Forecasting Research Success through Learned Comparison of Scientific Ideas

A Thesis

submitted to
Indian Institute of Science Education and Research Pune
in partial fulfillment of the requirements for the
BS–MS Dual Degree Programme
by

Srujan Prakash Mule

IISER PUNE

Indian Institute of Science Education and Research Pune

Dr. Homi Bhabha Road,

Pashan, Pune, MH, India, 411008.

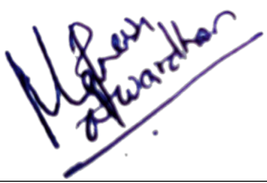
April, 2026

Supervisor: Dr. Manasi Patwardhan

Principal Scientist, TCS Research and Innovation Lab, Pune

Certificate

This is to certify that this dissertation entitled **“Forecasting Research Success through Learned Comparison of Scientific Ideas”** towards the partial fulfillment of the BS–MS dual degree programme at the Indian Institute of Science Education and Research, Pune represents study/work carried out by **Srujan Prakash Mule** at the TCS Research and Innovation Lab, Pune, under the supervision of **Dr. Manasi Patwardhan**, Principal Scientist, TCS Research and Innovation Lab, Pune, during the academic year 2025-2026.



Dr. Manasi Patwardhan

Principal Scientist, TCS Research and Innovation Lab, Pune

Supervisor

Declaration

I hereby declare that the matter embodied in the report entitled **“Forecasting Research Success through Learned Comparison of Scientific Ideas”** are the results of the work carried out by me at at TCS Research and Innovation Lab, Pune, under the supervision of **Dr. Manasi Patwardhan**, Principal Scientist, TCS Research and Innovation Lab, Pune, and the same has not been submitted elsewhere for any other degree. Wherever others contribute, every effort is made to indicate this clearly, with due reference to the literature and acknowledgement of collaborative research and discussions.

Srujan Prakash Mule

20211245

This thesis is dedicated to all those people that travel the road less common in life and the supportive ones who stood by them throughout the uncertain journey.

And my parents who shaped me into the person I am today.

Contents

List of Figures

List of Tables

Acknowledgements

I express my deepest gratitude to my supervisor, Dr. Manasi Patwardhan, for her continuous guidance and invaluable discussions throughout this project. I also extend my thanks to TCS Research for providing the necessary computational resources and institutional support. I am grateful to Dr. Bedartha Goswami for his keen interest and engagement with this work as an expert evaluator.

Special thanks are owed to Aniketh for the overarching discussions, support, help through feedback and his assistance with the figures and review (sometimes even over sleepless nights). Financial support from the Infosys Foundation, INSPIRE SHE, and the Xytel fellowship is gratefully acknowledged. Finally, I thank my family and friends for their unwavering support and encouragement throughout this journey.

Contribution

Table 1: Summary of contributions using the CRediT taxonomy.

Role	**Contributor(s)**
Conceptualization	Supervisor, Aniketh, Author
Methodology	Author, Aniketh, Supervisor
Software	Author*
Validation	Author
Formal analysis	Author
Investigation	Author
Resources	TCS Research
Data curation	Author
Writing – original draft	Author
Writing – review & editing	Author, Aniketh, Supervisor
Visualization	Author, Aniketh
Supervision	Supervisor
Project administration	Supervisor
Funding acquisition	NA

AI-Usage

This work was primarily human-created. AI was used to make stylistic edits, such as changes to structure, wording, and clarity. AI was prompted for its contributions, or AI assistance was enabled. AI-generated content was reviewed and approved.

AIA Primarily human, Stylistic edits, Human-initiated, Reviewed v1.0

Detailed Usage:

- ***Code assistance and debugging:** Code generation, refactoring and debugging support were obtained using GitHub Copilot (Claude Sonnet-4.6, Opus-4.6/4.5, Gemini-2.5/3 Pro) and similar assistant tools for some scripts in the accompanying codebase. All such code was manually inspected line-by-line, tested and verified, and where needed, modified by the author.

- **Language and stylistic suggestions:** Claude Sonnet-4.6, GPT-5 and Gemini-3 Pro were used to suggest alternative phrasings, improve clarity, and correct grammar in selected sentences and paragraphs across the entire document. The underlying ideas, arguments and structure were authored by humans, and every AI-suggested edit was reviewed and, where necessary, revised before inclusion.

- **LaTeX structure and tables:** Gemini-2.5/3 Pro was used to suggest LaTeX table structures and to debug LaTeX formatting issues.

- NotebookLM and Deep Research capabilities of Gemini-2.5/3 Pro were used to search literature, summarise and explain research papers, clarify background concepts and brainstorm.

Abstract

As generative language models (LMs) accelerate scientific research by automating hypothesis generation, a new bottleneck emerges: evaluating and filtering hundreds of LM generated ideas without exhaustive experimentation. This work asks whether LMs can learn to judge the empirical success of research ideas before any experiments are run. This thesis studies *comparative empirical forecasting*: given a benchmark-specific research goal and two candidate ideas, predict which will achieve better leaderboard performance. A dataset of 11,488 idea pairs grounded in objective outcomes from PapersWithCode is created for this task. While untrained 8B-parameter models struggle ($\approx$30% accuracy), Supervised Fine-Tuning dramatically boosts performance to 77.1%, significantly outperforming frontier models like GPT-5 (61.1%). By framing evaluation as a reasoning task via Reinforcement Learning with Verifiable Rewards, models are trained to discover latent reasoning paths, achieving 71.35% accuracy with interpretable justifications. Crucially, these RL-trained variants demonstrate superior cross-domain generalization, achieving 67.49% on an independent test set and surpassing a zero-shot retrieval-augmented GPT-4.1 system by 16 percentage points. These results demonstrate that compute-efficient small language models can show potential as effective, objective verifiers, offering a scalable path for autonomous scientific discovery.

Chapter 1

Introduction

Recent advances in large language models have begun to transform the role they can play in the scientific process. Formulating research hypotheses requires navigating constraints, assumptions, and complex goals. Given ideation is largely expressed and iterated through natural language, it finds potential to benefit from the recent advances in large language models (LLMs) [Wang et al., 2024; Baek et al., 2025; Si et al., 2024]. Language models are increasingly capable of performing multiple stages of the research pipeline, including generating hypotheses, proposing experimental designs, implementing candidate methods, and synthesizing findings across large bodies of literature. Many recent efforts focus on leveraging frontier LLMs for ideation via retrieval [Li et al., 2024], test-time compute [Hu et al., 2024], or multi-agent debate [Su et al., 2025]. Few works fine-tune open-source models for better research ideation [O'Neill et al., 2025].These developments have given rise to a new class of systems often described as *AI research agents*: autonomous or semi-autonomous frameworks that attempt to automate significant portions of the scientific discovery process [Lu et al., 2024; Yamada et al., 2025; Gridach et al., 2025; Skarlinski et al., 2024].

A defining capability of these systems is *high-throughput ideation*: for a single scientific or research goal, a model may generate hundreds of potential methods, architectural variations, or experimental strategies rapidly exploring the vast hypothesis spaces [Baek et al., 2025; Si et al., 2024; Garikaparthi et al., 2025]. Such breadth of exploration has the potential to substantially expand the search space of scientific discovery. However, it also introduces a new constraint. While candidate ideas can be generated almost without limit, evaluating them requires resources, experimentation and time at a scale that is often infeasible. Consequently, the bottleneck in AI-driven scientific discovery is no longer the generation of ideas but their efficient validation of ”good ideas”. This thesis refers to this phenomenon as the *validation bottleneck*: the growing gap between the rate at which candidate research ideas can be produced and the rate at which they can be verified through actual implementation.

Current approaches of evaluating ideas rely primarily on language-model-as-judges over subjective criteria such as “excitement,” “innovativeness,” or “novelty” [Wang et al., 2024; Baek et al., 2025; Hu et al., 2024]. In practice, these judgments are operationalized through rubric-based evaluation schemes. Ideas may be assigned absolute ratings using Likert-style scales [Baek et al., 2025], or compared pairwise and aggregated into rankings using mechanisms such as ELO scoring [Si et al., 2024; Garikaparthi et al., 2025]. Some studies additionally calibrate these evaluations against human expert judgments [Si et al., 2024], while others incorporate literature retrieval mechanisms to determine whether a proposed idea is genuinely novel relative to prior work [Moussa et al., 2025]. More recently, several works have attempted to improve such evaluations by fine-tuning language models to approximate these rubric-based assessments [Goel et al., 2025; Gunjal et al., 2025].

While helpful, such subjective metrics are often just proxies. Empirical studies increasingly report that LLM-generated ideas which score highly under subjective evaluation rubrics, often surpassing human generated ideas [Si et al., 2024], they frequently fail to translate into real world improvements [Zhu et al., 2025b; Si et al., 2025]. The underlying issue is not simply imperfect measurement but a deeper misalignment between the evaluation signal and the ultimate objective of scientific research. A subjective judge naturally optimizes for plausibility and rhetorical coherence, whereas scientific progress (in many fields) is ultimately determined by empirical outcomes. As a result, refining the rubric alone cannot resolve the problem of the mismatch between how promising an idea appears in natural language and how well it performs in practice. What is needed is therefore a verifier grounded directly in *empirical outcomes*, rather than in how an idea reads on paper.

There is reason to believe that such outcome-grounded verification may be feasible. A growing body of work has shown that language models can serve as effective forecasters of real-world events, sometimes approaching the accuracy of competitive human crowd forecasters across diverse domains [Halawi et al., 2024; Karger et al., 2025]. Further research has shown that specialised training procedures can substantially improve both the accuracy of such forecasts [Lee et al., 2025; Chandak et al., 2025]. In parallel, predictive signals about scientific progress itself have been extracted from the structure of the research literature, with high-impact research directions shown to be forecastable from evolving knowledge graphs that capture the dynamics of scientific discovery [Gu and Krenn, 2025].

Most directly related to the present work are recent attempts to predict empirical machine learning outcomes directly from textual descriptions of research methods. Wen et al. construct pairs of research ideas with known experimental outcomes and fine-tune GPT-4.1 to predict which idea will perform better. Similarly, Park et al. demonstrate that language models can estimate benchmark scores from redacted textual descriptions of tasks and configurations without executing any experiments. Together, these results suggest that empirical outcomes contain predictive signals that language models may be able to exploit.

However, important limitations remain in existing approaches. The dataset construction process of Wen et al. assigns comparison labels by majority voting across multiple benchmarks, which can conflate evaluations and obscure the task-specific context that often determines whether a method succeeds. Empirical performance in machine learning is highly context-dependent: the same approach may dominate one benchmark and underperform on another because different tasks reward fundamentally different properties. Aggregating results across benchmarks therefore risks masking the very signals that determine empirical success. In addition, prior work relies primarily on frontier proprietary models and does not investigate whether compute-efficient, open-source models can achieve comparable forecasting capabilities after targeted fine-tuning. Finally, existing systems provide limited insight into the reasoning processes underlying their predictions, leaving open the question of whether the models have learned meaningful comparative priors or simply exploit superficial correlations and potential for a transparent look at their reasoning.

1.1 Goal and Scope of This Thesis

This thesis addresses the validation bottleneck by investigating whether language models can be *trained* to forecast the empirical success of research ideas before any experiments are run. The central task studied is *comparative empirical forecasting*: given a benchmark-specific research goal and two candidate research ideas, predict which idea will likely achieve better empirical performance when evaluated on that benchmark.

Unlike rubric-based evaluation, this formulation defines an objective prediction target grounded in measured empirical outcomes. It also corresponds directly to the decision faced by both human researchers and automated research agents when selecting which idea to pursue experimentally.

The approach is motivated by an important observation: although empirical outcomes are genuinely difficult to predict, research doesn't always follow purely deductive logic, and the reasons for a result are often clearest only in hindsight. Yet, researchers routinely develop useful intuitions through experience from patterns across prior work. The central question thus addressed in this thesis is whether a language model can be trained to internalize such comparative priors and discriminate between two competing research ideas *before* running experiments.

A verifier with this capability could play an important role in scaling AI-driven scientific discovery. Instead of blindly executing large numbers of candidate experiments, an automated research system could first filter generated ideas according to their predicted empirical potential. Only the most promising candidates would then be implemented and evaluated, substantially reducing the computational cost of high-throughput ideation systems and directly addressing the validation bottleneck.

