BUILDING THE RELIABLE EVALUATION LAYER OF AUTO RESEARCH WITH ON-DEMAND BENCHMARKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Auto Research is emerging as a new paradigm that extends Large Language Model (LLM)-Agent assistance to the autonomous exploration and discovery of scientific experiments. However, research on novel topics may lack reliable benchmarks for supporting its claims. While Auto Research systems generate test cases to fill this gap, the resulting evaluations may favor their own methods, raising concerns about fair comparison. To this end, we introduce Idea2AutoEval, a general framework that constructs reliable evaluations on demand from research intents for Auto Research. The framework iteratively organizes prior benchmarks into domain taxonomies to identify established evaluations and clarify the scope of new ones. Building on the in-domain benchmark library, it combines research claims and benchmark coverage to define the evaluation scope and scoring criteria. To support evaluation integrity, we adopt the information separation mechanism that exposes only task inputs to evaluated models while reserving reference data exclusively for the evaluator. Guided by these, our framework couples benchmark generation with iterative quality assurance to support the validity of model comparisons through semantic verification and alignment with research goals. Experiments span eight benchmark settings across four task categories, with datasets produced by two generation models. Spearman correlations with public model rankings exceed 0.8 in 14 of 16 comparisons, while rankings from the two generators correlate above 0.85 in seven of eight settings. On four representative tasks, Idea2AutoEval outperforms BenchMaker and BenchBench in macro-average Spearman correlation with public rankings, reaching 0.8810. Further analyses show that feedback can improve agreement with public rankings, while calibration of the intended difficulty remains task dependent. These findings support generated benchmarks for model comparison and suggest their potential as evaluation feedback for Auto Research. Our code is available at `https://anonymous.4open.science/r/Idea2AutoEval-D1DB/`.

1 INTRODUCTION

The rise of Auto Research is broadening the role of Large Language Model (LLM)-agents in scientific discovery beyond assistance with individual research tasks (Yamada et al., 2025; Liu et al., 2026; Achiam et al., 2023). The methods developed by these systems typically require iterative empirical evaluation to refine their design and support their resulting claims. For novel topics, however, available benchmarks may not cover the capabilities or conditions needed to obtain that evidence. Synthesizing evaluation datasets can address this gap, but their task distribution may favor the proposed method when the same system controls both method development and evaluation design (Yuan et al., 2026b). Access to search tools may expose Auto Research systems to publicly available benchmark solutions, creating a risk that reported scores reflect answer retrieval rather than the proposed method's performance (Shao et al., 2026). These concerns motivate that Auto Research systems introduce the external auto-evaluations, separating evaluation from method development.

When reliable third-party benchmarks are available, existing Auto Research systems can automate the assessment of method performance (Chan et al., 2025; Liu et al., 2024b; Mialon et al., 2024). Furthermore, intermediate evaluation results can be used for the system to iterate its solution (Liu et al., 2026; Zhou et al., 2024; Xie et al., 2024). If such iterations call for evaluation beyond existing benchmark coverage, recent methods translate evaluation goals set by benchmark developers

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

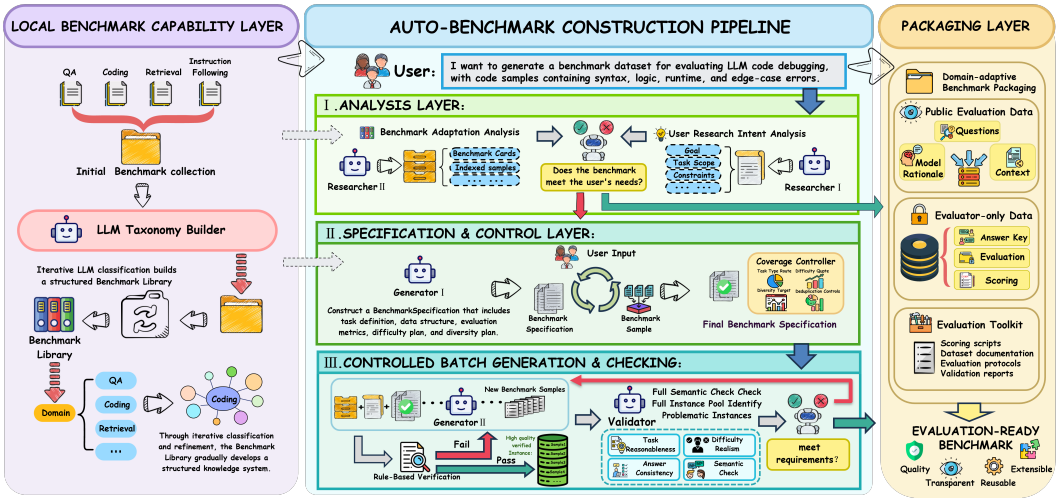

Figure 1: Overview of Idea2AutoEval. The framework separates candidate generation from benchmark release through coverage control, structural validation, domain-specific validation, semantic quality auditing, and private evaluation packaging.

into task descriptions that guide data generation (LI et al., 2025; Butt et al., 2024). To improve data quality for specific methods, cross-checking reference answers helps detect labeling errors that could distort method comparisons (Yuan et al., 2026a; Zheng et al., 2026). In parallel, research systems use protected metric reporting to limit score manipulation by experiment code within their own workflows (Liu et al., 2026). These advances leave open how Auto Research can obtain fair evaluations on demand while keeping evaluation separate from method development.

In this paper, we introduce Idea2AutoEval, an external evaluation component that constructs benchmarks on demand for Auto Research (Figure 1). We argue that Auto Research should separate evaluation design and scoring from method iteration and finalization. The framework first builds an extensible benchmark library and iteratively organizes its knowledge into domain taxonomies. These taxonomies map the capabilities covered by existing benchmarks and the evaluation protocols they use within each domain (Srivastava et al., 2022; Liang et al., 2022; Wang et al., 2018). Matching each research intent with this map identifies appropriate evaluations and coverage gaps, which inform the task distribution and scoring criteria for additional data (Zheng et al., 2023). Iterative quality assurance uses semantic verification to check generated instances against these requirements, while replacements retain the coverage assigned to rejected instances. Our framework thus grounds evaluation design and validation in research intent and established benchmark knowledge to provide evidence for validating the effectiveness of proposed methods.

We evaluate benchmark reliability across eight settings from four task categories, with DeepSeek and GPT-5.6 Luna constructing datasets under the same evaluation requirements. Agreement with public model rankings exceeds a Spearman correlation of 0.8 in 14 of 16 comparisons. Rankings from the two generators also correlate above 0.85 in seven settings, supporting stability under generator replacement. With the generation backend fixed to DeepSeek, we compare Idea2AutoEval with baselines (Yuan et al., 2026a; Zheng et al., 2026) on four representative tasks. Idea2AutoEval leads in eight of twelve task–metric comparisons and attains the highest macro-average Spearman correlation with public rankings, at 0.8810. We further examine whether feedback can improve benchmarks with weaker initial agreement. Averaged over the two generators, one feedback round improves public-reference agreement on QA–GPQA and Coding–Aider133, although gains vary by generator and later rounds are not consistently beneficial. Difficulty analysis recovers the intended ordering in ten of sixteen benchmark variants when scores are averaged across evaluated models, revealing task-dependent limits in translating specified difficulty into empirical performance.

Idea2AutoEval provides Auto Research with an external evaluation component whose design is grounded in research requirements and established benchmark knowledge. The experiments demonstrate that its generated benchmarks can preserve model comparisons across task categories, while analyses of refinement and difficulty characterize how evaluation behavior depends on construc-

tion choices. Together, these contributions provide an empirical basis for on-demand benchmark construction and support its potential role in supplying reference signals for subsequent research.

2 Related Work

Auto Research and Its Evaluation Needs. Auto Research extends language model agents from assisting individual research tasks to executing broader scientific workflows, integrating idea generation, implementation, experimentation, and paper writing into automated loops (Lu et al., 2024; Schmidgall et al., 2025), with more recent systems further incorporating agentic search, iterative refinement, failure recovery, and human intervention (Yamada et al., 2025; Liu et al., 2026). Empirical evaluation is therefore embedded throughout the research process, with intermediate results guiding subsequent iterations and final results supporting research claims. Existing benchmarks evaluate research agents on predefined objectives such as machine-learning experimentation and engineering (Huang et al., 2023; Chan et al., 2025), data-driven scientific analysis (Chen et al., 2025), computational reproducibility (Siegel et al., 2024), and paper replication (Starace et al., 2025). However, fixed tasks cannot cover every capability or condition introduced by open-ended research ideas.

