

Graph Based Business Process Anomaly Detection with Edge Feature Reconstruction and Advanced Linear Networks

Teoman Berkay Ayaz*, Rabia Çevik*, Alper Özcan[†], Akhan Akbulut^{*‡}

^{*}*R&D Center of Next4biz Sahrayicedit Mah, Pakpen Plaza,
No:40/4, 34734 Istanbul, Turkey*

teoman.ayaz@next4biz.com, rabia.cevik@next4biz.com, akhan.akbulut@next4biz.com

[†]*Department of Computer Engineering Akdeniz University Antalya, Turkey
alperozcan@akdeniz.edu.tr*

[‡]*Department of Computer Engineering Istanbul Kültür University 34536 Istanbul, Turkey
a.akbulut@iku.edu.tr*

Abstract—Business Process Management (BPM) as an interdisciplinary field between Managerial Sciences and Computer Science is a subject ever-increasing in importance. This holds more and more true as the business landscape becomes faster and more complex each passing day. Given the management of a businesses operational activities is essential to maintain a healthy lifecycle, the early detection of these inefficiencies and potentially malicious activity becomes more and more crucial. As these deviations can significantly impact a businesses lifecycle, anomaly detection solutions in this domain is that much more lucrative. The pursuit of detecting these deviations gave rise to the field of Business Process Anomaly Detection. Building upon previous research, our study focuses on constructing an advanced Graph Autoencoder (GAE) architecture using various graph convolutional operators, and boost the performance further with advanced linear networks. By comprehensively evaluating 3 distinct encoder architectures and 4 distinct decoder selections, our study comprehensively evaluates the possible ways to combine various encoders and decoders on 6 distinct datasets. The empirical results show a wide range of results with varying trends between different encoder-decoder combinations, ranging from 0.674 to 0.219 F1-score in anomaly detection performance.

Index Terms—Graph Neural Network, Autoencoder, Business Process, Anomaly Detection, Unsupervised, Machine Learning

I. INTRODUCTION

Business Process Management (BPM) is crucial in improving organizational productivity by designing and refining the management processes. But even in real world business execution, processes tend not to follow their designed procedures or flow due to varying degrees of inefficiencies, fraud, error, and system failure. These anomalies when classified can lead to significant discrepancies in process performance and regulatory compliance needs [1]. As the level of automation within organizations' workflows increases, so does the need for highly effective and timely detection of anomalies. BPM systems exhibit various shortcomings, including but not limited to irregular user behavior, extended execution times, deletion of process activities, excessive repetitions, and events occurring in an

unexpected order. Addressing these anomalies is critical to ensure process integrity, regulatory compliance, and resource efficiency. Identifying these anomalies in complex business environments is essential to maintaining operational integrity, regulatory compliance, and optimal resource allocation.

Adaptability to dynamic and large-scale business environments involving identification of anomalies has imposed limitations on the conventional techniques such as statistical thresholding and rule-based detection due to their dependence on pre-defined patterns and static thresholds [2]. Formalization of this notion-very recent advances in machine learning, deep learning, and graph-based approaches, holds more adaptive and data-driven solutions for anomaly detection in BPM systems. A graph-based approach for the business process anomaly detection can help models to capture the intrinsic dependencies within business process traces allowing for better deep learning. Additionally, advanced linear networks can then further push this by learning significant feature representations of process anomalies. Our study investigates the usage of hybrid models in combination with the Graph Autoencoder (GAE) architecture. We comprehensively test 3 distinct graph encoder architectures, paired with 3 advanced linear decoders in addition to a GRU based sequential decoder. In summary, our paper:

- Introduces a novel encoder architecture for GAE construction.
- Investigates and introduces advanced linear networks into GAE based reconstruction tasks.
- Provides insights on advanced model construction.

Remainder of this paper will briefly lay the groundwork for this study under the Related Works section, delve into the approaches and explain the logic behind choices made in this study under the Methodology section, present the reader empirical results under the Experimental Results section, summarize the findings under the Conclusions section and discuss future works under the Discussion & Future Works section.

