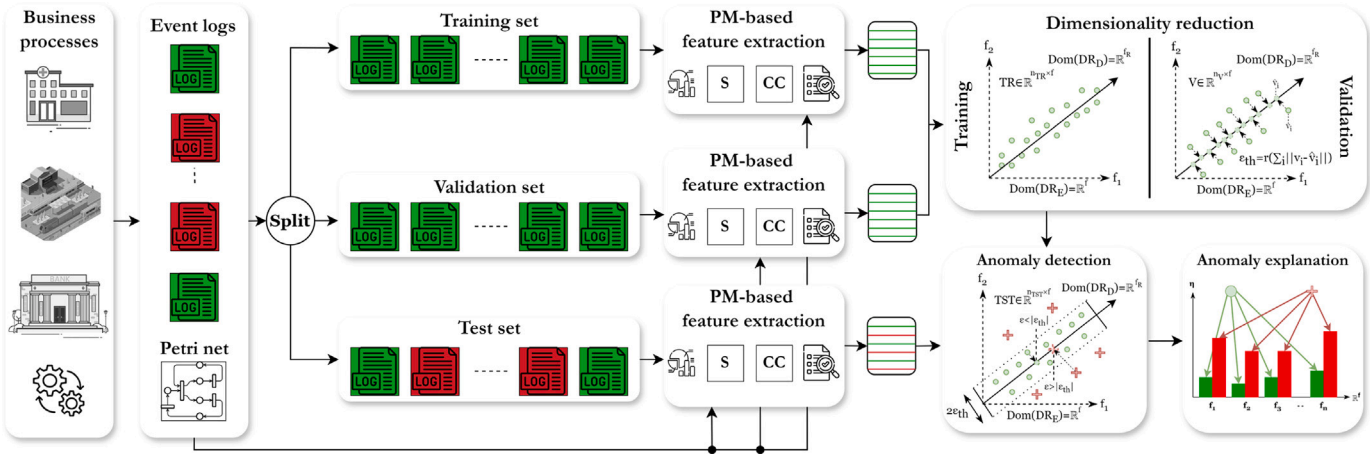


Fig. 6. The refined view of the proposed framework for combining process mining-based feature extraction with dimensionality reduction for control-flow anomaly detection.

The dimensionality reduction phase is two-stepped. These are the training and validation steps. Firstly, the encoding part of a dimensionality reduction technique $\mathcal{DR}_E$ is used to project the training tabular data $TR \in \mathbb{R}^{n_{TR} \times f}$ onto a new lower-dimensional coordinate system with domain $\mathbb{R}^{f_R}$. This coordinate system is used to project validation tabular data $V \in \mathbb{R}^{n_V \times f}$ onto it. After this projection, each validation sample is reconstructed in the original space with the decoding part of the dimensionality reduction technique $\mathcal{DR}_D$. The process repeats with different configurations of the dimensionality reduction technique until the reconstruction error on the validation set is minimized. In the following, we review the dimensionality reduction techniques we use in Section 5. We denote $T \in \mathbb{R}^{n \times f}$ a matrix obtained by process mining-based feature extraction with centered columns, i.e., the mean of each column is 0 and their standard deviation is 1, $t \in \mathbb{R}^f$ a row vector of T, and X' (x') the transpose of a generic matrix (row vector) X (x).

Principal component analysis. The Principal Component Analysis (PCA) is a widespread linear approach for dimensionality reduction [43]. The PCA projects T onto a new coordinate system by eigendecomposition of the T's covariance matrix: $\frac{1}{n}T'T = U\Delta U'$, where $U \in \mathbb{R}^{f \times f}$ is the matrix whose columns are eigenvectors and $\Delta \in \mathbb{R}^{f \times f}$ is a diagonal matrix containing the eigenvalues corresponding to the eigenvectors. The columns of U represent the so-called principal components, whose linear combination with the rows of T results in the projection of T onto the new coordinate system. The PCA is such that the first few principal components concentrate most of the explained variance. Hence, on the one hand, the encoding part of the PCA can be formulated as $\mathcal{DR}_E(t) = tU_P$, where $U_P \in \mathbb{R}^{f \times f_R}$ is U with the first f_R columns retained. On the other hand, the decoding part of the PCA can be formulated as $\mathcal{DR}_D(t) = \mathcal{DR}_E(t)U_P'$.

Sparse principal component analysis. The Sparse PCA (SPCA) extends the PCA by leaving out some features of the input space when building the new coordinate system to include only the significant ones [44]. This is enforced by converting the PCA formulation as a sparse regression problem with a regression regularization parameter and a sparsity coefficient, leading to the sparse matrix $U_S \in \mathbb{R}^{f \times f}$ and the reduced version $U_{S,P} \in \mathbb{R}^{f \times f_R}$. The encoding and decoding parts of the SPCA are the same as the PCA.

Kernel principal component analysis. The Kernel PCA (KPCA) extends the PCA by finding a high-dimensional hyperplane where T can be linearly separated using the so-called kernel trick [45]. Firstly, each row of T is projected onto a higher-dimensional hyperplane with domain $\mathbb{R}^{f_e}, f_e > f$ by a mapping $\phi : \mathbb{R}^f \to \mathbb{R}^{f_e}$. The covariance matrix of the transformed (centered) data is as follows: $C = \frac{1}{n}\sum_{i=1}^{n} \phi(t_i)'\phi(t_i) \in \mathbb{R}^{f_e \times f_e}$, where $t_i \in \mathbb{R}^f$ is a row of T. Let us consider the kernel function $k(t_i, t_j) = \phi(t_i)\phi(t_j)' \in \mathbb{R}$ and the kernel matrix $K \in \mathbb{R}^{n \times n}$ such that $K_{ij} = k(t_i, t_j)$. The eigenvalue equation $\lambda V = CV$, where $\lambda \in \mathbb{R}$ and $V \in \mathbb{R}^{f_e}$, transforms to $K\nu = n\lambda\nu$, where $\nu \in \mathbb{R}^n$ is an eigenvector of K. The eigenvectors represent the kernel principal components, leading to the matrix $U_e \in \mathbb{R}^{n \times n}$ and the reduced version $U_{e,P} \in \mathbb{R}^{n \times f_R}$. The encoding part of the KPCA can be formulated as $\mathcal{DR}_E(t) = \kappa(t)U_{e,P}, \kappa(t) = (\phi(t, t_1), \ldots, \phi(t, t_n)) \in \mathbb{R}^n$. The decoding part involves ridge regression, which requires solving the following minimization problem: $\min_\beta\{\|U_{e,P}\beta - T\|^2 + \gamma\|\beta\|^2\}$, where $\|\cdot\|^2$ indicates the squared norm, $\gamma \in \mathbb{R}$ is a regularization term, and $\beta \in \mathbb{R}^{f_e \times f}$ the matrix of regression components. Finally, the decoding part of the KPCA can be formulated as $\mathcal{DR}_D(t) = \kappa(t)\beta$.

Autoencoder. The autoencoder is a widespread feed-forward neural network approach for dimensionality reduction which accounts for non-linear dependencies in T's features through the non-linear transformations of the network's neurons [46]. The simplest autoencoder structure is a symmetric feed-forward network with three fully-connected hidden layers, namely the encoder, code and decoder layers. The code layer implements $\mathcal{DR}_E(t) = W_C\sigma_E(W_E t + b_E) + b_C$, where $b_E \in \mathbb{R}^{f_{E,D}}$ and $b_C \in \mathbb{R}^{f_R}$ are the encoder and code bias vectors, $W_E \in \mathbb{R}^{f_{E,D} \times f}$ and $W_C \in \mathbb{R}^{f_R \times f_{E,D}}$ are the encoder and code weight matrices, and σ_E the activation function applied by the encoder layer's neurons. The output provides the reconstructed data, implementing the decoding part $\mathcal{DR}_D(t) = W_O\sigma_D(W_D\mathcal{DR}_E(t) + b_D) + b_O$, where $b_D \in \mathbb{R}^{f_{E,D}}$ and $b_O \in \mathbb{R}^f$ are the decoder and output bias vectors, $W_D \in \mathbb{R}^{f_R \times f_{E,D}}$ and $W_O \in \mathbb{R}^{f \times f_{E,D}}$ are the decoder and output weight matrices, and σ_D the activation function applied by the decoder layer's neurons.

Table 2 briefly describes the techniques above, mentioning the approach followed, the complexity and type of the encoding and decoding parts, and the framework formulation. The choice of the specific technique depends on the computational resources available and the data at hand. For example, on the one hand, the PCA can be preferred to other methods if the feature space of data does not exhibit non-linear relationships and/or there are limited computational resources available. On the other hand, if computational resources are not constrained and there are loads of data with non-linear relationships in the feature space, the autoencoder could provide better results. Finally, it is worth mentioning that there are other dimensionality reduction techniques, such as the linear discriminant analysis [47] and t-distributed stochastic neighbor embedding [48], as well as further extensions of the ones we presented, such as the robust PCA [49] and variational autoencoder [50].