Automated Benchmark Construction and Evaluation. Automated benchmark construction has evolved from template-based behavioral testing and human–model interaction (Ribeiro et al., 2020; Kiela et al., 2021) to continuously refreshed test sets that reduce exposure to static public data (White et al., 2025; Jain et al., 2025). Recent LLM-based systems automate domain discovery, task decomposition, instance generation, and validation (LI et al., 2025; Butt et al., 2024); BenchMaker introduces a generic generation pipeline with explicit quality control, while BenchBench uses structured domain cards, quota-controlled generation, and multi-model validation (Yuan et al., 2026a; Zheng et al., 2026). Automatic evaluators further enable scalable scoring of open-ended outputs (Liu et al., 2023b; Zheng et al., 2023; Kim et al., 2024), but validity remains vulnerable to judge bias (Thakur et al., 2025), generator self-preference (Yuan et al., 2026b), adaptive overfitting (Dwork et al., 2015), and protocol shortcuts (Shao et al., 2026).

3 From Idea To Auto Evaluation (Idea2AutoEval)

In this section, we present Idea2AutoEval, an automated framework which constructs evaluation from research requirements for Auto Research systems . We first describe how prior benchmarks are organized into a knowledge base that supports semantic retrieval and adaptation (Section 3.1). Building on this knowledge base, we then explain how research requirements are translated into an Evaluation Specification that governs instance generation and validation under coverage constraints (Section 3.2). Finally, we describe how validated benchmarks are packaged for model evaluation, separating task inputs from the private reference information used for scoring (Section 3.3).

3.1 Benchmark Knowledge Base and Semantic Routing

Existing benchmarks typically offer information about which capabilities are assessed and how, providing design knowledge even when their datasets do not fit the current research request. To support subsequent retrieval and adaptation, Idea2AutoEval standardizes each benchmark as a Benchmark Card and organizes these profiles by capability, with successive levels for domain and task. For the i-th Benchmark Card p_i, we first identify its primary capability κ_i. Its domain δ_i and concrete task form ϑ_i complete the semantic address $\mathbf{z}_i = (\kappa_i, \delta_i, \vartheta_i)$. This triplet provides each benchmark with a semantic address in the capability–domain–task space. The capability level groups benchmarks by what they assess, while the domain and task levels distinguish the settings and forms of that assessment. At each semantic position u, the system maintains a routing contract $\Omega_u^{(j)}$, which specifies admissible child tasks as well as their permitted task structures and evaluation constraints.

Let $\mathcal{P}_u$ denote the set of Benchmark Cards associated with semantic position u. The accumulated semantic address $\mathbf{z}_u$ describes the current path, while ℓ_u identifies the semantic axis to be partitioned at the next level. The system constructs an initial routing contract from representative Benchmark Cards at the current position. The construction process $\mathrm{Draft}(\cdot)$ follows the target partitioning

dimension within the existing semantic path:

$$\Omega_u^{(0)} = \mathrm{Draft}\left(\mathrm{ProfileSample}(\mathcal{P}_u), \mathbf{z}_u, \ell_u\right). \tag{1}$$

Routing uses the task definition and input–output schema to establish the benchmark's task structure. Its evaluation protocol also informs the mapping of p_i to the next-hop semantic position $\hat{u}_i$ at iteration j, given by $(\hat{u}_i, \gamma_i, \epsilon_i) = \mathrm{Route}(p_i, \Omega_u^{(j)})$. Here, ϵ_i records the task and evaluation features supporting the routing decision, and $\gamma_i \in [0, 1]$ is the normalized confidence output by the classifier. A higher γ_i indicates stronger agreement between the Benchmark Card and the routing contract of the target position. When $\gamma_i < \lambda$, where λ is a predefined threshold, the route is marked as low confidence and retained as evidence for subsequent taxonomy revision. If the current semantic structure contains no position that can accommodate the benchmark, we set $\hat{u}_i = \emptyset$.

A routing decision alone does not guarantee that a benchmark can support downstream evaluation design. The system additionally checks the actual task definition and input–output structure against the routing contract of the target position. The evaluation protocol must also satisfy that contract. We set $\mathrm{Contract}(p_i, \hat{u}_i) = 1$ when all three conditions are satisfied, and 0 otherwise. This binary compatibility check serves a different role from the continuous routing confidence γ_i. We define the semantic routing gap rate $\Delta_u^{(j)}$ to measure the hard coverage deficiency at a semantic position:

$$\Delta_u^{(j)} = \frac{1}{|\mathcal{P}_u|} \sum_{p_i \in \mathcal{P}_u} \mathbb{I}\left[\hat{u}_i = \emptyset \ \vee \ \mathrm{Contract}(p_i, \hat{u}_i) = 0\right]. \tag{2}$$

This quantity captures two types of hard failures: no suitable semantic position exists in the current structure, or a route exists but its task structure or evaluation protocol violates the target contract. Low-confidence cases do not directly increase $\Delta_u^{(j)}$, because they still have provisionally acceptable routes. Yet they may reveal ambiguous category boundaries or insufficient definitions and are therefore retained for revision. The feedback set includes benchmarks with no admissible route or a route that violates the contract. Benchmarks assigned with low confidence are retained in the same set:

$$\mathfrak{F}_u^{(j)} = \{p_i \mid \hat{u}_i = \emptyset \ \vee \ \gamma_i < \lambda \ \vee \ \mathrm{Contract}(p_i, \hat{u}_i) = 0\}. \tag{3}$$

$\mathfrak{F}_u^{(j)}$ retains broader evidence for revision. The system first performs local adjustments. If local adjustments fail to reduce the routing gap, the system expands the revision scope and repartitions the task structure under the current capability–domain position. The system uses the feedback set to progressively refine the current routing contract through $\mathrm{Amend}(\cdot)$, with the revision history $\Xi_u^{(j)}$ guiding the update:

$$\Omega_u^{(j+1)} = \mathrm{Amend}\left(\Omega_u^{(j)}, \mathfrak{F}_u^{(j)}, \Xi_u^{(j)}\right). \tag{4}$$

Revising the taxonomy helps distinguish gaps in its representation from gaps in the available benchmarks. After multiple rounds of routing and contract revision, the system assembles the local benchmark knowledge base from the refined routing structure and the benchmark information it organizes:

$$\mathcal{Z} = (\mathcal{P}, \mathcal{R}, \mathcal{A}, \mathcal{I}). \tag{5}$$

Here, $\mathcal{P}$ denotes the standardized Benchmark Card collection, organized by the capability–domain–task semantic routing structure $\mathcal{R}$. The final semantic addresses of benchmarks are stored in $\mathcal{A}$, and the retrieval index $\mathcal{I}$ supports subsequent Benchmark Analysis.

3.2 Research-Requirement-Driven Benchmark Construction

In Figure 1, Researcher I denotes the intent-understanding module, while Researcher II denotes the Benchmark Analysis module. The generation module is shown in two roles: Generator I defines the Evaluation Specification, while Generator II produces candidate instances under its coverage constraints.

Intent Representation and Benchmark Retrieval. The research requirement provides an interface between method development and the external evaluator. Given a research requirement r expressed

in natural language, the intent-understanding module transforms it into a structured research intent $\mathbf{h}$. This representation retains the research objectives and task requirements directly relevant to evaluation design and serves as a query for accessing the local benchmark knowledge base.

The system uses $\mathbf{h}$ to query the retrieval index $\mathcal{I}$ and obtains a candidate benchmark set $\mathcal{C}_r$ relevant to the current requirement. The Benchmark Analysis module examines each candidate through its semantic address, assessing whether its task structure and evaluation protocol suit the current requirement. This analysis determines whether the evaluation structure of an existing benchmark can satisfy the target research requirement.

Adaptation Analysis and Evaluation Specification. For a candidate benchmark $p \in \mathcal{C}_r$, $\mathrm{DirectFit}(p, \mathbf{h})$ indicates that its task structure and evaluation protocol satisfy the target intent without modification. $\mathrm{ReusableSkeleton}(p, \mathbf{h})$ indicates that its task structure or evaluation protocol can serve as a reusable basis, though the evaluation requires adaptation. The data may require changes in content or coverage, while explicit evaluation constraints may also need adjustment. The Benchmark Analysis module selects a construction strategy s as

$$s = \begin{cases} \textsc{Reuse}, & \exists p \in \mathcal{C}_r : \mathrm{DirectFit}(p, \mathbf{h}), \\ \textsc{Adapt}, & \neg\exists p \in \mathcal{C}_r : \mathrm{DirectFit}(p, \mathbf{h}) \wedge \exists p \in \mathcal{C}_r : \mathrm{ReusableSkeleton}(p, \mathbf{h}), \\ \textsc{Construct}, & \text{otherwise.} \end{cases} \tag{6}$$

Under REUSE, the system directly adopts the corresponding evaluation scheme. Under ADAPT, the system preserves the reusable task skeleton while modifying content or coverage as needed. It can also adjust constraints where the target requirement calls for a change. Under CONSTRUCT, no candidate benchmark provides a suitable task structure or evaluation protocol, and the system designs a new benchmark. In addition to s, the analysis module records adaptation rationales indicating which design components can be inherited and which must be redefined.