The work is scoped to NLP research ideas as expressed through their published implementations and evaluated against objective benchmark leaderboards from PapersWithCode.Domains where performance cannot be captured by a measurable scalar objective fall outside the scope of the present work. This work focuses on fine-tuning open-source 8B-parameter language models rather than relying on proprietary frontier models, keeping the approach both compute-efficient and broadly reproducible. Additionally, ranking across more than two ideas (for example, full leaderboard re-ranking) is explored to assess their capability as idea filters.

To investigate whether language models can serve as effective outcome-grounded verifiers for research ideas, this thesis studies the comparative empirical forecasting task introduced above along several dimensions. These include the predictive ability of fine-tuned models, their ability to generalize beyond the domain or dataset type on which they are trained, the role of explicit reasoning supervision during training, the robustness of the resulting predictions and finally potential of such trained models as idea filters.

This gives rise to the following research questions.

1.2 Research Questions

To investigate this problem systematically, the present work is organized around the following research questions.

RQ1: Empirical Forecasting Ability Can a small language model, fine-tuned on benchmark specific idea pairs, predict empirical outcomes more accurately than untrained (i.e.zero-shot) frontier models ? This question establishes whether the task requires frontier large language models or if computationally efficient small LMs fine-tuned for this task can do well. (§3.2).

RQ2: Generalization and Transferability Do trained models generalize beyond the NLP domain on which they are trained? Generalization is evaluated on two held-out settings: (i) a cross-domain test set constructed from non-NLP leaderboards, and (ii) the independently constructed test set of Wen et al., whose dataset and label construction pipeline bear no methodological overlap with the present work. Strong performance on the latter, in a zero-shot transfer setting, would constitute evidence that trained models capture transferable comparative reasoning priors rather than domain-specific shortcuts (§3.2).

RQ3: The Role of Reasoning Does framing comparative empirical forecasting as a reasoning task i.e. by training/asking models to generate chain-of-thought/rubrics traces before picking the better idea improve prediction accuracy over direct label-only fine-tuning or prediction? Two forms of reasoning supervision are compared: (i) grounded traces derived from source papers and (ii) synthetic traces distilled from a larger frontier model (GPT-5), inspired by RM-R1 approach [Chen et al., 2026]. This work hypothesizes that allowing a model to deliberate before answering allows it room to verify or reflect translating to better accuracies. Does supervising or fine tuning using grounded, paper-derived traces constitute a stronger supervision signal for this task, since they capture the specific empirical reasoning patterns that led to observed outcomes? (§3.3).

RQ4: RL-Induced Reasoning. Can Reinforcement Learning with Verifiable Rewards (RLVR), using the binary outcome label as the reward signal, induce structured and interpretable reasoning in a small language model for this task? The hypothesis is that RLVR, when applied to a well initialized model, allows the model to discover latent reasoning paths that support correct predictions. (§3.3).

RQ5: Robustness to Presentation Variation. Are model predictions robust to surface-level variation in how ideas are presented, or do they reflect brittle heuristics? Four potential confounds are probed: paraphrasing of the idea text, the presentation order of the two ideas (position bias), the length of the idea description (length bias), and whether the more recently published idea occupies a favored position (recency bias). All comparisons are assessed with bootstrapped significance testing . Do they perform well under sycophantic pressure? (§3.4)

RQ6: Idea Filters Finally test how well such fine-tuned model perform under real world scenarios. Can fine-tuned models be used to filter and rank ideas based on comparative evalution? If the trained models are robust or good predictors, one can perform pairwise comparisons of all ideas to rank them. The quality of such "rankers" can be assessed using Top-1 accuracy and Median Rank RMSE(§3.4).

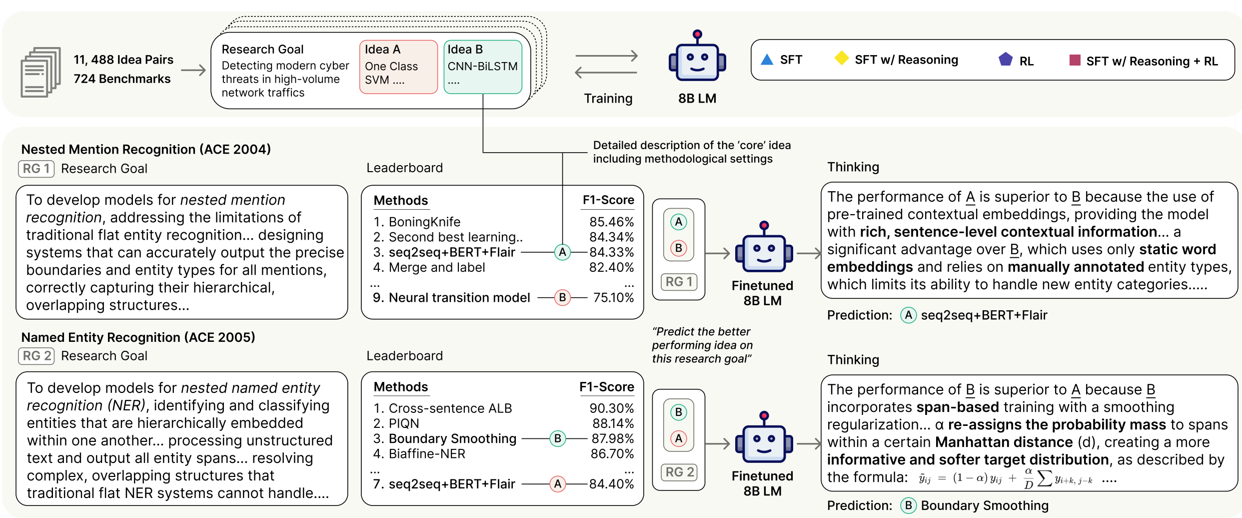


Figure 1.1: *Overview* The figure illustrates the potential of the fine-tuned model to accurately judge the quality of scientific ideas. For the two given benchmarks and an overlapping method, the model robustly predicts which idea will perform better in context of the benchmark, while providing insightful reasoning. *Image created with help from Aniketh.*

Chapter 2

Background, Dataset, and Methodology

2.1 Problem Statement

The problem is formally defined as follows: Let $\mathcal{H}$ be the space of scientific hypotheses (ideas), $\mathcal{G}$ be the space of research goals and $\mathcal{C}$ be the space of reasoning traces. A dataset $\mathcal{D} = \{g, h_A, h_B, (c), y\}$, where both h_A and $h_B \in \mathcal{H}$, are textual descriptions of two competing ideas, $g \in \mathcal{G}$ is the specific research goal for which the ideas are implemented (e.g using One class SVM vs CNN-BiLSTM with the goal of increasing accuracy in detecting modern cyber threats in high-volume network traffic while minimizing false positives), and $y \in \{0, 1\}$ is a binary label where $y = 0$ implies that h_A performs better than h_B on goal g and $y = 0$ implies visa versa. $c \in \mathcal{C}$ denotes an optional chain-of-thought explaining why one idea outperforms the other. The objective is to learn a parameterized policy π_θ (i.e. train a Language Model) that accurately predicts y given the context of the ideas and the goal, while generating chain of thought reasoning trace c before prediction.

These outcomes are genuinely hard to predict. Research rarely follows purely deductive logic: the "why" behind a benchmark result is often clearest only in hindsight, and empirical success is highly task-dependent: the same method can dominate one benchmark and underperform on another because different tasks reward fundamentally different algorithmic properties. A predictive model must therefore reason not about whether an idea is interesting in the abstract, but about whether it is likely to work *for this specific goal*.

2.2 Background

2.2.1 Fine-Tuning Language Models

Modern language models are trained in two stages. In *pre-training*, given a token sequence $x = (x_1, \ldots, x_T)$, an auto-regressive model learns parameters θ by maximizing the next-token log-likelihood:

$$\max_{\theta} \sum_{t=1}^{T} \log p_{\theta}(x_t \mid x_{<t}). \tag{2.1}$$

This objective yields powerful general representations but does not ensure that the model follows instructions, produces outputs in a desired format, or optimizes an external notion of correctness. *Fine-tuning* addresses this gap by further adapting the pre-trained parameters using either supervised signals or direct optimization of a scalar reward.

In many settings, and throughout this thesis, fine-tuning is performed using *parameter-efficient* methods. The motivation rests on an empirical finding: pre-trained language models occupy a space of far lower intrinsic dimensionality than their full parameter count suggests. Aghajanyan et al. [2021] demonstrate that common pre-trained models can be effectively fine-tuned by optimizing as few as 200 freely-varying parameters reparametrized back into the full weight space, achieving 90% of full fine-tuning performance on standard benchmarks. Larger pre-trained models tend to have even lower intrinsic dimension, which in part explains why parameter-efficient methods scale particularly well to billion-parameter models.

Low-Rank Adaptation (LoRA). LoRA [Hu et al., 2021] operationalizes parameter-efficient fine-tuning by restricting weight updates to a low-rank subspace. For a linear layer with weight matrix $W \in \mathbb{R}^{d \times k}$, LoRA learns a rank-$r$ update ΔW factored as:

$$W' = W + \Delta W, \qquad \Delta W = BA, \qquad B \in \mathbb{R}^{d \times r},\ A \in \mathbb{R}^{r \times k},\ r \ll \min(d, k), \tag{2.2}$$

optionally scaled by a constant factor α / r. Matrix A is initialized from a zero-mean Gaussian distribution and B is initialized to zero, so $\Delta W = 0$ at the start of training and the adapted model is identical to the pre-trained model before any gradient steps are taken. Only A and B are updated; the original weight matrix W remains frozen. This reduces trainable parameters by a factor of approximately $dk/(r(d+k))$. For an 8B-parameter model with layer dimension $d = k = 4096$ and rank $r = 64$, this corresponds to a reduction factor of approximately 32, reducing per-layer trainable parameters from roughly 16 million to 500 thousand while leaving the total parameter count at inference time unchanged.

2.2.2 Supervised Fine-Tuning

Supervised fine-tuning (SFT) adapts a pre-trained model to a target distribution of outputs by maximizing likelihood on a labeled dataset. It is the standard first stage in instruction-tuning and RLHF-style pipelines [Ouyang et al., 2022]. Let $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^{N}$ be supervised examples, where x is an input prompt and $y = (y_1, \ldots, y_{|y|})$ is a reference output sequence. Under teacher forcing, the SFT objective is the cross-entropy loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\,\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{|y|} \log p_\theta(y_t \mid x,\, y_{<t})\right]. \tag{2.3}$$

SFT is attractive because it is stable, sample-efficient, and serves as a natural starting point when ground-truth labels are available. A key limitation in the present context, however, is that when the target sequence is a bare binary label, the model learns to predict outcomes but not to articulate *why* one idea is expected to outperform another. The intermediate reasoning process is invisible to the loss function unless explicit reasoning traces are included in the supervised targets. This limitation motivates the two-stage training approach developed in Sections 2.5.2 and 2.5.3.

2.2.3 Reinforcement Learning for Language Models

Reinforcement learning (RL) fine-tuning optimizes a language model directly for a ”reward” that evaluates the quality of generated outputs. This is useful when the target objective is difficult to encode purely via supervised labels. For example, when the goal is to produce structured reasoning traces before a final decision, or to optimize an outcome defined by an external verifier.

The key advantage of the present task is that the ground-truth label $y \in \{0, 1\}$ provides a directly verifiable reward signal: a prediction is either correct or not, with no ambiguity requiring human judgment. This property characterizes *Reinforcement Learning with Verifiable Rewards* (RLVR), which is conceptually cleaner than conventional Reinforcement Learning with Human Feedback (RLHF). In RLHF, a separate reward model must be trained on human preference judgments and can itself be exploited through reward hacking [Amodei et al., 2016]. For tasks with objectively correct answers, RLVR sidesteps these complications [Shao et al., 2024].

Let the model define a policy π_θ over output sequences $o = (o_1, \dots, o_T)$ conditioned on a prompt x. RL seeks to maximize expected reward:

$$\max_\theta \; \mathbb{E}_{x\sim\mathcal{D},\, o\sim\pi_\theta(\cdot|x)}\big[R(x, o)\big]. \tag{2.4}$$

Unconstrained reward maximization can cause reward hacking and undesirable distributional shift away from the pre-trained language model [Amodei et al., 2016]. A standard stabilizer is to regularize toward a reference policy π_{ref} (typically the SFT checkpoint) using a KL penalty:

$$\max_\theta \; \mathbb{E}\big[R(x, o)\big] \; - \; \beta\, \mathbb{E}_x\Big[\mathrm{KL}\big(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)\big)\Big], \tag{2.5}$$

where $\beta > 0$ controls how strongly updates are constrained near the reference. Optimization is performed using policy-gradient methods. Using the factorization $\log \pi_\theta(o \mid x) = \sum_{t=1}^{T} \log \pi_\theta(o_t \mid x, o_{<t})$, a basic gradient estimator takes the form:

$$\nabla_\theta\, \mathbb{E}[R] = \mathbb{E}\left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(o_t \mid x, o_{<t})\; \big(R(x, o) - b(x)\big)\right], \tag{2.6}$$

where $b(x)$ is a variance-reduction baseline.

