Process Mining Encoding via Meta-learning for an Enhanced Anomaly Detection

Gabriel Marques Tavares[1(✉)] and Sylvio Barbon Junior[2]

[1] Università degli Studi di Milano (UNIMI), Milan, Italy
gabriel.tavares@unimi.it
[2] Londrina State University (UEL), Londrina, Brazil
barbon@uel.br

Abstract. Anomalous traces diminish the event log's quality due to bad execution or security issues, for instance. Focusing on mitigating this phenomenon, organizations spend efforts to detect anomalous traces in their business processes to save resources and improve process execution. Conformance checking techniques are usually employed in these situations. These methods rely on the comparison of the event log obtained and the designed process model. However, in many real-world environments, the log is noisy and the model unavailable, requiring more robust techniques and expert assistance to perform conformance checking. The considerable number of techniques and reduced availability of experts pose an additional challenge to detecting anomalous traces for particular event log scenarios. In this work, we combine the representational power of encoding with a Meta-learning strategy to enhance the detection of anomalous traces in event logs towards fitting the best discriminative capability between common and irregular traces. Our method extracts meta-features from an event log and recommends the most suitable encoding technique to increase the anomaly detection performance. We used three encoding techniques from different families, 80 log descriptors, 168 event logs, and six anomaly types for experiments. Results indicate that event log characteristics influence the representational capability of encodings differently. Our proposed Meta-learning method outperforms the baseline reaching an F-score of 0.73. This performance demonstrates that traditional process mining analysis can be leveraged when matched with intelligent decision support approaches.

Keywords: Anomaly detection · Meta-learning · Encoding · Process mining · Recommendation

The authors would like to thank CNPq (National Council for the Scientific and Technological Development) for their financial support under Grant of Project 420562/2018-4 and 309863/2020-1 and the program "Piano di sostegno alla ricerca 2020" funded by Università degli Studi di Milano.

L. Bellatreche et al. (Eds.): ADBIS 2021 Short Papers, Workshops and Doctoral Consortium, CCIS 1450, pp. 157–168, 2021.
https://doi.org/10.1007/978-3-030-85082-1_15

1 Introduction

Organizations rely on the correct execution of business processes to achieve their goals. However, anomalous instances in event logs are harmful to process quality. This way, stakeholders are interested in detecting and mitigating anomalies so that business processes correspond to their expected behavior. Detecting anomalies is not only beneficial for resource-saving but also to avoid security issues [23]. Process Mining (PM) is an area devoted to extracting valuable information from organizational business processes. Within PM, conformance checking methods are dedicated to finding anomalies. Conformance techniques compare process models and event logs, quantifying deviations and, consequently, identifying anomalies [21]. Traces not complying with the model are interpreted as anomalous, either from a control-flow or data-flow perspective. Although conformance checking supports the recognition of anomalous traces, the methods are model-dependent, hindering their applicability since the model might not be available in many scenarios.

Several approaches have been proposed for anomaly detection in business processes. In [6], the authors use likelihood graphs to model process behavior and support anomaly detection. The method is applicable to control- and data-flow perspectives, although the quality of discovered models limits its performance. Recently, many methods are relying on Machine Learning (ML). In [17], the authors use an autoencoder to model process behavior and detect irregular cases. The authors from [18] use a deep neural network trained to predict the next event. An activity with a low probability score (extracted from the network) is recognized as an anomaly. In [22], the authors use language-inspired trace representations to model process behavior. Cases isolated in the feature space are identified as abnormal.

As pointed by the authors in [4], a suitable encoding technique is crucial for the quality of posterior methods applied to the event log. This way, by finding the appropriate encoding, one can improve the identification of anomalous instances. In other words, a suitable encoding technique allows the best representation of typical behavior and adjusted discriminant capacity of anomalous traces. In this work, we propose a Meta-learning (MtL) strategy to recommend the best encoding for a given event log, maximizing the number of identified anomalies by fitting each particular event log pattern based on its features. MtL has been applied as a recommender system, succeeding in emulating expert decisions [13] for a wide range of applications. Taking advantage of structural and statistical light-weight meta-features from event logs, we propose an MtL approach to suggest the superior encoding technique from a group. Our MtL approach was built using 80 meta-features, trained over 168 event logs (meta-instances) for guiding the superior one of three promising encoding methods (meta-target). Moreover, the proposed approach is easily scalable, allowing the inclusion of additional encoding techniques. Results show the MtL approach outperforms current baselines and can improve the detection of anomalous traces independently of the applied learning algorithm.

