Can LLMs Write Reliable Rubrics? A Meta-Evaluation for Experiment Reproduction

Anonymous ACL submission

Abstract

Rubric-based evaluation is a promising approach for assessing open-ended outputs from LLM-based research agents, particularly in paper reproduction, where direct paper-to-repository comparison is prone to hallucination. However, constructing paper-specific rubrics requires substantial expert effort, limiting the scalability of benchmarks such as PaperBench. In this work, we present, to our knowledge, the first systematic meta-evaluation of LLM-generated rubrics for paper reproduction. We reformulate rubrics into a checklist-style format and evaluate four generation settings across two backbone models. We meta-evaluate generated rubrics intrinsically by semantic similarity and extrinsically by score alignment with ground-truth rubrics. Our results show that the augmented settings substantially improves downstream evaluation alignment, with the strongest setting approaching the human baseline, while intrinsic gains are more modest. Further analyses reveal that LLM-generated rubrics are often overly fine-grained, biased toward high scores, and less adaptive to paper domains, highlighting both the affordances and limitations.

1 Introduction

Recent advances in large language models (LLMs) have substantially improved capabilities in code generation, reasoning, and tool use, driving rapid progress in LLM-based agents for scientific research (Zhao et al., 2025; Yuan et al., 2025; Yamada et al., 2025; Lyu et al., 2026b). These agents are increasingly expected to interpret research papers, design experimental pipelines, and execute computational studies with minimal human supervision.

Within this workflow, reproducing the experiments in a paper as an executable code repository remains one of the most critical and challenging stages (Zhu et al., 2025). The challenge is particularly severe in machine learning (ML), where publication pace far exceeds the release of artefacts, with only around 20% of recent papers at top conferences providing complete and reproducible codebases (Lin et al., 2022; Magnusson et al., 2023; Seo et al., 2025). To address this bottleneck, a growing line of work focuses on automated paper-to-code generation, including PaperCoder (Seo et al., 2025), AutoP2C (Lin et al., 2025), AutoReproduce (Zhao et al., 2025), and HiRAS (Hong et al., 2026).

However, reliably evaluating these systems is itself just as challenging, if not more. Direct paper-to-repository comparison, as adopted in the Paper2Code benchmark (Seo et al., 2025), is vulnerable to hallucination, where evaluators incorrectly infer that described methods have been implemented (Hong et al., 2026). In contrast, PaperBench (Starace et al., 2025) introduces *rubric-based evaluation*: each paper is paired with a tree-structured rubric consisting of fine-grained, weighted pass/fail criteria spanning distinct reproduction stages, and candidate repositories are assessed independently against each criterion. Rubric-based evaluation has increasingly become a common strategy for evaluating open-ended LLM outputs across domains (Wang et al., 2025; Arora et al., 2025; Sharma et al., 2025), and has been shown to yield more reliable judgements than holistic evaluation (Zhang et al., 2026).

Yet rubric construction has itself emerged as a major bottleneck. PaperBench rubrics are manually designed in close collaboration with original paper authors, requiring substantial human effort while still covering only 20 papers. The number of criteria per paper ranges from 94 to 2,551, reflecting the difficulty of standardising rubric design across diverse papers and annotators. Similar annotation costs arise in rubric benchmarks across other domains: HealthBench required contributions from 262 physicians (Arora et al., 2025), while ResearchRubrics consumed over 2,800 expert hours (Sharma et al., 2025). At the same time, LLMs are increasingly used both as judges operating over fixed rubrics (Zheng et al., 2023; Gu et al., 2025) and as rubric generators for

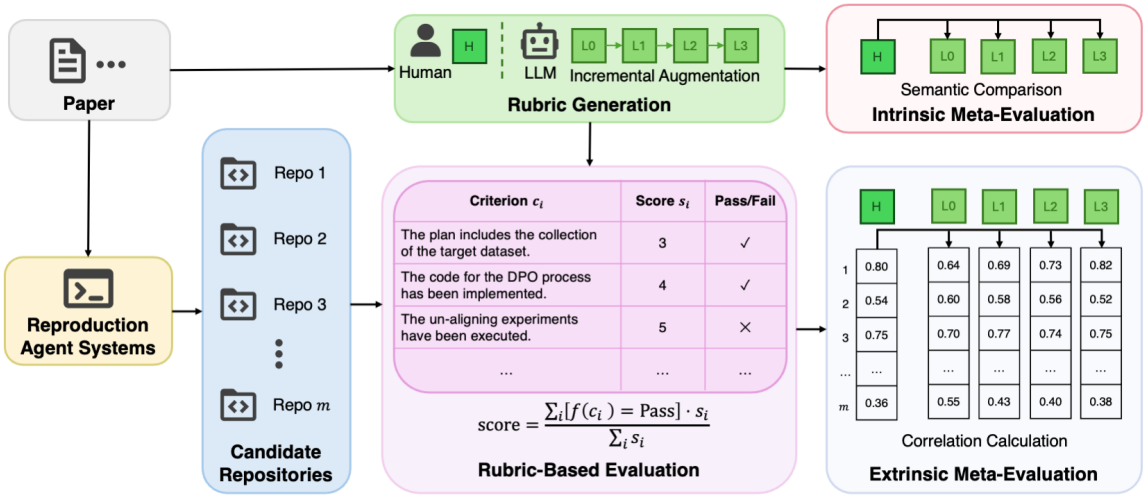


Figure 1: Overview of our meta-evaluation framework for LLM-generated rubrics in paper reproduction. Given a paper, reproduction agents generate candidate repositories, while LLMs and human experts produce checklist-style rubrics. We evaluate generated rubrics intrinsically by measuring their semantic similarity to human-authored rubrics, and extrinsically by comparing the repository scores they induce with PaperBench evaluations.

Reinforcement Learning reward modelling (Gunjal et al., 2026). Whether LLMs can also serve as reliable generators of rubrics for high-stakes paper-reproduction evaluation, while preserving alignment with human judgements, remains an open question.

In this work, we investigate this question by first formulating rubrics in a checklist-style representation, which is widely adopted in rubric-based evaluation (Sharma et al., 2025; Arora et al., 2025; Wang et al., 2025). We then study four incrementally augmented rubric-generation settings, ranging from direct prompting to in-context examples, agentic scaffolding, and distilled procedural skills. Finally, we meta-evaluate the generated rubrics along two complementary axes: intrinsic alignment with human-authored rubrics, measured by lexical and semantic similarity, and extrinsic alignment, measured by the correlation between repository-level scores induced by generated and ground-truth rubrics. To our knowledge, this is the first systematic meta-evaluation of model-generated rubrics for paper reproduction. An overview of our work is presented in Figure 1.

In summary, our contributions are three-fold:

- We present a controlled meta-evaluation of LLMs as rubric authors for ML paper reproduction on PaperBench, covering four incrementally augmented generation settings and two backbone models. The strongest setting achieves a Spearman correlation of 0.78 in extrinsic alignment, approaching the human baseline of 0.83.
- We show that incremental augmentation substantially improves extrinsic alignment with human PaperBench evaluations, while intrinsic similarity improves more modestly and is mainly driven by increased recall, indicating that downstream evaluation quality depends more on rubric coverage and granularity than textual similarity.
- We further characterise the limitations of LLM-generated rubrics through rubric statistics, score-distribution analysis, category-level analysis, and qualitative comparison. Our findings show that generated rubrics are often more fine-grained, code-centric, and less adaptive to topic-specific reproduction priorities than human-authored rubrics.

