Towards Evaluation Guidelines for Empirical Studies involving LLMs

Stefan Wagner*, Marvin Muñoz Barón*, Davide Falessi†, Sebastian Baltes‡
*TUM School of Computation, Information and Technology
Technical University of Munich, Heilbronn, Germany
{stefan.wagner, marvin.munoz-baron}@tum.de
†University of Rome "Tor Vergata", Rome, Italy
falessi@ing.uniroma2.it
‡University of Bayreuth, Bayreuth, Germany
sebastian.baltes@uni-bayreuth.de

***Abstract*—In the short period since the release of ChatGPT, large language models (LLMs) have changed the software engineering research landscape. While there are numerous opportunities to use LLMs for supporting research or software engineering tasks, solid science needs rigorous empirical evaluations. However, so far, there are no specific guidelines for conducting and assessing studies involving LLMs in software engineering research. Our focus is on empirical studies that either use LLMs as part of the research process or studies that evaluate existing or new tools that are based on LLMs. This paper contributes the first set of holistic guidelines for such studies. Our goal is to start a discussion in the software engineering research community to reach a common understanding of our standards for high-quality empirical studies involving LLMs.**

***Index Terms*—Large language models, generative artificial intelligence, empirical studies**

I. Introduction

While artificial intelligence (AI) has been used in software engineering (SE) for a long time, success used to be limited [1]. Recently, the rise of large language models (LLMs) has opened new avenues for the application of AI in software engineering [2], [3]. These models offer many possible use cases, ranging from code generation and bug detection to requirements analysis and software maintenance. For instance, LLM-based tools were able to generate logging statements [4], generate test cases [5], and support education [6].

As a result, we are starting to see an increasing number of evaluation studies either using LLMs as part of the research process [7] or as part of tools that automate or improve software engineering tasks. These studies explore the effectiveness, performance, and robustness of LLMs in different contexts, such as improving code quality, reducing development time, or supporting software documentation. However, it is often unclear how valid and reproducible results can be achieved with empirical studies involving LLMs – or what effect their usage has on the validity of empirical results. This uncertainty poses significant challenges for researchers aiming to draw reliable conclusions from empirical studies.

The work of Davide Falessi has been partially supported by the 2022JJ3PA5 - PRIN2022 project. This work also was partially supported by the German Federal Ministry of Education and Research in the project MEKI (21IVP016F).

One of the primary risks in creating unreproducible results stems from the variability in LLM performance due to differences in training data, model architecture, evaluation metrics, and the inherent non-determinism of those models. For example, slight changes in the training data or the hyperparameters can lead to significantly different outcomes, making it difficult to reproduce studies. Also, the lack of standardized benchmarks and evaluation protocols further complicates the reproducibility. These issues highlight the need for clear guidelines and best practices to ensure that empirical studies with LLMs yield valid and reproducible results.

There has been extensive work developing guidelines for conducting and reporting specific types of empirical studies such as controlled experiments [8], [9] or their replications [10]. We believe that LLMs have specific intrinsic characteristics that require specific guidelines for researchers to achieve an acceptable level of reproducibility. For example, even if we know the specific version of an LLM used for an empirical study, the reported performance for the studied tasks can change over time, especially for commercial models that evolve beyond version identifiers [11]. Moreover, commercial providers do not guarantee the availability of old model versions indefinitely. Besides versions, LLMs' performance widely varies depending on configured parameters such as temperature. Therefore, not reporting the parameter settings impacts the reproducibility of the research.

Even for "open" models such as Llama, we do not know how they were fine-tuned for specific tasks and what the exact training data was [12]. For example, when evaluating LLMs' performance for certain programming tasks, it would be relevant to know whether the solution to a certain problem was part of the training data or not.

Therefore, with this paper, we provide two key contributions: (1) a classification of different types of empirical studies involving LLMs in software engineering research and (2) preliminary guidelines on how to achieve valid and reproducible results in such studies. The most recent version of the study types and guidelines is available online.[1]

[1]https://llm-guidelines.org/

II. Related Work

There are several established guidelines for empirical studies in software engineering, e.g., for experiments [8], [9]. While these guidelines continue to be useful, they were developed before the rise of LLMs. Therefore, this paper is a starting point for extending the existing set of guidelines. To the best of our knowledge, the only similar work is a paper by Sallou, Durieux, and Panichella [13], in which they also call for a broader discussion in the community. They mostly focus on a discussion of threats to validity, proposing guidelines that partly overlap with ours. However, they do not structure their guidelines according to different types of studies. We are convinced that the diversity of studies involving LLMs requires a differentiation between study types. Our taxonomy presented in Section III is a first step in that direction.