4.4. Anomaly detection and explanation

A threshold on the reconstruction error $\mathcal{E}_{th}$ can be calculated for classifying test data during anomaly detection. Let $\mathcal{E}_{v_i} = \|v_i - \hat{v}_i\|, i \in$

Table 2
A set of dimensionality reduction techniques with their framework formulation.

Technique	Approach	Complexity	Type	Formulation
PCA	Eigendecomposition	Low	Linear	$DR_E(t) = tU_P$ $DR_D(t) = DR_E(t)U'_P$
SPCA	Sparse regression	Medium	Linear	$DR_E(t) = tU_{S,P}$ $DR_D(t) = DR_E(t)U'_{S,P}$
KPCA	Kernel trick, eigendecomposition, kernel ridge regression	Medium	Non-linear	$DR_E(t) = \kappa(t)U_{e,P}$ $DR_D(t) = \kappa(t)\beta$
Autoencoder	Neural network	High	Non-linear	$DR_E(t) = W_C\sigma_E(W_E t + b_E) + b_C$ $DR_D(t) = W_O\sigma_D(W_D DR_E(t) + b_D) + b_O$

$\{1, \dots, n_V\}$ be the reconstruction errors of rows $v_i \in \mathbb{R}^f$ of validation tabular data $V \in \mathbb{R}^{n_V \times f}$. Let $\sum_i \|v_i - \hat{v}_i\|$ be the sum of such errors. $\mathcal{E}_{th}$ is calculated by a function r over the sum of reconstruction errors, i.e., $\mathcal{E}_{th} = r(\sum_i \|v_i - \hat{v}_i\|)$. For example, r can be the mean squared error: $\mathcal{E}_{th} = \frac{\sum_i \|v_i - \hat{v}_i\|^2}{n_V}$.

Definition 4.3 (*Anomaly Detection*)**.** Let $t \in \mathbb{R}^f$ be a test sample, DR_D the decoding part of a dimensionality reduction technique tailored to a training set, $\hat{t} = DR_D(t)$ the reconstruction of t using DR_D, $\mathcal{E}_t = \|t - \hat{t}\|$ the reconstruction error of t, and $\mathcal{E}_{th}$ a threshold to the reconstruction error obtained with a validation set. The classification of t is *normal* if $\mathcal{E}_t < \mathcal{E}_{th}$, *anomalous* otherwise.

The combination of a process mining-based feature extraction approach and a dimensionality reduction technique forms a *framework technique*. The **detection-definition** of framework techniques is linked to Definition 4.3: the framework techniques identify as anomalous those event logs whose process mining-based feature extraction leads to a reconstruction error higher than the threshold. However, while this definition outlines *what is an anomaly*, it does not provide insight into *why* the event log led to an anomalous reconstruction error. There are several anomaly explanation techniques available to provide an **explanation-definition**. In this work, we focus on additive feature-based explanation, which explains the output of an anomaly detection technique by superposition of the impact of the features of input data [51].

Definition 4.4 (*Anomaly Explanation*)**.** Let $t \in \mathbb{R}^f$ be a test sample, DR_D the decoding part of a dimensionality reduction technique tailored to a training set, $DR_D(t) = \hat{t}$ the reconstruction of t using DR_D, and $\mathcal{E}$ the function that calculates the reconstruction error. The additive explanation of $\mathcal{E}_t$ is $\tilde{\mathcal{E}}_t = \theta + \sum_{i=1}^{f} \eta_i \mathcal{E}_t, \theta, \eta_i \in \mathbb{R}$. $\eta = (\eta_1, \dots, \eta_f) \in \mathbb{R}^f$ is the *anomaly explanation*, where η_i denotes the effect of feature f_i on $\mathcal{E}_t$.

Several methods such as Local Interpretable Model-agnostic Explanations, DeepLift, and Classic Shapley Value Estimation can be used to calculate η. These methods are unified under a framework that calculates the so-called SHapley Additive exPlanations (SHAP) values [51]. These values have been used to explain the results of several dimensionality reduction approaches to anomaly detection [52,53]. In conclusion, we would like to remark that not only does this explanation-definition depend on the dimensionality reduction technique but also on process mining-based feature extraction. For example, SHAP values linked to the features of our proposed process mining-based feature extraction approach directly connect to the elements of the reference Petri net. This connection gives additional model-based insight into the reason why test data are classified as anomalous.

5. Experiments

In this section, we first describe the datasets used. Secondly, we detail the framework techniques and baseline conformance checking-based techniques with which we experiment. Finally, the goals of the experiments, the performance metrics used, and the results of the experiments are reported and discussed.

5.1. Datasets

The datasets include well-known benchmarks for anomaly detection in event logs (PDC 2020/2021), simulation related to a railways case study (ERTMS), and real-world data from a healthcare case study (COVAS).

PDC 2020/2021. The PDC datasets[1] are generated by simulation of a variety of Petri nets. For each PDC dataset, the different Petri nets are obtained by tuning several characteristics of a base Petri net, introducing more complex control-flow patterns such as non-local dependencies between activities. PDC 2020 contains two event log sets named `ground_truth` and `training`. Each event log of the `training` set is generated by randomly walking one of the Petri net variants 1000 times, thus producing 1000 traces. In addition to simulating the Petri net variant, the training set contains an event log corresponding to a Petri net variant with noise introduced in the traces. For example, given N the Petri net variant and L_N the simulated event log of the `training` set, the authors produce the noisy event log $\tilde{L}_N$. Regarding the `ground_truth` set, there are as many event logs of 1000 traces as Petri net variants. However, the authors did not mention whether these event logs were obtained by randomly walking the corresponding Petri net variant. Instead, they label each trace as either `positive` or `negative` to allow practitioners to test their algorithms. In our experimentation, we organize the dataset as follows. We take the `training` event logs of two Petri net variants and the corresponding `ground_truth` event logs. Then, we consider normal one of the two variants, and anomalous the other. Specifically, all the traces of the `training` event logs of the normal variant are considered normal, and `positive` traces of the `ground_truth` event log are also considered normal. Conversely, all traces of the `training` event logs of the anomalous variant are considered anomalous, and `positive` traces of the `ground_truth` event log are also considered anomalous because they fit the anomalous variant. The same process is applied to the PDC 2021 dataset. Finally, we consider the base Petri net in both datasets as the input Petri net for conformance checking process mining-based feature extraction.

ERTMS. The European Rail Traffic Management System (ERTMS)[2] is a standard that regulates European railways for the smooth and safe operation of trains across Europe. ERTMS prescribes textual requirements for several procedures and provides informal activity and state diagrams to facilitate the implementation of these procedures. In this work, we consider the RBC/RBC Handover procedure, which regulates the process that trains must follow when their journey involves changing their supervision from one RBC to another, where the RBC is an entity whose supervision scope involves a limited area of on-track equipment and trains. This procedure was considered in [15] and used as proof of concept for the experimentation of anomaly detection

[1] https://www.tf-pm.org/competitions-awards/discovery-contest
[2] https://www.ertms.net/

in simulated traces generated by a BPMN model of the RBC/RBC Handover. We replicate the anomaly injection process proposed by the authors as follows. We consider the resources associated with the activities of the BPMN model of the RBC/RBC Handover and randomly walk the model to generate 2000 traces. Then, we split these 2000 traces into two normal and anomalous event logs of 1000 traces each. Each trace of both event logs is injected with anomalies. This injection: randomly selects a resource among those involved in RBC/RBC Handover, considers the activities associated with this resource, and proceeds to introduce missed, duplicated, or wrongly-ordered activities in the trace uniformly according to a given probability. This probability depends on the nature of the trace. In our simulations, we set the probability to 5% for each anomaly type in normal traces and 25% for each anomaly type in anomalous traces. This means that, given a normal trace and assuming the introduction of each anomaly is independent, there is $1 - 0.95^3 \approx 15\%$ probability of having at least one of the anomalies, whereas, given an anomalous trace and assuming the same, there is $1 - 0.75^3 \approx 57\%$ probability of having at least one of the anomalies.

COVAS. The COVID-19 Aachen Study (COVAS) involved monitoring the treatment process of COVID-19 patients from March 2020 to June 2021 [54,55]. The data are preprocessed by cleaning incomplete traces, using abstractions for some activities of negligible significance, and filtering infrequent variants. Furthermore, the data are split into three event logs, according to three different time frames matching the first, second and third infection waves. This split is backed up by literature, as there have been significant differences in COVID-19 patients' clinical course and treatment among these three infection waves. The three COVAS event logs have 106, 59, and 22 traces. The traces are labeled following the viewpoint of the first infection wave. Hence, the first-wave event log contains normal traces, whereas the other second- and third-wave event logs are anomalous. To achieve a dataset whose dimension is comparable to the others, we augment the data by random re-sampling of traces of the same kind. Finally, we consider a Petri net related to the first infection wave and discovered by authors in [55] as the input Petri net for conformance checking process mining-based feature extraction.

