Detecting Usability Issues in Recommender Systems with Multimodal Large Language Models

Sebastian Lubos[1,*], Alexander Felfernig[1], Viet-Man Le[1] and Thi Ngoc Trang Tran[1]

[1]Graz University of Technology, Graz, Austria

Abstract

User-oriented dimensions such as user control and explainability are critical for *recommender systems (RecSys)* but are rarely assessed in early development phases due to the cost of manual *usability* evaluation. To support this process, we present an automated framework that uses *multimodal large language models (MLLMs)* to perform criterion-based analyses of recommender *user interfaces (UIs)* from screen recordings that showcase their functionalities. The framework applies RecSys-specific usability criteria and produces structured outputs that include issue descriptions, severity ratings, and explicit recommendations to resolve them. We instantiate the framework in a prototype tool that supports repeatable evaluations. The feasibility and robustness of the framework are evaluated on three prototype recommenders using two state-of-the-art MLLMs over multiple runs. The results indicate that the MLLM-based analyses often identify plausible RecSys-relevant usability issues and generate actionable improvement suggestions, which can provide support through low-effort reviews.

Keywords

Recommender Systems Usability, Usability Evaluation, Multimodal Large Language Models

1. Introduction

Many modern online applications include *recommender systems (RecSys)* as a core component to help users navigate large catalogs with personalized item suggestions [1]. While algorithmic accuracy is a key evaluation criterion [2], the success of a RecSys in practice also depends on its *usability*. The *usability* of a system determines how effectively, efficiently, and satisfactorily users can achieve their goals by interacting with the *user interface (UI)* [3]. Good usability of recommenders enables users to express preferences, review recommendations, and make informed decisions [4, 5]. If UIs are not intuitive for users, even highly accurate recommendations may fail to provide value.

In practice, usability is assessed through *usability testing* with real users [6] and *expert inspections* using guidelines [7] such as *Nielsen's heuristics* [8] or recommender-specific criteria [9]. Both approaches require substantial manual effort and expertise, which limits their regular use in many companies [10].

Various approaches to automating this task have been proposed, including rule-based tools and heuristic checkers [11, 12]. However, these cover only selected usability dimensions and struggle with context-specific or subjective issues [13]. A newer direction uses *multimodal large language models (LLMs)* [14] by providing visual representations of UIs as evaluation context. Different studies report the potential of this approach to identify usability issues in design mockups [15], mobile interfaces [16], product configurators [17], and web applications [18, 19]. For RecSys UIs, an initial study applied recommender-specific heuristics for MLLM-based evaluations to obtain helpful insights [20].

Our work builds on these findings by presenting a framework and prototype tool for conducting criterion-based usability inspections of RecSys UIs. Practitioners, such as designers and developers, can upload screen recordings that showcase the UI's features and functionality to obtain structured usability feedback for domain-specific criteria. The feedback includes issue descriptions, severity ratings, and

IntRS'26: Joint Workshop on Interfaces and Human Decision Making for Recommender Systems, September 28, 2026, Minneapolis, Minnesota, USA

*Corresponding author.

sebastian.lubos@tugraz.at (S. Lubos)

0000-0002-5024-3786 (S. Lubos); 0000-0003-0108-3146 (A. Felfernig); 0000-0001-5778-975X (V. Le); 0000-0002-3550-8352 (T. N. T. Tran)

improvement recommendations. While prior work relied on static screenshots [20], we include the possibility to use screen recordings, which capture dynamic elements and transitions.

To assess the framework's feasibility and robustness, we developed three RecSys prototypes across different domains and recorded screen sessions showcasing the systems' main features and functionalities. These recordings were used as input for repeated evaluations with two state-of-the-art MLLMs to examine stability across runs. Our findings indicate that the MLLM-based analyses often surface plausible RecSys-relevant issues and provide actionable suggestions in reviews with comparably low manual effort. This way, the approach can be seen as a complement to expert evaluations.

Overall, this paper makes the following contributions:

- We present a framework that applies RecSys-specific, criterion-based checks to screen recordings and returns structured reports that include descriptions of usability issues, severity ratings, and explicit recommendations for improvement.
- We evaluated the feasibility and robustness of the MLLM-based usability evaluation by applying it to three RecSys prototypes using two state-of-the-art MLLMs across repeated runs. We analyze the stability of usability issue descriptions, improvement recommendations, and severity ratings.

