FORECASTING RESEARCH SUCCESS THROUGH LEARNED COMPARISON OF SCIENTIFIC IDEAS

Srujan P Mule[1,2] **Aniketh Garikaparthi**[2] **Manasi Patwardhan**[2]

[1]IISER Pune [2]TCS Research

srujan.mule@students.iiserpune.ac.in
{aniketh.g, manasi.patwardhan}@tcs.com

ABSTRACT

As potential AGI-level systems begin to reshape science, hypothesis generation represents the most immediate shift, with language models (LMs) generating research ideas at a scale far exceeding the capacity to validate them through experimentation. This creates the risk of wastefully allocating resources to ideas that fail to translate into real-world gains. As a step towards filtering the most promising ideas, we study comparative empirical forecasting: given a research goal and two candidate ideas, predict which one will achieve better empirical performance *before* any experiments are run. We construct a dataset of 11,488 idea pairs grounded in objective benchmark outcomes from PapersWithCode, and find that 8B-parameter models fine-tuned on this data achieve 77.1% accuracy, outperforming frontier models like GPT-5 (61.1%). Further, we attempt to establish interpretable grounds for *trust* by training these models to articulate their reasoning via Reinforcement Learning with Verifiable Rewards, achieving 71.35% accuracy. Such models can serve as transparent filters enabling oversight as the scientific process continues to evolve rapidly.

1 INTRODUCTION

Language Models are starting to function as autonomous research agents that can generate hypotheses, run experiments, and analyze results (Lu et al., 2024; Yamada et al., 2025; Gridach et al., 2025). A recurring pattern is high-throughput ideation (Baek et al., 2025; Si et al., 2024; Garikaparthi et al., 2025), which makes filtering "good ideas" crucial. However, current evaluation approaches rely on language-model judgments over *subjective* criteria like "excitement" or "novelty" (Wang et al., 2024; Baek et al., 2025). While helpful, an idea can be novel and well-argued but still fail to work in practice (Si et al., 2024; Zhu et al., 2025). This gap motivates our study of *comparative empirical forecasting*: given a research goal and two candidate ideas, predict which one will achieve better performance when evaluated on a benchmark. We ask whether language models can be trained to internalize priors and discriminate between two competing ideas *before* running experiments.

To support this task, we construct a large-scale dataset by scraping public benchmark leaderboards to retrieve linked papers, extracting benchmark-specific research goals, descriptions representing competing ideas, and empirical scores. This produces 11,488 labeled idea pairs across 724 valid benchmark leaderboards, grounded in objective outcomes. As a first step, we cast *comparative empirical forecasting* as a direct preference-prediction problem to fine-tune a simple black-box baseline following prior work (Wen et al., 2025). Further to encourage and capture intermediate reasoning, we use a two-stage training process: first, Supervised Fine-Tuning (SFT) on a curated dataset with reasoning traces; then fine-tuning using Reinforcement Learning (RL) variants (Jia et al., 2025; Shao et al., 2024; Yu et al., 2025) to let the model discover reasoning paths. Our results show that while base models struggle (Qwen3-8B achieving only 20.13% accuracy), supervised fine-tuning dramatically improves performance (77.1%). Variants trained to output interpretable reasoning achieve 71.35% accuracy, outperforming GPT-5 (61.10%) while being substantially more compute-efficient.

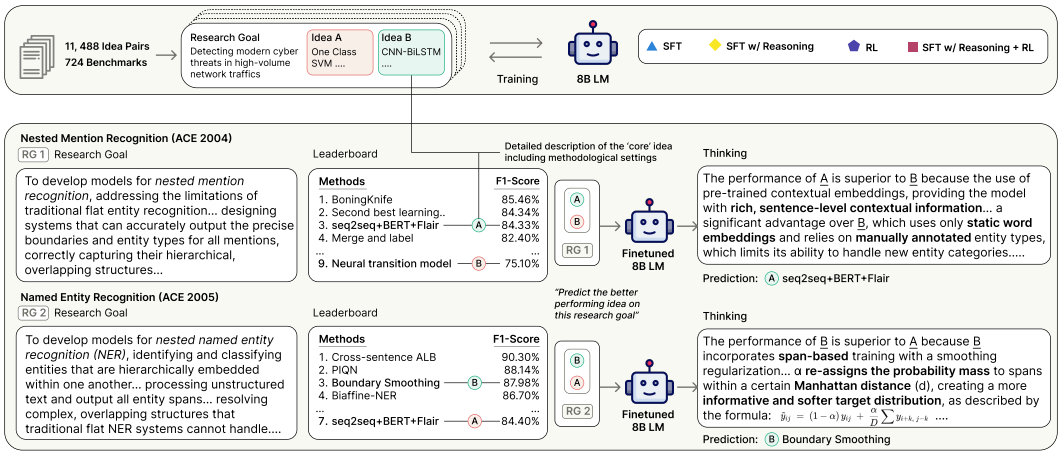

Figure 1: **Task Overview** We explore various methods to fine-tune small language models and discover the potential of fine-tuned models to accurately judge the quality of scientific ideas.

Our main contributions are:

- We introduce a large-scale dataset of research idea pairs with *fine-grained* research goals and outcomes, enabling comparative forecasting on research ideas (§2).
- We show that 8B models can outperform frontier models on comparative scientific forecasting (§5.1), and with RL, learn to produce coherent explanations (§5.3).
- We demonstrate non-trivial cross-domain generalization to non-NLP benchmarks where 8B models show slightly better performance than frontier models. This suggests that comparative forecasting captures transferable scientific priors rather than benchmark-specific heuristics (Cross Domain performance§5.3). Finally, we conduct various ablations and stress tests to validate our results (Robustness under different settings:§E.5).

2 Benchmark

To ground comparative forecasting in objective reality, we develop a pipeline (Appendix C Figure 2) that transforms raw leaderboards into statistically robust comparisons. We scrape 1,918 NLP leaderboards from `paperswithcode.com` and retrieve 5,713 linked **Result Reporting (RR)** and 832 **Original** papers (which originally introduced the ideas, in case RR paper did not), utilizing LLMs to extract detailed idea descriptions and canonical research goals. To ensure rigorous evaluation, we employ a dynamic, RR paper based splitting strategy that ensures there is no leakage of ideas among train and test. We then compute a **Unified Score** for each entry by normalizing and averaging reported metrics, generating $\binom{n}{2}$ idea pairs within each benchmark. Finally, pairs are stratified into difficulty tiers (σ-intervals) based on score divergence, resulting in 11,488 (Train:10619 and Test: 869) unique labeled pairs, across 724 leaderboards, grounded in specific empirical contexts. The test set is manually verified. We also extract the reasoning traces grounded in the actual reasoning, if present, in the RR paper for ideas that have the same RR papers. This results in a very small subset of train (170 pairs) with such grounded reasoning Chain of Thoughts. To rigorously assess generalization, we additionally curate a cross-domain (CD) test set comprising 705 pairs across 46 non-NLP benchmarks from PapersWithCode, filtered for benchmarks with at least three entries, each dated $\geq$ 2024. We release our dataset.

3 Problem Formulation and Methodology

We formally define the problem over the space of scientific hypotheses $\mathcal{H}$, research goals $\mathcal{G}$, and reasoning traces $\mathcal{C}$. We construct a dataset $\mathcal{D} = \{g, h_A, h_B, (c), y\}$, where $h_A, h_B \in \mathcal{H}$ are textual descriptions of competing ideas, $g \in \mathcal{G}$ is the specific research goal (e.g., detecting cyber threats with minimal false positives), and $y \in \{0, 1\}$ is a binary label where $y = 0$ implies h_A outperforms h_B on goal g. $c \in \mathcal{C}$ denotes an optional chain-of-thought. Our objective is to learn a parameterized policy π_θ that accurately predicts y given the context (g, h_A, h_B), while optionally generating the reasoning trace c. To achieve this, we employ a two-phase training pipeline: SFT to instill scientific intuition and RL to discover latent reasoning paths. An overview of our task is provided in Figure 1.

Supervised Fine-Tuning (SFT) Base models lack the comparative intuition to map idea differences to benchmark performance. We bridge this gap via SFT, using standard Cross Entropy loss to maximize the likelihood of the target tokens o_t conditioned on the research goal g, idea pair h_A, h_B, and preceding tokens $o_{<t}$. We apply this in two settings. For **Direct Prediction**, we train on the full dataset where the target is solely the label y, effectively reducing the objective to Binary Cross Entropy. For **Reasoning Cold-Start** (used to initialize RL), we train on the sparse subset of available reasoning traces, where the target sequence includes explanation c followed by y. This step is crucial; initial experiments with pure RL resulted in severe reward hacking (Amodei et al., 2016) and incoherent outputs. The cold-start phase aligns the model with the style of scientific argumentation before exploration begins.

Reinforcement Learning (RL) To induce interpretable reasoning where ground truth is absent, we treat the explanation as a latent variable optimized via RL using the unlabeled portion of the dataset. We initialize the policy π_θ with the cold-start model. For an input $x = (g, h_A, h_B)$, the policy samples a group of G outputs $\{o_i\}_{i=1}^{G}$, each containing a reasoning trace c_i and prediction $\hat{y}_i$. The total reward $R(o)$ sums a correctness score ($r_{\text{cor}} = \pm 3.0$ based on $\mathbf{1}(\hat{y} = y)$) and a formatting score ($r_{\text{fmt}} = \pm 0.5$ for enforcing $\langle think \rangle$ and "Answer:" tags). To incentivize the discovery of valid reasoning paths without succumbing to length bias or instability, we utilize two GRPO variants: **DAPO** (Yu et al., 2025) and **Dr. GRPO** (Liu et al., 2025b). This optimizes the model to discover valid reasoning traces that consistently lead to the correct prediction.

4 Experiments

Metric We design our evaluation to mitigate LLM's vulnerabilities to position bias (e.g., strong preference the first item in a tuple). To achieve this, we consider a prediction **consistent** if the model predicts the same idea to be better for both original and swapped position pairs. For accuracy, an idea pair is considered correctly classified only if the prediction is consistent *and* correct.

Experimental Setup We evaluate **Qwen3-8B** (Yang et al., 2025) and **Llama3.1-8B-Instruct** (Grattafiori et al., 2024), alongside Gemini 2.5 Flash Comanici et al. (2025) and GPT-5 (OpenAI, 2025), in zero-shot setting, for comparison. All training utilized BF16 precision on NVIDIA-A100-40GB GPUs. We split the benchmark train set (§2) into 90% train and 10% validation, stratified by σ-category.

Hyperparameters We apply LoRA (rank 64, α 128) with weight decay 0.01. **SFT:** Trained for 1 epoch with learning rate 2e-4 (cosine scheduler), batch size 2, and 0.1 dropout. **RL:** Trained with learning rate 5e-6, batch size 1, group size 4, dropout 0, and max tokens 3600. We use a small KL-penalty $\beta =$ 1e-5 for regularization (Liu et al., 2025a) and *vllm* via Unsloth for inference.

5 Results and Discussion

5.1 Can LMs predict better ideas based on their likely empirical outcomes?

Table 1 compares the untrained models against their SFT counterparts. We observe that untrained models perform poorly with Qwen3-8B scoring 25.31% and Llama3 scoring a 30.02%, since we consider inconsistent predictions to be wrong by default, the accuracies account for position bias and remain around random guessing (25%).

However, Direct-SFT yields dramatic improvements. **Qwen3** reaches 77.10% accuracy. This result demonstrates that even SLMs can predict the better idea based on their likely empirical outcome, and this doesn't require frontier models as hypothesized by (Wen et al., 2025).

5.2 The Role of Reasoning

To assess the importance of reasoning for our, given the reasoning intensive nature, we study how allowing a LM to think before answering affects predictive performance. We operationalize "reasoning" differently across model families. For Qwen3, we compare the standard (non-thinking) variant against its thinking counterpart. For GPT-5, we vary the thinking budget (low, medium, high). For

Llama 3.1, we prepend a CoT instruction and ask the model to reason before producing its final prediction.

Across these settings, models that are explicitly trained to reason benefit more reliably from deliberation. Qwen3 improves by $\sim 5\%$ points with thinking variant. GPT-5 shows a consistent (but diminishing) accuracy gain as the thinking budget increases. In contrast, naïvely prompting CoT can be counterproductive: Llama 3.1 reduces accuracy from 30.02% to 27.38%.

5.3 Does RL Induce Reasoning?

We further explore whether we can induce correct reasoning. The initial SFT with reasoning (*Reason-SFT*) achieves an overall accuracy of (37.51%).While improving over the Base model, it naturally exhibits lower accuracy compared to the *Direct-SFT*, given the scarcity of samples with reasoning data. However, RL proved effective in recovering predictive performance. Both *Reason-SFT-DAPO* and *Reason-SFT-DrGRPO* achieve accuracies of $\approx 71\%$ (Table 1).