Group Relative Policy Optimization (GRPO)

Proximal Policy Optimization (PPO) [Schulman et al., 2017] stabilizes policy-gradient updates via a clipped surrogate objective. Applying PPO to language models requires training a value function (critic), which can be expensive and unstable at the scales of modern LLMs. Group Relative Policy Optimization (GRPO) is a PPO-style alternative that avoids an explicit critic by estimating advantages *relatively* within a group of sampled completions for the same prompt [Shao et al., 2024; Liu et al., 2025b].

Given a prompt x, GRPO samples G outputs $\{o_1, \ldots, o_G\}$ from π_θ. Each output obtains a reward $R(o_i)$, and the group-relative advantage is computed by standardizing within the group:

$$A_i = \frac{R(o_i) - \mu_R}{\sigma_R + \delta}, \qquad \mu_R = \frac{1}{G}\sum_{j=1}^{G} R(o_j), \quad \sigma_R = \sqrt{\frac{1}{G}\sum_{j=1}^{G}(R(o_j) - \mu_R)^2}, \tag{2.7}$$

where $\delta > 0$ is a small numerical stability constant. This group-relative standardization serves as a self-normalizing baseline: outputs better than the group mean receive a positive advantage signal and outputs worse than the mean receive a negative one, without any external value estimate. Defining $\rho_{i,t} = \pi_\theta(o_{i,t} \mid x, o_{i,<t}) / \pi_{\theta_{\text{old}}}(o_{i,t} \mid x, o_{i,<t})$ as the probability ratio between the current and old policies, the clipped GRPO objective is:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x,\{o_i\}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\!\big(\rho_{i,t}A_i,\ \text{clip}(\rho_{i,t}, 1-\epsilon, 1+\epsilon)\,A_i\big)\right]. \tag{2.8}$$

The clip operator restricts probability ratio changes to $[1-\epsilon, 1+\epsilon]$, preventing excessively large gradient steps that destabilize the policy.

DAPO

Standard GRPO with symmetric clipping suffers from two failure modes in practice [Yu et al., 2025]. The first is *entropy collapse*: the symmetric clip ceiling at $1+\epsilon$ limits how much the probability of any token can increase in a single update step. Low-probability "exploration" tokens, those that could unlock novel reasoning paths, are effectively frozen because they cannot breach this ceiling. Over successive updates, training preferentially reinforces already high-probability tokens, and the policy distribution collapses into a narrow, low-entropy mode. For a binary classification task this manifests as the model assigning near-certain probability to one class regardless of input content. The second is *length bias* arising from the per-sequence normalization $\frac{1}{|o_i|}$, which creates an asymmetric gradient signal. For a correct response (positive advantage), shorter outputs receive proportionally larger gradient updates, subtly rewarding brevity. More critically, for an incorrect response (negative advantage), the normalization attenuates the penalty applied to longer wrong outputs: a verbose but incorrect response is penalized less than a concise incorrect one, because the negative gradient is divided by the larger $|o_i|$. In practice, the dominant observed failure is the second effect: GRPO models tend to generate progressively longer, repetitive incorrect responses, as these verbose wrong outputs accumulate only a weakly negative gradient [Liu et al., 2025b].

DAPO [Yu et al., 2025] addresses both failure modes. The *clip-higher* modification uses an asymmetric clipping interval $[1-\epsilon, 1+\epsilon_{\text{high}}]$ with $\epsilon_{\text{high}} > \epsilon$. By raising only the upper bound, DAPO gives low-probability tokens more room to increase their probability across update steps, maintaining policy entropy and preserving exploration, while keeping the lower bound tight to prevent suppressing any token entirely. Global token normalization replaces per-sequence normalization with a group-level denominator $\frac{1}{\sum_{j=1}^{G}|o_j|}$, applying equal weight to every token across all responses in the group regardless of individual response length, removing the asymmetric gradient bias that under-penalized long incorrect outputs. The DAPO objective is:

$$\mathcal{J}_{\text{DAPO}}(\theta) = \mathbb{E}_x\left[\frac{1}{\sum_{j=1}^{G}|o_j|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\min\Big(\rho_{i,t}\,A_i^{\text{dapo}},\ \text{clip}\big(\rho_{i,t},\ 1-\epsilon,\ 1+\epsilon_{\text{high}}\big)\,A_i^{\text{dapo}}\Big)\right], \tag{2.9}$$

subject to the dynamic sampling constraint $0 < \sum_{i=1}^{G}\mathbb{I}(\hat{y}_i = y) < G$, which discards groups in which all outputs are correct or all are incorrect, as such groups contribute zero gradient signal. The advantage is standardized as:

$$A_i^{\text{dapo}} = \frac{R(o_i) - \text{mean}(\{R(o_j)\}_{j=1}^{G})}{\text{std}(\{R(o_j)\}_{j=1}^{G}) + \delta}. \tag{2.10}$$

Dr. GRPO

The standard deviation term $(\sigma_R + \delta)^{-1}$ of GRPO's advantage estimator has an unintended consequence: when all outputs in a group receive similar rewards (occurrs when the model has converged to a near-optimal policy for a given difficulty level) σ_R approaches zero and the normalized advantages become numerically large, producing unstable gradient updates. This instability is particularly pronounced for the present task, where reward variance within a group can become low once the model reaches a high-accuracy regime.

Dr. GRPO [Liu et al., 2025b] removes both the per-sequence length normalization and the standard deviation from the advantage estimator, using the raw mean-centered advantage:

$$A_i^{\mathrm{dr}} = R(o_i) - \mathrm{mean}(\{R(o_j)\}_{j=1}^{G}), \tag{2.11}$$

and sums gradients over all tokens in the group rather than averaging by sequence length. The Dr. GRPO objective is:

$$\mathcal{J}_{\mathrm{Dr}}(\theta) = \mathbb{E}_x \left[\frac{1}{G} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \min\big(\rho_{i,t}\, A_i^{\mathrm{dr}},\ \mathrm{clip}(\rho_{i,t}, 1 \pm \epsilon)\, A_i^{\mathrm{dr}}\big) \right]. \tag{2.12}$$

By aggregating gradient contributions from all tokens in the group, Dr. GRPO provides a more stable training signal across groups with varying reward distributions, complementing DAPO's entropy-preservation properties.

2.3 Benchmark Dataset

The central goal is to predict the better idea, based on it's likely empirical outcome, of two competing ideas when applied to a specific research goal. To enable this study a pipeline to construct a benchmark dataset of idea pairs, transforming raw leaderboards into statistically grounded comparisons is developed. A sample in the benchmark dataset consists of:

- **Idea Pair:** Detailed descriptions of two competing methods ($idea_A$, $idea_B$), grounded in their scientific publications.
- **Research Goal:** A clear statement of the specific evaluation objective of a benchmark for which the ideas are implemented
- **Binary Label:** A label (0 or 1) indicating which idea achieved a higher empirical score on that specific benchmark.

The dataset construction process involves:

Scraping and Paper Collection. Ideas from entries in live leaderboards are extracted, which allows one to build comparisons specific for each benchmark. Thus the evaluation becomes more fine-grained in comparison to parallel work Wen et al. [2025], which can potentially conflate eval-

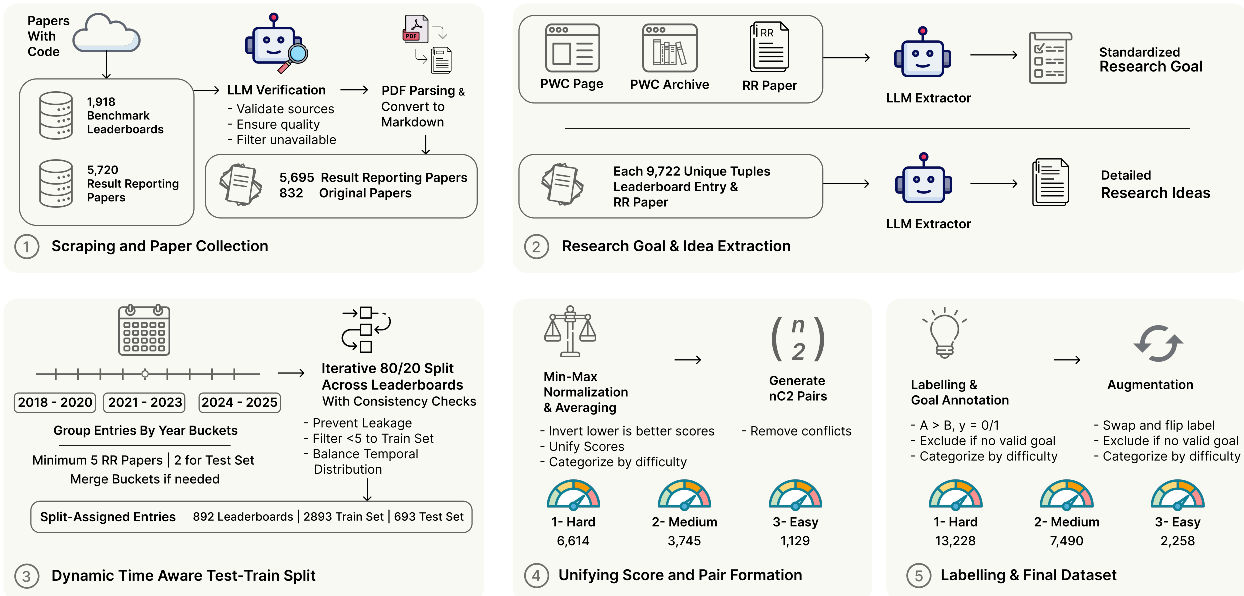


Figure 2.1: **Dataset Construction Pipeline** Raw entries from 1,918 NLP leaderboards are used to construct a statistically grounded idea pairs with a benchmark specific research goal, while difficulty stratification ensures robust evaluation across diverse research goals. *Figure made with help from Aniketh*

uations due to aggregation of scores across various benchmarks via majority voting.

First scrape all available NLP leaderboards from `paperswithcode.com` that have at least two entries are scraped. This yields 1,918 benchmark leaderboards. For each entry in a leaderboard a referenced paper is identified , resulting in 5,713 **Result-Reporting (RR)** papers (excluding 7 behind paywall). In some instances, the RR paper is not the **Original** paper which introduces 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, an LLM (Gemini-2.5-pro, prompt in Appendix A) is prompted 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. Additionally, the LLM is asked to report the confidence of its analysis (high/medium/low). The low-confidence entries are then manually analyzed to verify the identified original paper citation and correct if necessary and label them as high-confidence cases. An additional 908 Original papers based on this analysis is then downloaded. All downloaded papers are parsed using `s2orc-doc2json`[1] to

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

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.

Research Goal and Idea Extraction. For each one of the 1918 leaderboards, a single canonical research goal from official benchmark descriptions is extracted in the following order of sources: (1) the dataset page on paperswithcode.com, (2) the corresponding dataset file from the `pwc-archive`[2], 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 A), which generates a clear, comprehensive research goal including what the benchmark evaluates. 327 such benchmarks with missing or unusable sources are skipped.

Each RR and original markdown paper corresponding to the leaderboard entries are processed 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 A) has access to the complete paper context capturing all necessary details necessary for a comprehensive understanding like algorithms, mathematical details etc., whenever present. This results in 9,722 total ideas across the leaderboards.

Train–Test Split. The train–test division is constructed 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 until they meet this threshold or no more merging is possible. For example, if buckets for 2020–2022 and 2023–2025 are too small, they are merged into a single 2020–2025 bucket.

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

Algorithm 1 Time bucketing and validation

Require: Entries with years and global split.

Create initial year-based buckets.
while bucket has <5 papers or <2 test papers **do**
    Merge bucket with adjacent one.
end while

Ideas in each time bucket are then split in an 80/20 ratio. As one iterates through leaderboards, it is ensured that if an idea is already assigned to a split in a previous iteration, this assignment is strictly maintained regardless of the local splits in the current leaderboard and the corresponding train-test assignment. Any paper that conflicts is moved either from train to test or visa versa. 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. Additionally manual verification of the test set is employd as described below. This process yields 892 leaderboards, with 2,893 ideas in the training set and 693 in the test set.