For ADAPT and CONSTRUCT, the generation module uses the candidate-benchmark analysis and adaptation information to translate the research requirement into an Evaluation Specification $\mathcal{S}$:

$$\mathcal{S} = (\Gamma_{\mathrm{task}}, \boldsymbol{\pi}, \mathcal{E}_{\mathrm{eval}}) . \tag{7}$$

Here, the task contract Γ_{task} describes the input–output relation and response format. The coverage plan $\boldsymbol{\pi}$ constrains the target sample distribution. The evaluation contract $\mathcal{E}_{\mathrm{eval}}$ defines the information and domain-specific rules required for automated scoring. The specification supplies a common reference for generation and validation, making the intended comparison explicit before large-scale construction. Explicit user requirements are incorporated into the Evaluation Specification $\mathcal{S}$ with priority. Before large-scale construction, the system generates a small set of pilot instances from $\mathcal{S}$. In automated experiments, these instances undergo the same validation as final instances, with structural checks complemented by domain-specific and semantic assessment. The system proceeds to large-scale construction only after the pilot stage satisfies the current Evaluation Specification $\mathcal{S}$.

Coverage-Constrained Instance Generation. A generator's tendency to produce certain task types need not reflect the coverage required by the research question. After the Evaluation Specification $\mathcal{S}$ is confirmed, the Coverage Controller converts coverage targets into instance-level quotas. Let $\mathcal{W}$ denote the set of coverage units and N the target number of instances. For a coverage unit ω, n_ω is the allocated instance count, while π_ω is its target proportion. The system first imposes user-specified quotas as hard constraints, then allocates the remaining budget to make the realized coverage approximate the target distribution:

$$\begin{aligned} \min_{\{n_\omega\}} \quad & \sum_{\omega \in \mathcal{W}} \left| \frac{n_\omega}{N} - \pi_\omega \right|, \qquad \text{s.t.} \qquad \sum_{\omega \in \mathcal{W}} n_\omega = N, \qquad n_\omega \in \mathbb{Z}_{\geq 0}, \\ & n_\omega = c_\omega, \qquad \forall \omega \in \mathcal{W}_{\mathrm{exact}}, \qquad n_\omega \geq l_\omega, \qquad \forall \omega \in \mathcal{W}_{\min}. \end{aligned} \tag{8}$$

$\mathcal{W}_{\mathrm{exact}}$ denotes coverage units with explicitly specified exact quotas, and c_ω is the corresponding quota. $\mathcal{W}_{\min}$ denotes coverage units with minimum-count requirements, and l_ω is the associated lower bound. For units without explicit user constraints, the system allocates the remaining budget according to $\boldsymbol{\pi}$. The resulting coverage units are expanded into instance-level slots and ordered into a coverage queue $\mathcal{Q}$. Each $q \in \mathcal{Q}$ retains its associated task requirements and evaluation conditions.

The generation module produces a candidate instance x for the current slot, ensuring that every generation attempt corresponds to a predefined coverage objective rather than freely sampling from the full Evaluation Specification $\mathcal{S}$.

Validation and Targeted Refill. An instance can be well formed yet fail to test the capability required by the research intent. A candidate instance enters the valid instance pool $\mathcal{X}$ only after passing all quality gates. The final outcomes of structural and domain-specific validation are denoted by $V_{\mathrm{str}}(x)$ and $V_{\mathrm{dom}}(x)$. Semantic validation yields $V_{\mathrm{sem}}(x)$. The instance validity is defined as:

$$V(x) = V_{\mathrm{str}}(x) \cdot V_{\mathrm{dom}}(x) \cdot V_{\mathrm{sem}}(x). \tag{9}$$

All three validation outcomes belong to $\{0, 1\}$. A value of 1 indicates that the candidate passes the corresponding check, whereas 0 indicates rejection. Structural validation checks conformance to the unified data interface and the correct separation of public and private information. Domain-specific validation determines whether an instance can be validly executed and scored according to the rules of the current task. Semantic validation checks for task drift and answer conflicts that are not fully covered by the first two levels. It also checks for near-duplicate instances.

Thus, a candidate instance is accepted into the valid instance pool if and only if $V(x) = 1$, namely, $x \in \mathcal{X}$. Domain-specific and semantic validation may use program execution or rule-based checking. Model-based analysis may be applied on its own or combined with these checks. Regardless of the internal mechanism, their outputs are converted into unified binary gate signals so that all candidate instances follow the same acceptance criterion.

Filtering alone can leave frequently rejected task types underrepresented, allowing generation difficulty to narrow the evaluation scope. When $V(x) = 0$, the system records the candidate instance and its failure reason in a rejection log $\mathcal{J}_{\mathrm{rej}}$. Suggestions produced by validators are not directly accepted as valid instances. Instead, the system preserves the original coverage slot and generates a new candidate for that slot. The new instance must undergo the complete validation procedure again. This design keeps refilling tied to the original coverage plan, so quality checks can trigger replacement without redefining the assigned coverage requirements.

3.3 Release Packaging and Evaluation Preparation

Once the valid instance pool $\mathcal{X}$ reaches the target size, the system applies a final release gate. This stage checks whether planned coverage requirements are satisfied and verifies that public data correctly correspond to private evaluation information. Inputs accessible to evaluated models must exclude reference answers. Hidden tests and private labels are also reserved for the evaluator. Access to reference information during prediction can change what a score measures, even when the scoring procedure is unchanged.

Let $\mathrm{PublicView}(x)$ denote the model-visible public projection of a valid instance x. The final public input collection $\mathcal{X}_{\mathrm{pub}}$ consists of public views of all valid instances:

$$\mathcal{X}_{\mathrm{pub}} = \{\mathrm{PublicView}(x) \mid x \in \mathcal{X}\}. \tag{11}$$

After passing the release gate, the final benchmark is represented as $\mathcal{B}_{\mathrm{final}} = (\mathcal{X}_{\mathrm{pub}}, \mathcal{Y}, \mathcal{H}, g)$. $\mathcal{Y}$ stores reference answers and scoring metadata. $\mathcal{H}$ stores evaluator-only information. The automated scoring function g is defined by the Evaluation Specification $\mathcal{S}$.

During evaluation, model inference accesses only $\mathcal{X}_{\mathrm{pub}}$, whereas the scoring function g computes results using $\mathcal{Y}$ and $\mathcal{H}$. This separation establishes distinct information boundaries for model inference and result adjudication while maintaining consistency through the same Evaluation Specification $\mathcal{S}$.

Established benchmark knowledge gives the external evaluator a basis for choosing evaluation conditions, and the Evaluation Specification carries those conditions through construction. Coverage control and targeted refill retain the assigned coverage requirements during validation, while the public interface keeps scoring references out of model inputs. These mechanisms allow Auto Research to delegate evaluation design and assessment to an external component while retaining responsibility for method iteration. The generated benchmarks provide a basis for assessing proposed methods under the requested conditions, including those beyond the coverage of existing datasets.

Table 1: Consistency between synthetic benchmarks and public leaderboards. DeepSeek-Generated and Luna-Generated denote benchmarks constructed using the corresponding generation models.

	DeepSeek-Generated			Luna-Generated		
Dataset	ρ	τ	r	ρ	τ	r
QA-MMLU-Pro (5)	1.000	1.000	0.901	0.872	0.738	0.877
QA-GPQA (8)	0.714	0.571	0.745	0.299	0.182	0.550
Coding-MBPP (8)	0.929	0.857	0.922	0.850	0.691	0.856
Coding-Aider133 (8)	0.861	0.741	0.771	0.838	0.691	0.819
Retrieval-MTEB-Rerank (8)	1.000	1.000	0.930	0.952	0.857	0.954
Retrieval-SciFact (8)	0.976	0.929	0.990	1.000	1.000	0.970
Instruction-IFEval (8)	0.881	0.714	0.969	0.929	0.857	0.988
Instruction-IFBench-OOD (8)	0.905	0.786	0.960	0.833	0.714	0.905

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUPS

Task Categories. We evaluate Idea2AutoEval across four broad categories that cover distinct forms of model capability: question answering, coding, retrieval, and instruction following. Table 2 in Appendix A.1 summarizes their evaluation scope. Each category contains two complementary task settings, yielding eight public–synthetic benchmark alignment settings in total. For every setting, Idea2AutoEval constructs a synthetic benchmark that preserves the target capability, task format, evaluation protocol, and primary metric while generating new evaluation instances.