Model / Method	1-σ	2-σ	3-σ	Overall	CD Test
Qwen3					
Base	18.42	26.05	10.99	20.14	3.55
Base (Reasoning)	15.38	27.11	26.11	25.31	12.62
Direct-SFT	**70.85**	**85.56**	**84.62**	**77.10**	45.67
Reason-DAPO	69.43	75.00	83.52	72.73	45.96
Reason-SFT-DAPO	64.57	79.23	83.52	71.35	48.37
Reason-SFT-DrGRPO	66.19	76.41	83.52	71.35	**49.08**
Llama3.1					
Base	37.36	31.33	27.93	30.03	3.83
Base (Reasoning)	26.52	30.63	21.98	27.39	18.22
Direct-SFT	53.64	58.10	67.03	56.50	31.20
GPT-5					
Reasoning (low)	58.70	58.45	49.45	57.65	42.84
Reasoning (med)	59.10	61.62	56.04	59.61	45.25
Reasoning (high)	61.94	61.27	56.04	61.10	45.96
Gemini 2.5 Flash					
Base (Reasoning)	41.90	40.14	36.26	40.73	-

Table 1: Accuracy (%) breakdown across different difficulty subsets i.e. (σ)-categories, on CD test set and models.

While RL mostly restored the accuracy, an inspection of the generated reasoning traces revealed a divergence between the *outcome* and the *process*: (i) **Reason-DAPO** demonstrate incoherent reasoning, partway through the training. The model stops generating any reasoning, while still receiving format rewards and attending only towards the final answer. To prevent this, we introduce a penalty for responses shorter than 600 characters and tweak β to $1e-4$. This results in the model repeating 3-4 sentences before predicting the label. (ii) **Reason-SFT-DAPO** produces consistent and coherent reasoning traces prior to the final answer, while being resilient to the form of reward hacking seen in *Reason-DAPO*, showcasing that it is possible to induce interpretable reasoning in LMs for our task. (iii) **Reason-SFT-DrGRPO** tends to minimize the reasoning trace over the training iterations. The outputs often devolve into superficial justifications (e.g., stating one idea is better because it is "more recent", even when ideas lack any information about their dates).

Cross Domain Test To assess how well the trained models generalize to other domains, we test the performance of the models on the CD test set (§2). We observe that all trained *Qwen3* models perform at par or better than GPT-5 (under all the reasoning effort, and zero-shot setting). Furthermore, we see that RL tuned models like *Reason-SFT-DAPO* and *Reason-SFT-DrGRPO* perform better than *Direct-SFT* ($\approx 3\%$) showing their learning is more robust (Table 1).

6 Conclusion

We show that comparative evaluation grounded in objective outcomes enables language models to forecast research success, addressing the validation bottleneck inherent to automated discovery. Our results show that this capability does not require massive scale: fine-tuned 8B-parameter models achieved 77.1% accuracy, surpassing frontier models by over 15% points and proving that *scalable oversight* can be compute-efficient. Furthermore, we attempt to generate coherent justifications through RL-trained variants (71.35% accuracy) which can foster the *trust* and *transparency* required for meaningful understanding and learning by an observer (which could be both Human and LMs). By serving as effective reward models that filter and potentially rank ideas, these systems can pave a path towards more successful autonomous agents. Finally, our pipeline's ability to leverage expanding leaderboards ensures a continually improving foundation, making scientific oversight resilient as the AGI landscape evolves.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety, 2016. URL https://arxiv.org/abs/1606.06565.

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. Researchagent: Iterative research idea generation over scientific literature with large language models, 2025. URL https://arxiv.org/abs/2404.07738.

Nikhil Chandak, Shashwat Goel, Ameya Prabhu, Moritz Hardt, and Jonas Geiping. Scaling open-ended reasoning to predict the future, 2025. URL https://arxiv.org/abs/2512.25070.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL https://arxiv.org/abs/2507.06261.

Aniketh Garikaparthi, Manasi Patwardhan, Lovekesh Vig, and Arman Cohan. Iris: Interactive research ideation system for accelerating scientific discovery, 2025. URL https://arxiv.org/abs/2504.16728.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Mourad Gridach, Jay Nanavati, Khaldoun Zine El Abidine, Lenon Mendes, and Christina Mack. Agentic ai for scientific discovery: A survey of progress, challenges, and future directions, 2025. URL https://arxiv.org/abs/2503.08979.

Danny Halawi, Fred Zhang, Chen Yueh-Han, and Jacob Steinhardt. Approaching human-level forecasting with language models, 2024. URL https://arxiv.org/abs/2402.18563.

Xiang Hu, Hongyu Fu, Jinge Wang, Yifeng Wang, Zhikun Li, Renjun Xu, Yu Lu, Yaochu Jin, Lili Pan, and Zhenzhong Lan. Nova: An iterative planning and search approach to enhance novelty and diversity of llm generated ideas, 2024. URL https://arxiv.org/abs/2410.14255.

Ruipeng Jia, Yunyi Yang, Yongbo Gai, Kai Luo, Shihao Huang, Jianhe Lin, Xiaoxi Jiang, and Guanjun Jiang. Writing-zero: Bridge the gap between non-verifiable tasks and verifiable rewards, 2025. URL https://arxiv.org/abs/2506.00103.

Ezra Karger, Houtan Bastani, Chen Yueh-Han, Zachary Jacobs, Danny Halawi, Fred Zhang, and Philip E. Tetlock. Forecastbench: A dynamic benchmark of ai forecasting capabilities, 2025. URL https://arxiv.org/abs/2409.19839.

Sang-Woo Lee, Sohee Yang, Donghyun Kwak, and Noah Y. Siegel. Advancing event forecasting through massive training of large language models: Challenges, solutions, and broader impacts, 2025. URL https://arxiv.org/abs/2507.19477.

Long Li, Weiwen Xu, Jiayan Guo, Ruochen Zhao, Xingxuan Li, Yuqian Yuan, Boqiang Zhang, Yuming Jiang, Yifei Xin, Ronghao Dang, Deli Zhao, Yu Rong, Tian Feng, and Lidong Bing. Chain of ideas: Revolutionizing research via novel idea development with llm agents, 2024. URL https://arxiv.org/abs/2410.13185.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models, 2025a. URL https://arxiv.org/abs/2505.24864.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025b. URL https://arxiv.org/abs/2503.20783.

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

OpenAI. Gpt-5 system card, 2025. URL https://cdn.openai.com/gpt-5-system-card.pdf. Accessed: 2026-01-05.

Jungsoo Park, Ethan Mendes, Gabriel Stanovsky, and Alan Ritter. Look before you leap: Estimating llm benchmark scores from descriptions, 2025. URL https://arxiv.org/abs/2509.20645.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers, 2024. URL https://arxiv.org/abs/2409.04109.

Chenglei Si, Tatsunori Hashimoto, and Diyi Yang. The ideation-execution gap: Execution outcomes of llm-generated versus human research ideas, 2025. URL https://arxiv.org/abs/2506.20803.

Haoyang Su, Renqi Chen, Shixiang Tang, Zhenfei Yin, Xinzhe Zheng, Jinzhe Li, Biqing Qi, Qi Wu, Hui Li, Wanli Ouyang, Philip Torr, Bowen Zhou, and Nanqing Dong. Many heads are better than one: Improved scientific idea generation by a llm-based multi-agent system, 2025. URL https://arxiv.org/abs/2410.09403.

Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. SciMON: Scientific inspiration machines optimized for novelty. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 279–299, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.18. URL https://aclanthology.org/2024.acl-long.18/.

Jiaxin Wen, Chenglei Si, Yueh han Chen, He He, and Shi Feng. Predicting empirical ai research outcomes with language models, 2025. URL https://arxiv.org/abs/2506.00794.

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, 2025. URL https://arxiv.org/abs/2504.08066.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Minjun Zhu, Qiujie Xie, Yixuan Weng, Jian Wu, Zhen Lin, Linyi Yang, and Yue Zhang. Ai scientists fail without strong implementation capability, 2025. URL https://arxiv.org/abs/2506.01372.

A LIMITATIONS

Our benchmark relies on a snapshot from the now-defunct PapersWithCode platform, though our methodology for outcome-supervised comparisons is transferable to other active leaderboard[1]. Finally, the current dataset is restricted to NLP benchmarks due to the high cost of validation, leaving expansion into other scientific domains for future work.

B RELATED WORK

Research Ideation Research ideation, being inherently language-intensive, benefits significantly from advances in LLMs Wang et al. (2024); Baek et al. (2025); Si et al. (2024). Recent efforts leverage frontier LLMs via retrieval Li et al. (2024), test-time compute Hu et al. (2024), or multi-agent debate Su et al. (2025). Crucially, these ideas frequently fail to translate into real-world empirical improvements Zhu et al. (2025); Si et al. (2025).

Evaluation Methodologies Most systems evaluate candidates using LLM-judges augmented with retrieval or agents Baek et al. (2025); Garikaparthi et al. (2025). Assessments are typically rubric-driven, focusing on novelty, feasibility, and clarity Li et al. (2024), occasionally calibrated via human studies Si et al. (2024). Methodologically, scoring relies on absolute ratings Baek et al. (2025) or aggregated pairwise rankings Si et al. (2024); Garikaparthi et al. (2025). Verifiers that reward objective performance beyond surface plausibility remain largely underexplored Wen et al. (2025).

LLMs for Forecasting LLMs have demonstrated potential as forecasters of real-world events, approaching competitive human crowd benchmarks Halawi et al. (2024); Karger et al. (2025). Several works employ specialized training for such tasks Lee et al. (2025); Chandak et al. (2025). Closest to our setting are efforts to forecast *empirical ML outcomes*: Wen et al. (2025) train GPT-4.1 to predict better-performing ideas from pairs, while Park et al. (2025) estimate benchmark scores from textual descriptions without experimentation.

C BENCHMARK DATASET

C.1 DETAILED CONSTRUCTION

We describe the full benchmark dataset construction process in detail below.

Scraping and Paper Collection. We extract ideas from entries in live leaderboards, this allows us to build comparisons specific for each benchmark. Thus our evaluation becomes more fine-grained in comparison to parallel work Wen et al. (2025), which can potentially conflate evaluations due to aggregation of scores across various benchmarks via majority voting.

We first scrape all available NLP leaderboards from `paperswithcode.com` that have at least two entries. This yields 1,918 benchmark leaderboards. For each entry in a leaderboard we identify a referenced paper , resulting in 5,713 **Result-Reporting (RR)** papers (excluding 7 behind paywall). We observe for some instances, the RR paper is not the **Original** paper which introduced the method, but rather the paper reporting results on the benchmark using that method. Relying on such papers for idea extraction (to be done in the later stage) would result in generic or incomplete descriptions. Therefore, we prompt an LLM (Gemini-2.5-pro, prompt in Appendix F) to verify whether each RR paper is the one that originally introduces the idea , and if not, look at the full content of RR paper for citations and the reference section to find the original paper. We also ask the LLM to report the confidence of its analysis (high/medium/low). Two NLP experts then manually process the low-confidence entries to verify the identified original paper citation and correct if necessary.We download an additional 908 Original papers based on this analysis. All downloaded papers are parsed using `s2orc-doc2json`[2] to convert the full text into Markdown format, providing clean and structured input for subsequent processing. Papers with unresolvable parsing errors are discarded, resulting in 5,695 RR and 832 Original markdown papers.

[1] hyper.ai

[2] https://github.com/allenai/s2orc-doc2json

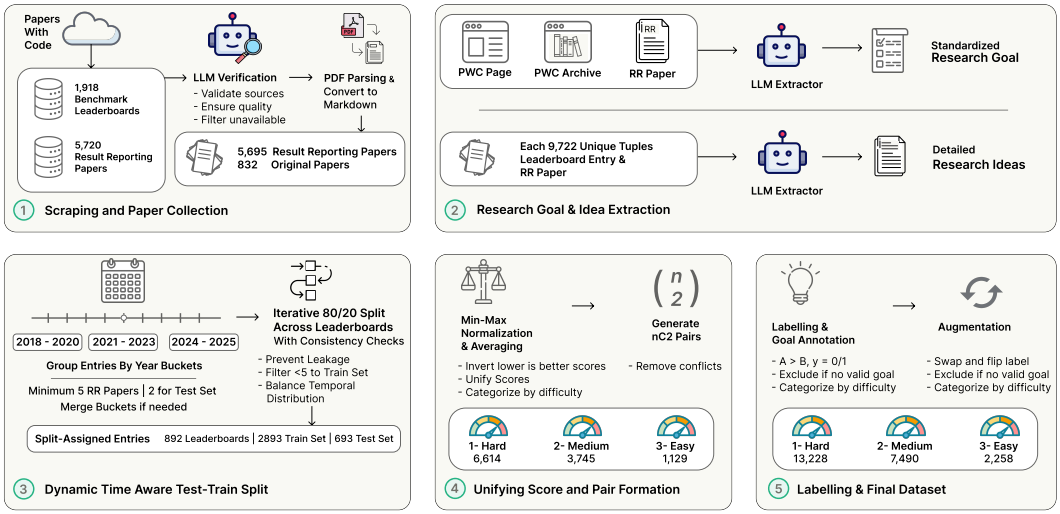

Figure 2: **Dataset Construction Pipeline** We use raw entries from 1,918 NLP leaderboards to construct a statistically grounded idea pairs with a benchmark specific research goal, while difficulty stratification ensures robust evaluation across diverse research goals.

Research Goal and Idea Extraction. For each one of the 1918 leaderboards, we extract a single canonical research goal from official benchmark descriptions in the following order of sources: (1) the dataset page on paperswithcode.com, (2) the corresponding dataset file from the `pwc-archive`[3], or (3) the RR paper when the above sources are unavailable (for 278 benchmarks). The extracted textual description is provided as input to an LLM (Prompt in Appendix F), which generates a clear, comprehensive research goal including what the benchmark evaluates. 327 such benchmarks with missing or unusable sources are skipped.

