OLAF: Towards Robust LLM-Based Annotation Framework in Empirical Software Engineering

Mia Mohammad Imran
Missouri University of Science and Technology
Rolla, Missouri, USA
imranm@mst.edu

Tarannum Shaila Zaman
University of Maryland Baltimore County
Baltimore, Maryland, USA
zamant@umbc.edu

Abstract

Large Language Models (LLMs) are increasingly used in empirical software engineering (ESE) to automate or assist annotation tasks such as labeling commits, issues, and qualitative artifacts. Yet the reliability and reproducibility of such annotations remain underexplored. Existing studies often lack standardized measures for reliability, calibration, and drift, and frequently omit essential configuration details. We argue that LLM-based annotation should be treated as a measurement process rather than a purely automated activity. In this position paper, we outline the **Operationalization for LLM-based Annotation Framework (OLAF)**, a conceptual framework that organizes key constructs: *reliability, calibration, drift, consensus, aggregation*, and *transparency*. The paper aims to motivate methodological discussion and future empirical work toward more transparent and reproducible LLM-based annotation in software engineering research.

Keywords

Large Language Models, Annotation, Measurement, Reproducibility, Empirical Software Engineering

1 Introduction

Empirical software engineering depends on labeled data, such as issues identified as bugs, code comments marked as self-admitted technical debt, or developer commit messages classified by intent [14]. Traditionally, producing such annotations has required substantial human effort. Manual annotation, though considered as gold standard, is labor-intensive and can be inconsistent [17]. The rise of large language models (LLMs) has transformed this process. In recent years, Researchers have employed models such as GPT, Claude, or Llama to generate preliminary labels or even to replace manual annotation entirely [2, 10, 13].

While this approach accelerates data production, it also introduces new methodological concerns. When an LLM assigns a label, it effectively functions as a *measurement instrument.* Just as code metrics must be validated before use, labels produced by LLMs need to be tested for reliability and reproducibility. As noted by Wagner et al., many SE studies involving LLMs neglect to report standard measures of reliability or calibration [25]. Baumann et al. observe that minor configuration choices, such as prompt phrasing or sampling temperature, can completely reverse outcomes, a form of prompt drift – which they refer as *LLM hacking* [6]. Ahmed et al. find that LLMs can perform on par with human annotators, but only under carefully controlled evaluation conditions [2]. Meanwhile, Wang et al. empirically studied LLM-as-a-Judge in SE tasks and noted that LLMs could potentially substitute human evaluations in certain SE tasks while they may be provide satisfactory and consistent performance in other tasks [26].

These findings highlight a key gap: the lack of operationalization—the formal link between abstract concepts (e.g., "label accuracy") and measurable procedures [15]. Without clear operational definitions for constructs like reliability, consensus, or drift, ESE cannot guarantee reproducibility. While existing guidelines, such as those by ACM and Baltes et al. [4], offer general advice on adopting LLMs in SE, they do not specifically address the need for operationalizing annotations in SE tasks. To fill this gap, we introduce the *Operationalization for LLM-based Annotation Framework (OLAF)*, a structured approach that treats LLM-based annotation as a measurable and auditable process. To lay the groundwork for OLAF, we first explore why operationalization is essential and how current practices fall short in ensuring methodological rigor.

2 Need for Operationalization in LLM-based Annotation

2.1 Fragmented Annotation Practices

The challenge of using LLMs for annotation lies not in performance but in methodological rigor. Many studies lack clear label definitions and consistent operationalization of reliability, calibration, or consensus, and often omit key configuration details (e.g., temperature). Based on recent evaluations [2, 6, 25, 26], we summarizes these weaknesses and their implications.

2.1.1 Missing Details. Ahmed et al. [2] noted that even when models reach human-level accuracy, details such as prompt templates, model versions, and decoding parameters are sometimes omitted. Without these details, replication becomes uncertain. Several patterns appear from recent observations [2, 6, 25]:

- **Limited reliability reporting.** Some studies provide only raw accuracy without chance-corrected measures (e.g., Cohen's κ).

- **Incomplete configuration description.** Prompt templates, model versions, and decoding parameters are often underspecified.
- **Minimal calibration assessment.** Model confidence and stability across versions are rarely examined. For example, researchers reported the performance difference between different GPT model versions during annotation [3].