Generators. For each task setting, we independently construct one synthetic benchmark with DeepSeek (Liu et al., 2024a) and another with GPT-5.6 Luna (OpenAI, 2026). Both generators receive the same frozen Evaluation Specification but produce benchmark instances independently.

Evaluation Protocol and Metrics. Each task uses its native automatic metric: accuracy for question answering, executable-test pass rate for coding, average precision for MTEB reranking, nDCG@10 for SciFact retrieval, and strict all-constraints-satisfied accuracy for instruction following. We measure agreement between benchmark score vectors using Spearman's rank correlation ρ (Spearman, 1961), Kendall's rank correlation τ (Kendall et al., 1938), and Pearson's score correlation r. Public–DeepSeek and Public–Luna quantify alignment with the corresponding public benchmark, while DeepSeek–Luna measures stability across generated benchmark versions.

Comparison Protocol. All task-level comparisons use the same matched model set across the score vectors being compared, and correlations are computed independently for each task rather than pooling scores across tasks. This common protocol applies throughout the subsequent experiment sections; complete task mappings, model lists, metric definitions, and experiment-specific configurations are documented in Appendix A.

4.2 PUBLIC–SYNTHETIC BENCHMARK ALIGNMENT.

Table 1 shows strong overall alignment between the synthetic Benchmarks and their public counterparts. Across the eight settings, the unweighted average Spearman, Kendall, and Pearson correlations are 0.908, 0.825, and 0.898 for the DeepSeek-generated Benchmarks and 0.822, 0.716, and 0.865 for the Luna-generated Benchmarks, respectively. Moreover, 14 of the 16 public–synthetic comparisons achieve a Spearman correlation above 0.8. These results demonstrate the effectiveness of Idea2AutoEval in constructing synthetic Benchmarks that faithfully reproduce both the relative model rankings and the score-level performance patterns observed on established public benchmarks. Table 1 further shows that the alignment is robust across generators and task domains. Both the DeepSeek- and Luna-generated Benchmarks achieve Spearman correlations above 0.8 in seven of the eight settings, covering MMLU-Pro in QA and every setting in Coding, Retrieval, and Instruction Following. This shared pattern indicates that the strong public-benchmark alignment is not tied

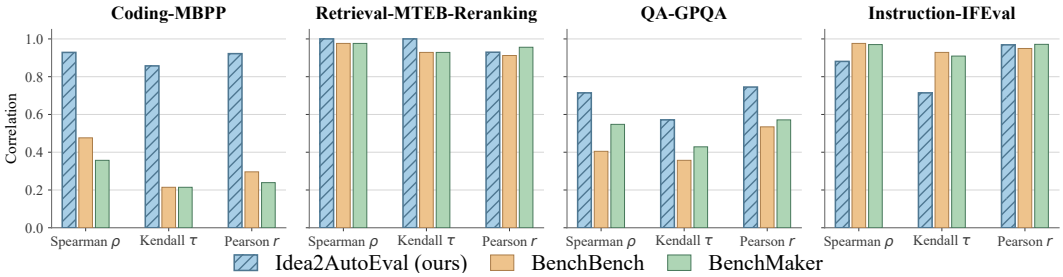

Figure 2: Correlation between model performance on automatically constructed test sets and the corresponding public benchmarks across four domains.

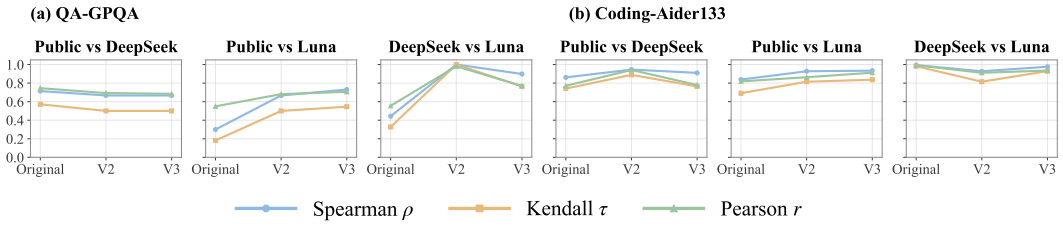

Figure 3: Correlation changes during feedback-driven iterative benchmark refinement.

to a particular generator or a single task domain. Full-precision results and direct DeepSeek–Luna comparisons are provided in Appendix A.3.

4.3 Comparison with Automated Benchmark Construction Baselines

Figure 2 presents the consistency between the constructed benchmarks and their corresponding public benchmarks. Overall, Idea2AutoEval achieves the highest macro-average Spearman correlation across the four domains ($\rho = 0.8810$), and obtains the best result in 8 out of 12 domain–metric comparisons. Idea2AutoEval achieves the best performance on all three metrics for both Coding-MBPP and QA-GPQA. In particular, it obtains a Spearman correlation of $\rho = 0.9286$ on Coding-MBPP, indicating that it can better reproduce the model ranking of the public benchmark. On Retrieval-MTEB-Reranking, Idea2AutoEval achieves perfect ranking consistency, with both Spearman ρ and Kendall τ reaching 1.0000. The results on Instruction-IFEval are more metric-dependent: BenchBench achieves stronger rank correlations, whereas BenchMaker obtains a slightly higher Pearson correlation. Overall, these results show that Idea2AutoEval provides stronger and more consistent alignment with public benchmarks than existing automated benchmark construction methods.

4.4 Feedback-Driven Iterative Benchmark Refinement

Figure 3 shows that feedback-driven regeneration can improve the consistency between self-constructed and public benchmarks, but the benefit is concentrated in the first refinement round. For QA-GPQA, the average Spearman correlation increases from 0.5069 in *Original* to 0.6667 in *V2*, with corresponding improvements in Kendall's τ and Pearson's r. The Luna-generated benchmark shows the clearest correction, whose Spearman correlation with the public benchmark rises from 0.2994 to above 0.6 after one iteration and continues to improve in *V3*. By contrast, the DeepSeek-generated benchmark changes only modestly across rounds, indicating that refinement is most useful when the initial benchmark exhibits larger disagreement. A similar pattern appears on Coding-Aider133. *V2* achieves the strongest overall consistency, with average Spearman ρ, Kendall τ, and Pearson r reaching 0.9368, 0.8519, and 0.9023, respectively. Further refinement in *V3* does not consistently improve these results: the Luna-generated benchmark continues to gain alignment, whereas the DeepSeek-generated version declines, causing the overall consistency to fall slightly from *V2*. The agreement between independently generated DeepSeek and Luna benchmarks also changes non-monotonically across rounds in both tasks. Overall, feedback-driven refinement is effective for correcting initially misaligned benchmarks, but most of the improvement is obtained

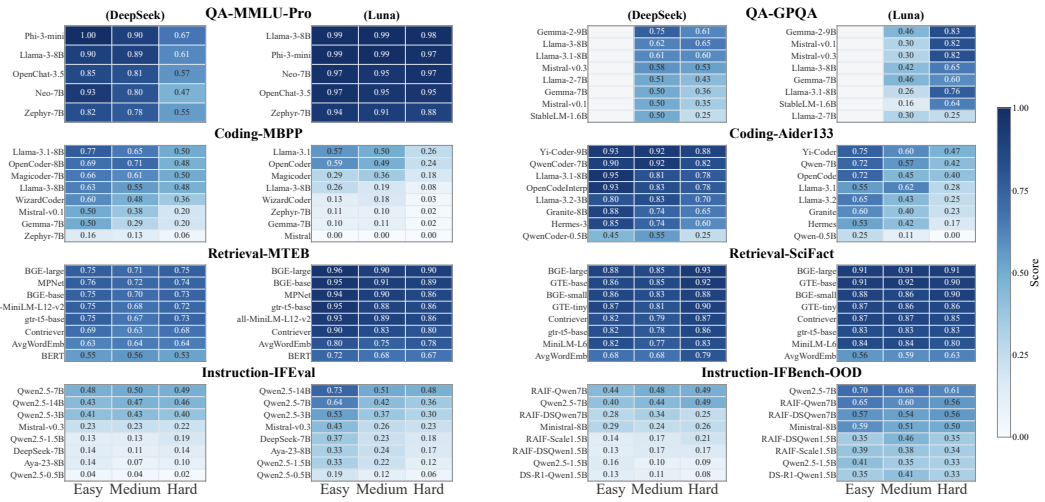

Figure 4: Difficulty-stratified main-metric heatmaps for 16 synthetic Benchmark variants.

after a single regeneration round. Additional iterations provide limited marginal benefit and may even reduce consistency, suggesting that one refinement round is generally sufficient in this setting.