II. RELATED WORKS

The subject of anomaly detection under the Business Process Management (BPM) domain has been a prominent area of research due to the growing need to ensure process efficiency, compliance, and operational integrity. Traditional anomaly detection techniques such as rule-based systems and statistical thresholding have suffered setbacks in dynamic business environments due to the reliance on predefined patterns and static thresholds [3]. To overcome the limitations of traditional approaches, researchers like Guan et al. [4] and Nolle [5] investigated the use of sequence capturing deep models to detect anomalies that are typically invisible to more commonly employed methods. Other researchers, such as Huo et al. [6] went towards the direction of unsupervised graph-based methods to achieve state-of-the-art anomaly detection performance on business process traces.

Enabled by recent developments in deep learning, Graph Neural Networks (GNNs) provide an opportunity to overcome the limitations of traditional methods, enhance adaptability and performance in BPM anomaly detection frameworks. BPM workflows designed in compliance with the Business Process Management Notation (BPMN) 2.0 closely resemble graph structured data, where business process event can be represented by nodes and transitions can be represented by edges. Given this, graph processing of business process datasets can provide significant benefit in the context of machine learning [7]. This especially holds true when we take the highly complex and interconnected nature of business process workflows into account.

In a study, predecessor to the current one [3], our research team investigated edge-attribute reconstruction with edge-computing graph encoders and advanced linear networks. Additionally, our team investigated the use of dimension increasing autoencoder architectures, which in theory can result in richer latent feature representations resulting in better generalization and overall higher anomaly detection performance. The study leveraged a dataset from an undisclosed Next4biz BPM platform tenant and used naturally occurring anomalies, discovered by statistical and rule based analysis in addition BPM feedbacks. With a total of 2 encoder-decoder combinations and 2 distinct latent feature dimensions, our research team trained an edge-feature reconstruction model for business process anomaly detection. With a reconstruction error percentile based thresholding approach, where each threshold represents an anomaly score point and above, our research team managed to achieve a trace level F1-score of 0.70 with both dimension increasing and standard autoencoders. Our current research aims to build on top of its predecessor using more advanced neural network architectures to increase the anomaly detection performance. Additionally, current study uses a more informative graph construction scheme which include more features in an attempt to boost the anomaly detection performance.

III. METHODOLOGY

Our study employs a wide range of approaches in both advanced model construction and in feature engineering. Initially, each of the 6 datasets used in this study undergo an injection phase where artificial anomalies are injected for model performance evaluation. After that, a preprocessing process follows to ensure machine learnability and employ feature engineering to enrich the dataset. Once the preprocessing is complete graph construction begins where each distinct business process trace gets converted into a distinct knowledge graph, allowing models to capture the intricate and complex dependencies existing within process traces. For each dataset, a pipeline similar to a grid-search is put to work where each distinct encoder-decoder combination for Graph Autoencoder construction gets trained and tested for a comprehensive evaluation of the available solutions.

A. Datasets

The study was conducted on multiple datasets which table I shows a brief overview of. The priority dataset is the private dataset, also leveraged in the predecessor study [3], the Next4biz Tenant dataset. The dataset originates from a Next4biz BPM platform tenant whose identity will be kept anonymous. The dataset consists of 44,413 process events, 8 columns and 6,178 distinct business process traces. The remaining datasets are a collection process event logs related to travel expense claims, published as a part of the Business Process Intelligence (BPI) Challenge 2020 [8]. These datasets provide diverse workflows to test the validity of the proposed approach on in addition to providing a generalizable benchmark. The study includes the full collection which include domestic declarations (BPIC 2020-D), international declarations (BPIC 2020-I), travel permit data (BPIC 2020-P), request for payment (BPIC 2020-R) and prepaid travel costs (BPIC 2020-PR).

