Pollux: A Compact Judge and Workflow for Criteria-Based LLM Evaluation

Ulyana Isaeva[1], Egor Maslennikov[1], Alisa Glushko[1], Nikita Martynov[2], Anastasia Mordasheva[1], Dmitriy Gorbetskiy[3], Elina Basyrova[1], Sergey Skachkov[1], Victoria Berestova[4], Alena Fenogenova[1], Danil Astafurov[1]

[1]DAIMLD [2]New Economic School
[3]HSE University [4]Avito
Correspondence: danil31219as@gmail.com

Abstract

Evaluating open-ended generation remains non-trivial because quality varies across style, correctness, grounding, and safety. Moreover, a fixed prompt–response abstraction is insufficient for a reusable evaluation system as deployed applications increasingly expose multi-turn dialogues, retrieved evidence, and tool-calling traces. We present Pollux, a universal criteria-based evaluation framework centered on Pollux Judge, a compact Qwen3-4B-based LLM-as-a-Judge that evaluates such heterogeneous interactions through free-form user-defined criteria, scales, and output formats. Pollux follows the criteria methodology represented in the Pollux Benchmark (35 task groups, 2,115 expert-authored prompts), using it as a human-grounded resource for judge validation. The judge is trained with GRPO on Russian- and English-language data and is designed for bilingual deployment through a language-independent interaction schema. On the held-out expert-labeled test set, Pollux reaches a Spearman correlation of 0.744, exceeding every evaluated general-purpose baseline, including models with up to 1.1T parameters, while remaining suitable for local or OpenAI-compatible serving.

1 Introduction

Evaluating open-ended outputs of large language models (LLMs) remains difficult because quality depends on many criteria, including factuality, usefulness, safety, formatting, grounding, and style. Pairwise preference platforms and generative benchmarks have improved evaluation scalability (Zheng et al., 2023; Chiang et al., 2024; Lin et al., 2025), and specialized LLM-as-a-Judge models have shown strong agreement with human judgments (Kim et al., 2024b; Pombal et al., 2025). However, production teams still often rely on large general-purpose LLMs as zero-shot or few-shot judges, since many dedicated judge models are hard to adapt to real product workflows.

Following the Pollux criteria taxonomy, benchmark construction, and expert annotation protocol introduced in Martynov et al. (2025), we present **Pollux** as a deployable criteria-based evaluation framework for open-ended generation. The framework is centered on **Pollux Judge**, an enhanced compact and bilingual LLM-as-a-Judge trained on Russian- and English-language data: it replaces large general-purpose LLM judges in repeated evaluation runs while keeping the scoring protocol explicit and controlled. Pollux treats judging as a production interface problem: heterogeneous artifacts are represented as message-native evaluation requests, allowing the same judge endpoint to score final answers, RAG answers with retrieved evidence, side-by-side outputs, release monitoring, safety reviews, and agentic tool-use traces.

Pollux operationalizes a criteria-based methodology: each request pairs the artifact with one task-relevant criterion rather than eliciting a single holistic preference. Free-form criteria, score scales, and output formats make the interface robust to new protocols without retraining, while passing one separately formulated criterion per request reduces criterion interference and makes potential bias inspectable.

The framework combines Pollux Judge with two supporting components. The **Pollux Benchmark** grounds this methodology in human-labeled data: 35 major task groups, 152 fine-grained task types, 2,115 expert-authored instructions, and 66 criteria with score rubrics. Reusable local and OpenAI-compatible workflows package the judge for practical use by preparing inputs, applying criteria, validating parseable outputs, and exporting reports. We evaluate Pollux Judge by expert agreement, robustness across formats and scenarios, generalization to unseen task-criterion combinations, bias behavior, parse reliability, and practical latency, memory, and token-cost characteristics.