For all datasets, the normal and anomalous traces are grouped into event logs of 5 traces each, resulting in two sets of normal and anomalous event logs. All anomalous event logs are retained in the test set, whereas the normal event logs are split as follows. 25% of normal event logs are held out and placed into the test set and the remainder is used as the training set. Next, 20% of the training set is held out and placed into the validation set. Table 3 collects the properties of the datasets. These properties outline whether the target application is real or abstract, the nature is synthetic or real-world, and the labeling is synthetic or done by domain experts. For example, let us consider the ERTMS dataset. It is a real application because it involves a railway standard, its nature is synthetic because the data is obtained by simulating the model, and the labels are synthetic since they depend on the probabilistic injection of anomalies. Furthermore, we report the number of normal and anomalous traces, and the resulting training, validation and test event logs obtained according to the percentages above.

Table 3
The properties of the datasets for the experiments.

Property	COVAS	ERTMS	PDC 2020	PDC 2021
Application	Real	Real	Abstract	Abstract
Nature	Real-world	Synthetic	Synthetic	Synthetic
Labeling	By domain experts	Synthetic	Synthetic	Synthetic
Normal traces	1000 (augmented)	1000	3020	4250
Anomalous traces	1000 (augmented)	1000	2496	2125
Training event logs	120	120	363	511
Validation event logs	30	30	90	127
Test event logs	250	250	650	637

5.2. Techniques

5.2.1. Baselines

In this work, we consider the conformance checking-based techniques that rely on fitness thresholding as the baselines [6–9]. As mentioned, Fig. 3 highlights that setting a threshold for the fitness measure is challenging since the fitness values are spread inconsistently, even though the normal event log and the reference Petri net are, respectively, the set of first-wave traces and Petri net from the COVAS case study. This is a critical issue, especially because the reference Petri net was obtained after laborious pre-processing, hence it should be of reasonable quality. Regardless, conformance checking-based techniques that rely on fitness thresholding are worth comparing due to their explainable nature. In fact, the fitness measure comes from a clear-cut model-based definition (e.g., Definition 3.5), which provides a straightforward explanation of why an event log is classified as anomalous.

We implement two baseline conformance checking-based techniques. Firstly, token-based/alignment-based conformance checking is applied to all event logs in the validation set against the reference Petri net to obtain the fitness measures for all the validation event logs. Secondly, a threshold is automatically assigned using the minimum of all the fitness measures. This is because, given that every event log in the validation set is normal, the best generalization should be achieved by considering the least fitting event log. We name AB_CC_B and TB_CC_B the baselines employing token-based and alignment-based conformance checking, respectively.

5.2.2. Framework

The framework allows implementing a wide range of conformance checking-based and conformance checking-independent control-flow anomaly detection techniques by combining a process mining-based feature extraction approach with a dimensionality reduction technique. This combination allows assessing the impact of diverse process mining-based feature extraction approaches on the results [4,10] and replicating the reconstruction-based approach for control-flow anomaly detection [16–20].

We implement 16 framework techniques for control-flow anomaly detection. These techniques combine the proposed (Section 3) and reviewed (Section 4) process mining-based feature extraction approaches, namely the alignment-based conformance checking (AB_CC), token-based conformance checking (TB_CC), N-grams (NG) and directly-follows (DF) approaches, and the reviewed (Section 4) dimensionality reduction techniques, namely the PCA, SPCA, KPCA and autoencoder (AE). Table 4 provides an overview of the parameters used to configure the dimensionality reduction techniques during the validation step of the framework, which performs an exhaustive search of all the parameter values to minimize the reconstruction error. Each framework technique is labeled by a pair of values (PF, DR), where PF $\in$ {AB_CC, TB_CC, NG, DF} indicates the process mining-based feature extraction approach and DR $\in$ {PCA, KPCA, SPCA, AE} the dimensionality reduction technique.

5.3. Evaluation

The experiments are designed to evaluate two aspects of our research, which are the following: (1) compare to the baseline conformance checking-based techniques the detection effectiveness and explainability of the framework techniques using our proposed process mining-based feature extraction approach; and (2) compare the detection effectiveness of conformance checking-based and conformance checking-independent framework techniques and evaluate the factors that impact their performance. The performance of the techniques is evaluated by their accuracy (A), recall (R), precision (P) and F1-Score ($F1$).

Table 4
Parameters used to configure the dimensionality reduction techniques during the validation step of the framework. N/A = Not Applicable.

DR	Parameter #1	Parameter #2	Parameter #3	Parameter #4	Parameter #5	Parameter #6
PCA	f_R {2, 4, 8, 16}	N/A	N/A	N/A	N/A	N/A
SPCA	f_R {2, 4, 8, 16}	Regression regularization {0.01, 0.1, 0.25, 0.5, 0.75, 1.0}	Sparsity coefficient {0.1, 0.5, 1, 2, 3}	N/A	N/A	N/A
KPCA	f_R {2, 4, 8, 16}	Regression regularization {0.01, 0.1, 0.25, 0.5, 0.75, 1.0}	Kernel {poly, rbf, sigmoid}	Kernel coefficient {0.01, 0.05, 0.1}	Polynomial degree {3, 4, 5, 6}	N/A
Autoencoder	f_R {2, 4, 8, 16}	Hidden neurons {32, 64, 128, 256}	Optimizer {adam, rmsprop, SGD}	Batch size {8, 16, 32, 64}	Epochs {100, 250, 500}	Activation function {ReLu}

Definition 5.1 (*Accuracy, Recall, Precision and F1-score*)**.** Let $\mathcal{L} \in (\mathcal{B}(\mathcal{A}^*))^n$ be an n-tuple of labeled test event logs. Let positive samples be normal event logs and negative samples be anomalous event logs. Finally, let TP, TN, FP and FN be, respectively, the true positives, true negatives, false positives, and false negatives. Accuracy A, recall R, precision P, and F1-Score $F1$ are defined as follows:

$$A = \frac{TP+TN}{TP+TN+FP+FN};\ R = \frac{TP}{TP+FN};$$
$$P = \frac{TP}{TP+FP};\ F1 = \frac{2RP}{R+P}. \tag{9}$$

The software for the experiments is implemented using Python and was run on a Windows 10 machine with an Intel® Core™ i9-11900K CPU @ 3.50 GHz and 32 GB of RAM memory. The software uses libraries for machine learning and process mining, such as `scikit-learn`, `tensorflow` and `pm4py`. In addition, it includes a set of DOS batch scripts to automatically replicate the experiments carried out in this work, available online on GitHub.[3]

5.3.1. Baselines comparison

In this part, we compare the detection effectiveness of the framework techniques using our proposed process mining-based feature extraction approach, namely (AB_CC, PCA), (AB_CC, SPCA), (AB_CC, KPCA) and (AB_CC, AE), with the baselines. Each technique is applied to each dataset 5 times to account for statistical fluctuations due to the random split of training, validation and test sets.

Results. Table 5 reports the mean A, R, P and $F1$ measures and their standard deviation in subscripts for each technique and dataset. Except for AB_CC_B's R measure for the COVAS dataset (89.1%), the conformance checking-based framework techniques using our proposed process mining-based feature extraction approach outperform both baselines, in some cases by a very large degree. For the PDC 2020 and 2021 datasets, the (AB_CC, AE) technique achieves 97.3% and 87.2% $F1$, whereas the AB_CC_B and TB_CC_B baselines achieve 36.1% and 19.4% $F1$ for the PDC 2020 dataset, and 33.4% and 56.6% $F1$ for the PDC 2021 dataset. The performance gap is smaller for the ERTMS and COVAS datasets, for which (AB_CC, AE) and (AB_CC, KPCA) achieve 85.2% and 88.5% $F1$, whereas the AB_CC_B and TB_CC_B baselines both achieve 78.8% $F1$ for the ERTMS dataset, and 79.1% and 18.4% $F1$ for the COVAS dataset.