The remainder of this paper is organized as follows: Section 2 summarizes usability heuristics for RecSys UIs. Section 3 presents our MLLM-based usability analysis framework. Section 4 describes the study setup for feasibility and robustness. Section 5 reports results and discusses implications. Section 6 outlines limitations. Finally, the paper is concluded in Section 7.

2. Usability Heuristics for Recommender UIs

User satisfaction with recommender systems depends on multiple aspects of interaction. However, evaluations often focus on algorithmic accuracy and personalization strategies [2, 21], while the user-oriented aspect of usability and user experience is often neglected. One approach to assess these dimensions is *usability analysis*, which examines how effectively users navigate, interpret, and interact with different parts of the system [4, 5]. This includes mechanisms for expressing preferences, reviewing and understanding recommendations, and providing feedback.

Structured usability analyses often rely on explicit criteria, for example, *Nielsen's heuristics* [8], which describe general usability principles. While these also apply to RecSys, they overlook recommender-specific interactions, such as *preference elicitation* and *recommendation presentation*. A user-oriented evaluation framework for RecSys was presented in [9]. It covers algorithm-related aspects (e.g., perceived accuracy and novelty) and UI-related dimensions (e.g., interface adequacy and ease of use, and transparency), which provide a broad basis for collecting feedback from test users. However, some parts of the evaluation frameworks rely on subjective factors (e.g., intention to reuse), which reduces the direct applicability to structured expert or MLLM-based evaluations.

Similar to [20], we adapt the framework metrics [9] to derive explicit usability criteria for reviewing general interface heuristics aimed at recommender-specific interactions (preference elicitation and recommendation presentation). In this sense, we omitted subjective aspects (e.g., intention to reuse) and kept criteria that can be assessed by observing the UI behavior in screen recordings. The resulting set of criteria is summarized in Table 1. These criteria form the basis of our usability evaluation framework, which aims to assess how well a recommender interface supports users in understanding, reviewing, and providing feedback on recommendations.

3. MLLM-based Usability Analysis

We present a lightweight, criterion-based framework for analyzing the usability of recommender UIs using MLLMs. The framework expects standardized visual input, specifically a screen recording that showcases the recommender UI, as context for the analysis. Based on this, the framework checks

Table 1
Usability criteria focusing on RecSys-related aspects of user interactions. These criteria serve as a basis for the MLLM-based analysis of recommender UIs.

Category	ID	Usability Criterion
General	G1	Is the layout clear and visually structured?
	G2	Are interactive elements (e.g., buttons, icons) clearly recognizable?
	G3	Is the amount of information per item appropriate and helpful?
	G4	Are interface elements used consistently (e.g., icons, labels, colors)?
Preference Elicitation	P1	Can users explicitly express preferences (e.g., ratings, likes, categories)?
	P2	Is there transparency about how input affects recommendations?
	P3	Do users have control and flexibility (e.g., skip, edit, undo inputs)?
Recommendation Presentation	R1	Are recommendations clearly labeled as such?
	R2	Are different types of recommendations distinguishable (e.g., "Because you liked...")?
	R3	Are there explanations for why items are recommended?
	R4	Can users interact with recommendations (e.g., rate, hide, save)?

usability criteria to identify issues within the recommender. These are summarized as a machine-readable output. Overall, the framework involves three main steps:

1. *Input Preprocessing*: A single UI screen recording is used as context for the analysis. If the used MLLM can natively accept video contexts, the screen recording is submitted directly. For MLLMs that do not support this, *FFmpeg*[1] is used to extract an ordered sequence of frames, which is then provided as input context.
2. *Criterion-Based Analysis*: The MLLM is instructed to sequentially evaluate the usability criteria specified in Table 1 based on the screen recording, one criterion per prompt. A system prompt (see Listing 1) specifies the general expected behavior, including the role and boundaries. A criterion-specific user prompt template (see Listing 2) is used to inject the target criterion. The MLLM generates a JSON response with a fixed schema that includes a list of issues. Each issue contains an `issue description`, `improvement recommendation`, and a `severity rating`.[2] The severity indicates the importance of resolving issues based on their impact on the user experience. If no issue is found for a criterion, the output is an empty list.
3. *Result Persistence and Review*: The generated usability feedback is persisted. Users review the MLLM-identified usability issues manually by reading the issue descriptions to support *human-in-the-loop* validation.

```
You are a usability expert for recommender system user interfaces.
Your role is to assess the provided screen recordings of interfaces for one usability criterion at a time.
Report only issues that are clearly observable in the recording.
Return concise issue descriptions and actionable recommendations.
Avoid assumptions that are not visually grounded.
```

Listing 1: System prompt used to define the evaluator role and boundaries of the usability analysis.