We process each RR and original markdown paper corresponding to the leaderboard entries with an LLM to extract the detailed idea, excluding any details, empirical results, comparisons, unique identifier like author/model names, year etc. The LLM (Prompt in Appendix F) has access to the complete paper context capturing all necessary details like algorithms, mathematical details etc., whenever present. This results in 9,722 total ideas.

Train–Test Split. We construct the train–test division based on the ideas from the extracted papers belonging to each leaderboard entry by iterating over the leaderoboards. Within a leaderboard, ideas are first grouped by their publication year into respective time buckets. Buckets with fewer than five unique papers are merged with adjacent years.

Algorithm 1 Time bucketing and validation

Require: Entries with years and global split.
1: Create initial year-based buckets.
2: **while** bucket has <5 papers or <2 test papers **do**
3: Merge bucket with adjacent one.
4: **end while**

Ideas in each time bucket are then split in an 80/20 ratio. As we iterate through leaderboards, we ensure that if an idea is already assigned to a split in a previous iteration, this assignment is strictly maintained. Leaderboards with fewer than four total ideas are assigned entirely to the training set. This approach helps prevent an idea appearing in both the train and test avoiding information leakage, while also ensuring similar temporal distribution of the ideas. We additionally employ manual verification of the test set as described below. This process yields 892 leaderboards, with 2,893 ideas in the training set and 693 in the test set.

[3]https://huggingface.co/datasets/pwc-archive/datasets

Details of manual verification of the test set: To ensure that the LLM based idea extractions are accurate, we manually verify the correctness of the idea summary extracted from the parsed papers by consulting the original PDFs.

We observe that the errors/inaccuracies can be classified into 2 main categories:

- **Incomplete**: When the summary falls short of the full detail necessary to correctly summarize the idea. For example, in one instance the description did not include the special loss function that was introduced as part of the idea. In such case we add the necessary details.
- **Incorrect**: When the details of the ideas doesn't correspond to actual ideas/methods. These can be further classified into 2 cases:
  - **Minor**:Minor mistakes like wrong output dimension etc.. Necessary changes are made in such cases.
  - **Major**:When the summary of the idea is completely incorrect. For example, hallucinated details like adversarial component in a BERT based system, when the original method involves simple fine-tuning. We remove such ideas.

About 4% of the test dataset has *Incomplete* ideas and 8% has *Incorrect* ideas, before the final version is obtained.

Unified Score and Pair Formation We next construct idea pairs within each benchmark. Relying solely on the benchmark leaderboard ranks to decide the winner of an idea pair can be misleading, as the performance gap between ranks can vary significantly across leaderboards. To obtain a consistent and quantitative basis for comparison, we compute a **Unified Score** for every idea within a benchmark. To combine heterogeneous metrics, we only use metric columns that were universally reported for all entries in a benchmark. If no metric has universal coverage, the benchmark is skipped. For each benchmark, we first apply min–max normalization for the results of each metric, across all entries. Let $m_i^{(k)}$ be metric k for entry i. We normalize:

$$\tilde{m}_i^{(k)} = \frac{m_i^{(k)} - \min_k}{\max_k - \min_k} \tag{1}$$

We handle "lower-is-better" metrics (e.g., perplexity) by inversion. We calculate the Pearson correlation between $\tilde{m}^{(k)}$ and the rank r. If corr$(\tilde{m}^{(k)}, r) > 0$ (implying higher value = worse rank, e.g., perplexity), we invert:

$$\hat{m}_i^{(k)} = 1 - \tilde{m}_i^{(k)} \tag{2}$$

Approximately 85% of benchmarks report only a single metric. If there is more than one metric, the normalized results are averaged across all to yield a Unified Score for each idea. The final score s_i is the arithmetic mean across adjusted metrics:

$$s_i = \frac{1}{|M|} \sum_{k \in M} \hat{m}_i^{(k)} \tag{3}$$

This procedure captures the relative performance distribution within a benchmark, regardless of metric scale or density.

To ensure the unified scores broadly agree with leaderboard ranks, we use a strict pairwise discordance test. A pair (i, j) is discordant if:

$$\text{discordant}(i, j) = \begin{cases} 1 & \text{if } (r_i < r_j \text{ and } s_i > s_j) \\ & \text{or } (r_i > r_j \text{ and } s_i < s_j) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Algorithm 2 Iterative discordance removal

Require: Entries with ranks and unified scores.
Compute discordance fraction $f = D/\binom{n}{2}$.
while $f > 0$ and at least 2 entries remain **do**
    Identify entry involved in max discordant pairs.
    Remove entry.
    Recompute f.
end while

All possible $\binom{n}{2}$ ideas pairs are generated for each benchmark within each split. We calculate the standard deviation (σ) of the Unified Scores across all entries in that benchmark and use it to define normalized score differences (Δ) for each pair. Based on Δ, we categorize pairs into three mutually exclusive difficulty tiers based on how close their unified scores are: **1σ (hard)**, **2σ (medium)**, and **3σ (easy)**, using a 20% tolerance margin (e.g., 0.8σ–1.2σ for 1σ).

- **1-sigma:** $0.8 \leq \Delta_{ij} \leq 1.2$
- **2-sigma:** $1.8 \leq \Delta_{ij} \leq 2.2$
- **3-sigma:** $2.8 \leq \Delta_{ij} \leq 3.2$

This categorization enables controlled difficulty evaluation based on empirical performance separation.

Algorithm 3 Pair generation

Require: Validated bucket with scores s and std σ.
for each unordered pair (i, j) **do**
    $\Delta_{ij} = |s_i - s_j|/\sigma$.
    if Δ_{ij} in sigma-window **then**
        Emit pair (i, j) with label 1.
        Emit swapped pair (j, i) with label 0.
    end if
end for

Labeling and Final Dataset. Given a pair $(\text{idea}_A, \text{idea}_B)$, we assign $y = 0$ if idea_A has better Unified Score and $y = 1$ otherwise. For each valid pair (i, j) where $s_i > s_j$:

- Generate record: $\{idea_A : i, idea_B : j, label : 1\}$
- Generate swap: $\{idea_A : j, idea_B : i, label : 0\}$

This augmentation ensures the model is robust to input order and the class distribution is perfectly balanced. Each pair is annotated with the corresponding research goal of its benchmark. Benchmark leaderboards without a research goal and all the pairs from such benchmarks are removed. Given the asymmetric removal due to the lack of research goals, we end up with $\approx 90/10$ train/test split in the end. We manually verify each sample in test

	Train Set				Test Set			
Category	**Original**	**Excluded**	**Final (/w Reasoning)**	**Augmented**	**Original**	**Excluded**	**Final**	**Augmented**
1σ	8,881	2,761	6,120 (120)	12,240 (240)	687	193	494	988
2σ	4,827	1,366	3,461 (45)	6,922 (90)	401	117	284	568
3σ	1,436	398	1,038 (5)	2,076 (10)	169	78	91	182
Total	**15,144**	**4,525**	**10,619 (170)**	**21,238 (340)**	**1,257**	**388**	**869**	**1,738**

Table 2: Dataset distribution across sigma categories. Pairs classified as "Excluded" were removed due to the lack of a valid research goal.

Reasoning extraction Predicting empirical outcomes is challenging because research does not always follow clean, deductive logic; often, the explanatory "reasoning" consists of insights gained

only *after* results are observed. Given the reasoning intensive nature of the task, we want Chain-of-Thought (CoT) reasoning traces to train the models. Wen et al. show that using *self-generated* CoT reasoning leads to performance degradation compared to the zero-shot setting for comparative prediction tasks. To avoid this, we take a different approach to extract CoT reasoning. We consider all the idea pairs such that both of the ideas within each pair have the same RR paper. This way, we can be sure that such comparisons actually exist and are presented within the paper and not a case of inferred reasoning. We then prompt an LLM (Prompt in Appendix F) to look for the presence of any explanation for the better performance of one method over the other and extract this as a paragraph reflecting the grounded reasoning. In case such reasoning or justification is not present, the LLM simply has to state that such reasoning is not available. The input prompt contains the full RR paper and a list of all the methods that were reported in this paper.

Cross domain Test set We employ similar pipeline as described above to curate a new test set of idea pairs from **non-NLP** leaderboards (e.g. Speech synthesis, Molecular property prediction etc.) from PapersWithCode that have at least 3 entries, with the years of RR papers ≥ 2024. Unlike the previous case, we use GPT-5 with high reasoning for idea extraction to introduce linguistic diversity relative to the training distribution. Additionally, we do not categorise the pairs based on difficulty, instead use individual metrics directly to form all $\binom{n}{2}$ pairs. For example, if two entries in a leader have 2 reported metrics say accuracy and F1 score, and A has better accuracy than B, while B has better F1 score than A, we assign the labels based on the specific metric and include within the research goal that the aim is to achieve greater accuracy/F1 score. Thus, A and B would result in 2 pairs with slightly different goals and thus different labels. This results in 705 idea pairs across 46 leaderboards. This dataset allows us to test if the trained models can pay attention to metric based nuances, while testing for their ability to generalize to cross domains and taking care of the potential knowledge-cutoff confound.

C.2 Dataset Details

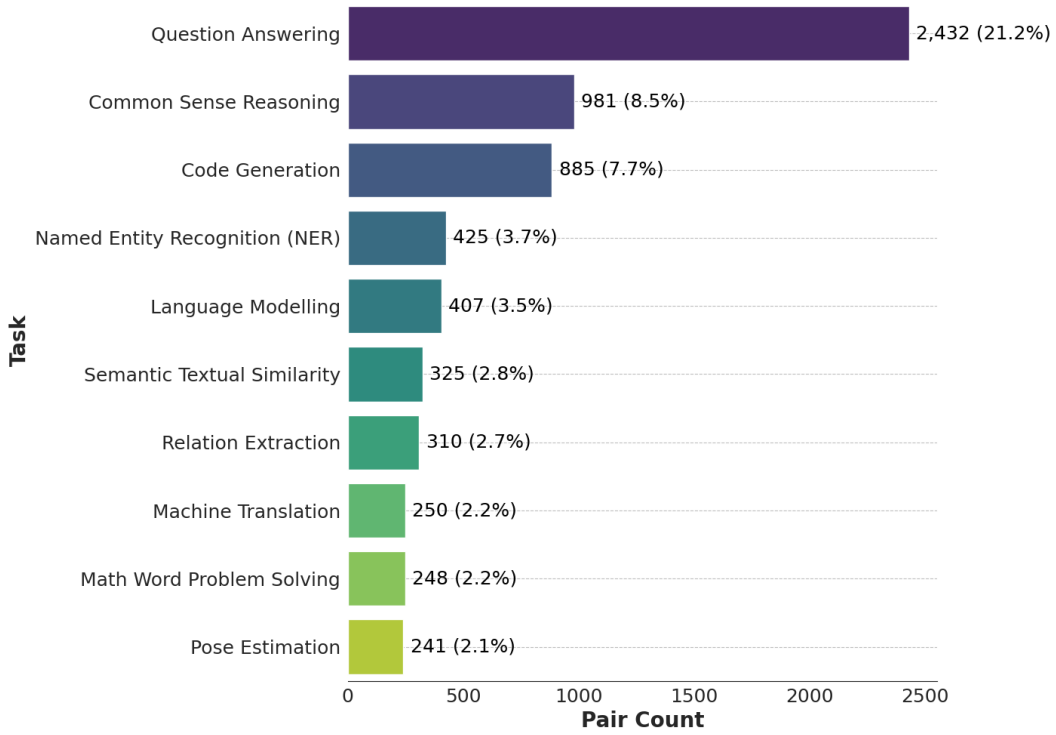

Figure 3: Top 20 Tasks based on the total pairs (includes both train and test)

Dataset Release Notes We release our dataset under Creative Commons License, to be used for research purposes only, here.

Tasks and Benchmarks A benchmark in our dataset is defined as a "Task" (e.g. Question Answering) on a specific dataset (e.g. PIQA). Table 3 shows the top 20 benchmarks based on the total pairs (including train and test) in our dataset. Question Answering and Common Sense Reasoning are most common NLP tasks in our dataset. Further, Figure 3 shows the top 20 "Tasks" and the number of pairs from each of them. Question Answering is the most common task in our dataset (21.2% of total pairs), followed by Common Sense Reasoning with a drastic drop (8.5% of total pairs).

Temporal Distribution Figure 4 and 5 show the temporal distribution of entries of all the leaderboards based on the year of publication of their Result Reporting Paper. The temporal distribution is uni modal in both the train and test sets, with the test set more skewed to the left.

Excluded Pairs Table 2 gives us the full statistics of the final dataset. Even after employing manually verified LLM based Research Goal Synthesis in §(2), we miss out on a large chunk of pairs (close to 30%).

Benchmark	Total Pairs
Code Generation On Mbpp	864
Common Sense Reasoning On Winogrande	454
Question Answering On Copa	381
Named Entity Recognition Ner On Conll 2003	372
Question Answering On Boolq	363
Question Answering On Piqa	331
Common Sense Reasoning On Arc Challenge	227
Math Word Problem Solving On Math	217
Question Answering On Squad11	203
Question Answering On Natural Questions	200
Question Answering On Squad11 Dev	198
Relation Extraction On Docred	179
Question Answering On Webquestions	176
Aspect Based Sentiment Analysis On Semeval	170
Pose Estimation On Mpii Human Pose	163
Word Sense Disambiguation On Words In Context	157
Deblurring On Gopro	155
Common Sense Reasoning On Arc Easy	142
Entity Alignment On Dbp15K Zh En	134
Common Sense Reasoning On Commonsenseqa	126

Table 3: Top 20 Benchmarks by Total Pairs (includes train and test)

Excluded Benchmark Leaderboards Apart from missing Research Goals, some of the benchmarks are excluded during the Train-Test as described in Section §(2) because of the following scenario: Consider a case where the leaderboard has only 2 entries with 2 corresponding RR papers, and due to the iterative nature of of the train-test split, if one of it has been assigned to train and the other to test (based on the splits form the previous leaderboards) we will be unable to form pairs within the train or test subsequently. So, this benchmark leaderboard would get skipped in the process.