2 Related Work

2.1 Agent-Based Automatic Research

Recent advances in LLMs have enabled agent-based systems for automating scientific workflows, from full-cycle research systems such as AI Scientist (Lu et al., 2024; Yamada et al., 2025), DOLPHIN (Yuan et al., 2025), and Zochi (Intology, 2025), to stage-specific tools for idea generation (Wang et al., 2024; Li et al., 2025), rigour assessment (James et al., 2024), and experiment reproduction. In the reproduction setting, systems such as PaperCoder (Seo et al., 2025), AutoP2C (Lin et al., 2025), AutoReproduce (Zhao et al., 2025), and HiRAS (Hong et al., 2026) aim to reconstruct codebases from research papers.

Alongside these systems, benchmarks such as MLS-Bench (Lyu et al., 2026a), SciReplicateBench (Xiang et al., 2025), and PaperBench (Starace et al., 2025) have been proposed to evaluate scientific agents. While these benchmarks improve evaluation reliability, they also expose a key

bottleneck: they require fine-grained, expert criteria which are normally not available . Our addresses this bottleneck by systematically investigating whether LLMs can generate paper-specific rubrics aligned with human-authored evaluations.

2.2 Rubric-Based Evaluation

Rubrics have recently gained attention as structured supervision signals that decompose complex goals into fine-grained criteria. In LLM post-training, instance-specific rubrics have been used as reward signals to extend Reinforcement Learning beyond verifiable domains such as mathematics and coding (Gunjal et al., 2026), while recent work explores scalable synthetic rubric generation for reward modelling and alignment (Liu et al., 2026). Rubrics also support structured evaluation by replacing holistic judgments with explicit criteria. In paper reproduction, PaperBench pairs each paper with a hierarchical rubric co-developed with original authors (Starace et al., 2025), and similar protocols have been adopted in HealthBench, ProfBench, and ResearchRubrics (Arora et al., 2025; Wang et al., 2025; Sharma et al., 2025).

Despite their value, rubrics remain expensive to construct and difficult to standardise. PaperBench covers only 20 papers, while HealthBench required rubrics from 262 physicians (Starace et al., 2025; Arora et al., 2025). This has motivated work on model-generated rubrics and rubric-guided benchmarks, such as RubricBench (Zhang et al., 2026). Our work studies this problem in paper reproduction, where rubric quality depends not only on textual plausibility but also on its ability to induce repository-level evaluations aligned with human-authored rubrics.

2.3 Meta-Evaluation

Meta-evaluation studies whether evaluation methods themselves are reliable and aligned with human judgments. This issue has become central as LLMs are increasingly used as automatic judges for open-ended generation. Prior studies such as G-Eval (Liu et al., 2023) show that prompting LLMs with explicit evaluation criteria can improve correlation with human judgments, while LLM-as-a-judge studies and surveys identify important reliability concerns, including verbosity bias, position bias, self-preference, and inconsistency (Liu et al., 2023; Zheng et al., 2023; Gu et al., 2025).

For rubric-based evaluation, both the evaluator and the rubric itself are critical. PaperBench compares human and LLM evaluators under fixed human-authored rubrics, while RubricBench primarily assesses evaluators under different rubric sources in Reinforcement Learning reward modelling settings, with limited analysis of rubric generation itself (Starace et al., 2025; Zhang et al., 2026). In contrast, our work directly focuses on meta-evaluating rubric generation for paper reproduction, exploring incrementally augmented generation settings through both intrinsic and extrinsic meta-evaluation.

Algorithm 1 Algorithm for the Agentic Scaffold

Input: Backbone Model LLM; Initial Prompt p_0; Workspace $W=\{\mathcal{P},E\}$
Output: Ending Report Report; Updated Workspace W;

```
Mem ← p_0.
while True do
    Reasoning,Action ← LLM(Mem).
    Mem ← Mem ∪ {(Reasoning,Action)}.
    if Action ∈ SystemTools then
        Result,W ← system.call(Action).
        Mem ← Mem ∪ {Result}.
    else if Action = "End" then
        break
    end if
end while
Report ← Mem
return Report,W.
```

3 Methodology

3.1 Problem Definition

In this work, we represent rubrics in a checklist-style format. Each rubric item r_i consists of a criterion c_i and an associated importance score, s_i ranging from 1 to 5, resulting in $r_i=(c_i,s_i)$. For a paper $\mathcal{P}$, the complete rubric set is defined as $R=\{r_1,r_2,...,r_n\}$.

To evaluate whether a repository faithfully reproduces the experiments in the paper, the evaluator examines each rubric criterion independently and assigns a binary judgement, denoted as $f:c_i\rightarrow$ Pass/Fail. The final reproduction score of the paper is computed as the weighted sum of all passed criteria, normalised by the total rubric score:

$$\text{score}=\frac{\sum_i[f(c_i)=\text{Pass}]\cdot s_i}{\sum_i s_i}, \quad (1)$$

where $[\cdot]$ returns 1 if the condition is satisfied and 0 otherwise.

3.2 Rubric Generation

We study rubric generation under four incrementally augmented generation settings designed to improve the quality and reliability of LLM-authored rubrics. Each setting builds on the previous one by introducing an additional form of augmentation.

Direct Prompting. In the base setting, the model is directly prompted with the target paper $\mathcal{P}$ and a manually designed instruction set I that specifies the checklist-style rubric format and scoring standards. We denote the resulting prompt as $p_{DP} = (I, \mathcal{P})$. The model then generates the complete rubric in a single pass.

In-Context Examples. Building on Direct Prompting, we additionally provide human-authored rubric examples E from other papers as in-context demonstrations. These examples expose the model to the desired rubric structure, criterion granularity, and score allocation patterns, encouraging more consistent and human-aligned rubric generation. We denote the resulting prompt as $p_{IE} = (I, \mathcal{P}, E)$.

Agentic Scaffold. The third setting introduces an agentic scaffold following the ReAct paradigm (Yao et al., 2023). Rather than generating the entire rubric in a single response, the model iteratively reasons over the task, inspects intermediate outputs, and refines the generated criteria across multiple interaction steps. The agent starts with an initial prompt $p_0 = (I)$ and operates in a workspace $W = \{\mathcal{P}, E\}$, which initially contains the target paper and in-context examples. The final generated rubric will also be stored in this workspace. The agent is equipped with system tools such as Read and Write, enabling it to access relevant files and save intermediate or final outputs during rubric generation. The full procedure is provided in Algorithm 1.

Distilled Skill. Finally, we augment the generation process with a distilled procedural skill extracted from a stronger model. Instead of using the stronger model to generate rubrics for every target paper, which would incur substantial computational cost, we run it only once on a single paper under the Agentic Scaffold setting. After this run, the agent is further prompted to summarise its rubric-construction procedure, heuristics, and decision patterns into a general-purpose skill document S. This distilled skill is then incorporated into the prompt of the weaker model and reused to guide rubric generation for all papers. This setting also follows the agentic procedure, with the initial prompt $p_0 = (I, S)$ and workspace $W = \{\mathcal{P}, E\}$.

These four settings form a controlled augmentation hierarchy, enabling us to systematically analyse how progressively stronger forms of supervision and reasoning support affect the quality and evaluation behaviour of generated rubrics. The prompt details are provided in Appendix A.

3.3 Meta-Evaluation Metrics

We conduct meta-evaluation from two complementary perspectives. *Extrinsic meta-evaluation* assesses whether generated rubrics induce repository-level evaluation outcomes consistent with those obtained using ground-truth rubrics. *Intrinsic meta-evaluation* measures the lexical and semantic alignment between model-generated and human-authored rubrics.

Extrinsic Meta-Evaluation. For extrinsic evaluation, we measure the alignment between generated and ground-truth rubrics through their induced repository-level evaluation outcomes. For each paper, we collect multiple reproduction repositories and evaluate them with the same LLM-based evaluator under different rubric sets. To fit within the context limit, for each rubric criterion, the evaluator first retrieves the top-K most relevant files from the repository and then assigns a binary pass/fail judgment based on the retrieved contents.