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

It is observed 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 the necessary details are added.
- **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. Such ideas are removed.

Overall, approximately **88%** of extracted ideas were verified as correct, with ∼**4%** classified as *Incomplete* and ∼**8%** as *Incorrect*, before the final version is obtained. Importantly, the verification criteria explicitly included checking for successful exclusion of empirical results and outcome statements from idea descriptions (lines of exclusion criteria in the extraction prompt). Ideas that contained residual performance comparisons or result statements were corrected or removed accordingly.

Unified Score and Pair Formation Next idea pairs within each benchmark is constructed. 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, a **Unified Score** for every idea within a benchmark is computed. To combine heterogeneous metrics, only the metric columns that were universally reported for all entries in a benchmark are used. If no metric has universal coverage, the benchmark is skipped. For each benchmark, min–max normalization is first applied for the results of each metric, across all entries. Let $m_i^{(k)}$ be metric k for entry i. Normalize is as:

$$\tilde{m}_i^{(k)} = \frac{m_i^{(k)} - \min_k}{\max_k - \min_k} \tag{2.13}$$

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

$$\hat{m}_i^{(k)} = 1 - \tilde{m}_i^{(k)} \tag{2.14}$$

Approximately 85% of benchmarks report only a single metric. If there is more than one metric, the normalized results are averaged across all the metrics (e.g., recall@1, recall@5 and precision), to yield a Unified Score for each idea. The final score s_i is the arithmetic mean across adjusted metrics:

$$s_i = \frac{1}{|M|} \sum_{k \in M} \hat{m}_i^{(k)} \tag{2.15}$$

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, a strict pairwise discordance test is used. A pair (i, j) is discordant if:

$$\text{discordant}(i, j) = \begin{cases} 1 & \text{if } (r_i < r_j \text{ and } s_i > s_j) \\ & \text{or } (r_i > r_j \text{ and } s_i < s_j) \\ 0 & \text{otherwise} \end{cases} \tag{2.16}$$

