IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 January 2025, accepted 2 February 2025, date of publication 21 February 2025, date of current version 27 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3544268

RESEARCH ARTICLE

Multi-Graph Anomaly Detection in Business Processes With Scalable Neural Architectures

STANLEY HSU[1], EGE GÜLCE[1], TEOMAN BERKAY AYAZ[1], ALPER OZCAN[2], AND AKHAN AKBULUT[1,3], (Member, IEEE)

[1]Research and Development Center, Next4biz, 34734 Istanbul, Türkiye
[2]Department of Computer Engineering, Akdeniz University, 07070 Antalya, Türkiye
[3]Department of Computer Engineering, Istanbul Kültür University, 34452 Istanbul, Türkiye

Corresponding author: Akhan Akbulut (a.akbulut@iku.edu.tr)

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under Grant TEYDEB-3231136.

ABSTRACT Business Process Management (BPM) solutions are critical for organizational efficiency, but their potential remains limited by inadequate effectiveness in anomaly detection capabilities for real-world deployment. This study addresses key challenges in developing production-ready anomaly detection systems that are scalable, efficient, and adaptable across diverse business domains. We propose several enhancements to a state-of-the-art graph-based autoencoder model to overcome these barriers. This includes improved artificial anomaly injection methods that more accurately reflect real-world scenarios to overcome the scarcity of annotated datasets in real-world environments. A comprehensive study of multiple model architectures is conducted, incorporating Graph Attention v2 in the encoder and replacing Gated Recurrent Unit (GRU) decoders with Transformers, thereby achieving comparable or superior performance with half the computational cost. Introducing a denoising objective alongside reconstruction, we lay the foundation for targeted training on domain-specific anomalies without compromising general detection capabilities. We demonstrate the solution's reliability and generalizability in varied business domains by conducting comprehensive evaluations on diverse public and private datasets. The results indicate significant improvements in scalability and real-world applicability while maintaining and enhancing detection accuracy, with results showing up to 22% increase in anomaly detection performance.

INDEX TERMS Anomaly detection, attention mechanisms, business process management, graph neural networks, transformers, gated recurrent unit.

I. INTRODUCTION

Business Process Management (BPM) solutions are software frameworks that enable businesses to automate repetitive tasks, manage operations, monitor workflows, and optimize their processes. BPM solutions enhance operational efficiency and reduce costs by facilitating operational design, implementation, and management [1]. Originating from information technology and managerial sciences, BPM has garnered significant research attention for its role in boosting organizational profitability and growth. Despite that, underdeveloped or ineffective BPM solutions can disrupt operations, decrease productivity, and increase costs, negatively impacting efficiency, profitability, and reputation. In today's competitive and fast-paced business environment, reliable BPM solutions are critical for success - allowing employees to focus on higher-value activities. Consequently, organizations are incentivized to adopt advanced BPM solutions to increase profitability, support continuous growth, and identify operational improvement and optimization opportunities.

Given the automated nature of BPM solutions, it generates vast amounts of business process event logs. These logs contain a series of timestamped events accompanied by relevant information, such as the activity or task enacted and the entity (personnel, system, etc.) that performed the action. These detailed records of an organization are potent assets for building analytical tools to facilitate growth

The associate editor coordinating the review of this manuscript and approving it for publication was Yassine Maleh.

and mitigate risks. Naturally, they contain many processes containing irregularities that require further investigation and correction [2]. In business domains, minor irregularities in the records can often be symptoms of more serious complications, ranging in severity from inefficiencies to fraudulent activities. Therefore, an effective and scalable anomaly detection and prevention solution is essential to promote organizational efficiency and safeguard interests. In BPM, an anomaly is often defined as a deviation from the standard in a business process workflow [3]. These deviations can reveal efficiency bottlenecks during workflow design by highlighting how and when the processes fail to operate as intended. Moreover, by analyzing the automated records, anomaly detection solutions can help organizations identify and address malicious behavior by members, dealers, or partners. It can also play an important role in ensuring an organization's full compliance with regulatory frameworks. As violations are rare, anomaly detection systems can flag potential violations early and notify organizations to take proactive measures to prevent non-compliance. Given the numerous benefits, sophisticated anomaly detection systems remain an important research subject.

There are numerous challenges in developing production-ready business process anomaly detection systems. Due to the sensitive nature of business process logs, publicly available high-quality datasets are scarce [4], and private datasets are rarely shared with third parties to safeguard client privacy and regulations such as EU-GDPR. Another major challenge is the absence of annotated anomalies in publicly available event logs [5]. Analyzing large quantities of automated logs is labor and resource-intensive, often restricting the analysis to specific business domains not applicable to general processes. Consequently, to evaluate the effectiveness of detection models, artificial injection of anomalies is a popular solution to circumvent the difficulties in obtaining quality datasets that are generally applicable to a wide variety of processes [6]. Furthermore, like any pattern recognition system, anomaly detection is vulnerable to unseen types of malicious activity. Similar to cybersecurity, malicious behaviors in business processes often evolve and overcome pattern recognition systems with time; therefore, its ability to generalize beyond the training data remains an active area of research.

Production-ready business process anomaly detection systems must meet several key criteria. As a prevention mechanism, the model must be able to precisely diagnose the locations and reasons anomalies occur within a workflow in real-time, making predictions on incomplete cases to help organizations prevent potential mishaps in time. Secondly, the model must be reliable in detecting various anomalies across different business domains. This requires the model to have high recall and precision, as false negatives represent missed anomalies that could pose significant risks, and false positives may cause organizations to waste resources investigating non-anomalous cases. Thirdly, the solution should be adaptable to a diverse range of business domains, assisting organizations in identifying anomalies that are more common in their industry. This necessitates that the model maintains a strong baseline performance across all domains while also being capable of adapting to anomalies specific to each domain. Consequently, a post-training policy on explicitly defined domain-specific anomalies that seamlessly integrates the original unsupervised training objective with later targeted training on known anomalies must be developed, enabling versatile customization without compromising the model's general capabilities. Lastly, the architecture must be efficient in training, serving, and providing a low-cost solution that can rapidly scale to thousands of clients, each with numerous and complex workflows.

The ultimate goal and motivation of our study can be described as developing a production-ready business process anomaly detection solution that is both scalable and reliable enough for real-world use. Our study investigates methods in which a more reliable, scalable, efficient, and domain-specific anomaly detection system can be developed from the GAMA architecture, a multi-graph unsupervised learning approach proposed by Guan et al. [7], by testing different training objectives, architectural variations, and anomaly evaluation approaches. To summarize the key contributions of our study,

- We refined anomaly injection methods to reflect real-world scenarios better, enabling more applicable and comprehensive model evaluations.
- We enhanced a state-of-the-art business process anomaly detection solution, reducing computational costs by up to 50% while maintaining or improving detection accuracy.
- We introduced a new training objective for adaptable anomaly detection across diverse business domains.
- We demonstrated the superiority of our approach through extensive testing on multiple datasets, achieving improvements of up to 21.9% in trace-level and 14.7% in attribute-level F1 scores compared to previous state-of-the-art models. This showcases the robustness and reliability of our enhancements.

The remainder of this paper is structured as follows: Related Works introduces to the reader the relevant research done within the business process anomaly detection domain, in addition to introducing them to some preliminaries. The methodology section will elaborate on the leveraged approach in detail, answering any questions the reader might have about the implementation of our method. The Experimental Results section will present the empirical results of our study, presenting the reader with the yieldings of our proposed system. Lastly, the Conclusion section will conclude our study by presenting an overview of our research and discussing possible future research directions.

II. RELATED WORKS

In recent years, despite anomaly detection in business processes being a niche subject in machine learning,

TABLE 1. Sample event logs.

TID	EID	Time	Activity	UserID
t7	e1	2024-09-12 09:12:36	Process Start	918
t7	e2	2024-09-12 09:16:47	Load Data	16
t7	e3	2024-09-12 09:22:12	Data Processing	212
t7	e4	2024-09-12 10:07:11	Train Models	47
t7	e5	2024-09-12 14:18:12	Validate Results	47
t7	e6	2024-09-12 14:21:21	Process Complete	16

there have been significant advancements in deep learning-based approaches. These advancements include the application of sequential models through Natural Language Processing (NLP) techniques to analyze business process event logs and the use of Graph Neural Networks (GNNs) to represent event logs as a graph structure, thereby capturing complex interdependencies within workflows. In this section, we briefly introduce some recent advancements in this research area.

A. PRELIMINARIES OF BUSINESS PROCESS ANOMALY DETECTION

Business process event logs are organized hierarchically into three distinct levels: trace (case), event, and attribute level, with each level serving as a sub-component of the one above. Table 1 provides an example of a trace within an event log, where each trace has a unique case ID and consists of a sequence of events. Each event contains attributes relevant to its occurrence, including but not limited to timestamps, activity, and user ID. The hierarchical structure allows for numerous classification methods for anomalies. Typically, anomalies are detected at the event level and then propagated upwards, such that a trace is considered anomalous if it contains an anomalous event. More advanced methods enable anomaly detection at the attribute level. However, the finer granularity of attribute-level detection and the challenges with developing detection policies suitable for a wide variety of attribute types make it an ongoing area of research. In addition to the granularity, approaches that detect anomalies across multiple aspects or viewpoints of a process are known as multi-perspective methods. The two most commonplace perspectives are the data perspective and the control flow perspective. Methods focusing on the data perspective identify abnormal systems or user-generated data, while the control flow perspective emphasizes the order of events in a workflow [7].

B. PREVIOUS STUDIES

In 2016, Nolle et al. [8] conducted a study utilizing denoising autoencoder (AE) architecture designed for capturing the workflow's underlying structure from noisy data. The dimensions of the latent representation are smaller than the input size, which is a means to build a robust internal model through dimension compression by optimizing reconstruction error obtained as Mean Squared Error (MSE) between the original input and output prediction. To refine the proposed method and improve its usability, trace-level reconstruction error is used alongside the event-wise reconstruction errors to enable event-level anomaly detection. Notably, unlike common approaches, this design is not dependent on a reference model, making it more suitable for deployment in real-world cases. Furthermore, its performance verifies the autoencoder architecture's capability for training on data that contain significant noise. Building on this work, Nolle et al. [9] further developed the autoencoder architecture for unsupervised anomaly detection in 2018, utilizing a Feed-Forward Network (FFN) and introducing Gaussian noise to the training data to prevent overfitting. By parsing the input and output vectors into sub-vectors, this iteration enables attribute-level anomaly detection by calculating the deviation between the input and the reconstructed individual attributes.