TABLE I
LIST OF INCLUDED DATASETS

Dataset Name	Trace Count	Event Count	Unq. Event Count
Next4biz Tenant	6178	44413	12
BPIC 2020-D	10500	56437	17
BPIC 2020-I	6449	72151	34
BPIC 2020-P	7065	86581	51
BPIC 2020-R	6886	36796	19
BPIC 2020-PR	2099	18246	29

Table II shows an overview of the general structure of study datasets after the extraction from Extensive Event Stream (XES) files and initial processing. The initial processing stage contains the XES file to CSV file conversion and basic processing of the dataset, including the ordering of the process traces and calculating event durations in addition to preparing

columns for artificial injections. The trace id (TID) column contains the distinct case identifier for each separate business process trace. The transition column contains information on the process lifecycle transitions. Event column contains the event stream where per row there is a single event and other relevant attributes. UserID column contains the identifier for the user responsible in the event. The duration feature is an engineered feature calculated from the difference between current and next timestamps for each row. Lastly, the timestamp column contains the time information of when an event took place.

TABLE II
GENERAL STRUCTURE OF DATASETS

TID	Transition	Event	UserID	Dur.	Time
t7	Starting	Process Start	918	12	2024-09-12
t7	To Process	Process Data	918	38	2024-09-12
t7	To Training	Training Model	216	188	2024-09-12
t7	Model Trained	Model Testing	211	44	2024-09-12
t7	Tests Done	Validating Model	211	47	2024-09-12

B. Artificial Anomaly Injections & Natural Anomalies

As finding labeled business process anomaly detection datasets is a challenging subject, researchers leverage options such as artificial anomaly injections [9]. In our study, we leveraged two distinct types of label sets. The initial label set is the label set acquired by statistical and analytical methods in the predecessor study [3]. The acquisition process was conducted in accordance with the feedback from the Business Intelligence (BI) specialists available in the Next4biz R&D center. The latter sets of artificially injected labels are custom designed anomalies in accordance descriptions given by BI specialists for the AnomalyNet system requirements.

- **Over-time** : Too much time spent for a given event. Gets calculated separately for each event based on statistical info.
- **Over-repeat** : Too many repeats of a single event or a subsequence of events. A repetition of minimum 3 and maximum of 6 for injections.
- **Illegal User** : An unauthorized or rare user within an event.
- **Skip Event** : The skipping of an event in the normal sequence. Gets injected at a random time step of the trace.
- **Insert Event** : The insertion of an event that should not be in the order of events. Inserted with a random insertion length for each injection.
- **Out of Order** : The swapping of two consecutive events.

C. Graph Construction

The predecessor study leverages a graph construction algorithm, proposed by Huo et al. [6] where the researchers use

edge positional embeddings adapted for the Next4biz tenant dataset. In this study, the edge positional embeddings were retained as a part of the feature set, as a one hot encoded representation of an edge's order within the trace. Where max_t is the maximum length of a trace and emb is an empty embedding space with $max_t - 1$ dimensions, the creation of pre-aggregation edge embeddings can be described as the assignment of $f[i]$ into $emb[i]$ where i represents the order of feature f within a trace. After assignments into the empty embedding space takes place, aggregation can be described as the summation of each duplicate $(node_t, node_{t+1})$ edge. As the only continuous numeric feature in the datasets was the duration feature, as opposed to 1 or 0 binary labels into the embedding space, the duration of each event was assigned into the embedding space with the aim of combining both features. The other relevant features were kept as one-hot encoded feature vectors. Where $\parallel$ represents feature concatenation and:

- v_e represents the one-hot encoded event vector,
- v_t represents the one-hot encoded transition vector,
- v_u represents the one-hot encoded user vector,
- v_{dp} represents the positional duration embeddings,
- $fset_{node}$ represents the node feature vector
- $fset_{edge}$ represents the edge feature vector