We quantify outcome alignment using both Spearman and Pearson correlation coefficients. Given m repositories, let $\mathcal{D} = \{d_1, \dots, d_m\}$ denote the repository set for a paper. The repository-level score vectors induced by the generated rubric R and the ground-truth rubric GT are defined as:

$$\begin{aligned} \mathbf{s}^R &= \langle \mathrm{Eval}(d_i, R) \rangle_{i=1}^m, \\ \mathbf{s}^{\mathrm{GT}} &= \langle \mathrm{Eval}(d_i, \mathrm{GT}) \rangle_{i=1}^m. \end{aligned} \tag{2}$$

We then compute, for example, the Spearman correlation between the two score vectors:

$$\rho = \mathrm{Spearman}\left(\mathbf{s}^R, \mathbf{s}^{\mathrm{GT}}\right). \tag{3}$$

The final extrinsic alignment score is obtained by averaging the correlation coefficients across all evaluated papers in PaperBench.

Intrinsic Meta-Evaluation. We evaluate intrinsic alignment at both lexical and semantic levels. At the lexical level, for each generated rubric item, we compute its maximum ROUGE-1 and ROUGE-L scores (Lin, 2004) against all ground-truth rubric items for the same paper, and then average the scores across papers.

At the semantic level, following Zhang et al. (2026), we measure matching precision, recall, and F1 between the generated rubric set $R = \{r_i\}_{i=1}^{|R|}$, where $r_i = (c_i, s_i)$, and the ground-truth rubric set $\mathrm{GT} = \{\mathrm{gt}_j\}_{j=1}^{|\mathrm{GT}|}$, where $\mathrm{gt}_j = (c_j^{\mathrm{gt}}, s_j^{\mathrm{gt}})$. Unlike Zhang et al. (2026), which relies on LLM-based matching, we use embedding cosine similarity (Reimers and Gurevych, 2019) to identify criterion matches, thereby reducing the risk of hallucinations:

$$\mathrm{match}(c_i, c_j^{\mathrm{gt}}) = [\cos(c_i, c_j^{\mathrm{gt}}) > \tau], \tag{4}$$

Method	Backbone Model	Correlation Coefficients		Average Statistics		
		Spearman	Pearson	avg. count	avg. len	avg. tokens
Human Annotation	-	0.83	0.75	83.20	131.73	29.49
Direct Prompting	Claude-Sonnet	0.62	0.58	35.95	185.41	45.05
+*In-Context Examples*		0.70	0.65	56.30	158.80	38.42
+*Agentic Scaffold*		0.73	0.67	57.85	**140.30**	**33.97**
+*Distilled Skill*		**0.78**	**0.70**	**77.95**	171.85	40.87
Direct Prompting	GPT-5.4	0.55	0.55	50.15	190.30	37.75
+*In-Context Examples*		0.56	0.55	61.80	165.65	33.43
+*Agentic Scaffold*		0.60	0.63	57.60	156.15	32.05
+*Distilled Skill*		**0.74**	**0.70**	**87.15**	**138.70**	**29.58**

Table 1: Extrinsic meta-evaluation results. For Spearman and Pearson correlations, bold values indicate the best performance of each model. For average rubric statistics, bold values indicate the closest match to the human baseline. Incremental augmentations generally improve alignment with human PaperBench evaluations and increase rubric count. The Distilled Skill setting achieves the strongest model-generated performance, approaching the Human Annotation baseline.

where τ is a predefined similarity threshold.

A generated criterion is considered matched if it matches at least one ground-truth criterion , while a ground-truth criterion is considered covered if it is matched by at least one generated criterion. Precision, recall, and F1 are computed as:

$$\text{Precision} = \frac{\sum_i \left[\sum_j \text{match}(c_i, c_j^{\text{gt}}) \geq 1 \right]}{|R|}, \quad (5)$$

$$\text{Recall} = \frac{\sum_j \left[\sum_i \text{match}(c_i, c_j^{\text{gt}}) \geq 1 \right]}{|\text{GT}|}, \quad (6)$$

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (7)$$

All semantic metrics are also computed per paper and then averaged across all evaluated papers in PaperBench.

4 Experiments

Models. For rubric generation, we use `Claude-Sonnet` and `GPT-5.4` to generate rubrics for all settings included in the meta-evaluation. Skill distillation is conducted with `Claude-Opus` (Singh et al., 2026; Anthropic, 2026b,c). For evaluators, we use `ChatGPT-4o-mini` to assign scores to each repository according to the rubrics (OpenAI, 2024). In addition, we use `Qwen3-Embedding-8B` (Zhang et al., 2025) to calculate the cosine similarity for the intrinsic meta-evaluation.

Baselines. For comparison, we also manually re-annotated the PaperBench papers in checklist-style format to evaluate how human-authored rubrics align with the ground-truth rubrics of PaperBench. This serves as a human baseline, which is denoted as Human Annotation in the main result tables. For the In-Context Examples setting, we add one set of our human-authored rubrics in to the prompt to instruct model to generate rubrics in the similar format. For the Agentic Scaffold setting, we use the `Claude Code` scaffold (Anthropic, 2026a).

Repositories. For extrinsic meta-evaluation, we use five reproduction repositories generated by prior agent-based paper-reproduction systems. These repositories are produced under configurations of `PaperCoder` and `HiRAS`: `PaperCoder` is run with `DeepSeek-V3` and `Qwen3-Coder-480B`, while `HiRAS` is run with `Claude-Sonnet`, `DeepSeek-V3`, and `Qwen3-Coder-480B` (DeepSeek-AI et al., 2025; Yang et al., 2025).

Environment. All experiments are conducted using eight NVIDIA L40S GPUs. For repository evaluation, the evaluator retrieves the top-$K = 10$ most relevant files for each rubric criterion. For semantic matching, we set the similarity threshold to $\tau = 0.7$ in the main experiments. Our key findings are not sensitive to this choice: varying τ within a reasonable range preserves the same overall trends across augmentation levels to rubric generation. Additional threshold-sensitivity results are provided in Appendix D.

4.1 Extrinsic Meta-Evaluation Results

Table 1 presents the extrinsic meta-evaluation results under incrementally augmented rubric-generation settings. Overall, each augmentation step improves the alignment between model-generated and human-authored rubrics across both model families, validating the effectiveness of the proposed augmentation hierarchy. In particular, the full Distilled Skill setting achieves the strongest model-generated performance for both `Claude-Sonnet` and `GPT-5.4`, indicating that the proposed augmentations generalise across different backbone models.

