Bus Inf Syst Eng
https://doi.org/10.1007/s12599-025-00935-5

RESEARCH PAPER

Case ID Revealed HERE: Hybrid Elusive Case Repair Method for Transformer-Driven Business Process Event Log Enhancement

Felix Zetzsche · Robert Andrews · Arthur H. M. ter Hofstede · Maximilian Röglinger · Sebastian Johannes Schmid · Moe Thandar Wynn

Received: 22 July 2024 / Revised: 14 December 2024 / Accepted: 23 January 2025

Abstract Process mining is a data-driven technique that leverages event logs to analyze, visualize, and improve business processes. However, data quality is often low in real-world settings due to various event log imperfections, which, in turn, degrade the accuracy and reliability of process mining insights. One notable example is the elusive case imperfection pattern, describing the absence of case identifiers responsible for linking events to a specific process instance. Elusive cases are particularly problematic, as process mining techniques rely heavily on the accurate mapping of events to instances to provide meaningful and actionable insights into business processes. To address this issue, the study follows the Design Science Research paradigm to iteratively develop a method for repairing the elusive case imperfection pattern in event logs. The proposed Hybrid Elusive Case Repair Method (*HERE*) combines a traditional, rule-based approach with generative artificial intelligence, specifically the Transformer architecture. By integrating domain knowledge, *HERE* constitutes a comprehensive human-in-the-loop approach, enhancing its ability to accurately repair elusive cases in event logs. The method is evaluated by instantiating it as a software prototype, applying it to repair three publicly accessible event logs, and seeking expert feedback in a total of 21 interviews conducted at different points during the design and development phase. The results demonstrate that *HERE* makes significant progress in addressing the elusive case imperfection pattern, particularly when provided with sufficient data volume, laying the groundwork for resolving further data quality issues in process mining.

Keywords Process mining · Event log quality · Event log repair · Generative artificial intelligence · Transformer · Business process management

1 Introduction

Process mining analyzes event log data from information systems to extract meaningful insights into business processes (Rinderle-Ma et al. 2023). By examining historical data on process executions, process mining uncovers inefficiencies, identifies bottlenecks, detects deviations from standard workflows, and ensures regulatory compliance (van der Aalst 2022). These applications are in turn associated with numerous economic benefits, including cost reduction, higher process efficiency, and data-driven strategic decision-making (Badakhshan et al. 2022; Galic and Wolf 2021; Grisold et al. 2021).

As a data-driven technology, the success of process mining initiatives is highly dependent on the quality of the input data. Poor-quality data can lead to inaccurate outcomes, a phenomenon known as garbage-in, garbage-out (Beerepoot et al. 2023). For instance, many process mining techniques, such as those focused on process discovery or conformance checking, rely heavily on mapping events to specific process instances using high-quality case identifiers (IDs) (van der Aalst 2022). This mapping, also known

Accepted after 1 revision by the editors of the Special Issue.

F. Zetzsche (✉) · M. Röglinger · S. J. Schmid
FIM Research Center for Information Management, University of Bayreuth, Branch Business and Information Systems Engineering of the Fraunhofer FIT, Wittelsbacherring 10, 95444 Bayreuth, Germany
e-mail: felix.zetzsche@fit.fraunhofer.de

R. Andrews · A. H. M. ter Hofstede · M. T. Wynn
Queensland University of Technology, 2 George St, Brisbane City, QLD 4000, Australia

Published online: 19 March 2025

as the primary correlation problem, is essential for reconstructing the sequence of activities within each process instance, allowing for accurate discovery and analysis (van der Aalst 2016b). However, real-life event logs are often subject to erroneous case IDs, leading to challenges in process mining analyses (Fischer et al. 2022). One notable example is the *elusive case* imperfection pattern, describing a common problem (van der Aalst et al. 2012) in which events are not linked to their corresponding case IDs (Suriadi et al. 2017). This issue can significantly hinder the efficacy of process mining endeavors, potentially making them impractical or resulting in erroneous insights, as case IDs are essential for capturing the relationships between individual events (Tajima et al. 2023). Consequently, organizations affected by this imperfection pattern may fail to realize the full potential of process mining (Suriadi et al. 2017).

Several repair methods have been developed to address the elusive case imperfection pattern, which involve reconstructing missing case IDs or establishing accurate associations between events and process instances. For instance, Martin et al. (2022) propose a query language to detect events affected by elusive cases. Bayomie et al. (2023) introduce a probabilistic optimization method for grouping events into cases, while Pegoraro et al. (2022) apply machine learning for the purpose of log segmentation. Additionally, De Fazio et al. (2024) suggest heuristics for detecting case IDs. Despite these advancements, current methods encounter challenges, as they either rely on supplementary well-defined data alongside the event log or do not fully leverage the potential of the existing data. Moreover, these approaches are often restricted to regrouping all events, making it impossible to assign individual events to existing groups of events with error-free case IDs. This limitation becomes particularly problematic when only a small percentage of events are affected, while the majority remain correct. Manual repair of elusive cases, although theoretically feasible, is impractical due to the massive volume of data, resulting in substantial resource and cost constraints.

Generative artificial intelligence (AI) offers a promising alternative to tackle these challenges. Its ability to understand complex data patterns and reconstruct missing or erroneous data makes it well-suited for this task (Hofmann et al. 2021). Moreover, generative AI excels at capturing intricate, long-term data patterns within event logs, an area where traditional machine learning algorithms often fall short (Banh and Strobel 2023). Accordingly, the capabilities of generative AI have been showcased in various subdisciplines of process mining research. For instance, within predictive business process monitoring, Transformer architectures are utilized to forecast subsequent activities in an ongoing process instance (Bukhsh et al. 2021). For process discovery, generative AI enables the extraction of processes from textual data (Busch et al. 2023). Using generative adversarial networks (GANs), van Dun et al. (2023) demonstrate that generative AI can facilitate the generation of ideas for business process improvements. In the domain of event log quality enhancement, GANs are utilized to tackle timestamp-related data quality issues in event logs (Schmid et al. 2023), while a Transformer model has been applied to address activity-related quality issues (Wu et al. 2024). Additionally, Nguyen et al. (2019) employed an autoencoder to reconstruct missing activity and timestamp values. Beyond these successful applications, vendors and market-research organizations assume that generative AI has the potential to significantly streamline data preparation tasks (Kerremans and Kerremans 2023; Reinkemeyer et al. 2023), which currently account for approximately 61% to 80% of the efforts involved in applying process mining (Wynn et al. 2022). Hence, we conclude that generative AI has the potential to repair event logs that are subject to the elusive case imperfection pattern. Consequently, this research addresses the following research question: *How can generative AI be used to repair the elusive case imperfection pattern?*

To answer this question, we follow the Design Science Research (DSR) paradigm proposed by Peffers et al. (2007) and introduce *HERE*, the **H**ybrid **E**lusive Case **RE**pair Method, which aims to reestablish the link between events and their corresponding case IDs. Given the potential for inaccuracies in outputs generated by generative AI models (Feuerriegel et al. 2024), we establish rule-based methods by integrating human interactions into the model's architecture through domain knowledge, thus enabling a human-in-the-loop approach (Mosqueira-Rey et al. 2023). We instantiate *HERE* as an open-source software prototype and evaluate it following the framework for evaluation in DSR (FEDS) as proposed by Venable et al. (2016). By doing so, we first refine our design specification with the help of 11 interviews with practitioners and researchers. Afterwards, we provide a proof of concept of *HERE* by instantiating it as a real-world prototype and using it to repair a total of three different event logs with nine degrees of elusiveness each, hence demonstrating feasibility and effectiveness. Lastly, a proof of value with real users and tasks is given by letting ten researchers and practitioners use the research prototype to repair an event log in a simulated environment.

Our primary contribution is the development of a novel method to address elusive cases in process mining. We provide design knowledge on adapting existing architectures for effective event log repair. This work builds on prior research at the intersection of generative AI and event log quality improvement, advancing the field of process data quality management. Additionally, we provide an

open-source software tool that enables both researchers and practitioners to repair elusive cases.

This paper is structured as follows: In Sect. 2, we discuss previous work in the field. Section 3 provides a detailed account of our activities within the DSR paradigm. Section 4 presents our artifact *HERE*, while Sect. 5 discusses its evaluation within FEDS. In Sect. 6, the results are discussed. Lastly, Sect. 7 provides a summary of our findings.

2 Theoretical Background

2.1 Process Mining and Event Log Quality

Process mining focuses on optimizing business processes by systematically examining event data (van der Aalst 2022). The objectives of process mining may be backward- (e.g., finding causes of past bottlenecks) or forward-looking (e.g., making predictions for ongoing process executions) (van der Aalst 2022). The process mining discipline encompasses various activities. Process discovery focuses on deriving process models from event data, whereas conformance checking seeks to detect discrepancies between the event data and the process model (van der Aalst 2016c). Additionally, enhancement involves refining an existing process model using event data (van der Aalst 2016c).

Typically stored as an extensible event stream (XES) file, process data is often represented in event logs (van der Aalst 2016b). Within an event log, various process instances known as cases are recorded where distinct events are associated with it (van der Aalst 2016b). The definition and boundaries of a case vary depending on the context (van der Aalst 2016a). An event log must typically contain three essential attributes: case ID, activity name, and timestamp (De Weerdt and Wynn 2022). In addition to these essential attributes, event logs may also include supplementary attributes such as the executing resource or the associated costs (van der Aalst 2022). Thus, event logs may contain both discrete and continuous attributes.

Employing data quality metrics allows to assess the quality of an event log. Such metrics may address different aspects of data quality and are categorized into dimensions such as accuracy, completeness, redundancy, readability, accessibility, consistency, usefulness, and trust (Batini and Scannapieco 2016). Among these metrics, accuracy, completeness, and consistency are particularly important for event log repair. Accuracy is generally defined as the closeness between a representation and the actual data value. Completeness refers to the extent to which data is not missing and is sufficiently comprehensive for the task at hand. Consistency can be assessed by verifying adherence to integrity constraints, which are properties that all instances must satisfy (Batini and Scannapieco 2016).

Event log quality issues can stem from various root causes, such as manual data entry errors, system design flaws, or problems encountered during the extraction of event logs (Andrews et al. 2022). To shed light on specific quality problems in event logs, Suriadi et al. (2017) classified them into 11 event log imperfection patterns. Thereby, the elusive case describes scenarios where events are not explicitly linked to their corresponding case ID. This absence of a case ID poses a significant challenge, rendering process mining analyses infeasible (Suriadi et al. 2017).

Several methods have been proposed to address imperfection patterns in event data, each with distinct objectives, including detection and repair. Detection approaches aim to identify existing errors in event data. For example, Andrews et al. (2018) introduced a log query language capable of identifying five imperfection patterns. Expanding on this work, Martin et al. (2022) proposed an approach based on activity logs, which extends detection to a total of ten imperfection patterns and tackles additional event log quality issues not covered previously. Furthermore, Sadeghianasl et al. (2019) presented a detection method that takes into account the context of activities. Lastly, approaches focused on quantifying quality metrics have been proposed (Fischer et al. 2022).

Repair approaches aim to improve the quality of event logs by rectifying errors. For instance, gamified crowdsourcing techniques have been utilized to enhance activity labels (Sadeghianasl et al. 2020, 2024). Such approaches, involving human-in-the-loop interactions and integrating domain knowledge, hold the potential to improve overall data quality (Chen et al. 2020). Moreover, hybrid methodologies offer diverse advantages (Raisch and Fomina 2024). For instance, the integration of domain knowledge can be leveraged to establish declarative integrity constraints, which, in turn, facilitate the development of declarative process models for process specification (Di Ciccio and Montali 2022; Pesic et al. 2007). Additionally, there exist repair approaches based on generative AI such as GANs or variational autoencoders (Nguyen et al. 2019; Schmid et al. 2023).

Various methods are employed to determine case IDs (Ferreira and Gillblad 2009; Pourmirza et al. 2017). Decision tree methods, for instance, are applied with behavioral profiles and statistical heuristics to identify case IDs. However, they encounter difficulties in complex or overlapping cases due to ambiguities and missing information (Bayomie et al. 2016a, b). Optimization-based approaches align event logs with detailed process models but depend heavily on specific domain knowledge, which limits their adaptability in dynamic or incomplete process

environments (Bayomie et al. 2019, 2022). Domain knowledge-driven techniques, such as the one by De Fazio et al. (2024), leverage expert-defined heuristics for event similarity. This method demands significant input from experts, making it resource-intensive and challenging to scale, particularly with large or rapidly changing datasets. Similarly, Bayomie et al. (2023) apply domain-specific rules within an optimization framework, but the reliance on predefined rules restricts the method's applicability across diverse processes. Expert-driven approaches like the one proposed by Burattin and Vigo (2011) involve manual case review, which enhances accuracy but introduces subjectivity and potential inconsistency. Machine learning techniques, such as the neural network approach by Pegoraro et al. (2022), automate case ID assignment. These methods require extensive labeled data and may lack interpretability. Overall, these limitations underscore the need for a more flexible and scalable approach to case ID determination. An effective solution would minimize dependence on domain knowledge, handle complex cases without expert intervention, and ensure explainability of machine-learning-driven processes.

2.2 Generative Artificial Intelligence in Process Mining

Machine learning algorithms have proven to be effective in handling incomplete event data (Weinzierl et al. 2024). As a subset of AI, machine learning encompasses algorithms that learn to perform tasks by processing data, rather than relying solely on explicit programming instructions (Banh and Strobel 2023). A commonly used approach within machine learning is supervised learning, where a model is trained using a dataset that includes both input data and the corresponding correct outputs (Janiesch et al. 2021). Within the broader field of machine learning, more specialized techniques exist. Deep learning involves the use of multi-layer neural networks to model complex data patterns. These networks learn hierarchical representations of data, allowing the model to identify increasingly abstract patterns as it processes more layers of information. Generative AI as a subclass of deep learning, models complex data distributions to generate new samples that closely mirror the structure, patterns, and characteristics of the training data. Discriminative AI, on the other hand, focuses on modeling the boundary between different classes in the data, rather than the data distribution itself (Banh and Strobel 2023). Prominent architectures of generative AI include GANs, Transformers, and variational autoencoders (Feuerriegel et al. 2024).

