What Makes Usability Smells Detectable? On the Role of Representation in Interaction Log Analysis

Michelle Xiao-Lin Foo
Siemens AG
Erlangen, Germany
LMU Munich
Munich, Germany
michelle.foo@siemens.com

Sven Mayer
TU Dortmund University
Dortmund, Germany
Research Center Trustworthy Data Science and Security
Dortmund, Germany
info@sven-mayer.com

Abstract

Automated usability evaluation using interaction logs offers a scalable way to identify potential usability problems. However, it remains unclear how different representations of interaction data influence the detection of usability issues. In this paper, we investigate usability smell detection from a representation-centric perspective. Using the AMUSED dataset, we decompose interaction logs into behavioral, structural, and contextual representations and evaluate their contribution to task-level smell detection with smell-specific gradient boosting models. We further compare learned models with rule-based detectors derived from prior work. Our results show that usability smells are representation-dependent and context-sensitive: no single representation performs best across all smells or applications and combining representations is beneficial only selectively. Learned models generally achieve more balanced precision and recall than rule-based detectors, while rule-based approaches remain useful for structurally explicit smells but depend strongly on available telemetry. These findings suggest that representation design should be treated as a central design decision in log-based automated usability smell evaluation.

CCS Concepts

• **Human-centered computing** → **Usability testing**.

Keywords

usability testing, usability smells, automated usability smell detection

ACM Reference Format:
Michelle Xiao-Lin Foo and Sven Mayer. 2026. What Makes Usability Smells Detectable? On the Role of Representation in Interaction Log Analysis. In *Proceedings of Mensch und Computer 2026 (MuC '26), August 30–September 02, 2026, Duisburg, Germany*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3820253.3831380

1 Introduction

Usability testing can provide rich insight into usability problems, but it is often resource-intensive and difficult to scale [4, 5]. Interaction logs offer a scalable basis for automated usability evaluation [18]. Building on this idea, Garrido et al. [8] formalized recurring interaction log patterns as *usability smells*: indicators of potential usability problems. Existing approaches include rule-based detectors that encode predefined smell patterns [9, 10, 16] and more recent machine learning (ML) models trained on engineered interaction-log features [17]. While these approaches demonstrate that usability smells can be detected from logs, they primarily focus on *what* to detect and on how accurately detectors perform. Less attention has been paid to *how* the choice of representation influences detection performance. Here, *representation* refers to how raw interaction logs are transformed into engineered features that capture different aspects of user interaction. This matters because representation choices can determine whether a usability smell is visible at all to an automated detector.

This paper investigates the detection of usability smells from a representation-centric perspective. We compare three types of evidence available in interaction logs: (1) behavioral information about user actions; (2) structural information about interacted interface elements; (3) contextual information about tasks. Instead of investigating which model has the best detection performance, we ask how different representations of interaction logs affect the detectability of task-level usability smells, and how learned detectors compare with rule-based detectors.

We evaluate gradient boosting (GB) models, an ensemble of decision trees, trained on individual and combined representations, and compare their performance with rule-based detectors derived from prior work [10, 11, 16]. For this, we use the only available dataset with expert-annotated usability smells, AMUSED [17]. Our results show that no single representation is universally optimal. Instead, the most effective representation varies across smells and application contexts: some smells are captured by behavioral signals, others require structural or contextual information, and some benefit from combining multiple representations.

We make the following contributions: (1) an empirical comparison of behavioral, structural and contextual representations for task-level usability smell detection; (2) evidence that usability smells detection performance is representation-dependent and application-sensitive: the most effective representation varies across smells and applications, and adding more features does not necessarily improve detection; and (3) practical guidance for automated usability smell detection, clarifying when learned models are preferable, when rule-based detectors remain sufficient and why both depend on the availability of appropriate interaction log evidence.

MuC '26, Duisburg, Germany

ACM ISBN 979-8-4007-2611-8/26/08
https://doi.org/10.1145/3820253.3831380

2 Usability Smells from Interaction Logs

Early research has shown that interaction logs can support scalable usability evaluation by identifying recurring behavioral patterns associated with usability problems [18]. Building on this idea and code smells [7], Garrido et al. [8] introduced usability smells to formalize such patterns as indicators of potential usability issues. Subsequent work established smell catalogs, linked them to usability refactorings, and proposed rule-based detectors over web and mobile interaction logs [8–10, 15, 16].