4.5 Difficulty-Stratified Evaluation.

Figure 4 shows that the intended difficulty ordering is recovered in 10 of the 16 benchmark variants when scores are averaged across models. The pattern is most consistent for QA–MMLU-Pro and the two Coding tasks. For example, on the DeepSeek-generated MMLU-Pro benchmark, mean accuracy decreases from 0.900 on `easy` instances to 0.574 on `hard` instances. Similar easy-to-hard declines appear on both MBPP and Aider133 variants, and most individual models follow the same downward trend. This indicates that the assigned strata capture meaningful empirical difficulty differences in these settings. The calibration is nevertheless generator- and task-dependent. On MMLU-Pro, the DeepSeek version shows a clear easy-to-hard decrease, whereas the Luna version remains near ceiling across strata. GPQA shows an even stronger discrepancy, with the DeepSeek version decreasing from `medium` to `hard`, while the Luna version increases. Retrieval tasks also exhibit weaker or inconsistent gradients, including nearly flat or reversed trends. Overall, difficulty-conditioned generation is effective in producing measurable performance gradients in many settings.

5 Conclusions

We introduced Idea2AutoEval, an external evaluation framework that constructs benchmarks on demand from research intents for Auto Research. Its main contribution is to connect evaluation design with established benchmark knowledge while separating method development from benchmark construction and scoring. Idea2AutoEval organizes prior benchmarks into domain taxonomies, translates research requirements and coverage gaps into explicit evaluation specifications, and combines coverage-constrained generation with iterative validation and private reference packaging. Across eight settings spanning question answering, coding, retrieval, and instruction following, the generated benchmarks largely preserved public model rankings across two independent generators, with 14 of 16 public–synthetic comparisons achieving Spearman correlations above 0.8. The framework also obtained the highest macro-average rank correlation among the evaluated automated benchmark construction methods. These results support Idea2AutoEval as a practical evaluation component for Auto Research, while the variation observed in GPQA and difficulty calibration shows that reliability remains task- and generator-dependent. Future work will therefore focus on more robust calibration and stopping criteria for refinement under such challenging settings.

AI USE STATEMENT

We used DeepSeek and GPT-5.6 Luna to generate benchmark instances and generative AI tools to assist with manuscript drafting and language editing. The generated instances were validated using the procedures described in the paper. The authors take full responsibility for the final content of this work, including all text, data, results, and claims.

REPRODUCIBILITY STATEMENT

We provide an anonymous link to our implementation to support reproducibility. The paper describes the benchmark construction and validation procedures, and the appendix documents the task settings, benchmark generators, evaluated models, and evaluation protocols. Detailed model scores and correlation results are also provided to facilitate verification of the reported comparisons and analyses.

REFERENCES

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone, 2024. URL https://arxiv.org/abs/2404.14219.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Viraat Aryabumi, John Dang, Dwarak Talupuru, Saurabh Dash, David Cairuz, Hangyu Lin, Bharat Venkitesh, Madeline Smith, Jon Ander Campos, Yi Chern Tan, et al. Aya 23: Open weight releases to further multilingual progress. *arXiv preprint arXiv:2405.15032*, 2024.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Marco Bellagente, Jonathan Tow, Dakota Mahan, Duy Phung, Maksym Zhuravinskyi, Reshinth Adithyan, James Baicoianu, Ben Brooks, Nathan Cooper, Ashish Datta, et al. Stable lm 2 1.6 b technical report. *arXiv preprint arXiv:2402.17834*, 2024.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Natasha Butt, Varun Chandrasekaran, Neel Joshi, Besmira Nushi, and Vidhisha Balachandran. Benchagents: Multi-agent systems for structured benchmark creation. *arXiv preprint arXiv:2410.22584*, 2024.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, et al. Mle-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, volume 2025, pp. 50466–50494, 2025.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, volume 2025, pp. 96934–96990, 2025.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toni Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. *Advances in neural information processing systems*, 28, 2015.

IBM Granite Team. Granite 3.0 language models. *URL: https://github. com/ibm-granite/granite-3.0-language-models*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. Mlagentbench: Evaluating language agents on machine learning experimentation. *arXiv preprint arXiv:2310.03302*, 2023.

Siming Huang, Tianhao Cheng, Jason Klein Liu, Weidi Xu, Jiaran Hao, Liuyihan Song, Yang Xu, Jian Yang, Jiaheng Liu, Chenchen Zhang, et al. Opencoder: The open cookbook for top-tier code large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33167–33193, 2025.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*, 2021.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Maurice G Kendall et al. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. Dynabench: Rethinking benchmarking in nlp. In *Proceedings of the 2021 conference of the North American chapter of the Association for Computational Linguistics: human language technologies*, pp. 4110–4124, 2021.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4334–4353, 2024.

XIANG LI, Farzaan Kaiyom, Evan Liu, Yifan Mai, Percy Liang, and Tatsunori Hashimoto. Autobencher: Towards declarative benchmark construction. In *International Conference on Learning Representations*, volume 2025, pp. 94357–94386, 2025.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

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

Jiaqi Liu, Shi Qiu, Mairui Li, Bingzhou Li, Haonian Ji, Siwei Han, Xinyu Ye, Peng Xia, Zihan Dong, Meng Chen, et al. Autoresearchclaw: Self-reinforcing autonomous research with human-ai collaboration. *arXiv preprint arXiv:2605.20025*, 2026.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023a.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, volume 2024, pp. 52989–53046, 2024b.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 2511–2522, 2023b.

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

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. Wizardcoder: Empowering code large language models with evol-instruct. In *International Conference on Learning Representations*, volume 2024, pp. 27168–27188, 2024.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *International Conference on Learning Representations*, volume 2024, pp. 9025–9049, 2024.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. Mteb: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2014–2037, 2023.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, et al. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 9844–9855, 2022.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hanna Hajishirzi. Generalizing verifiable instruction following. *Advances in Neural Information Processing Systems*, 38, 2026.

Yulei Qin, Gang Li, Zongyi Li, Zihan Xu, Yuchen Shi, Zhekai Lin, Xiao Cui, Ke Li, and Xing Sun. Incentivizing reasoning for advanced instruction-following of large language models. *Advances in Neural Information Processing Systems*, 38:108337–108401, 2026.

Jeffrey Quesnelle and Chen Guang. Hermes 3 technical report. *arXiv preprint arXiv:2408.11857*, 2024.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 4902–4912, 2020.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using llm agents as research assistants. *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 5977–6043, 2025.

Jiaqi Shao, Hanck Chen, Wei Zhang, Maxm Pan, and Bing Luo. Do agent benchmarks measure capability? protocol validity in the age of agentic ai. *arXiv preprint arXiv:2607.22368*, 2026.

Zachary S Siegel, Sayash Kapoor, Nitya Nadgir, Benedikt Stroebl, and Arvind Narayanan. Corebench: Fostering the credibility of published research through a computational reproducibility agent benchmark. *arXiv preprint arXiv:2409.11363*, 2024.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33: 16857–16867, 2020.

Charles Spearman. The proof and measurement of association between two things. 1961.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, et al. Paperbench: evaluating ai's ability to replicate ai research. *arXiv preprint arXiv:2504.01848*, 2025.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow,

Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. Judging the judges: Evaluating alignment and vulnerabilities in llms-as-judges. In *Proceedings of the Fourth Workshop on Generation, Evaluation and Metrics (GEM^2)*, pp. 404–430, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro Von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7534–7550, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.609. URL https://aclanthology.org/2020.emnlp-main.609/.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. Openchat: Advancing open-source language models with mixed-quality data. In *International Conference on Learning Representations*, volume 2024, pp. 57021–57040, 2024a.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024b.

Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Empowering code generation with oss-instruct. *arXiv preprint arXiv:2312.02120*, 2023.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, et al. Livebench: A challenging, contamination-limited llm benchmark. In *International Conference on Learning Representations*, volume 2025, pp. 91595–91631, 2025.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pp. 641–649, 2024.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

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

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Guoyin Wang, Heng Li, Jiangcheng Zhu, Jianqun Chen, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

Peiwen Yuan, Shaoxiong Feng, Yiwei Li, Xinglin Wang, Yueqi Zhang, Jiayi Shi, Chuyi Tan, Boyuan Pan, Yao Hu, and Kan Li. Llm-powered benchmark factory: Reliable, generic, and efficient. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35882–35903, 2026a.