Framework techniques explanation. Let us consider the anomaly explanation of techniques (AB_CC, PCA), (AB_CC, SPCA), (AB_CC, KPCA), (AB_CC, AE) for the PDC 2020 dataset, whose Petri net is depicted on the left side of Fig. 5. Fig. 7 shows the SHAP (absolute) values for each technique. These values show the influence of the per-activity cost of each event log activity and labeled transition of the reference Petri net on the reconstruction error. For each activity, the mean SHAP value of anomalous (red) and normal (green) event logs is shown. The bar plot highlights that the high detection effectiveness of each framework technique is due to: (1) the small reconstruction error of per-activity costs in normal event logs; and (2) the high reconstruction error of per-activity costs in anomalous event logs. For example, the SHAP values of (AB_CC, AE) highlight that the influence on reconstruction error of per-activity costs of normal event logs is strictly less than the reconstruction error of per-activity costs of anomalous event logs. In addition, each technique outlines that `t21` and `t75` influence the reconstruction error significantly, which suggests that the traces of anomalous event logs often lead to mismatching moves such as (`t21`, ≫), (≫,`t21`), which, in turn, could be due to `t21` being duplicated, skipped, or wrongly-ordered according to the Petri net's control-flow relations. In the case of `t75`, the activity is simply unknown, hence leading to the mismatching move (`t75`, ≫) in the alignments.

Baselines explanation. Let us consider the distribution of fitness measures of normal and anomalous event logs for the AB_CC_B and TB_CC_B baselines. Fig. 8 shows these distributions for each dataset, highlighting the intersection of the anomalous and normal distributions. These histograms provide insight into why the baseline approaches may fail at correctly classifying normal and anomalous event logs. In fact, the worst performance is obtained when there is excessive overlap of fitness values from normal and anomalous event logs, such as the PDC 2020/2021 datasets, where the four histograms of AB_CC_B and TB_CC_B show a significant overlap of normal and anomalous fitness values. Clearly, a higher-quality Petri net could provide much better results with the baselines. This is the case for the ERTMS dataset, where the anomalous and normal histograms of both the baselines are better separated, and, therefore, fitness thresholding is much more effective. Finally, the COVAS dataset leads to little overlap for AB_CC_B, whereas, for TB_CC_B, there is significant overlap.

5.3.2. Framework techniques comparisons

In this part, we compare the detection effectiveness of framework techniques using all four process mining-based feature extraction approaches. Each technique is applied to each dataset 5 times to account for statistical fluctuations due to the random split of training, validation and test sets.

Results. Table 6 reports the mean A, R, P and $F1$ measures and their standard deviation in subscripts for each technique and dataset. For the PDC 2020 and COVAS datasets, (AB_CC, AE) and (AB_CC, KPCA) show the best performance, achieving 97.3% and 88.5% $F1$. For the PDC 2021 and ERTMS datasets, (TB_CC, SPCA) and (TB_CC, AE) show the best performance, achieving 88.9% and 90.5% $F1$. Arguably, TB_CC process mining-based feature extraction is better than AB_CC for the PDC 2021 and ERTMS dataset because of the explanation in Fig. 8. From this figure, the fitness measures of the TB_CC_B baseline with respect to the ERTMS dataset are better separated than those fitness measures of the AB_CC_B baseline. Hence, the TB_CC approach could have led to better features than AB_CC. Fig. 9 depicts the bar plots that capture the impact of the different process mining-based feature extraction approaches on the performance measures for each dataset. In particular, given a PF value, the dimensionality reduction technique providing the best performance is considered to compare the best-performing PF-DR value pairs. For example, given the PDC 2020 dataset, the best-performing PF-DR value pairs are (AB_CC, AE), (TB_CC, AE), (DF, AE) and (NG, AE), with $F1$ equal to 97.3%, 95.0%, 65.4% and 80.3%.

[3] https://github.com/francescovitale/pm_dr_framework

Table 5
The mean *A*, *R*, *P* and *F*1 measures and their standard deviation in subscripts for each framework/baseline technique and dataset. Measures in bold indicate the highest figure per dataset.

Technique	PDC 2020				PDC 2021				ERTMS				COVAS			
	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)
(AB_CC, PCA)	$96.2_{1.0}$	$92.7_{1.2}$	$96.5_{2.2}$	$94.4_{1.4}$	$86.0_{2.4}$	$83.5_{2.4}$	$84.8_{3.0}$	$84.0_{2.6}$	$87.6_{1.3}$	$76.8_{4.6}$	$82.3_{1.6}$	$78.8_{3.4}$	$85.1_{6.4}$	$77.5_{5.8}$	$78.5_{9.9}$	$77.5_{7.7}$
(AB_CC, SPCA)	$96.8_{1.0}$	$93.8_{1.4}$	$97.1_{2.0}$	$95.3_{1.4}$	$86.3_{2.9}$	$83.1_{2.6}$	$85.8_{3.8}$	$84.1_{3.1}$	$88.2_{2.3}$	$78.8_{5.5}$	$82.5_{3.4}$	$80.3_{4.7}$	$91.0_{4.8}$	$87.6_{7.2}$	$85.8_{7.5}$	$86.5_{7.0}$
(AB_CC, KPCA)	$96.4_{0.7}$	$92.1_{1.5}$	$97.8_{0.4}$	$94.6_{1.1}$	$86.6_{1.4}$	$83.5_{2.3}$	$85.9_{1.6}$	$84.4_{1.9}$	$89.3_{1.7}$	$\mathbf{87.8_{2.3}}$	$83.1_{3.4}$	$84.7_{1.7}$	$\mathbf{92.8_{1.0}}$	$87.9_{3.2}$	$\mathbf{89.5_{1.1}}$	$\mathbf{88.5_{2.0}}$
(AB_CC, AE)	$\mathbf{98.1_{0.3}}$	$\mathbf{96.6_{1.3}}$	$\mathbf{98.1_{0.5}}$	$\mathbf{97.3_{0.5}}$	$\mathbf{88.9_{1.1}}$	$\mathbf{86.2_{1.3}}$	$\mathbf{88.4_{1.4}}$	$\mathbf{87.2_{1.3}}$	$\mathbf{91.2_{1.9}}$	$83.3_{5.8}$	$\mathbf{88.3_{1.6}}$	$\mathbf{85.2_{4.1}}$	$91.4_{3.1}$	$85.6_{6.5}$	$87.0_{5.0}$	$86.2_{5.7}$
AB_CC_B	$36.6_{4.3}$	$58.6_{2.2}$	$62.3_{0.5}$	$36.1_{4.0}$	$38.7_{4.6}$	$53.8_{3.2}$	$65.9_{1.9}$	$33.4_{6.7}$	$82.3_{10.0}$	$87.6_{5.6}$	$78.2_{8.7}$	$78.8_{10.3}$	$82.8_{4.9}$	$\mathbf{89.1_{2.7}}$	$77.3_{4.4}$	$79.1_{5.2}$
TB_CC_B	$23.1_{0.2}$	$49.9_{0.6}$	$54.7_{7.6}$	$19.4_{0.5}$	$57.0_{2.4}$	$67.6_{1.8}$	$71.4_{0.7}$	$56.6_{2.7}$	$81.9_{11.9}$	$87.7_{6.5}$	$78.7_{9.5}$	$78.8_{12.0}$	$21.2_{0.0}$	$50.4_{0.0}$	$56.9_{4.0}$	$18.4_{0.0}$

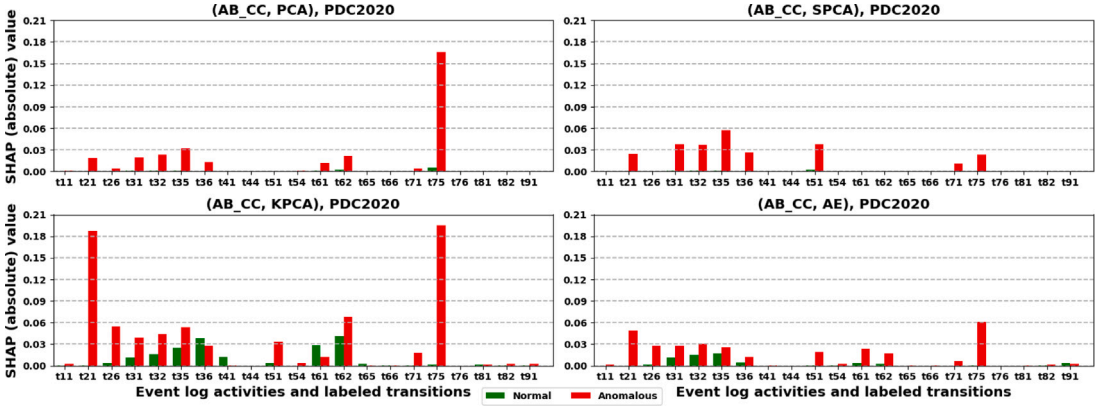


Fig. 7. SHAP (absolute) values related to the additive feature-based explanation of the four conformance checking-based framework techniques (AB_CC,PCA), (AB_CC,SPCA), (AB_CC,KPCA), (AB_CC,AE) using the PDC 2020 dataset.