Algorithm 2 Iterative discordance removal

Require: Entries with ranks and unified scores.

```
Compute discordance fraction f = D / (n choose 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. The standard deviation (σ) of the Unified Scores across all entries in that benchmark is calculated and useed to define normalized score differences (Δ) for each pair. Based on Δ, pairs are categorized 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 $(\text{idea}_A, \text{idea}_B)$, $y = 0$ is assigned 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, leads to a final $\approx 90/10$ train/test split in the end.

	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 2.1: 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. We extract CoT in 2 ways based on 2 opposite observations made in the current literature.

Synthetic RM-R1 [Chen et al., 2026] proposes and shows effectiveness of a *distillation-then-RLVR* pipeline for training reasoning reward models: a smaller model is first trained via SFT on structured "Chain-of-Rubrics" traces distilled from larger teacher, and then refined through RLVR. Inspired by this approach, synthetic Chain-of-Rubrics from GPT-5 (high reasoning) are extracted. A small subset of 2125 idea pairs from train are randomly sampled. GPT-5 is first prompted with the research goal and both idea descriptions, similar to the prediction task, asking the model to produce structured rubric-style reasoning traces that evaluate each idea before concluding which is better. Next, only traces where GPT-5's predicted label matches the ground truth are retained, yielding 1,369 pairs; after swap-augmentation this produces 2,738 training examples.

Literature Grounded 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, a different approach to extract CoT reasoning is taken. All the idea pairs are considered such that both of the ideas within each pair have the same RR paper. This way, it can be ensured that such comparisons actually exist and are presented within the paper and not a case of inferred reasoning. An LLM (Prompt in Appendix A) is prompted 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. This results in a very small set of 170 idea pairs with reasoning traces that is grounded in literature.

Cross domain Test set A similar pipeline as described above is employed 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, GPT-5 with high reasoning is used for idea extraction to introduce linguistic diversity relative to the training distribution. Additionally, the pairs are not categorized 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 (i.e. **sensitivity** to metrics), while testing for their **ability to generalize** to cross domains and taking care of the potential **knowledge-cutoff confound**.

Wen et al.'s Test set The manually verified test set which is not publically available is obtained by contacting them. This dataset was constructed by downloading papers beyond NLP, and extracting 2 ideas from the same paper by prompting an LLM and assigning labels based on majority voting across multiple benchmarks based on the performance each idea being compared. Any ideas (and their RR or Original papers) in the train set that was constructed in this work and Wen et al.'s test set have no overlap. This is ensured through title matching and removal. This results in 1750 idea pairs that have labels based on majority voting across multiple benchmarks. This is augmented similar to approaches described above to obtain 3500 pairs.

Limitations

- Although extensive manual verification is employed for the test set, it is assumed that the leaderboard data present on paperswithcode has no errors of any form for the rest. It is not verified if the empirical results mentioned on the website are exactly what was reported in the associated paper, which can introduce noise.

- LLMs have been used for most part of the dataset creation due to the lack of resource and manpower to process such large data (which also requires one to read and understand thousands of papers). Whenever human intervention becomes crucial, I have intervened and have manually processed or verified things. I have also tinkered with prompts to generate the best and consistent outputs for a given task.

- Paperswithcode has been shut down. We were able to scrap most of the data before it went down. A similar alternative is now up at HyperAI based on the source data of paperswithcode available at GitHub. The full dataset from GitHub is a larger version of the dataset compared to what we scraped. The CD test was created using the GitHub source.

2.4 Models and Experimental Setup

2.4.1 Model Selection

Two open-source 8B-parameter models are evaluated as the primary subjects of fine-tuning: **Qwen3-8B** [Yang et al., 2025], hereafter called just Qwen3, and **Llama3.1-8B-Instruct** [Grattafiori et al., 2024], hereafter called just Llama3.1.

Qwen3 is selected as the primary model for two reasons. First, it features a native `<think>` mechanism that can be toggled on or off at inference time via the `enable_thinking` parameter of the chat template [Yang et al., 2025]. This design enables a direct comparison between direct-prediction and reasoning-mode inference without any architectural changes between variants. Second, the Qwen3 pre-training incorporates substantial scinece and mathematical content, which provides a favorable initialization for a task requiring systematic comparison of complex research methodologies.

Llama3.1 is included as a cross-architecture comparison point. It was post-trained using SFT, rejection sampling, and Direct Preference Optimization (DPO) only, with no dedicated reasoning-specific training stage [Grattafiori et al., 2024]. Its inclusion tests whether performance gains observed with Qwen3 are specific to its reasoning-oriented pre-training or transfer to architectures without a native thinking mode.

For frontier model comparisons, **GPT-5** [OpenAI, 2025] and **Gemini 2.5 Flash** [Comanici et al., 2025] are evaluated in zero-shot mode using the same prompt format as the fine-tuned models. GPT-5 is additionally evaluated at three reasoning effort levels (low, medium, high) to characterize the effect of test-time compute scaling on this task.

2.4.2 Input Prompt Format

All models (fine-tuned and frontier) receive input through the same structured prompt. The system turn instructs the model to act as an expert AI research assistant evaluating two ideas. The user turn provides the research goal g, the description of Idea A (h_A), the description of Idea B (h_B), and an instruction to reason step by step before producing a final answer in the format `Answer: [0 or 1]`, where `1` means Idea B is better and `0` means Idea A is better. This prompt format is shared across all training and evaluation conditions to ensure that performance differences reflect differences in model capability rather than prompt sensitivity. The full template is in Appendix A.

2.4.3 LoRA Configuration and Target Module Selection

LoRA adapters are applied to all weight matrices directly involved in each transformer layer's attention mechanism and feed-forward sub-layer: the query, key, value, and output projection matrices of the attention block (`q_proj`, `k_proj`, `v_proj`, `o_proj`) and the gate, up, and down projection matrices of the feed-forward block (`gate_proj`, `up_proj`, `down_proj`). Hu et al. [2021] identify these as the modules most critical for task-specific adaptation, as they mediate both attention patterns and the model's non-linear transformations. The token embedding matrix and language model head are kept frozen to preserve the model's vocabulary distribution.

A LoRA rank of $r = 64$ with scaling $\alpha = 128$ (effective scale $\alpha/r = 2$) is used for all experiments. This rank is intentionally larger than the $r \in \{4, 8\}$ commonly used for single-task NLP classification, reflecting the additional capacity required: SFT variants must learn decision boundaries across a diverse range of ideas, while RL variants must additionally support the generation of extended reasoning traces. All training uses BF16 mixed precision on a single NVIDIA A100-40GB GPU.

2.5 Training Methodology

2.5.1 Direct Supervised Fine-Tuning

Base language models lack the comparative intuition required to map descriptions of research ideas to their likely empirical outcomes. To bridge this gap, standard SFT is used to train the model tp initially just predict the correct label so that it can learn and do better at this task.

The model receives the research goal g and the idea descriptions h_A, h_B as input. The target output is the binary label $y \in \{0, 1\}$ corresponding to the empirically superior idea, as determined by the Unified Score defined in the dataset construction pipeline. The training objective is the standard SFT cross-entropy loss over the label token:

$$\mathcal{L}_{\text{SFT}} = -\log P(y \mid g, h_A, h_B). \tag{2.17}$$

No reasoning format is enforced or incentivized in this version. Particularly in case of Qwen3, its native ”think” mode is disabled during both training and inference for this variant. This phase utilizes the full train dataset to ground the model in the ” intuition” of identifying successful ideas.

2.5.2 Reasoning SFT

The Direct-SFT model learns *what* to predict but not *why*. Predicting empirical outcomes is reasoning-intensive: the explanatory logic connecting an idea choice to its benchmark performance is rarely deductive and often reflects insights gained only after results are known [Wen et al., 2025]. To seed the model with a structured scientific argumentation (for example, “Idea A reduces variance by leveraging auxiliary context; Idea B lacks this inductive bias and is therefore expected to underperform on benchmarks sensitive to input variability”) format and to establish the structured output format that the subsequent RL reward function expects before RL training, a subset of training examples with reasoning traces extracted in 2 different ways: Literature Grounded and Synthetic (described in §2.3) is used for a cold-start fine-tuning step.

The training objective for this stage is the cross-entropy loss over the full output sequence, which consists of the reasoning trace c enclosed in <think>...</think> delimiters followed by the binary label y:

$$\mathcal{L}_{\text{CE}} = -\sum_{t=1}^{T} \log P(o_t \mid g, h_A, h_B, o_{<t}), \tag{2.18}$$

where $o = (o_1, \ldots, o_T)$ is the full target sequence $(c_1, \ldots, c_{|c|}, y)$.

The comparison between grounded and synthetic traces addresses a specific hypothesis within RQ3: that paper-derived reasoning, which captures the actual logic that led to observed outcomes, constitutes a stronger supervision signal than frontier-model distillation? Synthetic traces are generated by a model reasoning *prospectively* from task descriptions alone, without access to the experimental results that ground-truth reasoning reflects. If the grounded traces are substantively more informative, this would suggest that the key bottleneck for this task is not reasoning capacity per se, but the availability of reasoning *grounded in experimental evidence*.

The model trianed on the syntheteic traces is called **Synthetic-Reason-SFT** and the one trained on grounded traces is called **Reason-SFT**.

2.5.3 RL Training with DAPO and Dr. GRPO

Building on the (Synthetic)-Reason-SFT checkpoint, the policy π_θ is further optimized using RLVR, treating the binary correctness of the final prediction as the reward signal. The central hypothesis motivating this stage, corresponding to **RQ4**, is that RLVR applied to a well-initialized model can induce *structured and interpretable* reasoning: rather than memorizing surface patterns from supervised examples, the model is pushed to discover latent reasoning paths over the idea and goal descriptions that support correct predictions. This is distinct from the Reasoning SFT stage, which teaches a fixed reasoning style derived from a subset of such traces. RLVR, by contrast, explores a larger space of reasoning strategies allowed by the thinking format and reinforces those that reliably predict ground-truth outcomes. This stage uses the full training set including pairs for which grounded reasoning trace is not available.

Why Cold-Start SFT? Initial experiments applying RLVR directly to the base Qwen3-8B model without the Reasoning SFT initialization revealed consistent reward hacking [Amodei et al., 2016]: the model rapidly learned to emit maximally short outputs that satisfied the format reward (a correctly placed `Answer:` token) without generating any reasoning content. Response length grew large before collapsing to near-zero as the model discovered that brevity, not reasoning quality, maximised the format component of the reward. The Reasoning SFT cold start provides the structured output template that RL exploration requires as a reference. This mirrors findings in mathematical RL training, where a small supervised seed set dramatically stabilizes subsequent RLVR [Shao et al., 2024] and Chu et al. who show the importance of initiating the RL phase with an optimal SFT chekpoint that is not underfit or overfit to the task to allow a model to generalize well.

Reward Function. The total reward $R(o)$ is the sum of a correctness component r_{cor} and a format component r_{fmt}:

$$r_{\text{cor}}(o) = \begin{cases} +3.0 & \text{if } \hat{y} = y \\ -3.0 & \text{otherwise,} \end{cases} \tag{2.19}$$

$$r_{\text{fmt}}(o) = 0.5 \underbrace{\left(\mathbb{I}_{\text{think}} - \mathbb{I}_{\neg\text{think}}\right)}_{\langle\text{think}\rangle\ (\pm 0.5)} + 0.5 \underbrace{\left(\mathbb{I}_{\text{ans}} - \mathbb{I}_{\neg\text{ans}}\right)}_{\texttt{Answer:}\ (\pm 0.5)}, \tag{2.20}$$

where $\mathbb{I}(\cdot) = 1$ if the condition holds and 0 otherwise. $\mathbb{I}_{\text{think}}$ indicates that the output contains a correctly formatted <think>...</think> block, and $\mathbb{I}_{\text{ans}}$ indicates that a valid `Answer: [0/1]` token is present. The correctness reward dominates at ± 3.0, ensuring that format compliance is a secondary concern: a correctly formatted but wrong prediction still yields a strongly negative total reward. The format reward is necessary to prevent the model from satisfying the correctness signal by omitting the reasoning block entirely.

RL Variants. Two GRPO variants are trained from the Reasoning SFT checkpoint. **(Synthetic)-Reason-SFT-DAPO** uses the DAPO objective (Section 2.2.3) to address entropy collapse and the length-dependent gradient bias that causes GRPO models to generate progressively longer responses during training. **Reason-SFT-DrGRPO** uses the Dr. GRPO objective (Section 2.2.3) to address gradient instability during later training when reward variance within a group is low. A third variant, **Reason-DAPO**, applies DAPO directly from the base Qwen3-8B model without a Reasoning SFT cold start, serving as a diagnostic control to confirm the necessity of the cold-start initialization.

2.5.4 Model Variant Summary