III. Types of Studies

The development of empirical guidelines for studies involving LLMs in software engineering is crucial for ensuring the validity and reproducibility of results. However, these guidelines must be tailored for different study types as they may pose unique challenges. Therefore, understanding the classification of these studies is essential for developing appropriate guidelines. We envision that a mature set of guidelines provides specific guidance for each of these study types, addressing their individual methodological idiosyncrasies.

A. LLMs as Tools for Researchers in Empirical Studies

LLMs can be leveraged as powerful tools to assist researchers conducting empirical studies. They can automate various tasks such as data collection, preprocessing, and analysis. For example, LLMs can extract relevant information from large datasets, generate summaries of research papers, and even assist in writing literature reviews. This can significantly reduce the time and effort required by researchers, allowing them to focus on more complex aspects of their studies.

1) LLMs as Annotators: LLMs can serve as annotators by automatically labeling artifacts with corresponding categories for data analysis. For example, in a study analyzing code changes in version control systems, researchers may need to categorize each individual change. For that, they may use LLMs to analyze commit messages and categorize them into predefined labels such as bug fixes, feature additions, or refactorings. This automation can improve the efficiency of the annotation process, which is often a labor-intensive and error-prone task when done manually. Moreover, in qualitative data analysis, manually annotating or coding text passages is also an often time-consuming manual process. LLMs can be used to augment human annotations, provide suggestions for new codes, or even automate the entire process. In such tasks, LLMs have the potential to improve the accuracy and efficiency of automated labeling processes [14], making them valuable tools for empirical research in software engineering. Hybrid human-LLM annotation approaches may further increase accuracy and allow for the correction of incorrectly applied labels [15].

2) LLMs as Judges: In empirical studies, LLMs can act as judges to evaluate the quality of software artifacts such as code, documentation, and design patterns. For instance, LLMs can be trained to assess code readability, adherence to coding standards, and the quality of comments. By providing rather objective and consistent evaluations, LLMs could help mitigate certain biases and part of the variability that human judges might introduce. This could lead to more reliable and reproducible results in empirical studies. However, when relying on the judgment of LLMs, researchers have to make sure to build a reliable process for generating ratings that considers the non-deterministic nature of LLMs and report the intricacies of that process transparently.

3) LLMs as Subjects: LLMs can be used as subjects in empirical studies to simulate human behavior and interactions. For example, researchers can use LLMs to generate responses in user studies, simulate developer interactions in collaborative coding environments, or model user feedback in software usability studies. This approach can provide valuable insights while reducing the need to recruit human participants, which can be time-consuming and costly. Additionally, using LLMs as subjects allows for controlled experiments with consistent and repeatable conditions. However, when using LLMs as study subjects, it is important that researchers are aware of their inherent biases [16] and limitations [17].

B. LLMs for New Tools Supporting Software Engineers

LLMs are being integrated into new tools designed to support software engineers in their daily tasks. These tools can include intelligent code editors that provide real-time code suggestions, automated documentation generators, and advanced debugging assistants. Empirical studies can evaluate the effectiveness of these tools in improving productivity, code quality, and developer satisfaction. By assessing the impact of LLM-powered tools, researchers can identify best practices and areas for further improvement. For example, Choudhuri et al. [18] conducted a student experiment in which they measured the impact of ChatGPT on the correctness and completion time for programming tasks.

C. Studying LLM Usage

Empirical studies can also focus on understanding how software engineers use LLMs in their workflows. This involves investigating the adoption, usage patterns, and perceived benefits and challenges of LLM-based tools. Surveys, interviews, and observational studies can provide insights into how LLMs are integrated into development processes, how they influence decision-making, and what factors affect their acceptance and effectiveness. Such studies can inform the design of more user-friendly and effective LLM-based tools. For example, Khojah et al. [19] investigated the use of ChatGPT by professional software engineers in a week-long observational study.