Peiwen Yuan, Yiwei Li, Shaoxiong Feng, Xinglin Wang, Yueqi Zhang, Jiayi Shi, Chuyi Tan, Boyuan Pan, and Yao Hu. Silencer: From discovery to mitigation of self-bias in llm-as-benchmark-generator. *Advances in Neural Information Processing Systems*, 38:129636–129658, 2026b.

Ge Zhang, Scott Qu, Jiaheng Liu, Chenchen Zhang, Chenghua Lin, Chou Leuang Yu, Danny Pan, Esther Cheng, Jie Liu, Qunshu Lin, et al. Map-neo: Highly capable and transparent bilingual large language model series. *arXiv preprint arXiv:2405.19327*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. Opencodeinterpreter: Integrating code generation with execution and refinement. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12834–12859, 2024.

Yandan Zheng, Haoran Luo, Zhenghong Lin, Wenjin Liu, and Luu Anh Tuan. Benchbench: Benchmarking automated benchmark generation. *arXiv preprint arXiv:2603.20807*, 2026.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024.

Paul Gauthier. Aider polyglot coding benchmark. Aider LLM Leaderboards, 2024. URL `https://aider.chat/docs/leaderboards/`.

Gemma Team. Gemma: Open models based on gemini research and technology, 2024. URL `https://arxiv.org/abs/2403.08295`.

Binyuan Hui et al. Qwen2.5-coder technical report, 2024. URL `https://arxiv.org/abs/2409.12186`.

OpenAI. GPT-5.6 Luna model documentation, 2026. URL `https://developers.openai.com/api/docs/models/gpt-5.6-luna`. Accessed 2026-09-26.

An Yang et al. Qwen2.5 technical report, 2024. URL `https://arxiv.org/abs/2412.15115`.

A Supplementary Experimental Details

A.1 Experimental Setup

This section consolidates the full experimental configuration used for the public–synthetic benchmark alignment study. We evaluate four capability categories, instantiate two task settings per category, and independently construct each setting with two benchmark generators. This design yields eight alignment settings and sixteen synthetic benchmark versions.

Table 2: Task categories and evaluation scope.

Task category	Evaluation scope
Question answering	Measures subject-specific knowledge and reasoning at different levels of difficulty through automatically scored multiple-choice tasks.
Coding	Covers both function-level program synthesis and repository-style code editing, with correctness determined by executable tests.
Retrieval	Evaluates candidate reranking and scientific-evidence retrieval, capturing both general ranking quality and domain-specific relevance estimation.
Instruction following	Tests compliance with verifiable constraints in both familiar settings and settings requiring generalization to unseen constraints.

Task Settings and Public Reference Benchmarks. Table 3 reports the complete mapping between the eight task settings, their public reference benchmarks, and their primary evaluation metrics.

Table 3: Complete mapping between task categories, concrete task settings, public reference benchmarks, and primary metrics.

Broad category	Concrete task setting	Public reference benchmark	Primary metric
Question answering	Psychology multiple-choice knowledge and reasoning	MMLU-Pro (Psychology) (Wang et al., 2024b)	Accuracy
	Graduate-level science multiple-choice reasoning	GPQA Diamond (Rein et al., 2023)	Accuracy
Coding	Function-level Python program synthesis	MBPP+ in EvalPlus (Austin et al., 2021; Liu et al., 2023a)	Test pass rate
	Multilingual code editing with executable tests	Aider coding benchmark (133-task setting) (Gauthier, 2024)	Test pass rate
Retrieval	Candidate-document reranking	MTEB Reranking (Muennighoff et al., 2023)	Average precision
	Scientific-claim document retrieval	SciFact (Wadden et al., 2020)	nDCG@10
Instruction following	In-distribution verifiable constraint following	IFEval (Zhou et al., 2023)	Strict accuracy
	Generalization to unseen verifiable constraints	IFBench-OOD (Pyatkin et al., 2026)	Strict accuracy

Benchmark Generators. For each task setting, we independently construct one synthetic benchmark with DeepSeek and one with GPT-5.6 Luna, producing sixteen benchmark versions in total. DeepSeek is an API-served general-purpose model from the DeepSeek family, whose recent architecture is documented in the DeepSeek-V3 technical report (Liu et al., 2024a). GPT-5.6 Luna is the cost-efficient, high-throughput member of OpenAI's GPT-5.6 family (OpenAI, 2026). Both generators receive the same frozen Evaluation Specification, including the target capability, task format, scoring protocol, target size, and coverage requirements. Apart from this shared specification, the generated questions, code tests, candidate documents, and instruction instances are produced independently.

Evaluated Models. For each setting, we select the intersection of models for which both public-reference scores and synthetic-benchmark scores are available. QA–MMLU-Pro contains five matched models, while each of the other seven settings contains eight. Within a task, the Public–DeepSeek, Public–Luna, and DeepSeek–Luna comparisons always use the same matched model set.

- **QA–MMLU-Pro:** Phi-3-mini (Abdin et al., 2024), Llama-3-8B (Grattafiori et al., 2024), OpenChat-3.5 (Wang et al., 2024a), MAP-Neo-7B-Instruct (Zhang et al., 2024), and Zephyr-7B (Tunstall et al., 2023).
- **QA–GPQA:** Gemma-2-9B (Team et al., 2024), Llama-3.1-8B, Gemma-7B (Gemma Team, 2024), Meta-Llama-3-8B, Mistral-v0.3, StableLM, Llama-2-7B-Chat, and Mistral-v0.1 (Grattafiori et al., 2024; Jiang et al., 2023; Bellagente et al., 2024; Touvron et al., 2023).
- **Coding–MBPP:** OpenCoder (Huang et al., 2025), Magicoder (Wei et al., 2023), Llama-3.1, Meta-Llama, WizardCoder (Luo et al., 2024), Mistral, Gemma, and Zephyr (Grattafiori et al., 2024; Jiang et al., 2023; Gemma Team, 2024; Tunstall et al., 2023).
- **Coding–Aider133:** Qwen2.5-Coder-7B-Instruct and Qwen2.5-Coder-0.5B-Instruct (Hui et al., 2024), Yi-Coder-9B-Chat (Young et al., 2024), Llama-3.1-8B-Instruct, OpenCodeInterpreter-DS-6.7B (Zheng et al., 2024), Hermes-3-Llama-3.1-8B (Quesnelle & Guang, 2024), Llama-3.2-3B-Instruct, and Granite-3.0-8B-Instruct (Granite Team, 2024).
- **Retrieval–MTEB-Reranking:** bge-large-en-v1.5, all-mpnet-base-v2, bge-base-en-v1.5, all-MiniLM-L12-v2, gtr-t5-base, contriever-base-msmarco, average-word, and bert-base-uncased (Xiao et al., 2024; Song et al., 2020; Wang et al., 2020; Ni et al., 2022; Izacard et al., 2021; Devlin et al., 2019).
- **Retrieval–SciFact:** gte-base, bge-large, bge-small, gte-tiny, contriever, gtr-t5, multi-qa-MiniLM, and average-word (Li et al., 2023; Xiao et al., 2024; Izacard et al., 2021; Ni et al., 2022; Wang et al., 2020).
- **Instruction–IFEval:** Qwen2.5-14B, Qwen2.5-7B, Qwen2.5-3B, Mistral, Aya-23, Qwen2.5-1.5B, DeepSeek-LLM, and Qwen2.5-0.5B (Yang et al., 2024; Jiang et al., 2023; Aryabumi et al., 2024; Bi et al., 2024).
- **Instruction–IFBench-OOD:** RAIF-Qwen2.5-7B, Qwen2.5-7B, RAIF-DeepSeek-Qwen-7B, Ministral-8B, Qwen2.5-1.5B, RAIF-DeepSeek-Qwen-1.5B, RAIF-DeepScaleR-1.5B, and DeepSeek-R1-Distill (Qin et al., 2026; Yang et al., 2024; Jiang et al., 2023; Guo et al., 2025).