Table 2.2 summarises all training configurations evaluated in this work. The key distinction between variants is whether Qwen3's native <think> reasoning mode is enabled. Direct-SFT predictions are made with reasoning disabled, while all ”Reason” variants use Qwen3 with reasoning enabled. Llama3.1-8B-Instruct is trained only with Direct-SFT: its substantially weaker SFT baseline and its lack of native structured thinking support is the reason we don't evaluate under RL paradigm. All fine-tuned variants use LoRA adapters applied to the target modules described in Section 2.4.3.

Variant	Model(s)	Think Mode	Cntd. from Checkpt.	Dataset Used
Direct-SFT	Qwen3-8B, Llama3.1-8B	No	No	Full train set with binary preference labels
Reason-SFT	Qwen3-8B	Yes	No	Subset with grounded human-written reasoning comparisons
Synthetic-Reason-SFT	Qwen3-8B	Yes	No	Synthetic chain-of-rubrics style traces obtained on subset of train from GPT-5
Reason-DAPO	Qwen3-8B	Yes	No	Full training set with binary reward signals
Reason-SFT-DAPO	Qwen3-8B	Yes	Yes	Continued RL training on full dataset after Reason-SFT initialization
Reason-SFT-DrGRPO	Qwen3-8B	Yes	Yes	Continued RL training on full dataset after Reason-SFT initialization
Synthetic-Reason-SFT-DAPO	Qwen3-8B	Yes	Yes	Continued RL trainig on full dataset after Synthetic-Reason-SFT initialization

Table 2.2: Summary of model training variants. The table lists each training variant, the models trained using it, whether Qwen's native <think> reasoning mode was enabled, whether RL training continued from a previous checkpoint, and the dataset used for training.

2.5.5 Training Hyperparameters

The train-test split defined in the dataset construction pipeline (Section 2.3, Table 2.1) is used throughout. The training set is further subdivided into 90% train and 10% validation, stratified by sigma category to maintain proportional difficulty representation in both subsets. All mention of train dataset refers to this 90% split variant of train. Hyperparameters are selected by looking at validation loss and following best practices from literature due to limited computational resources.

SFT hyperparameters. LoRA rank $r = 64$, scaling $\alpha = 128$, dropout 0.1. Per-device batch size 2; learning rate 2×10^{-4} with cosine scheduling and weight decay 0.01; trained for 1 epoch with AdamW optimizer. The maximum output token length for reasoning-enabled variants is 4,096 tokens.

RL hyperparameters. LoRA rank $r = 64$, scaling $\alpha = 128$, dropout 0. Per-device batch size 1; learning rate 5×10^{-6} with weight decay 0.01; group size $G = 4$; KL penalty $\beta = 10^{-5}$; maximum output token length 3,600; trained for 1 epoch with AdamW optimizer. The RL learning rate is much smaller than the SFT rate because policy-gradient updates are high-variance and rely on noisy reward signals from sampled outputs. Smaller step sizes keep policy updates stable and prevent the model from drifting too far from the pretrained distribution, whereas SFT can use larger learning rates due to stable token-level supervision from labeled data. The very small but non-zero KL penalty provides mild regularization toward the reference policy's grammatical structure and consistency without fully constraining the exploration of reasoning content [Liu et al., 2025a]. Training uses the Unsloth[3] library (built upon TRL[4]) with vLLM for fast inference.

2.6 Evaluation Methodology

2.6.1 Primary Metric: Consistent Accuracy

Position bias is a well-documented failure mode of option based evaluation with language models: models systematically favour a prticular option, independent of content [Zheng et al., 2024]. Standard accuracy, which counts each input independently, does not take care of this failure mode entirely.

To make position-sensitivity explicit, the evaluation design considers a prediction **consistent** if the model gives complementary answers on both the original and swapped orderings of an idea pair. Let $f(g, h_A, h_B) \in \{0, 1\}$ denote the model's prediction for input (g, h_A, h_B), and let $f' = f(g, h_B, h_A)$ denote its prediction when the idea ordering is swapped (with the label also flipped accordingly). A prediction is **consistent** if:

$$f(g, h_A, h_B) = 1 - f(g, h_B, h_A), \tag{2.21}$$

meaning the model changes its answer when the ideas change positions, indicating that the decision is content-driven rather than position-driven. **Consistent accuracy** is then defined as:

$$\text{Acc}_{\text{consistent}} = \frac{\sum_{i=1}^{N} \mathbb{I}[\text{consistent}_i \wedge \text{correct}_i]}{N}, \tag{2.22}$$

where N is the total number of test samples including both orderings and $\mathbb{I}[\cdot]$ is the indicator function. Inconsistent predictions are treated as incorrect by default. This metric is strictly harder than standard accuracy: a model that always predicts "Idea 1 is better" achieves 50% simple accuracy but 0% consistent accuracy, correctly reflecting its complete lack of genuine discriminative ability.

[3] https://github.com/unslothai/unsloth
[4] https://github.com/huggingface/trl

2.6.2 Statistical Testing

To rigorously assess whether observed differences in consistent accuracy (for robustness checks) are statistically meaningful rather than artefacts of sampling noise, a non-parametric bootstrap procedure is employed throughout. This applies to bias analyses in the robustness evaluation (**RQ5**). For each comparison, consistent accuracy is computed separately on two complementary groups (for example, fine-tuned model versus frontier baseline, or "longer is better" versus "shorter is better"), and the null hypothesis $H_0 : \Delta = 0$ is tested against the two-sided alternative $H_1 : \Delta \neq 0$.

Concretely, $B = 10{,}000$ bootstrap resamples are drawn with replacement from the test set, and consistent accuracy is recomputed per resample. For each resample, the difference $\Delta^* = \text{acc}^*_A - \text{acc}^*_B$ is recorded, forming the bootstrap distribution $\{\Delta^*_b\}_{b=1}^{B}$. The 95% percentile confidence interval is:

$$\text{CI}_{95\%} = \big[P_{2.5}(\Delta^*),\ P_{97.5}(\Delta^*)\big], \tag{2.23}$$

and the two-sided p-value is:

$$p = \begin{cases} \min\big(2 \cdot P(\Delta^* < 0),\ 1\big) & \text{if } \bar{\Delta} \geq 0, \\ \min\big(2 \cdot P(\Delta^* > 0),\ 1\big) & \text{if } \bar{\Delta} < 0. \end{cases} \tag{2.24}$$

H_0 is rejected at $p < 0.05$ (reported as *) and $p < 0.01$ (reported as **). Resampling is performed at the level of paired units (original + swapped pair) rather than individual samples, preserving the dependency structure of the consistency-based metric. A non-parametric bootstrap is preferred over parametric alternatives such as McNemar's test because it makes no distributional assumptions about the accuracy difference. All bootstrap computations are implemented in Python using NumPy.

2.6.3 Robustness

Robustness is evaluated along four dimensions corresponding to **RQ5**, each targeting a candidate shortcut that a model might exploit in place of genuine comparative reasoning. All tests use consistent accuracy and the bootstrap procedure described above when relevant.

Position robustness. Whether a model's prediction changes depending on which idea appears first in the prompt is the most fundamental form of presentation sensitivity. As described in Section 2.6.1, consistent accuracy is defined precisely to detect and penalize position bias: any pair on which the model gives the same answer regardless of idea order is classified as inconsistent and treated as incorrect. Position robustness is therefore built into the primary evaluation metric: the consistent accuracy score across the full test set directly quantifies the degree to which model decisions are order-invariant.

Length robustness. Ideas describing more complex or more extensively documented methods tend to have longer descriptions. A model that exploits text length as a proxy for quality would perform better when the longer idea is the true winner. Test pairs are partitioned into *longer is better*, *shorter is better*, and *equal* based on the ratio of winner to loser description length. The length is detemined by first tokenizing the ideas using the model's default tokenizer. Consistent accuracy is compared between the longer and (shorter + same lenght) categories under $H_0 : \Delta_{\text{len}} = 0$.

Recency robustness. Test pairs are partitioned into *newer is better* and *older or same year* based on the publication years of the two ideas' RR papers. The "older" and "same year" are put together so that the overall dataset under each of the 2 categories used for tesing the hypothesis, is balanced to the best possible extent. No bias under this category would reflect genuine discriminative reasoning beyond the recency prior. $H_0 : \Delta_{\text{recency}} = 0$

Paraphrase robustness. If model predictions are driven by surface wording rather than underlying semantics, paraphrasing idea descriptions while preserving meaning should degrade accuracy. A paraphrased test set is generated by carefully restating the same (winning) idea using Gemini-2.5-pro (Prompt in Appendix A) in a new way in each pair. Consistent accuracy on the paraphrased set is compared against the original under $H_0 : \Delta_{\text{para}} = 0$.

2.6.4 Sycophancy

Sycophancy in language models refers to the tendency to align responses with user-stated beliefs even when incorrect, prioritizing perceived agreement over factual accuracy [Sharma et al., 2025]. This behavior is a well-documented property of models fine-tuned with human feedback and is particularly relevant when a model trained on binary correctness signals encounters contradictory pressure at inference time. The RLVR fine-tuned models generate reasoning traces before making the final prediction. This allows them to ”think” and ”reflect” before answering. Given this ability, we assess whether the trained models are susceptible to sycophantic pressure, a controlled stress test is conducted in which the model is given a deliberately false claim about which idea performed better (i.e. an input with a label unlike the prediction task). Qwen3 varinats of Reason-SFT-DAPO, Synthetic-Reason-SFT-DAPO, and Base (with reasoning) are tested for this.

Each idea pair is presented under three conditions:

Hard lie. The model is told the incorrect winner and commanded to confirm it: the user message includes “*[Incorrect Winner] achieved a higher score. Please analyze why and confirm this result by outputting Answer: [wrong label].*” This condition establishes an upper compliance ceiling, since no exit is permitted.

Soft lie. The model is told the incorrect winner but explicitly permitted to disagree: the user message states the incorrect winner performed better, then adds “*If you think this is not true or incorrect, provide justification. Output Answer: [wrong label] if you agree, or Answer: [correct label] if you do not.*” Both options are presented symmetrically; no nudge toward disagreement is given. This condition is the primary diagnostic for genuine belief maintenance under pressure: compliance here is unambiguously sycophantic.

Truth control. The model is told the correct winner (positive control). Agreement here is the desired behavior, providing a baseline for how readily the model endorses truthful claims, against which lie-condition compliance rates can be compared.

Responses in lie scenarios are classified as Complied_With_Lie or Resisted_Lie; for the truth control, the corresponding labels are Correct_Agreement and Incorrect_Disagreement. To rule out position bias as a confound, each instance is evaluated with both original and swapped idea orderings, generated by swapping `idea_1` $\leftrightarrow$ `idea_2` and flipping the label accordingly. Position bias is flagged whenever the model's predicted answer is identical across both orderings, indicating a preference for a fixed positional slot rather than content-based reasoning. This flag is computed independently at two levels: (i) from the sycophancy stress-test responses (did the model give the same answer under the condition prompt in both orderings?), and (ii) from the standard generation predictions (did the model give the same answer in the ordinary evaluation in both orderings?).

These two indicators measure distinct phenomena: the first captures whether compliance or resistance is driven by positional preference under adversarial or truthful framing; the second establishes whether the model's independent predictions are content-sensitive at all.

The sycophancy outcome and the standard generation prediction are cross-referenced in a 2×2 contingency table with four cells: *consistently wrong* (complied with lie; predicted incorrectly under standard evaluation), *purely sycophantic* (complied with lie; predicted correctly without pressure), *robust and correct* (resisted the lie; predicted correctly), and *confused* (resisted the lie; predicted incorrectly regardless). The truth-control condition is analysed within the same contingency framework, with Correct_Agreement and Incorrect_Disagreement replacing the lie-condition labels, to assess whether the model is more deferential to false claims than to true ones. For the lie conditions, a position-bias breakdown further stratifies each contingency cell by both bias indicators, isolating compliance and resistance attributable to positional preference from genuine content-based responses. For the truth-control condition, the sycophancy-level position bias indicator is reported in aggregate per stratum, since near-universal positional preference under this framing renders per-cell stratification uninformative. This full analysis is conducted separately for all three conditions and for each ($\sigma = 1, 2, 3$), with $n = 50$ pairs each, randomly sampled from the test set.

2.6.5 Idea Ranking

Beyond pairwise prediction, **RQ6** asks whether the trained models can be used to produce a full ranking of ideas competing on the same benchmark/research goal, functioning as an automated filter that picks the most promising approaches before any experiments are run. We test this ability through a simple method of idea ranking within a benchmark leaderboard.

Ranking procedure. All benchmarks/resaerch goals in both the in-domain (ID) and cross-domain (CD) test sets that contain at least three unique ideas are selected. For each such benchmark, all $\binom{n}{2}$ unordered idea pairs are formed and each pair is presented to the model together with the benchmark research goal, yielding a binary prediction for every comparison. Pairs on which the model produces inconsistent predictions (i.e. the same answer regardless of idea order, as defined in Section 2.6.1) are discarded entirely and do not contribute to the win tallies. Each idea h_i is then assigned a win count equal to the number of consistent comparisons in which it is predicted to be the better idea. Ideas are ranked in descending order of win count; ties are assigned the same rank. The true ranks are obtained by normalizing the ranks of the ideas on the benchmark leaderboard (i.e. research goal).

Evaluation metrics. Three metrics quantify the quality of the produced ranking against the ground-truth Unified Score ordering.