Contribution. The task taxonomy, criteria and

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 50–57
November 6-10, 2026 ©2026 Association for Computational Linguistics

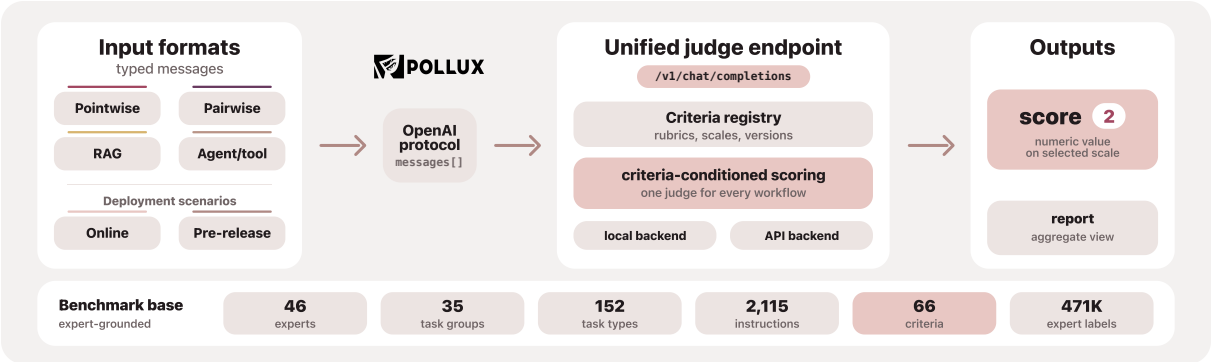


Figure 1: System view of Pollux. Diverse input formats and deployment scenarios are rendered as OpenAI-compatible message requests to a single criteria-conditioned judge endpoint. Its criteria registry is grounded in a human-built benchmark resource with task taxonomy, expert-authored instructions, criteria, rubrics, and expert labels.

rubrics, benchmark construction, and expert annotations were introduced by Martynov et al. (2025); our synthetic-data construction follows that task-and-criteria methodology. This paper contributes a compact bilingual Qwen3-4B judge trained with two-stage GRPO, a message-native evaluation interface, and reusable local or OpenAI-compatible workflows for serving, validation, aggregation, and reporting.

2 Pollux Framework Overview

Pollux is a judge-centered framework whose language-independent, message-native interface represents an evaluated interaction as a message sequence paired with one criterion per request, rather than as a fixed prompt–response pair. As shown in Figure 1, it connects **Pollux Judge**, the human-grounded **Pollux Benchmark**, and workflow tools for request preparation, validation, aggregation, and reporting. The same endpoint supports pointwise, side-by-side, RAG, safety, release-monitoring, and agent evaluation. Users may select a Pollux criterion or define a new criterion, score scale, and output format in free-form language. Figure 4 shows a concrete pointwise interaction in which the interface validates the request and returns both a criterion-level score and a structured rationale. A runnable system demo is available in the Pollux repository[1].

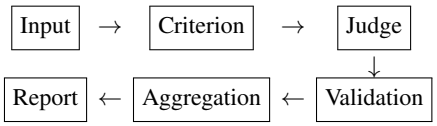


Figure 2: Pollux evaluation workflow from an input interaction to a validated, aggregated evaluation report.

Retrieved evidence, tool calls, and tool results are additional messages, so dialogues and agent traces require no separate judge architecture. For multiple criteria, the workflow issues independent requests and preserves criterion-specific scales; it validates each score against the requested scale, retries or flags invalid outputs, and reports both reasoning and example-level judgments. To support seamless integration into existing evaluation systems, we implement native Pollux adapters for DeepEval[2] and LightEval[3], allowing users to invoke the judge through each framework's standard evaluation workflow. The same repository provides the code and workflow scripts as an MIT-licensed installable release, and the Pollux-4B-Judge checkpoint can be downloaded or served from HuggingFace Hub[4].

3 Criteria-Based Evaluation Methodology and Benchmark

The methodological core of Pollux is criteria-based evaluation. Instead of assigning a single holistic preference score, each evaluation item pairs an artifact to be judged with a task-relevant criterion and a score rubric. This representation makes the judge behavior explicit: the model must decide how well the answer satisfies a specific criterion under a defined scale.

The Pollux Benchmark and its construction protocol were introduced in Martynov et al. (2025).

