LUDKA BENCH: CAN AI AGENTS SOLVE NONSTANDARD MACHINE LEARNING PROBLEMS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Solving challenging, nonstandard problems in artificial intelligence and machine learning requires identifying useful structure, choosing an effective approach, and translating it into a working solution under limited time and experimentation. We introduce a benchmark comprising 20 of the most challenging tasks from international AI olympiads and national qualifying competitions from multiple countries to evaluate these capabilities in language models and agentic systems. Beyond measuring independent problem solving, the benchmark asks whether providing the right direction is sufficient for a system to solve a difficult task. We evaluate three levels of assistance: no guidance, a strategic hint, and a detailed description of an approach used in a solution ranked among the top three on the original private leaderboard. In our current evaluation, the strongest tested system achieves 18.5 mean rank without guidance and 8.0 with a detailed description of a reference approach. Competition-experienced human participants achieve a mean rank of 6.09 under the same evaluation conditions with detailed solution descriptions, on tasks they had not previously solved. Additional guidance improves aggregate performance, but even full descriptions leave a gap relative to these participants. These findings show that even detailed guidance based on a successful solution is often insufficient for agents to achieve a comparable result without further human assistance.

1 INTRODUCTION

Large language model (LLM) agents can write training code, improve models, and run experiments (Lu et al., 2024; Jiang et al., 2025). Benchmarks increasingly evaluate them on practical machine learning tasks, including competitions and research engineering (Chan et al., 2025; Huang et al., 2024; Wijk et al., 2025). Yet success depends on choosing which ideas to test within the available time and compute. Finding a suitable representation, recognizing a useful invariance, or testing competing explanations can matter more than further hyperparameter tuning. When only a few experiments are possible, choosing promising directions is especially important (Toledo et al., 2025).

Final scores alone, however, do not show whether an agent chose a useful idea, implemented it correctly, or selected a reliable final submission. A low score may reflect failure at any of these stages. A high score may come from a familiar pipeline, broad trial and error with many weak candidates, or an ensemble of numerous methods and models, without showing that the agent understood the problem's structure. Both choosing an approach and putting it into practice are important ML skills.

AI olympiad tasks provide a natural setting for this comparison. They challenge human participants to find effective solutions under time and compute constraints, with performance measured by task-specific scoring rules. We collect 20 tasks from international AI olympiads and national qualifying competitions to study whether agents can discover useful approaches and apply them, independently or with guidance. We call this ability to identify and prioritize a promising approach strategy discovery. Only a small number of such tasks are available, which may limit opportunities for targeted training and reduce the risk of benchmark-specific overfitting.

In practical use, a person may ask an agent to solve a task independently, suggest a promising direction, or describe an approach for the agent to develop into a working solution. We study these

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.

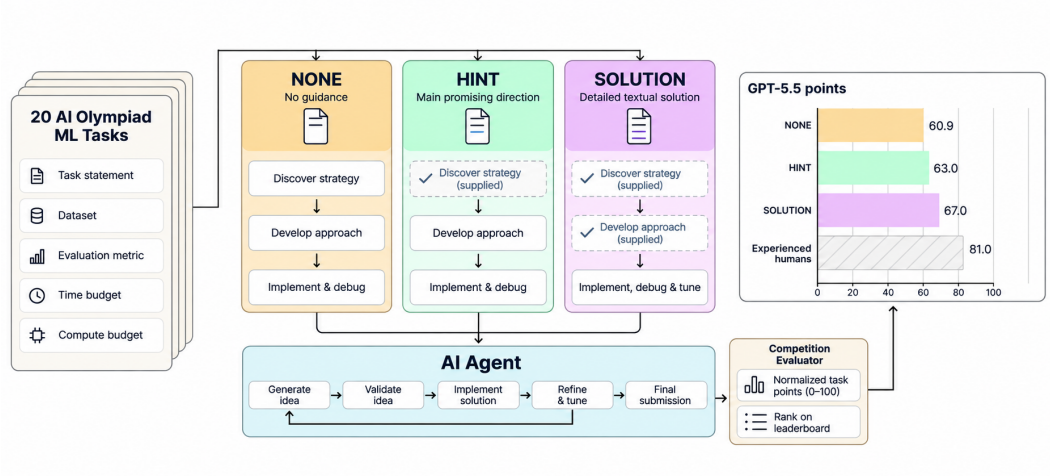


Figure 1: Overview of LUDKA Bench, which evaluates agents on 20 AI olympiad tasks under three guidance conditions with the same task-specific time and compute budgets across conditions. The right panel reports mean normalized points across the 20 tasks for GPT-5.5 (high reasoning, Codex) and experienced human participants, who were evaluated only in SOLUTION.

settings through three levels of initial guidance. In NONE, the agent receives the task without additional guidance. In HINT, it receives a concise strategic suggestion. In SOLUTION, it receives a more detailed description of a reference approach, including its main components and practical recommendations. The agent then works autonomously, making the remaining decisions, running experiments, and producing a final submission. This comparison evaluates both independent problem solving and the ability to turn supplied guidance into a competitive solution.

Building on this setup, we make three contributions:

1. We introduce a benchmark of 20 AI olympiad tasks for evaluating how agents find and apply ideas in challenging ML problems.
2. We provide a controlled evaluation protocol in which a concise hint and a detailed description are derived from the same reference solution and compared with unguided runs on the same task. By holding task resources and feedback rules fixed, the protocol measures how effectively agents use increasing amounts of solution information.
3. We evaluate several modern models and agentic systems alongside human participants, analyzing how guidance affects performance and which difficulties remain when a reference solution is provided.

2 RELATED WORK

End-to-end machine learning agents. Recent benchmarks evaluate language-model agents on complete machine learning workflows. MLAgentBench studies iterative experimentation through code modification, execution, and refinement based on feedback (Huang et al., 2024), while MLE-bench evaluates practical ML engineering across 75 Kaggle competitions (Chan et al., 2025). These benchmarks assess data processing, model selection, debugging, and experimentation. Their end-to-end scores, however, do not distinguish between choosing a poor approach, failing to turn a promising idea into an effective procedure, and errors in implementation or tuning.