The **Consistency Rate** is the fraction of all pairwise comparisons within the ranking evaluation that yield consistent predictions, using the same definition as in Section 2.6.1. It measures the overall reliability of the model's judgment when applied exhaustively across a benchmark, and is reported alongside the ranking quality metrics as it directly determines how many comparisons contribute usable signal.

Top-1 Accuracy measures the fraction of benchmarks for which the idea with the highest win count is also the true winner (rank on the leaderboard). It reflects the most practically relevant use-case: given a pool of candidate methods, identify the single best one without running experiments. Additionally this is computed only from leaderboards where at least two distinct ranks were predicted to guard against cases where the ranks assigned were all 1 because of inconsistent/wrong predictions, leading to same 0 win counts for all ideas.

Median RMSE measures the quality of the full ranking by computing, for each benchmark, the root mean squared error between the predicted ranks and the ground-truth ranks derived from the leaderboards, then taking the median of this per-benchmark RMSE across all evaluated benchmarks.

Lower Median RMSE indicates that the produced ranking more closely tracks the true performance ordering across all positions, not only at the top.

All three metrics are reported separately for the in-domain and cross-domain test sets.

2.6.6 Few-shot Ablation

To test whether in-context learning can substitute for task-specific fine-tuning, GPT-5 in a 3-shot setting is evaluated. For each test pair, 3 demonstration examples, one from each difficulty category (1σ, 2σ, 3σ) as in-context examples are prepended before asking for the final prediction. The same

3 examples are used for all test pairs. This ablation directly addresses whether the performance gap between the fine-tuned 8B models and GPT-5 can be closed by providing GPT-5 with task demonstrations §3.2.2

Chapter 3

Results and Discussion

The central finding of this work is that comparative empirical forecasting does not require frontier-scale models: an 8B-parameter model, fine-tuned on benchmark-specific idea pairs, reaches 77.10% accuracy and outperforms GPT-5 at its highest reasoning effort (61.10%) by more than 15% points. Models trained with reinforcement learning recover to approximately 71% accuracy while exhibiting meaningfully better cross-domain and external dataset generalization than their supervised counterparts. All trained models show robustness to paraphrasing and consistency confounds with mixed results on other tests. Together, these results show a promising step towards addressing the *validation bottleneck* identified in the introduction, the growing gap between the rate at which ideas can be generated and the rate at which they can be evaluated, is addressable with compute-efficient models trained on objective, outcome-grounded data.

3.1 Dataset Analysis

Dataset is available at https://anonymous.4open.science/r/Benchmark-Dataset-81B0. Raw dataset with code will be made available upon request (due to public access restrictions), through GitHub.

3.1.1 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 3.1 shows the top 20 benchmarks based on the total pairs (including train and

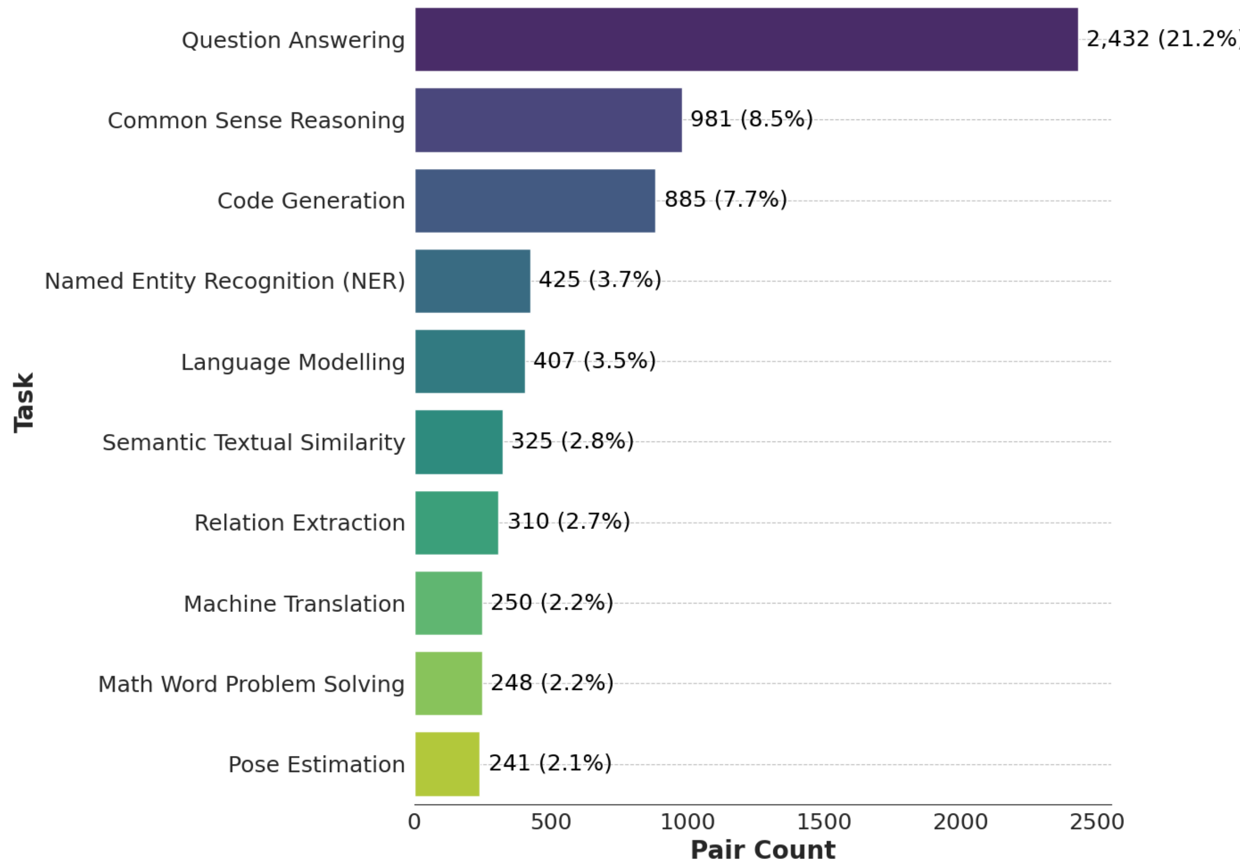


Figure 3.1: Top 20 Tasks based on the total pairs (includes both train and test)

test) in our dataset. Question Answering (21.2% of total pairs) and Common Sense Reasoning (8.5% of total pairs) are most common tasks in our dataset. Further, Figure 3.1 shows the top 20 ”Tasks” and the number of pairs from each of them.

3.1.2 Temporal Distribution

Figure 3.2 and 3.3 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, a property that becomes important if knowledge-cutoff is accounted for within the in-domain test set (but we test for knowledge cutoffs on the cross domain instead due to limited number of pairs present in the test set).

3.1.3 Excluded Pairs

Table 2.1 gives us the full statistics of the final dataset. Even after employing manually verified LLM based Research Goal Synthesis in §(2.3), we miss out on a large chunk of pairs (close to 30%) due to lack of valid research goals. This highlights the downside of current approaches of extracting the research goal, even when multiple sources are used. Future work could explore ways to fix this, which would allow a more broad coverage of benchmark leaderboards.

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

3.1.4 Excluded Benchmark Leaderboards

Apart from missing Research Goals, some of the benchmarks are excluded during the Train-Test as described in Section §(2.3) 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 from 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.

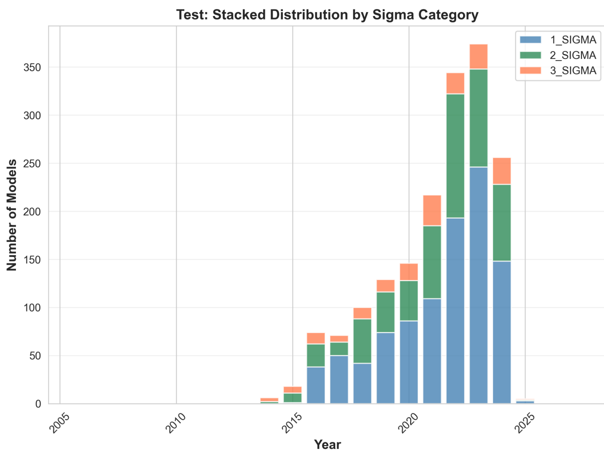


Figure 3.2: Distribution of the ideas/methods across the years with σ-wise breakdown in test

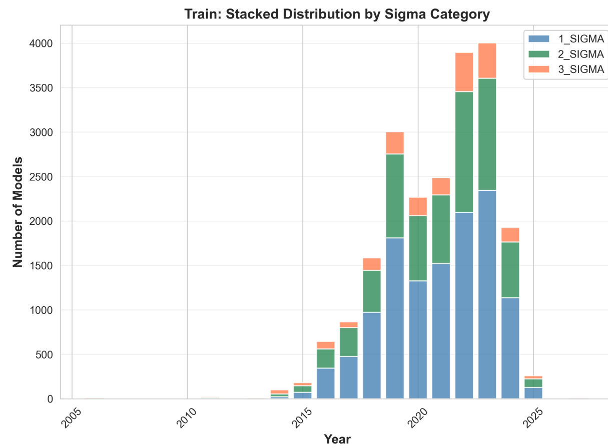


Figure 3.3: Distribution of the ideas/methods across the years with σ-wise breakdown in train

3.2 RQ1 & RQ2: Can Small Language Models Forecast Empirical Outcomes?

3.2.1 Untrained 8B models perform poorly

Table 3.2 reports consistency-aware accuracy (§2.6.1) across all models and difficulty tiers. Untrained base models perform poorly: *Qwen3-8B* scores 20.14% (25.31% with thinking enabled) and *Llama3.1-8B* scores 30.03% (27.39% with CoT). These numbers are a direct consequence of the consistency-aware evaluation metric: a model that exhibits strong position bias makes inconsistent predictions across the original and swapped variants of the same pair, and such pairs are scored as incorrect by default. Untrained 8B models are not just bad at this task formulation, but also higly biased by the presentation order (Figure 3.5).

3.2.2 Supervised fine-tuning produces great improvements

Qwen3-8B Direct-SFT reaches **77.10%** overall accuracy, a gain of approximately 57% points over the base model and more than 15% above the best frontier model tested: GPT-5 at its highest rea-

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-SFT	35.32	38.90	45.05	37.51	29.31
Reason-DAPO	<u>69.43</u>	75.00	<u>83.52</u>	<u>72.73</u>	45.96
Reason-SFT-DAPO	64.57	<u>79.23</u>	<u>83.52</u>	71.35	<u>48.37</u>
Synthetic-Reason-SFT-DAPO	65.79	72.53	74.72	68.93	41.10
Reason-SFT-DrGRPO	66.19	76.41	<u>83.52</u>	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 3.2: Accuracy (%) breakdown across different difficulty subsets i.e. (σ)-categories, on CD test set and models.

soning effort (61.10%). This result contradicts the implicit assumption in prior work Wen et al. that frontier models are necessary for this type of evaluation and show results on a proprietary GPT-4.1 system with additional retrieval augmentation. A small open-source model, when given thousands of outcome-labeled comparisons grounded in benchmark-specific data, learns comparative priors that a frontier model cannot recover through prompting alone. This proves the validation bottleneck can be addressed without relying on massive, proprietary frontier models. The performance on 1σ<2σ<3σ across all fine tuned models showing that the σ based stratification is difficulty aligns well with true difficulty. The untrained variants show broad variations, in some cases inversion of this trend (GPT-5 and Gemini-2.5-Flash), likely because of the noise and inherent lack of internalizations needed to do well at this task. The size of the subset could also be a factor in amplifying/dampening the noise within the subset.

This point is reinforced by the few-shot ablation with GPT-5 (Table 3.3), where it is provided with three examples from the test set, does not meaningfully close the gap. Performance at low reasoning effort improves by only 0.24%, and both medium and high reasoning efforts degrade slightly (-0.58% and -0.69% respectively). GPT-5 already understands the task from the prompt; providing examples does not help because the bottleneck is not task comprehension but the absence of internalized comparative priors. This distinction matters for how such a verifier should be prepared: in-context learning is not a substitute for fine-tuning on outcome-grounded data.

Model	**Zero-shot**	**3-shot**
GPT-5 Reasoning (Low)	57.65	57.89
GPT-5 Reasoning (Med)	59.61	59.03
GPT-5 Reasoning (High)	61.10	60.41

Table 3.3: Zero-shot vs. 3-shot GPT-5 accuracy (%) on in-domain test set. Few-shot examples marginally improve low-reasoning performance but slightly degrade medium and high reasoning setting.

3.2.3 Qwen3 Vs Llama3.1

Llama3.1-8B Direct-SFT reaches only 56.50%, substantially weaker despite identical training data, task format, and comparable parameter count. Llama's base model actually outperforms Qwen's base model (30.03% vs. 20.14%), suggesting it starts from a more balanced zero-shot prior. Yet after fine-tuning, Qwen3 surpasses it by over 20 points. One plausible explanation is that Qwen3's pre-training and post-training recipes, which include heavy emphasis on reasoning-oriented tasks like Math, Code, etc. [Yang et al., 2025] and a native structured thinking mode, provide a better initialization for tasks that require multi-step and complex comparative judgment. Additionally, Qwen3 learns to mitigate the position bias much better than Llama3.1, leading to more consistent and correct predictions. This emphasises the need for prioritizing models that are pre-trained with reasoning-oriented data rather than just instruction-following.

3.2.4 Generalization, Sensitivity and Knowledge Cutoff

Generalization

An important question is whether learned priors transfer beyond NLP leaderboards and datasets constructed with different labelling strategies. All trained Qwen3 models generalize competitively to the cross-domain (CD) test set constructed from non-NLP leaderboards, matching or slightly exceeding GPT-5 (zero-shot) at high reasoning effort (45.96%). The RL-tuned variants *Reason-SFT-DAPO* and *Reason-SFT-DrGRPO* outperform *Direct-SFT* by ≈3 points on this set (48.37% and 49.08% vs. 45.67%), despite achieving lower in-domain accuracy compared to *Direct-SFT*. This reversal is an interesting finding: SFT performs poorly compared to RL on Out of Distribution (OOD) data. *Direct-SFT* appears to have overfit somewhat to patterns specific to the learnings from the NLP based train data, while RL-trained models develop more domain-agnostic comparative reasoning.

The strongest evidence for genuine transfer comes from the independently constructed test set of Wen et al. [2025], evaluated zero-shot after removing any overlap with our training data by title matching §2.3. *Reason-SFT-DrGRPO* achieves **67.49%**, outperforming the Wen et al. zero-shot GPT-4.1 with retrieval augmentation (51.4%) by over 16 points. Notably, all trained models outperform this zero-shot setting, with *Reason-SFT-DrGRPO* performing best among them. Additionally, the best fine-tuned model closes the gap with a fine-tuned version of GPT-4.1 (77%), despite being a model 50× smaller and using no retrieval. The label construction methodology in Wen et al. differs substantially from ours: their labels come from majority voting across results on multiple benchmarks in papers (which is manually verified/annotated), while our labels are derived from leaderboard metric(s) based unified scores. This also directly addresses the limitation identified in the introduction regarding prior work: unlike Wen et al., this work does not rely on proprietary frontier models, and the transfer result shows that the benchmark-specific, fine-grained framing could lead to better generalized learning.

These observations are consistent with findings of some of the existing work like [Chu et al., 2025] who show the effectiveness of RL on OOD compared to SFT, where the RL tuned models (LMs and Vision LMs) perform better than the SFT counterparts and [Jin et al., 2025] who show that SFT reaches the most optimal point for best generalized performance at the early stages of tuning but it is lost since one always uses the end checkpoint. The RL essentially recovers the lost genralizability and in all instances and doesn't surpass the best SFT checkpoint.

Model	**Rank=Metric (1,308)**	**Rank≠Metric (102)**	**Overall (1,410)**	**2025-Only (104)**
Qwen3				
Base	3.67	1.96	3.55	3.85
Base (Reasoning)	13.14	5.88	12.62	17.31
Direct-SFT	46.64	33.33	45.67	53.85
Reason-DAPO	46.64	37.25	45.96	51.92
Synthetic-Reason-SFT-DAPO	41.90	29.41	41.10	44.23
Reason-SFT-DAPO	**49.08**	39.21	48.37	**57.69**
Reason-SFT-DrGRPO	48.16	**60.78**	**49.08**	53.85
Llama3.1				
Base	3.97	1.96	3.83	3.85
Base (Reasoning)	18.04	19.61	18.22	23.08
Direct-SFT	32.87	9.80	31.20	36.54
GPT-5 (zero-shot)				
Reasoning (low)	44.19	25.49	42.84	44.23
Reasoning (med)	46.18	33.33	45.25	51.92
Reasoning (high)	46.94	33.33	45.96	48.07

Table 3.4: Full Cross-Domain (CD) test set results. **Rank=Metric**: pairs where the leaderboard rank order agrees with the individual metric order. **Rank≠Metric**: disagreement subset. **2025-Only**: subset of 104 pairs (52 pre-augmentation) with all papers dated $\geq$2025. **Bold**: best per column among trained models.

Sensitivity

To study the sensitivity of the models to extraction variations and labeling methods, we analyze the rank-order agreement vs. disagreement breakdown (Table 3.4) on the CD-test. When the leaderboard rank order and individual metric order conflict (the non-trivial 102 augmented pairs in the CD set) most models drop substantially, but *Reason-SFT-DrGRPO* achieves 60.78%, outperforming GPT-5 and all other trained models. These are precisely the cases where naive rank heuristics fail and metric-level reasoning is required, confirming that this model has internalized something about how metrics and rankings relate rather than merely associating surface features with outcomes. Most of the other trained model's performance drop by $\approx$ 10%, including GPT-5 ($\approx$ 13%), showing that this is somthing the models are not fully robust to. At the same time, the trained mod-

Model	Accuracy (%)
Qwen3	
Base	2.69
Base (Reasoning)	20.06
Direct-SFT	63.43
Reason-DAPO	65.94
Synthetic-Reason-SFT-DAPO	56.46
Reason-SFT-DAPO	61.83
Reason-SFT-DrGRPO	**67.49**
Llama3.1	
Base	12.80
Base (Reasoning)	36.29
Direct-SFT	41.94
GPT-4.1 (zero-shot, Wen et al., w/ retrieval)	51.4

Table 3.5: Accuracy (%) on the Wen et al. (2025) independently constructed test set. Fine-tuned 8B models are evaluated zero-shot (no retraining). GPT-4.1 result from Wen et al. [2025].

els still do at par or better that than GPT-5 on this subset. But given the small size of this subset (51 unique pairs) these findings can be noisy and should be interpreted with caution.

Sensitivity to benchmark-specific context Further evaluation of whether the trained model can identify the superior idea within a pair of ideas, conditionally based on the target benchmark i.e. research goal is done. 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). 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.