D. Benchmarking LLMs for SE Tasks

Another typical type of study focuses on benchmarking LLM output quality on large datasets. In software engineering,

this may include the evaluation of LLMs' ability to produce accurate and robust outputs for input data from real-world projects or synthetically created SE datasets. In studies with generative models, the LLM output is often compared against a ground truth dataset using similarity metrics such as ROUGE, BLEU, or METEOR [3]. Moreover, the evaluation may be augmented by using task-specific or artifact-specific measures. Such measures may include code quality or performance metrics for code generation tasks or readability metrics for natural language SE artifacts (e.g., requirements documents). In this context, reference datasets such as HumanEval [20] play an important role in establishing standardized evaluations. However, benchmark contamination [21] has recently been identified as an issue. The careful creation of samples and corresponding input prompts is particularly important, as correlations between prompts may bias benchmark results [22].

IV. Preliminary Guidelines

While providing a comprehensive set of guidelines is beyond the scope of this position paper, we report a first set of guidelines based on a discussion session with other empiricism experts at the 2024 International Software Engineering Research Network (ISERN) meeting.[2] This paper is meant as a starting point for further discussions in the community with the aim of developing a common understanding of how we should conduct and report empirical studies involving LLMs.

A. Declare LLM Usage and Role

When conducting any kind of empirical study involving LLMs, it is essential to clearly declare that an LLM was used. This includes specifying the purpose of using the LLM in the study, the tasks it was applied to, and the expected outcomes. Transparency in the usage of LLMs helps in understanding the context and scope of the study, facilitating better interpretation and comparison of results. Beyond this declaration, we recommend that the authors be explicit about the LLM's exact role. Oftentimes, there is a complex layer around the LLM that preprocesses data, prepares prompts, or filters user requests. One example is ChatGPT, which can, among others, use the GPT-4o model. GitHub Copilot uses the same model as well, and researchers can build their own tools utilizing GPT-4o directly (e.g., via the OpenAI API). The infrastructure around the bare model can significantly contribute to the performance of a model in a certain task. Therefore, it is crucial that researchers clearly describe what the LLM contributes to the tool or method presented in a research paper.

B. Report Model Version and Date

It is also crucial for all types of studies to document the specific version of the LLM used in the study, along with the date when the experiments were conducted. LLMs are frequently updated, and different versions may produce varying results. By providing this information, researchers enable others to reproduce the study under the same conditions. Different model providers have varying degrees of information. For example, OpenAI provides a model version and a system fingerprint describing the backend configuration that can also influence the output. Therefore, stating "We used gpt-4o-2024-08-0, system fingerprint fp_6b68a8204b" provides clarity on the exact model and runtime environment. However, the main purpose of the system fingerprint is detecting changes and going back to a previous system fingerprint is impossible.

C. Report Model Configuration

Detailed documentation of the configuration and parameters used during any study is necessary for reproducibility. This includes settings such as the temperature that controls randomness, the maximum token length, and any other relevant parameters such as the consideration of historical context. Additionally, a thorough description of the hosting environment of the LLM or LLM-based tool should be provided, especially in studies focusing on performance or any time-sensitive measurement. For instance, researchers might report that "the model was integrated via the Azure OpenAI Service, and configured with a temperature of 0.7, top_p set to 0.8, and a maximum token length of 512," providing a clear overview of the experimental setup. Using seed values does further increase reproducibility, but does not completely mitigate the issue of non-determinism.[3]

D. Report Prompts and their Development

Reporting the exact prompts used in the study is essential for transparency and reproducibility. Prompts can significantly influence the output of LLMs [23], and sharing them allows other researchers to understand and reproduce the conditions of the study. For example, including the specific questions or tasks given to the LLM helps assess the validity of the results and compare them with other studies. This is an example where different types of studies require different information. When studying LLM usage, the researchers ideally collect and publish the prompts written by the users (if confidentiality allows). Otherwise, summaries and examples can be provided. Prompts also need to be reported when LLMs are integrated into new tools, especially if study participants were able to formulate (parts of) the prompts. For all other types of studies, researchers should discuss how they arrived at their final set of prompts. If a systematic approach was used, this process should be described in detail.

