

Workarounds Beware: Comparing Machine Learning Algorithms with Active Learning

Isa van Alphen¹, Amber Meulenbeld²[0000–0002–9571–2179], Inge van de Weerd¹[0000–0003–3420–6923], Saskia Haitjema²[0000–0001–5465–4868], Hajo A. Reijers¹[0000–0001–9634–5852], Wouter van der Waal¹[0000–0001–6514–7570]

¹ Utrecht University, Utrecht, The Netherlands

{i.a.e.vanalphen,g.c.vandeweerd,h.a.reijers,w.g.vanderwaal}@uu.nl

² University Medical Center Utrecht, Utrecht, The Netherlands

{a.meulenbeld-5,s.haitjema}@umcutrecht.nl

Abstract. Workarounds occur in many organizations, including hospitals, where users intentionally create deviations from the prescribed processes. Identifying such workarounds can provide valuable insights for improving hospital processes. To discover more workarounds with less effort required of domain experts, we use event logs and machine learning for workaround mining. We compare four machine learning algorithms, Logistic Regression, Support Vector Machines, Naive Bayes and Random Forests, trained using Active Learning to enable domain experts to focus solely on the most informative traces. We evaluate this approach in a case study at a hospital blood transfusion laboratory, where two domain experts independently labeled 25 traces as either being a workaround or a normative trace to train the algorithms. The results show that Support Vector Machines and Random Forests achieve the highest recall (0.883 and 0.763, respectively), outperforming Logistic Regression (0.400) and Naive Bayes (0.395). These findings indicate that Support Vector Machines and Random Forests are promising for discovering workarounds, and that Active Learning-based machine learning can support organizations to efficiently discover process deviations and gain actionable insights without extensive manual labeling.

Keywords: Workarounds · Machine Learning · Active Learning · Process Mining

1 Introduction

Organizations often establish structured processes to guide the activities of their employees, designed by management, based on policies and regulations [9]. However, when such processes run into obstacles, users may create workarounds [9]. These are deviations that lead to the same goal as the primary process, but would not be hindered by the underlying obstacle. This action is an intentional choice by the user [9]. By discovering these workarounds, it is possible to identify problems and make improvements accordingly in processes and systems.

Traditionally, workarounds have been discovered through interviews and observations [4]. However, interviews are limited in their ability to fully identify all workarounds, and observations provide only partial insight [7]. To solve these issues, recent work introduced the Semi-automated WORKaround Detection (SWORD) framework [17], shifting workaround detection towards a quantitative approach. Using event logs that record sequences of events, individual traces can be analyzed using patterns that indicate workarounds occurring within individual traces, which allows the discovery of new workarounds.

Additionally, traditional qualitative methods require a lot of time from domain experts [19]. Although the data-driven approach of the SWORD framework already decreases the effort required of experts, it is still difficult to evaluate for them. The output of the framework consists of multiple values for each trace that can be hard to interpret for domain experts [17]. To make optimal use of the limited time of experts, recent work applied Logistic Regression and Active Learning to limit the number of values and further improve the efficiency of the SWORD framework [16]. Active Learning is a machine learning approach where the algorithm improves by actively selecting the most informative unlabeled instances. It reduces the number of labeled examples needed [15].

While Active Learning proved useful, previous studies only tested Logistic Regression [16], leaving it unclear which machine learning algorithm is most useful for workaround mining. In a case study at the Central Diagnostic Laboratory of the University Medical Center Utrecht, we compare four algorithms. Although the general setup follows van der Waal et al. [16], this study differs by training four separate algorithms to identify workarounds using interview data, rather than focusing on a single algorithm. This shifts the focus from evaluating Active Learning to evaluating how we can discover more workarounds.

With our data-driven approach, the practical relevance of this study is twofold: First, with our data-driven approach and Active Learning, we limit the effort experts need to spend on labeling data to only 30 minutes to train a single workaround detection model. Second, the comparison of machine learning algorithms in this context shows that some models can detect many more workarounds than Logistic Regression in the same time. From a scientific perspective, the differences in how well workarounds can be detected between the various models provide new insights into the detection of workarounds and the optimization of the SWORD framework [17].