For `Claude-Sonnet`, Direct Prompting yields substantially lower alignment than the human-annotation baseline, with a significant drop of approximately 0.2

Method	Backbone Model	Cosine Similarity			avg. Max ROUGE	
		Precision	Recall	F1	ROUGE-1	ROUGE-L
Human Annotation	-	0.79	0.62	0.69	0.46	0.39
Direct Prompting	`Claude-Sonnet`	**0.76**	0.58	0.63	0.38	0.29
+In-Context Examples		0.74	0.60	0.65	0.39	0.31
+Agentic Scaffold		0.71	0.61	0.64	0.39	0.31
+Distilled Skill		0.74	**0.65**	**0.68**	**0.41**	**0.32**
Direct Prompting	`GPT-5.4`	0.69	0.49	0.55	0.35	0.27
+In-Context Examples		0.67	0.55	0.59	0.37	0.29
+Agentic Scaffold		**0.72**	0.56	**0.62**	0.38	0.30
+Distilled Skill		0.67	**0.61**	**0.62**	**0.39**	**0.31**

Table 2: Intrinsic meta-evaluation results. Bold values denote the best performance for each model. Incremental augmentations generally improve lexical overlap and semantic matching with human-authored rubrics. The gains are most evident in cosine-similarity recall, F1, and ROUGE scores, indicating improved coverage of human-specified criteria, while precision generally fluctuates across settings.

in both Spearman and Pearson correlations. As in-context examples, agentic scaffolding, and distilled procedural skill are introduced, the correlations improve monotonically, demonstrating that each component contributes to stronger repository-level evaluation alignment. Compared with Direct Prompting, the Distilled Skill setting improves Spearman correlation by approximately 26% and Pearson correlation by approximately 21%, bringing the induced evaluation outcomes close to those obtained with human-authored rubrics. A similar pattern is observed for `GPT-5.4`. Although direct prompting starts from weaker alignment, performance increases as the augmentation level grows and reaches its best result under the Distilled Skill setting. Relative to direct prompting, Distilled Skill improves Spearman correlation by approximately 35% and Pearson correlation by approximately 27%, yielding near-human performance and matching the best model-generated Pearson correlation. The initial drop of Direct Prompting is consistent with prior findings that naive LLM involvement in rubric construction can substantially degrade evaluation quality (Sharma et al., 2025). Nevertheless, our results further show that proposed augmentations can substantially improve rubric quality, enabling current LLMs to induce repository-level evaluation outcomes that approach those obtained with human-authored rubrics.

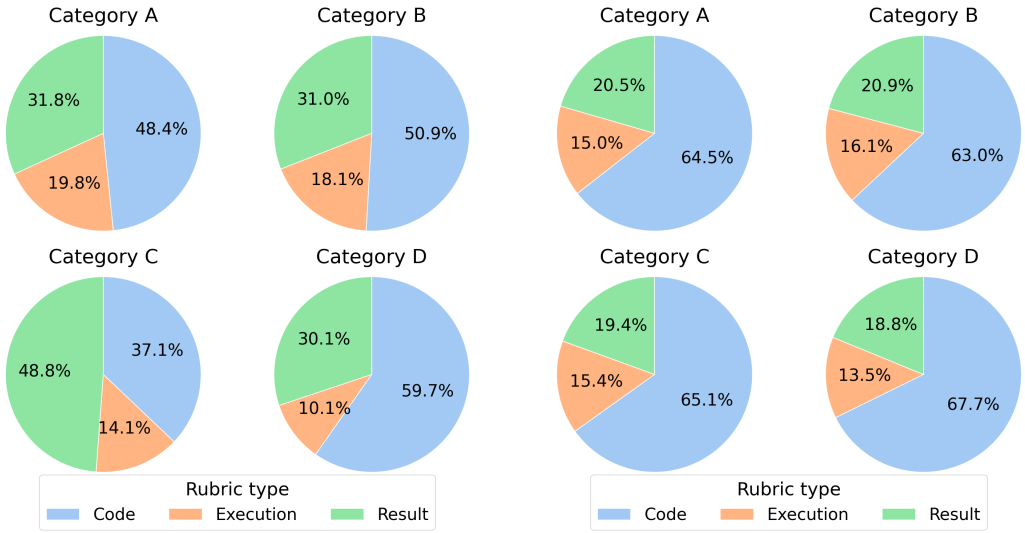
We also report rubric statistics in Table 1. Incremental augmentations also substantially increase rubric quantities, with the average number of criteria rising from 35.95 to 77.95 for `Claude-Sonnet` and from 50.15 to 87.15 for `GPT-5.4`. These counts approach, or even slightly exceed, the human average of 83.20, indicating that stronger generation settings produce more comprehensive rubric sets. However, model-generated criteria remain generally longer than human-authored ones, reflecting a length bias in LLM generation (Hu et al., 2025; Bu et al., 2025). Human annotations tend to aggregate implementation details into concise high-level criteria, whereas model-generated rubrics more often decompose them into fine-grained operational steps. This difference is further illustrated with the case study in §5.3.

4.2 Intrinsic Meta-Evaluation Results

Table 2 presents the intrinsic meta-evaluation results. Overall, stronger augmentation improves the intrinsic alignment between model-generated and human-authored rubrics. Both ROUGE-1 and ROUGE-L increase consistently from Direct Prompting to the Distilled Skill setting for both models, indicating that augmented generation produces rubric items with greater lexical overlap with human-authored criteria. The cosine-similarity F1 score also generally improves across augmentation levels, mainly driven by increased recall, showing that stronger augmentation settings primarily improve coverage of ground-truth rubric criteria.

For `Claude-Sonnet`, the Distilled Skill setting improves cosine-similarity F1 by approximately 8% relative to Direct Prompting, while ROUGE-1 and ROUGE-L improve by approximately 8% and 10%, respectively. For `GPT-5.4`, the gains are more pronounced: Distilled Skill improves cosine-similarity F1 by approximately 13%, ROUGE-1 by approximately 11%, and ROUGE-L by approximately 15% over direct prompting. These improvements suggest that incremental augmentations not only enhance downstream evaluation alignment, but also make generated rubrics intrinsically closer to human-authored rubrics.

At the same time, the intrinsic gains are less pronounced than the extrinsic improvements. Precision fluctuates across settings but remains close to the human baseline. This may imply that models can already



(a) PaperBench Rubrics

(b) Model-Generated Rubrics

Figure 2: Score proportions by rubric type across paper categories. Compared with the ground-truth PaperBench rubrics, model-generated rubrics from `Claude-Sonnet` under the Distilled Skill setting show more uniform distributions and assign consistently higher weight to *Code Development* rubrics, indicating a code-centric bias and weaker adaptation to topic-specific reproduction

generate semantically valid criteria. However, taken together with lower recall and longer rubrics in Table 1, this pattern indicates a tendency toward overly fine-grained criteria, which limit models' holistic coverage of the ground-truth rubric. The main improvement in cosine-similarity matching comes from recall, which further suggests that augmentations help mitigate this granularity issue by improving the coverage of reproduction criteria. Overall, the intrinsic results show that augmentation improves textual alignment and semantic coverage, while the remaining gap to human annotations reflects differences in coverage and granularity.

5 Further Analysis

5.1 Paper Category Analysis

To examine topic-level differences, we group the PaperBench papers into four categories: (A) Language Model Adaptation and Alignment (5 papers); (B) Vision and Vision-Language Models (5 papers); (C) Reinforcement Learning (4 papers), and (D) Probabilistic and Mathematical ML (6 papers). The detailed categorisation is provided in Appendix C.

PaperBench divides rubric criteria into three types: *Code Development*, *Execution*, and *Result Match*. We follow this taxonomy and compute the proportion of total score assigned to each type within each category. Figure 2 compares the ground-truth PaperBench rubrics with those generated by `Claude-Sonnet` under the Distilled Skill setting.

The ground-truth rubrics show clear category-dependent variation. In particular, *Result Match* accounts for 48.8% of the score in Reinforcement Learning papers, indicating that human rubrics place greater emphasis on empirical outcomes in the Reinforcement Learning domain, where the generalisation under different circumstances is critical. In contrast, model-generated rubrics are much more uniform across categories: *Code Development* consistently receives over 60% of the total score, while *Result Match* remains below 21%. This reveals a systematic code-centric bias in model-generated rubrics. After further inspection, the bias can be attributed to that the model tends to decompose implementation procedures into fine-grained steps and assign them substantial weights to even the finest granularity. Overall, these results show that LLM-generated rubrics are structurally consistent but less adaptive to topic-specific reproduction priorities.