The remainder of this paper is organized as follows. Section 2 presents the MtL-based methodology proposed in this paper. Moreover, the section explores the event logs, the extracted features, and the encoding techniques used along with the MtL approach. Section 3 compares our method's performance with two baselines and discusses the implications of the different anomalies. Lastly, Sect. 4 concludes the paper and highlights our contributions.

2 Methodology

This section presents the proposed methodology to enhance anomaly detection in event logs. The method is based on the combination of encoding representational power with MtL as the learning paradigm. We also present the design details and the materials used for experiments.

2.1 Meta-learning for Anomaly Detection in Process Mining

In this work, we investigate encoding methods that boost the performance of traditional algorithms for the anomaly detection task in business processes. For that, our proposed approach relies on MtL. The primary assumption is that the event log characteristics, i.e., descriptors, support the choice of the best encoding method. The best encoding is the technique that produces the highest anomaly detection rates when combined with a given ML-induced model. The boosted capability provided by a particular encoding method relies on the correctly and effectively discriminative capacity to represent traces [4]. However, identifying the best encoding is very tricky, depending on the expert's experience in the particular domain. Here, we follow the assumption that this experience could be emulated using MtL.

Figure 1 presents an overview of our approach. Starting from the event logs, the first step is the *Meta-feature extraction*, which mines descriptors that characterize the event logs. Next, we submit the event logs to encoding techniques. The encoding methods work at the trace level, and the encoded traces serve as input for an ML algorithm aiming to detect anomalies. Hence, we assess the performance metric (namely, F-score) that ranks the encoding algorithms for each event log. This step is called *Meta-target definition*, where each event log is submitted to all encoding methods, and the best encoding (meta-target) is identified. Then, the *Meta-database creation* joins the two previous steps. Here, a database is created using meta-features and the meta-targets extracted from the logs. Consequently, each meta-instance is a set of log descriptors associated with an encoding technique that leverages anomaly detection performance for that event log. Once the meta-database is created, we induce a *Meta-model* in the *Meta-learner* step. The meta-model is the final product of our workflow. Given a new meta-instance (log descriptors), the meta-model indicates the best encoding technique for that event log.

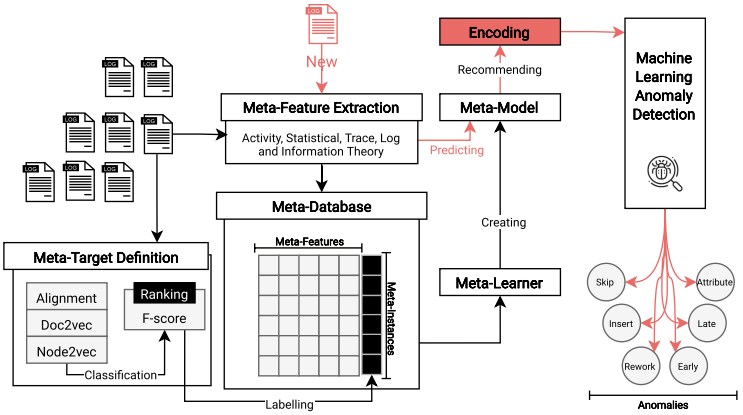


Fig. 1. Overview of the proposed approach.

2.2 Event Logs – Meta-instances

The more instances available, the more representative is the meta-database as it contains more examples of business process behaviors. Experiments on anomaly detection benefit from labeled datasets since one can compute traditional performance metrics to evaluate if anomalous cases are indeed captured. Considering these constraints, we built our meta-database from two groups of synthetic event logs composed of a wide range of behaviors originating from 12 different process models and affected by six types of anomalies.