Rule-based detectors are interpretable and useful when usability smells correspond to stable patterns such as repeated actions or excessive navigation [9]. However, they depend on predefined assumptions about how smells manifest and on whether the required telemetry such as interaction events, dwell times, or scroll depth, is actively captured. More recent work by Santos et al. [17] leverages ML-based approaches that combine behavioral, structural, and contextual features to detect usability smells from interaction data. Although they demonstrate promising performance, it remains unclear how interaction-log representations affect the detection of different usability smells.

This gap motivates our representation-centric analysis. We investigate how different representations of interaction data influence the detectability of usability smells using learned models.

3 Methodology

3.1 Dataset and Data Representation

Our analysis is based on the AMUSED dataset [17], which contains interaction logs from 70 users performing 44 tasks across three social web applications (SN1-3), together with expert annotations for seven task-level and four action-level usability smells. The descriptions of all smells are provided in Table 3. AMUSED is an appropriate dataset for this study because it provides three elements required for a representation-centric analysis: event-level interaction logs, task and application context, and usability-smell annotations. The dataset captures five types of user interactions (clicks, keypresses, scrolls, input changes, and URL changes) with event-specific metadata. Interaction logs are segmented into episodes, where each episode represents a sequence of user interaction events on a specific webpage. Because [17] reported low inter-rater agreement for action-level smells, we focus on task-level smells that reflect issues that emerge across sequences of interactions. Following established perspectives in process mining and HCI, we decompose interaction logs into three complementary representations. Process mining models event logs as activities executed within task instances and can be enriched with contextual attributes [20]. In parallel, the Object-Action Interface model conceptualizes interaction as users performing actions on interface objects [19]. Combining these perspectives, we therefore distinguish between behavioral, structural, and contextual representations:

- **Behavioral Representation (B)** captures what users do, independent of specific interface elements. It includes event counts and proportions, inter-event timing, event rates, and transition patterns.
- **Structural Representation (S)** captures properties of the interface elements involved in interaction. It includes DOM object information, XPath-based hierarchy, and element diversity.
- **Contextual Representation (C)** captures task-level context. It includes task duration, task completion status, and number of episodes.

This decomposition separates three practically distinct sources of evidence available for automated usability smell detection. A complete list of all engineered features for each representation is provided in Table 7.

3.2 Experimental Design

Santos et al. [17] showed that gradient boosting models consistently outperform other ML classifiers across all evaluated metrics for multilabel smell classification. Motivated by these findings, we trained GB models on individual and combined representations to evaluate their contribution to smell detection. As model selection or optimization was not the focus of this work, we did not investigate alternative ML approaches. We trained one GB model per smell instead of a multilabel classifier to isolate the effects of data representations on each smell. For GB evaluation, we used 5×5 repeated stratified cross-validation. We computed precision, recall, and F1-score on each held-out test fold. The reported performance corresponds to the mean across 25 folds. Stratified splitting was used to preserve class proportions across folds, while class-balanced sample weighting was applied during model training to reduce the influence of class imbalance. However, performance estimates may still be influenced by the particular cross-validation partitions and the characteristics of the dataset [6].

For the ablation study, we excluded SN3 with only 8 users, and compared results from SN1 (570 task instances, 32 users) with SN2 (356 task instances, 30 users). To compare GB models with rule-based detectors, we evaluated only on the SN1 dataset to avoid application-specific effects. We also performed a feature subsampling experiment to control for the substantially higher dimensionality of the structural representation. As subsampled and full structural representations achieved comparable performance, subsequent experiments used the full representation to retain all available information.

To contrast learned and rule-based detection, we implemented rule-based task-level smell detectors based on prior work where the required signals are available in the dataset: *Cyclic Task* [16], *Laborious Task* [16], *Missing Task Feedback* [16], *Repetition in Text Fields* [10], *Too Many Layers* [16]. Some rules could not be implemented reliably because the required telemetry, such as screen resolution normalization for *High Interaction Distance* [11] and clear form-submission triggers for *Late Validation* [16] was unavailable. Instead of treating this only as a limitation, we used it to illustrate a practical constraint of rule-based usability detection: rules often depend on specific instrumentation assumptions. Table 6 summarizes the adaptations made to the rules to account for differences between the original assumptions and the available interaction data. We assessed agreement between rule-based and ML-based methods by reporting the raw agreement rate and Cohen's kappa coefficient [2] to account for chance agreement [14].