The features in the constructed graphs can be described as:

$$fset_{node} = v_e$$

$$fset_{edge} = v_t \parallel v_u \parallel v_{dp}$$

D. Model Architectures

For anomaly detection, the study leverages a GAE model working towards edge feature reconstruction using the latent features of neighboring nodes. The deployed model architectures consist of 3 distinct encoder architectures and 4 distinct decoder architectures. The encoder architectures include an Edge Conditioned Convolution (ECC) Encoder based on the ECC operator introduced by Simonovsky et al. [10], inherited from the predecessor study [3]. This type of operator leverages a dedicated neural network to compute on edge attributes, making it a great general purpose edge-computing operator. The second option is the Transformer Encoder based on the Transformer Convolution Operator introduced by Shi et al. [11], originally for semi-supervised classification tasks. The decision to include this encoder arises from the experimental results, which show that the model developed yields some state-of-the-art results in a scenario where limited supervision is available. This study tests their performance in a fully unsupervised environment. The last encoder selection used in this study is a hybrid encoder, which was originally designed for node feature reconstruction tasks on edge-attributed graphs. As shown on Figure 1, the hybrid encoder composes of the two previously mentioned encoder architectures, a latent feature generator and two more encoders: the Graph Attention (GAT) and Graph Sampling and Aggregation (SAGE) encoders. The GAT encoder is based on the advanced GAT v2 convolutional operator introduced by Brody et al. [12]. As opposed to original GAT mechanism [13] which uses static attentions, GAT v2 introduces a dynamic attention mechanism, allowing for

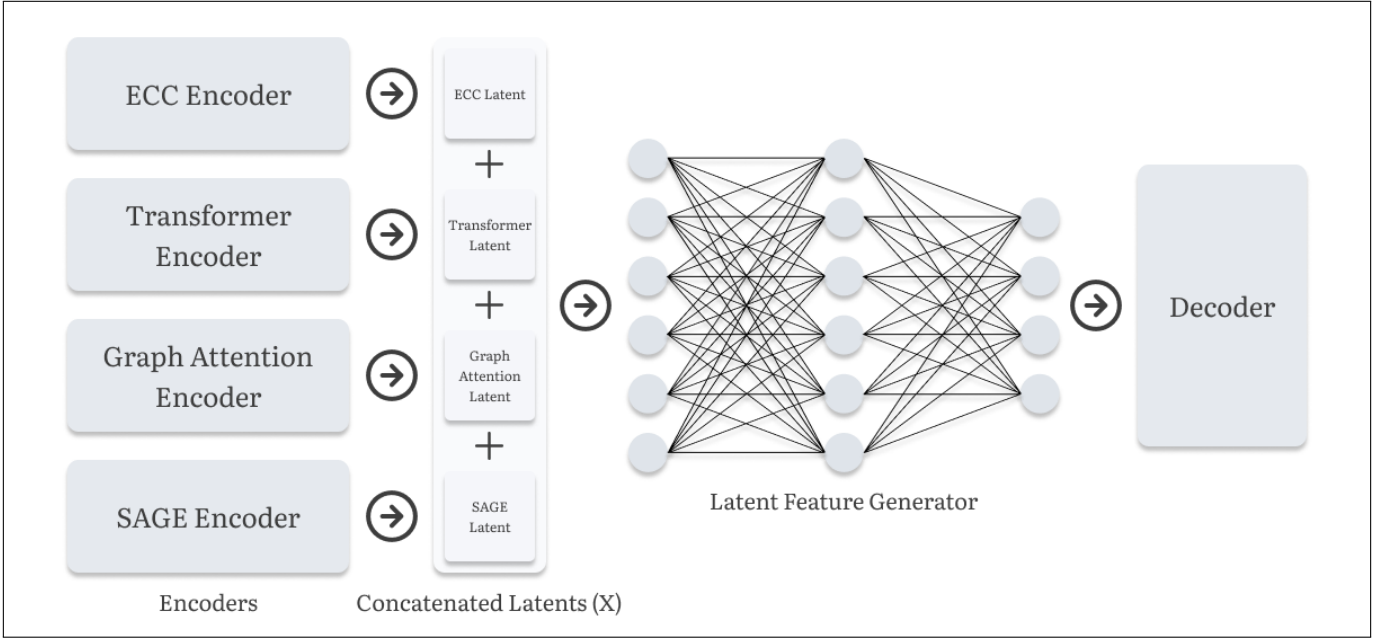


Fig. 1. Depiction of GAE with Hybrid Encoder

better learning on graph structured data. The SAGE encoder, introduced by Hamilton et al. [14], leverages sampling and aggregation across node neighborhoods for inductive learning. Despite the lack of edge-computation capabilities, the SAGE operator has shown considerable performance on graph related tasks. Given this, it was included for learning solely on node features. One thing that should be noted about the hybrid encoder, is that the tests were run with and without SAGE encoder to test and ensure its inclusion does not cause any confusion for the model. Regarding the latent feature generator, the hybrid encoder model uses a linear network to pool and process the different outputs into a single latent feature representation which will be fed to the decoder. The generator help transform the outputs, acting in a similar fashion to an attention mechanism, dynamically assigning weights to different features.