In 2020, Junior et al. [10] targeted instances that require unsupervised training due to a lack of datasets with labeled anomalies. Three algorithms were studied: Support Vector Machines (SVM), One-Class (OC) SVM, and Local Outlier Factor (LOF). Inspired by Natural Language Processing (NLP), the researchers used the Word2Vec to convert business process traces into a machine-learnable format by treating each activity as a word and aggregating resulting traces into sentences. This approach resulted in a corpus of unique activities, each with a distinct semantic meaning. The study provided a reliable solution to represent business process traces in a machine-learnable format without any additional domain expertise or base references.

Huo et al. [11] in 2021 utilized Graph Neural Networks by representing event logs as graph structures to capture connections between activities better. The conversion process creates an edge-attributed graph with positional embeddings, representing event data and its sequence within a trace. Utilizing a GNN-based autoencoder, the anomaly detection model was trained on the task of edge feature reconstruction. The resulting improvement demonstrated the superiority of representing business processes as graph data structures over common tabular sequences.

BINet by Nolle et al. [12] provided an anomaly detection solution that operates in environments that lack domain knowledge. The sequence-based architecture comprises Gated Recurrent Units (GRU) capable of multi-perspective anomaly detection, focusing on the control flow (activity sequence) of events. Additionally, the architecture is capable of detecting different types of anomalies by incorporating a rule-based classifier, building towards solutions with insights into the causes of the anomalies.

More recently, GRASPED, a sequenced-based model by Guan et al. [13], incorporates bi-directional GRU to efficiently capture temporal information within business process logs. The model incorporates an attention mechanism and a teacher-forcing method to enhance its ability to learn the sequential pattern in a trace while enabling multi-perspective anomaly detection on both the control and data flow at

TABLE 2. Related works and leveraged methodologies.

Author(s)	Year	Approach	Data Processed as	Training Objective
Nolle et al. [8]	2016	Auto-Encoders	Encoded Vectors	Denoising
Nolle et al. [9]	2018	Auto-Encoders	Encoded Vectors	Denoising
Junior et al. [10]	2020	One-Class Classification	Sequences	Predictive
Krajsic et al. [14]	2021	Sequential Auto-Encoders	Sequences	Reconstruction
Huo et al. [11]	2021	Graph Auto-Encoders	Graph Data Structure	Reconstruction
Nolle et al. [12]	2022	Sequential Auto-Encoders	Sequences	Reconstruction
Guan et al. [13]	2023	Sequential Auto-Encoders	Sequences	Reconstruction
Guan et al. [15]	2023	Sequential Auto-Encoders	Sequences	Reconstruction
Guan et al. [7]	2024	Graph Auto-Encoders	Graph Data Structure	Reconstruction

the attribute level. Common with autoencoder architectures, it uses a reconstruction-based approach to classify anomalies via reconstruction error, with a significant edge on noisy data.

Guan et al. [15] further introduced WAKE, a weakly supervised autoencoder architecture that is better equipped to handle class imbalances, performing well on a limited number of labels. The model incorporates a novel anomaly score generator, a multi-layer perceptron (MLP) that is fine-tuned on scarce anomaly labels. Their approach includes a pre-training stage to train the encoders to reconstruct non-anomalous business process traces, followed by an end-to-end optimization stage in which the autoencoder and an anomaly score generator are fine-tuned with unlabeled traces and labeled anomalies. The final stage fine-tunes a feature encoder to interpret the detected anomalies.

Finally, Guan et al. [7] introduced GAMA in 2024, a fully unsupervised, GNN-based autoencoder. The multi-graph approach uses a mixture of sequence and graph-based approaches, involving Graph Attention Networks [16] and GRU [17] to effectively capture both the structural and temporal characteristics of business processes, utilizing a multi-graph approach to process each attribute type separately on sequential label reconstruction, achieving state-of-the-art performance.

Table 2 summarizes recent related works involving unsupervised or weakly supervised methods on a limited amount of data. Unsupervised methods are preferred due to the scarcity of publicly available, high-quality, labeled business process datasets. A notable observation is that the sequential-based methods used to process business processes draw parallel from the methods employed in NLP, leveraging the sequential characteristics of business processes for anomaly detection. Lastly, a common approach in this domain is leveraging autoencoders for their effectiveness on noisy data. This approach allows the models to learn meaningful representations of business processes from real-life datasets where contamination is often inevitable without relying on base reference models, thereby making autoencoders more suitable for deployment.

Despite the advancements in the field, numerous challenges remain unresolved. Sequence-based models excel with regard to capturing the temporal dependencies within a trace, but they still struggle to capture the structural complexities of business process workflows. This limitation is mirrored by graph-based approaches, which effectively capture the structural characteristics of workflows but struggle with temporal dependencies. In addition, due to the construction of the graph data structure, it is often not possible to do event or attribute level anomaly detection such as in [11], making them incapable of diagnosing the root causes of anomalies. Furthermore, as outlined by Guan et al. [6], when tested in more challenging and realistic conditions, anomaly detection still lacks the performance suitable for deployment in real-world scenarios. Although much improvement has been made by the GAMA model [7], many of the aforementioned issues persist and await further developments.

III. METHODOLOGY

This study builds upon the GAMA model proposed by Guan et al. [7], introducing several architectural enhancements and exploring denoising training objectives. The major improvements include replacing the original Graph Attention (GAT) layers with Graph Attention version 2 (GATv2) proposed by Brody et al. [18] in the multi-graph encoder. The GATv2 convolutional operator from Torch Geometric library was used, as it is the official port of the convolutional operator. We evaluated three decoder variations: the original GRU decoder, a Transformer decoder, and a GRU-Transformer hybrid. Additionally, we introduce a denoising objective alongside the original reconstruction objective. This section provides a comprehensive overview of the datasets used, describes the improved anomaly injection process for evaluation, explains the multi-graph structure used to transform business process event logs into a machine-learnable format and elaborates on the implementation choices, tuning and training details of the Graph Auto-Encoder (GAE). Through these investigations, we aim to enhance the model's anomaly detection capabilities while improving its scalability and efficiency for real-world deployment scenarios.

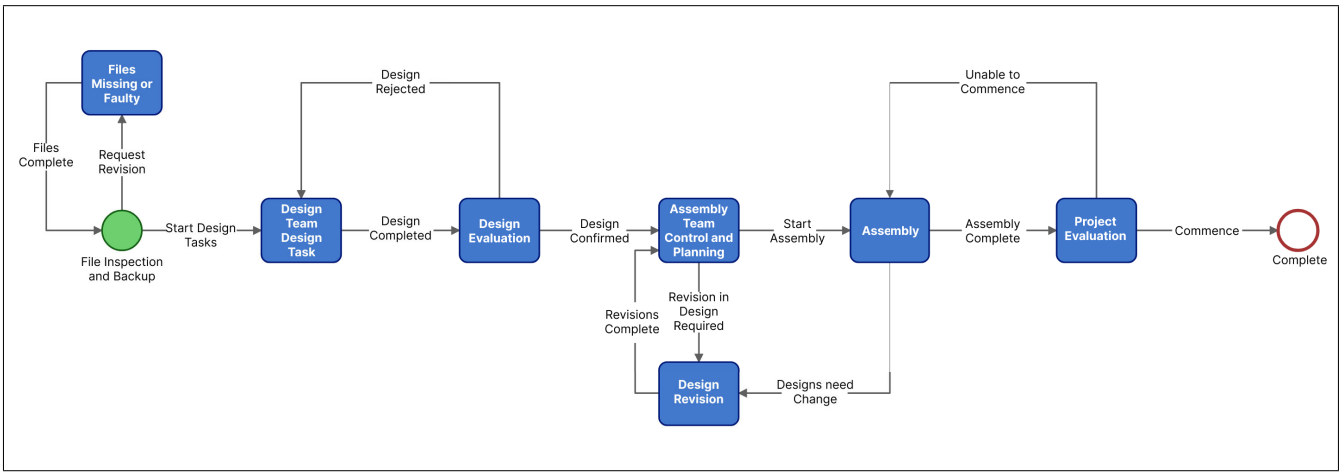


FIGURE 1. Example of a business process workflow from Next4biz BPM Platform.

A. DATASETS

The scarcity of publicly available high-quality datasets is a significant challenge in business process anomaly detection research [4]. This scarcity is largely due to the sensitive nature of business process event data, which often contains confidential information about a company's operations, activities, and clients. To address this issue, the BPI Challenge initiative has become a valuable source of benchmark datasets for the process mining community [19], [20]. Following the approach of previous researchers [7], [11], [13], [15], this study uses the BPI Challenge datasets. Additionally, we utilize process event log data from an anonymous tenant of the Next4biz (N4B) BPM platform as our core dataset. Due to legal reasons, the tenant's name and workflow details will be kept anonymous.

Next4biz BPM platform is a no-code BPM platform that offers users the ability to manage and analyze their business processes with little to no technical background. Many industry giants use it, from aviation, finance, and banking to telecommunication. The workflows found within the Next4biz BPM platform, as shown in Figure 1, consist of two components: the initial is the states that the trace travels through, represented by the blue boxes, and the second is the transitions, which are represented by the black lines connecting various states to another. The other datasets are sourced from three different years of the BPI Challenge: the 2012 challenge, the 2017 challenge, and the collection from the 2020 challenge. The BPI Challenge 2012 dataset comes from a Dutch bank. This dataset, originating from a financial institution, provides a comprehensive illustration of the application of BPM in a real-world business setting. The 2017 dataset comes from the same financial institution but contains significantly more fields and more than twice as many records. The remaining datasets form part of the collection for the BPI Challenge 2020, with each dataset showcasing a unique workflow within comparable structures. The BPI Challenge 2020 Domestic Declarations dataset [21] contains events related to domestic travel expense claims over a two-year period. The BPI Challenge 2020 International Declarations [22] contains processes for international travel expense claims. BPI Challenge 2020 Travel Permit Dataset [23] focuses on the approval process for international travel permits. The remaining two datasets, BPI Challenge 2020 Request For Payment [24] and BPI Challenge 2020 Pre-paid Travel Costs [25], deal with payment requests and pre-paid travel expenses. All these datasets follow a similar process flow, involving submission and approval by the travel administration, budget owner, and supervisor, with potential director approval in some cases. Table 3 presents an overview of the datasets the experiments were conducted on, providing information regarding the distinct number of traces, number of events, and number of attribute values for each.