In recent years, generative AI and Transformers, in particular, have gained significant popularity, driven by the success of large language models like ChatGPT (Feuerriegel et al. 2024). Transformers, as introduced by Vaswani et al. (2017), use a mechanism called self-attention to process input data all at once rather than sequentially. This mechanism allows the model to focus on different parts of the input when making predictions, capturing complex dependencies that might exist between various elements, such as activities in an event log or relationships between words in a sentence. This makes Transformers particularly effective for tasks involving sequential data with long-range dependencies, a common characteristic of event logs. Unlike traditional methods that struggle with such dependencies, Transformers excel at capturing these relationships, which makes them particularly useful in tasks involving complex, sequential data (Vaswani et al. 2017). The Transformer architecture is comprised of two main components: the encoder and the decoder (Vaswani et al. 2017). The encoder processes the input data, extracting key features, while the decoder uses this information to generate the output. Both the encoder and decoder are built from layers of self-attention mechanisms, allowing the model to refine its understanding of the data at each step. While the encoder and decoder work together during training, the decoder operates independently during inference, when the model is used to make predictions.

Machine learning, including generative AI, has been successfully applied in various process-related contexts (Weinzierl et al. 2024). For instance, in the domain of predictive business process monitoring, Bukhsh et al. (2021) employed a Transformer to predict the next activity in a trace, defined as a sequence of events within a given process. Similarly, Rivera Lazo and Ñanculef (2022) employed a Transformer for the same task, with their architecture allowing for multiple input attributes. Additionally, some approaches leverage GANs for this purpose (Hoffmann et al. 2022; Taymouri et al. 2020). Generative AI architectures have also been implemented for business process improvement. For instance, van Dun et al. (2023) used a GAN to generate improvement ideas, while Beheshti et al. (2023) proposed a Transformer to derive improvement recommendations. Furthermore, generative AI has been employed to enhance the quality of event logs. Nguyen et al. (2019) utilized an autoencoder architecture to reconstruct missing attribute values for activity and timestamp, whereas Schmid et al. (2023) addressed identical timestamp errors using a GAN. These applications demonstrate how generative AI models can be effectively adapted for solving challenges in business process management.

3 Research Method

The objective of this study is to develop an approach utilizing generative AI to repair the elusive case imperfection pattern, specifically focusing on determining case IDs for events that were previously lacking them. Elusive cases, among other event log quality issues, cause the results of process mining analyses to be misleading (Suriadi et al. 2017). Hence, this study addresses a significant business problem and thus falls under Type I machine learning research as per the categorization proposed by Padmanabhan et al. (2022). This category, which aims to resolve significant problems within economic and social contexts by developing machine learning methods (Padmanabhan et al. 2022), aligns closely with DSR, which in turn aims to design purposeful artifacts that solve real-world problem classes (Tuunanen et al. 2024). Moreover, DSR artifacts can specifically manifest as methods encompassing systematic procedures and techniques to solve a real-world business problem (Gregor and Hevner 2013; March and Smith 1995). Consequently, DSR is generally a suitable framework for our research endeavor. A fundamental principle of DSR is the iterative process of searching for an artifact that provides a satisfactory solution (Hevner et al. 2004; Tuunanen et al. 2024). Established DSR processes can facilitate this search process. In this study, we adopt the approach outlined by Peffers et al. (2007), which comprises six steps. Figure 1 illustrates these six steps and summarizes our activities in each step. In Sect. 1, we already (1) *identified and motivated the problem*, ensuring that the objective of our DSR project addresses a significant business problem. This paper aims to (6) *communicate* our findings.

3.1 Design Objectives

Based on our analysis of the research problem and existing literature in Sect. 2, we (2) *define design objectives* (DOs) aimed at integrating essential solution components (Peffers et al. 2007). A central challenge identified is the elusive case imperfection pattern, where events lack corresponding case IDs. Our primary emphasis lies in restoring these associations. Given the diverse nature of data attributes in event logs, a solution to the problem should encompass both discrete and continuous data types. Moreover, the incorporation of external sources of implicit knowledge is crucial for improving contextual understanding. Therefore, we summarize the first DO as follows:

DO 1. An approach to repairing the elusive case imperfection pattern should accommodate diverse input attribute types and integrate external knowledge beyond the event log.

Sometimes, historical process data may not fully reflect the current process reality, such as when control flows have changed over time or new dominant process variants are underrepresented in the data. In these cases, it is important to capture this new knowledge as an additional input. Since such information often cannot be inferred directly from the data, human expertise becomes essential. A human-in-the-loop approach facilitates this interaction, allowing humans to provide contextual insights that algorithms lack. Through a mutual exchange of information, humans and algorithms work together to refine the repair process, delivering the best possible outcomes. Therefore, the second DO is summarized as follows:

DO 2. An approach to repairing the elusive case imperfection pattern should incorporate external knowledge by means of a human-in-the-loop approach.

The quality of event logs is multidimensional, hence a solution should address several key quality metrics. First, accuracy ensures that the reconstructed process represents reality, maintaining close alignment between predicted and actual case IDs. Second, completeness ensures a sufficient number of events are mapped to case IDs. Third, consistency ensures adherence to known process patterns and rules during event reconstruction. Consequently, the third DO is summarized as follows:

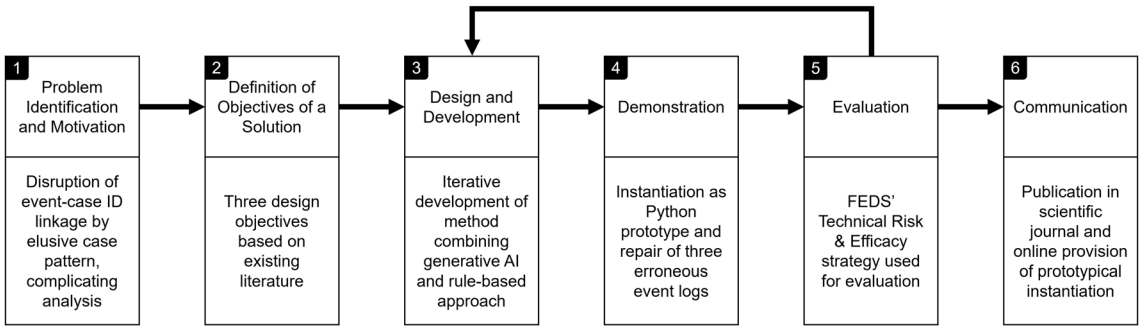

Fig. 1 Research activities within DSR

Table 1 Design iterations for developing *HERE*

Iteration	Design Activities	Evaluation Results	Key Insights
1	Reviewing literature on various generative AI architectures	Identified limitations in handling long-term dependencies and sequential data in some architectures	Self-attention mechanisms in Transformers effectively manage long-range dependencies
		Transformer found superior in performance and scalability	Transformer architecture is recommended
2	Establishing a preprocessing pipeline	Reasonable output is produced	Transformer can repair elusive cases
	Designing baseline Transformer architecture for case ID determination	Preprocessing pipeline accommodates multiple input attributes	
3	Integrating domain knowledge into the Transformer model	Rule-checking enables higher consistency and accuracy	Rule-based systems enhance Transformer's output with specific domain expertise
	Developing complementary rule-based approaches for sequence validation	Overall model performance increased with domain knowledge	
4	Conducting extensive hyperparameter tuning	Achieved optimized model performance and efficiency	Proper hyperparameter selection is critical for maximizing model effectiveness
		Significant reduction in prediction errors	

DO 3. An approach to repairing the elusive case imperfection pattern should satisfy multiple data quality dimensions, such as accuracy, completeness, and consistency.

These objectives guide the design and development phase towards achieving a robust solution that comprehensively repairs the elusive case.

3.2 Design and Development

The (3) *design and development* of our artifact follows an iterative process, as shown in Table 1, guided by the DOs and continuously refined by evaluation insights. During the initial iteration, our primary goal was to identify the most suitable generative AI architecture. After a comprehensive literature review, we selected the Transformer architecture due to its proven effectiveness in handling sequential data. This capability was crucial for our needs, as repairing the elusive case involves reconstructing case IDs for erroneous events based on other attributes in the event log. This process transforms the input from a sequence lacking case IDs to an output sequence where each event is accurately linked to the corresponding case ID. Therefore, this constitutes a sequence-to-sequence translation in machine learning, where input sequences are mapped to output sequences (Sutskever et al. 2014). In the subsequent iteration, our objective transitioned to achieving DO 1. We implemented architectural modifications, which enabled the determination of case IDs. In the third iteration, we addressed DO 2 to enhance the model's input by incorporating a human-in-the-loop and supplementing it with a rule-based approach grounded in domain knowledge. Finally, in the fourth iteration, we focused on DO 3, employing hyperparameter tuning to improve output quality.

3.3 Demonstration and Evaluation

To (4) *demonstrate* the feasibility of the design, it is instantiated as a Python software prototype[1]. Prototyping is a well-established method for evaluation in DSR (March and Storey 2008). We use our prototype to repair three publicly accessible event logs: two synthetic logs representing a journal paper review process (van der Aalst 2010) and the low-discrimination variant of a rental process (Pohl and Berti 2023), and one real-life log detailing a medical service billing process (Mannhardt 2017). Through these demonstrations, the prototype showcases the artifact's utility and suitability (Peffers et al. 2012).

The prototype itself also contributes to the (5) *evaluation* of our artifact. Our evaluation framework follows FEDS, a structure guiding evaluations in DSR projects (Venable et al. 2016). This framework categorizes evaluation strategies along two dimensions. The first dimension concerns the functional purpose of the evaluation episode, distinguishing between formative and summative evaluations. Formative evaluations aim to improve an artifact during its development phase through continuous feedback and iteration, whereas summative evaluations assess the

[1] Link to *HERE* instantiation source code: https://github.com/FlZtz/Hybrid_Elusive_Case_Repair_Method.

overall effectiveness and impact of the artifact after its development is completed. The second dimension relates to the evaluation paradigm, distinguishing between artificial and naturalistic approaches. Artificial evaluations are conducted in controlled, experimental settings where variables can be systematically manipulated and measured, while naturalistic evaluations occur in real-world environments, observing phenomena as they naturally unfold. One strategy within FEDS is the Technical Risk & Efficacy strategy, which assesses technical artifacts to mitigate uncertainty and risk while ensuring rigor. This strategy begins with an artificial formative evaluation, proceeds to an artificial summative evaluation, and concludes with a naturalistic summative evaluation.

For the artificial formative evaluation, we conducted 11 semi-structured expert interviews to validate our design specification in an artificial setting. Interviews are widely accepted evaluation methods in information systems research (Myers and Newman 2007). The panel comprised 11 experts selected through purposive sampling (Robinson 2014), representing both research and industry sectors (cf. Table 2). These interviewees were chosen for their expertise in process mining and business process management. Each interview averaged approximately 40 minutes.

In the artificial summative evaluation, we assess the effectiveness of the artifact. To do so, the software prototype is used to repair three publicly accessible event logs, comprising both synthetic and real-life data. For each log, we introduced errors at varying rates, ranging from 10% to 90% in 10% increments to simulate elusive cases. Furthermore, we conducted the repair routine with different configurations. Each event log and repair configuration is then evaluated using ten different metrics each addressing different aspects of repair quality such as accuracy, completeness, and consistency, as specified in DO 3.

Finally, we conducted a naturalistic summative evaluation through a second round of semi-structured interviews with ten experts from the initial panel. This approach had the advantage that the interviewees were already familiar with the design specification, enabling them to provide informed and detailed feedback on the prototype's functionality and its alignment with the objectives. Each interview lasted around 35 minutes on average. Interviewees were briefed on the results of our artificial summative evaluation and engaged in a simulated prototype interaction. Detailed information on the evaluation setup as well as the results are presented in Sect. 5.

4 Hybrid Elusive Case Repair Method

Our proposed artifact, denoted as *HERE*, encompasses three main activities: data preprocessing, Transformer training, and event log repair. As illustrated in Fig. 2, each activity involves a human-in-the-loop, which will be elaborated on in the subsequent subsections. *HERE* requires event logs in XES format as input, whereby it is assumed that the training event data is error-free and complete, consistent with typical supervised learning scenarios. Likewise, the event log designated for repair, which includes the elusive case requiring correction, should only exhibit this particular data quality issue, without any additional complexities, as illustrated in Table 3. The method is applicable to any XES event log that includes at least one attribute beyond the case ID, with activity name and timestamp being mandatory attributes and typically found in most event logs (De Weerdt and Wynn 2022). While the inclusion of additional attributes may enhance performance, they are not strictly necessary. The outcome of the method is an event log in XES format, representing the repaired version of the original erroneous log.