D METHODS OF FINE-TUNING IN DETAIL

D.1 DIRECT-SFT

In this phase, the task is strictly defined as correct label prediction. The model is provided with the research goal g and the descriptions of two ideas h_A, h_B. The target output is simply the binary

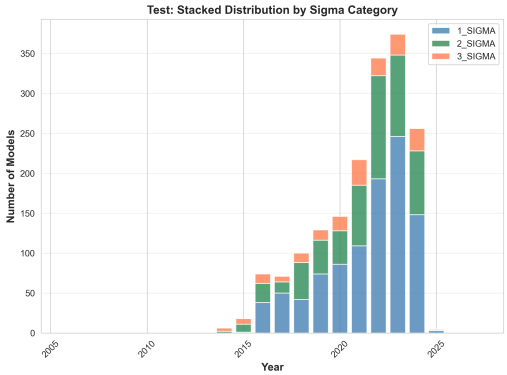

Figure 4: Distribution of the ideas/methods across the years with σ-wise breakdown in test

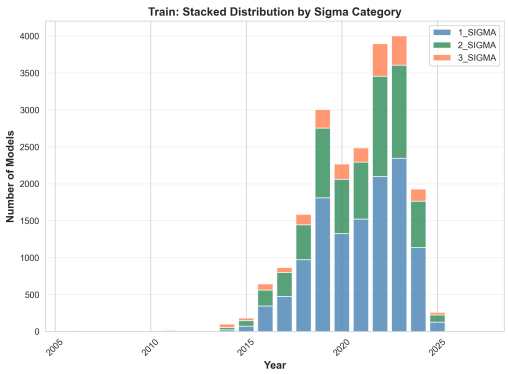

Figure 5: Distribution of the ideas/methods across the years with σ-wise breakdown in train

label $y \in \{0, 1\}$ corresponding to the empirically superior idea. We apply the following SFT loss:

$$\mathcal{L}_{SFT} = -\log P(y \mid g, h_A, h_B) \tag{5}$$

This phase utilizes the full train dataset to ground the model in the “scientific intuition" of identifying successful ideas.

D.2 Reinforcement Learning (RL)

1. Cold Start Finetuning (SFT-Reasoning) We fine-tune the model on the small subset of available reasoning traces. We apply Cross Entropy (CE) loss during fine-tuning:

$$\mathcal{L}_{\text{CE}} = -\sum_{t=1}^{T} \log P(o_t \mid g, h_A, h_B, o_{<t}) \tag{6}$$

where g, h_A, h_B constitute the input context and the target sequence $(o_1, \ldots, o_T)$ represents the reasoning trace c followed by the label y.

This step aims to teach models a *style* of scientific argumentation (e.g., "Idea A reduces variance by... therefore it is likely to outperform Idea B") and align well with human reasoning.

2. Variants of Group Relative Policy Optimization (GRPO) We initialize our policy π_θ and train on the remaining dataset using two variants of Group Relative Policy Optimization (GRPO) that address some of it’s limitations (Yu et al., 2025; Liu et al., 2025b). For a given input context $x = (g, h_A, h_B)$, the model samples a group of G outputs $\{o_1, \ldots, o_G\}$, where each output $o_i \in G$ comprises a generated reasoning trace c_i and a prediction $\hat{y}_i$. The policy is optimized using the advantage estimate A_i derived from the group rewards.

Objective Functions The objective functions of DAPO and Dr. GRPO are based on the probability ratio

$$\rho_{i,t} = \frac{\pi_\theta(o_{i,t}|x, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|x, o_{i,<t})} \tag{7}$$

DAPO utilizes global token normalization and decoupled clipping to encourage exploration, while also addressing the known length bias of GRPO during training.

The objective is:

$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_q \left[\frac{1}{\sum_{j=1}^{G} |o_j|} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \min \left(\rho_{i,t} A_i^{\text{dapo}}, \text{clip}(\rho_{i,t}, 1-\epsilon, 1+\epsilon_{\text{high}}) A_i^{\text{dapo}} \right) \right] \tag{8}$$

subject to the dynamic sampling constraint $0 < \sum \mathbb{I}(\hat{y}_i = y) < G$. The advantage is standardized:

$$A_i^{\text{dapo}} = \frac{R(o_i) - mean(\{R\}_{i=1}^G)}{std(\{R\}_{i=1}^G) + \delta}, \quad \text{and } \epsilon_{\text{high}} > \epsilon \tag{9}$$

Dr. GRPO To fix the length bias (during training) in standard GRPO, we use Dr. GRPO which removes the per-sequence length normalization ($\frac{1}{|o_i|}$) and the standard deviation from the advantage. It sums gradients over all tokens in the group: , which has the following objective function:

$$\mathcal{J}_{\text{Dr}}(\theta) = \mathbb{E}_q \left[\frac{1}{G} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \min(\rho_{i,t} A_i^{\text{dr}}, \text{clip}(\rho_{i,t}, 1 \pm \epsilon) A_i^{\text{dr}}) \right] \tag{10}$$

The advantage A_i^{dr}, is centered but not scaled: the standard deviation term in the denominator of Eq. (7) is removed.

E Analysis and Insights

E.1 Knowledge cutoffs and memorization

For Qwen3 with which we run our primary experiments, an official pretraining knowledge cutoff is not publicly documented. However, we argue that our evaluation is unlikely to be dominated by knowledge cutoff leakage for three reasons.

The prediction target is not a fact in the input text. Our labels are derived from benchmark-specific leaderboard outcomes via a unified score computed from reported metrics, including normalization and direction correction. At inference time, the model is shown only a benchmark-specific research goal and an idea description, while we explicitly remove empirical results and outcome statements from the paper text. Therefore, succeeding on our task requires mapping from a proposed methodological change to its expected empirical impact under a specific benchmark, not simply recalling a numeric result or a rank that appears verbatim in a paper.

Leakage would have to reconstruct a benchmark-conditional comparison, not a single-paper lookup. Even if a model had seen one or both papers during pretraining, the correct answer depends on (i) the specific leaderboard and metric normalization used in our pipeline, and (ii) the relative ordering between two ideas within that benchmark. Memorizing this at scale would require storing a large number of benchmark-conditioned pairwise outcomes across 1,918 leaderboards, rather than recalling isolated paper facts. This makes direct memorization an implausible explanation for performance gains.

Empirical evidence suggests the task is not solved by recall. Base (untrained) 8B models perform poorly (20–30% accuracy after accounting for position bias), which is substantially below chance under our consistency-based evaluation. Moreover, strong proprietary models (e.g., Gemini-2.5-Flash) also struggle zero-shot, indicating that broad pretraining exposure alone does not reliably recover the correct comparative outcome. These results are more consistent with the task requiring non-trivial comparative reasoning than with label retrieval.

Results on unseen CD Test Set Exact knowledge cutoff is not available for Qwen3 models. But Llama3.1 has a knowledge cutoff of December 2023. Inference results on the test set constructed from non-NLP leaderboards, where the entries have also been filtered based on year ($\geq$ 2024), which is post the knowledge cutoff, show that fine tuned models not only generalize well, but also do well, performing either at par or better than frontier models like GPT-5 (which has a much recent knowledge cutoff of September 2024) (Table 1)

E.2 Reward Hacking

Figure 6 shows the running average of rewards and Figure 7 shows the average length of the output/response generated within the reasoning traces of a group over the train iteration of Qwen3

model being trained directly with DAPO objective and a final answer and format based reward as described in §3. As the training closes towards the 24k steps, we see the response length blow up and coupled with a drastic drop in the rewards at the individual steps. This is followed by a drastic drop in the output length. It drops down to 9 and continues generating only 9 tokens. But we see a recovery in the rewards and the models continues to obtain rewards similar to before 23k step. Thus the model transitions to learning to better predict the correct final label corresponding to the better idea in a pair, while generating only the tokens necessary for the format reward and the final reward, and fully circumventing the reasoning. A clear case of "reward hacking".

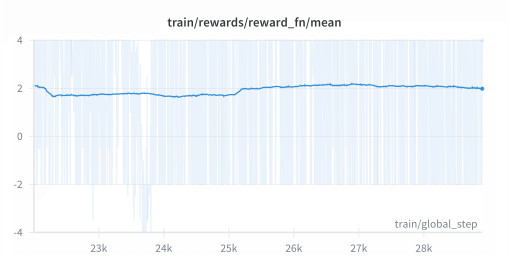

Figure 6: The average rewards through the training iterations of Reason-DAPO

Figure 7: The mean output length through the training iterations of Reason-DAPO

E.3 Qualitative Analysis of Reasoning Quality

The difficulty in generating and achieving the type of inductive bias and the quality necessary for great reasoning in Small Language Models, purely based on the reward of the final answer, has been noted in §5.3. Although we see some good examples in the case of Reason-SFT-DAPO, further qualitative analysis reveals that in most instances the main reason for one idea being better than the other falls into 3 main categories: training on a larger dataset, more sophisticated/recent approach or advancement, and larger model size. Although they achieve great final accuracies (being much better in cross domain tasks than the SFT model, showing great generalization (Table 1)), there seems to be a disconnect between the better idea prediction and the reasoning they generate. This is likely because of the very small subset of the reasoning traces we initially fine tuned the model on and the lack of enough "exploration" coupled with a heavy dependence on "exploitation". This is further evident from the reward hacking behaviour seen in §E.2

E.4 Sensitivity to Benchmark-Specific Research Goals

We evaluate whether the trained model can identify the superior idea within a pair of ideas, conditionally based on the target benchmark i.e. research goal. We observe that all of the Qwen3 trained models demonstrate robust contextual awareness of benchmark specific research goals For instance, the *Efficient Audio Transformer (EAT)* achieves SOTA results on the *Audio Classification on Balanced Audio Set*, but ranks significantly lower on *Audio Classification on ESC-50*, despite a high accuracy of 96% (vs. 99.1% SOTA). We observe that Qwen3 correctly predicts EAT as the superior candidate among the pairs for the *Audio Classification on Balanced Audio Set* benchmark and inferior for *Audio Classification on ESC-50*. This indicates that the model does not rely on superficial textual characteristics or large numerical margins. And exhibits *conditional reasoning*, correctly inferring relative utility of an idea based on a given benchmark.

E.5 Robustness analysis

We test the robustness of the trained models for some features they might be exploiting: (i) **Length:** categorize the idea pairs based on cases where longer idea is better and otherwise, (ii) **Recency:** categorize the idea pairs based on cases where the newer idea (published later) is better or worse, (iii) **Paraphrasing:** test for possible bias based on sentence structure by carefully restating the same (winning) idea using Gemini-2.5-pro (Prompt in Appendix F) in a new way in each pair.

We analyze the model robustness for these potential biases in Figure 8 based on the deviation (Δ) of accuracies on the corresponding subset as : (i) **Length**: larger positive value indicates a bias

towards longer ideas, (ii) **Recency**: larger positive value indicating a bias towards newer ideas, and **Paraphrase**: larger negative value indicates worse accuracy after paraphrasing.

All the trained models show minimal variations in performance due to paraphrasing. All models except Reason-DAPO show high recency bias on the easiest subset. We hypothesise this is likely due to the imbalanced data in respective subsets. Direct-SFT and Reason-SFT-DrGRPO remain most robust to length bias, while Reason-SFT-DrGRPO shows the highest resilience across all tests.

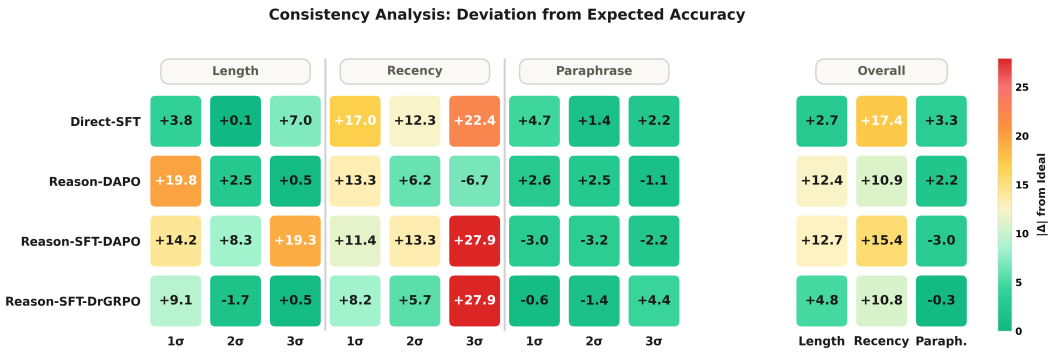

Figure 8: Differential Analysis (Δ based) of different Robustness tests across Difficulty Subsets (σ) and Overall Performance.

Table 4 shows a detailed breakdown of the performance, measured using accuracies on the respective subsets, of different trained Qwen3 models across different stress tests and difficulty levels of idea pair comparison.