Agents for machine learning research and scientific discovery. A parallel line of work evaluates broader parts of the ML research process. The AI Scientist targets end-to-end research from idea generation to manuscript writing (Lu et al., 2024). RE-Bench evaluates open-ended ML research engineering (Wijk et al., 2025), while PaperBench focuses on reproducing published research (Starace et al., 2025). EXP-Bench explicitly decomposes experimentation into hypothesis, design, imple-

mentation, execution, and analysis (Kon et al., 2026), while MLRC-Bench evaluates proposing and implementing methods in ML research competitions (Zhang et al., 2025).

Olympiad and challenge-based evaluation. Olympiad problems provide another source of difficult tasks designed to distinguish strong human problem solvers. OlympiadBench evaluates language and multimodal models on thousands of mathematics and physics problems drawn from olympiad-level competitions, emphasizing multi-step scientific reasoning (He et al., 2024). ScienceArena similarly evaluates recent models on problems from international physics, chemistry, and biology competitions using process-aware scoring rubrics (Zhao et al., 2026).

Diagnosing strategy, planning, and execution. Recent work increasingly isolates components of agent performance rather than relying only on final task success. MLE-Ideator separates strategic ideation from implementation (Zhang et al., 2026), while Agent Planning Benchmark evaluates planning independently from execution (Sun et al., 2026). Other approaches diagnose failures from agent trajectories by assessing intermediate competencies or erroneous steps (Ou et al., 2025; Fan et al., 2026).

LUDKA complements these studies by treating the amount of supplied solution information as an explicit evaluation variable. Whereas MLE-Ideator separates ideation and implementation within an agent system, LUDKA compares unguided runs with a strategic hint and a detailed description derived from the same reference solution, holding task resources and feedback rules fixed. This connects the independent problem solving evaluated in MLE-bench and MLRC-Bench with the implementation from descriptions studied in PaperBench. The resulting within-task comparisons reveal whether naming an effective direction is sufficient, whether further methodological detail helps, and which difficulties persist when that information is available.

3 Benchmark and Diagnostic Protocol

LUDKA contains 20 tasks from international AI olympiads and national qualifying competitions. We evaluate whether an agent can identify a useful approach to a nonstandard ML problem and turn it into a valid, competitive submission under the time and resource constraints of the original competition.

3.1 Target capability and task construction

We use *strategy discovery* to refer to identifying and prioritizing a promising approach from the task statement, data, model behavior, and experimental feedback. The selected tasks are designed so that strong performance typically requires discovering task-specific structure.

We constructed the benchmark from tasks released through international AI olympiads, national qualifying competitions, and closely related preparation rounds. The candidate pool was intended to cover essentially all publicly accessible, substantive tasks from these competitions that we could identify and for which a solution could still be evaluated, either using an open local test or by submitting to an available competition platform. The preserved task catalog contains 55 such tasks. We found publicly available solutions for 37 of them, which formed the pool for further selection.

We selected tasks using two criteria. First, we manually examined the reference solutions and excluded tasks whose solutions relied only on standard ML methods and routine tuning, without task-specific insights. Second, we ran a Codex agent with explicit instructions to use straightforward approaches without detailed data exploration or attempts to discover task-specific structure. We imposed no time limit or cap on submission attempts during this screening. We defined strong performance as placing in the top three on the original private leaderboard. When the agent reached this threshold, we manually inspected its solution to verify that it did not rely on task-specific insights before excluding the task.

A task was retained only if its reference solution contained a task-specific insight and the screening agent did not reach the top three with a straightforward solution. This procedure yielded 20 tasks for the core benchmark.

Table 1: Representative benchmark tasks and the task-specific structure that makes a standard pipeline insufficient. The examples illustrate different forms of strategy discovery

Task	Why a standard approach is limited	Structure to discover
Sticky Note Blindness	Object predictions are strongly affected by text written on an attached sticky note.	Suppress or disrupt the misleading textual signal while preserving the visual appearance of the object.
Drawn Apart	Labeled examples and test examples come from different visual domains, including photographs, cartoons, and sketches.	Find a representation or transfer mechanism that remains useful across these domains.
Broken BERT	A standard pretrained language model cannot be used directly because part of its word representation has been corrupted.	Identify the damaged components and reconstruct them using the remaining model structure and vocabulary.
Classifier Classifier	The objects being classified are themselves trained image classifiers.	Construct informative probe inputs whose responses reveal which class each model has learned.
Lost Interpreter	The semantics of the available program instructions are unknown.	Infer instruction behavior from observed programs and their outputs, then execute new programs correctly.

The selected tasks differ not only in modality but also in the type of task-specific structure that a solver must identify. Table 1 illustrates several representative forms of such strategy discovery across the benchmark.

Across the 20-task core set, the data include image and image-model problems, audio, text, multivariate time series, and program execution, while the underlying challenges include suppressing spurious signals, transferring across domains, repairing model components, reasoning under missing classes, constructing informative inputs, and reverse-engineering unknown transformations.

For each selected task, we identify a reference solution that ranks in the top three on the private leaderboard. These solutions are used to construct the two guidance conditions described below. In the guided conditions, the agent is explicitly told that the provided direction comes from a strong solution. The agent is nevertheless free to modify the described approach or follow an alternative strategy if its experiments suggest doing so.

4 Evaluation Protocol

4.1 Environment and resource allocation

We aim to place the agent under conditions close to those faced by the original competitors. At the start of each run, the workspace contains the materials available to competitors, including the task data and any supplied baseline code. The agent runs in an isolated environment without internet access. Outbound access is blocked, and we monitor attempts to leave the environment.

Required libraries and pretrained weights are installed locally before the task begins. For rounds that already prohibited internet access, we mirror the software and model resources available to competitors. For rounds that permitted internet access, we inspect a solution placing in the top three and make the libraries and model weights it uses available locally. We also include additional relevant libraries and weights so that the installed resources do not reveal which approach the reference solution took. Thus, the agent can use the necessary tools without receiving an implicit hint through a uniquely provisioned package.

We allocate time according to the original competition format. For a task held as a standalone competition problem, the agent receives the competition's time limit. When we select one task from a round containing multiple tasks, we divide the round's total time equally among its tasks.

We also approximate the computational resources available in the original round. Tasks run on CPU alone when competitors had CPU-only access. When competitors could use GPUs such as an L4 or RTX 6000, we allocate V100 GPUs in a configuration chosen to provide approximately comparable available GPU memory. We also tested whether timeouts were a major bottleneck on these tasks and found that they did not materially limit the agent's performance in the evaluated runs. Full details are provided in Appendix E.1.