To accompany these encoders, as mentioned a total of 4 decoders. The initial one is the Multi-Layer Perceptron (MLP) based decoder which is meant to act as the baseline for the decoder end of our study. Given that the MLP architecture is the most popular deep network architecture available due to its success in deep learning applicable domains [15], it acts as a great baseline for how a deep decoder should perform in our given setup. The second decoder selection available in the study is a Kolmogorov-Arnold Networks (KAN) based decoder, also inherited from the predecessor study [3]. The KAN architecture, introduced by Schmidt Hieber et al. [16], was introduced as an improvement over the MLP architecture. As opposed to the rectified linear unit (ReLU) activation functions, KANs use trainable spline activation functions which often leads to higher performance across various tasks. The third option which the CompleXNet (XNet) architecture, which

was proposed by Li et al. [17] as an improvement over the KAN and the MLP architectures. XNets use Cauchy activation functions as opposed to splines found in KANs. The activation is inspired by the Cauchy integral theorem which gives the network powerful approximation capabilities. XNets can also perform better with less layers which gives it higher efficiency than KANs. Overall, from a theoretical perspective the XNet architecture is a promising KAN replacement. Regarding the linear network architectures, one notable thing is that both activation functions solve the dying neuron problem, where neurons in a network end up giving static outputs. This boosts the representation power of both networks considerably.

The last decoder selection is a Gated Recurrent Unit (GRU) based decoder choice. Where (a, b) is an edge between $node_a$ and $node_b$, and $\parallel$ represents feature concatenation an edge latent can be formulated as:

$$latent_a \parallel latent_b = latent_{a,b}$$

As graphs are constructed by following the consecutive order of the process trace, this construction causes edges to represent two-step subsequences of $(node_t, node_{t+1})$. Despite the edge aggregation during graph construction, edges may still contain some sequential information which a GRU based decoder may be able to learn on. To test this, the study incorporates a GRU based decoder.

E. Training & Evaluation

Each model combination was trained for 25 epochs total with check-pointing in place to get the model weights with minimum loss. Model were trained on Mean Squared Error (MSE) loss to measure the deviation between the actual input and the reconstructed output. The check-pointing on minimum

TABLE III
MAXIMUM F1-SCORES WITH VARYING MODEL COMPONENT COMBINATIONS

Model		Dataset					
Encoder	Decoder	Next4biz Tenant	BPIC 2020-D	BPIC 2020-I	BPIC 2020-P	BPIC 2020-R	BPIC 2020-PR
ECC	MLP	0.583	0.541	0.272	0.280	0.544	0.323
	KAN	0.577	0.670	0.306	0.291	0.674	0.399
	XNet	0.586	0.623	0.325	0.256	0.625	0.340
	GRU	0.381	0.488	0.251	0.224	0.496	0.315
Transformer	MLP	0.579	0.467	0.244	0.252	0.561	0.234
	KAN	0.593	0.653	0.281	0.219	0.618	0.364
	XNet	0.576	0.574	0.255	0.239	0.605	0.314
	GRU	0.464	0.340	0.231	0.234	0.391	0.269
Hybrid ³	MLP	0.561	0.540	0.245	0.249	0.561	0.252
	KAN	0.576	0.629	0.248	0.256	0.667	0.438
	XNet	0.550	0.560	0.253	0.248	0.592	0.429
	GRU	0.487	0.519	0.235	0.241	0.456	0.273
Hybrid ⁴	MLP	0.570	0.593	0.227	0.295	0.582	0.234
	KAN	0.573	0.624	0.263	0.219	0.663	0.402
	XNet	0.571	0.574	0.236	0.241	0.622	0.352
	GRU	0.453	0.508	0.268	0.248	0.536	0.331

MSE loss enabled the selection of the best model state and solely carry over the top performing models to the evaluation stage. The evaluation was done by calculating the MSE of each individual graph represented process trace with mean and max aggregation. Following this, an F1-score conversion took place where anomaly score percentiles between 100 - 75 were tested to acquire the highest F1-score yielding threshold point.