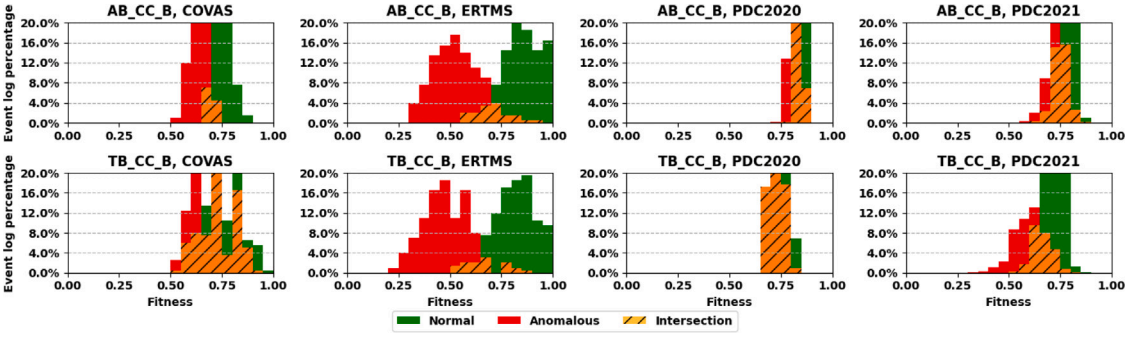


Fig. 8. The normal and anomalous histograms that show the frequency of occurrence of the fitness measures of normal and anomalous event logs per dataset for the AB_CC_B and TB_CC_B baseline approaches. The intersection highlights the overlap between anomalous and normal histograms.

Table 6
The mean *A*, *R*, *P* and *F*1 and their standard deviation in subscripts for each framework and baseline technique and dataset. Measures in bold indicate the highest figure per dataset.

Technique	PDC 2020				PDC 2021				ERTMS				COVAS			
	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)
(AB_CC, PCA)	$96.2_{1.0}$	$92.7_{1.2}$	$96.5_{2.2}$	$94.4_{1.4}$	$86.0_{2.4}$	$83.5_{2.4}$	$84.8_{3.0}$	$84.0_{2.6}$	$87.6_{1.3}$	$76.8_{4.6}$	$82.3_{1.6}$	$78.8_{3.4}$	$85.1_{6.4}$	$77.5_{5.8}$	$78.5_{9.9}$	$77.5_{7.7}$
(AB_CC, SPCA)	$96.8_{1.0}$	$93.8_{1.4}$	$97.1_{2.0}$	$95.3_{1.4}$	$86.3_{2.9}$	$83.1_{2.6}$	$85.8_{3.8}$	$84.1_{3.1}$	$88.2_{2.3}$	$78.8_{5.5}$	$82.5_{3.4}$	$80.3_{4.7}$	$91.0_{4.8}$	$87.6_{7.2}$	$85.8_{7.5}$	$86.5_{7.0}$
(AB_CC, KPCA)	$96.4_{0.7}$	$92.1_{1.5}$	$97.8_{0.4}$	$94.6_{1.1}$	$86.6_{1.4}$	$83.5_{2.3}$	$85.9_{1.6}$	$84.4_{1.9}$	$89.3_{1.7}$	$87.8_{2.3}$	$83.1_{3.4}$	$84.7_{1.7}$	$\mathbf{92.8_{1.0}}$	$\mathbf{87.9_{3.2}}$	$\mathbf{89.5_{1.1}}$	$\mathbf{88.5_{2.0}}$
(AB_CC, AE)	$\mathbf{98.1_{0.3}}$	$\mathbf{96.6_{1.3}}$	$\mathbf{98.1_{0.5}}$	$\mathbf{97.3_{0.5}}$	$88.9_{1.1}$	$86.2_{1.3}$	$88.4_{1.4}$	$87.2_{1.3}$	$91.2_{1.9}$	$83.3_{5.8}$	$88.3_{1.6}$	$85.2_{4.1}$	$91.4_{3.1}$	$85.6_{6.5}$	$87.0_{5.0}$	$86.2_{5.7}$
(TB_CC, PCA)	$96.1_{0.3}$	$91.7_{3.1}$	$97.5_{0.2}$	$94.1_{0.5}$	$90.0_{0.6}$	$86.6_{0.6}$	$90.7_{1.0}$	$88.2_{0.7}$	$89.6_{2.1}$	$75.6_{5.3}$	$90.6_{1.7}$	$80.0_{5.1}$	$52.1_{3.0}$	$57.9_{3.0}$	$55.2_{2.1}$	$48.6_{0.9}$
(TB_CC, SPCA)	$96.5_{0.7}$	$92.9_{1.4}$	$97.2_{1.5}$	$94.8_{1.1}$	$\mathbf{90.6_{0.9}}$	$\mathbf{87.1_{1.3}}$	$\mathbf{91.7_{0.8}}$	$\mathbf{88.9_{1.2}}$	$92.6_{1.0}$	$85.5_{3.6}$	$90.9_{1.7}$	$87.6_{2.3}$	$55.6_{1.7}$	$61.2_{1.3}$	$57.2_{0.8}$	$52.0_{1.2}$
(TB_CC, KPCA)	$96.5_{1.4}$	$92.6_{3.1}$	$97.5_{1.0}$	$94.7_{2.3}$	$67.0_{4.8}$	$70.2_{3.1}$	$68.2_{2.4}$	$66.3_{4.4}$	$92.4_{1.4}$	$88.6_{1.6}$	$88.1_{2.6}$	$88.2_{1.7}$	$40.8_{5.5}$	$53.1_{3.2}$	$52.4_{2.6}$	$39.6_{3.9}$
(TB_CC, AE)	$96.6_{0.4}$	$92.6_{1.0}$	$97.9_{0.2}$	$95.0_{0.7}$	$90.0_{0.6}$	$87.3_{1.8}$	$91.3_{0.8}$	$88.8_{1.5}$	$\mathbf{94.0_{0.5}}$	$\mathbf{89.8_{2.7}}$	$\mathbf{91.5_{1.3}}$	$\mathbf{90.5_{1.0}}$	$73.8_{3.1}$	$69.7_{2.8}$	$64.8_{2.3}$	$65.7_{2.7}$
(DF, PCA)	$58.6_{5.6}$	$60.4_{3.4}$	$57.4_{2.4}$	$54.5_{4.2}$	$79.4_{1.6}$	$77.1_{1.4}$	$77.0_{1.8}$	$76.9_{1.5}$	$89.2_{1.1}$	$76.9_{3.8}$	$87.1_{1.6}$	$80.4_{3.4}$	$89.5_{2.0}$	$78.6_{5.4}$	$86.6_{2.4}$	$81.4_{4.5}$
(DF, SPCA)	$62.2_{2.8}$	$62.4_{0.4}$	$58.9_{0.4}$	$57.2_{1.5}$	$80.4_{1.7}$	$78.6_{1.2}$	$78.3_{1.9}$	$78.2_{1.3}$	$88.4_{2.0}$	$74.8_{5.7}$	$86.7_{1.8}$	$78.4_{5.0}$	$89.2_{1.1}$	$78.4_{4.3}$	$85.8_{0.3}$	$81.1_{3.1}$
(DF, KPCA)	$70.3_{2.2}$	$67.9_{1.7}$	$63.7_{1.5}$	$64.1_{1.9}$	$78.4_{0.9}$	$76.6_{0.8}$	$75.9_{1.0}$	$76.1_{0.6}$	$89.2_{1.1}$	$77.2_{3.2}$	$86.8_{1.3}$	$80.6_{2.8}$	$91.9_{0.0}$	$83.5_{1.2}$	$90.1_{1.8}$	$86.2_{1.3}$
(DF, AE)	$71.6_{2.2}$	$69.3_{0.0}$	$64.8_{0.0}$	$65.4_{0.0}$	$70.1_{0.0}$	$72.3_{0.0}$	$69.8_{0.0}$	$69.2_{0.0}$	$89.7_{1.3}$	$80.2_{2.4}$	$86.0_{2.6}$	$82.6_{2.4}$	$90.8_{1.4}$	$83.9_{3.2}$	$86.4_{2.1}$	$85.0_{2.7}$
(NG, PCA)	$84.0_{1.0}$	$75.5_{2.1}$	$77.3_{1.6}$	$76.3_{1.8}$	$75.0_{2.5}$	$73.5_{2.2}$	$72.3_{2.4}$	$72.7_{2.4}$	$87.6_{1.1}$	$77.8_{3.7}$	$81.6_{1.5}$	$79.3_{2.6}$	$91.3_{0.8}$	$83.5_{1.4}$	$88.3_{1.7}$	$85.6_{4.0}$
(NG, SPCA)	$83.8_{0.0}$	$76.0_{1.7}$	$77.1_{1.0}$	$76.5_{1.1}$	$75.3_{0.9}$	$73.5_{1.8}$	$72.6_{1.0}$	$72.8_{1.3}$	$87.6_{2.0}$	$76.8_{6.8}$	$82.1_{2.2}$	$78.5_{5.4}$	$87.6_{0.8}$	$75.3_{1.5}$	$83.0_{1.7}$	$78.1_{1.5}$
(NG, KPCA)	$85.6_{0.7}$	$75.7_{3.1}$	$80.9_{0.9}$	$77.6_{2.0}$	$72.9_{2.5}$	$73.1_{1.6}$	$71.1_{1.4}$	$71.2_{1.6}$	$87.6_{1.4}$	$75.3_{1.1}$	$83.2_{3.9}$	$78.2_{2.0}$	$92.2_{2.1}$	$86.0_{5.6}$	$89.1_{2.5}$	$87.2_{4.0}$
(NG, AE)	$87.2_{0.0}$	$78.2_{0.0}$	$83.3_{0.0}$	$80.3_{0.0}$	$77.0_{0.0}$	$75.3_{0.0}$	$74.3_{0.0}$	$74.7_{0.0}$	$86.0_{0.8}$	$73.1_{2.3}$	$79.5_{1.4}$	$75.5_{2.0}$	$92.0_{0.0}$	$83.7_{0.0}$	$90.1_{0.0}$	$86.4_{0.0}$