4.2 INFORMATION CONDITIONS

Within this environment, we run each task under three levels of guidance. The time limit, computational allocation, task data, and evaluation procedure are the same across conditions for a given task. The conditions differ in the information provided about how to solve the task, with both guided conditions derived from the same reference solution ranked in the top three on the private leaderboard.

Table 2: The three information conditions used in the evaluation. For each task, both HINT and SOLUTION are based on the same reference solution, ranked in the top three on the private leaderboard. HINT provides the main promising direction for solving the task, while SOLUTION describes the reference approach in detail, leaving some practical choices to the agent.

Condition	Information given to the agent	What the agent still needs to do
NONE	No additional guidance.	Find an approach, implement it, run experiments, and submit.
HINT	The main promising direction from the reference solution.	Develop the idea, implement it, and evaluate the result.
SOLUTION	A detailed textual description of the reference solution.	Implement, debug, make remaining practical choices, and submit.

In SOLUTION, the agent receives a detailed description of a successful approach, including its main components and practical recommendations. This represents a setting in which a person explains an approach and delegates the remaining work to the agent. The agent is explicitly told that the guidance comes from a solution ranked in the top three on the private leaderboard. But it is free to adapt the suggested approach or choose an alternative if its experiments indicate that doing so could improve performance.

4.3 AGENT EXECUTION AND SUBMISSION PROTOCOL

For each evaluated agent configuration, we perform one run per task and information condition. At the start of each run, the agent receives the task instructions and, in the guided conditions, the corresponding hint or solution description. Each run starts from a clean state, with no access to previous attempts.

All runs are fully autonomous and end-to-end. Each agent uses the tools provided by its execution framework to inspect data, write and edit code, train models, and analyze experimental results. It controls the sequence of experiments and revisions without human intervention. During a run, the agent can inspect execution logs, error messages, and locally computed validation metrics.

The agent can request public evaluation of a candidate file or directory through the submission tool. Each run allows up to 19 interactive submissions and one reserved final evaluation. Public feedback helps the agent compare candidates and revise its approach, while private scores remain unavailable. After using all interactive submissions, the agent can continue training, testing locally, and updating its final artifact within the remaining time.

After the agent exits or reaches the deadline, the harness evaluates the artifact at the designated final path using the reserved attempt. This artifact determines the reported result, even if it differs from the last interactively evaluated candidate. Agents are informed of this rule. To select an earlier candidate, they must restore it to the final path. Waiting for interactive evaluation consumes the task's time budget, whereas the reserved final evaluation takes place after the run ends. The final artifact is evaluated on the private test set and scored as described in Section 4.5.

All reported runs used environments with the required libraries installed, following correction of an earlier dependency omission. Checks of workspace setup and agent invocation identified no technical errors in the reported runs.

4.4 HUMAN EVALUATION

We evaluated humans only in SOLUTION, primarily to assess whether experienced participants could use the solution descriptions to implement successful solutions under the evaluation conditions. This comparison provides evidence of the descriptions' validity and practical usefulness. Extending the comparison to NONE and HINT would require additional groups closely matched in experience and ability, which are very difficult to assemble in practice.

We recruited 23 participants who actively competed in national AI olympiad qualifying rounds, targeting the middle of the participant pool and excluding those in the top 30%. Their experience varied, but all had a working knowledge of machine learning. This selection was intended to assess whether the descriptions were useful to competent participants beyond the strongest competitors. Before participation, we asked each person about prior exposure to the benchmark tasks and their solutions and checked their Kaggle account histories for evidence of participation in the corresponding competitions.

Participants received the same textual solution descriptions and task materials as the agents in SOLUTION condition, with the same computational allocations, submission limits, and public feedback. They worked individually under proctoring and were not permitted to use LLM tools. Each task had the same elapsed-time limit as in the agent evaluation. This limit covered the entire task window, and including breaks. Final submissions were evaluated on the same private test sets using the same scoring procedure as for the agents.

4.5 MEASUREMENTS AND INTERPRETATION

The original tasks use different evaluation metrics. We first orient each raw metric so that larger values indicate better performance. For a given agent configuration, let $q_{i,c}$ be its private evaluation value on task i under condition c, and let b_i be the task's baseline value. The upper reference value is

$$u_i = \begin{cases} \max\{q_i^{\text{author}}, q_i^{\text{best}}\}, & \text{if an author solution is available,} \\ q_i^{\text{best}}, & \text{otherwise,} \end{cases} \tag{1}$$

where q_i^{best} is the best historical participant result. We convert each result to points using

$$s_{i,c} = \text{clip}_{[0,100]}\left(100\,\frac{q_{i,c} - b_i}{u_i - b_i}\right). \tag{2}$$

The baseline thus receives 0 points, while the stronger of the author and participant solutions receives 100 points when both are available. When no author solution exists, the best participant result defines 100 points. This follows the baseline-to-leading-solution scale used to approximate scoring in the original international competitions and their qualifying rounds. The transformation is defined for tasks with $u_i > b_i$.

We apply the same normalization to each human submission. For each task, we average the normalized points across the 23 participants. The overall human score is the equally weighted mean of these 20 task-level averages.

We also compare each selected submission with its task's historical private leaderboard and report its virtual rank. Mean and median ranks summarize these comparisons, while top-k counts show how often submissions rank among the top k entries. These counts are nested, so a first-place result contributes to Top-1, Top-3, and Top-10. Because leaderboards differ in size and participant strength, equal ranks across tasks need not indicate equal levels of performance.

Finally, we compare the agent's SOLUTION result with the recruited participants' performance under the same information condition. If $\bar{s}_{H,\text{SOLUTION}}$ denotes the human mean and $\bar{s}_{M,\text{SOLUTION}}$ the agent mean, we report

$$G_{H,M}^{\text{SOLUTION}} = \bar{s}_{H,\text{SOLUTION}} - \bar{s}_{M,\text{SOLUTION}}. \tag{3}$$