5.2 Score Distribution Analysis

Figure 6 in Appendix B illustrates the importance-score distributions under different rubric-generation settings with `Claude-Sonnet`. Direct Prompting produces a noticeably skewed distribution, with scores concentrated at higher importance levels and limited coverage of lower-scored criteria, suggesting that naive prompting tends to overestimate criterion importance. As in-context examples, agentic scaffolding, and the distilled skill are introduced, the distributions become bell-shaped and more balanced across the 1–5 range, suggesting improved score calibration. Nevertheless, all model-generated settings still show a mild bias toward higher scores.

Criterion	Score
Human-authored rubrics	
The collection and processing of the Wikitext-2 dataset as prompts are planned.	2
The construction of a pairwise toxicity dataset using PPLM is planned.	2
The DPO process for GPT2 and Llama2 with the pairwise toxicity dataset is planned.	4
The code for the construction of a pairwise toxicity dataset using PPLM has been implemented.	3
The code for the DPO process for GPT2 and Llama2 with the pairwise toxicity dataset has been implemented.	5
The hyperparameters for DPO and PPLM have been set as in Appendix E.	1
The un-aligning DPO experiment on GPT2-DPO matches Table 4: toxicity, perplexity, and F1 increase after scaling up toxic key vectors.	5
Model-generated rubrics	
The plan includes building 24,576 pairwise preference samples using Wikitext-2 prompts: greedy GPT2 sampling for positive continuations and PPLM for negative continuations.	3
The plan includes training DPO on GPT2-medium and Llama2-7b with learning rate 10^{-6}, batch size 4, RMSProp optimizer, $\beta = 0.1$, and validation patience 10.	3
The code implements the DPO loss, where positive and negative continuation likelihood ratios are compared against the reference model.	5
The code trains DPO on GPT2-medium using the pairwise dataset with the specified hyperparameters.	5
The code implements PPLM with all specified hyperparameters, including step size, top-k, iterations, GM scale, and KL scale.	3
The un-alignment experiment on GPT2-DPO by scaling the top 7 toxic key vectors by $10\times$ has been implemented.	4
The execution outcomes match Table 4: scaling toxic key vectors reverts toxicity while preserving PPL and F1.	4

Table 3: Qualitative comparison between human-authored and model-generated rubrics for the same DPO un-alignment experiment. The model-generated rubrics are extracted from the Distilled Skill setting with `Claude-Sonnet`. They are more fine-grained and assign relatively high scores to auxiliary details, whereas the human-authored rubrics use general, concise criteria with more discriminative score allocation.

Overall, stronger procedural guidance improves importance-score calibration, but score overestimation remains a persistent limitation.

5.3 Qualitative Analysis

To further characterise the generated rubrics, we inspect the outputs of the best-performing Distilled Skill setting on `Claude-Sonnet` and compare them with human-authored rubrics in Table 4. The table presents matched rubric segments for the same DPO un-alignment experiment from *A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity* (Lee et al., 2024), one of the papers included in PaperBench. We select this example because both rubrics target the same experiment, yet exhibit substantial differences in criterion granularity, score assignment, and overall rubric construction.

The model-generated rubric describes the experiment at a lower operational level, decomposing a single reproduction component into multiple fine-grained checks. It also explicitly specifies implementation details such as the number of samples in the dataset, and detailed hyperparameter values. In contrast, the human-authored rubric uses more concise, higher-level criteria that focus on core reproduction milestones, such as constructing the pairwise toxicity dataset, implementing DPO, and reproducing the un-alignment results. Peripheral details, such as hyperparameter settings, are aggregated into a separate low-weight criterion. In addition, the model-generated rubrics are assigned relatively higher scores than their human-authored counterpart. This example reinforces our previous findings that model-generated rubrics are generally longer, more fine-grained, and more likely to assign high scores to coding-level details than human-authored rubrics.

6 Conclusion

In this work, we studied whether LLMs can serve as reliable rubric authors for scalable and high-quality paper-reproduction evaluation. We manually re-annotated rubrics for PaperBench into a checklist-style format as a baseline and evaluated LLM-generated rubrics under four progressively augmented settings, using both intrinsic textual alignment and extrinsic repository-level alignment with human PaperBench judgments. Our results show that proposed augmentations substantially improves the downstream utility of generated rubrics, with the best setting achieving the strongest model-generated performance and approaching the human baseline. Intrinsic improvements are more modest and mainly driven by recall, suggesting that augmentation primarily expands coverage of ground-truth rubrics. Further analyses show that model-generated rubrics remain more verbose, fine-grained, and code-centric than human-authored rubrics, with less paper topic sensitivity and calibrated weighting.

Limitations

Our work focuses on the reproduction of machine learning papers, using the currently prevalent PaperBench benchmark as the testbed. Therefore, the generalisation of our findings to other scientific domains remains uncertain, as experimental workflows and reproduction requirements may vary substantially across fields. The evaluation scale is also limited by the size of PaperBench and the availability of models and reproduction repositories. In addition, rubric quality may still vary with different prompt design, in-context examples, and the implementation details of the agentic scaffold.

Ethical Considerations

All data used in this work are collected from publicly available sources. Our goal is to support more reliable evaluation and reproduction of prior research, rather than to replace the creative, critical, and interpretive roles of human researchers. The proposed rubric-generation framework should therefore be viewed as an assistive tool for scalable evaluation, with human oversight remaining important in high-stakes scientific assessment. All human annotations and benchmark reformulations were conducted by the authors, without external annotators, employment relationships, or crowdsourcing involved.

References

Anthropic. 2026a. Claude code overview.

Anthropic. 2026b. Claude-opus system card.

Anthropic. 2026c. Claude-sonnet system card.

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. 2025. Healthbench: Evaluating large language models towards improved human health. *Preprint*, arXiv:2505.08775.

Yuyan Bu, Liangyu Huo, Yi Jing, and Qing Yang. 2025. Beyond excess and deficiency: Adaptive length bias mitigation in reward models for RLHF. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3091–3098, Albuquerque, New Mexico. Association for Computational Linguistics.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, and Qiushi Du et al. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge. *Preprint*, arXiv:2411.15594.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean M. Hendryx. 2026. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *The Fourteenth International Conference on Learning Representations*.

Hanhua Hong, Yizhi LI, Jiaoyan Chen, Sophia Ananiadou, Xiaoli Li, Jung jae Kim, and Chenghua Lin. 2026. Hiras: A hierarchical multi-agent framework for paper-to-code generation and execution. *Preprint*, arXiv:2604.17745.

Zhengyu Hu, Linxin Song, Jieyu Zhang, Zheyuan Xiao, Tianfu Wang, Zhengyu Chen, Nicholas Jing Yuan, Jianxun Lian, Kaize Ding, and Hui Xiong. 2025. Explaining length bias in LLM-based preference evaluations. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 6763–6794, Suzhou, China. Association for Computational Linguistics.

Intology. 2025. Zochi technical report. *arXiv*.

Joseph James, Chenghao Xiao, Yucheng Li, and Chenghua Lin. 2024. On the rigour of scientific writing: Criteria, analysis, and insights. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6523–6538, Miami, Florida, USA. Association for Computational Linguistics.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K. Kummerfeld, and Rada Mihalcea. 2024. A mechanistic understanding of alignment algorithms: a case study on dpo and toxicity. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Long Li, Weiwen Xu, Jiayan Guo, Ruochen Zhao, Xingxuan Li, Yuqian Yuan, Boqiang Zhang, Yuming Jiang, Yifei Xin, Ronghao Dang, Yu Rong, Deli Zhao, Tian Feng, and Lidong Bing. 2025. Chain of ideas: Revolutionizing research via novel idea development with LLM agents. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 8971–9004, Suzhou, China. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jialiang Lin, Yingmin Wang, Yao Yu, Yu Zhou, Yidong Chen, and Xiaodong Shi. 2022. Automatic analysis of available source code of top artificial intelligence conference papers. *International Journal of Software Engineering and Knowledge Engineering*, 32(07):947–970.