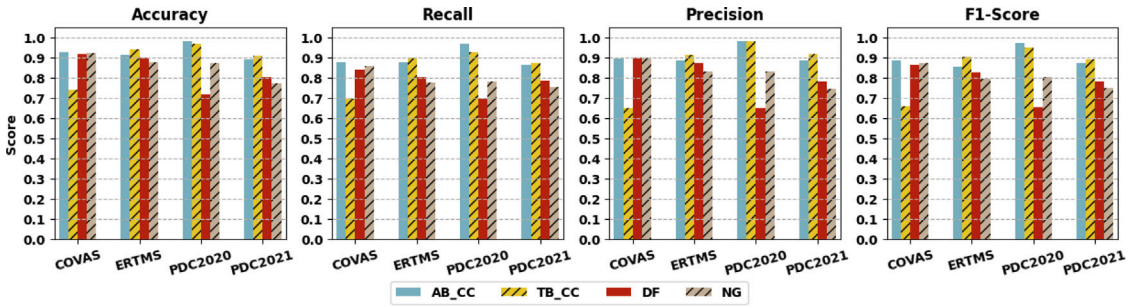


Fig. 9. The mean A, R, P and $F1$ per dataset of the best-performing dimensionality reduction technique for each process mining-based feature extraction approach.

Table 7
The mean A, R, P and $F1$ measures and their standard deviation in subscripts per dataset (DS) for each process mining-based feature extraction (PF) approach. Measures in bold indicate the highest figure per dataset. At the bottom is the significance of the DS and PF factors per metric X, where p_X indicates the p-value of the Friedman test ($p_X < 0.05$ means the factor is significant with 95% confidence) and I_X the percentage of variance in X explained by the factor.

DS	PF															
	AB_CC				TB_CC				DF				NG			
	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)	A(%)	R(%)	P(%)	F1(%)
PDC 2020	**$98.1_{0.3}$**	**$96.6_{1.3}$**	**$98.1_{0.5}$**	**$97.3_{0.5}$**	$96.6_{0.4}$	$92.9_{1.4}$	$97.9_{0.2}$	$95.0_{0.7}$	$71.6_{2.2}$	$69.3_{0.0}$	$64.8_{0.0}$	$65.4_{0.0}$	$87.2_{0.0}$	$78.2_{0.0}$	$83.3_{0.0}$	$80.3_{0.0}$
PDC 2021	$88.9_{1.1}$	$86.2_{1.3}$	$88.4_{1.4}$	$87.2_{1.3}$	**$90.6_{0.9}$**	**$87.1_{1.3}$**	**$91.7_{0.8}$**	**$88.9_{1.2}$**	$80.4_{1.7}$	$78.6_{1.2}$	$78.3_{1.9}$	$78.2_{1.3}$	$77.0_{0.0}$	$75.3_{0.0}$	$74.3_{0.0}$	$74.7_{0.0}$
ERTMS	$91.2_{1.9}$	$83.3_{5.8}$	$88.3_{1.6}$	$85.2_{4.1}$	**$94.0_{0.5}$**	**$89.8_{2.7}$**	**$91.5_{1.3}$**	**$90.5_{1.0}$**	$89.7_{1.3}$	$80.2_{2.4}$	$86.0_{2.6}$	$82.6_{2.4}$	$87.6_{1.1}$	$77.8_{3.7}$	$81.6_{1.5}$	$79.3_{2.6}$
COVAS	**$92.8_{1.0}$**	**$87.9_{3.2}$**	**$89.5_{1.1}$**	**$88.5_{2.0}$**	$73.8_{3.1}$	$69.7_{2.8}$	$64.8_{2.3}$	$65.7_{2.7}$	$91.9_{0.0}$	$83.5_{1.2}$	$90.1_{1.8}$	$86.2_{1.3}$	$92.2_{2.1}$	$86.0_{5.6}$	$89.1_{2.5}$	$87.2_{4.0}$

DS significance and importance: $p_A = 0.00$, $I_A = 8.7\%$; $p_R = 0.13$, $I_R = 2.0\%$; $p_P = 0.05$, $I_P = 3.2\%$; $p_{F1} = 0.02$, $I_{F1} = 2.6\%$
PF significance and importance: $p_A = 0.00$, $I_A = 20.0\%$; $p_R = 0.00$, $I_R = 34.3\%$; $p_P = 0.00$, $I_P = 17.8\%$; $p_{F1} = 0.00$, $I_{F1} = 25.7\%$
DS and PF significance and importance: $p_A = 0.00$, $I_A = 68.7\%$; $p_R = 0.00$, $I_R = 55.2\%$; $p_P = 0.00$, $I_P = 76.8\%$; $p_{F1} = 0.00$, $I_{F1} = 66.0\%$

Analysis of variance. We conduct an analysis of variance to evaluate whether the dataset, the process mining-based feature extraction approach, and the two combined result in statistically significant differences. To this aim, we design a two-factor full factorial experiment with factors PF and DS, of which the latter is the dataset factor. In like manner as Fig. 9, for each PF value, we take the best-performing dimensionality reduction technique for the given dataset. Table 7 reports the mean values of the performance measures for each DS-PF value pair, specifying whether the DS factor, PF factor, and DS and PF factors combined are significant for the target metric using the non-parametric Friedman test (a p-value less than 0.05 implies that the factor is significant with 95% confidence). In addition, the table shows how important the aforementioned factors are in terms of explained variance. Notably, the proposed AB_CC process mining-based feature extraction approach shows satisfying performance in all cases, e.g., $F1$ drops only as low as 85.2% for the ERTMS dataset and peaks at 97.3% for the PDC 2020 dataset. However, it is worth noting that AB_CC is outperformed by TB_CC for the ERTMS and PDC 2021 datasets. Therefore, it is safe to state that there is no one-size-fits-all process mining-based feature extraction approach valid for all datasets; besides, this is also confirmed statistically by the analysis of variance, which outlines that the interaction between the DS and PF factors is significant and important for all the measures considered.

6. Conclusions

In this paper, we took on the detection of control-flow anomalies in the business processes of organizations. To address the shortcomings of existing conformance checking-based techniques and support the development of competitive techniques for control-flow anomaly detection while maintaining the explainable nature of conformance checking, we proposed a novel process mining-based feature extraction approach with alignment-based conformance checking. This conformance checking variant aligns the deviating control flow with a reference process model; the resulting alignment can be inspected to extract additional statistics such as the number of times a given activity caused mismatches. Secondly, we integrated this approach into a flexible and explainable framework for developing techniques for control-flow anomaly detection. The framework combines process mining-based feature extraction and dimensionality reduction, which can handle high-dimensional feature sets, achieve detection effectiveness, and support explainability. In addition to our proposed process mining-based feature extraction approach, the framework allows employing other approaches, enabling flexible development of multiple conformance checking-based and conformance checking-independent techniques for control-flow anomaly detection. Using a set of synthetic and real-world datasets, we experimented with several conformance checking-based and conformance checking-independent framework techniques, and compared the results with baseline conformance checking-based techniques. The results showed that the conformance checking-based framework techniques using our approach outperform the baseline conformance checking-based techniques while maintaining the explainable nature of conformance checking. In particular, the best-performing framework techniques achieved 97.3%, 88.9%, 90.5% and 88.5% $F1$ for the PDC 2020, PDC 2021, ERTMS and COVAS datasets, whereas the baseline conformance checking-based techniques could achieve up to 36.1%, 56.6%, 78.8%, and 79.1%. We also provide an explanation of why existing conformance checking-based techniques may be ineffective. Finally, the results indicated that detection effectiveness is not solely dependent on the specific framework techniques used, as no one-size-fits-all process mining-based feature extraction approach is suitable for all the datasets. In fact, analysis of variance outlined that both the dataset and process mining-based feature extraction approach show statistically significant differences and high importance in $F1$ values.