IV. EXPERIMENTAL RESULTS

Table III shows the performance results of the tested encoder-decoder combinations with the maximum F1-score between mean and max aggregation. To start with the Next4biz tenant dataset, the top performing model is the Transformer-KAN combination with an F1-score of 0.593. When set to work with GRU and XNet decoders, Hybrid³ encoder seems to outperform Hybrid⁴ encoder which indicates the SAGE encoder is causing confusion for the overall feature extraction. It could also indicate a compression issue related to the encoder output dimensions where the latent output layers struggle to compress the information into a single latent space. Taking decoders into account, when paired with any encoder with the exception of ECC, it outperforms its counterparts. This indicates the spline activation functions found in the KAN architecture is far more suitable for the given setup. When we take a look at the results for the BPIC 2020-D dataset, KAN still is the top decoder architecture when paired with the ECC encoder, yielding an F1-score of 0.670. On the side of hybrid encoders, both configurations fall behind their Transformer and ECC based counterparts with yielding a maximum F1-score of 0.629. This is further indication of information bottleneck, as one would expect an encoder architecture which includes both to outperform them by significant margins. When we

look at the results for BPIC 2020-I and BPIC 2020-P datasets, the performances across the board show that all combinations underperform when testing on these datasets. Given these datasets have the highest unique event counts, 34 and 51 respectively, we can see that the models fail to grasp the underlying logic inside these process traces. The results from BPIC 2020-PR also confirm this observation, with a unique event count of 29 and the third lowest maximum F1-score of 0.438. One noticeable thing about the configurations when inspecting the results from the datasets, is that hybrid encoders seem to outperform their counterparts as the unique event count goes higher with BPIC 2020-I being an exception. On the dataset with the highest event count, BPIC 2020-P, the top performing combination is Hybrid⁴ encoder with MLP decoder. On BPIC 2020-PR the top performer is the Hybrid³ encoder, yielding an F1-score of 0.438. Hybrid³ outperforming its larger counterpart can be viewed as proof towards the idea of an information bottleneck within the encoder architecture. Lastly, the top performer for the BPIC 2020-R is the ECC-KAN combination with Hybrid³-KAN combination following close by with respective F1-scores of 0.674 and 0.667. The performances in general show a trend favoring the KAN decoder as the champion decoder model with the ECC encoder as its best performing combination.

V. CONCLUSION

The empirical results show that KAN based decoder architecture is the best pair with any available encoder selection whereas the GRU based decoder selection completely fails, yielding considerably less performance as opposed to its counterparts across the board. Both the XNet and MLP architectures fall behind the KAN based solution, indicating that

despite the more advanced approximation capabilities of the XNet architecture, spline activation of the KAN architecture provides better learning for reconstruction tasks. ECC and Transformer based models taking a lead over the hybrid solution shows that in this case, simpler solutions perform better than larger and more complex architectures. The Hybrid³ encoder outperforming Hybrid⁴ encoder in various cases indicate that there is an information bottleneck, potentially at the latent generation segment of the encoder architecture. Validating this theory requires further testing with a wide range of dimensions. Another potential culprit for the generally lower F1-scores is the GAE models learning to reconstruct too well. This is likely to happen in scenarios where either the latent feature dimensions are too large or the model learns to reconstruct the input so well that it successfully reconstructs anomalous inputs as well. Validating for both these scenarios require testing with smaller dimensions with a focus on reconstruction error. One last observation regarding the performance on Next4biz tenant dataset, is that the performance with the same ECC-KAN combination being significantly lower than in the predecessor study. This indicates the higher dimensionality of the edge attributes significantly decreased performance, especially dimensionality of features irrelevant to the anomalies of interest. With everything considered, the study successfully employs a fully unsupervised anomaly detection solution on all datasets with the exception of 2. Considering the injected anomalies were modelled after the feedback of BI specialists, they are fully capable of real-world outputs which can be of interest to real-world professionals working in the domain.