This study is structured as follows: Section 2 provides an overview of the technical background. In Section 3, the SWORD framework is described, and the training and evaluation of the algorithms are discussed. Section 4 reports on the performance of these algorithms through a case study. Finally, in Section 5, we discuss the findings and limitations, followed by the conclusion.

2 Technical Background

Workaround mining is a subset of process mining that focuses on the detection and analysis of process deviations based on event logs [1]. As described by Van

der Waal et al. [16], workaround mining has two primary objectives: discovering new classes of workarounds and detecting new cases that are based on these classes. Workarounds are deviations that lead to the same goal as the primary process, but do not address the underlying obstacle. This action is a deliberate choice by the user, as opposed to an unintentional action [9]. Traditionally, workarounds were detected by observations and interviews. This approach is limited when only one researcher is performing these qualitative methods due to a single perspective [6]. This increases the risk of researcher bias, potentially leading to unbalanced or incomplete data [20]. Furthermore, interviews do not fully identify all workarounds [7] and require a lot of effort from domain experts [16].

One of the first approaches to detect workarounds quantitatively is introduced by Weinzierl et al. [19]: with seven patterns, they developed a neural network approach for workaround detection. Using artificial data, they demonstrate that this approach can discover workarounds in multiple contexts. However, as this approach relies on neural networks, it requires a lot of expert-labeled data, making applications on real data challenging [19].

Both the approaches by Weinzierl et al. [19] and Beerepoot et al. [5] demonstrate that workarounds can be found by discovering patterns. Building on this pattern-based approach, the SWORD framework uses 22 patterns to identify workarounds from an event log [17]. Instead of using a neural network approach, this framework uses patterns with simpler data requirements rather than a full event log. This does not require prior labeling of data, and therefore reduces the time and effort needed from experts. Moreover, it also enables the discovery of workarounds that are not similar to those already present in the data [17]. In addition, there is an extended version of the SWORD framework, the LongSWORD [12], where two additional cross-case patterns are introduced. These patterns capture workarounds that span multiple instances rather than focusing on single cases.

To further decrease the effort required from domain experts, Van der Waal et al. [16] use Active Learning to automatically classify traces as workarounds. In contrast with Passive Learning, which requires many labeled samples [3], Active Learning takes a more adaptive approach, iteratively selecting the unlabeled data points with the highest potential information value for manual labeling. This achieves high model performance with fewer labeled samples, thereby reducing the cost of labeling data beforehand [3, 15]. As this study applies Active Learning using an Uncertainty Sampling strategy, only fitting algorithms are considered: Logistic Regression, Random Forests, Support Vector Machines and Naive Bayes.

To measure the performance of machine learning models in workaround detection tasks, there are four possible metrics: *accuracy*, *precision*, *recall*, and *F₁ score* [16]. We consider *Recall* to be the most important in the context of workaround mining. Since domain expert evaluation is required to validate any identified trace, [16, 17], a trace falsely identified as a workaround will still be filtered out by expert evaluations. However, a workaround trace that has been missed will never be found.

3 Method

In this section, we first explain the SWORD framework. Second, we explain how the selected machine learning algorithms are trained using an Active Learning approach based on Uncertainty Sampling, including the role of expert sessions in labeling. Finally, we outline the evaluation of the algorithms' performance.

3.1 SWORD Framework

Conform Section 2, we used the SWORD framework as a basis for workaround detection, which makes use of specific patterns that can indicate workarounds [17]. For every pattern, each case got a score that indicated how strongly it deviates from the norm in the data. In general, scores further from 0 indicated stronger deviations. While this resulted in a ranking of all traces per pattern, the scores between patterns could not be directly compared. Instead, we used the output from the SWORD framework as input for the various algorithms.

While there were 22 patterns available in the SWORD framework, not all may be applicable based on the data available. Usually, only a selection will be used within a single case. For details, we refer to the base framework as described in [17] and implementation details, including code in [18].