Future work will extend the capabilities of the framework by considering additional event log attributes including information related to other perspectives as well as the control-flow one [5]. Such information can be used in process mining-based feature extraction to enrich the tabular data, address other types of anomalies, and result in improved detection effectiveness. The extension will also include object-centric process mining, which allows for focusing on the activities of the different entities involved in a process instead of flattening the event data and treating all objects uniformly [56].

CRediT authorship contribution statement

Francesco Vitale: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Marco Pegoraro:** Writing – review & editing, Validation, Formal analysis, Conceptualization. **Wil M.P. van der Aalst:** Writing – review & editing, Supervision, Formal analysis, Conceptualization. **Nicola Mazzocca:** Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been partially supported by the Spoke 9 "Digital Society & Smart Cities" of ICSC - Centro Nazionale di Ricerca in High Performance-Computing, Big Data and Quantum Computing, funded by the European Union - NextGenerationEU (PNRR-HPC, CUP: E63C22000980007).

Data availability

The non-sensitive data are provided in the GitHub repository of the research project.

References

[1] N. Martin, D.A. Fischer, G.D. Kerpedzhiev, K. Goel, S.J.J. Leemans, M. Roglinger, W.M.P. van der Aalst, M. Dumas, M. La Rosa, M.T. Wynn, Opportunities and challenges for process mining in organizations: Results of a delphi study, Bus. Inf. Syst. Eng. 63 (5) (2021) 511–527, http://dx.doi.org/10.1007/s12599-021-00720-0.

[2] I. Beerepoot, C. Di Ciccio, H.A. Reijers, S. Rinderle-Ma, W. Bandara, A. Burattin, D. Calvanese, T. Chen, I. Cohen, B. Depaire, G. Di Federico, M. Dumas, C. van Dun, T. Fehrer, D.A. Fischer, A. Gal, M. Indulska, V. Isahagian, C. Klinkmüller, W. Kratsch, H. Leopold, A. Van Looy, H. Lopez, S. Lukumbuzya, J. Mendling, L. Meyers, L. Moder, M. Montali, V. Muthusamy, M. Reichert, Y. Rizk, M. Rosemann, M. Röglinger, S. Sadiq, R. Seiger, T. Slaats, M. Simkus, I.A. Someh, B. Weber, I. Weber, M. Weske, F. Zerbato, The biggest business process management problems to solve before we die, Comput. Ind. 146 (2023) 103837, http://dx.doi.org/10.1016/j.compind.2022.103837.

[3] J.N. Adams, C. Pitsch, T. Brockhoff, W.M.P. van der Aalst, An experimental evaluation of process concept drift detection, Proc. VLDB Endow. 16 (8) (2023) 1856–1869, http://dx.doi.org/10.14778/3594512.3594517.

[4] J. Ko, M. Comuzzi, A systematic review of anomaly detection for business process event logs, Bus. Inf. Syst. Eng. 65 (2023) 441–462, http://dx.doi.org/10.1007/s12599-023-00794-y.

[5] W.M.P. van der Aalst, J. Carmona, Process Mining Handbook, Springer, Cham, Switzerland, 2022, http://dx.doi.org/10.1007/978-3-031-08848-3.

[6] F. Bezerra, J. Wainer, W.M.P. van der Aalst, Anomaly detection using process mining, in: T. Halpin, J. Krogstie, S. Nurcan, E. Proper, R. Schmidt, P. Soffer, R. Ukor (Eds.), Enterprise, Business-Process and Information Systems Modeling, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 149–161, http://dx.doi.org/10.1007/978-3-642-01862-6_13.

[7] F.d. Bezerra, J. Wainer, Algorithms for anomaly detection of traces in logs of process aware information systems, Inf. Syst. 38 (2013) 33–44, http://dx.doi.org/10.1016/j.is.2012.04.004.

[8] D. Myers, S. Suriadi, K. Radke, E. Foo, Anomaly detection for industrial control systems using process mining, Comput. Secur. 78 (2018) 103–125, http://dx.doi.org/10.1016/j.cose.2018.06.002.

[9] A. Pecchia, I. Weber, M. Cinque, Y. Ma, Discovering process models for the analysis of application failures under uncertainty of event logs, Knowl.-Based Syst. 189 (2020) 105054, http://dx.doi.org/10.1016/j.knosys.2019.105054.

[10] G.M. Tavares, R.S. Oyamada, S. Barbon, P. Ceravolo, Trace encoding in process mining: A survey and benchmarking, Eng. Appl. Artif. Intell. 126 (2023) 107028, http://dx.doi.org/10.1016/j.engappai.2023.107028.

[11] A. Rawal, J. McCoy, D.B. Rawat, B.M. Sadler, R.S. Amant, Recent advances in trustworthy explainable artificial intelligence: Status, challenges, and perspectives, IEEE Trans. Artif. Intell. 3 (2022) 852–866, http://dx.doi.org/10.1109/TAI.2021.3133846.

[12] Z. Li, Y. Zhu, M. Van Leeuwen, A survey on explainable anomaly detection, ACM Trans. Knowl. Discov. Data 18 (2023) http://dx.doi.org/10.1016/j.knosys.2019.105054.

[13] W.M.P. van der Aalst, Process Mining: Data Science in Action, Second ed., Springer, Berlin, Heidelberg, 2016, http://dx.doi.org/10.1007/978-3-662-49851-4.

[14] P. Singh, M. Saman Azari, F. Vitale, F. Flammini, N. Mazzocca, M. Caporuscio, J. Thornadtsson, Using log analytics and process mining to enable self-healing in the internet of things, Environ. Syst. Decis. 42 (2022) 1–17, http://dx.doi.org/10.1007/s10669-022-09859-x.

[15] A. De Benedictis, F. Flammini, N. Mazzocca, A. Somma, F. Vitale, Digital twins for anomaly detection in the industrial internet of things: Conceptual architecture and proof-of-concept, IEEE Trans. Ind. Inform. (2023) 1–11, http://dx.doi.org/10.1109/TII.2023.3246983.

[16] T. Nolle, S. Luettgen, A. Seeliger, M. Mühlhäuser, Analyzing business process anomalies using autoencoders, Mach. Learn. 107 (2018) 1875–1893, http://dx.doi.org/10.1007/s10994-018-5702-8.

[17] D. Vasumathi, M. Vijayakamal, Unsupervised learning methods for anomaly detection and log quality improvement using process event log, Int. J. Adv. Sci. Technol. 29 (2020) 1109–1125, URL http://sersc.org/journals/index.php/IJAST/article/view/3603.

[18] E.A. Elaziz, R. Fathalla, M. Shaheen, Deep reinforcement learning for data-efficient weakly supervised business process anomaly detection, Big Data 10 (2023) 1–35, http://dx.doi.org/10.1186/s40537-023-00708-5.

[19] V. Chinnaiah, V. Veerabhadram, R. Aavula, S. Aluvala, PMiner: Process mining using deep autoencoder for anomaly detection and reconstruction of business processes, Int. J. Electr. Comput. Eng. Syst. 15 (2024) 531–542, http://dx.doi.org/10.32985/ijeces.15.6.7.

[20] D. Kan, X. Fang, Event log anomaly detection method based on auto-encoder and control flow, Multimedia Syst. 30 (2024) 1–15, http://dx.doi.org/10.1007/s00530-023-01199-3.

[21] M. Du, F. Li, G. Zheng, V. Srikumar, DeepLog: Anomaly detection and diagnosis from system logs through deep learning, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Association for Computing Machinery, New York, NY, USA, 2017, pp. 1285–1298, http://dx.doi.org/10.1145/3133956.3134015.

[22] J. Lahann, P. Pfeiffer, P. Fettke, LSTM-based anomaly detection of process instances: Benchmark and tweaks, in: M. Montali, A. Senderovich, M. Weidlich (Eds.), Process Mining Workshops, Springer Nature, Switzerland, Cham, 2023, pp. 229–241, http://dx.doi.org/10.1007/978-3-031-27815-0_17.

[23] T. Nolle, S. Luettgen, A. Seeliger, M. Mühlhäuser, BINet: Multi-perspective business process anomaly classification, Inf. Syst. 103 (2022) 101458, http://dx.doi.org/10.1016/j.is.2019.101458.

[24] W. Guan, J. Cao, Y. Gu, S. Qian, GRASPED: A GRU-AE network based multi-perspective business process anomaly detection model, IEEE Trans. Serv. Comput. 16 (2023) 3412–3424, http://dx.doi.org/10.1109/TSC.2023.3262405.