TABLE 3. Datasets experimented with.

Dataset Name	# Traces	# Attributes	# Attribute Values
Next4biz BPM Tenant	7020	4	66
BPIC 2012 [26]	13087	1	36
BPIC 2017 [27]	31509	2	175
BPIC 2020-D [21]	10500	3	26
BPIC 2020-I [22]	6449	3	44
BPIC 2020-P [23]	7065	3	61
BPIC 2020-R [24]	6886	3	29
BPIC 2020-PR [25]	2099	3	39

B. INJECTED ANOMALIES

The lack of labeled anomalies in real-world datasets makes it difficult to evaluate our model or any other given solution. As the acquisition of labels for naturally occurring anomalies is rather expensive, with the aim of solving this problem, researchers proposed injecting artificial anomalies

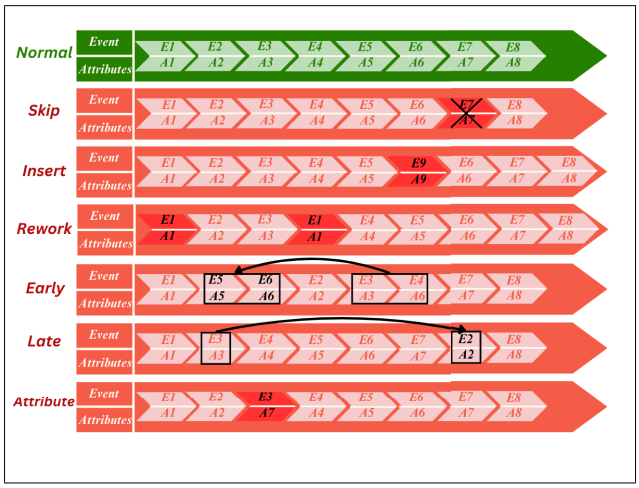


FIGURE 2. Illustration of the improved injected anomaly types.

into event logs to create a ground truth for evaluation [12], [28], [29]. This enables a controlled evaluation of model performance while preserving the fundamental structure and characteristics of real business processes. Our anomalies are as illustrated in Figure 2. The **skip** anomalies refer to an event or a sequence of events that is omitted from the trace. In the real world, it can represent skipped steps in a given process flow, indicating an anomalous occurrence. **Insert** anomalies refer to occurrences where a random event from the set of available events is inserted into the trace. In the real world, similar traces can arise from irregular management of a given flow. The **rework** type of anomaly in this context is a sequence of events that are frequently repeated, indicating an issue such as a bottleneck in the process flow. The **early** anomaly refers to an event that takes place significantly earlier in the trace than it is expected to. This kind of anomaly can indicate skipped steps in a given process flow, which can be an indicator of crookedness in the way the process was managed. The **late** type of anomaly refers to an event in the process flow taking place significantly later than it is expected to. This type of anomaly in the real world would indicate significant delays in the process. Lastly, an **attribute** anomaly refers to an event with certain of its attributes altered and changed in such a way that now the event constitutes an anomaly in the trace. This type of anomaly in the real world can represent any form of unwanted occurrence but would mostly indicate a resource misallocation or unauthorized activity.

In our study, we improved the previous techniques used by researchers in the labeling process. Particularly, our definitions of insert anomaly and attribute anomaly types are a more realistic representation of deviations that may occur in actual business process logs. For example, when injecting insert anomalies, we randomly insert events from the dataset's current set of probable events rather than creating wholly new events that did not previously exist, unlike past researchers. Similarly, for attribute anomalies, we alter attributes to other valid values in the dataset rather than adding non-existent attribute values.

These changes to the anomaly injection process are more fit for real-world circumstances, as most BPM systems limit users to a preset set of events and attribute values. By sticking to these constraints, we ensure that our injected anomalies represent real-world deviations rather than unrealistic ones. Although this method makes it more challenging for the model to detect anomalies, it offers a more accurate evaluation of the model's ability to identify them. However, it is important to consider a limitation of this approach. Real-world datasets contain natural anomalies that our artificial injection process does not and can not label. These pre-existing anomalies can impact the calculated F1-scores, as they will be correctly identified by the model but counted as false positives in the confusion matrix due to lack of labeling.

C. MULTI-GRAPH DATA STRUCTURE AND GRAPH GENERATOR

Business process datasets are typically structured hierarchically. Each trace is generated independently from a business process workflow, the underlying structure of which the model aims to learn in order to detect anomalies at various levels. A *log* (dataset) consists of a set of independent *traces* (cases), where each trace is identified by a unique case number. A trace represents a sequence of *events*, where each event consists of a set of relevant *attribute values* associated with it. The BPM software defines a set of attribute types, with each type having a range of possible values. For instance, in a transaction-based workflow, an *attribute type* such as "User" might have a predefined set of discrete values representing the role of said user within the workflow. Another attribute type could be "Transaction Amount," which may take on continuous values. Therefore, an event in this workflow is represented by the values of these two attribute types. In this study, only discrete attribute types are considered by the model, as the anomaly score is calculated on the probability distribution over categories.

To formalize the structure of the business process datasets, let $L = \{T_1, T_2, \ldots, T_n\}$ represent a log as a set of traces, where each trace T_i is defined as a sequence of events $T_i = \{e_1, e_2, \ldots, e_m\}$. Each event e_t consists of a tuple of attribute values $e_t = (a_{t1}, a_{t2}, \ldots, a_{tk})$ where a_{tk} corresponds to the value of the attribute type α_k at event e_t. The set of attribute types $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ is predefined by the BPM system.

Such data can naturally be represented by graph structures, hence the popularity of graph-based methods in existing literature. However, as noted by Guan et al. [7], anomalous patterns within a workflow may exhibit distinct behavior across various attributes. Consequently, anomaly detection can be improved by encouraging the model to learn the underlying workflow structure separately for each attribute type. To achieve this, we employ the graph generation process outlined in [7] to create a separate graph for each attribute type. However, we omitted their noise filtering mechanism

based on state occurrence frequency to allow the model to identify the precise location of anomalies within a trace in production, prioritizing anomaly localization at the cost of potentially noisier data and a less smooth training process.

Our graph generation process, as depicted in Figure 3, works as follows: For each trace T in a log L, a directed trace graph $G(T)$ is generated, with the events $\{e_1, e_2, \ldots, e_m\}$ of the trace serving as nodes, where an edge relation (e_{t-1}, e_t) denotes event e_{t-1} is chronologically followed by event e_t within trace T. Since each event e_t is a tuple of attribute values $e_t = (a_{t1}, a_{t2}, \ldots, a_{tk})$, we generate an attribute-specific graph G_{α_i} for each attribute type α_i in $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$, by replacing each event node e_t in the trace graph $G(T)$ with the corresponding attribute value a_{ti}. For instance, the attribute-specific graph for attribute type α_k is $G_{\alpha_k} = \{a_{1k}, a_{2k}, \ldots, a_{mk}\}$ for m events in T. Consequently, for any trace T, we generate a set of graphs: $T = \{G_{\alpha_1}, G_{\alpha_2}, \ldots, G_{\alpha_k}\}$, where each graph corresponds to an attribute type α.

D. THE GRAPH AUTO-ENCODER ARCHITECTURE

The model architecture investigated and developed further in this study is based on the GAMA model by Guan et al. [7], which proposed several key design choices with unique benefits that constituted an ideal foundation for this study. Firstly, as an unsupervised autoencoder, GAMA directly addresses the primary challenge in business process anomaly detection, the absence of quality-labeled anomalies across various business domains despite the fair quantity of BPM logs that are available both privately and publicly. Additionally, autoencoder-based architecture performs particularly well on noisy and wildly varied instances generated from the same workflow by compressing long sequences of event logs into compact latent feature representations that meaningfully represent the case instance within the context of the workflow-specific structure. This allows the model to better adapt to and capture the changes in the ever-evolving workflows of a variety of business domains in deployment. Secondly, by generating attribute-specific graphs for every case and utilizing separate autoencoders to process each of the graphs, the model allows the autoencoders to more closely analyze the normal and anomalous behavior that is unique to its respective attribute, followed by a holistic evaluation of the case through the attribute-specific attention mechanism on all attribute encoders' output latent feature representations. Such an approach retains the flexibility needed to learn attribute-specific behaviors, offering the possibility of a more precise diagnosis of the causes of anomalies at the attribute level while contributing to more robust overall anomaly detection at the case level. Figure 4 provides an overview of the GAMA architecture [7]. The model comprises of multiple interconnected stages that enable more granular control over the flow of information between attribute-specific autoencoders and across time steps. The following subsection outlines the details of the original encoders and decoders architectures, as well as variations investigated in this study.

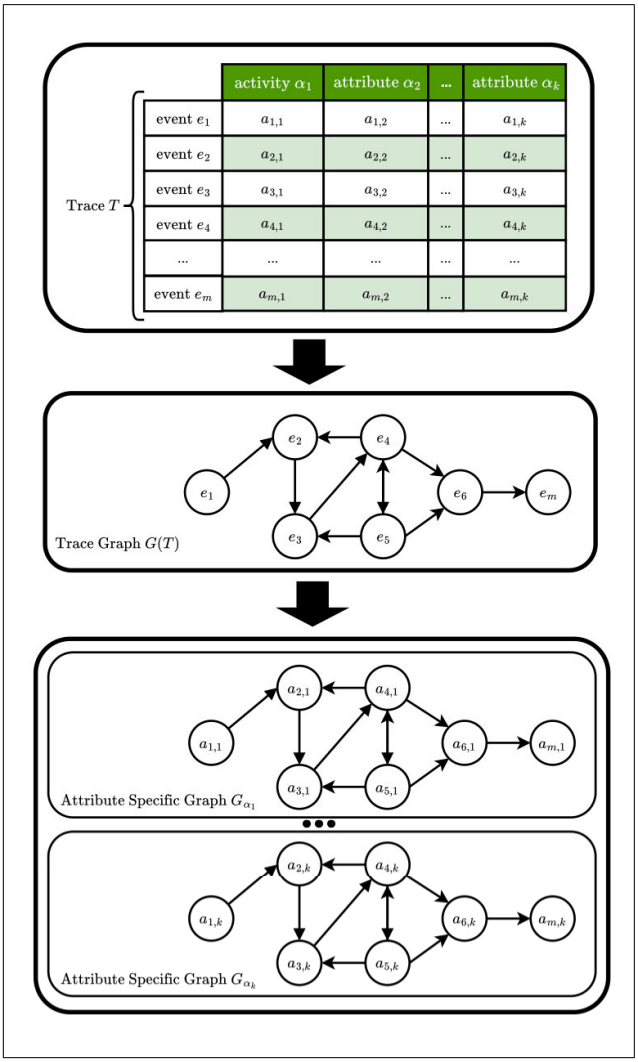


FIGURE 3. Illustration of the graph generation process for a single trace from raw logs.