3.2 Training

As discussed in Section 2, Logistic Regression, Random Forests, Naive Bayes and Support Vector Machines were compared, each trained using Active Learning with Uncertainty Sampling. In each iteration, the trace with the highest uncertainty was selected for training. For Logistic Regression, Random Forests and Naive Bayes, the trace closest to 0.5 was used for training; for Support Vector Machines, the trace closest to 0 was used to train the algorithm.

Training was based on sessions with two domain experts. During each session, the trace with the highest uncertainty was shown and classified as normal or workaround, and this label was used to retrain the algorithm. For the first algorithm, we performed an initial session where the first expert labeled a set number of traces. The same number was used for all other algorithms, with the second expert labeling an equal number too. In later sessions, labeling was expected to take less time as the experts would become faster in understanding traces. This means every algorithm got the same amount of labeled traces, ensuring that the evaluation focused on differences in learning performance rather than differences in expert time or labeling efficiency. If multiple traces had high uncertainty, one was selected at random.

3.3 Evaluation

In this study, domain experts labeled instances as either workaround or normal trace. Identified workaround types were used to formulate rules together with

an expert to generate labels for the test dataset. Each discovered workaround type was linked to a specific rule that enabled the identification of new instances of that same type. When experts encountered a workaround trace that was not captured by an existing rule, a new rule was defined to describe this behavior. The resulting labeled dataset reflected the types of workarounds the algorithm is expected to recognize, making it suitable for evaluating precision and recall.

Because the scores were limited to the workaround types for which rules exist, this method does not produce the true precision and recall values. Some unlabeled traces may still contain unnoticed workarounds, so precision represents a lower bound, while the remaining scores should only be considered an approximation. However, it provides a reasonable approach to the algorithm’s ability to detect known workarounds. More importantly, this strategy provides a practical solution to the lack of large, labeled datasets, as manually labeling traces would require unreasonable time and effort from experts.

4 Case Study

To compare the algorithms, we performed a case study at the University Medical Center Utrecht (UMCU). This is an academic medical center with more than 12,000 employees who take care of almost 230,000 patients each year. Our project was approved by the Medical Research Ethics Committee (MREC) Ned-Mec policy (number 20/1055). To store patient data, the UMCU uses HiX³ as its Hospital Information System (HIS). The data of the blood transfusion laboratory is highly structured compared to the more practical wards of the hospital. Therefore, data quality is high, making this a good fit for a data-oriented study. To evaluate the algorithms, two experts from the blood transfusion laboratory were involved. Expert 1 is a Hemovigilance Officer, responsible for monitoring and improving safety and quality throughout the entire blood transfusion chain. Expert 2 is a senior clinical chemist and blood transfusion specialist with managerial and educational responsibilities at the UMCU. Both have extensive knowledge of the processes and possible workarounds that can occur. In the rest of this section, the case study design is explained, including data collection, constructing, training, and evaluating the algorithms.

4.1 Data Preparation

The dataset initially contained over 1,000,000 rows, many of which were irrelevant due to legacy data from an old system. We are analyzing all traces where at least one event occurred in 2024, as this is the most recent period for which data are available for the entire year. As the dataset contains personal patient data, the raw event log is not publicly available. However, the direct output from the SWORD framework and all subsequent analysis scripts are publicly available ⁴.

³ <https://www.chipsoft.com/en/solutions/electronic-health-record-hix/>

⁴ <https://anonymous.4open.science/r/analysis-code-F164/>

With the available data, an event log was created using the columns *CaseId*, *Event*, *Timestamp*, and *Resource*. *CaseId* was populated from “bbag_Id”, indicating the id of the blood bag which goes through the blood transfusion process. *Event* is populated from the names of all timestamp-related columns, *Timestamp* from their values, and *Resource* indicates the employee associated with each event. Rows with NULL or NA timestamps were removed, as they did not represent valid events. This resulted in an event log containing the following events: Production, Receipt, Creation, Verification, Return, Check out, Start of the transfusion, End of the transfusion, Status, Last update, Expiration and Discontinuation. In total, this event log contained 224,691 events related to 24,194 cases. Table 1 shows an anonymous sample of a case.