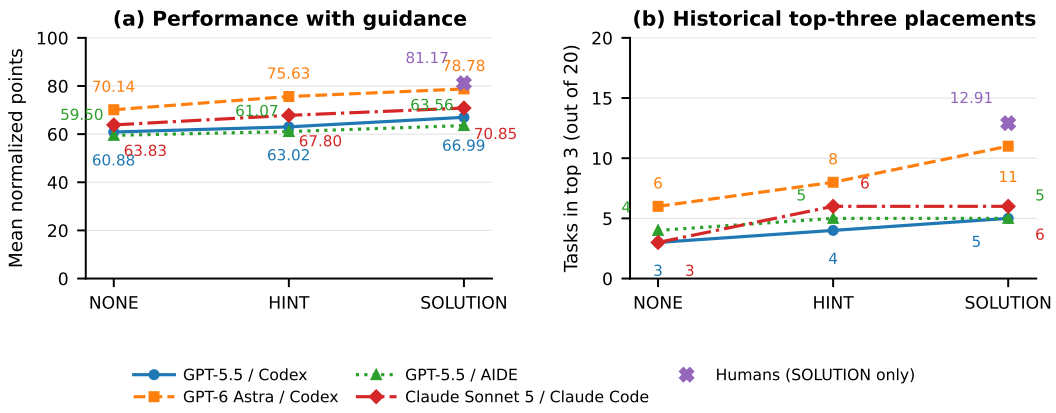


Figure 2: Effect of guidance on (a) mean normalized points and (b) the number of tasks with a top-three historical placement. Human markers appear only in SOLUTION, the condition in which participants were evaluated. Lines connect aggregate results across information conditions.

5 Results and Analysis

We examine how solution guidance affects agent performance, compare the results with experienced human participants, and analyze the difficulties that remain when a promising approach is supplied. Table 3 lists the four agent configurations in the evaluation.

5.1 Overall performance and the effect of guidance

Table 3: Performance on the 20 benchmark tasks. Reasoning settings and execution frameworks are included in the agent names. Mean points use the normalization in Section 4.5. Mean virtual rank and top-k counts refer to the historical private leaderboards. Human results are reported only for SOLUTION.

Agent	Condition	Mean points ↑	Mean rank ↓	Top-1	Top-3	Top-10
GPT-5.5 (high), Codex	NONE	60.88	25.50	1	3	11
	HINT	63.02	13.75	1	4	11
	SOLUTION	66.99	11.05	3	5	12
GPT-6 Astra (medium), Codex	NONE	70.14	18.50	3	6	13
	HINT	75.63	11.10	4	8	15
	SOLUTION	78.78	8.00	6	11	17
GPT-5.5 (high), AIDE	NONE	59.50	25.55	1	4	11
	HINT	61.07	18.50	2	5	11
	SOLUTION	63.56	18.20	3	5	12
Claude Sonnet 5 (medium), Claude Code	NONE	63.83	21.85	2	3	11
	HINT	67.80	12.05	3	6	13
	SOLUTION	70.85	9.50	2	6	15
Humans	SOLUTION	81.17	6.09	6.04	12.91	18.04

Guidance improves all completed systems. GPT-5.5 gains 6.11 normalized points from NONE to SOLUTION, while GPT-6 Astra gains 8.64. Astra also benefits more from the shorter HINT condition, suggesting that it can exploit partial strategic guidance more effectively.

GPT-6 Astra exceeds GPT-5.5 in mean points under every condition, and its unguided score is higher than GPT-5.5's score with a full description. The relative gains from SOLUTION are comparable in

magnitude, at approximately 10.04% for GPT-5.5 and 12.32% for GPT-6 Astra. The short hint produces a larger relative gain for Astra, at 7.83% compared with 3.52% for GPT-5.5. Thus, all systems continue to benefit from the supplied approach, with Astra obtaining more of its total improvement from the short hint.

The historical leaderboard comparisons show that these gains also improve competitiveness. Between NONE and SOLUTION, GPT-5.5's top-three count increases from three to five tasks, while GPT-6 Astra's increases from six to eleven. However, all agents remain outside the top three on a substantial part of the benchmark even when given a description derived from a top-three solution. The remaining difficulty therefore includes turning the supplied approach into a competitive implementation and selecting a successful final submission. We examine these steps through the qualitative analysis below.

5.2 HUMAN PERFORMANCE WITH SOLUTION DESCRIPTIONS

The 23 human participants each completed all 20 tasks in SOLUTION, obtaining an overall mean of 81 points. They received the same textual descriptions as the agents and worked under the conditions specified in Section 4.4.

Figure 2 places this reference alongside the agent results. The human mean exceeds GPT-5.5's SOLUTION score by approximately 14.18 points and GPT-6 Astra's by 2.39 points. GPT-6 Astra therefore approaches the cohort's average performance on this measure, although an aggregate comparison does not establish whether the same tasks remain difficult for humans and agents. Since participants were evaluated only in SOLUTION, the experiment does not estimate how much the descriptions improve human performance relative to independent problem solving.

The human results also support the practical usefulness of the SOLUTION descriptions. Participants achieved competitive results with the same guidance, task materials, resource allocations, and feedback rules as the agents. These results show that the supplied descriptions provide a usable basis for developing successful solutions under the benchmark conditions.

5.3 QUALITATIVE ANALYSIS OF SOLUTION GUIDANCE

To examine how agents use the supplied guidance, we inspected GPT-5.5 runs whose final submissions fell outside the top three on the private leaderboard. AI-assisted summaries identified relevant events in the execution logs, which we then examined manually. The rank threshold was used only to select runs for inspection and was not itself treated as evidence of an error. We examined the selected methods, their implementation, validation experiments, and final submission decisions. The observations below describe recurring patterns in the inspected runs; several patterns can occur within the same run.

Under NONE, several weak runs began with an approach that simplified away an important part of the task. Handcrafted features, small models, nearest-neighbor transfer, or template matching provided limited representations of the structure needed for a competitive solution. Subsequent implementation effort then refined this initially limited strategy.

Under HINT, the agent more often started in a useful direction, but public feedback or local validation could remain misleading. These checks sometimes encouraged further refinement of candidates that did not transfer reliably to private evaluation.

Under SOLUTION, the description more often led to an appropriate overall architecture. Difficulties remained in training stability, threshold selection, ensemble composition, postprocessing, and final candidate selection. Some runs also abandoned a recommended component after a weak implementation or an inconclusive experiment and continued with a simpler fallback. The two cases below illustrate this change of direction and the omission of a central modeling mechanism. Appendix D provides the corresponding recommendations, experiments, and final results.

In Brain Tumors, the agent tested a CNN trained with pseudo-labels, following the suggested semi-supervised direction. When this candidate scored below its existing RandomForest on public evaluation, it abandoned the CNN and continued improving the feature-based pipeline. The available feedback supported this choice among the tested candidates, but the run ended without developing the suggested neural approach into a competitive alternative.