4.1 Data Preprocessing

The original dataset needs several transformations to conform with the requirements of our approach, depending on its initial structure. Thus, we have outlined the essential data preprocessing steps of *HERE* in Fig. 3. The first step involves adjusting the event log data to a uniform time zone. This adjustment ensures that the temporal relationships between individual events are accurately represented. Next, the event log is sorted by timestamp. If multiple events share the same timestamp, their original order is preserved. To enable the Transformer to process the timestamp, they are transformed into time progression in seconds from the first event of the sorted event log. Next to the timestamp, discrete input attributes (DAs) represent attributes with a limited number of distinct values, such as activity labels. Conversely, continuous input attributes (CAs) theoretically assume an infinite range of values within a specified interval, although practical representation is constrained by measurement precision and computational limitations, as seen with timestamps. For the repair of elusive cases, the output attribute, specifically the case ID, is crucial and is treated separately, as the Transformer model distinguishes between input and output attributes. Each CA is normalized individually using min-max scaling. In this process, the maximum value is mapped to 1, the minimum value to 0, and all other values are proportionally scaled between these two bounds. The results are detailed in Table 4. This ensures that all attributes have an equal impact on the analysis. Subsequently, both DAs and CAs are concatenated separately, aligning with the early fusion approach for multiple attributes proposed by Rivera Lazo and Ñanculef (2022). This means that all attributes within a category are merged into a single composite attribute for

Table 2 Participants in the semi-structured interview rounds

ID	Sector	Role	Process Mining Experience (Years)	Country	Round 1	Round 2
1	Research	Professor	21	Germany	✓	✓
2	Industry	Manager	4	Germany	✓	✓
3	Research	Research Assistant	4	Liechtenstein	✓	✓
4	Research	Research Assistant	3	Germany	✓	✓
5	Industry	Consultant	6	Austria	✓	✓
6	Industry	Process Expert	4	Germany	✓	✓
7	Research	Professor	9	Switzerland	✓	✓
8	Industry	Head of Center of Excellence	7	Germany	✓	✓
9	Industry	Head of Process Mining	4	Germany	✓	✓
10	Industry	Senior Consultant	6	Germany	✓	✓
11	Research	Postdoctoral Researcher	11	Belgium	✓	

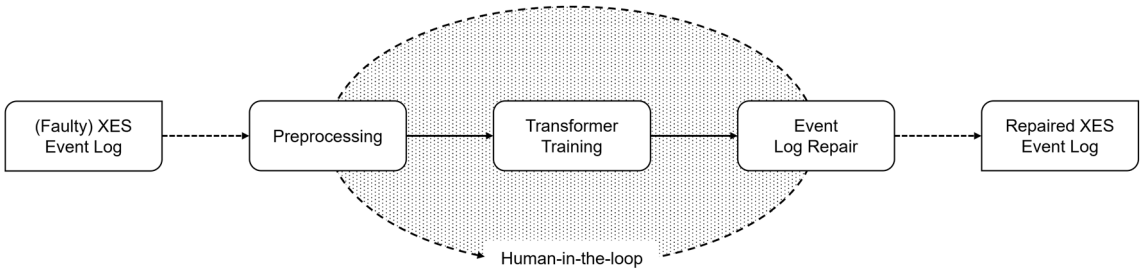

Fig. 2 Overview of *HERE*

Table 3 Event log excerpt for order-to-cash process showing elusive case pattern

Case ID	Activity	Timestamp	Resource
1	Order Received	2024-07-01T08:45:00+01:00	Staff A
	Order Processed	2024-07-01T09:00:00+01:00	Staff B
1	Payment Confirmed	2024-07-01T10:30:00+02:00	Staff C
..	..	..	..
	Order Delivered	2024-07-03T16:30:00+01:00	Staff D

each event. For instance, the DAs DA_1 to DA_d (where d denotes the total number of DAs) are combined into a single entity: $DA_1, \ldots, DA_d$. In our example, as illustrated in Table 4, this means that the value in the *Activity* column is combined with the value in the *Resource* column. The rationale behind this combination is that the Transformer model processes only one attribute at a time. However, according to the late fusion approach (Rivera Lazo and Ñanculef 2022), DAs and CAs are not merged at this stage.

In addition to event log data, our approach allows to integrate domain knowledge as per DO 1. This integration aims to enhance the output quality by explicitly stating domain knowledge that may not be fully captured in the data. Thereby, we define the target group to be researchers, process experts, and data engineers, while acknowledging that other stakeholders might be relevant as well. To facilitate interaction with the artifact, we employ a system that supports modular expansion with additional types of domain knowledge. For example, stakeholders may define domain knowledge such as *start activity*, *end activity*, and *directly following relationships*, based on predefined declarative rules (Di Ciccio and Montali 2022). While we have implemented expert knowledge concerning the control flow, the modular architecture also permits the integration of environmental, object, resource, and temporal aspects as distinct expert attributes (EAs). This flexible design does not require every EA to be specified, as stakeholders can include any number of EAs based on relevance. While performance may improve with the addition of more attributes, the system is designed to operate effectively even with minimal input. Each attribute requires specific values that satisfy defined criteria, along

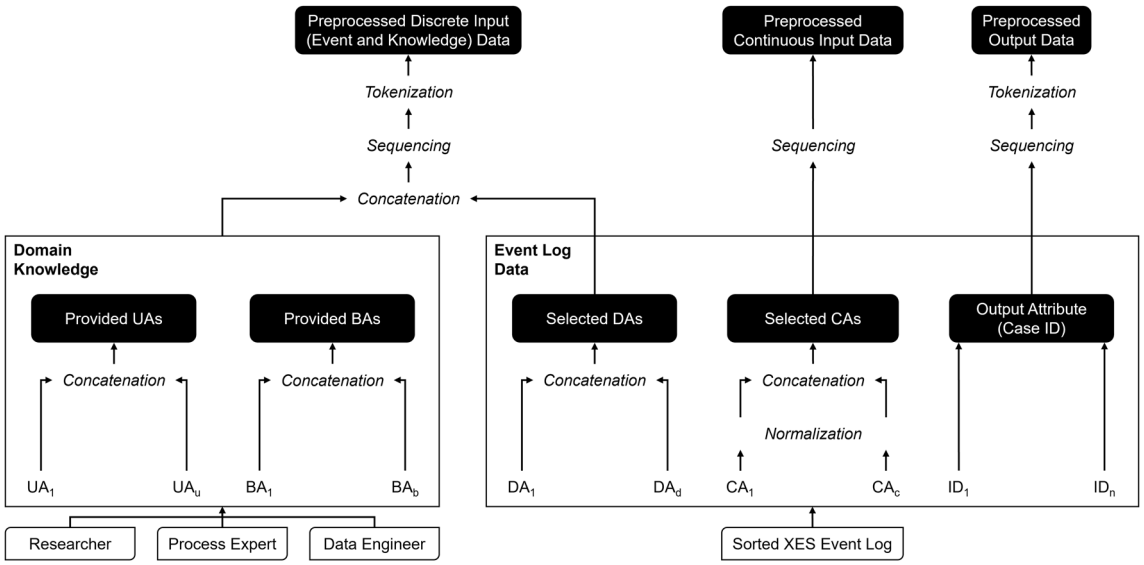

Fig. 3 Data preprocessing

Table 4 Preprocessed order-to-cash event log with relative time progression and concatenated activity and resource

Case ID	Activity	Relative Time Progression	Resource	Discrete Attributes
1	Order Received	0	Staff A	Order Received Staff A
	Order Processed	0.0045	Staff B	Order Processed Staff B
1	Payment Confirmed	0.0135	Staff C	Payment Confirmed Staff C
..	..	..	..	..
	Order Delivered	1	Staff D	Order Delivered Staff D

with specifying the frequency of fulfillment (always or sometimes) for each attribute value. For instance, an input could specify that events describing the *Order Received* activity must always be regarded as the start activity of a case. Stakeholders are supported through suggestions generated by querying the event log for these properties, potentially enabling non-experts to participate effectively. These suggestions streamline the input process for the implemented EAs, while other EAs may require additional coding, resulting in a more complex and time-consuming input process. To address this, stakeholders can save their inputs for automatic retrieval and pre-filling in future instances. We distinguish between unary and binary attributes: unary expert attributes (UAs) represent individual values (e.g., start activity), while binary expert attributes (BAs) denote relationships between two values (e.g., directly following). These attributes are incorporated as additional columns containing discrete values. For instance, the start activity attribute includes a value for each event indicating whether it always, sometimes, or never represents a start activity within the process. In our example, the start activity column would have the value *always start activity* for all events corresponding to the *Order Received* activity, while according to the expert input all other events would have the value *non start activity*, as demonstrated in Table 5. On the other hand, the directly following BA may necessitate the creation of multiple columns to account for distinct relationships. Initially, all predecessor activities are identified, with duplicates being removed. Each unique predecessor is then allocated a separate column (e.g., *Directly Following Order Received*), and the values in these columns indicate whether an activity directly follows its predecessor. These values are categorized as *non directly following*, *sometimes directly following*, or *always directly following*, depending on the input data. In our example, an expert may specify a directly following relationship between the activities *Order Received* and *Order Processed*, where the latter always follows the former, resulting in the outcome presented in Table 5. This procedure is repeated for each predecessor activity. Finally, the values from each column within the EA categories are concatenated for each event, following the procedure used for input attributes derived directly from the event log. Specifically, the values in the start activity column are combined with those in the end activity column, while the values in the directly following columns are concatenated together.

Table 5 Order-to-cash event log with start activity and directly following order received indicator

Case ID	Activity	..	Start Activity	Directly Following Order Received
1	Order Received	..	always start activity	non directly following
	Order Processed	..	non start activity	always directly following
1	Payment Confirmed	..	non start activity	non directly following
..	..	..	..	..
	Order Delivered	..	non start activity	non directly following

To provide these EAs for the Transformer's training, they are concatenated with the discrete attributes extracted from the event log data. This fusion consolidates all discrete input data into a unified representation for each event. Given d DAs, u UAs, and b BAs (where d, u and b represent the total counts of values in each category), the combined representation is structured as: $\mathrm{DA}_1, \ldots, \mathrm{DA}_d, \mathrm{UA}_1, \ldots, \mathrm{UA}_u, \mathrm{BA}_1, \ldots, \mathrm{BA}_b$. In our example, this includes values from columns such as activity, resource, start activity, directly following order received and potentially other attributes.

The discrete input data, continuous input data, and output data are systematically organized in chronological order to map the sequence of events within the ordered event log, accommodating potential case overlaps. Each of these sequences is uniform in length during training, corresponding to the number of events they include. By organizing the data into longer sequences instead of individual events, the Transformer model is better equipped to capture complex, long-term patterns within the sequence of events. This approach applies uniformly to all data, ensuring consistency in representation. After this step, as an example, the sequence of case IDs in the event log appears as follows: $\mathrm{ID}_{1+(i-1)k}, \ldots, \mathrm{ID}_{l+(i-1)k}; \ldots; \mathrm{ID}_{n-l_n+1}, \ldots, \mathrm{ID}_n$, where i denotes the sequence number, starting from 1 and increasing by 1 for each new sequence. The term k refers to the step value, which represents the fixed interval between two consecutive sequences. The variable l indicates the length of each sequence, i.e., the number of elements within each sequence. l_n specifically refers to the length of the final sequence in the set, while n represents the ID value of the last sequence, indicating its position in the overall order of the sequences. For instance, with a sequence length $l = 3$, the first sequence of input attributes would be: *Order Received Staff A always start activity non directly following*, ..., *Payment Confirmed Staff C non start activity non directly following*. During training, k is set to 1, ensuring consistent sequence lengths. However, in repair scenarios, k is adjusted to l, potentially resulting in the last sequence being shorter than others. The reason for the difference in k values between training and repair is that, during training, the goal is to learn the connections between events, whereas, during repair, the aim is to assign a single resulting case ID to each individual event. For instance, assuming a sequence length of 3, the first sequence consists of the case IDs for the first three events. The second training sequence would include the case IDs for the second through fourth events, ensuring overlap. This overlap helps the model capture relationships between consecutive events. In contrast, the second repair sequence would include the case IDs for the fourth through sixth events. This process continues until the case ID for the final event is mapped in a sequence.

In the final step, both the discrete input and output data are converted into tokens. Tokenization is a critical process, as it transforms the data into a format that is suitable for efficient processing by the Transformer model. During tokenization, each discrete value in the data is assigned a unique numeric ID based on a predefined dictionary. While the specific values of these IDs do not carry intrinsic meaning, ensuring that each token is uniquely identifiable is crucial for maintaining the integrity and accuracy of the data representation. In this study, we employ word-level tokens, where multi-word expressions are treated as single tokens by linking words with underscores. For instance, the activity *Order Received* is represented as *Order_Received*, which is then assigned a unique numeric ID, such as 5, within the dataset. In addition to the data-derived tokens, special tokens are employed for specific functions. These tokens, which do not represent actual data values, are essential for guiding the model's processing. The start-of-sequence (SOS) token indicates the beginning of a sequence, while the end-of-sequence (EOS) token marks its conclusion. These tokens help the model identify where sequences start and end, ensuring proper processing. Padding tokens are used to handle sequences of varying lengths by adding extra tokens to shorter sequences, making all sequences in a batch the same length. Unknown tokens are used to represent any values in the data that are not part of the predefined dictionary, ensuring the model can still process such cases. In practice, each input sequence is prefixed with an SOS token and suffixed with an EOS token, resulting in the format: $\mathrm{SOS}, \text{input sequence}, \mathrm{EOS}$. For example, the known sequence of input attributes would be represented as: $SOS, 5\,2\,7\,9, \ldots, 8\,4\,6\,9, EOS$. For the output sequence, only the SOS token is added at the beginning to signal the

commencement of the decoding process, with subsequent tokens generated sequentially. The EOS token is placed at the end of the training labels, which represent the ground truth sequences, to indicate the termination of the output. The EOS token plays a critical role during training, as it explicitly signals when the model should stop generating tokens. Without the EOS token, the model may produce superfluous or incorrect outputs. Therefore, learning to generate the EOS token at the appropriate time is essential for ensuring proper sequence termination and maintaining output accuracy. After tokenization, the data is represented as a sequence of numeric IDs. This standardized representation is essential for the Transformer's functionality, enabling consistent operations such as embedding (converting tokens into vectors), attention mechanisms that prioritize relevant input elements, and decoding to generate the output.

4.2 Training

After preprocessing, the data is partitioned into two subsets: a training set (90%) used to train the Transformer model, and a test set (10%) used to evaluate the model's performance on unseen data. The architecture of the Transformer is illustrated in Fig. 4. Initially, all three types of data, discrete input, continuous input, and output are embedded. Embedding is a technique that transforms the data into compact vector representations, which are multi-dimensional numerical arrays that capture the inherent properties of the event data. This transformation is crucial because the Transformer model requires these vector representations to effectively process and understand the input data. In this embedded space, semantically unrelated events are represented by vectors that are distantly spaced, while related or similar events are positioned closer together. Following embedding, positional encoding is applied to the inputs of both the encoder and decoder. This technique adds a unique vector to each token's embedding to indicate its position within the sequence, as Transformers process all tokens simultaneously and lack an inherent sense of order. Positional encodings, created by applying sine and cosine functions to token positions within the sequence, enable the model to identify relationships and contextual information among tokens based on their positions, thus improving its ability to manage sequential data.

The encoder in the Transformer operates on discrete input data through multiple layers sharing identical structures. Central to the encoder's design, the multi-head attention mechanism computes weighted dependencies between input elements, allowing the model to capture the relationships between them. By generating multiple parallel representations of each input, it enables the simultaneous extraction of diverse patterns, enhancing the model's ability to encode complex relational structures within the event data. Following the attention mechanism, the outputs undergo normalization to ensure stable training. This process adjusts the activations, or output values, produced by each layer and addresses challenges such as exploding or vanishing gradients. Exploding gradients cause instability in training by producing excessively large updates, while vanishing gradients hinder learning by making updates too small. Gradients, which are the partial derivatives of the loss function with respect to model parameters, guide the updating of the model's parameters during training. By stabilizing the gradient flow, normalization ensures that extreme values do not disrupt training, thus facilitating effective parameter updates and accelerating convergence. The data is then passed through a feed forward network, which consists of multiple fully connected layers. Each layer refines the input representation by applying a series of transformations, progressively enhancing its ability to capture and express relevant features. Finally, additional normalization is applied to maintain consistency and stability in the output. Collectively, these operations allow the model to generate more accurate and robust representations of the input data. To integrate continuous input data into the processing pipeline, the model employs a late fusion approach, where continuous data is first encoded using a feed forward network. This step is essential as it converts the data into a fixed-size format compatible with the Transformer's architecture, enabling effective processing by the attention mechanism. The encoded continuous data is then concatenated with the output of the encoder. This approach, as outlined by Rivera Lazo and Ñanculef (2022), enables the model to seamlessly handle both discrete and continuous data types within a unified modeling framework, enhancing its capacity to process heterogeneous data sources within the event log.