The first group of event logs was initially presented by Nolle et al. [18] and replicated in [2] and [22]. Six models were generated using the PLG2 tool [8]. PLG2 randomly generates process models representing several complementary business patterns such as sequential, parallel, and iterative control-flows. Moreover, PLG2 allows the configuration of the number of activities, breadth and width, hence, providing a complex set of models that capture diverse behavior. One additional process model, P2P, extracted from [19] was added to the pool. Then, the authors adopted the concept of likelihood graphs [6] to introduce long-term control-flow dependencies. The likelihood graphs are able to mimic complex relations between event to event transitions and attributes attached to these events. This way, the control-flow perspective is constrained by probability distributions that coordinate the model simulation. For instance, an activity may follow another given a probability. The combination of stochastic distributions with a set of process models leverages the similarity between produced event logs and real-world logs. Four event logs were simulated in each process model, generating a total of 28 logs. The final step added anomalies to the traces within the synthetic event logs, which is a traditional practice in related work [5,6]. We applied six anomaly types for all event logs with a 30% incidence: i) Skip: a sequence of 3 or less necessary events is skipped; ii) Insert: 3 or less random activities are inserted in the case; iii) Rework: a sequence of 3 or less necessary

events is executed twice; iv) Early: a sequence of 2 or fewer events executed too early, which is then skipped later in the case; v) Late: a sequence of 2 or fewer events executed too late; vi) Attribute: an incorrect attribute value is set in 3 or fewer events.

The second group of synthetic event logs was proposed by Barbon et al. [4]. The authors also used the PLG2 tool to create five process models representing scenarios of increasing complexity, i.e., a higher number of activities and gateways. Then, the process models were simulated using the *Perform a simple simulation of a (stochastic) Petri net* ProM plug-in[1], producing 1000 cases for each log. As a post-processing step, the same anomalies used for the previous set of logs were applied in this set but with different configurations. The authors implemented four anomaly incidences (5%, 10%, 15% and 20%). Moreover, the dataset contains binary and multi-class event logs, meaning that some logs incorporate normal behavior and only one anomaly type (binary), and some logs contain both normal behavior and all anomalies at the same time (multi-class). The latter configuration is especially challenging given the higher complexity as more behaviors are present in the same log. In total, this set contains 140 event logs.

For all event logs, anomalies sit on the event level, but they can be easily converted to the case level. That is, cases containing events affected by any anomaly are considered anomalous cases. Table 1 shows the event log statistics for all event logs used in this work. As demonstrated, the set of logs presents a significant variation in the number of cases, events, activities, trace lengths, and variants. These characteristics support the creation of a heterogeneous meta-database, increasing business process representability.

Table 1. Event log statistics: each log contains different levels of complexity

Name	#Logs	#Cases	#Events	#Activities	Trace length	#Variants
P2P	4	5k	38k–43k	25	5–14	513–655
Small	4	5k	43k–46k	39	5–13	532–702
Medium	4	5k	28k–31k	63	1–11	617–726
Large	4	5k	51k–57k	83	8–15	863–1143
Huge	4	5k	36k–43k	107	3–14	754–894
Gigantic	4	5k	28k–32k	150–155	1–14	693–908
Wide	4	5k	29k–31k	56–67	3–10	538–674
Scenario1	28	1k	10k–11k	22–380	6–16	426–596
Scenario2	28	1k	26k	41–333	23–30	1k
Scenario3	28	1k	43k–44k	64–348	39–50	1k
Scenario4	28	1k	11k–13k	83–377	1–30	383–536
Scenario5	28	1k	18k–19k	103–406	1–37	637–737

[1] http://www.promtools.org/doku.php.

2.3 Log Descriptors – Meta-feature Extraction

Extracting high-quality descriptors is fundamental for the performance of our meta-model. Moreover, meta-feature extraction should have a low computational cost, otherwise, the MtL pipeline is unjustified. This way, we selected a group of lightweight features that contains reliable representational capacities. To retrieve a multi-perspective view of event logs, we extract features from several process layers: activities, traces, and logs. These features were first proposed in [3], which combines business process features from different sources.

Three subgroups capture activity-level descriptors: all activities, start activities, and end activities. Each subgroup contains 12 features: number of activities, minimum, maximum, mean, median, standard deviation, variance, the 25th and 75th percentile of data, interquartile range, skewness, and kurtosis coefficients. For trace-level descriptors, we extracted features related to trace lengths and trace variants. 29 features compose the trace length group: minimum, maximum, mean, median, mode, standard deviation, variance, the 25th and 75th percentile of data, interquartile range, geometric mean and standard variation, harmonic mean, coefficient of variation, entropy, and a histogram of 10 bins along with its skewness and kurtosis coefficients. Regarding the trace variants, we obtained 11 features: mean, standard variation, skewness coefficient, kurtosis coefficient, the ratio of the most common variant to the number of traces, and ratios of the top 1%, 5%, 10%, 20%, 50% and 75% to the total number of traces. Finally, for log-level descriptors, we extracted the number of traces, unique traces, and their ratio, along with the number of events.