Table 1. Anonymous sample of a case

CaseID	Activity	Timestamp	Resource
13438831	Production	DD-MM-2024 16:02	NA
13438831	Receipt	DD-MM-2024 14:18	NA
13438831	Creation	DD-MM-2024 14:18	325
13438831	Verification	DD-MM-2024 14:59	12
13438831	CheckOut	DD-MM-2024 14:43	153
13438831	TransfusionStart	DD-MM-2024 14:51	NA
13438831	TransfusionEnd	DD-MM-2024 23:45	NA

4.2 Algorithms Construction

To construct the algorithms, we applied the SWORD framework to get the pattern score. Data requirements vary across SWORD patterns [17]. For example, the "Activity frequency out of bounds" requires an activity in the event log, while the "Number of resources" requires a participant. By considering these requirements, we applied the following patterns of the SWORD framework to see how they differ between traces:

- Activity frequency out of bounds
- Occurrence of directly repeating activity
- Number of resources
- Relative number of resources
- Delay between start of trace and event
- Occurrence of activity outside of time period
- Duration of a trace out of bounds
- Time between activities out of bounds

4.3 Algorithms Training

Using Active Learning, the algorithms were separately trained by two domain experts from the blood transfusion laboratory by selecting the most informative

cases, i.e. the cases with the highest uncertainty. As an initial step, both experts first labeled six traces: the three traces with the highest pattern score and the three traces with the lowest pattern score. Subsequently, within a 30-minute session for each algorithm, they labeled 19 additional traces, which, combined with earlier labels, formed the training data for all algorithms. As the experts disagreed on one trace for each algorithm, the Active Learning approach selected partially different traces for each expert, causing them to evaluate slightly varied traces. In total, 42 unique traces were classified per algorithm; only traces where both experts agreed were added to the training set, while disagreements were excluded. Table 2 summarizes the classification outcomes per algorithm, including individual expert labels, time spent on labeling and their resulting consensus.

Table 2. Expert labeling and consensus per algorithm. Due to a partial overlap in evaluation between experts, the number of consensus traces is lower than their sum.

Algorithm	Source	Workaround	Normative	Time spent on labeling
Logistic Regression	Expert 1	6	19	27 minutes
	Expert 2	9	16	30 minutes
	Consensus	10	32	
Support Vector Machines	Expert 1	8	17	24 minutes
	Expert 2	11	14	17 minutes
	Consensus	14	28	
Random Forests	Expert 1	11	14	25 minutes
	Expert 2	17	8	16 minutes
	Consensus	23	19	
Naive Bayes	Expert 1	6	19	25 minutes
	Expert 2	8	17	29 minutes
	Consensus	9	33	

4.4 Workarounds

In the UMCU blood transfusion laboratory, the process a blood bag goes through is shown in Table 1. This example shows the normative process necessary to administer blood to a patient, starting from *Production* outside the hospital, then the first *Receipt* and *Creation* of the registration in the hospital, various checks before the actual *TransfusionStart* and *TransfusionEnd*, finishing with the final administration steps in *LastUpdate*. Although this sequence represents the intended workflow, our analysis shows that the process is not always executed accordingly, and workarounds are created. These mainly occur in the transfusion phase, where manual registration and time pressure play a larger role.

A workaround must occur when a required step is missing. Not all steps are required for each transfused component, we find three strict rules related to missing events: 1) Manual label verification is logged specifically for erythrocyte (EC) units, 2) a start time of the transfusion is given, and 3) an end time of the transfusion is registered. Note that for the last one, if the transfusion is not

explicitly ended in the system, the system automatically assigns a default end time of 23:45. As such, the experts considered any occurrence of this exact time an indication that the transfusion was completed, but registration was skipped.

In addition, timing constraints related to transfusion start times and duration also reveal workarounds. These time requirements differ strongly per type of blood component. As an overview, Table 3 shows the discovered workaround rules to identify similar occurrences in the event log.