The Transformer's decoder is similarly structured with multiple layers. Initially, it employs masked multi-head self-attention, enabling the model to focus on relevant segments of the input while preventing consideration of future tokens, which correspond to case IDs of following events. This ensures that each prediction in the output sequence relies solely on previous predictions, which is crucial for maintaining the autoregressive nature of sequence generation, where each output is generated step by step based on preceding outputs. Following this, layer normalization is applied to stabilize learning. Importantly, the decoder integrates information from the enhanced encoder's output through a second multi-head attention mechanism known as encoder-decoder attention. This dual mechanism establishes connections between input and output sequences. After incorporating the encoder's output, the decoder undergoes another normalization layer to refine its internal representations. It subsequently processes data

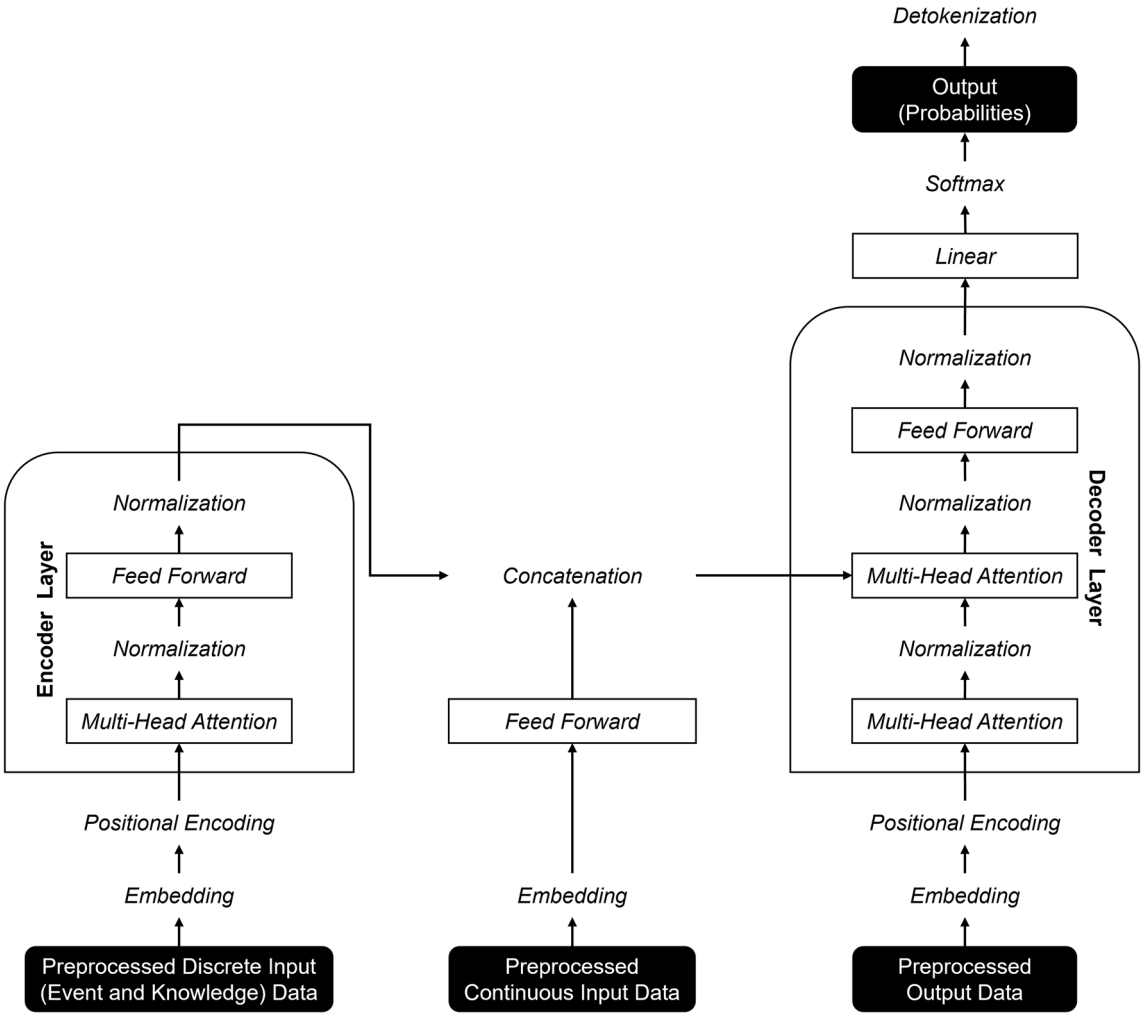

Fig. 4 Transformer architecture

through a feed forward network, followed by another normalization step. This iterative cycle of attention, normalization, and feed forward operations continuously refines the decoder's outputs by focusing on relevant input parts, stabilizing representations, and capturing complex patterns, all while leveraging the relationships between input and output to generate contextually accurate and coherent output sequences.

The decoder's outputs are then transformed from multi-dimensional embeddings into a sequence of probabilities via a final linear layer followed by a softmax operation. The linear layer maps embeddings to token logits, which are raw, unnormalized scores representing the likelihood of each token. The softmax function then converts these logits into a probability distribution over output tokens, normalizing them so that they sum to 1. This distribution represents the likelihood that a given event corresponds to a specific case ID. The token with the highest probability is then selected and detokenized, returning it to its original form as a case ID within the output sequence. During training, the model utilizes the probability distribution to compute loss. This loss measures the discrepancy between the predicted probability distribution and the true label for each token in the sequence, crucial for training and improving sequence generation accuracy. The total loss, summed across all tokens in the sequence, guides the computation of gradients with respect to the model weights. These gradients inform the optimization algorithm in updating the model weights, aiming to minimize loss and enhance the model's ability to generate accurate sequences. The model undergoes multiple training cycles, each comprising a forward pass (sequence generation) and a backward pass (optimization), during which it processes the entire dataset once and updates its weights. These cycles, referred to as epochs, enable the model to iteratively improve its performance over time.

4.3 Repair

Once the Transformer model is trained, it can be used to repair elusive cases. Our hybrid solution, as illustrated in Fig. 5, merges Transformer-based repairs with declarative rules within the repair process. This approach is selected as Transformer-generated outputs often lack interpretability due to the model's black-box nature. By incorporating declarative rules, we aim to enhance transparency and comprehensibility of the outputs, leveraging explicit domain knowledge. The preprocessing of event logs containing the elusive case imperfection pattern mirrors the steps outlined in Sect. 4.1 before the repair process commences.

Each repair iteration starts with an ex-ante rule check, which assigns case IDs based on the specified EAs. This step is followed by a Transformer-based repair and may conclude with an ex-post rule check that resets incorrectly assigned case IDs using the same set of EAs. If certain EAs were not utilized during the training phase, they can be integrated into the repair process. This integration is facilitated through a questionnaire-style query, prompting stakeholders to specify values and frequencies for the selected EAs, analogous to the training phase. This newly acquired knowledge enhances the existing domain knowledge and is incorporated into the rule checks by integrating it into the relevant variables. However, retraining the Transformer proves advantageous for incorporating significant new insights into the model, such as changes in activity sequences. Stakeholders determine the necessity of rule checks based on the available domain knowledge. At the end of each repair iteration, the repaired event logs are saved in both XES and comma-separated values formats. Furthermore, during the initial repair iteration, the proportion of events lacking case IDs is presented for both the original log and its repaired counterpart. Following each subsequent repair iteration, stakeholders are updated on the percentage of these events within the repaired event log at that stage. Should this percentage be greater than zero, stakeholders have the option to either accept the current outcomes and conclude the repair process, acknowledging that some events are still missing case IDs, or to continue with additional repair iterations. In each subsequent iteration, the repaired event log from the preceding iteration serves as input. Once the percentage of events without assigned case IDs reaches zero, the repair automatically concludes.

4.3.1 Ex-ante Rule Check

If an ex-ante rule check is performed, the objective is to determine case IDs according to predefined rules. These rules, grounded in domain knowledge, are represented as EAs. These attributes include the corresponding values and occurrences derived from questionnaire-like queries. Each attribute requires a unique implementation tailored to its specific assumptions. For instance, consider the start activity attribute. The event log is grouped into cases, with non-compliant cases identified as those beginning with an activity other than the prescribed start activities. Events that lack a case ID and correspond to a prescribed start activity are then detected. A search is conducted to identify the nearest non-compliant case, based on temporal proximity. This strategy ensures that the event is linked to the most temporally appropriate case, minimizing assignment errors. If the time difference between the event and the start activity of the case falls within a predefined window, the case ID of the closest match is assigned to the event. This

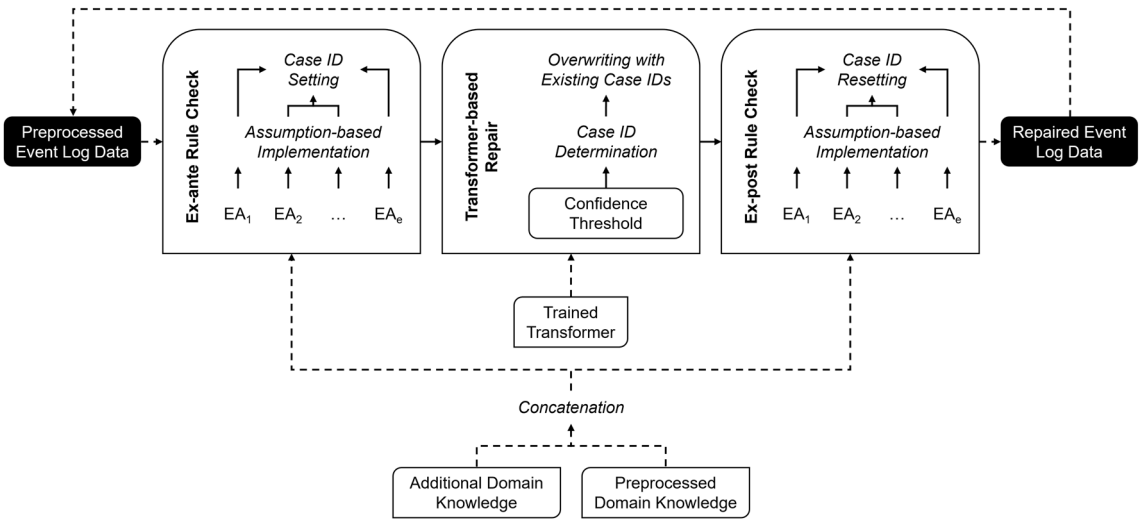

Fig. 5 Repair approach

time window serves to prevent erroneous assignments, such as linking an event to a case occurring much later. By adopting this approach, compliance with prescribed rules is ensured, as events are assigned to cases in a manner consistent with the correct temporal context. UAs like start activity, end activity, mandatory occurrence, or single occurrence apply to individual events. BAs, however, necessitate consideration of pairs of events. For instance, in the case of a directly following relationship between two activities, each relationship is analyzed individually. We define a directly following relationship such that each predecessor activity is followed by a single successor, ensuring a clear and unambiguous sequence. When an activity is involved in multiple relationships, the order in which these relationships are processed determines the evaluation sequence. For each case in the event data, we examine whether discrepancies exist between the counts of predecessors and successors, as such imbalances may indicate missing events. Predecessors without a case ID are assigned to the temporally closest case where the number of successors exceeds that of predecessors up to that point, provided that the time difference between the predecessor and its successor is within an acceptable range for that case. A similar procedure applies to successors without a case ID. Other BAs may include mutual exclusion and non-contiguity, which were not examined or implemented in this study.

4.3.2 Transformer-Based Repair

The Transformer-based repair process follows a structure similar to training, as illustrated in Fig. 4. First, the discrete input data is encoded as described above, and its output is integrated with the processed continuous input data. However, the decoder employs a different approach. Starting with only an SOS token, the decoder utilizes multi-head attention, normalization, and feed forward processing similar to the training phase. It then generates the case ID tokens iteratively, where each token is predicted based on the sequence of previously generated tokens. Specifically, the SOS token generates the first case ID token, which is subsequently used in combination with the SOS token to predict the second case ID token, and so on. At each step, the model selects the most probable next token from the softmax probability distribution, progressively constructing the full sequence. To avoid selecting tokens that represent unknown or uncertain predictions, the weights of such tokens are automatically adjusted before applying the softmax layer, reducing their probability and minimizing their chance of selection. Furthermore, a configurable confidence threshold enables stakeholders using *HERE* to exclude tokens that are uncertain, balancing output completeness with accuracy. Specifically, when identifying a case ID token, its softmax probability is compared against the threshold. Tokens exceeding this threshold are included; otherwise, a special token indicates an absence of value.

In typical language translation tasks, the input and output sequences can differ in length. However, in our approach, we require that the number of case IDs matches the number of input events exactly, with each input event being assigned a specific case ID. If a premature EOS prediction occurs, meaning the model predicts the end of the output sequence too early, we ignore that prediction and instead select the second most probable token. This strategy ensures that each event is generally assigned a corresponding case ID, thereby preserving the integrity of the output, as long as the Transformer is able to make a prediction. The repair process concludes upon achieving the desired sequence length, thereby preventing excessive case ID generation. Tokens are then converted back to their original values using the tokenizer employed during training. The resulting sequence of case IDs is then split into individual case IDs, which are reassigned to the event log. Enhancements include providing stakeholders with token probabilities and the likelihood of subsequent tokens, thereby increasing their awareness of data quality (Evron et al. 2022). In scenarios where not every event requires repair and a new case ID does not need to be determined for each event (i.e., when elusiveness is less than 100%), the case IDs for events that do not require repair remain unchanged and are reused. This is based on our assumption that all provided data is correct, aiming to avoid replacing existing correct case IDs with predictions by the Transformer. Additionally, previously determined case IDs, such as those from ex-ante rule checks, are thereby retained.