We implemented the framework in a prototype application to allow users to initiate the MLLM-based usability analysis and review the identified usability issues.[3] Users can start the analysis by uploading a screen recording that shows user interactions with the application to be analyzed. The provided interaction should show the main features and functionalities to provide visual context for MLLM-based analysis. The uploaded video is automatically processed and analyzed. The prototype application presents the identified usability issues in a structured UI after completing the analysis. As shown in the screenshot in Figure 1, the result view presents a list of identified usability issues for the uploaded sample, ordered by severity in descending order. For each issue, the description is shown, and the user

[1] https://ffmpeg.org
[2] Based on *Nielsen*'s severity scale, to distinguish *usability catastrophes*, *major problems*, *minor problems*, and *cosmetic problems*. See https://www.nngroup.com/articles/how-to-rate-the-severity-of-usability-problems/.
[3] Please contact the authors to access the prototype.

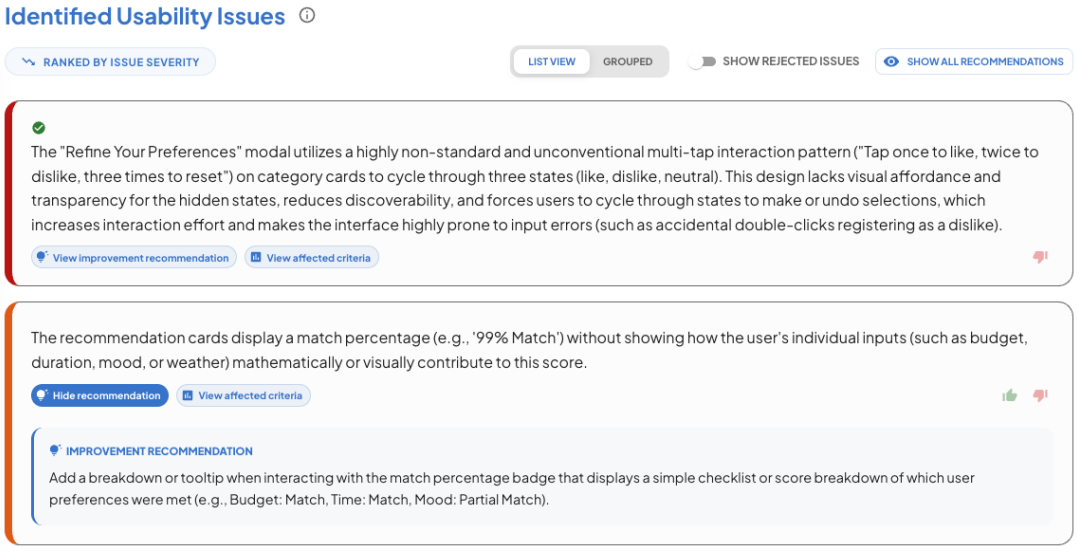

Figure 1: Screenshot crop of the prototype application showing the list of identified usability issues for the *PocketQuest* example (see Table 2) ordered by descending severity. While issue descriptions are always shown, improvement recommendations are shown only for expanded issues to reduce UI complexity. Users can validate issues and store their decisions using the thumbs-up and thumbs-down buttons.

can expand it to open the suggested improvements. Human validation is supported through thumbs-up and thumbs-down buttons to accept or decline issues.

```
Usability Criterion: {criterion}
Evaluation Instructions:
Evaluate the interface for the criterion listed above.
Follow these steps:
1. Identify any usability issues related to the criterion and explain each issue.
2. Provide a practical, actionable recommendation to fix each issue.
3. Assign severity using this exact scale:
    - 1: Cosmetic problem only
    - 2: Minor usability problem
    - 3: Major usability problem
    - 4: Usability catastrophe
Response Guideline:
Keep each issue's analysis and recommendation concise (1-3 sentences).
Avoid restating the criterion definition.
[FORMATTING INSTRUCTIONS]
```

Listing 2: Prompt template used to instruct an MLLM for the criterion-based usability analysis, which considers one usability criterion at a time. The usability criteria (see Table 1) are inserted for the {`criterion`} placeholder. Formatting instructions are omitted.