Table 3. Workaround rules at the UMCU blood transfusion laboratory

Type	Rule: There is a workaround if...
Missing event	bbag_TransfusionStartTime is missing.
	bbag_TransfusionEndTime is exactly 23:45.
	bbag_VerificationTime is missing for EC units.
Transfusion duration	transfusion exceeds 2 hours for TC, OMNI, GCB, or FFP units.
	transfusion lasts more than 6 hours for EC units.
	transfusion lasts under 30 minutes for EC or OMNI units.
	more than 1.5 hours pass between checkout and transfusion start.
	transfusion starts over 5 days after return for OMNI or FFP units.
	transfusion starts more than 1.5 hours after checkout or lasts more than 2 hours for GCB units.

4.5 Algorithms Evaluation

After training all algorithms, the test set was constructed using the ground truth rules. Recall was then calculated for each algorithm using this ground truth and labeled traces from both experts. The results are shown in Table 4.

The results show a clear distinction in recall between the algorithms. The first group concerns algorithms with relatively low recall. Logistic Regression and Naive Bayes performed poorly in identifying true workarounds (0.400 and 0.395, respectively), indicating that many workarounds were not detected. Both algorithms therefore capture only a small part of workarounds.

The second group involves algorithms with higher recall. Support Vector Machines achieve the highest recall among the evaluated algorithms (0.883). Random Forests also show a strong recall score (0.763). As such, both capture a large amount of true workarounds.

Table 4. Combined scores of the evaluated algorithms

Algorithm	Recall	Precision	F1	Accuracy
Logistic Regression	0.400	0.470	0.432	0.441
Support Vector Machines	0.883	0.524	0.657	0.511
Random Forests	0.763	0.571	0.653	0.569
Naive Bayes	0.395	0.466	0.428	0.437

5 Discussion and Conclusion

In this section, the results are interpreted in relation to existing literature, followed by a discussion of the implications, the limitations of the study, and suggestions for future work.

In this study, we analyzed the effectiveness of Logistic Regression, Support Vector Machines, Random Forests, and Naive Bayes in detecting workarounds in a healthcare laboratory setting. As shown, Support Vector Machines and Random Forests produced high recall scores of 0.883 and 0.763, respectively. Support Vector Machines better capture complex patterns separating workarounds from standard cases, while the tree-based structure of Random Forest handles variability more effectively, both contributing to their high recall scores. While recall is the primary performance metric, both algorithms also achieve higher precision (0.524 and 0.571, respectively) than Logistic Regression (0.470) and Naive Bayes (0.466), showing that high recall is not achieved at the cost of many false positives.

In contrast, Naive Bayes produced a low recall score of 0.395, likely due to its strong independence assumptions that are unlikely to hold in this context. Logistic Regression similarly achieved a low recall score of 0.400, suggesting its linear decision boundary fails to fully capture the underlying relationships in the data. While these results highlight the limitations of both algorithms within this study, the results of Logistic Regression also differ substantially from findings reported in earlier research. In particular, van der Waal et al. [16] reported a precision of 0.670 and a recall of 0.850, whereas this study achieved a precision of 0.470 and a recall of 0.400. Consequently, the F1 score obtained in this study (0.432) is lower than the score reported by van der Waal et al. [16] (0.750) and in the lower end of the range reported for the neural network approach by Weinzierl et al. [19], which varies between 0.330 and 0.830. These variations may be explained by differences in the datasets and process characteristics. Our dataset contains only real data without synthetically introduced workarounds [19], while its more structured nature compared to van der Waal et al. [16] means standard traces follow a more consistent pattern, making workarounds stand out more clearly. Furthermore, prior research by Kim and Kim [11] shows that Logistic Regression performance can be strongly affected by the ratio of positive and negative training examples, which may differ between the studies.

While no directly related work on workaround classification exists, comparisons to other domains show that high scores for Random Forest and low scores for Naive Bayes are common in medical prediction [2, 10, 13], electricity theft detection [14], and sentiment analysis [8]. However, Logistic Regression typically scores high in these domains but ranks low here, possibly due to strongly correlated attributes [2]. In contrast, Support Vector Machines rank higher than usual, likely because they handle the high-dimensional output of the framework well [2]. This suggests that workaround mining benefits from algorithms that capture complex relationships between patterns.