4.3.3 Ex-post Rule Check

At the end of each repair iteration, an ex-post rule check can be conducted. This procedure is similar to the ex-ante rule check, whereby various EAs and their implementations are used. Unlike its ex-ante counterpart, the primary objective in the ex-post rule check is to rectify inaccurately assigned case IDs by the Transformer. For example, when considering start activities, each case in the event log is assessed to determine if the designated start activity is present and correctly located. If found included but not as the first event, the case ID from preceding events is reset. This reset is applied only to those events that did not have a case ID recorded in the event log prior to the first repair cycle. Again, appropriate procedures for each EA are crucial to ensure the correct application of the rules. Upon completion of the ex-post rule check, the repaired event log is finalized, as exemplified in Table 6, and shared with stakeholders.

Table 6 Repaired order-to-cash event log

Case ID	Probability	Activity	Timestamp	Resource
1	–	Order Received	2024-07-01T08:45:00+01:00	Staff A
1	65.07%	Order Processed	2024-07-01T09:00:00+01:00	Staff B
1	–	Payment Confirmed	2024-07-01T10:30:00+02:00	Staff C
..	..	..	..	..
2	89.58%	Order Delivered	2024-07-03T16:30:00+01:00	Staff D

5 Evaluation

5.1 Artificial Formative Evaluation

In the initial stage of the evaluation, we conducted an artificial formative assessment to deeply understand the research problem, explore potential solutions, and gather feedback on the design to refine and improve it during development. This phase began with a literature review to identify existing approaches for addressing event log imperfections in general and the elusive case pattern in particular. We built on related work in the field and formulated DOs and a design specification (cf. Table 7). These were then assessed through 11 semi-structured interviews with experts from both research and industry. A condensed version of the interview structure is available in the online appendix (Appendix A; available online via http://link.springer.com). Each interview started with a detailed motivation and introduction to the research problem, focusing significantly on elusive cases and how they manifest. Afterwards, we presented our preliminary DOs and explained how the design specification was derived. The design specification was then introduced and demonstrated with examples.

Finally, interviewees were asked to provide general feedback on the design specification and to rate it using an end-labeled unipolar seven-point Likert scale (Höhne et al. 2021) across four criteria suitable for an ex-ante evaluation (Sonnenberg and vom Brocke 2012). Thereby, we chose *novelty* for justifying the problem statement, research gap and DOs as it ensures the contribution is unique and advances current knowledge; *understandability* was selected as the design must be clear and accessible to our target group encompassing both practitioners and researchers with diverse backgrounds and levels of expertise; *completeness* was important for validating that the design specification covers all necessary aspects of the research problem; and *applicability* was included to ensure the design can be effectively implemented in real-world settings (Sonnenberg and vom Brocke 2012). Each criterion was introduced to the interviewees using a definition and one or more guiding questions, as can be seen in Table 8. For each criterion, the interviewees provided qualitative feedback to justify their decision. In the end, additional time was given to discuss any additional feedback points not addressed by the criteria. The results of the quantitative evaluation are shown in Fig. 6.

Interviewees, denoted as "I" followed by the ID as per Table 2 for identification purposes, generally found the

Table 7 Design specification with references

Design Specification	References
Generative AI has demonstrated potential in data reconstruction, motivating us to apply this technology for reconstructing case IDs	Hofmann et al. (2021)
As we want to convert multiple input parameters (event log attributes) into a single output (case IDs), we define our machine learning problem as a translation task	Sutskever et al. (2014)
Proven state-of-the-art performance in sequence-to-sequence translation tasks motivates us to build on the Transformer architecture	Vaswani et al. (2017)
Event logs encompass various types of attributes, requiring us to integrate these attributes by concatenating DAs for the encoder's input and appending CAs to the encoder's output	Rivera Lazo and Ñanculef (2022)
Establishing declarative rules based on domain knowledge, through an interactive human-in-the-loop approach, allow us to improve robustness	Chen et al. (2020)
By integrating generative AI with human intelligence via traditional rule checks, problem-solving capabilities are enhanced	Raisch and Fomina (2024)
By presenting associated probabilities used to determine case IDs, stakeholder awareness regarding output quality is improved. Thereby, correctness and completeness of the output can effectively be balanced by stakeholders	Evron et al. (2022)

Table 8 Artificial formative criteria definitions and guiding questions

Criterion	Definition	Guiding Questions
Understandability	The comprehensibility of the elements incorporated in the design specification.	How accessible is the design specification? Is the objective apparent to you?
Novelty	The uniqueness and originality of the approach employed.	Are you aware of any comparable methods or strategies to address this issue?
Completeness	The extent to which our design specification reflects the objective of our research.	Are there any elements within the design specification that are missing or incomplete?
Applicability	The suitability of the design specification in solving the problem at hand.	What potential challenges do you foresee in applying the design specification in real-world scenarios?

Fig. 6 Average Likert scale ratings for artificial formative evaluation criteria

design specification easy to follow, facilitating a clear understanding of each component. However, they noted that this level of understandability is primarily for the desired target group. For instance, I8 mentioned, *"If you know how process mining works, then it's easy to understand"*, while I2 highlighted the limitation that *"For people who are not tech-savvy, it is more difficult to understand."* In addition to these limitations, specific components were criticized regarding understandability. I11 pointed out that it was unclear whether the method requires sequences of events or individual events one by one, suggesting that *"for non-technicians you should be more precise about the decoder and encoder working iteratively and you should also describe specifically how that works with an event log."* Additionally, I11 identified issues with understanding how rules defined by domain experts are incorporated, and I6 questioned the autonomy of the method without domain knowledge. Similarly, I5 also required clarification on the goals of involving a human-in-the-loop. While the feedback given for the criterion understandability did not directly affect the design specification of the artifact itself, it prompted us to address all these points more clearly when formulating Sect. 4.

Regarding novelty, the interviewees appreciated the use of a modern machine learning architecture and, more importantly, the integration of a human-in-the-loop approach. Thereby, all interviewees agreed that the approach is novel. However, some pointed out that using machine learning to estimate case IDs is not new. I6 directed attention to trace clustering, acknowledging its relevance but noting such algorithms are not case-specific, meaning they override the existing case ID logic and labels. This results in a complete regrouping of events while disregarding potentially correct events. Similarly, I1 remarked that while the individual technologies and components are new and their combination is novel, the problem of event correlation itself is not new. I1 nonetheless commended the approach, stating it has the potential to scale better and achieve higher accuracy than existing methods.

Interviewees agreed that the design is complete in addressing elusive cases but offered suggestions for enhancing the method. For example, I3 stated that *"Probabilities are assessed subjectively, which is why it is good to have comparative values."* Therefore, we have chosen to not only output the probability associated with the determined case ID, but also indicate the probability of the second most likely value. This change allows our target group to better assess the probability values, thus enhancing quality awareness for each prediction. I6 emphasized the importance of enriching an event log with domain knowledge and integrating additional data to improve predictions, recommending the integration of a realistic process model to assist the algorithm in understanding the underlying order of events. I5 suggested leveraging the natural language processing capabilities of Transformers to convert standardized process documents into a format suitable for an additional input layer, enhancing the

model's ability to interpret and utilize process information effectively. I6 also pointed out that the approach might suffer from other event log quality issues such as erroneous timestamps, suggesting a combination with other event log repair methods. Additionally, interviewees raised concerns about formulating the problem as a supervised machine learning task, with I1 recommending an *"architectural design to correlate event pairs"* instead of predicting a specific case ID for each event. We consider all these feedback points to be highly relevant and believe that each individual direction would make a contribution to the field. Due to the scope of this study, we decided to continue with the initial architectural design and provide a proof of concept for this architecture first. While we made a few adjustments, the foundational elements of the architecture were not altered. Nonetheless, we acknowledge these suggestions as promising opportunities for future research.

Interviewees agreed on the general applicability of the method but highlighted the importance of considering the cost and resources associated with its implementation. I2 furthermore stressed that for the method to be relevant in practice, it must be integrated into software natively and meet the necessary performance requirements of the use case: *"In research, there is a lot of fuzz about improving algorithms. In practice, no one uses such research prototypes. So next to cost, resources, etc. it is important that it is integrated in some software natively. Also, it must fulfill the requirements regarding the necessary accuracy of the use case."* I2 also questioned whether generative AI is the best solution compared to changing underlying systems to prevent errors: *"You should ask yourselves the question: do I have to use generative AI to solve that problem? Shouldn't I instead change the underlying system so the error does not even occur? In which scenarios are the resources needed for such a software smaller than to fight IT so they implement changes in the underlying systems?"* As these are all valid points of feedback, we position our research as a solution for addressing scenarios where the elusive case has already occurred and how it can be remedied. However, it is important to note that process mining analyses rely significantly on historical data. Hence, the overarching objective should focus on avoiding such errors proactively in the future. Concerns were also raised about the input data quality, with I11 doubting the output quality due to the assumption of perfect data and I10 noting the method's inapplicability to other event log formats required for object-centric process mining. Both points offer interesting directions for future research.

5.2 Artificial Summative Evaluation

Building on the insights gained from the formative stage, we continued with the artificial summative evaluation. In this phase, we refined our design specification and instantiated the method as a software prototype, serving as an initial proof of concept. To simulate a realistic context, we introduced elusive cases into event logs that were initially free from this pattern by randomly deleting case IDs from existing events. Our method was then applied to repair these logs, allowing us to compare the results against the known ground truth. This comparison enabled us to calculate various metrics, demonstrating the technical feasibility and effectiveness of our solution. To enable a fair comparison with other approaches, we repeated this whole evaluation procedure with various benchmarks. Hence, this stage serves as a proof of concept, providing preliminary evidence of the artifact's effectiveness and potential utility.

5.2.1 Evaluation Data

For our baseline evaluation data, we selected three publicly available event logs, chosen for their diverse characteristics, including both synthetic and real-world data. These logs differ in size and complexity, enabling us to assess the robustness and effectiveness of our approach. Table 9 provides a summary of the key characteristics of the event logs used in this evaluation phase.

For each log, elusive cases were introduced in increments of 10%, ranging from 10% to 90%. This means that each log was modified with nine different degrees of elusiveness: at the least severe stage, only 10% of events lacked a case ID, while at the most severe stage, 90% of events were missing a case ID. For each of the three baseline logs, we determined three EAs by analyzing the data, defining them as follows:

- Review: The start activity is always *invite reviewers*, while *accept* and *reject* can sometimes serve as end activities. In the baseline log, the consistency of these rules is 100%. Directly following relationships are sometimes observed between *invite additional reviewer* and itself, as well as between *invite additional reviewer* and *get review X*, and *invite additional reviewer* and *time-out X*.
- Renting: The start activity is always *Apply for Viewing Appointment*, while *Reject Prospective Tenant*, *Tenant Cancels Apartment*, and *Evict Tenant* can sometimes function as end activities. The baseline log exhibits 100% consistency for these rules. Directly following relationships are sometimes present between *Pay Rent* and itself, *Apply for Viewing Appointment* and *Set Appointment*, and *Set Appointment* and *View The Property*.
- Hospital Billing: The start activity is always *NEW*, while *BILLED*, *NEW*, and *DELETE* can sometimes serve as end activities. The original log shows 100%

Table 9 Event log statistics

Log Name	Events	Variants	Average Trace Length	Standard Deviation of Trace Length	Cases
Review (synthetic)	236,360	4,118	23	8	10,000
Renting (synthetic)	96,440	508	9	6	10,000
Hospital Billing (real-life)	451,359	1,020	4	2	100,000

consistency for the start activity and 94.12% for the end activity. Directly following relationships are sometimes observed between *FIN* and *RELEASE*, *RELEASE* and *CODE OK*, and *CODE OK* and *BILLED*.

Finally, all these EAs were added to each event log and elusiveness. Hence, this approach provides 27 event logs in total for repair, covering various characteristics and degrees of elusiveness.

5.2.2 Evaluation Metrics

Each repaired event log was evaluated based on ten metrics aligned with DO 3, which emphasizes multiple dimensions of data quality such as accuracy, consistency, and completeness. Among these, the first two metrics, which address *completeness* and *consistency*, were defined by the authors while the remaining eight metrics, which evaluate different aspects of *accuracy*, are adopted from the work by Bayomie et al. (2023).

Metric 1 (*Completeness*) quantifies the extent to which missing case IDs in the log are resolved after repair. Let L_{err} represent the erroneous event log and L' the repaired event log. The set of events in L_{err} with missing case IDs is denoted as $E_{\text{err}} \subseteq E$, where E is the set of all events in L. After repair, let $E_{\text{rep}} \subseteq E_{\text{err}}$ represent the subset of events with resolved case IDs in L'. Completeness is then defined as:

$$CPL = \begin{cases} 1, & \text{if } |E_{\text{err}}| = 0 \\ \dfrac{|E_{\text{rep}}|}{|E_{\text{err}}|}, & \text{otherwise,} \end{cases}$$

where $|\cdot|$ denotes the cardinality of the set. In cases where there are no events with missing case IDs ($|E_{\text{err}}| = 0$), *CPL* is defined as 1, indicating that the log is already complete.

Metric 2 (*Consistency*) measures the extent to which repaired logs conform to predefined domain-specific rules. Let $R = \{r_1, r_2, \ldots, r_k\}$ denote the set of rules applicable to cases in the log, where each rule r_i specifies a consistency condition based on domain knowledge. For a given rule r, let $C_r \subseteq I$ represent the set of cases in L' that satisfy r, where I is the set of all cases in L'. The consistency for rule r, CON_r, is defined as:

$$CON_r = \frac{|C_r|}{|I|}.$$

To compute the overall consistency *CON* across all rules, we aggregate the individual consistency values. This can be done either as an arithmetic mean or a weighted mean, depending on the relative importance of each rule r:

$$CON = \frac{\sum_{r \in R} w_r \cdot CON_r}{\sum_{r \in R} w_r},$$

where w_r is the weight assigned to rule r. If all rules are equally important, $w_r = 1$ for all r, whereby *CON* becomes the arithmetic mean, which we apply in Sect. 5.2.4.

For accuracy, we propose a more nuanced approach due to its inherently strict nature: while a repaired case ID may not exactly match the ground truth case ID, it can still be highly plausible while having no adverse effects on downstream analysis tasks. For instance, if two events are very similar – sharing the same activity, resource, and occurring close in time – switching their case IDs during repair will not impact the overall utility of analyses such as process discovery. To reflect this complexity, we adopt six case similarity and two time proximity metrics reflecting different aspects of elusive case repair as proposed by Bayomie et al. (2023).