E.5.1 Difficulty vs. Length Sensitivity

- **Reason-DAPO**: There is a strong inverse correlation between task difficulty and length sensitivity. As the difficulty increases (moving from 3-σ to 1-σ), the performance gap between "Longer" and "Shorter" inputs widens significantly.
  - 3-σ (Easy): Gap is small $\approx$ 0.4 %).
  - 2-σ (Medium): Gap increases ($\approx$ 2.5 %).
  - 1-σ (Hard): Gap maximizes ($\approx$ 19.9 %).
- **Direct-SFT**: This model shows no correlation between difficulty and length sensitivity. The performance gap between "Longer" and "Shorter" remains consistently low ($<$ 4 %) across all three sigma categories, regardless of task difficulty.
- **Reason-SFT-DAPO**: This model maintains a high length dependency across all levels, but unlike Reason-DAPO, the gap does not widen monotonically with difficulty; it remains large ($\approx$ 14-19 points) in both the easiest (3-σ) and hardest (1-σ) tiers.

E.5.2 Difficulty vs. Recency Variance

- **Reason-DAPO (Inversion Effect)**: The model's preference for "Newer" vs "Older" data inverts based on difficulty.
  - In 1-σ (Hard), it scores higher on "Newer" data (+12.1 %).
  - In 2-σ (Medium), it scores higher on "Older" data (-7.6 %).
  - In 3-σ (Easy), it scores slightly higher on "Older" data (-1.1 %).
- **Direct-SFT**: The "Recency Gap" (Newer minus Older) exists across all difficulties but is non-linear. The model is most robust to recency shifts in the 2-σ category (gap of 6.9 %) compared to the 1-σ (17.3 %) and 3-σ (30.7 %) categories.

E.5.3 Sample Size (N) vs. Accuracy's Stability

- **Variance Correlation**: There is a direct observable correlation between the sample size N and the variance of scores within a model's row.

  - High N (1-σ): Accuracies across splits (Recency/Length) are generally clustered. For example, **Reason-SFT-DrGRPO** has accuracies tightly between 56.8% and 70.9%.
  - Low N (3-σ): Accuracies exhibit extreme volatility. In this tier, **Reason-SFT-DrGRPO** spans from 41.7% to 100.0% accuracies.

- **Paraphrasing Stability**: The impact of paraphrasing is relatively uniform across difficulty levels (N counts are high for both Original and Para). For instance, **Reason-DAPO**'s gain from paraphrasing remains steady at roughly +2.5 to +2.7 % in both 1-σ and 2-σ categories.

E.5.4 Length Dominance across Models

- In the **1-σ (Hard)** category, the hierarchy of models changes depending on the input length.
  - On Longer Inputs: **Reason-DAPO** (79.1%) > **Direct-SFT** (72.9%).
  - On Shorter Inputs: **Direct-SFT** (69.1%) > **Reason-DAPO** (59.2%).

  This indicates that in the hardest difficulty tier, the comparative advantage of the RL model is conditional on the presence of longer context.

Table 4: Detailed Robustness Statistics: Accuracy (%) and Total Sample Count (N). The sample counts for each category (e.g., Longer, Newer) remain same across models for the same σ-subset.

Model	Length		Recency			Paraphrasing	
	Longer	Shorter	Newer	Older	Same	Original	Para.
1-σ (N)	**516**	**466**	**480**	**212**	**296**	**988**	**988**
Direct-SFT	72.9	69.1	79.6	62.3	62.8	70.9	75.5
Reason-DAPO	79.1	59.2	76.3	64.2	62.2	69.4	72.1
Reason-SFT-DAPO	71.3	57.1	70.4	50.9	64.9	64.6	61.5
Reason-SFT-DrGRPO	70.9	61.8	70.4	69.8	56.8	66.2	65.6
2-σ (N)	**278**	**290**	**338**	**134**	**96**	**568**	**568**
Direct-SFT	85.6	85.5	90.5	83.6	70.8	85.6	87.0
Reason-DAPO	76.3	73.8	77.5	85.1	52.1	75.0	77.5
Reason-SFT-DAPO	83.5	75.2	84.6	76.1	64.6	79.2	76.1
Reason-SFT-DrGRPO	75.5	77.2	78.7	83.6	58.3	76.4	75.0
3-σ (N)	**108**	**70**	**146**	**24**	**12**	**182**	**182**
Direct-SFT	87.0	80.0	89.0	58.3	83.3	84.6	86.8
Reason-DAPO	83.3	82.9	82.2	83.3	100.0	83.5	82.4
Reason-SFT-DAPO	90.7	71.4	89.0	58.3	66.7	83.5	81.3
Reason-SFT-DrGRPO	83.3	82.9	89.0	41.7	100.0	83.5	87.9
Overall (N)	**902**	**826**	**964**	**370**	**404**	**1738**	**1738**
Direct-SFT	78.5	75.8	84.9	69.7	65.4	77.1	80.4
Reason-DAPO	78.7	66.3	77.6	73.0	60.9	72.7	74.9
Reason-SFT-DAPO	77.4	64.7	78.2	60.5	64.9	71.4	68.4
Reason-SFT-DrGRPO	73.8	69.0	76.1	73.0	58.4	71.4	71.0

We also plot the percentage of consistent pairs (§4) various models generate post inference on our test set (Figure 10). We observe that the trained models are more robust towards the position bias i.e. are not influenced by the order of presentation, with consistency exceeding 85%.

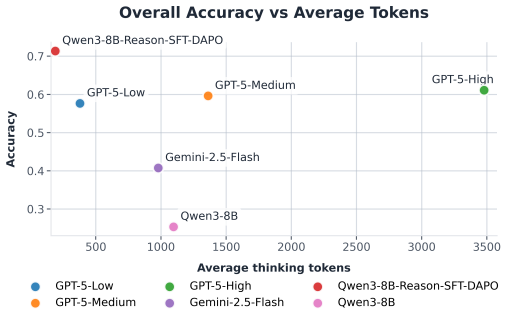

Figure 9: Overall Accuracy (%) Vs Mean Number of tokens generated during reasoning.

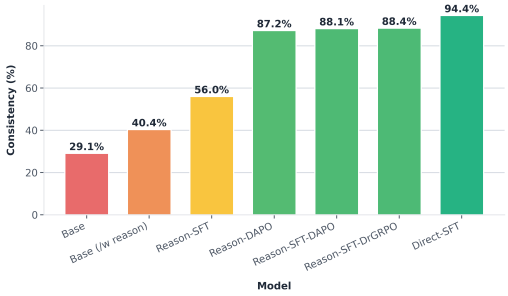

Figure 10: Consistency (%) across different stages and training paradigm of Qwen3 Model.

E.6 Token Efficiency

In addition to observations made in §5.2, Figure 9 shows the average number of reasoning tokens generated before prediction. Reason-SFT-DAPO achieves higher accuracy while using a fraction of the tokens compared to frontier models like GPT-5.

E.6.1 Distribution of Train Dataset

We further analyse the distribution of the idea pairs based on Recency and Length as described in §E.5 (Table 5).

We see that even within the different σ-Categories, the distribution of shorter and longer is relatively balanced. It also contains a very small fraction of pairs that have equal length (The test set doesn't have any). This likely enabled the models to learn much better without introducing such bias due to imbalance. The distribution is largely imbalanced under the newer, older and same year (§E.5) categories, with the number the newer category having almost 2 times the number of pairs in older and same year combined. Older category has consistently lower representation across all σ. This could additionally explain slightly large variations across different trained models, apart from the largely imbalanced distribution within the test itself.

		Recency (%)			**Length (%)**		
Data Split	**Total**	**Newer**	**Older**	**Same**	**Shorter**	**Longer**	**Equal**
Full Train Set	19,113	57.08	17.73	25.19	47.97	51.83	0.20
Breakdown by Sigma (σ)							
$\sigma = 1$	11,016	52.35	19.64	28.01	48.04	51.82	0.15
$\sigma = 2$	6,229	62.93	15.49	21.58	48.40	51.31	0.29
$\sigma = 3$	1,868	65.42	13.97	20.61	46.09	53.69	0.21

Table 5: Distribution of dataset preferences in percentages in Train. The **Total** column indicates the number of samples, while other columns show the percentage breakdown of Recency and Length preferences within each split.

E.7 Conditional Accuracies

We define conditional accuracy as the accuracy of predictions within all the consistent pairs i.e. we consider only the subset of total pairs that show no position bias to compute the accuracy§4. Table 6 reports the conditional accuracy of various models. The number in brackets denote the total number of consistent pairs. The total here represents the augmented set.

Model / Method	**1-σ** (988)	**2-σ** (568)	**3-σ** (182)	**Overall** (1738)
Qwen3				
Base	69.47 (262)	72.55 (204)	50.00 (40)	69.17 (506)
Base (Reasoning)	63.86 (404)	64.71 (238)	46.67 (60)	62.68 (702)
Direct-SFT	75.92 (922)	88.69(548)	90.59 (170)	81.71 (1640)
Reason-DAPO	80.90 (848)	85.54 (498)	86.36 (176)	83.05 (1522)
Reason-SFT-DAPO	75.59 (844)	87.55 (514)	87.36 (174)	80.94 (1532)
Reason-SFT-DrGRPO	76.58 (854)	86.11 (504)	85.39 (178)	80.73 (1536)
Llama3.1				
Base	52.87 (522)	64.96 (274)	64.15 (106)	57.87 (902)
Base (Reasoning)	65.17 (402)	64.93 (268)	68.97 (58)	65.38 (728)
Direct-SFT	78.17 (678)	84.62 (390)	81.33 (150)	80.62 (1218)
GPT-5				
Reasoning (low)	67.13 (864)	67.48 (492)	56.25 (160)	66.09 (1516)
Reasoning (med)	69.03 (846)	70.00 (500)	62.96 (162)	68.70 (1508)
Reasoning (high)	70.67 (866)	68.24 (510)	60.00 (170)	68.69 (1546)
Gemini 2.5 Flash				
Base (Reasoning)	72.13 (574)	66.67 (342)	55.93 (118)	68.47 (1034)

Table 6: Conditional Accuracy (%) breakdown across different σ categories. Values in parentheses denote the total number of samples (N) for that category that were consistent. They are not directly comparable since N changes.

Model	**In-Domain (ID)**			**Cross-Domain (CD)**		
	Con. (%) ↑	**Top-1 (%)** ↑	**RMSE** ↓	**Con. (%)** ↑	**Top-1 (%)** ↑	**RMSE** ↓
Qwen3						
Base	29.55	40.00	1.87	5.35	38.46	2.45
Base (Reason)	40.09	31.43	1.73	19.34	28.21	1.87
Direct-SFT	**90.71**	44.76	1.22	79.84	31.82	1.96
Reason-SFT-DAPO	83.96	42.86	**1.12**	77.37	28.89	1.73
Reason-DAPO	84.85	**51.43**	1.29	76.95	33.33	1.83
Reason-SFT-DrGRPO	87.72	50.48	**1.12**	80.25	41.30	**1.65**
GPT-5						
Low	85.82	38.10	1.48	77.50	36.96	1.76
Medium	86.88	36.19	1.41	81.48	34.78	1.78
High	85.77	35.24	1.41	**82.99**	**43.48**	1.77

Table 7: Performance comparison on In-Domain (ID) and Cross-Domain (CD) test sets. Metrics reported are Overall Consistency Rate (Con.), Top-1 Accuracy, and Median RMSE. **Bold**: Best, Underline: Second Best within each domain. (↓) lower is better; (↑) higher is better.

E.8 IDEA RANKING

We analyze how well the trained models do when used for idea ranking on both in-domain (ID) (from the train-test split) and cross-domain (CD) test set. We pick all leaderboards/research goals that have at least 3 unique entries/ideas in them. We do $\binom{n}{2}$ many comparison. We rank the ideas based on the number of times each idea wins when compared to all others. Ideas with ties are given the same rank. Comparisons that have inconsistent predictions are dropped and not considered. We assess the quality of ranking using Top-1 Accuracy and Root Mean Square Error (RMSE) of the true and predicted ranks.

- We observe that the untrained base models of *Qwen3*, have a high position bias, thus have a poor overall consistency rate (Table 7). Meanwhile, the fine tuned models show drastic improvement, with consistency almost at par (or better in case of In-Domain test) with GPT-5.