[1] https://github.com/ai-forever/POLLUX

[2] https://github.com/ulyanaisaeva/deepeval

[3] https://github.com/ulyanaisaeva/lighteval

[4] https://huggingface.co/ai-forever/Pollux-4B-Judge

We summarize them here because they supply the criteria, rubrics, and expert-labeled splits used by the deployable judge system. The benchmark instantiates this methodology as human-labeled data. It provides (i) a taxonomy of open-ended generation tasks, (ii) an inventory of criteria and rubrics aligned with those tasks, and (iii) expert criterion-level judgments split into *human train* and *human test* for judge validation. Synthetic training data for Pollux Judge follows the same task and criteria structure and is described in Section 4. Robustness, bias, and production stress slices are derived from the *human test* split (see Section 3.3) and reported as part of the system evaluation in Section 5. The released benchmark includes the full task and criteria metadata and is published on HuggingFace Hub[5].

3.1 Task Taxonomy and Expert Instructions

Pollux first defines the space of open-ended tasks that the judge must support. The taxonomy was derived from real user-LLM interactions: after cleaning a large subset of WildChat, we clustered 181K user prompts and merged annotated cluster labels into a compact set of major task types. Expert panels then refined this bottom-up structure into a hierarchy of **35 major task groups** and **152 fine-grained task types**, covering the main functional textual styles and multiple complexity levels. For each task group, experts wrote new instructions rather than reusing web texts or benchmark items. This reduces contamination risk and makes the benchmark suitable as a gold evaluation set. The current benchmark contains **2,115 expert-authored instructions**, stratified by task type and difficulty.

3.2 Criteria and Rubrics

Pollux does not evaluate answers with a single holistic score. Each task is paired with criteria that are necessary for assessing quality in that task: for example, fact preservation may be central for journalistic style transfer, while style consistency may be more important in a fictional setting. The criteria therefore depend on both task type and textual domain. The criteria were built from a general evaluation matrix covering context-independent quality, quality relative to the user request, and quality relative to external data sources, each split into form and content aspects. Expert panels populated this matrix for their task areas; overlapping criteria were then merged, and domain- or task-specific criteria were added. The final inventory contains **66 criteria**, corresponding to 58 unique criterion labels, grouped into:

- Critical criteria, capturing major violations;
- Fine-grained criteria, capturing general aspects of answer quality;
- Subjective criteria, capturing human-perceived qualities such as expressiveness;
- Domain-specific criteria, tied to textual styles;
- Task-specific criteria, tied to task families.

Each criterion is paired with a score scale and rubric. Most criteria use a discrete 0/1/2 scale, with values corresponding to poor, partially acceptable, and good performance; critical criteria are binary, and some STEM tasks use a wider scale. This rubric design makes scores interpretable and lets the judge produce criterion-specific rationales rather than opaque global preferences.

3.3 Expert Annotation and Data Splits

The annotation stage turns the task and criteria methodology into a gold human reference for judge quality. Strong LLMs generated answers to the expert-written instructions, and expert panels evaluated each instruction-answer pair under the relevant criteria, producing criterion-level scores and rationales. After filtering, Pollux contains 471,515 point estimates. The annotation effort required 24,447 expert hours and cost approximately $262K. This scale is central to the system motivation: expert annotation is too expensive to repeat for every release, but it provides the human-grounded reference needed to train and validate an automatic judge. Judge training and validation use three functional splits. *Synthetic train* contains **1M criterion-level scores** and provides broad training coverage under the Pollux task and criteria structure. The expert-labeled gold reference is split into *human train* with **100K scores**, used to train the judge to reproduce expert preferences, and *human test* with **50K scores**, reserved for final validation. We reserve **10 task types and their associated criteria** as a strict out-of-training subset: they are excluded from every training dataset, including both *synthetic train* and *human train*, and appear only in *human test*. This split tests whether the judge transfers the criteria-based methodology to new evaluation settings rather than memorizing a fixed label set.