Table 1: Comparison of rule-based (*Rule*) and gradient boosting (*GB*) approaches for usability smell detection in SN1. Only smells for which rule-based detection can be reliably implemented are included. For each smell, we report the total number of positive instances (n^+), and the mean values obtained from 5×5 repeated stratified cross-validation for F1-score (F1), precision (P), recall (R), agreement rate (Agr. (%)), and Cohen's kappa (κ).

Smell	n^+	$F1_{Rule}$	$F1_{GB}$	P_{Rule}	P_{GB}	R_{Rule}	R_{GB}	Agr. (%)	κ
Cyclic Task	80	.236	.603	.307	.650	.200	.578	84.4	.188
Laborious Task	192	.406	.748	.904	.764	.270	.725	75.2	.303
Missing Task Feedback	233	.330	.615	.655	.632	.223	.603	68.2	.245
Repetition in Text Fields	44	.447	.731	.460	.892	.451	.642	92.5	.394
Too Many Layers	84	.433	.424	.404	.507	.475	.373	83.3	.325

4 Results

4.1 Representation Ablation: Per Smell and Per Application Classification

As shown in Table 2, no single feature representation consistently achieved the best performance across all smells and applications. Five of the six smells exhibited partial overlap in their best-performing representations between SN1 and SN2, indicating that certain representations remained informative across applications. In particular, structural features appeared in the best-performing combinations for *High Interaction Distance*, *Late Validation* and *Missing Task Feedback* in both applications. However, the optimal feature combinations still varied across applications. For example, *Laborious Task* was best detected using B+S+C in SN1 but only C in SN2, while *Cyclic Task* favored B+S in SN1 and B+C in SN2. *Too Many Layers* showed no overlap, suggesting that its most informative representation is highly application dependent. For *Repetition in Text Fields*, no comparison could be performed due to insufficient test data for SN2. The complete ablation results can be found in Table 4 and 5.

4.2 Rule-based vs ML comparison

Across the evaluated smells, the GB model consistently outperforms the rule-based approach in terms of F1-score except for the smell *Too Many Layers*, as shown in Table 1. As for precision and recall, the rule-based method achieves considerably higher precision for *Laborious Task* and higher recall for *Too Many Layers* than the GB model. Agreement rates between the two approaches range from 68.2% to 92.5%, with corresponding Cohen's kappa values between .188 and .394, showing slight to fair agreement [13].

5 Discussion

Representation Dependence of Usability Smell Detection. In response to our research question on how different representations of interaction logs affect the detectability of task-level usability smells, the ablation results show that usability smell detectability is representation-dependent and application-sensitive: no single feature representation consistently performs best across smells or applications. This finding is consistent with prior work that treats usability smells as indicators inferred from different kinds of observable evidence in user interaction data, including task structures, user actions, interface elements and interaction event patterns [1, 9, 10, 12, 15, 16]. While some smells benefit from combining multiple representations, others are best detected using individual representations or smaller feature combinations. This variability suggests that feature representation should be treated as a primary modeling decision rather than a fixed preprocessing step. Because the best-performing representations varied across applications, a representation that performs well in one application may not transfer directly to another.

Rule-based vs Learned Detection. To answer our question of how learned detectors compare with rule-based detectors, the results show that learned models generally provide a more balanced precision and recall trade-off, while rule-based detectors remain useful for smells tied to explicit behavioral or structural patterns when the required telemetry is available. Across all evaluated smells except *Too Many Layers*, GB achieves substantially higher F1-scores, primarily because it balances precision and recall more effectively. In contrast, rule-based detectors can achieve relatively high precision but low recall, indicating that they accurately identify only a subset of smell instances. Agreement analysis further highlights this difference: although agreement rates range from 68% to 93%, Cohen's κ remains low across all smells, indicating limited agreement beyond chance. This suggests that the two approaches often identify different positive instances despite agreeing on many negative cases, which is consistent with prior work showing that automated smell-based approaches do not necessarily reproduce the same findings as traditional usability evaluation methods [9]. The comparatively

Table 2: Best-performing feature representation for each smell and application. Representations correspond to the highest F1-score in the within-application (SN1, SN2) ablation study. The Overlap row indicates agreement across applications (Partial = at least one shared feature type, No = no overlap).