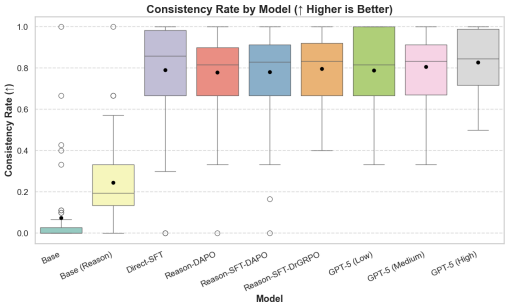

Figure 11: Distribution of consistency rate (%) across different research goals/leaderboards for the cross-domain test set

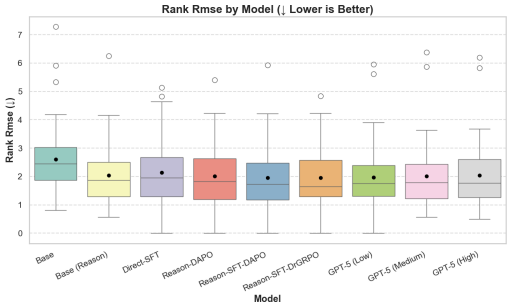

Figure 12: Distribution of RMSE across different research goals/leaderboards for the cross-domain test set

- *Reason-SFT-DrGRPO* achieves better Top-1 accuracy than GPT-5 under low and medium reasoning effort on the CD test. All of the trained models achieve better Top-1 accuracy than GPT-5 on the ID test.
- Before comparing models for their RMSE, it is important to keep in mind that though median is a more robust statistic which is not affected by outliers, it depends highly on the number of samples one has. To make meaningful comparisons, one should have at least similar sample size. Similar consistency rates allow for fair comparisons. For example if we had 10 leaderboards with each having 4 entries and 0% consistency, all the entries within each leaderboard would be assigned a rank of 1. So all of them would have an RMSE of 1.87, which in turn would also be the median. This is lower than what you see for Direct-SFT even when you have no meaningful comparisons or predictions anywhere! This example would also have a 100% Top-1 accuracy! To prevent this we only use Top-1 accuracy from leaderboards that have at least 2 different ranks predicted. This might still not be enough since we see high Top-1 accuracy for base models when they actually don't do very well.
- *Reason-SFT-DAPO* and *Reason-SFT-DrGRPO* achieve better RMSE than GPT-5 (across all reasoning efforts) on CD test, hence showing the potential of such models in filtration and idea re-ranking. All trained models achieve better RMSE compared to GPT-5 on the ID test.
- Figure 11 & 13 and 12 & 14 show the distribution of the consistency and RMSE across different leaderboards/research goals for ID and CD test respectively. The highest outlier of Direct-SFT and Reason-SFT-DrGRPO is lower than that of GPT-5, while Reason-DAPO and Reason-SFT-DAPO have comparable RMSE on CD test.
- The consistency distribution is left-skewed for all cases except for the base models, with most values at high consistency. And the RMSE distribution is right-skewed in all almost all cases. The skewness is greater on ID test compared to CD test, but has more number of outliers in general as well.
- *Reason-SFT-DrGRPO* shows least drastic change in all three metrics for ranking compared to other models showing that it has learnt to generalize better than the others.

These observations show that fine-tuned 8B models that are trained for comparative empirical forecasting could be used as an idea re-ranker, performing better than frontier models under zero-shot setting.

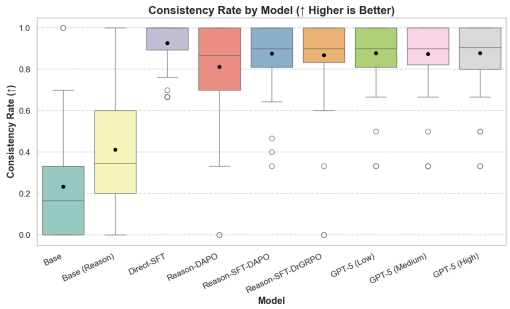

Figure 13: Distribution of consistency rate (%) across different research goals/leaderboards for the in-domain test set

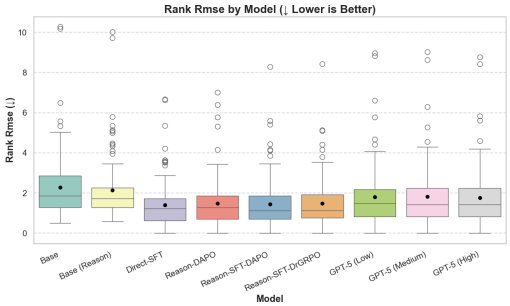

Figure 14: Distribution of RMSE across different research goals/leaderboards for the in-domain test set

F PROMPTS

Idea Extraction

```
Extract the fundamental scientific idea or contribution, methodology and goal from
     this research. Focus exclusively on **what** """ + model_name + """ proposes
     and **how** it works mechanistically, IGNORING all empirical evaluations,
     comparisons, limitations, and benefits.
    (Note that sometime the model or the methods name may be Authors name,
        authorname with year, short forms, their method applied on or over existing
         method or model, humans performance, names of certain teams, simply "
        ensemble" meaning combination of methods or models, or model or method name
         along with certain implementation specifics (like zero-shot etc.). In some
         cases the full text might not explicitly metion this or could be a noisy
        version but try to correlate.)

Paper Information:
Title: """ + proposal_title + """
Context: """ + papers_context + """
Model Introduced in Associated Paper: """ + introduced_info + """
Full Text: """ + full_content + """

Specific Requirements

INCLUDE:
- Core scientific or technical contribution of """ + model_name + """: The main
     idea, algorithm, or method proposed by """ + model_name + """
- Detailed mechanism of """ + model_name + """: Step-by-step explanation of how """
      + model_name + """ operates
- Theoretical foundations of """ + model_name + """: Underlying principles,
     assumptions, and theoretical justifications of """ + model_name + """
- Technical innovations of """ + model_name + """: Novel components, modifications,
      or adaptations introduced.
- Implementation specifics of """ + model_name + """: Key technical details
     necessary for understanding """ + model_name + """
- Parameter definitions in """ + model_name + """: What each variable,
     hyperparameter, and component represents in """ + model_name + """
- Computational processes in """ + model_name + """: How information flows through
     """ + model_name + """
And any other details that are essential to understand """ + model_name + """.

INCLUDE ONLY IF PRESENT IN FULL TEXT:
- Mathematical formulations of """ + model_name + """: All equations, formal
     definitions, and mathematical relationships etc. specific to """ + model_name
     + """
- Architectural details of """ + model_name + """: Model components, layers,
     connections, transformations etc. specific to """ + model_name + """
- Algorithmic procedures of """ + model_name + """: Training procedures, inference
     steps, optimization methods etc. specific to """ + model_name + """


EXCLUDE:
- Empirical results, performance metrics, accuracy scores
- Comparisons or improvements over baseline methods or previous work
- Experimental setup, evaluation protocols
- Advantages, benefits, improvements over existing methods
- Limitations, drawbacks, or failure cases
- Future work suggestions or applications
- Background literature review or related work discussion
- Names given to the model or method or approach

Format your response as a JSON object:
```

```
{
    "paper_id": "Short identifier based on title",
    "title": \"""" + proposal_title + """\",
    "core_idea": "A comprehensive, cohesive scientific exposition of """ +
        model_name + """'s fundamental idea presented as a natural flowing
        paragraph about """ + model_name + """, without the mention of """ +
        model_name + """."
}

CRITICAL REQUIREMENTS:
- Present the extracted idea as ONE comprehensive scientific passage in natural
     flowing prose about """ + model_name + """ as if it is a new idea being
     proposed.
- Present the core idea of """ + model_name + """ as though (or in the tone of) a
     new idea is being proposed instead of summarising existing work.
- Create a cohesive technical exposition that reads like a thorough scientific
     description of """ + model_name + """
- Focus on mechanism and methodology of """ + model_name + """, NOT performance or
     comparisons or improvements
- Extract ideas and research goal objectively, grounded in scientific language and
     full text content only. Ensure you pay attention to all the relevant details
     in the full text about """ + model_name + """
- Present the research goal from the benchmark paper context as a separate,
     detailed paragraph that fully explains the research motivation
- DO NOT INCLUDE model or method name in the core_idea.

Examples of good extractions:
""" + demos + """

Now extract the core research ideas specific to """ + model_name + """ from the
     provided research content:
```

Prompt for Paraphrasing the Idea

```
Restate the following research idea in a different way while preserving ALL
     technical details exactly.
Use different sentence structures and word choices, but do not add, remove, or
     change any information.

**Original:**
{idea}

Return ONLY a JSON object (no markdown):
{{"restated_idea": "..."}}
```

Reasoning Extraction

```
You are analyzing a benchmark paper to extract explicit comparative reasoning
     between specific models/methods. Your task is to identify and extract the
     actual reasoning from the paper that explains WHY one model/method performs
     better than another, using only information explicitly present in the paper.
     Present the reasoning in the form of Chain of Thoughts, along which a strong
     reasoning model would think, analyse and deduce, step-by-step which of the
     given model/method is better than the other, and WHY (grounded in the actual
     truths of the paper). Essentially you are producing a deatiled reasoning
     mimicking an actual human expert's thought process (by grounding it in the
     actual truths of the paper) and presenting it as if it were your own reasoning
     .
CRITICAL CONSTRAINT: Only consider comparisons between the models/methods listed
     below. Ignore all other models.

Benchmark Paper: {benchmark_paper_title}
Models/Methods to Analyze: {models_list}

Paper Content:
{benchmark_content}

Your Task:

The paper may compare multiple models/methods from the list above. For each pair
     where explicit comparative reasoning exists, extract that reasoning. Only
     extract pairs where the paper provides actual reasoning about why one is
     better than or differs from the other.

What to Extract:

Extract a single, cohesive reasoning statement (form of Chain of Thoughts, along
     which a strong reasoning model would think, analyse and deduce, step-by-step
     which of the given model/method is better than the other, and WHY (grounded in
      the actual truths of the paper)) that captures ALL the reasoning the paper
     provides about why idea_A is better or differs from idea_B or visa versa. (
     Essentially you are producing a deatiled reasoning mimicking an actual human
     expert's thought process (by grounding it in the actual truths of the paper)
     and presenting it as if it were your own reasoning.) This should be
     comprehensive and include any and all explanations the paper gives. The
     reasoning could involve:

- Any differences in approach, design, or methodology
- Any factors that contribute to superiority or differences
- Any explanations for why one works better than the other
- Any limitations overcome or advantages gained
- Any combination of factors mentioned in the paper

Be inclusive: Capture all reasoning provided by the paper, even if it doesn't fit
     into traditional categories. You may include a hypothesis as a statment if
     experminetal results support/prove it.

Critical Requirements:

- Use idea_A and idea_B notation: Replace actual model/method names with "idea_A"
     and "idea_B" in your reasoning
- Single cohesive reasoning: Combine all aspects into ONE comprehensive reasoning
     statement per pair
- Ground in paper content: Extract actual reasoning from the paper, not your
     interpretation
- First-person style: Present reasoning as direct statements, not "the paper says
     ..." or "according to the paper..."
```

```
- Only pairs with reasoning about superiority: Only include model pairs where the
    paper explains WHY one is better, not just WHAT the differences are
- Only specified models: Ignore comparisons with any models not in the list: {
    models_list}

What to EXCLUDE:

ONLY exclude the following:
- Pure statements about empirical performance (e.g., "idea_A achieves 95\% accuracy
     while idea_B achieves 90\%") or any details of empirical results.
- Simple restatements that one is better based solely on numbers or metrics without
     any explanation of WHY
- Mere descriptions of differences without explanation of superiority (e.g., "
    idea_A uses attention while idea_B uses convolutions" - this is just a
    difference, not reasoning about why one is better)
- Experimental setup details without reasoning or conclusions obtained after
    running the experiments.
- General background information
- Future work or speculation
- Comparisons with models NOT in the specified list
- Mention of the paper anywhere in your reasoning (e.g., avoid phrases like "the
    paper states..." or "according to the authors...")
- Any direct quotes/texts or refrences to the paper
- Conclusion or inference drawn from looking at the performance. There should be NO
     mention of performance based on the experimental results.

KEY DISTINCTION: The paper must explain WHY the difference leads to one being
    better, not just WHAT the differences are.

Example of what NOT to extract:
- "idea_A uses transformers while idea_B uses LSTMs" (just a difference)
- "The user is asking me to reason why Idea A is better than Idea B....." (
    starting by already assuming or declaring which of the ideas is better, and
    you have to just figure out why is this the case)
- ".........The paper specifies that idea_A, the hybrid model, 'simply retains two
     self-attention layers' in addition to its H3 layers.\n2. Next, we evaluate
    the impact of this difference. The paper suggests that this architectural
    choice is not arbitrary but is designed to leverage the 'complementary
    strengths of SSMs and attention.'\n3....." (Mention of paper explicitly
    )
- "To determine which model is better, let's analyze their core design and
    resulting performance characteristics based on the paper's findings....." (
    Explicitly mentions the paper)
- "Despite these significant advantages in size, data, and training, idea_B only
    achieves an 8-point performance gain over idea_A....." (mentions performance
     numbers)

Example of what TO extract:
- "idea_A uses transformers which can capture long-range dependencies more
    effectively than the LSTM architecture in idea_B, allowing it to handle
    complex contexts that idea_B struggles with" (explains why the difference
    matters)
- "We want to compare idea_A and idea_B to figure out which of the two is better.
    Let us break this down step by step....." (Starts in a way that does NOT
    already declare one of the ideas being better than the other and as if the
    real task is simply supposed to reason out why this is the case)
- ".........idea_A, the hybrid model, retains two self attention layers in
    addition to its H3 layers. The architecture design leverages complmentary
    strengths of SSMs and attentions....." (no mention of paper explicitly,
    and captures the reasoning about WHY this design choice matters)
```

```
Everything else should be included - if the paper provides ANY reasoning or
     explanation for why one performs better, extract it. Don't limit yourself to
     specific types of reasoning.

Output Format:

Return a JSON object with this structure:

{{
    "benchmark_paper": "{benchmark_paper_title}",
    "models_analyzed": [{models_list}],
    "comparative_reasoning": [
        {{
            "idea_A": "Actual Model A name from the list",
            "idea_B": "Actual Model B name from the list",
            "reasoning": "Single cohesive explanation using idea_A and idea_B,
                 combining all reasoning factors from the paper. Capture whatever
                 explanation the paper provides for why idea_A performs better or
                 differs from idea_B or vise versa. This could be due to design
                 differences, training approaches, data choices, theoretical
                 properties, implementation details, or any other factors the paper
                 mentions - include it all in one coherent statement.",
            "source_section": "Brief context where this reasoning was found (e.g., '
                 Related Work', 'Results Analysis', 'Discussion')"
        }}
    ],
    "overall_assessment": "Summary of how many pairs had comparative reasoning, or
         'No explicit comparative reasoning found between the specified models'"
}}

VALIDATION RULES:
1. Both idea_A and idea_B MUST be actual model names from the list: {models_list}
2. All model names in the "reasoning" text MUST be replaced with "idea_A" or "
     idea_B"
3. Reasoning must be grounded in actual paper content, not inferred
4. Only include pairs where the paper explains WHY one is better, not just
     describes differences
5. The reasoning must connect differences to advantages or why one is better than
     the other.
6. If no such reasoning exists for any pair, return empty comparative_reasoning
     array
7. Capture ALL reasoning the paper provides - don't limit to specific types

Important Notes:
- Multiple models: You may receive 3, 4, or more models to analyze. Extract
     reasoning for each pair where the paper explains superiority.
- Not all pairs need reasoning: If the paper doesn't explain why one is better (
     just mentions differences), don't extract a reasoning for that pair.
- Be comprehensive: Whatever reasoning the paper provides about which one is better
     , capture it all in the reasoning field.
- Reasoning vs Differences: The paper must explain the consequence or advantage of
     the difference, not just state the difference.

Example with 4 models: If you receive models A, B, C, D, you might extract
     reasoning for pairs (A,B), (B,C), and (C,D) if the paper provides reasoning
     for those comparisons, but skip (A,C), (A,D), (B,D) if no reasoning exists for
      those pairs.

Now extract the comparative reasoning from the paper content:
```