1) DENOISING AND RECONSTRUCTION AS TRAINING OBJECTIVES

The primary challenge in business process anomaly detection is a scarcity of datasets with high-quality labeled anomalies, which necessitates an unsupervised training process with artificially injected anomalies to evaluate a model's effectiveness. Models trained on direct label prediction reconstruct the original input labels from feature-rich latent representations. This approach is sound because the model is encouraged to capture latent features that accurately represent non-anomalous workflow structures. Consequently, the model correctly predicts the input labels for non-anomalous instances but tends to misclassify labels for anomalous instances. This method is particularly beneficial for general unsupervised anomaly detection, where it is difficult to explicitly define anomalies that apply to all domains. However, it does not incentivize the model to understand the deviation patterns of anomalies and how they structurally

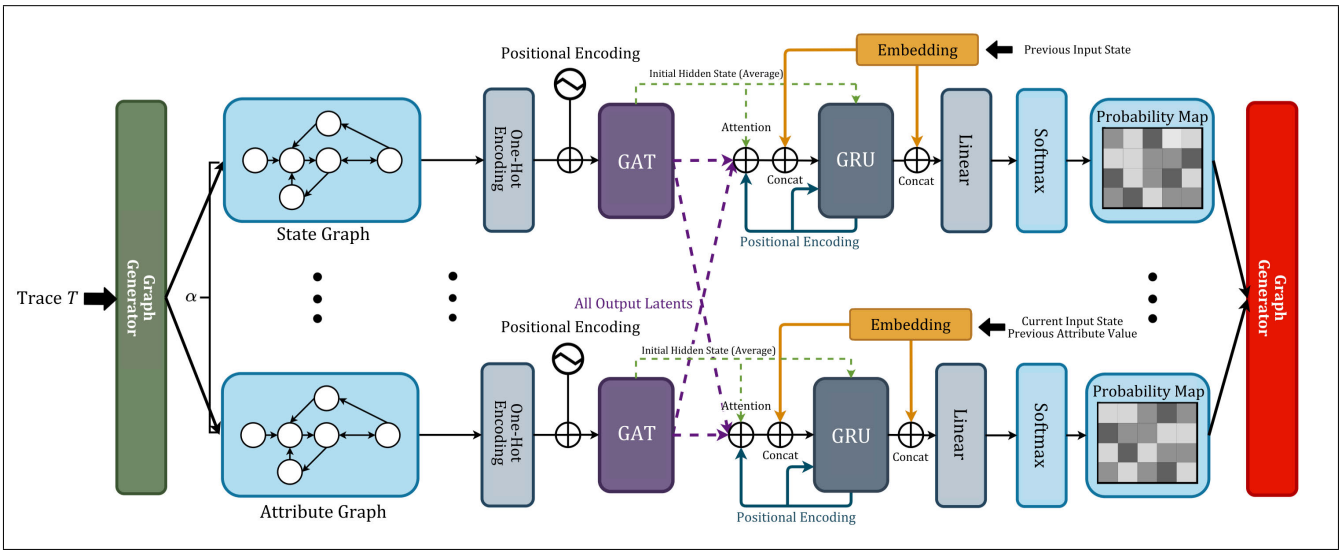


FIGURE 4. An illustration of the original GAMA architecture utilizing GRU decoders [7].

manifest within workflows. To address this limitation, we sought to test whether the model can also be trained on a denoising task to reproduce a clean sequence of events from an anomalous input when given an anomalous input trace, thereby suggesting corrective actions for the input trace. This reduces the reliance on the model making incorrect predictions based solely on predominantly clean training data while retaining much of the robust latent building on the non-anomalous instances in the non-contaminated portion of the training dataset. The purpose of exploring the denoising task is to investigate the viability of post-training on explicitly defined anomalies for expensive and deployed systems in various business domains. This aims to integrate the model's unsupervised general anomaly detection capabilities with later domain-specific anomalies that leverage the feature-rich latent the trained model already possesses without being constrained to the natural limitations of the available data and architectural limitations with purely unsupervised methods.

2) ATTRIBUTE-SPECIFIC GRAPH ATTENTION ENCODER AND GATv2

Each encoder receives its attribute-specific graph as input, with the discrete attribute values at the nodes one-hot-encoded into vectors. In order to output context-rich latent representations of the case, both the sequence of events within the case and its meaning in the context of the overall structure of the workflow must be encoded. Therefore, Guan et al. [7] proposes using Graph Attention Networks (GAT) [16] to effectively capture the structural latent representation of the cases through attention mechanism on neighboring nodes, regardless of how chronologically distant the events are. However, a shortcoming of GAT lies in its lack of consideration for the chronological order of events, which is crucial for determining whether a node instance is anomalous. To address this, the sine and cosine positional encoding, standard in Transformer architectures [30], is applied to the node feature vectors:

$$PE(\text{pos}, 2i) = \sin\left(\frac{\text{pos}}{10000^{\frac{2i}{d}}}\right) \tag{1}$$

$$PE(\text{pos}, 2i+1) = \cos\left(\frac{\text{pos}}{10000^{\frac{2i}{d}}}\right) \tag{2}$$

where *pos* denotes the position of the event in the trace, i is the dimension in the node features, and d is the number of possible attribute values of the specified attribute type. The positionally-encoded node features are then input into the GAT layers, which outputs a sequence of feature-rich node latent feature representations. The encoding process is depicted in Figure 4.

To improve the original GAT Encoder deployed by Guan et al. [7], we tested GATv2 proposed by Brody et al. [18]. In contrast to the static attention mechanism in the original, v2 utilizes a more expressive dynamic attention mechanism, which may allow the encoders to capture more complex attribute relationships.

3) ATTRIBUTE-SPECIFIC DECODER VARIATIONS

After encoding node features with structural context from the sequence of events, the decoders reconstruct the input by outputting a probability distribution over each attribute type's possible values. To achieve this, the primary component of the decoders are sequence-based models, namely GRU [17] and Transformers, which focus on predicting input labels chronologically. Crucially, each encoder was specifically designed to focus on the attribute-specific behavior of the trace without any exchange of information between attribute types. Therefore, an attribute-specific attention mechanism on the output latents of all attribute types at the current time step is used to provide the decoder with

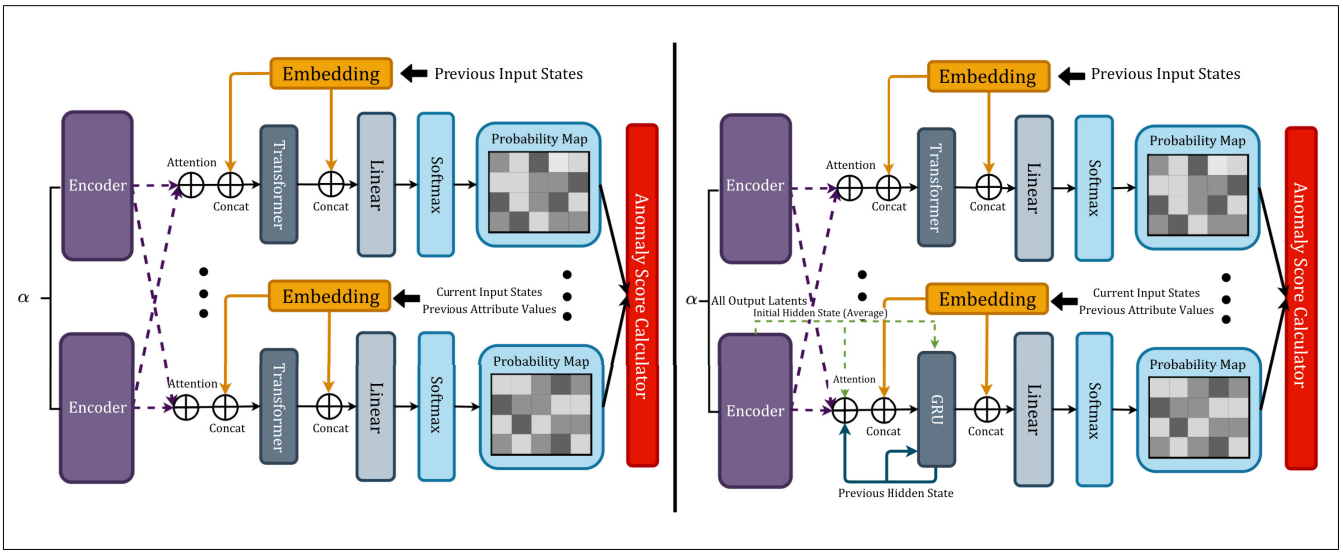


FIGURE 5. An illustration of the transformer-only version 1 decoder architecture (Left) and Transformer-GRU-Hybrid version 2 decoder architecture (Right).

additional information regarding the other attribute types in the workflow. Keeping consistent with the original GAMA architecture [7], teacher forcing is applied. Consequently, for all three decoder variants described below, there are two separate versions for each: state decoder and attribute decoder. Each model has a single state decoder that handles the activity nodes, the primary attribute type responsible for the control flow of the case; therefore, its recurrent network takes the previous input state and the attention output at each time step as input. The remaining attribute decoders differ from the state decoder in that they take the previous input attribute value, the current input state, and the attention output as input to the recurrent component.