Evaluation Protocol and Metrics. Every evaluated model receives only the task input and any context permitted by the corresponding task definition. Gold answers, executable hidden tests, relevance labels, and verifiable-constraint annotations are withheld during inference. For a model m and an evaluation instance x, the instance-level score is

$$\mathrm{Metric}_{\mathcal{B}}(m,x) = \begin{cases} \mathbb{I}[\hat{y}_m(x) = y(x)], & \text{QA}, \\ \mathbb{I}[\mathrm{Pass}(m,x)], & \text{Coding}, \\ \mathrm{AP}(m,x), & \text{MTEB Reranking}, \\ \mathrm{nDCG@10}(m,x), & \text{SciFact Retrieval}, \\ \mathbb{I}[\mathrm{AllConstraintsSatisfied}(m,x)], & \text{Instruction Following}. \end{cases} \tag{A.1}$$

Here, $\hat{y}_m(x)$ and $y(x)$ are the predicted and gold answers; $\mathrm{Pass}(m,x)$ indicates that the generated code passes the executable tests; $\mathrm{AP}(m,x)$ is average precision over the candidate-document ranking; and $\mathrm{AllConstraintsSatisfied}(m,x)$ requires every verifiable instruction constraint to be satisfied. The benchmark score is the mean instance-level score,

$$S_{\mathcal{B}}(m) = \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \mathrm{Metric}_{\mathcal{B}}(m,x), \tag{A.2}$$

with larger values indicating better performance for every task.

Let $\mathbf{s}^{(a)}$ and $\mathbf{s}^{(b)}$ denote the score vectors produced by two benchmarks over the same n matched models. Spearman's rank correlation measures whether the two benchmarks induce the same monotonic model ordering (Spearman, 1961). We assign average ranks to tied scores and compute

$$\rho = \mathrm{corr}\Big(\mathrm{rank}(\mathbf{s}^{(a)}), \mathrm{rank}(\mathbf{s}^{(b)})\Big), \tag{A.3}$$

where $\mathrm{corr}(\cdot,\cdot)$ is Pearson correlation. Thus, $\rho = 1$ indicates identical rank order, $\rho = -1$ indicates exactly reversed order, and values near zero indicate little monotonic agreement.

Kendall's rank correlation evaluates agreement through pairwise ordering (Kendall et al., 1938). A pair of models is concordant when both benchmarks order it in the same direction and discordant when the order is reversed. Because benchmark scores can contain ties, we use the tie-aware Kendall τ_b:

$$\tau_b = \frac{C - D}{\sqrt{(C + D + T_a)(C + D + T_b)}}, \tag{A.4}$$

where C and D are the numbers of concordant and discordant model pairs, and T_a and T_b count pairs tied only under benchmarks a and b, respectively. Like Spearman's ρ, τ_b ranges from -1 to 1; however, it directly reflects the balance between correctly and incorrectly ordered model pairs.

We additionally report Pearson's score correlation r to measure linear agreement in score magnitudes. Public–DeepSeek and Public–Luna quantify alignment with the public reference benchmark, whereas DeepSeek–Luna measures the stability of model discrimination across independently generated benchmark versions. All correlations are computed separately for each task setting; no scores are pooled across tasks.

A.2 Experiment-Specific Configurations

Public–Synthetic Benchmark Alignment. This experiment uses all eight task settings listed in Table 3. For each task, the public-reference, DeepSeek-generated, and Luna-generated score vectors contain the same matched models. We report Public–DeepSeek and Public–Luna correlations to measure alignment with established benchmarks, together with DeepSeek–Luna correlations to measure robustness to generator replacement.

Automated Benchmark Construction Baselines. We compare Idea2AutoEval with BenchMaker (Yuan et al., 2026a) and BenchBench (Zheng et al., 2026) on one representative task from each category: QA–GPQA, Coding–MBPP, Retrieval–MTEB-Reranking, and Instruction–IFEval. To control for generator effects, all three methods use the same DeepSeek API backend and the same fixed domain-specific task specification, while retaining their respective construction and filtering procedures. Each task uses eight models that can be matched one-to-one with the corresponding public leaderboard, and the constructed-to-public agreement is measured with Spearman ρ, Kendall τ, and Pearson r.

Feedback-Driven Iterative Benchmark Refinement. We select QA–GPQA and Coding–Aider133 because they exhibit the lowest initial public–synthetic correlations. For each task and generator, we evaluate three successive benchmark versions: *Original*, *V2*, and *V3*. The two generators operate independently under the same task definition, and each subsequent version incorporates feedback from the preceding evaluation round. Correlations with the public benchmark and between the two generated benchmarks are recomputed after every round using the common matched-model protocol.

Difficulty-Stratified Evaluation. We apply difficulty-stratified evaluation to all eight task settings and both generators, yielding sixteen synthetic benchmark variants. During instance construction, each example is assigned an `easy`, `medium`, or `hard` label. For every model, the corresponding task-native metric is aggregated separately within each available difficulty subset; QA–GPQA contains no `easy` instances. The labels are generation-time conditions rather than a globally calibrated scale, so their empirical ordering is assessed separately for every task–generator combination.

Table 4: Complete pairwise correlation analysis among public, DeepSeek-generated, and Luna-generated benchmark evaluations.

Dataset	Pairwise Comparison	Spearman ρ	Kendall τ	Pearson r
QA-MMLU-Pro (5)	Public vs DeepSeek	1.0000	1.0000	0.9005
	Public vs Luna	0.8721	0.7379	0.8767
	DeepSeek vs Luna	0.8721	0.7379	0.8172
QA-GPQA (8)	Public vs DeepSeek	0.7143	0.5714	0.7452
	Public vs Luna	0.2994	0.1818	0.5495
	DeepSeek vs Luna	0.4431	0.3273	0.5549
Coding-MBPP (8)	Public vs DeepSeek	0.9286	0.8571	0.9220
	Public vs Luna	0.8503	0.6910	0.8564
	DeepSeek vs Luna	0.9222	0.8365	0.7780
Coding-Aider (8)	Public vs DeepSeek	0.8614	0.7407	0.7710
	Public vs Luna	0.8383	0.6910	0.8185
	DeepSeek vs Luna	0.9940	0.9820	0.9917
Retrieval-MTEB-Rerank (8)	Public vs DeepSeek	1.0000	1.0000	0.9296
	Public vs Luna	0.9524	0.8571	0.9538
	DeepSeek vs Luna	0.9524	0.8571	0.9857
Retrieval-SciFact (8)	Public vs DeepSeek	0.9762	0.9286	0.9896
	Public vs Luna	1.0000	1.0000	0.9703
	DeepSeek vs Luna	0.9762	0.9286	0.9622
Instruction-IFEval (8)	Public vs DeepSeek	0.8810	0.7143	0.9686
	Public vs Luna	0.9286	0.8571	0.9883
	DeepSeek vs Luna	0.9524	0.8571	0.9501
Instruction-IFBench-OOD (8)	Public vs DeepSeek	0.9048	0.7857	0.9601
	Public vs Luna	0.8333	0.7143	0.9048
	DeepSeek vs Luna	0.9286	0.7857	0.9530

A.3 Complete Pairwise Correlation Analysis

Table 4 supplements the public–synthetic alignment summary in Table 1 of the main text. In addition to the Public–DeepSeek and Public–Luna correlations summarized there, it reports the full-precision results and the DeepSeek–Luna comparisons used to assess robustness to generator replacement.

The direct DeepSeek–Luna comparisons show that independently generated benchmarks usually preserve similar model discrimination. Seven of the eight settings obtain Spearman correlations above 0.85, and six reach at least 0.922. The agreement is especially high for Coding–Aider133 (0.994), both retrieval settings (0.952 and 0.976), and both instruction-following settings (0.952 and 0.929). These values complement the public–synthetic comparisons by showing that the observed rankings are generally reproducible under generator replacement.

A.4 Complete Per-Model Results

Figure 5 and Tables 5–12 expose the model-level results underlying the aggregate correlations. Each table reports the public rank and score, the two synthetic-benchmark scores, and the change in rank relative to the public benchmark. An upward arrow denotes a higher synthetic rank, a downward arrow denotes a lower rank, and a dash denotes no change.

Most points in Figure 5 remain close to the diagonal, indicating that the synthetic benchmarks largely retain the public ordering while allowing the score scales to differ. The detailed tables below identify which models account for the remaining rank movements.