E. Use an Open LLM as a Baseline

To ensure the reproducibility of results, we recommended findings be reported with an open LLM as a baseline. This applies both when using LLMs as tools for supporting researchers in empirical studies and when benchmarking LLMs for SE tasks. In case LLMs are integrated into new tools, this is also preferable if the architecture of the tool allows it. If the effort of changing models is too high, researchers should at least report an initial benchmarking with open models, which enables more objective comparisons. Open LLMs can either be hosted via cloud platforms such as *Hugging Face* or used

[2]https://isern.fraunhofer.de

[3]Open AI Cookbook: The new seed parameter

locally via tools such as *ollama* or *LM Studio*. A replication package for papers using LLMs should include clear instructions that allow other researchers to reproduce the findings using open models. This practice enhances the credibility of the study and allows for independent verification of the results. Researchers could, e.g., mention that "results were compared with those obtained using Meta's Code LLAMA, available on the Hugging Face platform" and point to a replication package.

We are aware that the definition of an "open" model is actively being discussed, and many open models are essentially only "open weight" [12]. We consider the *Open Source AI Definition* proposed by the *Open Source Initiative* (OSI) to be a first step towards defining true open-source models [24].

F. Use Human Validation for LLM Outputs

Especially in studies where LLMs are used to support researchers, human validation should always be employed. While LLMs can automate many tasks, it is important to validate their outputs with human annotations, at least partially. For natural language processing tasks, a large-scale study has shown that LLMs have too large a variation in their results to be reliably used as a substitution for human judges [25]. Human validation helps ensure the accuracy and reliability of the results, as LLMs may sometimes produce incorrect or biased outputs. Incorporating human judgment in the evaluation process adds a layer of quality control and increases the trustworthiness of the study's findings, especially when explicitly reporting inter-rater reliability metrics. For instance, "A subset of 20 % of the LLM-generated annotations were reviewed and validated by experienced software engineers to ensure accuracy. An inter-rater reliability of 90 % was reached." For studies using LLMs as annotators, the proposed process by Ahmed et al. [26], which includes an initial few-shot learning and, given good results, the replacement of *one* human annotator by an LLM, might be a way forward.

V. Conclusions

In this paper, we outlined preliminary guidelines for researchers reporting on empirical studies involving Large Language Models (LLMs) in software engineering research. While researchers can already use these guidelines to improve the reproducibility of their studies and the reporting in their papers, we see a demand for more tailored guidelines focusing on the different study types we identified in Section III and also for extending the guidelines to include missing aspects such as ethical implications of using LLMs in research [27]. One aspect to focus on could be the particular types of threats to validity that arise from using LLMs in the context of different study types, building on related work [13]. Another direction could be to conduct a critical review of published studies involving LLMs, assessing how many papers already adhere to the guidelines we suggest.

References

[1] S. Martínez-Fernández, J. Bogner, X. Franch, M. Oriol, J. Siebert, A. Trendowicz, A. M. Vollmer, and S. Wagner, "Software engineering for AI-based systems: A survey," *ACM TOSEM*, vol. 31, no. 2, 2022.

[2] A. Fan, B. Gokkaya, M. Harman, M. Lyubarskiy, S. Sengupta, S. Yoo, and J. M. Zhang, "Large language models for software engineering: Survey and open problems," in *ICSE 2024: Future of Software Engineering (ICSE-FoSE)*, pp. 31–53, IEEE, 2023.

[3] X. Hou, Y. Zhao, Y. Liu, Z. Yang, K. Wang, L. Li, X. Luo, D. Lo, J. Grundy, and H. Wang, "Large language models for software engineering: A systematic literature review," *ACM TOSEM*, Sept. 2024.

[4] Y. Li, Y. Huo, Z. Jiang, R. Zhong, P. He, Y. Su, L. C. Briand, and M. R. Lyu, "Exploring the effectiveness of LLMs in automated logging statement generation: An empirical study," *IEEE TSE*, 2024.

[5] R. Santos, I. Santos, C. Magalhaes, and R. de Souza Santos, "Are we testing or being tested? exploring the practical applications of large language models in software testing," in *ICST 2024*, 2024.

[6] J. Pereira, J.-M. López, X. Garmendia, and M. Azanza, "Leveraging open source LLMs for software engineering education and training," in *CSEE&T 2024*, pp. 1–10, 2024.