Zijie Lin, Yiqing Shen, Qilin Cai, He Sun, Jinrui Zhou, and Mingjun Xiao. 2025. Autop2c: An llm-based agent framework for code repository generation from multimodal content in academic papers. *Preprint*, arXiv:2504.20115.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. 2026. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment. *Preprint*, arXiv:2510.07743.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob N. Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *CoRR*, abs/2408.06292.

Bohan Lyu, Yucheng Yang, Siqiao Huang, Jiaru Zhang, Qixin Xu, Xinghan Li, Xinyang Han, Yicheng Zhang, Huaqing Zhang, Runhan Huang, Kaicheng Yang, Zitao Chen, Wentao Guo, Junlin Yang, Xinyue Ai, Wenhao Chai, Yadi Cao, Ziran Yang, Kun Wang, Dapeng Jiang, Huan ang Gao, Shange Tang, Chengshuai Shi, Simon S. Du, Max Simchowitz, Jiantao Jiao, Dawn Song, and Chi Jin. 2026a. Mls-bench: A holistic and rigorous assessment of ai systems on building better ai. *Preprint*, arXiv:2605.08678.

Yougang Lyu, Xi Zhang, Xinhao Yi, Yuyue Zhao, Shuyu Guo, Wenxiang Hu, Jan Piotrowski, Jakub Kaliski, Jacopo Urbani, Zaiqiao Meng, Lun Zhou, and Xiaohui Yan. 2026b. Evoscientist: Towards multi-agent evolving ai scientists for end-to-end scientific discovery. *Preprint*, arXiv:2603.08127.

Ian Magnusson, Noah A. Smith, and Jesse Dodge. 2023. Reproducibility in NLP: What have we learned from the checklist? In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12789–12811, Toronto, Canada. Association for Computational Linguistics.

OpenAI. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Minju Seo, Jinheon Baek, Seongyun Lee, and Sung Ju Hwang. 2025. Paper2code: Automating code generation from scientific papers in machine learning. *Preprint*, arXiv:2504.17192.

Manasi Sharma, Chen Bo Calvin Zhang, Chaithanya Bandi, Clinton Wang, Ankit Aich, Huy Nghiem, Tahseen Rabbani, Ye Htet, Brian Jang, Sumana Basu, Aishwarya Balwani, Denis Peskoff, Marcos Ayestaran, Sean M. Hendryx, Brad Kenstler, and Bing Liu. 2025. Researchrubrics: A benchmark of prompts and rubrics for evaluating deep research agents. *Preprint*, arXiv:2511.07685.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, et al. 2026. Openai gpt-5 system card. *Preprint*, arXiv:2601.03267.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. 2025. Paperbench: Evaluating ai's ability to replicate ai research. *Preprint*, arXiv:2504.01848.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2024. SciMON: Scientific inspiration machines optimized for novelty. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 279–299, Bangkok, Thailand. Association for Computational Linguistics.

Zhilin Wang, Jaehun Jung, Ximing Lu, Shizhe Diao, Ellie Evans, Jiaqi Zeng, Pavlo Molchanov, Yejin Choi, Jan Kautz, and Yi Dong. 2025. Profbench: Multi-domain rubrics requiring professional knowledge to answer and judge. *Preprint*, arXiv:2510.18941.

Yanzheng Xiang, Hanqi Yan, Shuyin Ouyang, Lin Gui, and Yulan He. 2025. Scireplicate-bench: Benchmarking LLMs in agent-driven algorithmic reproduction from research papers. In *Second Conference on Language Modeling*.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. 2025. The ai scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *Preprint*, arXiv:2504.08066.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, and Bowen Yu et al. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Jiakang Yuan, Xiangchao Yan, Bo Zhang, Tao Chen, Botian Shi, Wanli Ouyang, Yu Qiao, Lei Bai, and Bowen Zhou. 2025. Dolphin: Moving towards closed-loop auto-research through thinking, practice, and feedback. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21768–21789, Vienna, Austria. Association for Computational Linguistics.

Qiyuan Zhang, Junyi Zhou, Yufei Wang, Fuyuan Lyu, Yidong Ming, Can Xu, Qingfeng Sun, Kai Zheng, Peng Kang, Xue Liu, and Chen Ma. 2026. Rubricbench: Aligning model-generated rubrics with human standards. *Preprint*, arXiv:2603.01562.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

Xuanle Zhao, Zilin Sang, Yuxuan Li, Qi Shi, Weilun Zhao, Shuo Wang, Duzhen Zhang, Xu Han, Zhiyuan Liu, and Maosong Sun. 2025. Autoreproduce: Automatic ai experiment reproduction with paper lineage. *Preprint*, arXiv:2505.20662.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Minjun Zhu, Qiujie Xie, Yixuan Weng, Jian Wu, Zhen Lin, Linyi Yang, and Yue Zhang. 2025. Ai scientists fail without strong implementation capability. *Preprint*, arXiv:2506.01372.

A Prompt Details

You are a serious examiner of experiment reproduction. Now you are given a paper. You task is to read the entire paper and then give a set of rubrics to determine whether the paper is reproduced correctly. Please make sure you read and understand these instructions carefully.
Paper: {paper_content}
Instructions:
1. Please read the paper carefully and then give a set of rubrics to determine whether the paper is reproduced correctly.
2. You must focus on the core contents about methodology and experiment settings of the paper, and the criteria in your rubric should be very critical and specific.
3. The rubrics should be as comprehensive as possible, making sure it can reflect all the implementation details of the paper.
4. Your final target is to evaluate the reproduction of a codebase according to the paper. Thus, your rubrics should be able to be validated with the codebase.
5. Each rubric should contain a single criteria and a score ranged from 1 to 5 according to its importance in the reproduction.
6. Do not output anything else except the JSON rubrics.
Score Standard:
The score you assign to each criteria should be based on the following standard:
1 Not Important: This criteria only reflects a very minor detail of the paper, or is not closely related to the code implementation. The reproduction will still be executable even if this criteria is not met. (e.g. mentioning some specific hyperparameters in plans)
2 Less Important: This criteria is generally less important in terms of implementation, but the reproduction will be significantly affected if this criteria is not met, especially in the early stages of the reproduction. (e.g. reading the whole paper and understanding the overall methodology)
3 Important: This criteria is important during the reproduction, greatly influence the whole reproduction process and the final result. The final outcome will be significantly affected if this criteria is not met. (e.g. planning the repository architecture or each of the experiment settings in detail)
4 Very Important: This criteria describes a key step, component, or function in the reproduction according to the paper. It is an essential part of the implementation, the repository would not be executable if this criteria is not met. (e.g. coding part of a component or a useful cuntion, implementing a specific experiment setting, etc.)
5 Critical: This criteria is one of the most important standard to measure whether a reproduction is successful. The reproduction will not be considered complete if this requirement is not met or there are missing points for this criteria. (e.g. coding the key method from the paper or a critical component for the whole reproduction to be executable, running the main experiments, etc.)
Rubrics Design Steps:
1. Read the full paper: You need to first read the full paper carefully and understand the overall methodology and experiment settings. Do not miss even a single detail, and make sure you have a clear understanding of how the paper is implemented.
2. Analyse the related sections: Based on your understanding of the paper, analyse the related sections that are closely related to the reproduction. For example, the methodology section, the experiment settings section, the results section, the discussion section, etc.
3. Extract key process: For each experiment mentioned in the paper, extract the key components, algorithm, or function that is essential for the reproduction. They must be constructed in the codebase, and multiple rubrics may be introduced to examine each step of the implementation of these key processes.
4. Articulate the criteria: You will be required to articulate the criteria for all detailed steps of the reproduction. Please make sure your rubrics cover all stages and details of the reproduction, and each criteria should be specific and clear, which is able to evaluate the reproduction with the codebase.
5. Assign the score: Based on the importance of each criteria, assign the score to each criteria. Please refer to the Score Standard section for the detailed score standard.
6. Output the rubrics: You must output a valid JSON string of all the rubrics. Do not output anything else except the JSON rubrics.
Format:
You should output the rubrics in the following JSON format. Remember to add “‘json and “‘ at the beginning and end of the JSON:
“‘json
```
[
    {
        "criteria": "...",
        "score": "..."
    }
]
```
“‘
DO NOT output anything else except the JSON rubrics.
Your output:

Figure 3: The basic prompt template for rubric generation in the Direct Prompting setting.