Question Answering. Tables 5 and 6 report the complete question-answering results. On MMLU-Pro, the DeepSeek-generated benchmark reproduces all five public ranks, while the Luna-generated

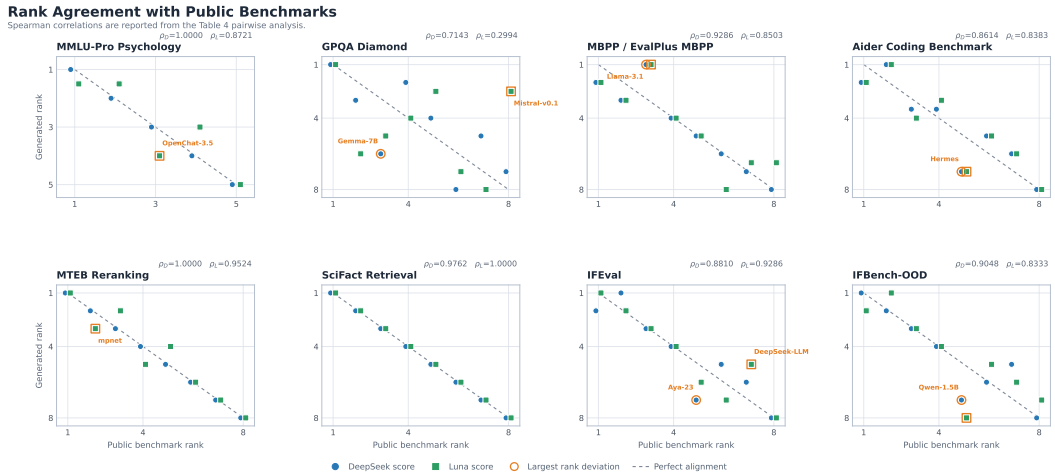

Figure 5: Rank agreement between public benchmarks and DeepSeek/Luna-based evaluations across eight tasks. Each point represents a model, with its public benchmark rank on the x-axis and the rank induced by DeepSeek (blue circles) or Luna (green squares) scores on the y-axis. Scores are independently converted into descending within-benchmark ranks; lower rank is better. The dashed diagonal indicates perfect rank alignment, while orange outlines highlight models with the largest rank deviation. ρ_D and ρ_L denote Spearman rank correlations for the DeepSeek and Luna evaluations, respectively.

benchmark introduces only local changes among the middle models and retains the endpoints of the ordering. The GPQA table records the corresponding model-level reorderings behind its aggregate correlation results.

Table 5: Detailed results of different models on the QA-MMLU-Pro benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
Phi-3-mini	1 / 65.20	85.00 −−	98.75 −−
Llama-3-8B	2 / 59.40	80.75 −−	98.75 ↑ 1
OpenChat-3.5	3 / 54.50	75.00 −−	95.75 ↓ 1
Neo-7B	4 / 36.20	72.75 −−	96.00 ↑ 1
Zephyr-7B	5 / 28.20	71.75 −−	90.75 −−

Table 6: Detailed results of different models on the QA-GPQA benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
Gemma-2-9B	1 / 38.38	69.27 −−	64.25 −−
Llama-3.1-8B	2 / 31.31	60.34 ↓ 2	50.84 ↓ 4
Gemma-7B	3 / 29.29	44.13 ↓ 3	52.51 ↓ 2
Meta-Llama-3-8B	4 / 26.77	63.69 ↑ 2	53.63 −−
Mistral-v0.3	5 / 24.24	55.87 ↑ 1	55.87 ↑ 3
StableLM	6 / 21.72	39.66 ↓ 2	39.66 ↓ 1
Llama-2-7B-Chat	7 / 19.70	47.49 ↑ 2	27.37 ↓ 1
Mistral-v0.1	8 / 16.16	43.58 ↑ 1	55.87 ↑ 6

Coding. Tables 7 and 8 provide the function-level synthesis and repository-style editing results. On Aider133, the two independently generated benchmarks assign the same direction of rank change, or no change, to seven of the eight models. This model-level agreement is consistent with their Spearman correlation of 0.994.

Table 7: Detailed results of different models on the Coding-MBPP benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
OpenCoder	1 / 82.00	63.29 ↓ 1	44.06 ↓ 1
Magicoder	2 / 70.60	59.09 ↓ 1	27.97 ↓ 1
Llama-3.1	3 / 68.30	64.34 ↑ 2	44.41 ↑ 2
Meta-Llama	4 / 64.60	55.59 – –	17.48 – –
WizardCoder	5 / 58.50	48.25 – –	11.89 – –
Mistral	6 / 51.90	36.36 – –	0.00 ↓ 2
Gemma	7 / 47.10	32.87 – –	8.04 ↑ 1
Zephyr	8 / 42.10	11.54 – –	8.04 ↑ 2

Table 8: Detailed results of different models on the Coding-Aider133 benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
Qwen2.5-Coder-7B-Instruct	1 / 57.90	88.72 ↓ 1	57.14 ↓ 1
Yi-Coder-9B-Chat	2 / 54.10	90.98 ↑ 1	60.90 ↑ 1
Llama-3.1-8B-Instruct	3 / 37.60	84.21 – –	49.62 ↓ 1
OpenCodeInterpreter-DS-6.7B	4 / 30.10	84.21 ↑ 1	51.88 ↑ 1
Hermes-3-Llama-3.1-8B	4 / 30.10	72.93 ↓ 3	37.59 ↓ 3
Llama-3.2-3B-Instruct	6 / 26.30	78.20 ↑ 1	44.36 ↑ 1
Granite-3.0-8B-Instruct	7 / 20.30	75.19 ↑ 1	40.60 ↑ 1
Qwen2.5-Coder-0.5B-Instruct	8 / 14.30	42.86 – –	12.03 – –

Retrieval. Tables 9 and 10 show particularly stable orderings. The DeepSeek-generated MTEB benchmark exactly reproduces all eight public ranks, and the Luna-generated SciFact benchmark does the same. In the other two retrieval comparisons, every displacement is at most one rank, showing that the model ordering remains stable despite differences in absolute score scale.

Table 9: Detailed results of different models on the Retrieval-MTEB-Reranking benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
bge-large-en-v1.5	1 / 60.03	73.58 – –	92.05 – –
all-mpnet-base-v2	2 / 59.36	73.44 – –	90.03 ↓ 1
bge-base-en-v1.5	3 / 58.86	72.30 – –	91.81 ↑ 1
all-MiniLM-L12-v2	4 / 58.44	71.45 – –	89.07 ↓ 1
gtr-t5-base	5 / 54.23	71.24 – –	89.33 ↑ 1
contriever-base-msmarco	6 / 53.14	66.30 – –	84.22 – –
average-word	7 / 44.75	63.82 – –	77.23 – –
bert-base-uncased	8 / 43.44	54.92 – –	68.94 – –

Table 10: Detailed results of different models on the Retrieval-SciFact benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
gte-base	1 / 76.77	87.68 ↓ 1	91.01 −−
bge-large	2 / 74.61	86.95 ↑ 1	90.91 −−
bge-small	3 / 71.28	85.18 −−	87.77 −−
gte-tiny	4 / 67.51	84.63 −−	86.51 −−
contriever	5 / 65.51	81.93 −−	86.41 −−
gtr-t5	6 / 59.74	81.19 −−	83.22 −−
multi-qa-MiniLM	7 / 54.03	79.87 −−	83.20 −−
average-word	8 / 29.53	70.99 −−	59.44 −−

Instruction Following. Tables 11 and 12 report strict constraint-satisfaction scores. For both tasks and both generators, the two strongest public models remain the top two synthetic models. On IFBench-OOD, the four highest-ranked public models also remain above the lower-ranked group in both generated benchmarks, with differences occurring only within these groups.

Table 11: Detailed results of different models on the Instruction-IFEval benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
Qwen2.5-14B	1 / 84.20	45.40 ↓ 1	56.40 −−
Qwen2.5-7B	2 / 80.60	49.00 ↑ 1	46.80 −−
Qwen2.5-3B	3 / 67.90	41.20 −−	40.00 −−
Mistral	4 / 59.00	22.80 −−	29.80 −−
Aya-23	5 / 50.60	10.40 ↓ 2	24.60 ↓ 1
Qwen2.5-1.5B	6 / 49.50	14.60 −−	22.20 ↓ 1
DeepSeek-LLM	7 / 48.90	13.00 ↑ 2	25.40 ↑ 2
Qwen2.5-0.5B	8 / 34.40	3.20 −−	12.20 −−

Table 12: Detailed results of different models on the Instruction-IFBench-OOD benchmark.

Model	Public Rank / Score	DeepSeek Score (ΔRank)	Luna Score (ΔRank)
RAIF-Qwen2.5-7B	1 / 29.82	47.20 −−	60.40 ↓ 1
Qwen2.5-7B	2 / 28.23	44.20 −−	66.40 ↑ 1
RAIF-DeepSeek-Qwen-7B	3 / 20.06	29.20 −−	55.40 −−
Ministral-8B	4 / 16.66	26.40 −−	53.00 −−
Qwen2.5-1.5B	5 / 15.64	11.60 ↓ 2	36.20 ↓ 3
RAIF-DeepSeek-Qwen-1.5B	6 / 12.92	15.60 −−	39.20 ↑ 1
RAIF-DeepScaleR-1.5B	7 / 12.58	17.40 ↑ 2	37.00 ↑ 1
DeepSeek-R1-Distill	8 / 8.80	10.80 −−	36.60 ↑ 1