[5] https://huggingface.co/datasets/ai-forever/POLLUX

4 Pollux Judge

Pollux Judge is the central reusable model in the framework. It operationalizes the criteria-based methodology from Section 3 as a judge endpoint: the input contains a message sequence, a user-defined criterion and score scale, and optional scenario-specific evidence such as a reference answer, retrieved context, or a tool trace. The output contains criterion-specific reasoning followed by a strict parseable score aligned with the requested scale. By default, Pollux Judge works in a reference-free setting, while the same message-native input schema also covers RAG faithfulness checks, side-by-side comparisons, and tool-trace evaluation. We instantiate the judge on top of Qwen3-4B so that criteria-based evaluation can be run repeatedly under local or OpenAI-compatible serving without relying on large general-purpose LLM judges for every release check.

4.1 Judge Training Data

Pollux Judge is trained with GRPO in two consecutive stages: *synthetic train* provides broad task–criterion coverage, then expert-labeled *human train* aligns the model with human preferences by continuing from the first-stage checkpoint; no supervised fine-tuning stage is used. The two stages contain Russian- and English-language interactions under the same language-independent schema, while *human test* remains held out for final validation. The training examples follow the same message-native, criteria-based schema used at inference time: each example contains an instruction, a generated answer, a task-relevant criterion, a rubric, and a target score, with scenario-specific fields added when available. Side-by-side supervision is derived from the same pointwise labels in *synthetic train* and *human train*. We form pairs only between answers to the same instruction under the same criterion, then assign labels by rule: two score-0 answers become *both bad*, two score-2 answers become *both good*, and otherwise the higher-scored answer is preferred. Pairs where both answers have score 1 are excluded because the middle rubric value does not reliably indicate either equality or a clear quality class. To construct this corpus, we first prompted DeepSeek-R1 (Guo et al., 2025), GPT-4o[6], and o3-mini[7] with task and complexity definitions from the Pollux taxonomy, obtaining 78,000 candidate instructions. After filtering and deduplication, 26,000 instructions remained and were paired with the criteria associated with their task types. For answer generation, we used a balanced pool of 15 open-weight LLMs spanning the Qwen, Gemma, Llama, Falcon, Phi, Vikhr, and RuAdapt families; exact checkpoint identifiers are released with the per-sample metadata. We then obtained rubric-aligned target scores and explanatory comments for each instruction–answer–criterion triple using DeepSeek-V3 (Liu et al., 2024) as a teacher judge. The full generation process produced 8,000,000 synthetic instruction–answer–criterion score examples. We then performed task-stratified sampling to obtain the final training set of **1M synthetic scores**. Using separate systems for instruction generation, answer generation, and score annotation reduces family-specific leakage and discourages the judge from learning one model's stylistic preferences. Finally, to remove syntactic and semantic duplicates while preserving diversity (within each task type), we computed pairwise similarities using Qwen2-7B embeddings and chrF. We then calibrated task-specific thresholds as the 95th percentile of cosine and chrF computed on expert-written instructions and, whenever a sample exceeded its task-specific threshold, applied an LLM-as-a-Judge (GigaChat-2-Max) to decide whether it was a semantic duplicate, removing it if confirmed. The resulting synthetic instruction set contains no semantic duplicates and preserves diversity comparable to the expert data.

4.2 Experiments

We apply the same GRPO (Shao et al., 2024) optimization setup to Qwen3-4B (Qwen Team, 2025) in both training stages, implemented in verl (Sheng et al., 2024). We use a learning rate of 1×10^{-6} and sample $K = 10$ rollouts per prompt at temperature 1.0 for exploration, with a training batch of 128 prompts partitioned into minibatches of 32 per policy update. Following the DrGRPO recipe, we do not normalize group advantages by their standard deviation and set the entropy bonus to zero. KL regularization is applied as a token-level reward penalty against a fixed reference policy initialized from the same checkpoint used to start each GRPO stage.