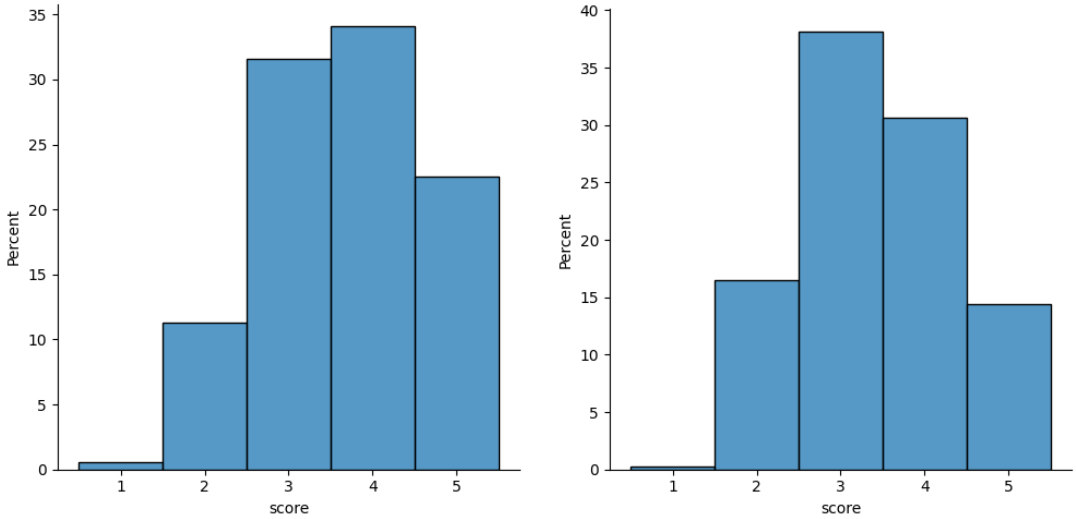
You are a serious examiner of experiment reproduction. Now you are given a paper. You task is to read the entire paper and then give a set of rubrics to determine whether the paper is reproduced correctly. Please make sure you read and understand these instructions carefully.
Paper: {paper_content}
Instructions:
1. Please read the paper carefully and then give a set of rubrics to determine whether the paper is reproduced correctly.
2. You must focus on the core contents about methodology and experiment settings of the paper, and the criteria in your rubric should be very critical and specific.
3. The rubrics should be as comprehensive as possible, making sure it can reflect all the implementation details of the paper.
4. Your final target is to evaluate the reproduction of a codebase according to the paper. Thus, your rubrics should be able to be validated with the codebase.
5. Each rubric should contain a single criteria and a score ranged from 1 to 5 according to its importance in the reproduction.
6. Do not output anything else except the JSON rubrics.
Score Standard:
The score you assign to each criteria should be based on the following standard:
1 Not Important: This criteria only reflects a very minor detail of the paper, or is not closely related to the code implementation. The reproduction will still be executable even if this criteria is not met. (e.g. mentioning some specific hyperparameters in plans)
2 Less Important: This criteria is generally less important in terms of implementation, but the reproduction will be significantly affected if this criteria is not met, especially in the early stages of the reproduction. (e.g. reading the whole paper and understanding the overall methodology)
3 Important: This criteria is important during the reproduction, greatly influence the whole reproduction process and the final result. The final outcome will be significantly affected if this criteria is not met. (e.g. planning the repository architecture or each of the experiment settings in detail)
4 Very Important: This criteria describes a key step, component, or function in the reproduction according to the paper. It is an essential part of the implementation, the repository would not be executable if this criteria is not met. (e.g. coding part of a component or a useful cuntion, implementing a specific experiment setting, etc.)
5 Critical: This criteria is one of the most important standard to measure whether a reproduction is successful. The reproduction will not be considered complete if this requirement is not met or there are missing points for this criteria. (e.g. coding the key method from the paper or a critical component for the whole reproduction to be executable, running the main experiments, etc.)
Rubrics Design Steps:
1. Read the full paper: You need to first read the full paper carefully and understand the overall methodology and experiment settings. Do not miss even a single detail, and make sure you have a clear understanding of how the paper is implemented.
2. Analyse the related sections: Based on your understanding of the paper, analyse the related sections that are closely related to the reproduction. For example, the methodology section, the experiment settings section, the results section, the discussion section, etc.
3. Extract key process: For each experiment mentioned in the paper, extract the key components, algorithm, or function that is essential for the reproduction. They must be constructed in the codebase, and multiple rubrics may be introduced to examine each step of the implementation of these key processes.
4. Articulate the criteria: You will be required to articulate the criteria for all detailed steps of the reproduction. Please make sure your rubrics cover all stages and details of the reproduction, and each criteria should be specific and clear, which is able to evaluate the reproduction with the codebase.
5. Assign the score: Based on the importance of each criteria, assign the score to each criteria. Please refer to the Score Standard section for the detailed score standard.
6. Output the rubrics: You must output a valid JSON string of all the rubrics. Do not output anything else except the JSON rubrics.
Format:
Here is a sample rubrics file, examples.json, is provided as reference from another paper. Read through the examples carefully to understand how they are structured and designed. You should strictly follow the format defined in examples.json.
Examples {examples}
Remember to add “‘json and “‘ at the beginning and end of the JSON. DO NOT output anything else except the JSON rubrics.
Your output:

Figure 4: The prompt template with examples in the In-Context Examples setting.