2.1.2 Configuration Sensitivity. Baumann et al. [6] found that minor configuration changes, such as prompt wording, temperature, or model version, can alter results in text-annotation tasks. This sensitivity complicates the expectation of stable measurement conditions. In practice, it appears as:

- **Prompt variation:** minor lexical variations (e.g., "bug fix" vs. "defect repair") yield different outputs. The problem of *Prompt Perturbation* [8] is still an under-explored area in general;
- **Model variation:** API updates modify model behavior under the same identifier;
- **Sampling variation:** stochastic decoding introduces variability across runs.

2.1.3 Unstandardized Constructs and Metrics. ESE commonly employs metrics for inter-rater reliability and measurement validity. In contrast, LLM-based studies rarely define comparable constructs for annotation quality. Ahmed et al. [2] propose human-model agreement ($\kappa_{H\text{-}M}$) and model-model consensus ($\rho_{M\text{-}M}$) as candidate indicators, while Baumann et al. discuss model sensitivity to configuration [6].

2.1.4 Consequences for Research Validity. The absence of operationalization undermines the validity of results across all levels of empirical reasoning. Construct validity suffers when labels lack defined meaning; internal validity is threatened by uncontrolled configuration sensitivity; and external validity is compromised by opaque or evolving model configurations that hinder replication.

2.2 Problem Summary

LLM-based annotation has advanced rapidly, but its methodological foundations remain underdeveloped. Existing studies follow fragmented practices, report reliability inconsistently, and rely on loosely defined constructs, making results difficult to interpret or reproduce. As a result, LLM-generated labels are often treated as measurements without being properly validated. Recent methodological analyses [2, 6, 25] highlight the need to reconceptualize annotation as a structured measurement process rather than a purely automated task. Moreover, when annotation targets lack a clear gold standard or involve irreducible expert disagreement, reliability and calibration become ill-defined, rendering probabilistic outputs unsuitable for safety-critical or compliance-sensitive settings. Addressing these challenges requires standardized constructs, transparent configuration reporting, and reproducible aggregation methods. In the following section, we introduce *OLAF*, a framework designed to meet these requirements.

3 Operationalization for LLM-Based Annotation Framework (OLAF)

We provide detailed description of the proposed framework below.

3.1 Measurement of LLM Annotation

The methodological issues that we discussed earlier, fragmented practices, configuration sensitivity, and undefined constructs, stem from treating LLM annotation as an *automation task* rather than as *measurement*. When a model assigns a label, it acts as a measurement instrument whose reliability must be empirically established.

Recognizing annotation as measurement provides a foundation for explicit operational constructs and reproducible procedures. Following Ahmed et al. [2] and Wagner et al. [25], we define six key dimensions that characterize LLM annotation as a measurable process: *reliability, consensus, calibration, drift, aggregation*, and *transparency*. Table 1 summarizes them. Each specifies a property of measurement that can be operationalized and reported. We note that *these constructs presuppose that the target construct is observable and supports a stable measurement interpretation. When this prerequisite is not met, the proposed operationalization may not be suitable for the task.* Below we describe the six metrics that form the conceptual and operational basis of *OLAF*:

(1) **Annotator Reliability** [11] measures agreement across individual annotators, human or model. Chance corrected statistics such as Cohen's κ [11] and Krippendorff's α [18] quantify consistency beyond random alignment.
(2) **Consensus** [7] reflects group level agreement, indicating whether annotators converge on a shared judgment or expose ambiguity in the task. Correlation based metrics capture the strength of this collective alignment.
(3) **Aggregation** [19] combines multiple annotators into a single label. We outline two approaches: i) probabilistic aggregation, which infers latent labels by modeling annotator error patterns (e.g., Dawid–Skene, GLAD, MACE) [16]; and ii) majority voting, suitable when agreement is high and multiple LLMs have comparable reliability [24].
(4) **Transparency** [20] concerns documentation quality, ensuring reproducibility through complete disclosure of model identifiers, parameters, prompts, and settings, following practices such as Datasheets and Model Cards [23].
(5) **Calibration** [12] evaluates probability correctness by comparing predicted confidences with empirical accuracy. Metrics such as expected calibration error and Brier Score quantify miscalibration and indicate how trustworthy model probabilities are when weighting or aggregating labels [28].
(6) **Drift** [1] captures stability, measuring how much model behavior changes under prompt or context variation. Drift is quantified using activation distances (L2 or cosine), output distribution metrics such as Jensen-Shannon divergence (JSD), or prompt sensitivity indices like POSIX [8].

Note that not all six dimensions may not be required for every Software Engineering annotation task. For example, some SE tasks that rely on subjective judgments (e.g., *sentiment/emotion analysis, bug severity identification*) may benefit from reporting reliability or consensus, while tasks with more deterministic outputs (e.g., *API extraction, test-input generation*) may not require all dimensions. Practitioners should select the dimensions that meaningfully characterize the measurement properties of the specific task rather than applying the full set by default.