Metric 3 (*Trace-to-Trace Similarity*) assesses how closely two event logs capture the same control-flow by using a string-edit distance $\Delta_{\text{ins\_del}}$ based on insertions and deletions of activities to compare unique traces between the ground truth and repaired logs. Thereby, let $T = \{t_1, t_2, \ldots, t_{|T|}\}$ and $T' = \{t'_1, t'_2, \ldots, t'_{|T'|}\}$ represent the set of distinct traces in the ground truth log L and the repaired log L', respectively. For each trace $t \in T$, the trace-closest trace $t^* \in T'$ is defined as the trace in L' which minimizes $\Delta_{\text{ins\_del}}(t, t^*)$. Hence, the trace-to-trace similarity S_{trace} is defined as:

$$S_{\text{trace}} = 1 - \frac{\sum_{t \in T} \Delta_{\text{ins\_del}}(t, t^*)}{\sum_{t \in T}(|t| + |t^*|)},$$

with $|t|$ and $|t^*|$ being the lengths of the respective traces.

Metric 4 (*Trace-to-Trace Frequency Similarity*) evaluates how closely two event logs align by considering both

the structure of their traces and their frequencies. The metric finds the optimal one-to-one mapping between cases in the ground truth log L and the repaired log L' that minimizes the total string-edit distance Δ_{total} across all traces in L. The trace-to-trace frequency similarity S_{freq} is then defined as:

$$S_{\text{freq}} = 1 - \frac{\Delta_{\text{total}}}{2 \times |E|},$$

where $|E|$ is the total number of events in the logs.

Metric 5 (*Partial Case Similarity*) evaluates how similar two event logs are by counting the overlap of events in cases that share the same starting event. For each pair of cases $\sigma \in S(L)$ in the ground truth log and $\sigma' \in S(L')$ in the repaired log, the function $\text{intersect}(\sigma, \sigma')$ counts the number of common events (excluding the first event) which is then averaged across all cases to compute the partial case similarity S_{partial}:

$$S_{\text{partial}} = \frac{\sum_{\sigma \in S(L),\, \sigma' \in S(L') : \sigma[1] = \sigma'[1]} \text{intersect}(\sigma, \sigma')}{|E| - |I|},$$

where $|E|$ is the total number of events in the logs, $|I|$ is the total number of cases, and $\sigma[1]$ represents the first event in case σ.

Metric 6 (*Bigram Similarity*) evaluates how similar two event logs are based on the overlap of bigrams (sequences of two consecutive events) between the logs. For each case $\sigma \in S(L)$, the function $\text{occurs}_1(\langle e, e' \rangle, L')$ checks if a bigram $\langle e, e' \rangle$ from the ground truth log L also appears in the repaired log L'. It is defined as:

$$\text{occurs}_1(\langle e, e' \rangle, L') = \begin{cases} 1, & \text{if the bigram } \langle e, e' \rangle \text{ exists in } L' \\ 0, & \text{otherwise.} \end{cases}$$

The bigram similarity S_{bigram} is then computed as the average proportion of bigrams in the ground truth log L that also occur in L', normalized by the number of events in each case:

$$S_{\text{bigram}} = \frac{1}{|I|} \sum_{\sigma \in S(L)} \frac{1}{|\sigma| - 1} \sum_{i=1}^{|\sigma|-1} \text{occurs}_1(\langle \sigma(i), \sigma(i+1) \rangle, L'),$$

with $|I|$ as the total number of cases in the logs, and $|\sigma|$ as the number of events in a case σ.

Metric 7 (*Trigram Similarity*) evaluates how similar two event logs are based on the overlap of trigrams (sequences of three consecutive events) between the logs. For each case $\sigma \in S(L)$, the function $\text{occurs}_2(\langle e, e', e'' \rangle, L')$ checks if a trigram $\langle e, e', e'' \rangle$ from the ground truth log L also appears in the repaired log L'. It is defined as:

$$\text{occurs}_2(\langle e, e', e'' \rangle, L')) = \begin{cases} 1, & \text{if the trigram } \langle e, e', e'' \rangle \text{ exists in } L' \\ 0, & \text{otherwise.} \end{cases}$$

The trigram similarity S_{trigram} is then computed as the average proportion of trigrams in the ground truth log L that also occur in L', normalized by the number of trigrams in each case:

$$S_{\text{trigram}} = \frac{1}{|I|} \sum_{\sigma \in S(L)} \frac{1}{|\sigma| - 2} \sum_{i=1}^{|\sigma|-2} \text{occurs}_2(\langle \sigma(i), \sigma(i+1), \sigma(i+2) \rangle, L'),$$

with $|I|$ as the total number of cases in the logs, and $|\sigma|$ as the number of events in a case σ.

Metric 8 (*Case Similarity*) measures the extent to which two event logs match in terms of their cases. It compares the sets of cases from the ground truth log L and the repaired log L' and determines the proportion of identical cases. Formally, it is defined as:

$$S_{\text{case}} = \frac{|S(L) \cap S(L')|}{|I|},$$

where $S(L) \cap S(L')$ represents the set of cases that are identical in both logs, and $|I|$ is the total number of cases.

Metric 9 (*Event-Time Deviation*) evaluates the difference in elapsed times of events between the ground truth log L and the repaired log L'. It uses the symmetric mean absolute percentage error (SMAPE) to quantify deviations, hence constituting a measure based on relative errors. Formally, it is defined as:

$$S_{\text{time}} = \frac{\sum_{e \in E} \frac{|\text{ET}(L,e) - \text{ET}(L',e)|}{|\text{ET}(L,e)| + |\text{ET}(L',e)|}}{|E| - |I|},$$

where $\text{ET}(L, e)$ is the elapsed time of event e in the ground truth log L, $\text{ET}(L', e)$ is the elapsed time of event e in the repaired log L, $|E|$ is the total number of events, and $|I|$ is the number of cases.

Metric 10 (*Case Cycle Time Deviation*) assesses the relative deviation in cycle times between the ground truth log L and the repaired log L'. This metric compares pairs of cases starting with the same event and uses the SMAPE for its calculation:

$$S_{\text{cycle}} = \frac{1}{|I|} \sum_{\substack{\sigma \in S(L),\, \sigma' \in S(L') \\ \sigma(1) = \sigma'(1)}} \frac{|\text{CT}(\sigma) - \text{CT}(\sigma')|}{|\text{CT}(\sigma)| + |\text{CT}(\sigma')|},$$

with $\text{CT}(\sigma)$ as the cycle time of case σ, computed as the elapsed time between its first and last events, $\sigma \in S(L)$ and

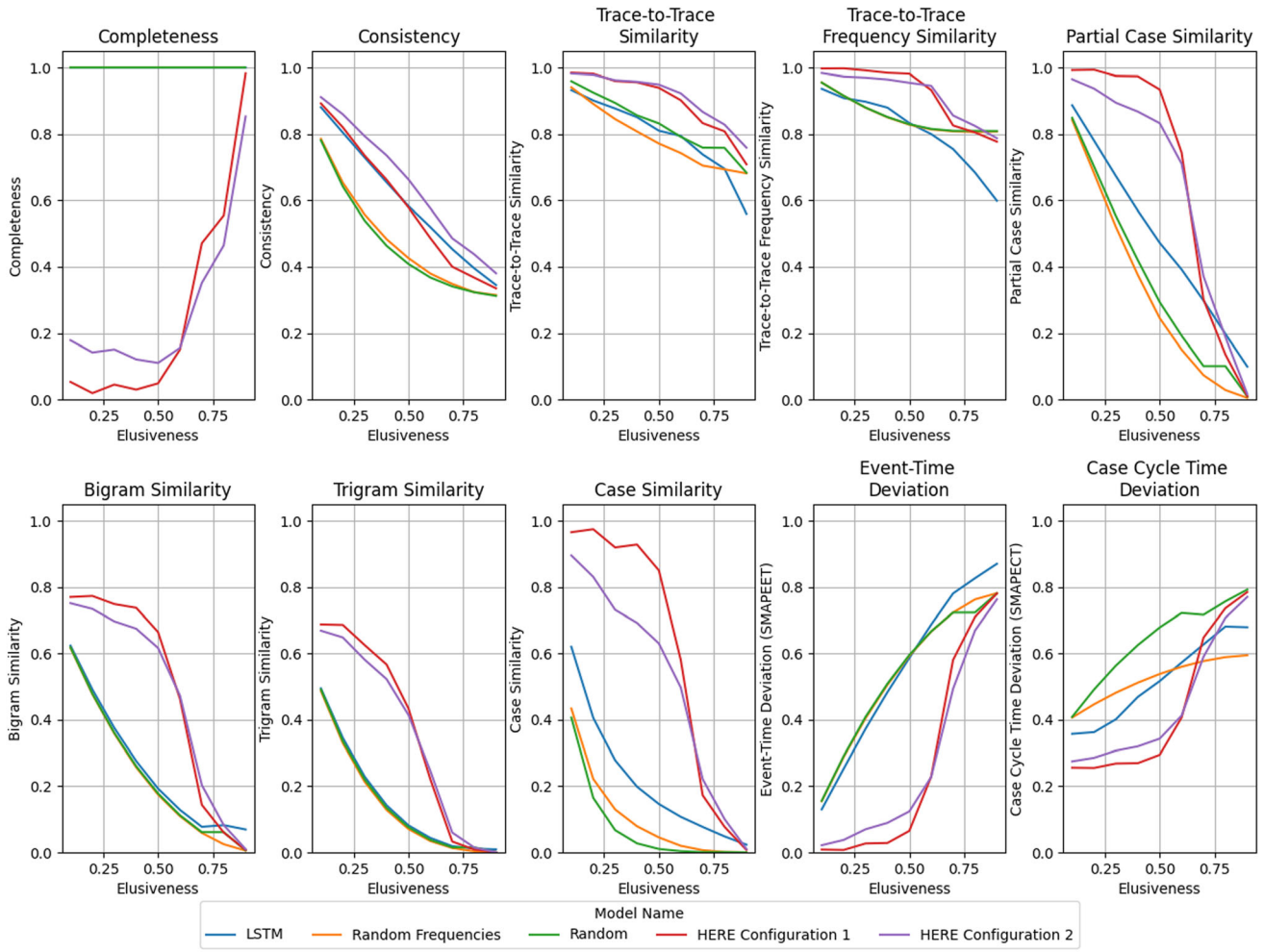

Fig. 7 Sensitivity analysis for the Hospital Billing log

$\sigma' \in S(L')$ as the cases in the ground truth and repaired logs, and $|I|$ as the total number of cases.

5.2.3 Evaluation Benchmarks

To conduct a comprehensive evaluation of our method, we benchmarked it against three alternative approaches: *Long Short-Term Memory Network (LSTM)*, *Random Frequencies*, and *Random*. Unfortunately, we were unable to benchmark against competing state-of-the-art methods from the literature either due to the unavailability of their prototypes or differing assumptions regarding additional data that we lack. For the *LSTM* benchmark, we employed an LSTM network as they are well-suited for sequential data (Hochreiter and Schmidhuber 1997) and are applied in various process mining activities (van Dun et al. 2023; Schmid et al. 2023). For this benchmark, we implemented a simple three-layer LSTM architecture, combined with dropout layers to prevent overfitting. The model predicts the most likely case ID for each event based on the sequence of preceding events. For more details on the *LSTM* benchmark implementation, we refer to our code repository. The *Random Frequencies* benchmark utilizes the frequency with which each case ID appears in the correct events of a log. Case IDs are assigned to events randomly, but their probabilities are weighted based on the observed frequency distribution in the correct data. The *Random* benchmark assigns case IDs completely at random, with each case ID being equally likely. Unlike the *Random Frequencies* benchmark, this approach does not consider the actual distribution of case IDs in the original data, providing a purely stochastic baseline.

5.2.4 Evaluation Results

Our evaluation revealed critical insights into the strengths and application areas of our method, along with its limitations. Most notably, our approach demonstrates strong performance in scenarios where sufficient training data is available, particularly at low levels of elusiveness (below

30%). For the Hospital Billing dataset, which is the largest and only real-life log in our evaluation, our method outperforms all other benchmarks by a significant margin, showcasing its potential for practical, real-world applications. However, the results also indicate a dependency on data volume, particularly at higher levels of elusiveness (above 80%). For instance, in the Renting event log, the smallest log in our data collection, a 90% elusiveness level leaves us with only approximately nine thousand events for training – a volume insufficient to learn meaningful patterns. This dependency aligns with the general understanding that machine learning methods, including our Transformer-based approach, require substantial amounts of data to train effectively. In Fig. 7 we illustrate this observation by means of a sensitivity analysis, showing a clear advantage in performance under favorable conditions. Consequently, the results highlight that when enough data is available, our method is highly effective and scalable to complex real-world settings.

Furthermore, the inclusion of rule checking as a fallback mechanism (Configuration 2) further enhances the robustness of our approach. Configuration 1 employs the standalone Transformer, with a maximum of 5 repair iterations, input variables such as DAs activity and resource, CA timestamp, three EAs (specifically UAs start activity, end activity, and BA directly following), and a 0% threshold. In contrast, Configuration 2 integrates the Transformer with the same input variables, but augments it with both ex-ante and ex-post rule checking. For the Hospital Billing dataset, the inclusion of rule checking as a fallback mechanism at higher levels of elusiveness significantly enhances performance. Conversely, at lower levels of elusiveness, rule checking tends to degrade performance slightly. The gap in performance is particularly pronounced at the beginning but narrows as elusiveness increases. In contrast, for the Renting and Review logs, where data volume remains consistently low, the difference between configurations is already negligible at lower levels of elusiveness. However, as elusiveness increases, Configuration 2 starts to outperform the standalone Transformer configuration, with this transition occurring at a much earlier stage than observed in the Hospital Billing log. This finding highlights that rule checking acts as a fallback mechanism, that helps especially under challenging conditions of high elusiveness in low-data scenarios.

In cases where enough training data is available, meaning low elusiveness or higher volume baseline logs, we demonstrated that our method is regularly more effective than the benchmarks. In smaller synthetic logs like Renting and Review, however, *HERE's* performance aligns more closely with the LSTM benchmark at lower elusiveness levels. In these two logs, we mostly observe either no substantial variation between both approaches or the LSTM performing better under certain conditions. This parity suggests that while our method holds promise, a consistent outperformance of established techniques like LSTM is not yet definitive. This further underscores the importance of data volume for the Transformer approach to deliver optimal results.