4. Study Design

In this preliminary study, we aim to validate the feasibility of our MLLM-based usability analysis and assess its robustness across repeated runs with different MLLMs. To reflect one intended use case of the framework, i.e., an early, user-oriented assessment of RecSys UIs, we built three interactive recommender prototypes in different domains using *Figma Make*.[4] These prototypes resemble modern RecSys applications that display sample data, while the underlying algorithms are not yet implemented. We did not actively include usability issues within the prototypes. However, due to their early-stage development phase, they are likely to have several issues.

All prototypes were deployed locally to record interactions showcasing their main features and functionality. These visualize how the RecSys is presented to the end user. While these recordings do not

[4] https://www.figma.com/make/

reflect actual user interactions, which may indicate specific issues within the UI, these recordings contain sufficient visual information for early-stage usability reviews. Overall, we consider three applications (see Table 2): (*i*) *PocketQuest*, which recommends activities in the user's current surroundings, (*ii*) *Eco-Bite*, a sustainability-aware grocery shopping assistant, and (*iii*) *DeepWork Radio*, a personalized music station tailored to current cognitive performance.

Table 2
Recommender prototypes used for the usability analysis study. The table indicates the item domain (i.e., which content is recommended), the prototype name, the duration of the recorded interaction (in minutes), and the link to the prototype repository.

ID	Domain	Name	Duration	Repository
1	Activities	*PocketQuest*	02:33	https://github.com/AIG-ist-tugraz/Pocketquest
2	Food	*Eco-Bite*	04:19	https://github.com/AIG-ist-tugraz/Ecobite
3	Music	*DeepWork Radio*	02:09	https://github.com/AIG-ist-tugraz/Deepworkradio

To assess the robustness of usability analyses across repeated runs, we consider two state-of-the-art MLLMs capable of processing video input (see Table 3). Other candidates were excluded due to limited applicability to our setting (e.g., *claude-haiku-4-5* by *Anthropic*[5] and *mistral-small-4* by *Mistral*[6] could not process all frames of the sample recordings in a single request).

Table 3
MLLMs used for the experiments.

ID	Name	Version	Reference
1	*gemini-2.5-flash*	*stable*	https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash
2	*gpt-5.4-mini*	*2026-03-17*	https://developers.openai.com/api/docs/models/gpt-5.4-mini

For more deterministic comparisons, both models use identical prompts (see Section 3) and decoding parameters[7]. While the video can be directly ingested into *gemini-2.5-flash*, *gpt-5.4-mini* requires preprocessing and uses a sequence of frames as input. We standardized the sampling rate to 1 frame per second for both approaches. For each model, we repeated the evaluations 5 times for each application and usability criterion, yielding 55 outputs per application–MLLM pair and 330 outputs in total.[8]

To evaluate the results, we combine a quantitative and qualitative approach. Firstly, we compare the raw counts of identified issues across runs and MLLMs. Secondly, we assess the consistency of issues across runs and determine whether similar issues yield consistent improvement suggestions and severity ratings. Finally, we assess the plausibility of MLLM-identified issues and recommendations through a manual review by the authors of this work.

5. Results and Discussion

Overall, we observed substantial differences in the number of identified usability issues across runs for each MLLM and between MLLMs. *gemini-2.5-flash* produced completely identical outputs in terms of wording and identified issues across all five runs in nearly half of the usability analyses (16 out of 33 total application-criterion pairs). In contrast, *gpt-5.4-mini* never yielded identical outputs across five runs. On average, *gemini-2.5-flash* identified 0.521 issues (standard deviation: 0.677), with a range of 0 to 3 issues per analyzed usability criterion. *gpt-5.4-mini* identified 2.139 issues on average (standard

[5]https://www.anthropic.com/claude/haiku
[6]https://docs.mistral.ai/models/model-cards/mistral-small-4-0-26-03
[7]`temperature = 0.0` and `top_p = 1.0`
[8]The used screen recording, prompts, and results are available in our repository: https://github.com/AIG-ist-tugraz/RecSys-Usability-Issues-IntRS_2026

deviation: 0.833) and a range of 1 to 4 issues per criterion. In 16.4% of individual usability analysis instances, both MLLMs reported the same number of issues.