In Visual Relations, the guidance described a model for predicting relations between directed pairs of objects. The executed pipeline instead learned image-level labels and transferred relation annotations from similar training images. Further training reduced the training loss without improving the public score of the transfer pipeline. The final solution therefore differed from the main mechanism described in the guidance, despite using visual features and relation predictions.

Across the inspected runs, difficulties arose both in constructing a competitive candidate and in selecting the final submission. An agent could adopt the supplied approach without developing an effective implementation, or its validation procedure could fail to identify which candidate to submit. Guidance helped establish a useful direction while these practical decisions remained unresolved. The two detailed cases also show why assessing the use of guidance requires examining the executed solution and its development history. The final implementation may omit or approximate an important component of the recommendation.

An accompanying analysis of GPT-5.5's stopping behavior found no runs ending because of the time limit across the three conditions. Runs instead ended after the submission allowance was exhausted, public performance plateaued, the local plan was completed, or the agent considered the task solved. Appendix E provides the breakdown. These observations describe how the runs ended.

6 Conclusion

We introduced LUDKA Bench, a collection of 20 nonstandard machine learning tasks from AI olympiads and national qualifying competitions. The benchmark evaluates whether agents can develop competitive solutions independently or with guidance supplied at the start of the task. Three conditions, ranging from no guidance to a detailed description of a successful approach, allow us to examine how effectively agents use this information under fixed time and resource constraints.

All evaluated agents benefit from guidance. Detailed solution descriptions raise mean normalized scores from 60.88 to 66.99 for GPT-5.5 and from 70.14 to 78.78 for GPT-6 Astra. Astra approaches the human cohort's mean of 81 points under the same guidance condition, although it remains outside the historical top three on nine of the twenty tasks. Analysis of GPT-5.5 trajectories shows that incomplete implementation, unstable training, unreliable validation, and selection of a weaker final submission remain obstacles even when an approach is provided. Some runs also abandon promising approaches before resolving implementation problems.

The results show that guidance improves performance, but even detailed recommendations based on a successful solution often leave agents below that solution's level. Knowing a promising approach does not consistently translate into a competitive result without further human assistance. LUDKA provides a setting for studying both independent problem solving and the ability to turn supplied guidance into effective solutions.

AI use statement

We used generative AI tools to assist with drafting and revising the manuscript, conducting preliminary qualitative analysis of agent execution traces and logs, and writing portions of the experimental code. For the trace analysis, AI tools summarized runs and highlighted key decisions, implementation issues, and experimental outcomes for subsequent human inspection. We checked these observations against the original traces and logs and made the final judgments about the reported failure patterns and their interpretation. The autonomous agents evaluated in the benchmark are described separately in the evaluation protocol. We take responsibility for the final content of the paper, including text, claims, analyses, and code produced with the assistance of generative AI.

Ethics statement

LUDKA Bench evaluates autonomous ML agents on tasks drawn from existing AI competitions. Agent runs take place in isolated environments without internet access, and private evaluation scores are withheld from the agents during execution. The human evaluation involves competition-experienced participants and is intended to assess whether the supplied solution descriptions support successful implementation under the benchmark conditions. We report aggregate human performance and interpret it as a reference for this specific cohort and setting, without generalizing to

human ability more broadly. The use and redistribution of competition tasks, datasets, and reference materials remain subject to their original licenses and permissions.

REPRODUCIBILITY STATEMENT

The supplementary materials include the experimental code, all task statements and guidance texts, task-specific baselines, and the reference values used for score normalization. We also provide scripts for obtaining the data, preparing the execution environments and locally available pretrained weights, and evaluating submissions on the public and private test sets. Section 4.1 describes the execution environment, available resources, and allocation of time and compute. Sections 4.2 and 4.3 specify the information conditions, independent-run procedure, accessible feedback, submission limit, and final-submission selection rule. Section 4.5 defines score normalization and aggregation, while Section 4.4 describes the human evaluation protocol. Appendix C provides examples of the strategic hints and solution descriptions used to illustrate the guidance conditions.

REFERENCES

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/7e3767db483c942b883eb4f8cfb74e31-Abstract-Conference.html.

Shengda Fan, Xuyan Ye, Yupeng Huo, Zhi-Yuan Chen, Yiju Guo, Shenzhi Yang, Wenkai Yang, Shuqi Ye, Jingwen Chen, Haotian Chen, Xin Cong, and Yankai Lin. AgentProcessBench: Diagnosing step-level process quality in tool-using agents. *arXiv preprint arXiv:2603.14465*, 2026. URL https://arxiv.org/abs/2603.14465.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20271–20309. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24y.html.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. AIDE: AI-driven exploration in the space of code. *arXiv preprint arXiv:2502.13138*, 2025. doi: 10.48550/arXiv.2502.13138. URL https://arxiv.org/abs/2502.13138.

Patrick Tser Jern Kon, Qiuyi Ding, Jiachen Liu, Xinyi Zhu, Jingjia Peng, Jiarong Xing, Yibo Huang, Yiming Qiu, Jayanth Srinivasa, Myungjin Lee, Mosharaf Chowdhury, Matei Zaharia, and Ang Chen. EXP-Bench: Can AI conduct AI research experiments? In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/c411f5b2d9c55f1685e72db224ad8b0e-Abstract-Conference.html.

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. URL https://arxiv.org/abs/2408.06292.

Tianyue Ou, Wanyao Guo, Apurva Gandhi, Graham Neubig, and Xiang Yue. AgentDiagnose: An open toolkit for diagnosing LLM agent trajectories. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 207–215, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-demos.15. URL https://aclanthology.org/2025.emnlp-demos.15/.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's ability to replicate AI research. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 56843–56873. PMLR, 2025. URL https://proceedings.mlr.press/v267/starace25a.html.

Haoyu Sun, Wenxuan Wang, Mingyang Song, Jujie He, Weinan Zhang, Yang Liu, Yang Yang, and Yu Cheng. Agent planning benchmark: A diagnostic framework for planning capabilities in LLM agents. *arXiv preprint arXiv:2606.04874*, 2026. URL https://arxiv.org/abs/2606.04874.