The **original GAMA decoders (v0)** found in the architecture employ GRU to process each time step sequentially, maintaining granular control over the information available to the decoder in a chronological manner, which is illustrated in Figure 4. The attribute-specific latents at all time steps are first averaged to obtain the initial attribute-specific decoder hidden state, which is compact and feature-rich, effectively representing the attribute-specific trace. At each time step, attention is applied to the decoder's hidden state and encoder outputs of all other attribute types. The state and attribute decoders differ in their GRUs' input for teacher forcing. For the state decoder, the input is the concatenation of the attention output and the embedding of the previous input state. For the attribute decoders, the input is the concatenation of the attention output, the embedding of the current input state, and the embedding of the previous attribute value. The GRU's output, concatenated with its input, is then linearly projected into output logits. The decoder thus outputs both the logits and the GRU's last hidden state for the next time step. By forcing the decoder to make predictions with minimal context, it is incentivized to build a resilient internal model to produce an expressive hidden state for the next time step. Consequently, any deviation from the standard workflow structure in the anomalous trace is more likely to produce an incorrect prediction for the input label, which is beneficial in unsupervised anomaly detection. However, such granular control results in expensive training runs for larger clients.

To circumvent the increased training cost of the GRU decoder, its recurrent component is substituted with a **Transformer (v1)** [30] to enable parallelization, resulting in several key changes to the decoder, which is depicted in Figure 5. Firstly, the decoder's hidden state is substituted for the attribute-specific latent features from the encoder, as the Transformer will be underutilized if it produces hidden states sequentially as input for the next time step. Consequently, the latents are not averaged, and attention is instead applied to the attribute-specific latents and the encoder output of every attribute type at their time step. Consistent with the original decoder, a distinction is made for the state and attribute decoders on the input to the recurrent component. The output of the Transformer is followed by linear projection to obtain output logits. Several characteristics arise from these alterations. Firstly, the incentive for the decoder to build a better internal model by utilizing hidden states for the next time step is substituted with the Transformer's internal attention mechanism. The additional expressiveness may enable the decoders to reconstruct the sequence input labels more accurately, as it directly computes attention scores on latent features at different time steps instead of sequentially passing information. In addition, as the encoder output latent features at each time step are taken directly as input to the decoder without being averaged, it may alleviate pressure on the attribute encoders to produce more compact latents. With increased context and potentially less compact latents, the Transformer variant has the potential to overfit anomalous

traces to produce correct input labels even for anomalous traces, which would be detrimental to unsupervised anomaly detection through input label prediction, but can also increase the performance of denoising task. Therefore, the application of a causal mask is critical to the performance of the different training objectives.

As the state (activity attribute) is responsible for the control flow of cases, it is likely to have more complex structural and sequential behavior relative to other attributes. Therefore, for direct label reconstruction, an increased context for the state decoder may be beneficial to accurately reconstruct the non-anomalous sequence of states, while the attribute decoder may benefit from a limited context from teacher forcing of the previous attribute value and current state. As a supplementary investigation for the structural complexity of states relative to other attributes and the impact of context on attribute value prediction, this variant uses a **Hybrid Decoder (v2)** with a Transformer for the state decoder and GRU decoders for all other attributes, as illustrated in Figure 5.

As [31], a learnable temperature parameter for each attribute-specific decoder is included, allowing the model to optimize to be more or less confident in its prediction by tuning the sharpness of the output probability distribution over possible attribute values.

4) ANOMALY SCORE CALCULATION AND DETECTION: RELATIVE VS ABSOLUTE PROBABILITY

With the output logits, a probability distribution over possible values for each attribute type at a given time step is obtained via softmax operation on the logits. A model outputs a higher probability for attribute values that are likely to occur given the typical workflow structure and observed sequence of events and a lower probability for out-of-distribution observations. Therefore, the anomaly score for each attribute at an event is calculated to be higher for unlikely occurrences and lower for standard occurrences. In [7], the anomaly score of an attribute at an event is calculated as the sum of probabilities greater than the probability of the input attribute value:

$$S_{a,e} = \sum_{p^a_{e,i} > p^a_{e,i_{input}}} p^a_{e,i} \tag{3}$$

where $p^a_{e,i}$ is the probability of the i-th value of attribute type a at event e, and $p^a_{e,i_{input}}$ denotes the probability of the input value.

The anomaly score thereby predicts an attribute is anomalous not just because the input value is unlikely but because there are numerous more likely candidates, which neatly results in a 0 anomaly score for the most probable attribute value. However, several implications arise from relative likelihood. Firstly, it is less sensitive to absolute likelihood, as the probability aggregation of more likely candidates is affected by the quantity of more likely candidates, regardless of how much higher the individual probabilities are compared to the observed value. This means that it may be less effective in separating infrequently observed values and highly unlikely instances. Conversely, the method can also yield low anomaly scores for anomalous instances if there are fewer other values with higher probabilities. Furthermore, the aggregation operation is more expensive for attribute types with a large set of possible values. Overall, relative likelihood may be less effective for larger workflows with complex interaction and attribute types with a large set of possible values. For different business domains, how anomalies should be defined is central to what method to use to classify anomalies. In some use cases, instances are of interest when there are more probable alternatives, where the context of the choice among several options is significant. In domains primarily concerned with outliers, any rare event is of interest. Therefore, we compare the original relative likelihood method to an absolute likelihood one by taking the negative logarithm of the probability of the observed attribute value $p^a_{e,i_{input}}$:

$$S_{a,e} = -log(p^a_{e,i_{input}}) + 10^{-9} \tag{4}$$

which results in a higher sensitivity to absolute likelihood and better computational efficiency. However, there are tradeoffs to be made with using absolute likelihood. Firstly, it may lack relative likelihood's resilience to uniformly low probability distribution for some attribute types. Secondly, by computing anomaly score directly on the model's prediction for the original label, it can overemphasize rare events in datasets and use cases where context is of higher importance. Lastly, negative log likelihood is more sensitive to model's calibration on absolute probability predictions. Intuitively, both anomaly score calculation methods apply to reconstruction and denoising objectives alike, as both operations incentivize the model to output low probabilities for unlikely input attributes. As the original GAMA implementation [7], any attribute at an event with an anomaly score exceeding threshold τ is labeled as anomalous:

$$L_{a,e} = \begin{cases} 1 & \text{if } S_{a,e} > \tau \\ 0 & \text{if } S_{a,e} \leq \tau \end{cases} \tag{5}$$

Event and trace level anomalies are inferred from the existence of attribute anomalies within them:

$$L_{trace} = \begin{cases} 1 & \text{if } \exists L_{a,e} = 1 \text{ for } e \in t, \text{ for } a \in \alpha \\ 0 & \text{if } \forall L_{a,e} = 0 \text{ for } e \in t, \text{ for } a \in \alpha \end{cases} \tag{6}$$

IV. EXPERIMENTAL RESULTS

A. TRAINING PARAMETERS

The experiments were run on a dedicated research and development server running on Ubuntu 22.04.4 LTS. The CPU is an Intel i9-14900KS with 128 GB of dedicated RAM and 256 GB of swap. The conducted experiments were run on an NVidia RTX A6000 with cuda 12.1 and a CPU usage limit of 50% of the maximum. The code was run on Python 3.11.8 with torch 2.3.1, torch-geometric 2.5.3 for

graph learning purposes, numpy 2.0.0, pandas 2.2.2, and pm4py 2.7.11.11.

All datasets are separately trained on 1%, 5%, 10%, 25%, and 50% contamination rate for 20 epochs with checkpointing set in place to get the best possible model weights. Adam optimizer [32] was used with a learning rate of $5e^{-3}$. As [7], we utilize a hidden dimension of 64, with 0.1 dropouts in the positional encoders and 0.4 dropouts within the GATs and decoders' recurrent components to prevent overfitting. Two GAT layers are used in the encoder, the first using 4 heads and the second using 1 head. For the recurrent component in the decoders, GRUs have 2 layers, while Transformer encoder has 1 layer with 8 heads. 4 main architectures are tested: GAMA retains the same configurations as [7], while v0, v1, and v2 utilize GATv2 instead of GATv1 in the encoders and has the learnable temperature parameter enabled, but differ in the decoder architecture. Where transformers are used in the decoder, causal masks are used. Unless explicitly stated, anomaly scores are calculated with negative log-likelihood for computational efficiency and all datasets are evaluated at trace, event, and attribute levels, each containing F1-score, precision, and recall on a test size of 0.2.

B. PERFORMANCE OF GRU AND TRANSFORMER AS STATE AND ATTRIBUTE DECODERS

Table 4 reports the highest F1 scores achieved across any contamination rate, comparing the peak performances of the different training objectives and model architectures. The results demonstrate no single model has a dominating performance across all datasets and training objectives. Nevertheless, it is useful to investigate how effective Transformers and GRUs are the primary recurrent components in state and attribute decoders. We attempt to isolate the effects of the different decoder architectures by picking pairs of models for comparison, where the two share identical portions of architecture that are not the focus of the analysis, in order to contrast the state and attribute decoder portions more directly. Figure 6 displays the differences in peak-F1 between models, each isolating performance delta when changing from GRU to Transformer decoders.

Figure 6(a) compares the performance delta when transitioning from GRU to Transformer state decoders for reconstruction tasks. Generally, Transformers slightly outperform GRU state decoders in trace-level detection (blue) but perform worse in attribute-level detection (orange). Similar trends are observed in denoising tasks (Figure 6(b), where the performance decrease is particularly pronounced with the BPIC2017 dataset. The performance changes appear to be dataset-dependent, as the directions of change in attribute-level performance are consistent across both training objectives, though the magnitude varies. These results suggest the hypothesis of state transitions being more complex than attribute transitions is incorrect since state decoders do not appear to benefit from longer context and direct attention score calculation on the different time steps. Notably, the BPIC12 dataset contains a single attribute type that is treated as the state, meaning its models contain no attribute decoders and provide a direct comparison between GRU and Transformer state decoders, where the differences in attribute-level detection appear minimal. This may be caused by the different ways in which the state decoder variations utilize their attribute-specific encoder outputs, as Transformer decoders use their encoders' outputs directly, while GRU decoders use the mean of encoder outputs across all time steps as the initial hidden state. This may force the encoders to build latent features in incompatible ways that confuse the model when the latent features are exchanged via the attention mechanism, which does not occur when there is a single attribute.