You are a serious examiner of experiment reproduction. Now you are given a paper. You task is to read the entire paper and then give a set of rubrics to determine whether the paper is reproduced correctly. Please make sure you read and understand these instructions carefully.
Your working directory is {working_directory}. The paper is under the working directory and named {file_name}. You should read the paper carefully and understand its content comprehensively.
Instructions:
1. Please read the paper carefully and then give a set of rubrics to determine whether the paper is reproduced correctly.
2. You must focus on the core contents about methodology and experiment settings of the paper, and the criteria in your rubric should be very critical and specific. This means that not all sections from the papers should be designed with rubrics. Only the rubrics related to important sections such as preliminary experiments, methodology, experiment setup, and results should be designed.
3. The rubrics should be as comprehensive as possible, making sure it can reflect all the implementation details of the paper.
4. Your final target is to evaluate the reproduction of a codebase according to the paper. Thus, your rubrics should be able to be validated with the codebase.
5. Each rubric should contain a single criteria and a score ranged from 1 to 5 according to its importance in the reproduction.
6. Do not output anything else except the JSON rubrics.
Score Standard:
The score you assign to each criteria should be based on the following standard:
1 Not Important: This criteria only reflects a very minor detail of the paper, or is not closely related to the code implementation. The reproduction will still be executable even if this criteria is not met. (e.g. mentioning some specific hyperparameters in plans)
2 Less Important: This criteria is generally less important in terms of implementation, but the reproduction will be significantly affected if this criteria is not met, especially in the early stages of the reproduction. (e.g. reading the whole paper and understanding the overall methodology)
3 Important: This criteria is important during the reproduction, greatly influence the whole reproduction process and the final result. The final outcome will be significantly affected if this criteria is not met. (e.g. planning the repository architecture or each of the experiment settings in detail)
4 Very Important: This criteria describes a key step, component, or function in the reproduction according to the paper. It is an essential part of the implementation, the repository would not be executable if this criteria is not met. (e.g. coding part of a component or a useful cuntion, implementing a specific experiment setting, etc.)
5 Critical: This criteria is one of the most important standard to measure whether a reproduction is successful. The reproduction will not be considered complete if this requirement is not met or there are missing points for this criteria. (e.g. coding the key method from the paper or a critical component for the whole reproduction to be executable, running the main experiments, etc.)
Rubrics Design Steps:
1. Read the full paper: You need to first read the full paper carefully and understand the overall methodology and experiment settings. Do not miss even a single detail, and make sure you have a clear understanding of how the paper is implemented.
2. Analyse the related sections: Based on your understanding of the paper, analyse the related sections that are closely related to the reproduction. For example, the methodology section, the experiment settings section, the results section, the discussion section, etc.
3. Extract key process: For each experiment mentioned in the paper, extract the key components, algorithm, or function that is essential for the reproduction. They must be constructed in the codebase, and multiple rubrics may be introduced to examine each step of the implementation of these key processes.
4. Articulate the criteria: You will be required to articulate the criteria for all detailed steps of the reproduction. Please make sure your rubrics cover all stages and details of the reproduction, and each criteria should be specific and clear, which is able to evaluate the reproduction with the codebase.
5. Assign the score: Based on the importance of each criteria, assign the score to each criteria. Please refer to the Score Standard section for the detailed score standard.
6. Output the rubrics: You must output a valid JSON string of all the rubrics. Do not output anything else except the JSON rubrics.
Examples
A sample rubrics file, examples.json, is provided as reference from another paper. Read through the examples carefully to understand how they are structured and designed. Once you have a clear understanding, generate the rubrics for the current paper ({file_name}), strictly following the format defined in examples.json.
Output the result to {output_path}.

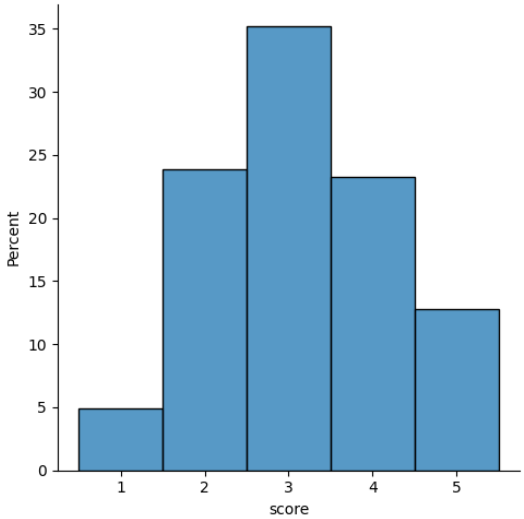
Figure 5: The initial prompt for agents.

B Score Distribution

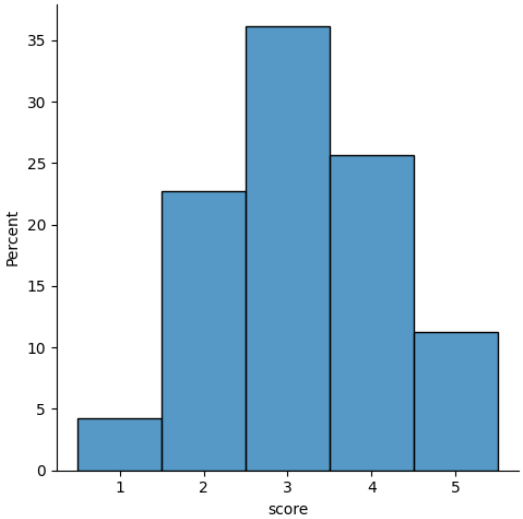


(a) Direct Prompting

(b) In-Context Examples



(c) Agent Scaffold



(d) Distilled Skill

Figure 6: Rubric score distributions across four generation settings for `Claude-Sonnet`. The distributions become increasingly bell-shaped as the augmentation level increases.

C Paper Categoristion

Paper Title
A. Language Model Adaptation and Alignment
APT: Adaptive Pruning and Tuning Pretrained Language Models for Efficient Training and Inference
A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity
BBox-Adapter: Lightweight Adapting for Black-Box Large Language Models
Stay on Topic with Classifier-Free Guidance
What Will My Model Forget? Forecasting Forgotten Examples in Language Model Refinement
B. Vision and Vision-Language Models
Bridging Data Gaps in Diffusion Models with Adversarial Noise-Based Transfer Learning
LCA-on-the-Line: Benchmarking Out-of-Distribution Generalization with Class Taxonomies
Robust CLIP: Unsupervised Adversarial Fine-Tuning of Vision Embeddings for Robust Large Vision-Language Models
Sample-specific Masks for Visual Reprogramming-based Prompting
Test-Time Model Adaptation with Only Forward Passes
C. Reinforcement Learning
Fine-tuning Reinforcement Learning Models is Secretly a Forgetting Mitigation Problem
RICE: Breaking Through the Training Bottlenecks of Reinforcement Learning with Explanation
SAPG: Split and Aggregate Policy Gradients
Unsupervised Zero-Shot Reinforcement Learning via Functional Reward Encodings
D. Probabilistic and Mathematical ML
All-in-one Simulation-Based Inference
Batch and Match: Black-Box Variational Inference with a Score-Based Divergence
Challenges in Training PINNs: A Loss Landscape Perspective
Refined Coreset Selection: Towards Minimal Coreset Size under Model Performance Constraints
Sequential Neural Score Estimation: Likelihood-Free Inference with Conditional Score Based Diffusion Models
Stochastic Interpolants with Data-Dependent Couplings

Table 4: Paper categorisation for PaperBench

D Threshold Discussion

We report the main intrinsic evaluation results with the matching threshold set to $\tau = 0.7$ in Table 2. To assess sensitivity to this choice, we also provide results for $\tau = 0.65$ and $\tau = 0.75$. Across these threshold values, our key finding remains consistent: stronger generation settings primarily improve recall, while precision fluctuates across settings.

Method	Backbone Model	Cosine Similarity		
		Precision	Recall	F1
Human Annotation	-	0.62	0.42	0.49
Direct Prompting	Claude-Sonnet	**0.57**	0.31	0.38
+*In-Context Examples*		0.53	0.36	0.41
+*Agentic Scaffold*		0.47	0.36	0.39
+*Distilled Skill*		0.54	**0.41**	**0.44**
Direct Prompting	GPT-5.4	0.41	0.24	0.28
+*In-Context Examples*		0.42	0.29	0.33
+*Agentic Scaffold*		**0.49**	0.32	**0.37**
+*Distilled Skill*		0.43	**0.35**	**0.37**

Table 5: Intrinsic meta-evaluation results with $\tau = 0.75$

Method	Backbone Model	Cosine Similarity		
		Precision	Recall	F1
Human Annotation	-	0.91	0.85	0.88
Direct Prompting	Claude-Sonnet	**0.90**	0.78	0.83
+*In-Context Examples*		0.88	0.80	0.83
+*Agentic Scaffold*		0.87	0.83	0.84
+*Distilled Skill*		0.88	**0.86**	**0.87**
Direct Prompting	GPT-5.4	0.87	0.77	0.81
+*In-Context Examples*		0.87	0.78	0.82
+*Agentic Scaffold*		**0.88**	0.79	0.82
+*Distilled Skill*		0.84	**0.81**	**0.83**

Table 6: Intrinsic meta-evaluation results with $\tau = 0.65$