Smell	SN1	SN2	Overlap
Cyclic Task	B+S	B+C	Partial
High Interaction Distance	S	B+S+C	Partial
Laborious Task	B+S+C	C	Partial
Late Validation	B+S+C	B+S	Partial
Missing Task Feedback	S	S+C	Partial
Repetition in Text Fields	B+S	-	-
Too Many Layers	S	C	No

better performance of the rule-based detector for *Too Many Layers* suggests that smells tied to explicit structural properties can be captured effectively through deterministic rules. However, the performance gap for *Cyclic Task*, *Laborious Task*, *Missing Task Feedback*, and *Repetition in Text Fields* suggests that many smells are characterized by combinations of interaction signals that are difficult to encode as fixed thresholds or patterns. Earlier rule-based approaches typically define smells through specific event patterns, task-tree properties, or threshold-based criteria [9, 10, 16]. Our results show that such definitions are useful, but may not capture the full variability with which smells appear across applications and users. Several rule-based methods from prior work [11, 16] could not be implemented reliably because the required telemetry was unavailable. In contrast, learned models can exploit broader feature representations when labeled data are available, making them less dependent on hand-crafted thresholds and exact instrumentation assumptions.

Overall, these findings refine how usability smells should be understood in log-based evaluation. Usability smells are not uniformly observable from interaction traces, and their detectability depends on whether the log representation captures the relevant behavioral, structural, or contextual evidence. This helps explain why fixed rule sets can work well for some smells but miss others, and why adding more features does not automatically improve detection. For automated usability evaluation tools, the key design question is therefore not only which detector to use, but which representation makes the target smell observable.

Implications for Practitioners. Figure 1 summarizes a decision flow for applying usability smell detection. The first step is to identify the smells of interest and verify that the available telemetry captures the behavioral, structural, or contextual evidence needed to make the smells observable. When labeled usability smells are available, practitioners should use learned models, since they provide more balanced precision and recall across most smells. However, practitioners should evaluate multiple feature representations and validate the selected model in the target application, rather than assuming a single representation or representation combinations will generalize across smells or applications. When labeled data are unavailable, rule-based detectors can be used if the target smell has a clear rule and the telemetry supports it. They are more interpretable than learned models, but less suitable for smells expressed through complex signal combinations or when the required instrumentation assumptions are not met.

Limitations and Future Work. This study is based on a single dataset and two applications from the same domain. While the observed differences between applications suggest that application context influences smell detection, additional datasets are needed to assess the generalizability of these findings. Furthermore, only smells for which sufficient positive instances were available could be analyzed, and some rule-based detectors could not be implemented due to missing telemetry. Moreover, the subjective nature of usability smell annotations introduces uncertainty in the ground truth, which may limit achievable detection performance regardless of the detection approach.

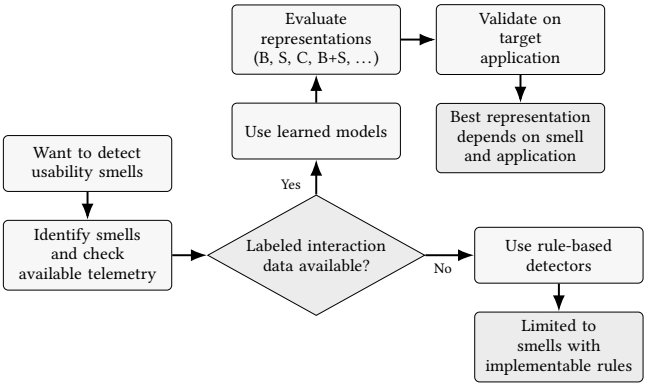

Figure 1: Decision flow for applying usability smell detection. Practitioners should first identify the target smells and verify that the available telemetry can make them observable. When data labeled with usability smells are available, practitioners should use learned detectors, compare alternative feature representations, and validate the chosen approach in the target application context. Without labeled data, detection is limited to smells that can be implemented as rules.

6 Conclusion

This paper examined usability smell detection from a representation-centric perspective. Using interaction logs from the AMUSED dataset, we compared behavioral, structural, and contextual representations for task-level smell detection, and compared learned ML models with rule-based detectors. Our results show that the most effective representation varies across smells and applications, indicating that usability smell detection is representation-dependent and application-sensitive. While some feature types remain consistently informative, no single representation is universally optimal. The comparison with rule-based detectors further shows that ML models generally achieve better detection performance and identify smell instances that are not captured by predefined rules. At the same time, the inability to implement some rule-based detectors highlights the practical dependence of rule-based approaches on specific instrumentation assumptions. Overall, these findings suggest that representation design should be a central design consideration in automated usability smell detection, and data-driven approaches such as learned models provide a more flexible foundation for detecting diverse usability issues across applications.