Prompts and responses are capped at 4096 and 2048 tokens, respectively, and overlong prompts are filtered out rather than truncated. We train for up to 15 epochs and validate every 5 optimization

[6] https://openai.com/index/hello-gpt-4o/

[7] openai-o3-mini

	Pointwise			Pointwise robustness			RAG	Pairwise	Tool use
Benchmark	**BiGGenBench**	**FLASK**	**Pollux** human test	**Free-form** criteria	**Prompt** format	**Output** format	**RAG** Truth	**Reward** Bench	**ToolPRM** Bench
Metrics	Pearson	Pearson	RMSE / F1 / Spearman	RMSE / F1 / Spearman	RMSE / F1 / Spearman	RMSE / F1 / Spearman	F1	Accuracy	Acc./ tie rate
Pollux Judge (Pollux prompt)	0.61	0.65	**0.57 / 0.71 / 0.74**	**0.65 / 0.62 / 0.62**	**0.60 / 0.67 / 0.70**	**0.60 / 0.67 / 0.71**	**0.43**	0.86	0.36/0.35
Pollux Judge (Selene prompt)	0.59	0.58	0.58 / 0.63 / 0.69	0.87 / 0.49 / 0.39	0.69 / 0.59 / 0.65	0.66 / 0.61 / 0.67	0.40	0.75	0.34/0.36
Atla Selene Mini (Selene prompt)	0.58	0.61	0.95 / 0.28 / 0.23	1.05 / 0.22 / 0.19	0.91 / 0.23 / 0.21	0.94 / 0.24 / 0.21	0.41	**0.89**	0.21/0.39
Atla Selene Mini (Pollux prompt)	0.21	0.23	1.71 / 0.21 / 0.23	1.85 / 0.17 / 0.18	1.73 / 0.19 / 0.20	0.74 / 0.18 / 0.20	0.23	0.53	0.16/0.60
Prometheus 2 7B	0.50	0.55	0.79 / 0.39 / 0.52	1.08 / 0.31 / 0.25	0.91 / 0.36 / 0.42	0.88 / 0.37 / 0.44	0.37	0.72	0.23/0.41
Qwen3.5-4B	0.49	0.54	0.72 / 0.44 / 0.58	0.98 / 0.38 / 0.32	0.80 / 0.46 / 0.50	0.78 / 0.47 / 0.53	0.39	0.81	0.24/0.43
Qwen3.5-122B-A10B	0.63	0.66	0.61 / 0.48 / 0.67	0.86 / 0.48 / 0.40	0.68 / 0.58 / 0.63	0.66 / 0.60 / 0.65	0.41	0.87	0.32/0.33
GPT-4o	**0.65**	**0.69**	0.60 / 0.56 / 0.68	0.80 / 0.54 / 0.49	0.62 / 0.63 / 0.67	0.61 / 0.64 / 0.68	0.43	0.85	**0.39/0.26**

Table 1: Evaluation matrix for Pollux and baseline judges across pointwise, RAG, side-by-side, and tool-use settings. BiGGenBench and FLASK report Pearson correlation; Pollux *human test* and robustness cells report RMSE, F1, and Spearman correlation. RAGTruth reports F1, RewardBench reports accuracy, and ToolPRMBench reports accuracy/tie rate. Best results are shown in **bold**; lower is better for RMSE and tie rate, and higher is better for all other metrics.

steps. Each rollout is required to follow a strict output format: an explicit `</think>` reasoning trace followed by a single integer prediction. We parse this prediction into a scalar and score it with a negative mean-squared-error reward, $r = -(\hat{y} - y)^2$; rollouts whose answer cannot be parsed as an integer receive a fixed penalty of -5. Rollouts are generated with vLLM (Kwon et al., 2023) at tensor-parallel size 2; training uses FSDP2 with offloading, gradient checkpointing, sequence packing, and dynamic token-length batching. Experiments run on 32 GPUs.

5 Evaluation and Results

We evaluate Pollux Judge along four deployment-relevant axes: agreement with human criterion-level judgments, transfer to external judge workflows, robustness of the message-native interface, and production reliability. Table 1 summarizes the evaluation matrix; all values, including baselines, are recomputed by us under the same splits, protocols, and metric implementations. Each external column follows its native target: BiGGen-Bench (Kim et al., 2024a) and FLASK (Ye et al., 2024) test fine-grained open-ended generation quality, mapped to aligned Pollux criteria and scored by Pearson correlation; RAGTruth (Niu et al., 2024) tests retrieved-evidence faithfulness and reports F1; RewardBench (Lambert et al., 2024) uses prompt–chosen–rejected preference triples and reports pairwise accuracy; ToolPRMBench (Li et al., 2026) compares correct and rejected next tool actions and reports accuracy/tie rate.