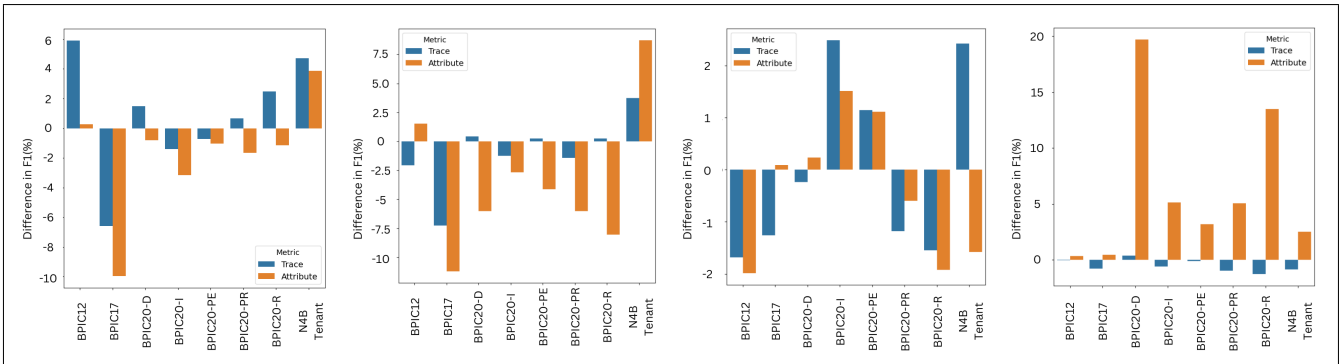
Figure 6(c) shows the choice of attribute decoders has much less impact on the performance of reconstruction tasks, with both decoder variations showing less than 2% difference in attribute-level F1. While the differences in F1 scores for reconstruction tasks could be attributed to randomness in the training process, the performance boost offered by Transformer attribute decoders is far more pronounced, with half of the datasets gaining more than 5% and two datasets gaining more than 10% in attribute-level F1 score. This behavior is expected, given the additional capacity and context required for models to learn how anomalies structurally manifest within a workflow to suggest corrective actions to anomalous traces. However, this additional capacity offered little to no performance gains in trace-level detection, which suggests attribute-level predictions may not always be effectively propagated to the trace level.

C. EFFECT OF CONTAMINATION RATE

Figure 7 demonstrates the mean F1-score across dataset contamination rates for denoising and label reconstruction objectives. It can be observed that the F1-score of trace-level prediction and contamination rate have a strong positive correlation for both training objectives. This phenomenon is also observed by Guan et al. [7], which the authors reasoned could be due to higher contamination rates obscuring the effects of naturally occurring anomalies in the datasets. Although a positive correlation is also observed for attribute-level detection in reconstruction tasks, its performance in denoising tasks is unaffected by the contamination rate. It is possible that by explicitly training on correcting anomalies, denoising tasks with higher contamination rates allows the model to learn how anomalies structurally manifest within a workflow but also leaves a smaller portion of uncontaminated examples to learn the workflow's underlying structure from, with the effects of the two learning perspectives canceling each other out - unaffected by contamination rate. Interestingly, this result suggests that at higher contamination rates, reconstruction tasks are better suited for attribute-level anomaly detection despite not training on correcting anomalies explicitly. Furthermore, the decoupled performance changes between attribute and trace-level detection on denoising tasks are also observed in Figure 6(d). This occurred despite

TABLE 4. F1 score comparison of trace (T) and attribute (A) level detection across architectures and training objectives. Performance reported by existing literature is included for comparison. The disparity between the results from the original GAMA [7] and the current run is a result of margins added to the decoder input, as the input and output may have different lengths with the stated anomaly injection method and denoising objective. As the GAMA model outperforms every previously proposed approach with minimal exceptions, the comparison between the proposed improvements and approaches previous to GAMA using advanced anomaly injections was excluded as they fell redundant.

Dataset		Next4biz		BPIC12		BPIC17		BPIC20_D		BPIC20_I		BPIC20_PE		BPIC20_PR		BPIC20_R	
Type	Model	T	A	T	A	T	A	T	A	T	A	T	A	T	A	T	A
Existing Works	OC-SVM [33]	-	-	0.331	-	0.383	-	0.379	-	0.385	-	0.385	-	0.386	-	0.378	-
	Naive [29]	-	-	0.451	-	0.484	-	0.803	-	0.696	-	0.620	-	0.727	-	0.797	-
	GAE [11]	-	-	0.411	-	0.383	-	0.554	-	0.420	-	0.405	-	0.463	-	0.549	-
	DAE [9]	-	-	0.585	0.291	0.414	0.234	0.873	0.361	0.586	0.209	0.558	0.200	0.596	0.209	0.850	0.330
	VAE [34]	-	-	0.576	0.338	0.404	0.135	0.663	0.202	0.488	0.154	0.466	0.159	0.486	0.159	0.604	0.185
	LAE [34]	-	-	**0.862**	0.477	0.386	0.169	0.880	0.497	0.702	0.447	0.613	0.457	0.666	0.487	0.847	0.479
	BINet [12]	-	-	0.669	0.470	0.665	0.366	0.809	0.535	0.769	0.482	0.768	0.468	0.749	0.485	0.843	0.553
	GRASPED [13]	-	-	0.700	0.503	0.735	**0.533**	0.784	0.468	0.748	0.511	0.711	0.498	0.716	0.466	0.848	0.507
	GAMA [7]	-	-	0.773	**0.563**	**0.745**	0.485	**0.935**	**0.618**	**0.849**	**0.593**	**0.797**	**0.554**	**0.783**	**0.653**	**0.921**	**0.618**
Recon	GAMA	0.944	0.706	0.818	0.540	0.797	0.509	0.949	**0.728**	0.912	**0.713**	**0.882**	**0.655**	0.918	**0.710**	0.940	**0.743**
	V0	0.921	0.687	0.842	0.601	**0.814**	**0.522**	0.945	0.723	0.906	0.710	0.870	0.647	0.917	**0.710**	0.920	0.740
	V1	**0.992**	0.710	0.884	0.584	0.735	0.423	0.958	0.717	**0.917**	0.694	0.874	0.648	0.912	0.687	0.930	0.709
	V2	0.968	**0.726**	**0.901**	**0.604**	0.748	0.422	**0.960**	0.715	0.892	0.679	0.862	0.637	**0.924**	0.693	**0.945**	0.728
Denoise	GAMA	0.939	0.531	**0.904**	0.569	0.808	**0.533**	0.966	0.480	**0.892**	0.614	0.858	0.606	0.898	0.603	**0.975**	0.532
	V0	0.938	0.553	0.806	0.536	**0.964**	0.488	0.898	0.605	0.854	0.597	**0.900**	**0.631**	**0.960**	0.525	0.905	0.561
	V1	0.967	**0.665**	0.885	**0.580**	0.725	0.428	**0.972**	**0.624**	0.880	**0.630**	0.856	0.588	0.876	**0.621**	0.949	**0.580**
	V2	**0.976**	0.640	0.885	0.576	0.733	0.423	0.969	0.427	0.886	0.579	0.857	0.556	0.886	0.571	0.962	0.445



(a) State decoder for reconstruction objective: F-scores of v2 minus F-scores of V0, as both versions have GRU attribute decoders.

(b) State decoder for denoising objective: F-scores of v2 minus F-scores of V0, as both versions have GRU attribute decoders.

(c) Attribute decoder for reconstruction objective: F-scores of V1 minus F-scores of v2, as both versions have Transformer state decoders

(d) Attribute decoder for denoising objective: F-scores of v1 minus F-scores of v2, as both versions have Transformer state decoders

FIGURE 6. Peak-F1 changes (Transformer minus GRU) to isolate performance delta when changing the decoder recurrent component from GRU to transformer.

anomalies at the trace level being defined by the existence of attribute-level anomalies within the trace. The decoupling of the F1-score changes at different levels suggests that the interaction between the attribute-specific autoencoder is complex and that the ways in which each individual autoencoder contributes to the anomaly classification at the trace level still require further investigation.

D. ANOMALY SCORE: RELATIVE VS ABSOLUTE LIKELIHOOD

Figure 8 shows the changes in Peak F1 score when switching from negative log-likelihood to relative likelihood as outlined in [7]. For reconstruction tasks, most datasets saw better performance when utilizing relative likelihood over absolute likelihood, with BPIC12 and BPIC17 datasets gaining more than 5% in attribute level detection with relative likelihood. It is notable that the BPIC17 dataset has approximately 15 times the number of attribute values compared to the collection of BPIC20 datasets, which suggests the number of candidate values has no negative impact on anomaly score calculation. This demonstrates relative likelihood, which is a robust method for attribute-level anomaly detection with reconstruction tasks. With denoising tasks, only 3 datasets see changes in attribute-level F1 of more than 2% when switching from absolute likelihood to relative likelihood, suggesting the anomaly score calculation method has a smaller impact on denoising operations. However, both BPIC12 and BPIC17

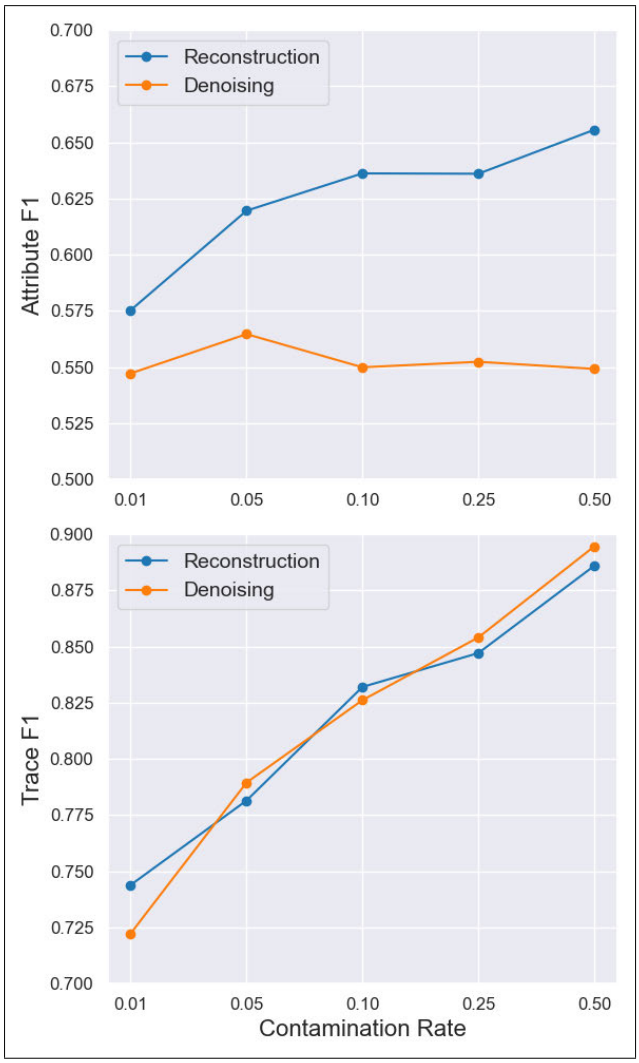
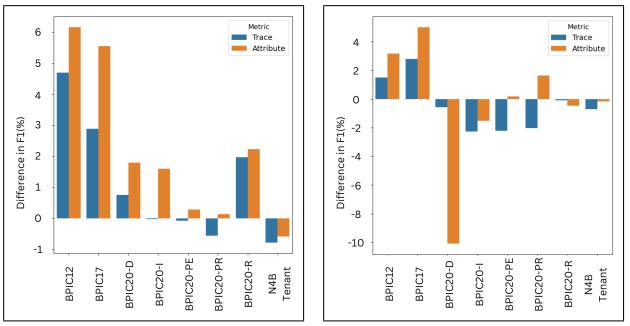


FIGURE 7. Mean F1-scores across contamination rates.