To compare the consistency of results across runs of each model, we measured how consistently a model reported the same number of issues across repeated runs. For each model, application, and usability criterion, we measured the fraction of run pairs with identical issue counts (1.0 = perfectly stable; 0.0 = completely unstable). The results in Table 4 show high stability for *gemini-2.5-flash* and moderate stability for *gpt-5.4-mini* in counts. For 15 out of 33 criteria (11 criteria across 3 applications), *gemini-2.5-flash* did not identify any issue in all runs. If only cases with at least one issue are considered, the mean consistency rises to 92.9%.

Table 4
Consistency of identified issue counts per usability criterion across five independent runs for each MLLM.

Name	*Mean Consistency (Count)*	*Median Consistency (Count)*
gemini-2.5-flash	0.830	1.0
gpt-5.4-mini	0.645	0.6

As issue-count stability is a coarse indicator and does not imply that the same issues were identified, we assessed whether models identified the same issues across runs using *semantic embeddings*[9] and *agglomerative clustering* [22] with a 0.7 similarity threshold. This clustering approach was applied within the identified issues for each prototype application and usability criterion to group semantically similar issue descriptions into distinct clusters that cover the same usability problems. This bottom-up approach starts with each issue description as its own cluster and merges them based on average cosine similarity to effectively identify recurring problems even when paraphrased or described with varying levels of detail across runs. We measured the *consistency score*, defined as the average frequency of unique issue clusters across runs (1.0 = issue found in every run). The results (see Table 5) show that *gemini-2.5-flash* is more consistent than *gpt-5.4-mini*, although both models are not fully consistent.

Table 5
Semantic consistency score and description similarity across five runs, based on embedding-based clustering (similarity threshold: 0.7).

Name	*Consistency Score*
gemini-2.5-flash	0.72
gpt-5.4-mini	0.45

To evaluate the consistency of issue descriptions and whether identical issues received the same severity across runs and yielded the same improvement recommendations, we analyzed agreement within the identified clusters. The results are reported in Table 6. We found high stability across both models for these metrics, though *gemini-2.5-flash* demonstrated higher internal consistency. The high *description similarity* values within clusters indicate that recurring issues are described with high semantic coherence. Also, the *severity rating agreement* showed that severity ratings were estimated consistently for similar issues. While the improvement *recommendation similarity* also remained highly consistent for *gemini-2.5-flash*, we observed greater variation for *gpt-5.4-mini*, though they still targeted the same underlying issues.

Regarding the similarity of issues between the two MLLMs, we observed low overlap between their identified issues, ranging from 0.0% to 3.4% across the recommender prototypes. A qualitative inspection suggests that the reason for these low values lies in different reporting styles adopted by the MLLMs. While *gemini-2.5-flash* provides specific, single-point issues, *gpt-5.4-mini* groups related observations into broader descriptions. This divergence highlights that while both models may observe the same problems, their grouping differs fundamentally.

[9]Using the *all-MiniLM-L6-v2* model, see https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.

Table 6
Consistency of description similarity and improvement recommendations (based on semantic similarity), and severity ratings agreement (standard deviation of ratings), for similar usability issues across runs for each MLLM.

Model	*Description Similarity*	*Recommendation Similarity*	*Severity Rating Agreement*
gemini-2.5-flash	0.96	0.94	0.07
gpt-5.4-mini	0.81	0.67	0.15

To assess the plausibility of the identified usability issues, we focus on the output from *gemini-2.5-flash*, given its higher stability. Table 7 documents the number of identified issues per application divided by their assigned severity for a single representative run. For each application, between 4 and 10 issues of minor or major issue severity were found, while no usability catastrophes or cosmetic problems were found.

Table 7
Major and minor usability issues identified by *gemini-2.5-flash* for each recommender prototype application in a single representative evaluation run.

Application	*Major Issues*	*Minor Issues*
PocketQuest	1	3
Eco-Bite	2	3
DeepWork Radio	5	5

Across the three prototypes, the identified issues align well with the usability criteria and are clearly localized in the UIs. For general criteria, we see plausible observations, such as high information density that may overwhelm users (*G3*) and inconsistent action-button colors that cause confusion (G4). The descriptions clearly address the issues, e.g., "*The color scheme for primary and secondary action buttons is inconsistent between the main interface and modal dialogs. Primary actions in the main flow are orange (e.g., "Next", "Choose This Adventure"), while in modal dialogs, primary actions are blue (e.g., "Confirm", "Save Changes"). Secondary actions also follow different color patterns, leading to user confusion about action hierarchy.*" However, some issues were incorrectly observed, for instance, missing click affordances were identified for elements that are not clickable (G2).