Research Goal Extraction

```
You are a research scientist tasked with converting benchmark information into a
     comprehensive research goal. Your job is to identify the core research
     objective that this benchmark addresses and articulate it as a single, well-
     structured paragraph.

{benchmark_info}

REQUIREMENTS FOR THE RESEARCH GOAL:
1. Write as a SINGLE comprehensive paragraph (not multiple sections)
2. Focus on the core RESEARCH OBJECTIVE that this benchmark addresses
3. Include what type of input data is used, what output is expected, and how
     performance is measured
4. Be specific about the research challenge and why it is important
5. Use scientific language but keep it readable and focused
6. Mention the specific benchmark/dataset name
7. Keep the research goal between 3-5 sentences

INSTRUCTIONS:
- Write a cohesive paragraph that flows naturally
- Start with the research objective or problem being addressed
- Include input/output specifications naturally within the paragraph
- Mention evaluation approach without making it a separate section
- Focus on the RESEARCH GOAL, not just describing the benchmark
- Avoid bullet points or structured formatting

EXAMPLES OF GOOD RESEARCH GOALS:

Example 1: "This research aims to develop language models that can accurately
     determine logical relationships between premise-hypothesis text pairs,
     addressing the fundamental challenge of natural language inference. The
     research involves training models to process paired natural language sentences
      and classify whether the hypothesis is entailed, contradicted, or neutral
     with respect to the premise, with performance measured using classification
     accuracy and F1-score across the three relation types. This work is essential
     for advancing machine reading comprehension and logical reasoning capabilities
      in natural language processing systems."

Example 2: "This research focuses on creating neural machine translation models
     that can produce high-quality translations between low-resource language pairs
      using minimal parallel training data. The objective is to develop systems
     that can process source language sentences from news articles and web
     documents and generate target language translations that preserve semantic
     meaning and grammatical correctness, evaluated using BLEU scores, chrF scores,
      and human evaluation ratings for fluency and adequacy. This research
     addresses the critical need for effective translation systems in
     underrepresented languages where large parallel corpora are not available."

BENCHMARK TO PROCESS: {benchmark.benchmark_name}

Generate a single research goal paragraph:
```

True Original Paper analysis

```
You are analyzing an academic paper to determine if the given method or models were
     originally introduced.

Paper Title: "{paper_title}"
Models or methods to analyze: {models_list}

CRITICAL: Treat each model name as ONE COMPLETE MODEL OR METHOD NAME. Do NOT split
     model names like "RNN-1024 + 9 Gram" into separate components. Each model or
     method name listed above should get exactly ONE analysis entry, regardless of
     what symbols ("+", "&", "with", etc.) it contains.

{content_section}For each model or method, follow this process:

1. Analyze each model name as a complete unit: Take the EXACT model or method name
     as given and analyze it as one single model/method, even if it contains
     symbols like "+", "&", "with", etc.

2. Check if originally introduced: You may look for phrases like "we propose", "we
     introduce", "we present [exact_model_name]", "our [exact_model_name]",
     detailed descriptions indicating novelty or any other relavant context.

3. If NOT originally introduced: Look for citations when the complete model or
     method name is mentioned:
   - Find phrases like "using [exact_model_name] from [citation]", "based on [
       exact_model_name] [citation]", "[exact_model_name] (Author et al.)" etc.,
       but be mindful of cases where the exact model name is just a variant of the
       original (Like MethodX(unidirectional) etc.). Or any other form of citations
        present with the model name anywhere else (like in tables etc.).
   - Locate the citation in the references/bibliography section
   - Extract the original paper title and authors from the reference

4. For combination-style model or method names (e.g., "ModelA + ModelB", "Enhanced
     ModelX", "ModelY with additional components (like trained on certain dataset
     etc.)"):
   - Treat the ENTIRE name as ONE MODEL - do not analyze components separately
   - If the complete combination is a novel approach, mark as
       introduced_in_this_paper = true
   - If the complete combination cites prior work, identify those supporting papers
   - Include supporting papers as supporting_paper_title and supporting_authors if
       present

5. Use citations to find original papers: When a model is cited, go to the
     references section and find the complete bibliographic information for that
     citation.

Return JSON format with EXACTLY ONE entry per model name provided:
{{
    "models": [
        {{
            "model_name": "EXACT_MODEL_NAME_AS_PROVIDED",
            "introduced_in_this_paper": true/false,
            "original_paper_title": "Title of original paper (if different, else
                null)",
            "original_authors": "Authors if available from citations (else null)",
            "confidence": "high/medium/low",
            "reasoning": "Brief explanation including citation info if found"
        }}
    ]
}}
```

```
IMPORTANT: You must return exactly {len(unique_models)} model entries, one for each
      model name provided. Do NOT split model names into components.
```

Prompt format used for FT with RL

```
system_prompt = (
"You are an expert AI research assistant. Evaluate two research ideas and determine
      which one is better."
)
user_content = (
    "Research Goal: " + research_goal + "\n\n"
    "Idea A: " + idea_A + "\n\n"
    "Idea B: " + idea_B + "\n\n"
    )
user_content += (
    "Please reason step by step about which idea is better. "
    "Then provide your final answer in the format: \"Answer: [0 or 1]\" where 0
        means Idea B is better and 1 means Idea A is better."
    )
```

G EXAMPLES

G.1 QWEN3-8B-REASON-SFT-DAPO

An example where the RL trained model successfully reflects and reasons the probable cause for one idea being better than the other and successfully predicts the correct answer. And a second example where the same model reasons and reflects but the reasoning leads to wrong final answer prediction.

Input

```
    "research_goal": "The primary research objective is to develop robust deep
        learning models capable of learning accurate image classifiers from
        training data corrupted by realistic, human-generated label noise. This
        research addresses the significant real-world challenge of training models
        when ground-truth labels are unreliable, a common scenario in crowd-sourced
         or large-scale data collection. Utilizing the CIFAR-10N benchmark, models
        are trained on CIFAR-10 images paired with their associated noisy labels,
        with the ultimate goal of accurately predicting the true class for unseen
        images. The performance of these noise-tolerant algorithms is evaluated by
        their classification accuracy on the original, clean CIFAR-10 test set,
        demonstrating their ability to generalize beyond the imperfect training
        data."

    "idea_A": "A learning paradigm is proposed for robustly training deep neural
        networks in the presence of noisy labels by simultaneously training two
        networks and having them teach each other. The methodology involves
        maintaining two neural networks, f (with parameters w_f) and g (with
        parameters w_g), which have identical architectures but are initialized
        differently to encourage diverse learning behaviors. The training proceeds
        in a mini-batch fashion. For each mini-batch of data, both networks perform
         a forward pass and calculate the loss for every instance. Based on these
        loss values, each network independently selects a fraction of the instances
         it deems most likely to be correctly labeled, identified as those with the
         smallest training losses. The core mechanism is a cross-update procedure:
        network f is updated via backpropagation using only the small-loss
        instances selected by its peer, network g, and conversely, network g is
```

```
    updated using the small-loss instances selected by network f. This process
    is founded on two principles. First, it leverages the memorization effect
    of deep networks, where models tend to learn from clean, easy examples
    before fitting to noisy, hard ones; thus, small-loss instances are likely
    to be clean, especially in the early stages of training. Second, the use of
     two networks and a cross-update step prevents the accumulation of error.
    Since the two networks have different learning abilities, they can filter
    different types of noise. If one network mistakenly selects a noisy
    instance, the error is passed to its peer for an update but is not
    immediately fed back into the originating network, breaking the cycle of
    error accumulation common in self-evolving systems. The fraction of data
    selected, R(T), is dynamically adjusted over training epochs T. It starts
    at 1 (using all data) and gradually decreases according to the schedule R(T
    ) = 1 - \u03c4 * min(T / T_k, 1), where \u03c4 is a hyperparameter related
    to the estimated noise rate and T_k defines the number of epochs before the
     drop rate stabilizes. This allows the networks to learn general patterns
    from all data initially and become progressively more selective as they
    gain the capacity to overfit to noise."

"idea_B": "A loss correction procedure is proposed to train neural networks on
    data with class-conditional label noise by modifying the model's
    predictions to account for the noise process. The approach assumes the
    existence of a known `c x c` noise transition matrix `T`, where `T_ij = p
    (\u1ef9 = e_j | y = e_i)` specifies the probability of a true class `i`
    being observed as class `j`. The core mechanism operates by altering the
    output of the network before the loss calculation. For a given input `x`,
    the network first produces logits `h(x)`, which are then transformed by an
    inverse link function `\u03c8\u207b\u00b9` (e.g., the softmax function) to
    yield a vector of predicted clean label probabilities, `p(y|x)`. Instead of
     comparing this clean prediction to the noisy label `\u1ef9`, the procedure
     projects the clean prediction into the noisy label space by left-
    multiplying it with the transition matrix `T`. The final loss is then
    computed between the observed noisy label `\u1ef9` and this transformed
    prediction, `T * \u03c8\u207b\u00b9(h(x))`. This method is theoretically
    justified for any proper composite loss, `\u2113_\u03c8`, such as cross-
    entropy or square loss. The corrected loss, defined as `\u2192\u03c8(h(x))
    = \u2113(y, T * \u03c8\u207b\u00b9(h(x)))`, is itself a proper composite
    loss with a new, implicitly defined link function `\u03c6 = (T\u207b\u00b9)
    \u2218\u03c8`. A key theoretical result demonstrates that the minimizer of
    this corrected loss under the noisy data distribution is identical to the
    minimizer of the original loss under the clean data distribution.
    Specifically, minimizing `E_{x,\u1ef9}[\u2192\u03c8(y, h(x))]` leads to an
    optimal model that targets `\u03c6(p(\u1ef9|x))`, which simplifies to `\
    u03c8((T\u207b\u00b9)p(\u1ef9|x)) = \u03c8(p(y|x))`. This is the same
    target as minimizing the original loss on clean data, thus guaranteeing
    noise robustness at the minimizer without requiring an explicit inversion
    of the matrix `T` during computation."
```

Output

```
"thinking": "<think>\nThe performance of idea_A is superior to idea_B because
    idea_A is a self-evolving method that trains two networks to teach each
    other, allowing them to learn from all data. In contrast, idea_B is a
    static method that only corrects the loss of a single network, which limits
     its capacity to learn from all data. The training of idea_A is designed to
     break the cycle of error accumulation, a capability not present in the
    single-network approach of idea_B. Furthermore, idea_A employs a dynamic
    noise ratio (R(T)) that starts by using all data and gradually selects less
     data as training progresses, a mechanism that enables the model to learn
    general patterns from all data initially and then focus on difficult, noisy
```

```
         instances. This dynamic selection, combined with the cross-update
        procedure between two networks, allows idea_A to learn from all data
        throughout training, whereas idea_B, being a single network, cannot learn
        from all data and only learns from the clean data. This comprehensive
        learning capability of idea_A is a significant factor in its superior
        performance compared to the static, single-network idea_B.\n</think>"

    "prediction": 0,

    "ground_truth": 0
```