5.1 Implications

This study has several implications for both research and practice. From a research perspective, the findings provide insights that extend beyond the performance of the algorithms. The strong results of Support Vector Machines and Random Forests indicate that these algorithms are not only effective in detecting known workarounds but can also be used to uncover workarounds that were previously unknown. Because these algorithms learn from combinations of SWORD pattern scores, their predictions reveal how certain patterns tend to occur together within traces. When multiple patterns appear together in workaround traces, the algorithms learn that these combinations are important, which provides an indication of overlap between patterns. This is underlined by the low performance of the Logistic Regression algorithm, which cannot handle such correlation well [2]. In addition, when the algorithms assign a high workaround score to traces that were not identified during the interviews, this often results from new combinations of patterns that are similar to earlier workaround traces. Such insights can contribute to a deeper understanding of the relationships between patterns in the SWORD framework, helping to streamline the framework by revealing where patterns overlap or capture similar workaround behavior.

What seems especially useful for practice is the use of the Active Learning approach which demonstrates that meaningful algorithm performance can be obtained with minimal expert input, requiring only 19 labeled traces in 30 minutes. Given the limited availability and high cost of domain experts, this finding shows that machine learning algorithms can be trained efficiently without extensive labeling efforts, which would otherwise take at least 20 hours in qualitative studies [4, 6]. This efficiency makes it suitable for organizations to continuously monitor processes or update algorithms as new workaround patterns show up, proving a scalable and practical solution to manage processes using workaround mining.

5.2 Limitations

This study also contains limitations that should be discussed. First of all, the dataset used in this study is specific to the UMCU blood transfusion laboratory, which may limit the generalizability of the findings to other healthcare settings. Previous research by Weinzierl et al. [19] has shown that the performance of methods can be highly context-dependent, with factors such as process structure, variability, and the presence of deviations strongly influencing outcomes. This suggests that the effects observed in this study may differ in other departments or hospitals. However, this is somewhat mitigated by published research able to find workarounds using the same method in a different context [16]. Especially contexts with less well-structured data may give different results. As data dimensions would not change strongly due to the selection of patterns in the SWORD framework, the biggest resulting difference will likely be the correlation between the patterns. Since it is highly influenced by such correlation, we would expect that especially Logistic Regression could perform a lot better

in different contexts [2]. However, future research should validate the findings in other healthcare environments to assess their broader applicability and to explain the remaining differences in results.

Second, the reliance on expert-defined rules for identifying workarounds may be biased, as the rules may not capture all possible workarounds and could lead to an incomplete representation of the ground truth. The quality of the expert labeling can vary, potentially affecting the training and evaluation of the algorithms. In this study, however, the rules were created based on interviews with two independent domain experts, which helps to improve the reliability of the labeling. Future studies could explore alternative methods for identifying workarounds, such as unsupervised machine learning techniques.

Finally, similar to Van der Waal et al. [16], the rule-based approach of workaround detection serves as an approximation for the true precision and recall. As we cannot be exhaustive in classifying traces as workarounds, given the size of the dataset, it is not possible to obtain the exact values. However, as we have found that experts learn to classify individual traces in this context quite quickly, future studies could attempt to let experts classify a random subset of the data for a more traditional testing set.

5.3 Future Work

This study shows promising directions for future work. Although Active Learning mitigates the challenges posed by the relatively small size of the expert-labeled dataset used, increasing the dataset size could further improve algorithm performance, particularly for detecting less frequent workarounds. Future research could investigate how algorithm performance scales with larger datasets and whether additional data reduces variability and improves algorithm stability.

Another direction for future work is to evaluate the generalizability of the algorithms in different healthcare contexts. Testing these algorithms in other hospital departments or in different hospitals would provide insights into how well they handle process variations and help identify factors that influence performance across settings, ultimately supporting broader applicability in both healthcare and non-healthcare environments.

Acknowledgements We would like to thank Dr. Karen de Vooght and Ingrid Molina Rioseco for their time and contribution to training the models. This research was funded by the Bakkenist Foundation.