Table 1: Operationalization for LLM-based Annotation Framework (OLAF)

Construct	Measures	Metric / Indicator	Reporting Focus
Annotator's Reliability [11]	Annotator agreement	Cohen's κ [11], Krippendorff's α [18]	Stability of individual annotators
Consensus [7]	Group agreement	Correlation metrics	Collective alignment
Aggregation [19]	Combining annotators	i) probabilistic [16]: Dawid-Skene, GLAD, MACE; ii) Majority voting [24]	Quality of fused labels
Transparency [20]	Documentation quality	Model (versions, parameter size, etc) and prompt details	Reproducible annotation setup
Calibration [12]	Probability correctness	ECE, Brier Score [28]	Reliability of confidence values
Drift [1]	Behavioral stability	L2, cosine, JSD, POSIX [8]	Robustness to prompt variation

Table 2: Configurations of LLM Participation in Annotation Workflows

Configuration	Human Role	Primary Benefit	Main Risk
Human-in-the-Loop [22]	High	Reduces cost by auditing low-confidence model outputs.	Confidence miscalibration; selective review bias.
LLM/Model-in-the-Loop [5]	Moderate	Combines model suggestion and cross-model validation to improve consistency and speed.	Anchoring bias; prompt drift; compounded error propagation.
Verifier-in-the-Loop [27]	Moderate	Prioritizes human review via automated verification of LLM-generated labels.	Verifier bias; dependence on small gold label sets; explanation quality sensitivity.
LLM-as-a-Pre-screen-Filter [9]	High	Scalable pre-screening of large corpora before human annotation.	Coverage bias; missing rare but important samples.
LLM-as-a-Judge [26]	Low	Objective rubric-based evaluation of generated artifacts.	Evaluation bias; rubric drift; lack of interpretability.
LLM-as-an-Annotator [2]	None	Fully automated dataset labeling at minimal cost.	Reproducibility loss from hidden drift and version changes.

3.2 Context of Annotation Workflows

To operationalize these constructs in practice, it is necessary to account for how LLMs are embedded in the annotation workflow. Below we discuss six common annotation configurations [2, 6, 25].

Human-in-the-Loop (HitL) configurations pre-label artifacts using a model and require human validation of uncertain or low-confidence outputs [22]. This reduces manual effort while maintaining quality control, though over-reliance on model confidence can propagate unverified errors. Typical indicators include κ, ECE, and selective-review accuracy.

LLM/Model-in-the-Loop (MitL) integrates one or more machine learning/large language models as intermediate agents in the annotation pipeline [5]. Models generate candidate labels, rationales, or validation feedback, which are reviewed by humans or auxiliary models. Key metrics include human-model agreement ($\kappa_{H\text{-}M}$), cross-model agreement ($\kappa_{M\text{-}M}$), prompt sensitivity index (POSIX), and intra-annotator variance.

Verifier-in-the-Loop (VitL) is a configuration where a dedicated verifier model assesses the validity of LLM-generated annotations prior to human review [27]. The verifier predicts a confidence score indicating whether each label should be accepted, based on inputs such as the source text and model-generated explanation. Typical indicators include verifier accuracy, ECE, and change in reliability ($\Delta\kappa$) after selective re-annotation.

Model/LLM-as-Initial-Filter configurations employ a model to discard clearly irrelevant items before manual or HitL processing [9]. This improves scalability and efficiency but risks coverage bias by omitting rare or ambiguous cases. Metrics include recall of relevant items, false-negative rate, and $\Delta\kappa$ across filter runs.

LLM-as-a-Judge (LLMaaJ) configurations treat models as automated evaluators applying rubrics to generated outputs such as explanations or summaries [21, 25, 26]. These yield consistent rubric enforcement but are vulnerable to evaluation bias and rubric drift. Representative measures include human-LLM agreement, rubric adherence, and prompt-sensitivity index.

LLM-as-an-Annotator configurations delegate full labeling responsibility to the model [2]. This maximizes throughput and internal consistency but removes interpretability and introduces dependence on model parameters and version stability. Core metrics include inter-model consensus (ρ), ECE, and drift ($\Delta\kappa$).

These configurations are summarized in Table 2.

4 Guidelines for LLM-Based Annotation in Empirical Software Engineering

We provide guideline for applying *OLAF* below.

Declare the Annotation Configuration. Researchers should explicitly state how LLMs participate in the annotation workflow (e.g., HitL, MitL, VitL, LLM-as-a-Filter, and LLMaaJ), as this determines which reliability and calibration measures are meaningful.