(a) Performance changes on reconstruction tasks (b) Performance changes on denoising tasks

FIGURE 8. Peak-F1 changes when switching anomaly score calculation from negative log likelihood to original anomaly score calculation as outlined in [7].

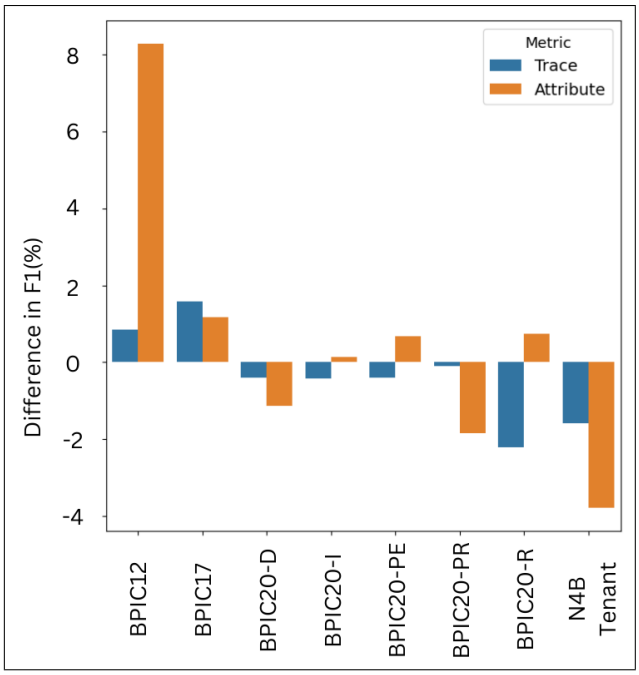


FIGURE 9. Peak-F1 changes when switching from between GATv1 to GATv2.

datasets saw larger increases in performance, an observation that is consistent with the reconstruction task, albeit at a smaller magnitude. An interesting result is a 10.1% decrease in denoising performance for the BPIC20_D dataset, showing worse performance with relative likelihood. This is unlikely to be caused by the number of candidate attribute values, as the collection of BPIC20 datasets contains approximately the same number of attribute values. This is likely caused by a combination of characteristics of the workflow structure and the training dynamics of denoising operations that lead to model calibrations that are more sensitive to absolute probabilities when calculating anomaly scores. The results in Figure 8 indicate the appropriateness of relative likelihood calculation methods for denoising and label reconstruction training objectives in most datasets.

E. GATv1 VS GATv2

To test whether the multi-graph architecture can benefit from a more expressive encoder by employing GATv2 [16], we conducted an auxiliary experiment on reconstruction tasks with the V0 architecture, with the learnable temperature parameter enabled for every decoder. As Figure 9 indicates, percentage changes in F1 are within 2% for most datasets, which can be attributed to randomness in the training process. However, the model trained on the BPIC12 dataset gained 8.28% in attribute level F1, while the N4B model observed a decrease of 3.80%. The results suggest that datasets with more complex inter-attribute relationships and structures may benefit from a more expressive encoder with a dynamic attention mechanism, while the additional expressiveness may lead to overfitting on simpler datasets.

F. MODEL EFFICIENCY

As shown by the mean run time results displayed in Figure 10, the parallelization capability of the transformer

TABLE 5. Mean dataset runtimes in seconds (Transposed).

Dataset / Model	GAMA	V0	V1	V2
N4B BPM Tenant	189.2	198.9	91.0	92.9
BPIC 2012	57.9	58.8	42.5	61.0
BPIC 2017	1402.9	1517.7	611.7	1163.4
BPIC 2020-D	140.4	142.4	64.7	141.3
BPIC 2020-I	20.1	19.5	13.4	20.3
BPIC 2020-P	67.1	68.7	44.4	70.3
BPIC 2020-R	95.1	96.3	65.4	99.3
BPIC 2020-T	193.5	196.9	69.6	169.8

models makes its time efficiency unrivaled. The closest performing equivalent, the Transformer + GRU v2 decoder model, runs for 226 seconds in the reconstruction task and 227 seconds in the denoising task, both times almost halved by the Transformer V1 decoder model, which yields 123 seconds for the reconstruction task and 127 seconds for the denoising task. Aside from the efficiency of the Transformer V1 decoder model, The GATv2 incorporation in the original GAMA architecture increased the running time by approximately 6%, where the original model took 265 seconds for the reconstruction and 276 seconds for the denoising tasks, the GATv2 incorporated GAMA V1 model took 282 seconds and 291 seconds per training objective respectively. One final observation in Figure 10 is the increased cost of using the denoising task, where, on average, it took 6.6 seconds more to run. When we inspect Table 5, we can see that the increase in runtime is not linear. The event count ratio between BPIC 2012 and BPIC 2017 datasets is approximately 4.5, but runtime with the original GAMA model proposed by Guan et al. [7] was 1402 seconds for the BPIC 2017 dataset and 57 seconds for the BPIC 2012 dataset - which is approximately a ratio of 24.6. This indicates that as the dataset scales up, the model's usability decreases significantly. Given this non-linear time increase, the incorporation of transformers is a significant improvement - the V1 model decreases the ratio to approximately 14.5, making it much more suitable from an efficiency perspective, especially in real-world deployment scenarios.

G. EFFECT OF CONTEXT LENGTH VIA CAUSAL MASKS IN TRANSFORMERS

In the original GAMA implementation [7], the architecture emphasizes the fine granularity of control over the information available to the decoder at each time step, giving the decoder minimal context to make sequential predictions over the possible attribute values. This leverages the limited context to incentivize robust latent building, such that the model predicts incorrect labels and high anomaly scores for anomalous inputs. Alternatively, one can reason that more context provides the model with a better capability to learn the workflow structure to provide better predictions.

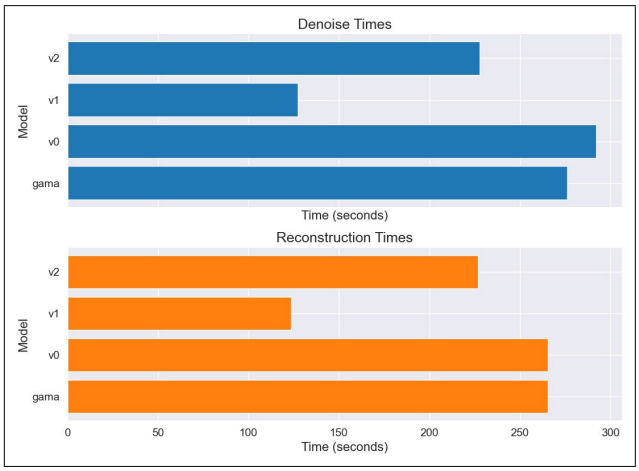
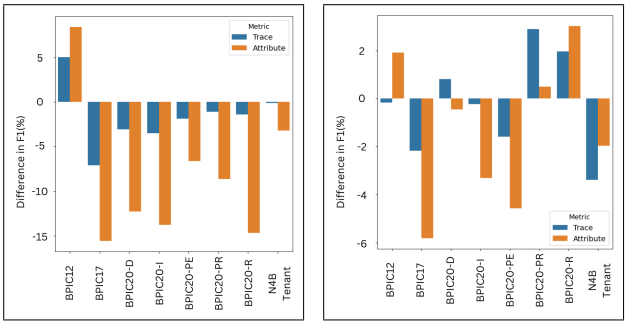


FIGURE 10. Mean model runtimes.



(a) Performance changes on reconstruction tasks (b) Performance changes on denoising tasks

FIGURE 11. Peak-F1 changes when omitting causal masks for V1 with transformer decoders.

These properties can be tested on decoder architectures utilizing transformers.

Figure 11 shows the effects on the F1 score when causal masks are removed in the V1 architecture that is fully comprised of transformer decoders. The results demonstrate the importance of causality in making sequential predictions for label reconstruction, as most datasets experience more than a 5% decrease in F1 on attribute-level detection when causal masks are removed, with 4 out of 8 of the datasets experiencing more than a 10% decrease. Despite the denoising task having a higher likelihood of benefitting from full context to make corrections to anomalous traces, the same results seen in reconstruction tasks are observed for the denoising task, albeit with smaller magnitudes. Interestingly, models trained on the BPIC12 datasets experienced modest increases in performance on attribute-level prediction on both training objectives. BPIC12 containing only a single attribute is a factor that may have contributed to this result, suggesting that attribute-level detection for single autoencoders can benefit from full context, and the interaction of attribute-specific autoencoders via attention on output latent features may have caused or amplified the importance of causality.

V. CONCLUSION

Since the introduction of the artificial neuron, neural networks have found various use cases that they have been effective at handling. Over the years, various architectures were developed to effectively handle domain-specific tasks, which further carried the field of artificial learning. As a solution oriented towards professional users, the usage of artificial neural networks in business process anomaly detection is a niche usage of artificially intelligent systems. As shown in the related works section, previous works have demonstrated the potential of artificially intelligent solutions in this domain.