VI. DISCUSSION & FUTURE WORKS

As mentioned, the hybrid encoders display behavior indicating a potential information bottleneck. To take the architecture further, future research will need to focus in this direction unlocking the capabilities of this architecture. Another potential culprit as mentioned is models learning to reconstruct anomalies too well, which certainly requires experimentation with lower latent dimensions and reconstruction errors. Lastly, another observation showed that when dimensionality increases unnecessarily, the F1-scores drop as generalization and reconstruction becomes more difficult for the model. Future research may additionally focus on a more optimized graph construction process where features not explicitly related to the anomalies can be kept as node features to preserve the info while not making it more difficult to reconstruct the edges.

ACKNOWLEDGMENTS

This study is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under grant no TEYDEB-3231136.

DISCLOSURE OF INTERESTS

This study is done as part of an ongoing Software R&D endeavor named AnomalyNet. The effort aims to develop a deployment-ready business process anomaly detection solution aiming to optimize business processes.

REFERENCES

- [1] W. Guan, J. Cao, H. Zhao, Y. Gu, and S. Qian, "Survey and Benchmark of Anomaly Detection in Business Processes," *IEEE Transactions on Knowledge & Data Engineering*, vol. 37, no. 01, pp. 493–512, Jan. 2025. [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/TKDE.2024.3484159>
- [2] D. Zhao and Z. Zhou, "Anomaly and interestingness detection in timed hierarchical business processes," *IEEE Transactions on Engineering Management*, vol. 71, pp. 12 619–12 634, 2024.
- [3] T. B. Ayaz, E. Gülce, S. Hsu, A. Özcan, and A. Akbulut, "Multi-aspect anomaly detection with graph neural networks and kolmogorov-arnold networks in business process management," in *2024 9th International Conference on Computer Science and Engineering (UBMK)*. IEEE, 2024, pp. 557–562.
- [4] W. Guan, J. Cao, Y. Gu, and S. Qian, "Grasped: A gru-ae network based multi-perspective business process anomaly detection model," *IEEE Transactions on Services Computing*, vol. 16, no. 5, pp. 3412–3424, 2023.
- [5] T. Nolle, S. Luetzgen, A. Seeliger, and M. Mühlhäuser, "Binet: Multi-perspective business process anomaly classification," *Information Systems*, vol. 103, p. 101458, 2022.
- [6] S. Huo, H. Völzer, P. Reddy, P. Agarwal, V. Isahagian, and V. Muthusamy, *Graph Autoencoders for Business Process Anomaly Detection*, 08 2021, pp. 417–433.
- [7] 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 *2024 Innovations in Intelligent Systems and Applications Conference (ASYU)*. IEEE, 2024, pp. 1–6.
- [8] B. van Dongen, "Bpi challenge 2020," 2020. [Online]. Available: https://data.4tu.nl/collections/_/5065541/1
- [9] S. Hsu, E. Gulce, T. B. Ayaz, A. Ozcan, and A. Akbulut, "Multi-graph anomaly detection in business processes with scalable neural architectures," *IEEE Access*, 2025.
- [10] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," 04 2017.
- [11] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," *arXiv preprint arXiv:2009.03509*, 2020.
- [12] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" *arXiv preprint arXiv:2105.14491*, 2021.
- [13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.
- [14] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.
- [15] M.-C. Popescu, V. E. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions on Circuits and Systems*, vol. 8, no. 7, pp. 579–588, 2009.
- [16] J. Schmidt-Hieber, "The kolmogorov-arnold representation theorem revisited," *Neural networks*, vol. 137, pp. 119–126, 2021.
- [17] X. Li, Z. Xia, and H. Zhang, "Cauchy activation function and xnet," *Neural Networks*, p. 107375, 2025.