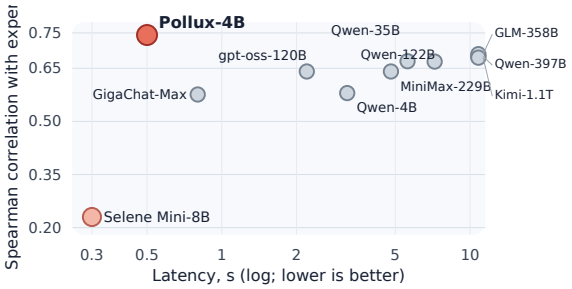


Figure 3: Quality–latency trade-off on Pollux *human test*, measured by us under a shared protocol. Pollux reaches 0.744 Spearman correlation with expert scores, higher than all evaluated general-purpose baselines, including models up to 1.1T parameters. Runs over 10 seconds share the rightmost bucket.

5.1 Criterion-Level Judge Quality

The main in-domain evaluation uses the held-out *human test* split from the Pollux Benchmark. Its strict generalization subset comprises 10 task types and their associated criteria that are absent from every training dataset, including both *synthetic train* and *human train*. This setting tests whether the judge transfers the criteria-based methodology to unseen task–criterion combinations rather than memorizing annotated labels. We report RMSE for score calibration, macro-F1 for discrete label accuracy, and Spearman correlation for ranking agreement. External pointwise benchmarks test generalization beyond the Pollux criteria inventory.

5.2 Transfer to External Workflows

We evaluate whether the same judge interface extends beyond single-answer pointwise scoring. Across RAG, preference, and tool-use settings, Pollux uses semantically aligned criteria while preserving the same message-native request structure.

5.3 Robustness Probes

We build robustness sets from the same held-out *human test* split used for judge-quality evaluation, so each perturbed item preserves an expert score and can be compared with the canonical rendering of the same instruction, answer, and criterion. These sets test whether the message-native interface remains stable under production-style variation rather than overfitting to one prompt envelope. The base pool contains out-of-training criteria and task instances; for each slice we sample criteria, instructions, and model answers to cover the three-point score scale. We then create three controlled perturbation families. *Free-form criteria* keeps the prompt template fixed but rewrites the criterion description through user-style paraphrases, minor typos, and compressed rubrics. *Prompt format* keeps the original criterion wording but changes the surrounding prompt envelope, from semi-structured task descriptions to freer natural-language requests and shuffled template fields. *Output format* keeps the evaluation content fixed but asks the judge to return different schemas, including score-only, JSON, and score-plus-explanation orders. For each family, we report RMSE, macro-F1, and Spearman correlation against the original human scores.

5.4 Self-Bias Evaluation

We measure the self-bias of Qwen3-4B using Pollux-4B as a reference judge on aligned prompts and answers generated by Qwen3-4B and Gemma-4-E4B-IT. For answer source m, we define $d_m = \bar{s}_{\text{Qwen}}(m) - \bar{s}_{\text{Pollux}}(m)$. To report bias as a non-negative effect size, we use the absolute difference-in-differences, $|d_{\text{Qwen}} - d_{\text{Gemma}}|$; the signs of the component gaps retain the direction of the effect.

5.5 Production Reliability and Efficiency

Finally, we evaluate production efficiency through the measured quality–latency trade-off in Figure 3. This measurement complements quality scores by testing whether Pollux Judge can be used as a practical local or OpenAI-compatible endpoint rather than only as an offline benchmark model.

Answer source	Qwen3-4B judge mean	Pollux-4B judge mean	d_m mean (std)
Qwen3-4B	0.8830	1.1489	-0.2660 (0.9179)
Gemma-4-E4B-IT	1.5319	1.3085	0.2234 (0.7056)
Absolute DiD self-bias $\lvert d_{\text{Qwen}} - d_{\text{Gemma}} \rvert$			**0.4894**

Table 2: Self-bias results on aligned prompts. The per-source gap is $d_m = \bar{s}_{\text{Qwen}}(m) - \bar{s}_{\text{Pollux}}(m)$. We report the absolute difference-in-differences magnitude, which is non-negative by definition; the signed source-level gaps show the effect direction.