LLM Usage and Open Science

LLM is used to improve the phrasing of sentences within the paper. We encourage readers to review, reproduce, and extend our findings. To achieve this goal, our analysis script is available via OSF (https://osf.io/myt9e/).

Author Contributions

Michelle Xiao-Lin Foo: Conceptualization, Formal Analysis, Investigation, Software, Validation, Visualization, Writing – original draft, Writing – review & editing; **Sven Mayer**: Funding acquisition, Supervision, Writing – review & editing

References

[1] Diogo Almeida, José Creissac Campos, João Saraiva, and João Carlos Silva. 2015. Towards a catalog of usability smells. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing* (Salamanca, Spain) *(SAC '15)*. Association for Computing Machinery, New York, NY, USA, 175–181. doi:10.1145/2695664.2695670

[2] Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20, 1 (1960), 37–46.

[3] Flávia de Souza Santos, Marcos Vinícius Treviso, Sandra Pereira Gama, and Renata Pontin de Mattos Fortes. 2022. *A Framework to Semi-automated Usability Evaluations Processing Considering Users' Emotional Aspects.* Springer International Publishing, 419–438. doi:10.1007/978-3-031-05311-5_29

[4] Asbjørn Følstad, Effie Law, and Kasper Hornbæk. 2012. Analysis in practical usability evaluation: a survey study. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 2127–2136. doi:10.1145/2207676.2208365

[5] Asbjørn Følstad, Effie Lai-Chong Law, and Kasper Hornbæk. 2010. Analysis in usability evaluations: an exploratory study. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*. Association for Computing Machinery, New York, NY, USA, 647–650. doi:10.1145/1868914.1868995

[6] George Forman and Martin Scholz. 2010. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. *SIGKDD Explor. Newsl.* 12, 1 (Nov. 2010), 49–57. doi:10.1145/1882471.1882479

[7] Martin Fowler. 1999. *Refactoring: improving the design of existing code.* Addison-Wesley.

[8] Alejandra Garrido, Gustavo Rossi, and Damiano Distante. 2011. Refactoring for Usability in Web Applications. *IEEE Software* 28, 3 (2011), 60–67. doi:10.1109/MS.2010.114

[9] Julián Grigera, Alejandra Garrido, José Matías Rivero, and Gustavo Rossi. 2017. Automatic detection of usability smells in web applications. *International Journal of Human-Computer Studies* 97 (2017), 129–148.

[10] Patrick Harms. 2016. *Automated Field Usability Evaluation Using Generated Task Trees.* Ph. D. Dissertation. University Goettingen Repository. doi:10.53846/goediss-5453

[11] Patrick Harms. 2019. Automated Usability Evaluation of Virtual Reality Applications. *ACM Transactions on Computer-Human Interaction* 26, 3 (April 2019), 1–36. doi:10.1145/3301423

[12] Patrick Harms and Jens Grabowski. 2014. Usage-based automatic detection of usability smells. In *Human-Centered Software Engineering: 5th IFIP WG 13.2 International Conference, HCSE 2014, Paderborn, Germany, September 16-18, 2014. Proceedings 5.* Springer, 217–234.

[13] J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics* (1977), 159–174.

[14] Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica* 22, 3 (2012), 276–282.

[15] Fabio Paternò, Antonio Giovanni Schiavone, and Antonio Conti. 2017. Customizable automatic detection of bad usability smells in mobile accessed web applications. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '17)*. ACM, 1–11. doi:10.1145/3098279.3098558

[16] Rafael Fontinele Ribeiro, Matheus de Meneses Campanhã Souza, Pedro Almir Martins de Oliveira, and Pedro de Alcântara dos Santos Neto. 2019. Usability problems discovery based on the automatic detection of usability smells. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (Limassol, Cyprus) *(SAC '19)*. Association for Computing Machinery, New York, NY, USA, 2328–2335. doi:10.1145/3297280.3299747

[17] Flávia de S. Santos, Marcos Treviso, Kamila R. H. Rodrigues, Renata P. M. Fortes, and Sandra P. Gama. 2026. AMUSED: A Multi-Modal Dataset for Usability Smell Identification. *IEEE Transactions on Affective Computing* 17, 1 (Jan. 2026), 726–738. doi:10.1109/taffc.2025.3632675

[18] I. Shah. 2008. Event Patterns as Indicators of Usability Problems. *Journal of King Saud University - Computer and Information Sciences* 20 (2008), 31–43. doi:10.1016/s1319-1578(08)80003-1

[19] Ben Shneiderman. 2010. *Designing the user interface: strategies for effective human-computer interaction.* Pearson Education India.