Overall, 80 features were extracted from the event logs. They capture complementary elements of business processes and encode information such as statistical dispersion, probability distribution shape and tendency, and log complexity.

2.4 Encodings – Meta-target Definition

In this work, the application of encoding for anomaly detection in PM is a fundamental step towards building the meta-database. Ultimately, the encodings are the meta-targets associated with log features. Given a log and its meta-features, we associate it with an encoding that maximizes the anomaly detection performance. Therefore, encoding techniques play a major role as they can excel in detecting anomalous instances for certain types of log behaviors. The application of encoding in PM has already been explored by several researches [4,10,15,20]. Barbon et al. [4] extensively evaluated trace encoding methods using feature quality metrics to assess encoding capacity. Moreover, the authors submit the encoding methods to a classification task for anomaly detection. The work proposes the application of three encoding families to event logs: PM-based encoding, word embedding, and graph embedding. The PM-based encodings are conformance checking techniques that compare an event log to a process model, measuring deviance and producing fitness results [21]. Word embeddings can naturally be applied in event logs when considering activities and traces as

words and sentences [2,10]. These techniques rely on context information captured by neural networks' weights when trained for context prediction. Lastly, graph embeddings are techniques that encode graph information, such as nodes, vertices, and their attributes. Graph embeddings are particularly interesting in the PM domain as they can represent process models (with limitations such as not capturing concurrency) and traces, modeling entity links and long-term relations.

Considering the three encoding families presented in [4], we selected one encoding method from each family. This way, we aim to reduce the representative bias and evaluate if there is a relation between encoding families and log behavior. For PM-based encodings, we used alignments as it has been considered the state-of-the-art conformance checking method [9]. Alignments compare the event log and process model and measure the deviations between the two. For that, it relates traces to valid execution sequences allowed by the model. This evaluation unfolds into three types. Synchronous moves are observed when both the trace and model can originate a move. Model-dependent moves are originated only from the model, and log-dependent moves are derived from traces but are not allowed by the model. Synchronous moves represent the expected behavior when model and log executions agree. The alignment technique searches for an optimal alignment, i.e., when the fewest number of the model- and log-moves are necessary. This process, measured by a cost function, produces a fitness value and other statistics regarding the states consumed by the model.

Word embedding techniques in PM have mostly relied upon *word2vec* and *doc2vec* to encode traces. In [2], the authors propose the word2vec encoding in conjunction with One-class Classification to detect anomalies in business processes. The authors in [10] use both word2vec and doc2vec to encode activity and trace information, respectively. In this work, we adopt the doc2vec encoding technique as it has been used to encode traces in similar works. Moreover, doc2vec is an extension of word2vec adapted to documents, independently of their length. Word2vec creates numerical representations for words and, for that, a neural network is trained to reconstruct the linguistic context of words in a corpus [16]. The word embeddings come from the weights of the induced neural network. The main advantage is that words appearing in similar contexts produce similar encoding vectors. However, this method is limited to unique word representations. Doc2vec extends word2vec by adding a paragraph vector in the encoding process [14]. This way, the document context is captured by the encoding.

For the graph embedding family, we employed *node2vec*, another encoding technique built on top of word2vec. Node2vec's primary goal is to encode graph data while maintaining graph structure. Given a graph, node2vec performs random walks starting from different nodes [12]. This process creates a corpus, which is used as input for word2vec. The second-order random walks balance a trade-off between breadth and width, capturing neighbor and neighborhood information. Hence, the method can represent complex neighborhoods given its node exploration approach.

Using the three defined encodings (alignment, doc2vec and node2vec), we performed the *Meta-target definition* step shown in Fig. 1. The goal is to identify which encoding enhances the detection of anomalous traces in event logs. For that, we applied a traditional ML pipeline where each log has its traces divided into an 80%/20% holdout strategy, 80% of traces are used for model inferring while 20% of traces are used for testing, i.e., testing if the trained model can correctly label traces. This process is repeated 30 times. We employed the Random Forest algorithm [7] due to its robustness and wide use in similar works by the ML community. The average F-score is obtained after 30 iterations, and the encoding techniques are ranked according to their average F-score. Thus, the best encoding technique for a given log is the one that produces the highest F-score value in the anomaly detection task. We chose F-score as the ranking metric as it successfully balances different performance perspectives.