Edan Toledo, Karen Hambardzumyan, Martin Josifoski, Rishi Hazra, Nicolas Baldwin, Alexis Audran-Reiss, Michael Kuchnik, Despoina Magka, Minqi Jiang, Alisia Lupidi, Andrei Lupu, Roberta Raileanu, Kelvin Niu, Tatiana Shavrina, Jean-Christophe Gagnon-Audet, Michael Shvartsman, Shagun Sodhani, Alexander Miller, Abhishek Charnalia, Derek Dunfield, Carole-Jean Wu, Pontus Stenetorp, Nicola Cancedda, Jakob Foerster, and Yoram Bachrach. AI research agents for machine learning: Search, exploration, and generalization in MLE-bench. In *Advances in Neural Information Processing Systems*, volume 38, pp. 39717–39756, 2025. doi: 10.52202/085713-1186. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/328b81881da145412f2bc56c998dfb6a-Abstract-Conference.html.

Hjalmar Wijk, Tao Roa Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, Lawrence Chan, Michael Chen, Joshua M Clymer, Jai Dhyani, Elena Ericheva, Katharyn Garcia, Brian Goodrich, Nikola Jurkovic, Megan Kinniment, Aron Lajko, Seraphina Nix, Lucas Jun Koba Sato, William Saunders, Maksym Taran, Ben West, and Elizabeth Barnes. RE-Bench: Evaluating frontier AI r&d capabilities of language model agents against human experts. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 66772–66832. PMLR, 2025. URL https://proceedings.mlr.press/v267/wijk25a.html.

Yunxiang Zhang, Muhammad Khalifa, Shitanshu Bhushan, Grant Murphy, Lajanugen Logeswaran, Jaekyeom Kim, Moontae Lee, Honglak Lee, and Lu Wang. MLRC-Bench: Can language agents solve machine learning research challenges? In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3036. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/82c96f3c90741ef2c9b248e65d9b5db0-Abstract-Datasets_and_Benchmarks_Track.html.

Yunxiang Zhang, Kang Zhou, Zhichao Xu, Kiran Ramnath, Yun Zhou, Sangmin Woo, Haibo Ding, and Lin Lee Cheong. Learning to ideate for machine learning engineering agents. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 436–447, Rabat, Morocco, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.eacl-short.32. URL https://aclanthology.org/2026.eacl-short.32/.

Guangxiang Zhao, Qilong Shi, Xusen Xiao, Wenpu Liu, Yaoming Li, Linfeng Hao, Shuyang Hou, Zijian Guo, Xinrui Zhang, Yuntian Zhao, Zhengyang Wang, Wenrui Liu, Yuhan Wu, Tong Yang, Lin Sun, and Xiangzheng Zhang. ScienceArena: Benchmarking LLMs on latest scientific olympiad competitions. *arXiv preprint arXiv:2608.30517*, 2026. URL https://arxiv.org/abs/2608.30517.

Table 4: Agent configurations included in the study.

Model	Reasoning setting	Framework
GPT-5.5	High	Codex
GPT-6 Astra	Medium	Codex
GPT-5.5	High	AIDE
Claude Sonnet 5	Medium	Claude Code

A Agent Configurations and Execution Details

Table 4 lists the evaluated combinations of model, reasoning setting, and execution framework. We performed one run for each task, assistance condition, and agent configuration. The common evaluation protocol is described in Section 4.3. Task-specific GPU allocations and time limits are listed in Appendix E.1.

Framework settings and initial instructions. Each framework retained its native tools and capabilities within the task environment. Multi-step execution and context management were handled by the framework itself. We imposed no additional limits on token usage, model calls, or cost. For AIDE, the search configuration used `steps=20` and `parent_selection=best_scored`. This search-step setting is separate from the public submission budget described below.

The harness generated a starting prompt for each task. The task instructions and, where applicable, the hint or solution description were included directly in this prompt. It also specified the required output path, the submission command, and the division of the evaluation budget into interactive attempts and a reserved final attempt.

Submission interface. To request public evaluation, the agent first wrote its candidate file or directory to the task-specific path `submission/<name>` and then executed `./submit` without arguments. The command contacted a local broker through a Unix socket. The broker copied the current artifact, validated it, and requested its public evaluation. Each request therefore evaluated a copy of the candidate present at the time of submission.

The command returned a JSON response containing `public_score`, the attempt number (`attempt_index`), the number of remaining interactive attempts (`remaining_interactive`), and an indicator of the reserved final attempt. The `metric` field contained the evaluation status and, when available, a submission status or identifier, or an error message. Private scores were not included in the response.

Attempt accounting and final artifact. The evaluation budget allowed up to 19 interactive submissions and one reserved final submission. Once the interactive budget was exhausted, further calls to `./submit` returned an error, but the agent could continue training, testing locally, and modifying its output until it finished or reached the time limit. Exhausting the interactive budget did not terminate the run.

After the agent exited or was stopped at the deadline, the harness evaluated the artifact at the designated final path using the reserved attempt. This artifact determined the reported result, even if it differed from the last interactively evaluated candidate. To select an earlier candidate, the agent therefore had to restore it to the final path before the run ended. Another interactive submission was not required.

Missing artifacts, local validation failures, and disallowed links did not consume an attempt. The counter advanced only after the evaluation handler returned successfully. However, an upload accepted by the competition platform and subsequently marked as erroneous still counted as an official attempt.

Timing and execution checks. Waiting for an interactive submission response consumed the agent's allotted runtime. Evaluation of the final artifact took place after the agent stopped and did not consume time available for solution development. No technical failures occurred during the final

Table 5: Detailed GPT-5.5 aggregates across the 20 benchmark tasks.

Metric	NONE	HINT	SOLUTION
Number of tasks	20	20	20
Sum of points	1217.50	1260.50	1339.72
Mean points	60.88	63.02	66.99
Sum of ranks	510	275	221
Mean virtual rank	25.50	13.75	11.05
Median virtual rank	9.00	6.50	6.00
Top-1	1	1	3
Top-3	3	4	5
Top-10	11	11	12

Table 6: GPT-6 Astra aggregates across the 20 benchmark tasks.

Metric	NONE	HINT	SOLUTION
Number of tasks	20	20	20
Sum of points	1402.80	1512.60	1575.60
Mean points	70.14	75.63	78.78
Sum of ranks	370	222	160
Mean virtual rank	18.50	11.10	8.00
Median virtual rank	7.0	4.0	3.0
Top-1	3	4	6
Top-3	6	8	11
Top-10	13	15	17

experimental runs. The experimental code included in the supplementary materials implements the submission interface, attempt accounting, and final-artifact evaluation.