6 Conclusion

Pollux is a deployable, bilingual framework for criteria-aligned evaluation of LLM interactions. It combines a human-grounded benchmark with a compact Qwen3-4B judge, free-form criterion requests, robustness checks, and reusable workflows for dialogues, RAG, comparisons, and tool traces. The public framework, data, and judge provide a transparent and efficient basis for repeated model evaluation.

Limitations

Pollux is validated on textual tasks, the Pollux Benchmark, and selected external judge scenarios. Specialized domains, multimodal artifacts, or different annotation norms may require recalibration. Filtering and deduplication of synthetic data address diversity and repetition, but we do not report an independent accuracy or consistency check of the DeepSeek-V3-generated scores. The subsequent expert-label training stage does not directly validate those synthetic labels. Its scores depend on the supplied criteria and rubrics, and automatic judging should support, not replace, expert review in high-stakes settings.

Ethical Statement

Automatic judges can lower evaluation cost but may amplify errors or biases without oversight. Pollux is intended for auditing, monitoring, and model comparison, and we release robustness and bias checks to support inspection and replication.

References

Andrei Alexandru, Antonia Calvi, Henry Broomfield, Jackson Golden, Kyle Dai, Mathias Leys, Maurice Burger, Max Bartolo, Roman Engeler, Sashank Pisupati, Toby Drane, and Young Sun Park. 2025. Atla selene mini: A general purpose evaluation model. Preprint, arXiv:2501.17195.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. Preprint, arXiv:2403.04132.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948.

Seungone Kim, Juyoung Suk, Ji Yong Cho, Shayne Longpre, Chaeeun Kim, Dongkeun Yoon, Guijin Son, Yejin Cho, Sheikh Shafayat, Jinheon Baek, Sue Hyun Park, Hyeonbin Hwang, Jinkyung Jo, Hyowon Cho, Haebin Shin, Seongyun Lee, Hanseok Oh, Noah Lee, Namgyu Ho, and 13 others. 2024a. The biggen bench: A principled benchmark for fine-grained evaluation of language models with language models. Preprint, arXiv:2406.05761.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024b. Prometheus 2: An open source language model specialized in evaluating other language models. Preprint, arXiv:2405.01535.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. Preprint, arXiv:2309.06180.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Rewardbench: Evaluating reward models for language modeling. Preprint, arXiv:2403.13787.

Dawei Li, Yuguang Yao, Zhen Tan, Huan Liu, and Ruocheng Guo. 2026. Toolprmbench: Evaluating and advancing process reward models for tool-using agents. Preprint, arXiv:2601.12294.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. 2023. Generative judge for evaluating alignment. arXiv preprint arXiv:2310.05470.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. 2025. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. In International Conference on Representation Learning, volume 2025, pages 47852–47870.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. arXiv preprint arXiv:2412.19437.

Nikita Martynov, Anastasia Mordasheva, Dmitriy Gorbetskiy, Danil Astafurov, Ulyana Isaeva, Elina Basyrova, Sergey Skachkov, Victoria Berestova, Nikolay Ivanov, Valeriia Zanina, and Alena Fenogenova. 2025. Eye of judgement: Dissecting the evaluation of russian-speaking LLMs with Pollux. Preprint, arXiv:2505.24616.

Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, Kashun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2024. Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. Preprint, arXiv:2401.00396.

José Pombal, Dongkeun Yoon, Patrick Fernandes, Ian Wu, Seungone Kim, Ricardo Rei, Graham Neubig, and André FT Martins. 2025. M-prometheus: A suite of open multilingual llm judges. arXiv preprint arXiv:2504.04953.

Qwen Team. 2025. Qwen3 technical report. Preprint, arXiv:2505.09388.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. arXiv preprint arXiv: 2409.19256.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. 2024. Flask: Fine-grained language model evaluation based on alignment skill sets. Preprint, arXiv:2307.10928.