2.5 Meta-database and Meta-model

We create the meta-database with the meta-features extracted from the logs and the defined meta-targets. Once the meta-database is built, the meta-learner step occurs. The meta-learner embeds a traditional ML pipeline, that is, the meta-database is submitted to an ML algorithm that infers a meta-model. In this scenario, we also employed the Random Forest algorithm with the same parameters (holdout and iterations). The algorithm produces a meta-model used to recommend the suitable encoding for an event log considering its meta-features.

3 Results and Discussion

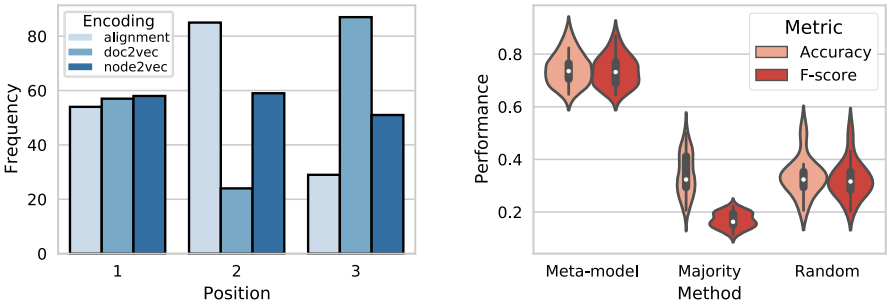
In this section, we present the performance of our approach and compare it with a baseline performance. Moreover, we develop a discussion regarding the impact of the anomalies in the encoding.

3.1 Meta-learning Performance

First, we report the results of the meta-target definition step, i.e., ranking the encodings for each meta-instance. Since we are following a data-driven strategy, it is worth observing the balance regarding data quality. For that, Fig. 2a shows the frequency of the encoding techniques in each position. The ranking is built using the F-scores obtained by each encoding algorithm. This analysis brings insights about the balanced scenario when selecting an encoding technique, illustrating the "no free lunch theorem" [1]. The alignment method was the best encoding for 54 event logs, while doc2vec was optimal for 56 logs and node2vec for 57 logs. These results highlight a balanced distribution between the best-ranked encodings. Regarding individual encoding, we note that node2vec has similar frequencies in all positions. On the other hand, doc2vec frequency in the third position is high, that is, this encoding was the worst for 87 event logs. Overall,

alignments were the most stable method since it appears more frequently in the second position than the others.

Figure 2b reports the performance of our method for the task of recommending the encoding technique that leverages anomaly detection in event logs. Considering the lack of literature in the area, we compare the MtL performance with two baselines: majority and random selection. Majority regards the encoding method with the highest frequency in the first position, hence always recommending the node2vec encoding. Random selection works by arbitrarily selecting one of the possible targets for each event log. From both accuracy and F-score perspectives, our approach outperforms the others with a large advantage. The meta-model obtains an average accuracy of 74% and an average F-score of 0.73. The violin visualization also demonstrates the MtL robustness since the density curve is compressed, i.e., most recordings are near the average mark. The majority approach produced accuracy and F-score averages of 34% and 0.17, respectively. The random method achieves 33% accuracy and 0.32 F-score. The density curves are more stretched, implying less robustness, which is expected given its random nature. It is worth mentioning that these results report the performance for selecting the best encoding method.



(a) Encodings position frequency.

(b) Recommendation performance.

Fig. 2. Encoding ranking extracted from the Meta-target definition step. The ranking is ruled by the F-score obtained by recommending a suitable encoding technique. Given a new event log, the Meta-model recommends the best encoding considering the Meta-features derived from the log. Figure 2b demonstrates the Meta-model performance in comparison with baseline approaches.

3.2 Anomaly Analysis

As introduced in Sect. 2.2, the event logs contain six different anomaly types. Moreover, a subset of the logs is struck by all six anomalies at the same time. Considering the anomaly perspective, Table 2 reports the F-score performance of all three encodings and compares with the MtL approach. Naturally, detecting anomalous instances in logs affected by *all* anomalies is the most difficult task.