[7] M. Bano, R. Hoda, D. Zowghi, and C. Treude, "Large language models for qualitative research in software engineering: exploring opportunities and challenges," *ASE Journal*, vol. 31, no. 1, p. 8, 2024.

[8] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer, 2024.

[9] A. Jedlitschka, M. Ciolkowski, and D. Pfahl, "Reporting experiments in software engineering," in *Guide to Advanced ESE* (F. Shull, J. Singer, and D. I. K. Sjøberg, eds.), pp. 201–228, Springer, 2008.

[10] A. Santos, S. Vegas, M. Oivo, and N. Juristo, "A procedure and guidelines for analyzing groups of software engineering replications," *IEEE TSE*, vol. 47, no. 9, pp. 1742–1763, 2021.

[11] L. Chen, M. Zaharia, and J. Zou, "How is ChatGPT's behavior changing over time?," *CoRR*, vol. abs/2307.09009, 2023.

[12] E. Gibney, "Not all 'open source' AI models are actually open," *Nature News*, 2024.

[13] J. Sallou, T. Durieux, and A. Panichella, "Breaking the silence: the threats of using LLMs in software engineering," in *ICSE 2024 NIER*, 2024.

[14] M. Wan, T. Safavi, S. K. Jauhar, Y. Kim, S. Counts, J. Neville, S. Suri, C. Shah, R. W. White, L. Yang, *et al.*, "TnT-LLM: Text mining at scale with large language models," in *ACM KDD 2024*, pp. 5836–5847, 2024.

[15] X. Wang, H. Kim, S. Rahman, K. Mitra, and Z. Miao, "Human-LLM collaborative annotation through effective verification of llm labels," in *CHI 2024*, pp. 1–21, 2024.

[16] R. Crowell, "Why AI's diversity crisis matters, and how to tackle it," *Nature Career Feature*, 2023.

[17] J. Harding, W. D'Alessandro, N. G. Laskowski, and R. Long, "AI language models cannot replace human research participants," *AI Soc.*, vol. 39, no. 5, pp. 2603–2605, 2024.

[18] R. Choudhuri, D. Liu, I. Steinmacher, M. Gerosa, and A. Sarma, "How far are we? The triumphs and trials of generative AI in learning software engineering," in *ICSE 2024*, pp. 1–13, 2024.

[19] R. Khojah, M. Mohamad, P. Leitner, and F. G. de Oliveira Neto, "Beyond code generation: An observational study of ChatGPT usage in software engineering practice," *PACMSE*, vol. 1, no. FSE, pp. 1819–1840, 2024.

[20] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, *et al.*, "Evaluating large language models trained on code," *arXiv:2107.03374*, 2021.

[21] S. Ahuja, V. Gumma, and S. Sitaram, "Contamination report for multilingual benchmarks," 2024.

[22] C. Siska, K. Marazopoulou, M. Ailem, and J. Bono, "Examining the robustness of LLM evaluation to the distributional assumptions of benchmarks," in *ACL 2024 (Long Papers)*, pp. 10406–10421, 2024.

[23] Y. Liu, T. Le-Cong, R. Widyasari, C. Tantithamthavorn, L. Li, X.-B. D. Le, and D. Lo, "Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues," *ACM TOSEM*, June 2024.

[24] OSI, "Open Source AI Definition 1.0." https://opensource.org/ai/open-source-ai-definition. Accessed 2024-11-11.

[25] A. Bavaresco, R. Bernardi, L. Bertolazzi, D. Elliott, R. Fernández, A. Gatt, E. Ghaleb, M. Giulianelli, M. Hanna, A. Koller, *et al.*, "LLMs instead of human judges? a large scale empirical study across 20 nlp evaluation tasks," *arXiv preprint arXiv:2406.18403*, 2024.

[26] T. Ahmed, P. Devanbu, C. Treude, and M. Pradel, "Can LLMs replace manual annotation of software engineering artifacts?," *arXiv preprint arXiv:2408.05534*, 2024.

[27] E. L. Ungless, N. Vitsakis, Z. Talat, J. Garforth, B. Ross, A. Onken, A. Kasirzadeh, and A. Birch, "Ethics whitepaper: Whitepaper on ethical research into large language models," *arXiv:2410.19812*, 2024.