Apart from data volume, no other significant factors appeared to influence the effectiveness of our method. Metrics across the Renting and Review logs were generally consistent and lower than those observed for the Hospital Billing log. However, we acknowledge that log-specific characteristics may unfold an impact on the method's performance as soon as the bottleneck of data volume is resolved. We believe that data volume currently serves as the biggest constraint, and addressing this limitation could reveal nuanced effects of log-specific characteristics on the overall effectiveness of our approach. The detailed results for all benchmarks, logs and levels of elusiveness can be found in the online Appendix B. Online Appendices C and D provide the sensitivity analysis for the Renting and Review log.

Finally, while we were unable to benchmark our method directly against approaches from the literature, such as Bayomie et al. (2023), their reported performance provides a useful reference point. For instance, their method – designed specifically for scenarios with 100% elusiveness – consistently achieves similarity scores above 80% and time deviation scores below 40%. Based on this, we would generally recommend their approach for such logs, where no case IDs are present. However, their approach may be less suitable for cases involving partial elusiveness, where preserving existing case ID logic is critical. Such scenarios can, for instance, arise from temporary system outages. In these situations, preserving existing case ID logic and specific labels is crucial, making our method more appropriate. This is especially true when high volumes of training data are available, which allows our Transformer-based approach to realize its full potential. This understanding further emphasizes the need to match repair techniques with the specific characteristics and constraints of the data at hand.

5.3 Naturalistic Summative Evaluation

The final evaluation phase comprised a naturalistic summative evaluation to establish a proof of value. To achieve that, we conducted another round of semi-structured interviews with ten of the experts from the initial interviews, while preserving their IDs as specified in Table 2, and let them interact with the fully developed prototype to repair a rental process event log (Pohl and Berti 2023). A condensed description of the interview structure is provided in the online appendix (Appendix E). While the ideal

evaluation would have involved observing the approach repairing an event log with the elusive case pattern in real-time, the training time associated with our Transformer architecture made a fully naturalistic application of the method infeasible within the limited time frame of ten interviews. Instead, we decided to repair the event log with 10% elusiveness multiple times in advance, considering all possible choices of our method that could affect the final outcome. Since our method takes an iterative approach in which the stakeholder can gradually add domain knowledge, this would result in an exponentially growing space of possible outcomes. We therefore decided to sensibly limit the selection options to encourage stakeholders to interact freely while seeing immediate results. However, this decision may influence interviewees' evaluations, as the restricted options might result in biased responses that do not fully reflect the range of stakeholders' experiences and perceptions. Nonetheless, this approach encompasses 44 distinct repair outcomes. This setup creates a semi-realistic environment for assessing the method, where real users engage in real tasks within a simulated system that approximates practical constraints. The prototype used for the interaction can be found in the code repository.

After interacting with the prototype and repairing the event log, we asked the interviewees to rate the artifact using an end-labeled unipolar seven-point Likert scale (Höhne et al. 2021) across four criteria suitable for an ex-post evaluation (Sonnenberg and vom Brocke 2012). We chose *usefulness* and *ease of use* to predict the user acceptance of the artifact (Davis 1989), as these criteria are critical for ensuring that the artifact meets user needs and can be adopted with minimal resistance. *Generality* was included to assess whether the artifact's scope is broad enough or if there are specific scenarios where it may not be suitable, ensuring the artifact's adaptability across various contexts. *Applicability* was chosen to confirm the artifact's practicality in real-world settings, highlighting its relevance and potential impact (Sonnenberg and vom Brocke 2012). Each criterion was presented to the interviewees using a definition and a set of guiding questions, as detailed in Table 10. For each criterion, we asked the interviewees to justify their assessment. Afterwards, we gave the interviewees the opportunity to add more feedback in case the criteria did not meet all aspects of a comprehensive assessment. The results of this evaluation stage are depicted in Fig. 8.

The results indicate overall positive feedback regarding the usefulness of the artifact. Almost all interviewees acknowledged that the artifact effectively addresses the problem of elusive cases, making it beneficial for stakeholders working with affected event log data. However, there were concerns about the quality of the results. For instance, I4 stated, *"It definitely helps because otherwise I could not work with the event logs and cleaning the data by hand is not feasible [...] However, what makes me a bit puzzled is the accuracy of the model."* Similarly, I5 emphasized the need for accuracy, stating, *"Accuracy should be at least above 60 percent."* Furthermore, I1 pointed out that the main issue is not accuracy itself but the validation of accuracy in real-life settings, as there is no ground truth data available to verify the repaired event log. I1 hence suggests to *"understand it as a problem from visual analytics. So, generate data and see what the results look like."* Implementing visual data validation techniques from classic data science and allowing domain experts to interact with the tool and evaluate the repaired process model could enhance the artifact's effectiveness.

Regarding ease of use, the feedback was mostly positive. Interviewees highlighted the advantages of a graphical user interface and a simplified code display that abstracts complexity. Many appreciated the extensive opportunities for stakeholder interaction and the incorporation of domain knowledge. However, concerns were raised about the artifact's ease of use depending on the specific target group. For instance, a data engineer might prefer more control over technical details, whereas a process owner might need an even simpler frontend. I8 commented, *"There are two target groups. One is the data engineering*

Table 10 Naturalistic summative criteria definitions and guiding questions

Criterion	Definition	Guiding Questions
Usefulness	The extent to which you perceive that the artifact is useful in addressing the problem.	How well do you think the artifact will improve working with elusive cases, and what benefits do you expect it to provide?
Ease of Use	The extent to which you perceive that using the artifact will be free of effort.	How user-friendly is the artifact? What features contribute to its ease of use? Are there any entry barriers?
Applicability	The extent to which you perceive the artifact instance as suitable for solving the intended problem.	How suitable is the artifact for solving the problem, and what challenges might arise in its application?
Generality	The extent to which you perceive the artifact instance as applicable across different contexts or domains.	How versatile is the artifact across various contexts, and what limits its applicability in different settings?

Fig. 8 Average Likert scale ratings for naturalistic summative evaluation criteria

team, for whom it will be easy to use. But the specialist department should not be fiddling around with data models. The frontend with the buttons is the maximum for this, but nothing in the backend." Similarly, I9 noted, *"This task is more likely to be undertaken by the person responsible for creating event logs. The data engineer therefore needs to interact with the prototype and the prototype might be too easy for him as he likes to touch the data."* Additionally, some interviewees mentioned that the method's complexity might require more user guidance. For non-technical stakeholders, I7 suggested explaining each decision point in simple terms and adding tooltips to make the method more self-explanatory.

Applicability received the lowest scores, though still highly positive overall. Interviewees raised concerns about computing times and resource utilization. While some emphasized the need for near-real-time interaction, others, like I6, found it acceptable if training could be completed overnight or within a working day: *"[...] uploading my data alone takes more than 1.5 hours. So you can simply let the model run overnight, provided you have a certain amount of confidence in the output."* I10 mentioned concerns about the trade-off between accuracy and completeness and suggested using heuristics to address low completeness: *"It's not important to somehow find a process model or something that really solves everything perfectly, but instead in finding a solution, simply making some kind of suggestion or using a heuristic for the remaining missing cases."* Additionally, there were doubts about the artifact's performance in scenarios with other types of errors, such as incorrect activity labels or timestamps. Applicability was also found to depend on the specific process and use case. I2 highlighted the potential value in high-stakes environments like hospitals: *"This can be worth a lot for hospitals. If you enable having a billing event log for a single hospital, you can enrich the whole chain of hospitals."* I2 also noted that using the artifact in such an environment could be more practical than manually fixing underlying legacy systems. Nonetheless, I2 emphasized that our method is a workaround and that ideally, the way of logging the events should be enhanced.

Regarding generality, all interviewees agreed that the method could be applied as long as an XES style event log is available. I8 commented that a *"reasonably relational data basis"* is typically sufficient and commonly available. However, some concerns were raised about varying performance across different event logs. I3 noted that *"Generality is questionable not because of industry or domain, but rather because of the characteristics of the process such as the number of variants."* Similarly, I1 acknowledged that more complex processes are harder to repair but suggested that *"room for specific data preprocessing paths"* could mitigate these issues. Allowing stakeholders to aggregate activity labels and incorporating data science techniques like undersampling or synthetic oversampling for rarer traces could improve the model's performance and better handle complex processes.

6 Discussion

Our objective was to design and develop a method to repair the elusive case imperfection pattern. To this end, we utilized generative AI, specifically the Transformer architecture, and incorporated a rule-based approach within a human-in-the-loop framework. This method was instantiated as a software prototype and evaluated by 21 expert interviews as well as by repairing three distinct event logs with elusiveness levels ranging from 10% to 90%. Our findings indicate that the method is able to reconstruct case IDs across different process contexts and complexities with the inclusion of domain knowledge (DO 1), acting as a fallback mechanism, typically enhancing its performance when low volumes of data are available. Thereby, human expertise can help in explicitly integrating contextual information, while rule checks facilitate the assignment of events to cases. This process allows for automated validation, lessening the effort for stakeholders to verify the output. Nonetheless, stakeholders are advised to conduct a final verification of the output before initiating process mining analyses. Utilizing output confidence values can assist in this validation process, allowing filtering for values falling below a specified threshold, thereby reducing subsequent manual efforts.

Our method entails some practical considerations. First, stakeholders face a trade-off between the two data quality

dimensions accuracy and completeness. Higher threshold confidence levels in the Transformer typically improve accuracy but reduce the number of identified case IDs. Conversely, prioritizing completeness allows for basic process discovery, even if some data points are less precise. This characteristic of our approach allows to prioritize certain data quality dimensions over others and, hence, enables an individual adjustment for the contextual factors of organizational application. Second, the output's quality relies on several factors. Our approach was designed under the assumption that the data is of high quality, with the exception of elusive cases. We acknowledge that this assumption does not always hold true in practice and necessitates specific data preprocessing to address various data quality issues beforehand. Nevertheless, our approach is intended to be one link in a chain of methods aimed at data quality improvement. The combined effect of these methods is expected to enable effective data quality management. Additionally, directly measuring the accuracy of the output is impractical in real-life applications where ground truth is unavailable. Thus, the results should be considered in combination with the results of process discovery, enabling a practical evaluation of their validity. Third, our findings indicate that our artifact produces best results when applied to large data volumes with low levels of elusiveness. As such, real-life application should take into account the limitations of the Transformer model when working with smaller datasets. For cases with limited data, alternative methods may offer superior performance, as the Transformer's efficacy increases with larger datasets.

From a theoretical standpoint, our method addresses a limitation of traditional repair approaches: the underutilization of event log data and complete regrouping of events. Thereby, our method can operate effectively on correct samples of data while keeping the existing case ID logic. The flexibility of our method ensures its applicability to a wide range of event log attributes and expert knowledge, enhancing its overall utility. This adaptability is particularly important in the increasingly prevalent object-centric paradigm, where processes involve multiple interacting entities rather than a single-case assumption. While transitioning to an object-centric paradigm necessitates method modifications, it is common for object IDs to be missing or incomplete. Furthermore, our approach demonstrates that generative AI can be extended beyond natural language processing to address event log data quality issues. Lastly, our results indicate that integrating a human-in-the-loop with generative AI leads to more promising outcomes. To the best of our knowledge, this is the first study to employ the Transformer architecture for linking events to process instances, highlighting the efficacy of generative AI in this domain.

Our work also entails some limitations. First, the utilization of a supervised learning framework leads to a situation where our model can only predict outcomes for case IDs included in the training dataset. Consequently, if the erroneous log includes IDs absent from the training data, new events cannot be associated with these cases. Similarly, the events occurring during repair cannot be attributed to more cases than those included in the training set. Nonetheless, as long as the sequence of events is accurately anticipated, our artifact remains effective. Second, a case is defined both by the training dataset and by expert knowledge specifications. In practice, however, case boundaries may be more fluid. For instance, actions within a buffer zone may still be considered part of the corresponding case after the actual end activity, whereas actions beyond this buffer may initiate a new case. Nevertheless, as such scenarios could also be accommodated using corresponding rules, we treated the boundaries of a case as fixed in our approach. Lastly, we have not considered the computing costs involved. In practical applications, these costs are significant factors influencing the feasibility of implementing the approach. Therefore, while our approach may show promise in theoretical contexts, its real-world applicability also relies on its computational efficiency. Nevertheless, we assert that the potential advantages of enabling process mining analyses outweigh the associated expenses.

Based on our findings, several promising avenues for future research can enhance and extend our approach. Specifically, focusing on unsupervised learning and integrating diverse data sources presents significant opportunities. First, unsupervised learning can uncover patterns without labeled data, which is advantageous when no correct data is available. Reformulating our machine learning problem as a clustering task and exploring modified Transformer architectures could therefore enhance applicability. This holds especially true in real-life applications where no correct data is available for training. Second, expanding the Transformer's input to include various data sources, such as process documentation or models can provide a more comprehensive view of events. This approach can uncover contextual information not present in event logs and domain knowledge alone. Evaluating the impact of each data type and identifying beneficial as well as adversarial effects will be crucial.

7 Conclusion

Process mining relies on high-quality event logs to provide accurate and reliable insights into business processes. A significant issue affecting event log quality is the elusive case imperfection pattern, where events lack case IDs, making it difficult to link them to specific process

instances. Since many process mining techniques rely on this linking, elusive cases undermine the effectiveness of process mining techniques, leading to inaccurate or incomplete analyses. To tackle this issue, we explored how generative AI can repair the elusive case imperfection pattern. We developed *HERE*, a method that combines a traditional rule-based approach with the Transformer architecture, enriched by domain expertise within a human-in-the-loop framework. Employing the DSR paradigm, we iteratively designed *HERE*, evaluated it through 21 expert interviews, and demonstrated its effectiveness by repairing a total of three event logs with elusiveness levels ranging from 10% to 90%. Our results demonstrate that *HERE* can effectively determine case IDs and outperform benchmark approaches, especially at lower levels of elusiveness, provided that a large data volume is available. To the best of our knowledge, we are the first to leverage generative AI for this specific data quality issue, a claim supported by expert assessments during our evaluation. Our contribution lies in providing an alternative solution to current methods for addressing the elusive case imperfection pattern by leveraging the extensive data available in the event log while keeping the existing case ID logic. Depending on context and requirements, our hybrid approach offers a balance between accuracy and completeness of event data, thus enabling process mining analyses. Additionally, we provide the instantiated method as a software prototype that can be utilized and further developed to address similar data quality challenges.