Hence, the F-score values are the worst in this scenario. Nonetheless, we observe that MtL reports the highest mean F-score, reaching 0.489. It is followed by alignments (0.468), doc2vec (0.429) and node2vec (0.427). The performance rises considerably in the other anomaly types as the problem is binary in these cases. *Insert*, *rework* and *skip* are the most detectable anomalies because they deeply affect the control-flow perspective of traces, therefore, this behavior change is easily captured by the encodings. For these anomalies, MtL reaches the highest performance values, producing F-score values close to 1. For *rework* and *skip* anomalies, node2vec ties with the MtL approach. Alignment follows the previous approaches, while doc2vec is the worst encoding for these anomalies. The encoding order changes when observing *early* and *late* anomalies. In these scenarios, MtL remains as the best technique, reaching 0.944 F-score for *early* and 0.94 F-score for *late*, but now is followed more closely by doc2vec, which reaches 0.942 and 0.939 for *early* and *late*, respectively. Finally, MtL and doc2vec tie as the best approaches for the *attribute* anomaly, followed by node2vec and alignment. Interpreting performance from the anomaly perspective reinforces the hypothesis that encodings perform differently in different scenarios, that is, log behavior is determinant when choosing the appropriate encoding. Hence, the MtL efficiency in this experiment exposes the influence of event log behavior on the encoding representational power. The results indicate that anomaly detection is enhanced when the relationship between event log descriptors and encodings is appropriately mapped. This mapping is mastered by our proposed MtL method, which outperforms the use of fixed encodings for all event logs.

Table 2. Comparison of anomaly detection performance using fixed encoding methods and MtL recommendation. Mean and standard deviation (in parenthesis) F-score values are reported for each anomaly type. Bold values indicate the best method for each anomaly.

Encoding	All	Attribute	Early	Insert	Late	Rework	Skip
Alignment	0.468 (0.15)	0.919 (0.04)	0.931 (0.04)	0.975 (0.02)	0.933 (0.03)	0.97 (0.02)	0.98 (0.02)
doc2vec	0.429 (0.21)	**0.931** (0.03)	0.942 (0.03)	0.932 (0.03)	0.939 (0.03)	0.943 (0.03)	0.945 (0.03)
node2vec	0.427 (0.11)	0.926 (0.04)	0.925 (0.04)	0.985 (0.01)	0.928 (0.04)	**0.99** (0.01)	**0.988** (0.01)
MtL	**0.489** (0.15)	**0.931** (0.03)	**0.944** (0.03)	**0.989** (0.01)	**0.94** (0.03)	**0.99** (0.01)	**0.988** (0.01)

We compared the F-score obtained by classifying all event logs using statistical analysis grounded on the non-parametric Friedman test to determine any significant differences between the usage of a unique encoding technique and meta-recommended ones. We used the post-hoc Nemenyi test to infer which differences are statistically significant [11]. As Fig. 3 shows, differences between populations are significant. Furthermore, there are no significant differences within two groups: doc2vec and node2vec, and node2vec and alignment. All other differences are significant. Thus, MtL for recommending individual encoding methods to maximize the predictive performance achieved superior results statistically different from the usage of only one encoding. In other words, the performance obtained using MtL was statistically superior to a single encoding technique.

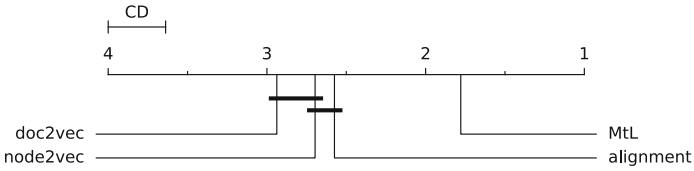


Fig. 3. Nemenyi post-hoc test (significance of $\alpha = 0.05$ and critical distance of 0.361) considering the F-score obtained from all event log classifications.

4 Conclusion

Organizations are interested in detecting anomalous instances in their business processes as a method to leverage process quality, avoid resource waste, and mitigate security issues. In this work, we proposed to combine encoding techniques with MtL to enhance the detection of anomalous traces in event logs. Our strategy relies on a powerful set of meta-features extracted from the event logs. We showed its viability by recommending the best encoding technique with an F-score of 0.73. MtL boosted the anomaly detection by fitting the optimal encoding technique for each event log, statistically outperforming the usage of a single encoding technique. Moreover, our method is highly scalable, which can lead to incremental advancements in the area. For future works, we plan to include more encoding techniques and propose additional features for the meta-feature extraction step.