For preference elicitation criteria, the model correctly flagged missing transparency about how inputs affect recommendations (P2) and proposes a concrete improvement: "*Add a clear label next to the percentage or provide a tooltip for the 'distracted' button to explain its function and the meaning of the changing percentage.*" It also notes missing or complicated explicit feedback mechanisms (P1), including multi-click interactions.

For the recommendation presentation criteria, missing explanations (R3), missing interaction feedback (R4), and absent labels indicating items as recommendations (R1) were correctly identified. A concrete suggestion was to "*Add explicit labels or indicators to individual product and recipe listings that clearly show their alignment with the user's 'Preference Matrix' settings,*" which gives an actionable idea on improving the user experience.

The severity ratings indicated a coherent behavior. Issues affecting core interactions or comprehension are marked as *major*, while confusions and inconsistencies are marked as *minor problems*.

Overall, *gemini-2.5-flash* produced more stable outputs across runs, supporting its potential for integration in practical scenarios. The qualitative review indicates mostly plausible, usability criterion-aligned issues that can guide low-effort, early-phase improvements. However, the low inter-model overlap and partly inconsistent findings across MLLMs suggest that human validation remains necessary, consistent with prior work [16, 19]. Future research should aim to validate whether the MLLM-based issues actually reflect problems that negatively affect user experience in practice. This requires the involvement of actual users and external reviewers, or the use of quantitative indicators, to assess their satisfaction and experience when interacting with a system. Accompanying these user studies with the

development of specific guidelines, including a benchmark for good user-oriented recommender design, could help to further quantify the accuracy of usability evaluation frameworks.

6. Limitations

This work acknowledges limitations in generalizability and coverage. Although we included different MLLMs and application domains, results may vary in other settings. We did not explore prompt variants because our objective was to assess reproducibility within a single setup. We also did not evaluate the completeness of identified issues, i.e., whether all usability problems were recognized. Finally, while the selected recommender-specific usability criteria provide a useful basis, they are not exhaustive and could be extended. We plan to address these aspects in future work.

7. Conclusions

This work introduced a framework that leverages *multimodal large language models (MLLMs)* to automate criterion-based usability evaluations of recommender-system UIs from screen recordings. The approach provides structured outputs, including issue descriptions, severity ratings, and explicit improvement recommendations, aligned with recommender-specific usability criteria. We assessed the framework's feasibility and robustness on three recommender prototypes using two MLLMs. In particular, *gemini-2.5-flash* produced notably stable outputs across runs, with largely plausible, context-aware issue descriptions. In practice, our work could support low-effort, user-oriented evaluations of recommender system UIs in early development phases. Our future work includes validating the usability evaluation approach with actual users of recommender systems to assess whether the framework improves the user experience.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-5, and Grammarly in order to: Grammar and spelling check, Paraphrase and reword, and Improve writing style. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Acknowledgments

The work presented in this paper has been developed within the research project GenRE (Generative AI for Requirements Engineering) funded by the Austrian Research Promotion Agency under the project number 915086.

References

[1] F. Ricci, L. Rokach, B. Shapira, Recommender Systems: Techniques, Applications, and Challenges, Springer US, New York, NY, 2022, pp. 1–35. doi:10.1007/978-1-0716-2197-4_1.
[2] E. Zangerle, C. Bauer, Evaluating recommender systems: Survey and framework, ACM Comput. Surv. 55 (2022). doi:10.1145/3556536.
[3] International Organization for Standardization, ISO/IEC/IEEE International Standard - Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts, ISO/IEC/IEEE 9241-11:2018(E) (2018).
[4] P. Pu, L. Chen, R. Hu, Evaluating recommender systems from the user's perspective: survey of the state of the art, User Modeling and User-Adapted Interaction 22 (2012) 317–355.
[5] B. P. Knijnenburg, M. C. Willemsen, Z. Gantner, H. Soncu, C. Newell, Explaining the user experience of recommender systems, User modeling and user-adapted interaction 22 (2012) 441–504.

[6] C. Hass, A Practical Guide to Usability Testing, Springer International Publishing, Cham, 2019, pp. 107–124. doi:10.1007/978-3-319-96906-0_6.