[25] L.P. Yuan, P. Liu, S. Zhu, Recompose event sequences vs. Predict next events: A novel anomaly detection approach for discrete event logs, in: Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, 2021, pp. 336–348, http://dx.doi.org/10.1145/3433210.3453098.

[26] X. Wang, L. Yang, D. Li, L. Ma, Y. He, J. Xiao, J. Liu, Y. Yang, MADDC: Multi-scale anomaly detection, diagnosis and correction for discrete event logs, in: Proceedings of the 38th Annual Computer Security Applications Conference, 2022, pp. 769–784, http://dx.doi.org/10.1145/3564625.3567972.

[27] W. Guan, J. Cao, H. Zhao, Y. Gu, S. Qian, WAKE: A weakly supervised business process anomaly detection framework via a pre-trained autoencoder, IEEE Trans. Knowl. Data Eng. 36 (6) (2024) 2745–2758, http://dx.doi.org/10.1109/TKDE.2023.3322411.

[28] G.M. Tavares, S.B. Junior, Process mining encoding via meta-learning for an enhanced anomaly detection, in: New Trends in Database and Information Systems, 2021, pp. 157–168, http://dx.doi.org/10.1007/978-3-030-85082-1_15.

[29] T. Nolle, A. Seeliger, N. Thoma, M. Mühlhäuser, DeepAlign: Alignment-based process anomaly correction using recurrent neural networks, in: Advanced Information Systems Engineering, Springer International Publishing, Cham, 2020, pp. 319–333, http://dx.doi.org/10.1007/978-3-030-49435-3_20.

[30] M. Boltenhagen, B. Chetioui, L. Huber, An alignment cost-based classification of log traces using machine-learning, in: S. Leemans, H. Leopold (Eds.), Process Mining Workshops, Springer International Publishing, Cham, 2021, pp. 136–148, http://dx.doi.org/10.1007/978-3-030-72693-5_11.

[31] P. Krajsic, B. Franczyk, Variational autoencoder for anomaly detection in event data in online process mining, in: Proceedings of the 23rd International Conference on Enterprise Information Systems - Volume 1: ICEIS, 2021, pp. 567–574, http://dx.doi.org/10.5220/0010375905670574.

[32] G. Li, W.M.P. van der Aalst, A framework for detecting deviations in complex event logs, Intell. Data Anal. 21 (2017) 759–779, http://dx.doi.org/10.3233/IDA-160044.

[33] J. Ko, M. Comuzzi, Detecting anomalies in business process event logs using statistical leverage, Inform. Sci. 549 (2021) 53–67, http://dx.doi.org/10.1016/j.ins.2020.11.017.

[34] S. Luftensteiner, P. Praher, Log file anomaly detection based on process mining graphs, in: G. Kotsis, A.M. Tjoa, I. Khalil, B. Moser, A. Taudes, A. Mashkoor, J. Sametinger, J. Martinez-Gil, F. Sobieczky, L. Fischer, R. Ramler, M. Khan, G. Czech (Eds.), Database and Expert Systems Applications - DEXA 2022 Workshops, Springer International Publishing, Cham, 2022, pp. 383–391, http://dx.doi.org/10.1007/978-3-031-14343-4_36.

[35] I. Mavroudopoulos, A. Gounaris, A comparison of proximity-based methods for detecting temporal anomalies in business processes, Mach. Learn. 112 (2022) 1–28, http://dx.doi.org/10.1007/s10994-022-06152-5.

[36] R. Accorsi, T. Stocker, On the exploitation of process mining for security audits: The conformance checking case, in: Proceedings of the 27th Annual ACM Symposium on Applied Computing, Association for Computing Machinery, New York, NY, USA, 2012, pp. 1709–1716, http://dx.doi.org/10.1145/2245276.2232051.

[37] R. Sarno, F. Sinaga, K.R. Sungkono, Anomaly detection in business processes using process mining and fuzzy association rule learning, J. Big Data 7 (2021) http://dx.doi.org/10.1186/s40537-019-0277-1.

[38] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. 41 (3) (2009) http://dx.doi.org/10.1145/1541880.1541882.

[39] C. Zhang, Q. Wang, T. Liu, X. Lu, J. Hong, B. Han, C. Gong, Fraud detection under multi-sourced extremely noisy annotations, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 2497–2506, http://dx.doi.org/10.1145/3459637.3482433.

[40] Y. Zhao, G. Zheng, S. Mukherjee, R. McCann, A. Awadallah, ADMoE: Anomaly detection with mixture-of-experts from noisy labels, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, 2023, http://dx.doi.org/10.1609/aaai.v37i4.25620, URL https://ojs.aaai.org/index.php/AAAI/article/view/25620.

[41] Y. Zhou, X. Song, Y. Zhang, F. Liu, C. Zhu, L. Liu, Feature encoding with autoencoders for weakly supervised anomaly detection, IEEE Trans. Neural Netw. Learn. Syst. 33 (2022) 2454–2465, http://dx.doi.org/10.1109/TNNLS.2021.3086137.

[42] M. Jiang, C. Hou, A. Zheng, X. Hu, S. Han, H. Huang, X. He, P.S. Yu, Y. Zhao, Weakly supervised anomaly detection: A survey, 2023, URL https://arxiv.org/abs/2302.04549, arXiv:2302.04549.

[43] H. Abdi, L.J. Williams, Principal component analysis, Wiley Interdiscip. Rev. Comput. Stat. 2 (2010) 433–459, http://dx.doi.org/10.1002/wics.101.

[44] H. Zou, T. Hastie, R. Tibshirani, Sparse principal component analysis, J. Comput. Graph. Statist. 15 (2) (2006) 265–286, http://dx.doi.org/10.1198/106186006X113430.

[45] V.H. Nguyen, J.C. Golinval, Fault detection based on kernel principal component analysis, Eng. Struct. 32 (11) (2010) 3683–3691, http://dx.doi.org/10.1016/j.engstruct.2010.08.012.

[46] M. Sakurada, T. Yairi, Anomaly detection using autoencoders with nonlinear dimensionality reduction, in: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, MLSDA '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 4–11, http://dx.doi.org/10.1145/2689746.2689747.

[47] A. Tharwat, T. Gaber, A. Ibrahim, A.E. Hassanien, Linear discriminant analysis: A detailed tutorial, AI Commun. 30 (2017) 169–190, http://dx.doi.org/10.3233/AIC-170729.

[48] A.C. Belkina, C.O. Ciccolella, R. Anno, R. Halpert, J. Spidlen, J.E. Snyder-Cappione, Automated optimized parameters for T-distributed stochastic neighbor embedding improve visualization and analysis of large datasets, Nat. Commun. 10 (2019) http://dx.doi.org/10.1038/s41467-019-13055-y.

[49] T. Bouwmans, E.H. Zahzah, Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance, Comput. Vis. Image Underst. 122 (2014) 22–34, http://dx.doi.org/10.1016/j.cviu.2013.11.009.

[50] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, in: Conference Track Proceedings of 2nd International Conference on Learning Representations, 2014, http://dx.doi.org/10.48550/arXiv.1312.6114.

[51] S.M. Lundberg, S.I. Lee, A unified approach to interpreting model predictions, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 4768–4777, URL https://dl.acm.org/doi/10.5555/3295222.3295230.

[52] N. Takeishi, Shapley values of reconstruction errors of PCA for explaining anomaly detection, in: 2019 International Conference on Data Mining Workshops, ICDMW, 2019, pp. 793–798, http://dx.doi.org/10.1109/ICDMW.2019.00117.

[53] L. Antwarg, R.M. Miller, B. Shapira, L. Rokach, Explaining anomalies detected by autoencoders using Shapley additive explanations, Expert Syst. Appl. 186 (2021) 115736, http://dx.doi.org/10.1016/j.eswa.2021.115736.

[54] M. Pegoraro, M.B.S. Narayana, E. Benevento, W.M. van der Aalst, L. Martin, G. Marx, Analyzing medical data with process mining: A covid-19 case study, in: International Conference on Business Information Systems, Springer, 2021, pp. 39–44, http://dx.doi.org/10.1007/978-3-031-04216-4_4.

[55] E. Benevento, M. Pegoraro, M. Antoniazzi, H.H. Beyel, V. Peeva, P. Balfanz, W.M.P. van der Aalst, L. Martin, G. Marx, Process modeling and conformance checking in healthcare: A COVID-19 case study, in: M. Montali, A. Senderovich, M. Weidlich (Eds.), Process Mining Workshops, Springer Nature, Switzerland, Cham, 2023, pp. 315–327, http://dx.doi.org/10.1007/978-3-031-27815-0_23.

[56] W.M.P. van der Aalst, Object-centric process mining: Unraveling the fabric of real processes, Mathematics 11 (12) (2023) http://dx.doi.org/10.3390/math11122691.