While our method successfully addresses the elusive case, it has certain limitations. These include its design as a supervised learning approach, the associated costs, and the definition of cases with fixed boundaries. Hence, there are several avenues for future research. For instance, exploring the suitability and effectiveness of alternative structures, such as unsupervised learning, for solving this type of data quality issue could be beneficial. Beyond event logs in XES format, it would be worthwhile to investigate how our approach can handle other data sources.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s12599-025-00935-5.

Acknowledgements We gratefully acknowledge the StMWi (Bavarian Ministry of Economic Affairs, Regional Development and Energy) for their support of the project "QUAPRO (DIK-2209-0017// DIK0440/02)" that made this paper possible. This study received ethical approval from QUT HREC – approval number 8060.

References

Andrews R, Emamjome F, ter Hofstede AH, Reijers HA (2022) Root-cause analysis of process-data quality problems. J Bus Anal 5(1):51–75

Andrews R, Suriadi S, Ouyang C, Poppe E (2018) Towards event log querying for data quality: Let's start with detecting log imperfections. In: Panetto H, Debruyne C, Proper HA, Ardagna CA, Roman D, Meersman R (eds) On the move to meaningful internet systems. Proceedings OTM 2018 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2018, Valletta, Part I. Springer, Cham, pp 116–134. https://doi.org/10.1007/978-3-030-02610-3_7

Badakhshan P, Wurm B, Grisold T, Geyer-Klingeberg J, Mendling J, vom Brocke J (2022) Creating business value with process mining. J Strateg Inf Syst 31(4):101,745

Banh L, Strobel G (2023) Generative artificial intelligence. Electron Mark 33:63

Batini C, Scannapieco M (2016) Data and information quality: dimensions, principles and techniques. Data-centric systems and applications. Springer, Cham

Bayomie D, Awad A, Ezat E (2016a) Correlating unlabeled events from cyclic business processes execution. In: Nurcan S, Soffer P, Bajec M, Eder J (eds) Advanced Information Systems Engineering: Proceedings 28th International Conference, CAiSE 2016, Ljubljana. Springer International Publishing, Cham, pp 274–289. https://doi.org/10.1007/978-3-319-39696-5_17

Bayomie D, Di Ciccio C, Mendling J (2023) Event-case correlation for process mining using probabilistic optimization. Inf Syst 114:102,167

Bayomie D, Di Ciccio C, La Rosa M, Mendling J (2019) A probabilistic approach to event-case correlation for process mining. In: Laender AHF, Pernici B, Lim E-P, de Oliveira JPM (eds) Conceptual modeling: Proceedings 38th International Conference, ER 2019, Salvador. Springer, Cham, pp 136–152. https://doi.org/10.1007/978-3-030-33223-5_12

Bayomie D, Helal IMA, Awad A, Ezat E, ElBastawissi A (2016b) Deducing case IDs for unlabeled event logs. In: Reichert M, Reijers HA (eds) Business Process Management Workshops. Springer, Cham, pp 242–254. https://doi.org/10.1007/978-3-319-42887-1_20

Bayomie D, Revoredo K, Di Ciccio C, Mendling J (2022) Improving accuracy and explainability in event-case correlation via rule mining. In: 4th International Conference on Process Mining, Bolzano

Beerepoot I, Di Ciccio C, Reijers HA, Rinderle-Ma S, Bandara W, Burattin A, Calvanese D et al (2023) The biggest business process management problems to solve before we die. Comput Ind 146:103,837

Beheshti A, Yang J, Sheng QZ, Benatallah B, Casati F, Dustdar S, Nezhad HRM, Zhang X, Xue S (2023) ProcessGPT: Transforming business process management with generative artificial intelligence. In: 2023 IEEE International Conference on Web Services, Chicago

Bukhsh ZA, Saeed A, Dijkman RM (2021) ProcessTransformer: predictive business process monitoring with transformer network. arXiv:2104.00721, accessed 29 May 2024

Burattin A, Vigo R (2011) A framework for semi-automated process instance discovery from decorative attributes. In: 2011 IEEE Symposium on Computational Intelligence and Data Mining, Paris

Busch K, Rochlitzer A, Sola D, Leopold H (2023) Just tell me: prompt engineering in business process management. In: 24th International Conference, BPMDS 2023, and 28th International Conference, EMMSAD 2023, Springer, Cham

Chen T, Han L, Demartini G, Indulska M, Sadiq S (2020) Building data curation processes with crowd intelligence. In: CAiSE Forum 2020, Springer, Cham

De Weerdt J, Wynn MT (2022) Foundations of process event data. In: van der Aalst WMP, Carmona J (eds) Process Mining Handbook. Springer, Cham, pp 193–211

Di Ciccio C, Montali M (2022) Declarative process specifications: reasoning, discovery, monitoring. In: van der Aalst WMP, Carmona J (eds) Process Mining Handbook. Springer, Cham, pp 108–152

Davis FD (1989) Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Q 13(3):319–340

De Fazio R, Balzanella A, Marrone S, Marulli F, Verde L, Reccia V, Valletta P (2024) CaseID detection for process mining: a heuristic-based methodology. In: ICPM 2023 International Workshops, Springer, Cham

van Dun C, Moder L, Kratsch W, Röglinger M (2023) ProcessGAN: supporting the creation of business process improvement ideas through generative machine learning. Decis Support Syst 165:113,880

Evron Y, Soffer P, Zamansky A (2022) Model-based analysis of data inaccuracy awareness in business processes. Bus Inf Syst Eng 64(2):183–200

Ferreira DR, Gillblad D (2009) Discovering process models from unlabelled event logs. In: 7th International Conference, BPM 2009, Springer, Heidelberg

Feuerriegel S, Hartmann J, Janiesch C, Zschech P (2024) Generative AI. Bus Inf Syst Eng 66(1):111–126

Fischer DA, Goel K, Andrews R, van Dun CG, Wynn MT, Röglinger M (2022) Towards interactive event log forensics: detecting and quantifying timestamp imperfections. Inf Syst 109:102,039

Galic G, Wolf M (2021) Delivering value with process analytics: process mining adoption and success factors. https://www2.deloitte.com/de/de/pages/finance/articles/global-process-mining-survey-2021.html, accessed 3 June 2024

Gregor S, Hevner AR (2013) Positioning and presenting design science research for maximum impact. MIS Q 37(2):337–355

Grisold T, Mendling J, Otto M, vom Brocke J (2021) Adoption, use and management of process mining in practice. Bus Process Manag J 27(2):369–387

Hevner M, Park R (2004) Design science in information systems research. MIS Q 28(1):75–105

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Hoffmann M, Malburg L, Bergmann R (2022) ProGAN: toward a framework for process monitoring and flexibility by change via generative adversarial networks. In: BPM 2021 International Workshops, Springer, Cham

Hofmann P, Rückel T, Urbach N (2021) Innovating with artificial intelligence: capturing the constructive functional capabilities of deep generative learning. In: 54th Annual Hawaii International Conference on System Sciences, Virtual

Höhne JK, Krebs D, Kühnel SM (2021) Measurement properties of completely and end labeled unipolar and bipolar scales in likert-type questions on income (in)equality. Soc Sci Res 97:102,544

Janiesch C, Zschech P, Heinrich K (2021) Machine learning and deep learning. Electron Mark 31:685–695

Kerremans I, Kerremans M (2023) The impact of generative AI on process mining. https://www.gartner.com/doc/reprints?id=1-2F77X9LO&ct=231003 &st=sb, accessed 2 May 2024

Mannhardt F (2017) Hospital billing - event log. https://data.4tu.nl/articles/_/12705113/1, accessed 4 June 2024

March ST, Smith GF (1995) Design and natural science research on information technology. Decis Support Syst 15(4):251–266

March ST, Storey VC (2008) Design science in the information systems discipline: an introduction to the special issue on design science research. MIS Q 32(4):725–730

Martin N, van Houdt G, Janssenswillen G (2022) DaQAPO: supporting flexible and fine-grained event log quality assessment. Expert Syst Appl 191:116,274

Mosqueira-Rey E, Hernández-Pereira E, Alonso-Ríos D, Bobes-Bascarán J, Fernández-Leal Á (2023) Human-in-the-loop machine learning: a state of the art. Artif Intell Rev 56:3005–3054

Myers MD, Newman M (2007) The qualitative interview in is research: examining the craft. Inf Organ 17(1):2–26

Nguyen HTC, Lee S, Kim J, Ko J, Comuzzi M (2019) Autoencoders for improving quality of process event logs. Expert Syst Appl 131:132–147

Padmanabhan B, Fang X, Sahoo N, Burton-Jones A (2022) Machine learning in information systems research. MIS Q 46(1):iii–xviii

Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S (2007) A design science research methodology for information systems research. J Manag Inf Syst 24(3):45–77

Peffers K, Rothenberger M, Tuunanen T, Vaezi R (2012) Design science research evaluation. In: 7th International Conference, DESRIST 2012, Springer, Heidelberg

Pegoraro M, Uysal MS, Hülsmann TH, van der Aalst WMP (2022) Uncertain case identifiers in process mining: a user study of the event-case correlation problem on click data. In: 23rd International Conference, BPMDS 2022 and 27th International Conference, EMMSAD 2022, Springer, Cham

Pesic M, Schonenberg H, van der Aalst WM (2007) DECLARE: full support for loosely-structured processes. In: 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), Annapolis

Pohl T, Berti A (2023) (Un)Fair Process Mining Event Logs. https://zenodo.org/records/8059489, accessed 4 June 2024

Pourmirza S, Dijkman R, Grefen P (2017) Correlation miner: mining business process models and event correlations without case identifiers. Int J Coop Inf Syst 26(2):1742,002

Raisch S, Fomina K (2024) Combining human and artificial intelligence: hybrid problem-solving in organizations. Acad Manag Rev. https://doi.org/10.5465/amr.2021.0421

Reinkemeyer L, Röglinger M, Kratsch W, Fabri L, Schmid S, Wittmann J (2023) Exploring the interplay of process mining and generative AI: research and recommendations for CoEs. https://www.celonis.com/report/fraunhofer-study/genai/download/, accessed 2 May 2024

Rinderle-Ma S, Stertz F, Mangler J, Pauker F (2023) Process mining - discovery, conformance, and enhancement of manufacturing processes. In: Vogel-Heuser B, Wimmer M (eds) Digital Transformation. Springer, Heidelberg, pp 363–383

Rivera Lazo G, Ñanculef R (2022) Multi-attribute transformers for sequence prediction in business process management. In: Discovery Science: 25th International Conference, DS 2022, Springer, Cham

Robinson OC (2014) Sampling in interview-based qualitative research: a theoretical and practical guide. Qual Res Psychol 11(1):25–41

Sadeghianasl S, ter Hofstede AH, Wynn MT, Türkay S (2024) Humans-in-the-loop: gamifying activity label repair in process event logs. Eng Appl Artif Intell 132:107,875

Sadeghianasl S, ter Hofstede AHM, Wynn MT, Suriadi S (2019) A contextual approach to detecting synonymous and polluted activity labels in process event logs. In: On the Move to Meaningful Internet Systems: OTM 2019 Conferences, Springer, Cham

Sadeghianasl S, ter Hofstede AH, Suriadi S, Turkay S (2020) Collaborative and interactive detection and repair of activity labels in process event logs. In: 2020 2nd International Conference on Process Mining, Virtual

Schmid SJ, Moder L, Hofmann P, Röglinger M (2023) Everything at the proper time: repairing identical timestamp errors in event logs with generative adversarial networks. Inf Syst 118:102,246

Sonnenberg C, vom Brocke J (2012) Evaluations in the science of the artificial – reconsidering the build-evaluate pattern in design science research. In: 7th International Conference, DESRIST 2012, Springer, Heidelberg

Suriadi S, Andrews R, ter Hofstede AH, Wynn MT (2017) Event log imperfection patterns for process mining: towards a systematic approach to cleaning event logs. Inf Syst 64:132–150

Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: 27th International Conference on Neural Information Processing Systems, Montreal

Tajima K, Du B, Narusue Y, Saito S, Iimura Y, Morikawa H (2023) Step-by-step case ID identification based on activity connection for cross-organizational process mining. IEEE Access 11:60,578-60,589

Taymouri F, La Rosa M, Erfani S, Bozorgi ZD, Verenich I (2020) Predictive business process monitoring via generative adversarial nets: the case of next event prediction. In: 18th International Conference, BPM 2020, Springer, Cham

Tuunanen T, Winter R, vom Brocke J (2024) Dealing with complexity in design science research: a methodology using design echelons. MIS Q 48(2):427–458

van der Aalst W (2010) Synthetic event logs – review example large.xes.gz. https://data.4tu.nl/articles/_/12716609/1, accessed 4 June 2024

van der Aalst W, Adriansyah A, de Medeiros AKA, Arcieri F, Baier T, Blickle T, Bose JC, et al (2012) Process mining manifesto. In: BPM 2011 International Workshops, Springer, Heidelberg

van der Aalst W (2016a) Analyzing "spaghetti processes''. In: van der Aalst W (ed) Process Mining. Springer, Heidelberg, pp 411–427

van der Aalst W (2016b) Getting the data. In: van der Aalst W (ed) Process Mining. Springer, Heidelberg, pp 125–162

van der Aalst W (2016c) Process mining: the missing link. In: van der Aalst W (ed) Process Mining. Springer, Heidelberg, pp 25–52

van der Aalst WMP (2022) Process mining: a 360 degree overview. In: van der Aalst WMP, Carmona J (eds) Process Mining Handbook. Springer, Cham, pp 3–34

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. arXiv: 1706.03762, accessed 3 May 2024

Venable J, Pries-Heje J, Baskerville R (2016) FEDS: a framework for evaluation in design science research. Eur J Inf Syst 25(1):77–89

Weinzierl S, Zilker S, Dunzer S, Matzner M (2024) Machine learning in business process management: a systematic literature review. Expert Syst Appl 253:124,181

Wu P, Fang X, Fang H, Gong Z, Kan D (2024) An event log repair method based on masked transformer model. Appl Artif Intell 38(1):2346,059

Wynn MT, Lebherz J, van der Aalst WMP, Accorsi R, Di Ciccio C, Jayarathna L, Verbeek HMW (2022) Rethinking the input for process mining: insights from the XES survey and workshop. In: ICPM 2021 International Workshops, Springer, Cham