[7] T. Hollingsed, D. G. Novick, Usability inspection methods after 15 years of research and practice, in: Proceedings of the 25th Annual ACM International Conference on Design of Communication, SIGDOC '07, Association for Computing Machinery, New York, NY, USA, 2007, p. 249–255. doi:10.1145/1297144.1297200.

[8] J. Nielsen, Enhancing the explanatory power of usability heuristics, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '94, Association for Computing Machinery, New York, NY, USA, 1994, p. 152–158. doi:10.1145/191666.191729.

[9] P. Pu, L. Chen, R. Hu, A user-centric evaluation framework for recommender systems, in: Proceedings of the Fifth ACM Conference on Recommender Systems, RecSys '11, Association for Computing Machinery, New York, NY, USA, 2011, p. 157–164. doi:10.1145/2043932.2043962.

[10] S. Lubos, A. Felfernig, V. M. Le, T. N. T. Tran, D. Suppan, R. Willfort, I. Dukic, J. Fuchs, M. Henrich, Enabling automated usability evaluation for configurator user interfaces, in: ConfWS'26: 28th International Workshop on Configuration, Aug 15–16, 2026, Bremen, Germany, 2026.

[11] A. Namoun, A. Alrehaili, A. Tufail, A review of automated website usability evaluation tools: Research issues and challenges, in: M. M. Soares, E. Rosenzweig, A. Marcus (Eds.), Design, User Experience, and Usability: UX Research and Design, Springer International Publishing, Cham, 2021, pp. 292–311.

[12] J. W. Castro, I. Garnica, L. A. Rojas, Automated tools for usability evaluation: A systematic mapping study, in: G. Meiselwitz (Ed.), Social Computing and Social Media: Design, User Experience and Impact, Springer International Publishing, Cham, 2022, pp. 28–46.

[13] E. Kuric, P. Demcak, M. Krajcovic, J. Lang, Systematic literature review of automation and artificial intelligence in usability issue detection, 2025. URL: https://arxiv.org/abs/2504.01415. arXiv:2504.01415.

[14] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, E. Chen, A survey on multimodal large language models, National Science Review 11 (2024). doi:10.1093/nsr/nwae403.

[15] P. Duan, J. Warner, Y. Li, B. Hartmann, Generating automatic feedback on ui mockups with large language models, in: Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems, CHI '24, Association for Computing Machinery, New York, NY, USA, 2024. doi:10.1145/3613904.3642782.

[16] A. E. Pourasad, W. Maalej, Does GenAI Make Usability Testing Obsolete? , in: 2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE), IEEE Computer Society, Los Alamitos, CA, USA, 2025, pp. 675–675. doi:10.1109/ICSE55347.2025.00138.

[17] S. Lubos, A. Felfernig, D. Garber, A. Kraljić, T. Kraljić, V.-M. Le, T. N. T. Tran, G. Leitner, J. Schwazer, D. Suppan, R. Willfort, I. Dukic, J. Fuchs, M. Henrich, Usability analysis of configurator user interfaces with multimodal large language models, 2026. URL: https://arxiv.org/abs/2605.29456. arXiv:2605.29456.

[18] S. Lubos, A. Felfernig, D. Garber, G. Leitner, J. Schwazer, M. Henrich, Investigating multimodal large language models to support usability evaluation, in: H. Fujita, A. Selamat, M. Ghazali, M. Ali (Eds.), Advances and Trends in Artificial Intelligence. Theory and Applications, Springer Nature Singapore, Singapore, 2026, pp. 50–62.

[19] S. Lubos, A. Felfernig, D. Garber, V.-M. Le, M. Henrich, Recommending usability improvements with multimodal large language models, Proc. ACM Softw. Eng. 3 (2026). doi:10.1145/3797121.

[20] S. Lubos, A. Felfernig, D. Garber, V. M. Le, T. N. T. Tran, Towards llm-based usability analysis for recommender user interfaces, in: Proceedings of the 12th Joint Workshop on Interfaces and Human Decision Making for Recommender Systems (IntRS 2025), volume 4027, CEUR-WS, 2025. URL: https://ceur-ws.org/Vol-4027/paper7.pdf.

[21] A. Gunawardana, G. Shani, S. Yogev, Evaluating Recommender Systems, Springer US, New York, NY, 2022, pp. 547–601. doi:10.1007/978-1-0716-2197-4_15.

[22] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, New York, NY, USA, 2008. Chapter 17: Hierarchical clustering.