Input


```
"research_goal": "The primary research objective is to develop and evaluate natural
     language processing models with a robust understanding of physical
    commonsense reasoning, a critical capability for real-world interaction. Using
     the Physical Interaction Question Answering (PIQA) benchmark, this research
    challenges models to process a question about a physical situation and select
    the more plausible of two potential solutions. Model performance is measured
    by accuracy in choosing the correct solution, directly assessing the system's
    ability to reason about the physical properties and affordances of everyday
    objects. This work is crucial for advancing AI systems beyond purely
    linguistic pattern matching towards a more grounded, human-like understanding
    of the world.",

"idea_A": "A two-stage approach is proposed for creating compact language models
    from larger, pre-trained source models. The first stage, termed targeted
    structured pruning, reduces the model to a pre-specified target architecture.
    This is formulated as a constrained optimization problem that learns pruning
    masks for various model substructures\u2014specifically layers, attention
    heads, intermediate dimensions, and hidden dimensions. The pruning decision
    for each substructure is controlled by a mask variable parameterized using a
    hard concrete distribution, which allows for discrete retain-or-prune
    decisions within a continuous optimization framework. Instead of targeting a
    general sparsity level, the method enforces constraints on the final model
    shape directly using Lagrange multipliers. The overall objective is a min-max
    optimization of the function L_prune(\u03b8, z, \u03bb, \u03d5) = L(\u03b8, z)
     + \u03a3_j L_head_j + \u03a3_j L_int_j + L_layer + L_hidden, where L(\u03b8,
    z) is the language modeling loss with masked weights, and the other terms are
    Lagrangian penalties. For instance, the constraint for the number of heads in
    a layer is L_head(\u03bb, \u03d5, z) = \u03bb_head * (z_head - H_T) + \
    u03d5_head * (z_head - H_T)^2, where H_T is the target number of heads. This
    process jointly optimizes model weights (\u03b8) and pruning masks (z) to find
     a subnetwork that matches the target architecture while preserving
    performance. After this stage, the highest-scoring components are retained to
    finalize the pruned model's structure.\n\nThe second stage involves continued
    pre-training of this pruned model, enhanced by a dynamic batch loading
    algorithm designed to address inefficient learning across different data
    domains. This algorithm adjusts the data sampling proportions on-the-fly based
     on the model's performance in each domain. A 'reference loss' (\u2113_ref) is
     established for each domain, which can be derived either by using a scaling
    law function fitted on a series of models of different sizes to predict the
    loss of a hypothetical model of the target size, or by using the source model'
    s validation loss. During training, the model's current validation loss (\
    u2113_t) is periodically evaluated for each domain. The data loading weights (
    w_t) for subsequent training batches are then updated in proportion to the
    difference between the current loss and the reference loss (\u2206_t[i] = max
    {\u2113_t[i] - \u2113_ref[i], 0}), effectively up-sampling data from domains
    where the model's performance is lagging. This ensures that the model's loss
```

```
    reduces more evenly across all domains, leading to a more efficient use of the
     training data.",

"idea_B": "A method is proposed for converting a pre-trained dense model into a
    sparse Mixture-of-Experts (MoE) architecture in a parameter-efficient manner.
    The process begins by replacing the feed-forward network (FFN) layers within
    the dense model's transformer blocks with MoE layers. Each MoE layer is
    composed of a set of experts and a gating router. During initialization, every
     expert within a given MoE layer is created as an identical copy of the
    original FFN layer from the dense model, inheriting its weights, denoted as \
    u03b8o. The core of the method lies in how these identical experts are
    differentiated during training. Instead of fine-tuning the large set of
    parameters \u03b8o for each expert, which is computationally expensive, the
    expert weights \u03b8o are kept frozen. Differentiation is achieved by
    inserting a small, trainable adapter module after each expert. For an input x
    processed by the shared expert E, the output is passed to a specific adapter
    Ai. The adapter's transformation is defined as Ai(E(x)) = \u03c3(E(x)W_i_down)
    W_i_up + E(x), where W_i_down and W_i_up are the trainable low-rank matrices
    of the i-th adapter, and \u03c3 is an activation function. The training
    process focuses on optimizing only the parameters of these adapters, \u03c9i,
    for each expert, along with the router weights. The final output of the MoE
    layer is a weighted combination of the adapter-modified expert outputs,
    formulated as y = \u03a3 R(x)i * Ai(E(x)), where R(x)i is the gating score for
     the i-th expert. This approach is theoretically grounded in the idea that the
     adapters, being universal approximators, can effectively approximate the
    functional changes of a fully fine-tuned expert with a small error, thus
    projecting the optimization into a much lower-dimensional space. To manage
    expert selection, a top-k gating router is employed, which uses trainable
    weights Wr to calculate logits R(x) = Wr \u2022 x. A KeepTopK function selects
     the top k experts for each token before a softmax normalization. To ensure
    balanced utilization of experts during training, an auxiliary load-balancing
    loss is incorporated, calculated as L = \u03b1 \u2022 n \u2022 \u03a3 fi \
    u2022 pi, where fi is the fraction of tokens dispatched to expert i, and pi is
     the fraction of router probability it receives.",
```

Output

```
   "thinking": "<think>\nThe performance of idea_A is superior to idea_B because
        idea_A undergoes a two-stage training process. The first stage, targeted
        structured pruning, is used to convert a large, dense model (like idea_B)
        into a smaller, more efficient model (idea_A) by removing redundant
        parameters. This process is formulated as a constrained optimization
        problem that jointly learns the model weights and pruning masks to retain
        the highest-scoring components, directly targeting a specific, smaller
        architecture (e.g., 7B parameters) rather than just a general sparsity
        level. The second stage involves continued pre-training of the pruned model
         with a dynamic batch loading algorithm, which adjusts the data sampling
        proportions on-the-fly based on the model's performance across different
        domains. This allows the smaller model (idea_A) to be trained efficiently
        on the same large corpus as the source model (idea_B), using only 15\% of
        the training data. This comprehensive training methodology enables idea_A
        to outperform the larger, directly fine-tuned idea_B on downstream tasks,
        as the training data and compute are focused on the smaller model.\n</think
        >",

"prediction": 0,

"ground_truth": 1
```

G.2 QWEN3-8B-REASON-DAPO

For the same example as the above (the second example in §(D.1), the model trained with DAPO applied directly to the base model with a penalty for shorter responses, we see clear reward hacking. The thinking starts off well and rightly acknowledges the need for evaluating likely performance on the specific task/research goal but then enters a repetitive loop of the same sentences before making the final prediction, thus failing to reason coherently or justifying the "why". Yet it gets the final prediction correct unlike the Qwen3-8B-Reason-SFT-DAPO model.

Output


```
"thinking": "<think>\nOkay, let's try to figure out which idea is better. So, we
    have two methods: one is a pruned model (Idea A) and the other is the MoE
    model (Idea B). Both are derived from the same large model, right? The
    question is about which one performs better on the PIQA task. \n\nFirst, I
    need to understand the differences between the two models. The MoE model (Idea
     B) is the model with the Mixture-of-Experts (MoE) method. The model with the
    Mixture-of-Experts (MoE) method (Idea B) is the model with the Mixture-of-
    Experts (MoE) method. The model with the Mixture-of-Experts (MoE) method (Idea
     B) is the model with the Mixture-of-Experts (MoE) method. The model with the
    Mixture-of-Experts (MoE) method (Idea B) is the model with the Mixture-of-
    Experts (MoE) method. The model with the Mixture-of-Experts (MoE) method (Idea
     B) is the model with the Mixture-of-Experts (MoE) method.\n\nThe model with
    the Mixture-of-Experts (MoE) method (Idea B) is the model with the Mixture-of-
    Experts (MoE) method. The model with the Mixture-of-Experts (MoE) method (Idea
     B) is the model with the Mixture-of-Experts (MoE) method. The model with the
    Mixture-of-Experts (MoE) method (Idea B) is the model with the Mixture-of-
    Experts (MoE) method.\nThe model with the Mixture-of-Experts (MoE) method (
    Idea B) is the model with the Mixture-of-Experts (MoE) method. The model with
    the Mixture-of-Experts (MoE) method (Idea B) is the model with the Mixture-of-
    Experts (MoE) method.\nThe model with the Mixture-of-Experts (MoE) method (
    Idea B) is the model with the Mixture-of-Experts (MoE) method.\nAnswer: 0\n</
    think>"

"prediction": 1,

"ground_truth": 1
```


G.3 QWEN3-8B-REASON-SFT-DRGRPO

As mentioned in §(5.3), the RL trained model using Dr. GRPO started generating very minimal number of think tokens, often just a direct statement of which idea is better and without justifying "why" this leads to better performance.

Input


```
"research_goal": "The primary research objective is to develop computational
    models capable of robust common sense reasoning, specifically for the task
    of pronoun coreference resolution in ambiguous contexts. This research
    utilizes the WinoGrande benchmark, which presents models with a sentence
    containing a pronoun and requires them to select the correct antecedent
    from two plausible options, a task designed to necessitate real-world
    knowledge. Performance is measured by classification accuracy, where the
    adversarially-filtered construction of the dataset ensures that success is
    indicative of genuine reasoning rather than reliance on superficial dataset
     artifacts. Advancing model capabilities on this benchmark is crucial for
    building AI systems that can comprehend and reason about nuanced human
    language."
```

```
"idea_A": "A prefix-guided multi-task learning framework is proposed to explore
     inter-task relationships and jointly train a model on numerous natural
    language understanding (NLU) tasks. The methodology begins by unifying all
    tasks into a multiple-choice-like format, where each training example
    consists of a context, a question, and a fixed number of `k` candidate
    options. To manage variability in the original datasets, examples with more
     than `k` options have redundant options randomly discarded, while those
    with fewer are padded with 'N/A' placeholders. Each resulting data instance
     is then formatted into a sequence by prepending a unique, task-specific
    prefix token, resulting in an input structure of `{[Prefix]: context,
    question, option}`. The model, based on an encoder-only architecture, is
    trained using a dual-objective function `L = L_mtl + \u03bbL_mlm`, where
    `\u03bb` is a balancing hyper-parameter. The first objective, `L_mtl`, is
    a supervised task loss where the model functions as a discriminator `g(\
    u2022, \u2022)`. For each example `(c_i, q_i)` with a set of options `r`,
    the model computes a matching score `g(c_i, q_i \u2022 r_j)` for each
    option `r_j`, and the model is trained with a classification loss to select
     the option with the highest score. The second objective, `L_mlm`, is a
    self-supervised denoising loss based on masked language modeling (MLM),
    defined as `L_mlm = -\u03a3_{i=1 to N} \u03a3_{j\u2208M} log p_\u03b8(t_{i,
    j} | x_i')`, where `x_i'` is a sequence with randomly masked tokens and `M
    ` is the set of masked indices. A key mechanism of this framework is that
    the MLM objective is applied to the entire input sequence, including the
    task-specific prefix tokens, which are also subject to random masking. By
    requiring the model to predict the correct prefix based on the content of
    the data instance, it is compelled to learn prefix embeddings that capture
    task-specific differences and common patterns. Consequently, the learned
    vector representations of these task prefixes serve as a direct means to
    probe and quantify the relationships between the various tasks."

"idea_B": "A parameter-efficient fine-tuning method is proposed that constructs
     a sparse Mixture-of-Experts (MoE) model by augmenting a frozen, pre-
    trained dense transformer. This is achieved by replacing the standard feed-
    forward network (FFN) layer in each transformer block with a specialized
    MoE block. This block consists of a set of experts and a top-k gating
    router. A key architectural principle is that the experts are not
    independent neural networks; instead, they are formed by combining a single
    , shared, frozen FFN from the base model with unique, expert-specific Low-
    Rank Adaptation (LoRA) matrices. Specifically, each of the `n` experts
    consists of the shared FFN weights `W` and its own set of trainable LoRA
    matrices, `B_k` and `A_k`, for each linear layer within the FFN. During
    the forward pass, for a given hidden state `h`, a trainable linear router
    `R` computes scores for each expert. A top-k gating mechanism then selects
    the `k` experts with the highest scores. The output of the MoE block is a
    weighted sum of the outputs from these selected experts, where the weights
    are the normalized scores from the router. The output of a single expert `
    E_k` for an input `h` is the result of the shared FFN computation,
    modified by the expert's specific LoRA update. The forward pass for the MoE
     block is defined as `MixLoRA(h) = \u03a3_{k=1 to K} R(h)_k * E_k(h)`,
    where `R(h)_k` is the router's weight for the k-th selected expert. The
    router itself uses a softmax function over the outputs of a linear layer `
    W_r` to generate a probability distribution, from which the top `k`
    experts are chosen: `R(h) = KeepTop-k(Softmax(W_r * h))`. To counteract the
     tendency for routers to disproportionately favor certain experts, an
    auxiliary load balancing loss is incorporated into the total training loss.
     This loss is calculated as `L_aux = \u03b1 * N * \u03a3_{i=1 to N} F_i *
    P_i`, where `N` is the number of experts, `F_i` is the fraction of tokens
    in a batch dispatched to expert `i`, `P_i` is the average router
    probability allocated to expert `i`, and `\u03b1` is a scaling coefficient
    . To further enhance model adaptation, the architecture also integrates
    separate, independent LoRA adapters into the query, key, value, and output
    projection matrices of the self-attention layers, which are trained
```

```
    concurrently. A computational optimization is also employed where, for the
    FFN's parallel linear projections (e.g., `W1` and `W3` in a LLaMA-style
    FFN), the computation is performed only once on the input sequence. The
    resulting activation tensors are then sliced and routed to the respective
    expert-specific LoRA computations, thus avoiding redundant calculations
    across the shared FFN components."
```

Output

```
"thinking": "<think>\nidea_A is a prefix-guided multi-task learning framework
    that jointly trains a model on numerous natural language understanding (NLU
    ) tasks, which gives it an advantage over idea_B, which is a parameter-
    efficient fine-tuning method.\n</think>",

"prediction": 0

"ground_truth": 0
```