[20] WMP van der Aalst. 2016. Process mining: data science in action.

A Appendix

Table 3: Description of (7) task-level and (4) action-level usability smells in the AMUSED dataset.

Usability Smell	Description
Task-level Smells	
Cyclic Task [16]	Occurs when the task requires performing many repetitive actions.
High Interaction Distance [11, 15]	Occurs when related content is arranged too far apart.
Laborious Task [16]	Occurs when the task execution demands the execution of a large number of actions and time.
Late Validation [9]	Occurs when input errors are only checked after task submission.
Missing Task Feedback [3]	Occurs when a task is repeated for the same inputs.
Repetition in Text Fields [10]	Occurs when the same text is entered multiple times in a task.
Too Many Layers [9, 16]	Occurs when users need to scroll through many pages to perform a task.
Action-level Smells	
Misleading Action [9]	Occurs when the user clicks on an element that switches pages but quickly returns to the previous page.
Missing Action Feedback [10, 16]	Occurs when an action is constantly repeated.
Undescriptive Element [9, 16]	Occurs when too many users try to hint at a particular element on the page.
Unnecessary Action [11]	Occurs when a page contains only one action.

Table 4: Representation ablation results for SN1: mean F1-scores from 5 × 5 repeated stratified cross-validation using smell-specific one-vs-rest gradient boosting classifiers. Feature counts are shown in parentheses. For each smell, the best score is highlighted in bold and the second-best score is underlined.

Smell	**B** (51)	**S** (178)	**C** (6)	**B+S** (229)	**B+C** (57)	**S+C** (184)	**B+S+C** (235)
Cyclic Task	.566	.561	.533	**.608**	.568	.579	.602
High Interaction Distance	.153	**.169**	.120	.141	.125	.131	.143
Laborious Task	.702	.715	.672	.721	.716	.728	**.733**
Late Validation	.590	.536	.474	.602	.601	.566	**.603**
Missing Task Feedback	.579	**.616**	.513	.609	.598	.606	.611
Repetition in Text Fields	.690	.667	.576	**.719**	.674	.656	.700
Too Many Layers	.360	**.488**	.334	.417	.363	.472	.424

Table 5: Representation ablation results for SN2: mean F1-scores from 5 × 5 repeated stratified cross-validation using smell-specific one-vs-rest gradient boosting classifiers. Feature counts are shown in parentheses. For each smell, the best score is highlighted in bold and the second-best score is underlined.

Smell	**B** (51)	**S** (90)	**C** (6)	**B+S** (141)	**B+C** (57)	**S+C** (96)	**B+S+C** (147)
Cyclic Task	.618	.331	.374	.574	**.633**	.408	.549
High Interaction Distance	.528	.499	.450	.584	.522	.541	**.589**
Laborious Task	.770	.714	**.783**	.768	.763	.756	.760
Late Validation	.515	.552	.427	**.582**	.554	.574	.551
Missing Task Feedback	.394	.437	.324	.456	.385	**.469**	.437
Too Many Layers	.139	.184	**.271**	.112	.133	.178	.133

Table 7: Overview of engineered features by representation type.

Feature Group	Features
Behavioral (B)	Event type count, event proportions, total events, inter-event time (mean, std, median, max), event rate, transition bigrams (consecutive event pairs), bigram entropy, repetition rate, back-and-forth rate, scroll time (mean), scroll count (mean), input change value length (mean, std), click position (x, y), click statistics (mean, std), click range
Structural (S)	DOM object count, DOM object proportions, DOM object entropy, unique DOM elements, unique XPath (the location of an element within the DOM hierarchy), XPath diversity, XPath depth (mean, std, max), input events target elements, per task top-3 clicked elements counts and proportions
Contextual (C)	Number of episodes, events per episode (mean, std, max), task duration, task completion status

Table 6: Mapping of rule-based detectors to prior work and implemented adaptations.

Smell	Adaptation
Cyclic Task [16]	Use only click, input change and tabchange events
Laborious Task [16]	Use Z-score to detect outliers in action count and task duration instead of IQR
Missing Task Feedback [16]	Group event sequences into episodes and compare within tasks
Repetition in Text Fields [10]	Approximate repetition using input length due to lack of raw input text
Too Many Layers [16]	Use threshold of 4 unique pages (within recommended range 3–5)
High Interaction Distance [11]	Not implemented (requires screen resolution normalization)
Late Validation [16]	Not implemented (missing form submission trigger)