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 Advances in Neural Information Processing Systems 36 (NeurIPS 2023) Datasets and Benchmarks Track, pages 46595–46623. Curran Associates, Inc.

A Related Work on LLM-as-a-Judge

LLM-as-a-Judge evaluation is now a common way to scale assessment of open-ended model outputs. MT-Bench and Chatbot Arena established judge- and preference-based protocols (Zheng et al., 2023; Chiang et al., 2024), while WildBench and Auto-J Eval extend automated judging to more realistic user requests (Lin et al., 2025; Li et al., 2023). Specialized evaluators such as Prometheus 2, M-Prometheus, and Atla Selene Mini show that smaller judge models can be competitive with

larger general-purpose LLM judges (Kim et al., 2024b; Pombal et al., 2025; Alexandru et al., 2025).

Recent benchmarks also make evaluation criteria and deployment scenarios more explicit. Prometheus 2 and BiGGen Bench use rubrics or fine-grained criteria instead of only holistic quality scores (Kim et al., 2024b,a). Scenario-specific resources include RAGTruth for retrieval-augmented faithfulness (Niu et al., 2024), RewardBench for preference-style comparisons (Lambert et al., 2024), and ToolPRMBench for tool-use process evaluation (Li et al., 2026). These lines of work define the judge models and external validation settings used in Section 5.

B Demo Interface

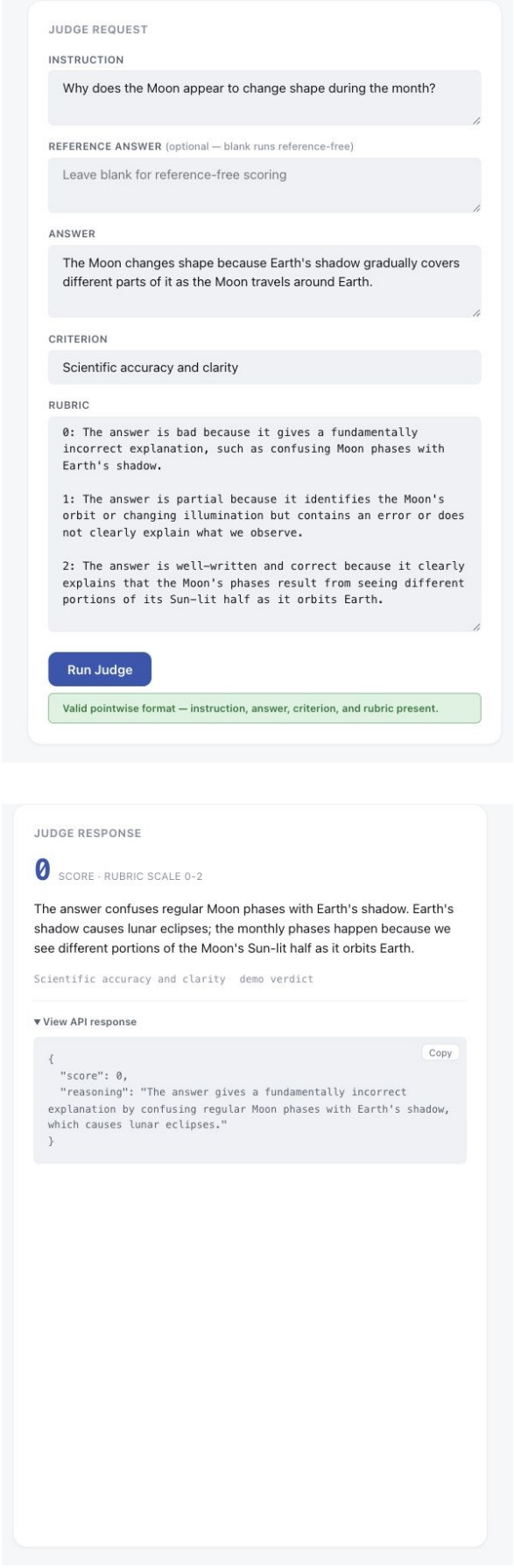


Figure 4: Pollux demo interface for pointwise criteria-based evaluation. The request panel defines the input and rubric; the response panel shows the validated score, rationale, and structured API response.