Document Model and Prompt Details. Report the LLM provider, model version, decoding parameters, and full prompt templates. Even minor prompt variations (e.g., 'bug' vs. 'defect') can affect annotation outcomes in SE tasks.

Aggregation and Reliability. When multiple LLMs are used, they should be treated as independent annotators and aggregated based on observed agreement. Majority voting or probabilistic aggregation can be applied. For example, in self-admitted technical debt detection, labels produced by different LLMs are compared to assess

inter-model agreement. If agreement exceeds the predefined threshold, majority voting is used; otherwise, probabilistic methods such as Dawid-Skene are applied to infer latent labels while accounting for systematic differences between models.

Calibration and Drift Tracking. For SE tasks where LLMs output confidence scores, calibration should be evaluated. Drift should be monitored using a fixed, task-specific calibration subset. For example, consider issue classificaton annotation, a calibration subset of 50-100 issues annotated by at least two experts with substantial agreement (e.g., Cohen's $\kappa \geq 0.6$) is retained. When the model version, prompt, or decoding parameters change, the LLM is re-applied to this subset and drift is quantified using changes in agreement ($\Delta\kappa$) and label distributions (e.g., Jensen-Shannon divergence). Another example, in a bug severity or priority annotation task, calibration is assessed by comparing model confidence scores against expert labels using Brier score or expected calibration error. Poor calibration indicates that confidence values should not be used for selective review or weighted aggregation.

We outline further detailed guideline and examples here: https://se-llm-annotation-olaf.github.io/olaf/

5 Limitations.

OLAF treats LLMs as measurement instruments with quantifiable but limited stability. This assumption is necessarily approximate, particularly for proprietary models such as GPT and Claude, whose training data, optimization processes, and update schedules are opaque, and whose API versioning does not ensure parameter immutability. Consequently, calibration, drift, and reliability metrics can only bound observable variability. Open-weight and open-source models, such as, Mistral, Llama, and Qwen, reduce some of these uncertainties through controlled inference settings and fixed checkpoints, but they still exhibit stochastic decoding, prompt sensitivity, and configuration-dependent variance. OLAF therefore adopts a notion of constrained stability, focusing on measurable regularities rather than deterministic or fully specified behavior.

6 Conclusion and Future Plan

LLMs are increasingly used in empirical software engineering, but their methodological grounding remains limited. This paper introduced *OLAF*, a framework that treats LLM-based annotation as a reproducible measurement process through six constructs: reliability, calibration, drift, consensus, aggregation, and transparency. *OLAF* is an initial step. Our future work will empirically evaluate its impact on reproducibility and transparency, refine its assumptions for socio-technical LLMs, and develop SE-specific operational examples such as calibration subsets and drift-check procedures.