References

1. Adam, S.P., Alexandropoulos, S.-A.N., Pardalos, P.M., Vrahatis, M.N.: No free lunch theorem: a Review. In: Demetriou, I.C., Pardalos, P.M. (eds.) Approximation and Optimization. SOIA, vol. 145, pp. 57–82. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12767-1_5
2. Barbon, S., Jr., Ceravolo, P., Damiani, E., Omori, N.J., Tavares, G.M.: Anomaly detection on event logs with a scarcity of labels. In: 2020 2nd International Conference on Process Mining (ICPM), pp. 161–168 (2020)
3. Barbon, S., Jr., Ceravolo, P., Damiani, E., Tavares, G.M.: Using meta-learning to recommend process discovery methods (2021). https://arxiv.org/abs/2103.12874
4. Barbon Junior, S., Ceravolo, P., Damiani, E., Marques Tavares, G.: Evaluating trace encoding methods in process mining. In: Bowles, J., Broccia, G., Nanni, M. (eds.) DataMod 2020. LNCS, vol. 12611, pp. 174–189. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-70650-0_11
5. Bezerra, F., Wainer, J.: Algorithms for anomaly detection of traces in logs of process aware information systems. Inf. Syst. **38**(1), 33–44 (2013)
6. Böhmer, K., Rinderle-Ma, S.: Multi-perspective anomaly detection in business process execution events. In: Debruyne, C., et al. (eds.) OTM 2016. LNCS, vol. 10033, pp. 80–98. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_5
7. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
8. Burattin, A.: PLG2: multiperspective processes randomization and simulation for online and offline settings (2015)

9. Carmona, J., van Dongen, B.F., Solti, A., Weidlich, M.: Conformance Checking. Relating Processes and Models, Springer, Cham (2018)
10. De Koninck, P., vanden Broucke, S., De Weerdt, J.: act2vec, trace2vec, log2vec, and model2vec: representation learning for business processes. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 305–321. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_18
11. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006). http://dl.acm.org/citation.cfm?id=1248547.1248548
12. Grover, A., Leskovec, J.: Node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, pp. 855–864. ACM, New York (2016)
13. He, X., Zhao, K., Chu, X.: AutoML: a survey of the state-of-the-art. Knowl. Based Syst. **212**, 106622 (2021)
14. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, Beijing, China, 22–24 June 2014, vol. 32, pp. 1188–1196. PMLR (2014)
15. Leontjeva, A., Conforti, R., Di Francescomarino, C., Dumas, M., Maggi, F.M.: Complex symbolic sequence encodings for predictive monitoring of business processes. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 297–313. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_21
16. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013)
17. Nolle, T., Luettgen, S., Seeliger, A., Mühlhäuser, M.: Analyzing business process anomalies using autoencoders. Mach. Learn. **107**(11), 1875–1893 (2018)
18. Nolle, T., Luettgen, S., Seeliger, A., Mühlhäuser, M.: BINet: multi-perspective business process anomaly classification. Inf. Syst. 101458 (2019)
19. Nolle, T., Seeliger, A., Mühlhäuser, M.: BINet: multivariate business process anomaly detection using deep learning. In: Weske, M., Montali, M., Weber, I., vom Brocke, J. (eds.) BPM 2018. LNCS, vol. 11080, pp. 271–287. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98648-7_16
20. Polato, M., Sperduti, A., Burattin, A., Leoni, M.d.: Time and activity sequence prediction of business process instances. Computing **100**(9), 1005–1031 (2018)
21. Rozinat, A., van der Aalst, W.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2008)
22. Tavares, G.M., Barbon, S.: Analysis of language inspired trace representation for anomaly detection. In: Bellatreche, L., et al. (eds.) TPDL/ADBIS -2020. CCIS, vol. 1260, pp. 296–308. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-55814-7_25
23. van der Aalst, W., de Medeiros, A.: Process mining and security: detecting anomalous process executions and checking process conformance. Electron. Notes Theor. Comput. Sci. **121**, 3–21 (2005). Proceedings of the 2nd International Workshop on Security Issues with Petri Nets and Other Computational Models (WISP 2004)