B SCORE AGGREGATION AND NUMERICAL PRECISION

B.1 NORMALIZED POINTS AND HISTORICAL RANKS

The benchmark combines tasks with different original metrics. The normalization in Section 4.5 maps each task's baseline to zero points and its upper reference to 100 points, after orienting the raw metric so that larger values indicate better performance. When an author solution is available, the upper reference is the stronger of that solution and the best historical participant result. Otherwise, the best historical participant result defines the upper reference.

Scores are clipped to the interval $[0, 100]$ and averaged with equal weight across the 20 tasks. Thus, mean points measure performance relative to task-specific reference values. They are neither classification accuracy nor the percentage of tasks solved. Upper clipping also means that improvements beyond a task's upper reference do not increase its contribution to the aggregate.

Virtual ranks provide a complementary comparison against the original private competition leaderboards. Each evaluated submission is compared with the historical entries for its task. The recruited human participants are not added to those leaderboards. Top-k counts are nested, so a first-place result contributes to Top-1, Top-3, and Top-10. Because the leaderboards differ in size and participant strength, equal ranks across tasks need not represent equal levels of difficulty.

B.2 DETAILED GPT-5.5 AND GPT-6 ASTRA AGGREGATES

Table 5 reports the GPT-5.5 aggregates at additional numerical precision. Mean points and mean ranks in the main results table are rounded to two decimal places. The median rank is included here to complement the mean, which is more sensitive to tasks with low leaderboard placements.

Table 7: Guidance effects calculated from the displayed aggregate means.

Agent	Comparison	Point change	Relative point change (%)	Mean-rank change
GPT-5.5	HINT vs. NONE	2.14	3.52	−11.75
	SOLUTION vs. NONE	6.11	10.04	−14.45
GPT-6 Astra	HINT vs. NONE	5.49	7.83	−7.40
	SOLUTION vs. NONE	8.64	12.32	−10.50

B.3 ABSOLUTE AND RELATIVE EFFECTS OF GUIDANCE

Absolute changes measure the additional normalized points obtained with guidance. Relative changes express that difference as a fraction of the unguided aggregate:

$$R_c = 100 \left(\frac{\bar{s}_c}{\bar{s}_{\text{NONE}}} - 1 \right), \qquad c \in \{\text{HINT}, \text{SOLUTION}\}. \tag{4}$$

This is a ratio of aggregate means, rather than an average of task-level percentage changes. The latter would be undefined for tasks with a zero-point unguided result and unstable for results close to zero.

Table 7 uses the rounded means displayed in the main results table for gpt-5.5 and gpt-6 astra models. A negative change in mean rank indicates an improvement in leaderboard placement.

GPT-6 Astra achieves a higher score without guidance and gains 12.32% under SOLUTION, compared with 10.04% for GPT-5.5. Under HINT, the relative improvement is also larger for Astra, at 7.83% versus 3.52%. Thus, the configuration with stronger unguided performance also shows larger relative gains from guidance in these comparisons. These results apply to the evaluated configurations and do not establish a relationship between parameter count and the effectiveness of guidance.

C EXAMPLE ASSISTANCE PROMPTS

The following English renderings illustrate the assistance provided for a geolocation task involving panoramic screenshots. The examples were originally written in Russian. They show the difference in strategic detail between HINT and SOLUTION; NONE supplies the task without this additional guidance.

Strategic hint.

> Use geographical image features rather than ordinary CNN fine-tuning: GeoCLIP with multiple panorama crops provides a strong baseline. Then refine the result through careful near-duplicate corrections and limited hill climbing over labels.

Full textual solution.

> Treat the task as geolocation from panoramic screenshots rather than ordinary image classification. First inventory the training and test sets, the country list, and the image regions: the central panorama, side fragments, lower interface, minimap, and interface text. The main approach is a geographical image encoder such as GeoCLIP with its parameters held fixed, a lightweight linear head over countries, multi-crop inference, and averaging logits across regions. Use pretrained models or GeoCLIP only if the package and weights are already available locally; otherwise build a stable stack from thumbnail and color-texture features, OCR cues from the interface, nearest neighbors, and regularized tabular models. After obtaining model probabilities, search for perceptual near-duplicates between training and test images and apply them as conservative postprocessing. Pitfalls: an ordinary weak CNN without transfer features is ineffective; manual public-leaderboard hill climbing overfits; country labels and test IDs must match exactly.

The hint identifies a promising representation and a small number of task-specific refinements. The full description also explains how to combine the representation with a prediction head, which image regions may contain useful information, and which implementation choices can undermine the

result. It still leaves practical decisions to the agent, including crop selection, regularization, aggregation weights, and thresholds for duplicate matching.

These decisions are part of the intended evaluation. Under SOLUTION, the agent must turn a reference approach into working code, choose suitable details, and diagnose weak results without unnecessarily discarding the useful mechanism. Consequently, the assistance conditions influence several stages of solution development.

References to public-score refinement remain subject to the common limit of 20 submissions per run. The agent never receives private labels or private evaluation feedback. The fallback described in the example is conditional on locally available assets and does not change the environment between assistance conditions.

D QUALITATIVE ANALYSIS OF AGENT SOLUTIONS

D.1 ANALYSIS PROCEDURE

We inspected GPT-5.5 runs whose final submissions fell outside the historical top three. AI-assisted summaries helped locate relevant decisions and experiments, followed by manual examination of the saved prompts, execution logs, submission records, and final results. The two cases below use GPT-5.5 with high reasoning in Codex under SOLUTION. The guidance was checked against the prompt saved for each run.

The cases were selected to illustrate how the supplied approach was tested, modified, or replaced. They are not a random sample or an exhaustive classification of failures. All scores below are raw task metrics, with larger values indicating better performance. Public scores were available to the agent during the run; private scores were available only for the final submission and were used retrospectively. Successful format validation does not imply that prediction quality was evaluated on a local holdout set.

D.2 BRAIN TUMORS: ABANDONING A CNN AFTER WEAK PUBLIC FEEDBACK

The guidance recommended using the small labeled set together with additional unlabeled images. It suggested a pretrained ResNet when weights were available locally, or a compact CNN with FixMatch or self-training. The agent first developed a RandomForest using handcrafted image features, reaching a public Macro-F1 of 0.56801. It then trained a CNN on 48 labeled images and 511 additional images assigned pseudo-labels by the RandomForest.