References
[1] Sahar Abdelnabi, Aideen Fay, Giovanni Cherubin, Ahmed Salem, Mario Fritz, and Andrew Paverd. 2025. Get my drift? catching llm task drift with activation deltas. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning*.
[2] Toufique Ahmed, Premkumar Devanbu, Christoph Treude, and Michael Pradel. 2025. Can llms replace manual annotation of software engineering artifacts?. In *2025 IEEE/ACM 22nd International Conference on Mining Software Repositories*.
[3] Mohammed Aldeen, Joshua Luo, Ashley Lian, Venus Zheng, Allen Hong, Preethika Yetukuri, and Long Cheng. 2023. Chatgpt vs. human annotators: A comprehensive analysis of chatgpt for text annotation. In *2023 International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 602–609.
[4] Sebastian Baltes, Florian Angermeir, Chetan Arora, Marvin Muñoz Barón, Chunyang Chen, et al. 2025. Guidelines for Empirical Studies in Software Engineering involving Large Language Models. arXiv:2508.15503
[5] Max Bartolo, Tristan Thrush, Sebastian Riedel, et al. 2022. Models in the Loop: Aiding Crowdworkers with Generative Annotation Assistants. In *Proceedings of the 2022 Conference of the NAACL: Human Language Technologies*.
[6] Julian Baumrin, Thomas Wachinger, Leonhard Dorsch, et al. 2025. Large Language Model Hacking: Quantifying the Hidden Risks of Using LLMs for Text Annotation. *arXiv preprint arXiv:2509.08825* (2025).
[7] Maarten Buyl, Hadi Khalaf, Claudio Mayrink Verdun, Lucas Monteiro Paes, et al. 2025. AI Alignment at Your Discretion. In *Proceedings of the 2025 ACM Conference on Fairness, Accountability, and Transparency*. 3046–3074.
[8] Anwoy Chatterjee, HSVNS Kowndinya Renduchintala, Sumit Bhatia, and Tanmoy Chakraborty. 2024. POSIX: A Prompt Sensitivity Index For Large Language Models. In *Findings of the ACL: EMNLP 2024*.
[9] Juhwan Choi, Jungmin Yun, Kyohoon Jin, and Youngbin Kim. 2024. Multi-News+: Cost-efficient Dataset Cleansing via LLM-based Data Annotation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.
[10] Alexander Churchill, Shamitha Pichika, Chengxin Xu, and Ying Liu. 2025. GPT models for text annotation: An empirical exploration in public policy research. *Policy Studies Journal* (2025).
[11] Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement* (1960).
[12] Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A Survey of Confidence Estimation and Calibration in Large Language Models. In *Proceedings of the 2024 Conference of the NAACL: HLT*.
[13] Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. ChatGPT outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences* 120, 30 (2023), e2305016120.
[14] Yann-Gaël Guéhéneuc and Foutse Khomh. 2019. *Empirical Software Engineering*. Springer International Publishing.
[15] Matthias Haucke, Rink Hoekstra, and Don Van Ravenzwaaij. 2021. When numbers fail: do researchers agree on operationalization of published research? *Royal Society Open Science* 8, 9 (2021), 191354.
[16] Zeyu He, Chieh-Yang Huang, Chien-Kuang Cornelia Ding, et al. 2024. If in a crowdsourced data annotation pipeline, a gpt-4. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*.
[17] Nasif Imtiaz, Justin Middleton, Peter Girouard, and Emerson Murphy-Hill. 2018. Sentiment and politeness analysis tools on developer discussions are unreliable, but so are people. In *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*. 55–61.
[18] Klaus Krippendorff. 2018. *Content Analysis: An Introduction to Its Methodology* (4th ed.). Sage Publications.
[19] Minzhi Li, Zhengyuan Liu, Shumin Deng, Shafiq Joty, Nancy Chen, and Min-Yen Kan. 2025. DnA-Eval: Enhancing Large Language Model Evaluation through Decomposition and Aggregation. In *Proceedings of the 31st International Conference on Computational Linguistics*. 2277–2290.
[20] Q Vera Liao and Jennifer Wortman Vaughan. 2024. AI Transparency in the Age of LLMs: A Human-Centered Research Roadmap. *Harvard Data Science Review* Special Issue 5 (2024).
[21] Sebastian Lubos, Alexander Felfernig, Thi Ngoc Trang Tran, Damian Garber, Merfat El Mansi, Seda Polat Erdeniz, and Viet-Man Le. 2024. Leveraging llms for the quality assurance of software requirements. In *2024 IEEE 32nd International Requirements Engineering Conference (RE)*. IEEE, 389–397.
[22] Robert Munro Monarch. 2021. *Human-in-the-Loop Machine Learning: Active learning and annotation for human-centered AI*. Simon and Schuster.
[23] Ernesto Lang Oreamuno, Rohan Faiyaz Khan, Abdul Ali Bangash, Catherine Stinson, and Bram Adams. 2024. The state of documentation practices of third-party machine learning models and datasets. *IEEE Software* 41, 5 (2024), 52–59.
[24] Rion Snow, Brendan O'connor, Dan Jurafsky, and Andrew Y Ng. 2008. Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 conference on EMNLP*.
[25] Stefan Wagner, Marvin Muñoz Barón, Davide Falessi, and Sebastian Baltes. 2025. Towards evaluation guidelines for empirical studies involving llms. In *2025 IEEE/ACM International Workshop on Methodological Issues with Empirical Studies in Software Engineering (WSESE)*. IEEE, 24–27.
[26] Ruiqi Wang, Jiyu Guo, Cuiyun Gao, Guodong Fan, Chun Yong Chong, and Xin Xia. 2025. Can llms replace human evaluators? an empirical study of llm-as-a-judge in software engineering. *Proceedings of the ACM on Software Engineering* 2, ISSTA (2025), 1955–1977.
[27] Xinru Wang, Hannah Kim, Sajjadur Rahman, Kushan Mitra, and Zhengjie Miao. 2024. Human-LLM Collaborative Annotation Through Effective Verification of LLM Labels. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. doi:10.1145/3613904.3641960
[28] Dustin Wright and Isabelle Augenstein. 2025. Aggregating soft labels from crowd annotations improves uncertainty estimation under distribution shift. *PLoS One* 20, 6 (2025), e0323064.