In this study, we proposed a series of improvements to a state-of-the-art business process anomaly detection system to enhance its capabilities and build toward a scalable and efficient solution. We initially refined existing artificial anomaly injection methods. In addition to tackling the persistent challenge of scarce annotated datasets available for research, our enhancements allowed for a better evaluation of proposed approaches. The incorporation of Graph Attention v2 in the encoder structure made the model architecture more up-to-date and allowed it to generate more meaningful latent feature representations. By leveraging Transformer decoders instead of GRU, we achieved comparable or superior performance at half the computational cost of the original implementation. This significantly increases the scalability and real-world applicability of our solution. Furthermore, we tested a denoising objective that showed great promise for developing a versatile solution capable of later targeted training on domain-specific anomalies. Finally, we introduced a new anomaly score calculation method, which helped the model make better decisions, resulting in higher performance. We rigorously tested the system on a diverse set of public and private datasets, demonstrating the robustness and generalizability of our improvements across various business domains. When the performance achieved in our study is put side by side with existing traditional machine learning or alternative deep learning-based approaches as shown in Table 4, advancements achieved in this study mark a significant step toward a scalable and reliable production-ready anomaly detection system for business processes. Despite its achievements, our solution brings one key issue to the table when taken into action. The graph data structure is computationally more expensive than traditional machine learning and more conventional neural networks; despite being minimized by the incorporation of transformers, our proposed approach does have a computational cost disadvantage.

In order to make a production-ready anomaly detection system, future work will focus on making detection methods more reliable when dealing with complex inter-attribute relationships, creating methods for real-time anomaly prediction on incomplete cases, and looking into data-specific attribute filtering and importance scoring to make models work better across business domains.

DISCLOSURE OF INTERESTS

This research aims to develop AnomalyNet, an advanced software framework for unsupervised anomaly detection, integrated into the Next4biz BPM platform as part of an ongoing R&D initiative.

REFERENCES

[1] W. M. P. van der Aalst, "Business process management: A comprehensive survey," *ISRN Softw. Eng.*, vol. 2013, Feb. 2013, Art. no. 507984.

[2] T. B. Ayaz, E. Gülce, A. Özcan, and A. Akbulut, "Semi-supervised detection of contaminated business process instances using graph autoencoders and dynamic edge convolutions for BPM anomaly detection," in *Proc. Innov. Intell. Syst. Appl. Conf. (ASYU)*, Oct. 2024, pp. 1–6.

[3] J. Ko and M. Comuzzi, "A systematic review of anomaly detection for business process event logs," *Bus. Inf. Syst. Eng.*, vol. 65, no. 4, pp. 441–462, Aug. 2023.

[4] J. N. Adams, J. Peeperkorn, T. Brockhoff, I. Terrier, H. Göhner, M. S. Uysal, S. V. Broucke, J. De Weerdt, and W. M. P. van der Aalst, "Discovering high-quality process models despite data scarcity," 2023, *arXiv:2310.11332*.

[5] R. Sarno, F. Sinaga, and K. R. Sungkono, "Anomaly detection in business processes using process mining and fuzzy association rule learning," *J. Big Data*, vol. 7, no. 1, p. 5, Dec. 2020.

[6] W. Guan. (2024). *Survey and Benchmark of Anomaly Detection in Business Process*. [Online]. Available: https://github.com/guanwei49/BPAD

[7] W. Guan, J. Cao, Y. Gu, and S. Qian, "GAMA: A multi-graph-based anomaly detection framework for business processes via graph neural networks," *Inf. Syst.*, vol. 124, Sep. 2024, Art. no. 102405.

[8] T. Nolle, A. Seeliger, and M. Mühlhäuser, "Unsupervised anomaly detection in noisy business process event logs using denoising autoencoders," in *Proc. 19th Int. Conf. Discovery Sci. (DS)*, Bari, Italy. Cham, Switzerland: Springer, 2016, pp. 442–456.

[9] T. Nolle, S. Luettgen, A. Seeliger, and M. Mühlhäuser, "Analyzing business process anomalies using autoencoders," *Mach. Learn.*, vol. 107, no. 11, pp. 1875–1893, Nov. 2018.

[10] S. B. Junior, P. Ceravolo, E. Damiani, N. J. Omori, and G. M. Tavares, "Anomaly detection on event logs with a scarcity of labels," in *Proc. 2nd Int. Conf. Process Mining (ICPM)*, Oct. 2020, pp. 161–168.

[11] S. Huo, H. Völzer, P. A. Reddy, P. Agarwal, V. Isahagian, and V. Muthusamy, "Graph autoencoders for business process anomaly detection," in *Proc. 19th Int. Conf. Bus. Process Manag. (BPM)*, Rome, Italy. Cham, Switzerland: Springer, Jan. 2021, pp. 417–433.

[12] T. Nolle, S. Luettgen, A. Seeliger, and M. Mühlhäuser, "BINet: Multi-perspective business process anomaly classification," *Inf. Syst.*, vol. 103, Jan. 2022, Art. no. 101458.

[13] W. Guan, J. Cao, Y. Gu, and S. Qian, "GRASPED: A GRU-AE network based multi-perspective business process anomaly detection model," *IEEE Trans. Services Comput.*, vol. 16, no. 5, pp. 3412–3424, May 2023.

[14] P. Krajsic and B. Franczyk, "Semi-supervised anomaly detection in business process event data using self-attention based classification," *Proc. Comput. Sci.*, vol. 192, pp. 39–48, Jun. 2021.

[15] W. Guan, J. Cao, H. Zhao, Y. Gu, and S. Qian, "WAKE: A weakly supervised business process anomaly detection framework via a pre-trained autoencoder," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 6, pp. 2745–2758, Jun. 2024.

[16] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[17] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," 2014, *arXiv:1409.1259*.

[18] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" 2021, *arXiv:2105.14491*.

[19] B. van Dongen, "BPI challenges: 10 years of real-life miscs," Tech. Rep., 2020.

[20] (2020). *BPI Challenges*. [Online]. Available: https://www.tf-PM.org/competitions-awards/bpi-challenge

[21] (2020). *BPI Challenge 2020: Domestic Declarations*. [Online]. Available: https://doi.org/10.4121/uuid:3f422315-ed9d-4882-891f-e180b5b4feb5

[22] (2020). *BPI Challenge 2020: International Declarations*. [Online]. Available: https://doi.org/10.4121/uuid:2bbf8f6a-fc50-48eb-aa9e-c4ea5ef7e8c5

[23] (2020). *BPI Challenge 2020: Travel Permit Data*. [Online]. Available: https://doi.org/10.4121/uuid: ea03d361-a7cd-4f5e-83d8-5fbdf0362550
[24] (2020). *BPI Challenge 2020: Request For Payment*. [Online]. Available: https://doi.org/10.4121/uuid:895b26fb-6f25-46eb-9e48-0dca26fcd030
[25] (2020). *BPI Challenge 2020: Prepaid Travel Costs*. [Online]. Available: https://doi.org/10.4121/uuid:5d2fe5e1-f91f-4a3b-ad9b-9e4126870165
[26] B. van Dongen, "BPI challenge 2012," Tech. Rep., 2012.
[27] (2017). *BPI Challenge 2017*. [Online]. Available: https://doi.org/10.4121/UUID:5F3067DF-F10B-45DA-B98B-86AE4C7A310B
[28] J. Ko and M. Comuzzi, "Detecting anomalies in business process event logs using statistical leverage," *Inf. Sci.*, vol. 549, pp. 53–67, Mar. 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID
[29] F. Bezerra and J. Wainer, "Algorithms for anomaly detection of traces in logs of process aware information systems," *Inf. Syst.*, vol. 38, no. 1, pp. 33–44, Mar. 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306437912000567
[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*.
[31] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," 2021, *arXiv:2103.00020*.
[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[33] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 823–839, May 2012.
[34] H. T. C. Nguyen, S. Lee, J. Kim, J. Ko, and M. Comuzzi, "Autoencoders for improving quality of process event logs," *Expert Syst. Appl.*, vol. 131, pp. 132–147, Oct. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417419302829

STANLEY HSU is currently pursuing the Bachelor of Science (B.Sc.) degree in computer science with University College London. His research interests include real-time semantic decoding of brain scans and business process anomaly detection.

EGE GÜLCE is currently pursuing the Bachelor of Science (B.Sc.) degree in computer science with University College London. He is currently with the Research and Development Center, Next4Biz, Istanbul. His research interest includes the optimization of software solutions through artificial intelligence.

TEOMAN BERKAY AYAZ received the Graduate degree (Hons.) in computer engineering from Istanbul Kültür University, in 2023. He is currently a Data Scientist and an Artificial Intelligence Researcher with the Research and Development Center, Next4biz. His research interest includes applications of artificial intelligent solutions into real-world use cases and their integrations with other software.

ALPER OZCAN received the B.S. and Ph.D. degrees in computer engineering from Istanbul Technical University, Istanbul, Türkiye, in 2007 and 2017, respectively. He is currently an Associate Professor in computer engineering with the Department of Computer Engineering, Akdeniz University. His research interests include social networks, data mining, and machine learning. He is teaching undergraduate and graduate-level courses. He is a member of the Learning from Data Research Laboratory, Department of Computer Engineering. His research interests include data mining, machine learning, and deep learning.

AKHAN AKBULUT (Member, IEEE) received the B.S. and M.S. degrees in computer engineering from Istanbul Kültür University (IKU), Türkiye, in 2001 and 2008, respectively, and the Ph.D. degree in computer engineering from Istanbul University, Türkiye, in 2013. From 2004 to 2013, he was a Research Assistant with the Department of Computer Engineering, IKU, where he was an Assistant Professor, from 2013 to 2017. From 2017 to 2019, he was a Postdoctoral Researcher with the Computer Science Department, North Carolina State University, NC, USA. In 2019, he rejoined the Department of Computer Engineering, IKU, and was promoted to an Associate Professor. He became a Full Professor, in 2024. He is currently the Chairperson of the Department of Computer Engineering, IKU. Since 2023, he has also been acting as the Research and Development Director of Next4biz. His current research interests include the design and performance optimization of software-intensive systems, machine-learning applications, internet architectures, and broadening participation in cloud computing research.