The CNN's training loss decreased from 0.10818 at epoch 10 to 0.03468 at epoch 20, but its public score was 0.49399, below the existing RandomForest. The agent explicitly rejected this candidate: "CNN scored 0.49399, so it's out." It returned to the RandomForest pipeline, adding thumbnail, local binary pattern, and Gabor features. The final submission obtained a public score of 0.63292 and a private score of 0.64927.

Table 8: Observed development steps in Brain Tumors under SOLUTION. Private scores for the intermediate candidates were unavailable.

Step	Candidate	Public	Private
Initial pipeline	RandomForest with handcrafted image features	0.56801	—
Neural attempt	CNN trained with labels and RandomForest pseudo-labels	0.49399	—
Final submission	RandomForest with expanded image features and adjusted parameters	0.63292	0.64927

This trajectory documents a switch away from the suggested neural approach after an unsuccessful experiment. The logs establish the use of a CNN and pseudo-labels, but do not establish a complete implementation of FixMatch.

D.3 VISUAL RELATIONS: IMAGE-LEVEL TRANSFER INSTEAD OF PAIR-LEVEL MODELING

The guidance described a relation detector operating on directed subject–object pairs. Its main components included features for both objects and their union box, relative geometry, and a relation head trained with multi-label binary cross-entropy and negative pair sampling. Annotation transfer was permitted as a fallback.

The executed pipeline used a different prediction unit. A ResNet18 was trained on 4,000 images to predict 103 image-level predicate and object-presence labels. Relation annotations, including bounding boxes, were transferred from similar training images, and image-level predictions were used to rerank the transferred relations. Table 9 contrasts the main recommendation with the implementation documented in the executed commands and training logs.

Table 9: The recommended and executed approaches in Visual Relations under SOLUTION. The comparison concerns the main modeling choices documented in the saved trace.

Component	Recommended approach	Observed implementation
Prediction unit	A directed pair of objects in the current image.	Predicate and object-presence labels for the whole image.
Representation	Subject, object, and union-box features with relative geometry.	Image-level CNN features combined with handcrafted features for retrieval.
Relation prediction	A learned relation head applied to object pairs.	Transfer of relation annotations from neighboring training images, followed by reranking.

After 18 epochs, the training loss had decreased from 0.74243 to 0.45090, and the resulting CNN-based transfer pipeline obtained a public score of 0.0058. Further training reduced the loss to 0.21505, while the public score fell to 0.0054. The agent then returned to the earlier checkpoint, combined its features with handcrafted features, and adjusted the transfer and reranking procedure. The final valid submission contained 91,824 relations across 1,000 images and obtained a public score of 0.0098 and a private score of 0.0200.

E RUN TERMINATION AND TIME BUDGETS

We inspected run termination to assess whether the time limit directly interrupted solution development. Table 10 reports the stopping reasons for GPT-5.5. The distribution differed across the three assistance conditions.

The submission limit was reached in 3 of the 20 runs under NONE, 10 under HINT, and 8 under SOLUTION, corresponding to 15%, 50%, and 40%, respectively. Across all three conditions, the public evaluation budget therefore directly constrained further feedback-driven experimentation in 21 of 60 runs (35%). Public-score plateaus accounted for 12, 5, and 7 runs under NONE, HINT, and SOLUTION, respectively. The corresponding counts were 3, 2, and 2 for completion of the agent's local plan, and 2, 3, and 3 for a maximum public score or agent-declared completion.

These categories describe why the agent stopped, rather than whether the resulting solution was optimal. In particular, a maximum public score or an agent's completion judgment does not establish that the private evaluation is solved. Similarly, a public-score plateau may reflect the limits of the current approach or of the feedback available to guide further changes.

No GPT-5.5 run ended through direct interruption by the time limit. Inspection of the GPT-6 Astra trajectories was also consistent with the time allowance being sufficient for agents to execute their chosen approaches. The observed termination behavior therefore does not support direct timeout interruption as the main explanation for the remaining performance gaps.

This conclusion concerns behavior under the evaluated protocol. It does not establish that additional runtime or more public submissions would have no value. A run that stops after completing its plan may still benefit from a different search strategy, and a run that exhausts its submissions may

Table 10: Immediate stopping reasons for GPT-5.5 runs. Each column covers the 20 benchmark tasks.

Stopping reason	NONE	HINT	SOLUTION
Public submission limit reached	3	10	8
Public-score plateau	12	5	7
Local plan completed	3	2	2
Maximum public score or agent-declared completion	2	3	3
Direct interruption by the time limit	0	0	0
Total	20	20	20

Table 11: GPU allocations and time limits for the 20 benchmark tasks. GPU counts refer to NVIDIA V100 devices.

#	Task	V100 GPUs	Time limit (hours)
1	`iaio_2026_sfr_image_classification`	1	16.00
2	`iaio_2026_sfr_timeseries_classification`	0	16.00
3	`aicc_round_2_audio_demixing`	1	24.00
4	`aicc_round_4_sticky_note_blindness`	0	24.02
5	`neoai_2025_underfitting_cv`	1	27.00
6	`neoai_2026_day_2_audio`	0	3.63
7	`aicc_round_3_drawn_apart`	1	48.00
8	`aicc_round_5_visual_relations`	1	24.00
9	`aicc_round_2_face_matching`	0	24.00
10	`neoai_2025_broken_bert`	0	27.00
11	`aicc_round_6_nuclei_reconstruction`	1	31.01
12	`aicc_round_6_classifier_classifier`	1	31.01
13	`aicc_round_7_oriented_ship`	1	48.04
14	`neoai_2026_day_2_cv`	1	3.63
15	`aicc_round_0_brain_tumors`	1	24.00
16	`aicc_round_4_extreme_condensation`	1	24.02
17	`neoai_2026_day_1_terminal_cuties_segmentation`	1	5.85
18	`aicc_round_1_defected_nuts`	1	12.11
19	`aicc_round_1_autocorrect`	1	12.11
20	`aicc_round_5_lost_interpreter`	0	24.00

benefit from a larger feedback budget. The termination analysis separates these possibilities from the narrower question of whether runs were stopped while still executing their chosen solutions.

E.1 TASK-SPECIFIC RESOURCE AND TIME LIMITS

Table 11 lists the GPU allocation and time limit for each benchmark task. Time limits are expressed in decimal hours and apply separately to each task–condition run. A GPU count of zero denotes CPU-only execution. These allocations are shared across agent configurations and assistance conditions.