References

1. van der Aalst, W.: Process mining. *Communications of the ACM* **55**(8), 76–83 (2012)
2. Akkaya, B., Çolakoglu, N.: Comparison of multi-class classification algorithms on early diagnosis of heart diseases (2019)

3. Al-Tamimi, A.K., Bani-Isaa, E., Al-Alami, A.: Active learning for arabic text classification. In: 2021 International Conference on Computational Intelligence and Knowledge Economy. pp. 123–126 (2021)
4. Beerepoot, I., Koorn, J., van de Weerd, I., van den Hooff, B., Leopold, H., Reijers, H.A.: Working Around Health Information Systems: The Role of Power. In: Fortieth International Conference on Information Systems ICIS (2019)
5. Beerepoot, I., Lu, X., Van de Weerd, I., Reijers, H.: Seeing the signs of workarounds: A mixed-methods approach to the detection of nurses' process deviations (2021)
6. Beerepoot, I., van de Weerd, I.: Prevent, Redesign, Adopt or Ignore: Improving Healthcare using Knowledge of Workarounds. In: Proceedings of the 26th European Conference on Information Systems (2018)
7. van Beijsterveld, J.A.A., van Groenendaal, W.J.H.: Solving misfits in ERP implementations by SMEs. *Information Systems Journal* **26**(4), 369–393 (2015)
8. Dangi, D., Dixit, D.K., Bhagat, A.: Sentiment analysis of COVID-19 social media data through machine learning. *Multimedia Tools and Applications* **81**(29), 42261–42283 (2022)
9. Ejnefjäll, T., Ågerfalk, P.J.: Conceptualizing workarounds: Meanings and manifestations in information systems research. *Communications of the Association for Information Systems* pp. 340–363 (2019)
10. Kanitkar, A., Joshi, V., Karwa, Y., Gindi, S., Kale, G.V.: Comparison of Machine Learning Algorithms for Cervical Abnormality Detection. In: Twelfth International Conference on Contemporary Computing. pp. 1–6 (2019)
11. Kim, J., Kim, J.: The impact of imbalanced training data on machine learning for author name disambiguation. *Scientometrics* **117**(1), 511–526 (2018)
12. Löhr, B., Bartelheimer, C., Köhne, F., Nordlohne, S., Alile, D., Latten, A.: Forging the LongSWORD: Exaptation and Enhancement of the SWORD Framework for Workaround Detection. In: Business Process Management Workshops. pp. 319–331 (2024)
13. Rajput, D.S., Basha, S.M., Xin, Q., Gadekallu, T.R., Kaluri, R., Lakshmana, K., Maddikunta, P.K.R.: Providing diagnosis on diabetes using cloud computing environment to the people living in rural areas of India. *Journal of Ambient Intelligence and Humanized Computing* **13**(5), 2829–2840 (2021)
14. Saqib, S.M., Mazhar, T., Iqbal, M., Shahazad, T., Almogren, A., Rehman, A.U., Hamam, H.: Enhancing electricity theft detection with ADASYN-enhanced machine learning models. *Electrical Engineering* **107**(8), 10525–10542 (2025)
15. Settles, B.: Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2009)
16. van der Waal, W., van de Weerd, I., Haitjema, S., Kappen, T., Reijers, H.A.: Whetting the SWORD: Detecting Workarounds by Using Active Learning and Logistic Regression. In: Proceedings of the 57th Hawaii International Conference on System Sciences (2024)
17. van der Waal, W., van de Weerd, I., Beerepoot, I., Lu, X., Kappen, T., Haitjema, S., Reijers, H.A.: Putting the SWORD to the Test: Finding Workarounds with Process Mining. *Business & Information Systems Engineering* **67**(2), 171–190 (2024)
18. van der Waal, W., van de Weerd, I., Reijers, H.A.: SWORD Extractor Pulling Workarounds out the Eventlog. In: ICPM Tool Demonstration Track (2024)
19. Weinzierl, S., Wolf, V., Pauli, T., Beverungen, D., Matzner, M.: Detecting temporal workarounds in business processes – a deep-learning-based method for analysing event log data. *Journal of Business Analytics* **5**(1), 76–100 (2022)
20. Zahle, J.: Bias and debiasing strategies in qualitative data collection. *Philosophy of Science* **92**, 1–27 (2024)