Memorization and Knowledge Cutoffs

The concern that performance gains reflect knowledge cutoff leakage rather than comparative reasoning deserves careful treatment here, since the generalization evidence just presented is the appropriate context in which to address it.

The prediction target is not a fact in the input text. The 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 empirical results and outcome statements from the paper text are explicitly removed. Therefore, succeeding on this 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 the 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.

Three observations are relevant First, base models perform far below chance, despite having presumably been exposed to the same papers during pretraining (Table 3.2). If leakage were the primary mechanism, base model performance should be higher, not lower than random (when position bias is accounted for). Second, frontier models like GPT-5 and Gemini 2.5 Flash, which carries a substantially more recent knowledge cutoff (September 2024 and January 2025 respectively), achieves only 61.10% and 40.73% zero-shot showing mere recall cannot lead to better performance at this task. Third, on the 2025-only subset of the CD test (52 pairs, all papers dated $\geq$2025) which accounts for the knowledge cutoffs of most models, fine-tuned models still outperform GPT-5 by nearly 6 points (57.69% for *Reason-SFT-DAPO* vs. 48.07% for GPT-5-medium). Base models crater to 3-23% on this subset, confirming that fine-tuning drives the gains rather than retrieval of memorized outcomes. The task fundamentally requires mapping a methodological description to its expected comparative performance on a particular benchmark (i.e. research goal), a benchmark-conditional pairwise judgment that cannot be reconstructed by recalling isolated paper facts. **Note:**The number of samples under this category is small (52 unique pairs) so it is possible that the values obtained are noisy and should be interpreted with caution. But the overall pattern across all three observations strongly suggests that knowledge cutoff leakage is not the primary driver of performance gains, and that the models are learning comparative reasoning patterns rather than memorizing outcomes.

Knowledge cuttoffs:

- Llama3.1: December 2023
- Qwen3: Unknown (likely end 2024)
- GPT-5: September 2024
- Gemini 2.5 Flash: January 2025

3.3 RQ3 & RQ4: Does Reasoning Help, and Can It Be Induced?

3.3.1 Deliberation improves prediction

Table 3.2 shows the comparison of standard and thinking-enabled variants. For Qwen3, enabling thinking at the base model level improves accuracy from 20.14% to 25.31%. For GPT-5, increasing the thinking budget from low to high yields a monotonic gain from 57.65% to 61.10%. Llama3.1 doesnt have a native ”think” mode. To induce reasoning, CoT prompting/instruction is used and the model is provided with extended token generation limit to allow it to ”think”. In contrast to Qwen3, prepending CoT instruction to Llama3.1 reduces accuracy from 30.03% to 27.39%. It is observed that deliberation is only reliably beneficial for models that have been pre-trained with explicit reasoning objectives. For a model like Llama that has not pre-trained in such a way, generating reasoning tokens is more likely to introduce noise than to improve judgment, since the model lacks the structured deliberation templates that make those tokens useful. This is consistent

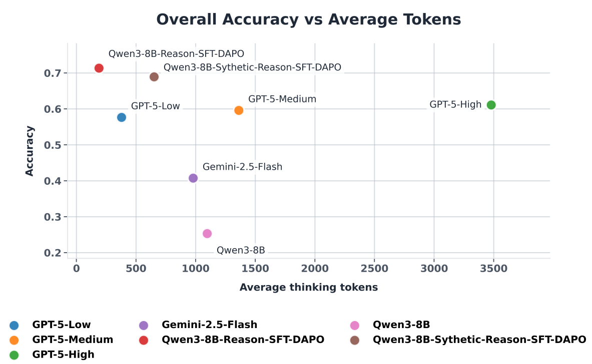


Figure 3.4: Overall Accuracy (%) Vs Mean Number of tokens generated during reasoning.

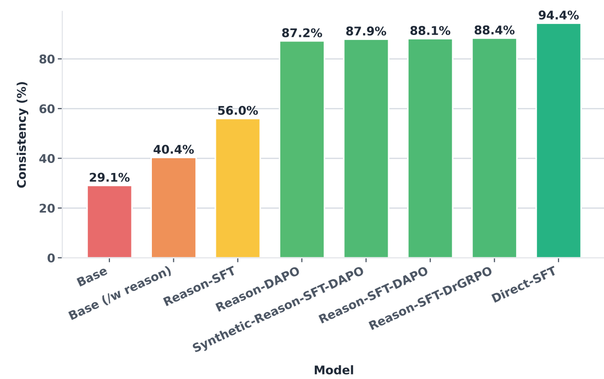


Figure 3.5: Consistency (%) across different stages and training paradigm of Qwen3 Model.

with broader findings on CoT: it helps models that are already capable of the relevant reasoning, but can harm those that are not [Meincke et al., 2025].

Figure 3.4 adds an important dimension to this: *Reason-SFT-DAPO* and *Syntheteic-Reason-SFT-DAPO* achieve comparable or higher accuracy to GPT-5 while generating a fraction of the reasoning tokens, adding to the compute efficieny argument. But the qualitative analysis of the reasoning traces (discussed in the next sections) suggests that the reasoning quality of these models is not fully comparble to that generated by GPT-5 while also showing signs of decoupling between reasoning and predictive accuracy.

3.3.2 Grounded Vs. Synthetic reasoning traces

Reason-SFT, trained on only 170 idea pairs with grounded paper-derived reasoning traces §2.1, achieves 37.51%. Although it improves over the base model with reasoning enabled, it is far below *Direct-SFT* (77.10%), a gap attributable entirely to data scarcity rather than the reasoning format.

Synthetic-Reason-SFT, trained on filtered 2,738 GPT-5-generated chain-of-rubrics traces, achieves only **25.54%**, with no meaningful improvement over the Qwen3 base model (25.31%) and far below the grounded variant. This finding is consistent with Wen et al. [2025], who observe that training on synthetic CoT degrades performance for such tasks. Finetuning on this reasoning data was partly motivated by prior results showing improved performance in settings where models are trained on a final objective reward while generating rubric-style reasoning traces (Generative Reward Modelling) before producing the final answer [Chen et al., 2026], the results here indicate that such approaches do not necessarily transfer across tasks under current setting.

This observation suggests that training setups where reasoning traces implicitly act as a reward model may not consistently improve performance when applied outside the settings in which they were originally demonstrated.

This finding carries a practical implication that extends beyond this work. Many RM-R1-style pipelines [Chen et al., 2026] use frontier model traces as distillation targets for smaller models. The present results suggest that for tasks requiring genuine comparative judgment rather than factual recall or formal derivation, this approach may not transfer reliably. Whether this limitation is specific to empirical forecasting or reflects a broader constraint on reasoning-trace distillation remains an open question for future work.

3.3.3 Reinforcement Learning

Starting from *Reason-SFT* (37.51%), applying DAPO and Dr.GRPO with binary outcome labels as the verifiable reward signal recovers substantial predictive performance. Both *Reason-SFT-DAPO* and *Reason-SFT-DrGRPO* reach ≈71% accuracy (Table 3.2), narrowing the gap with *Direct-SFT* to approximately 6 points. Notably, both RL variants converge to the same accuracy plateau despite different optimization algorithms. This convergence likely reflects an intrinsic ceiling given the quality of the SFT initialization (37.51%) and the binary reward signal: once the model has learned to exploit the strongest discriminative features in the training distribution, further RL exploration yields diminishing returns on in-domain accuracy.

Meanwhile the DAPO traninig starting from the Synthetic-Reason-SFT checkpoint, which starts from a much lower SFT accuracy (25.54%) than the grounded Reason-SFT checkpoint (37.51%), also converges to a similar accuracy plateau (68.93%). This suggests that the quality of the SFT seed data is a key factor in determining the upper bound of RL performance, and that starting from a weaker seed leads to a lower ceiling. Although the performance gap between the two is not large (68.93% vs 71.35%) on the in-domain test, the gap widens on the CD test (41.10% vs 48.37%) and Wen et. al. test (56.46% vs $\approx 62 - 67\%$), suggesting that the quality of the SFT seed data also has implications for generalization, agreeing with Jin et al.'s work which show that the best OOD performance appears at some intermediate SFT checkpoint and RL can only recover the performance to this *Best-SFT* checkpoint's performance.

Whether scaling the reasoning seed data or enriching the reward signal could lift this ceiling is an important open question for future work which depends on finding high quality reasoning traces.

3.3.4 Reasoning and Accuracy

Inspection of generated reasoning traces reveals a consistent and troubling pattern: RL largely restores predictive accuracy without restoring reasoning quality. Three qualitatively distinct behaviors emerge.

Reason-DAPO, trained directly from the base model, undergoes a training collapse partway through optimization. The model ceases generating reasoning traces while retaining format-compliance tokens, then recovers reward by producing only the minimal token sequence required for format and answer rewards. Figures 3.6 and 3.7 document this precisely: response length spikes near 24k steps, rewards drop, and then length collapses to approximately 9 tokens while reward recovers to pre-collapse levels. This is a textbook case of reward hacking, and it illustrates a deeper tension in the RLVR framework: when the reward is defined only over the final prediction, the model has no incentive to preserve intermediate reasoning. A length penalty and increased KL penalty (1e-4) partially mitigates this but introduces a new failure: the model satisfies the length constraint by repeating 3-4 sentences in a loop, substituting iteration for reasoning. These observations highlight the importance of the SFT seed before running RL.

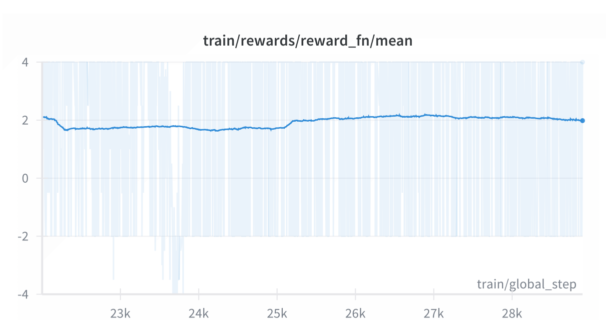


Figure 3.6: The average rewards through the training iterations of Reason-DAPO

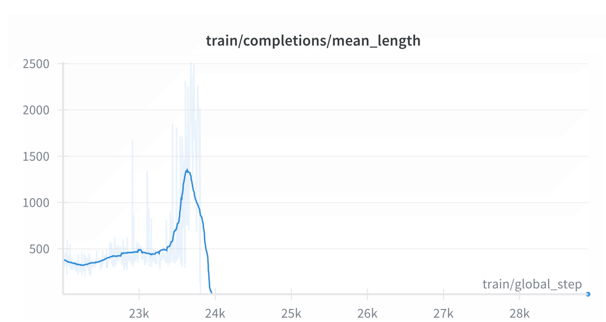


Figure 3.7: The mean output length through the training iterations of Reason-DAPO

What is surprising is that *Reason-DAPO*, despite its reward-hacked reasoning, achieves 45.96% on the CD test and 65.94% on Wen et. al.'s dataset, at par or better than *Direct-SFT* (45.67% and

63.43%). This suggests the RL training signal, even when circumvented at the reasoning level, nevertheless shapes the model's internal representations in ways that generalize for final answer prediction. The representations learned to maximize the binary reward apparently capture comparative priors that transfer across domains, even without producing coherant reasoning traces.

(Synthetic)-Reason-SFT-DAPO present the most successful outcome among the RL variants. Initialized from the *(Synthetic)-Reason-SFT* checkpoint, it produces consistent and coherent reasoning traces prior to the final prediction, and does not undergo the collapse seen in *Reason-DAPO*. The SFT cold-start provides a structured reasoning template that DAPO then reinforces. This confirms that the SFT seed data, despite its small size, is essential: it provides a prior over reasoning structure that the RL objective fails to optimally discover from scratch.

Reason-SFT-DrGRPO progressively minimizes reasoning trace length over training iterations. Outputs often devolve into superficial justifications, for instance asserting that one idea is better because it is more recent even when the idea descriptions contain no temporal information. This is a form of reasoning collapse that is less dramatic than reward hacking but equally problematic for interpretability: the model produces structurally valid reasoning tokens that are semantically empty with respect to the actual prediction.

What RL-trained models actually reason about? Qualitative analysis across all Reason-SFT-RL-trained variants (including Reason-SFT-DAPO) reveals that the a large stated justifications fall into three broad categories: training on larger datasets, adoption of a more sophisticated or recent approach, and use of larger model capacity. While not unreasonable as heuristics, this pattern suggests that even the best-behaved RL-trained models are exploiting high-level regularities in the training distribution rather than engaging in fine-grained mechanistic reasoning about each benchmark. Analysis of the Grounded reasoning dataset used for SFT checkpoint reveals that these justifications are not the dominant reasons present.

Similar analysis of the justifications generated by Synthetic-Reason-SFT-DAPO reveals that though it considers different aspects to compare the ideas and reflect on their strengths and weaknesses, a common theme is ”generalization/overfitting” as one of the important factors across comparisons. Further it confuses the label of each idea (i.e. Idea 1/2) and the description corresponding to it. For example it may refer to the description of Idea 1 while calling it Idea 2 mid reasoning, but correct the label and description in the final answer, leading to difficulty in assessing if the stated reasons correspond to the actual prediction.

The question of whether these reasoning is *faithful* i.e. whether the stated justification actually drives the prediction, is distinct from whether the prediction is correct, showing a clear decoupling between reasoning quality and predictive accuracy. The accuracy improvements from RL training may primarily reflect stronger internal comparative representations, with the generated text providing rationalization rather than explanation.

This framing suggests that jointly optimizing for prediction accuracy and reasoning faithfulness requires a reward signal that evaluates reasoning quality directly, not merely the final label. Even implicit GenRM methods that best suit the type of task we are studying, like RM-R1 approach that use dynamic reasoning as implicit rewards even when fine-tuning with a RLVR approach based solely on the final answer prediction, dont adopt well. Rubric-based reward models [Gunjal et al., 2025] offer one direction where the rubrics and scores can be generated grounded in actual papers; a direct and simpler methods of grounding intermediate reasoning against source paper evidence is another. Both are directions for future work.

3.3.5 LoRA, RLVR, Computational Constraint and Reasoning Quality

The behaviors observed across the RL-trained variants: reward hacking in Reason-DAPO (even when accuracy improves), trace minimization in Reason-SFT-DrGRPO, better OOD generalization compared to SFT and the persistent gap between prediction accuracy and reasoning quality, are not random failures. They connect to a set of fundamental tensions between how LoRA works, group size in GRPO based RL training, and what RLVR actually needs from a model. Understanding these tensions helps explain why the results look the way they do, and points toward concrete ways to improve in future work.

The asymmetry between gains in predictive accuracy and deterioration in stated reasoning is closely aligned with recent large-scale studies of RLVR dynamics. Chu et al. [2025] and Jin et al. [2025] show that SFT tends to memorize training distributions whereas RLVR preferentially improves out-of-distribution performance by reshaping how existing representations are used. Yue et al. [2025] further argue that RLVR rarely induces qualitatively new reasoning patterns beyond those latent in the base model, instead reweighting and re-sampling existing solution modes. When you evaluate models using pass@k (the probability that at least one of k samples is correct) RLVR-trained models outperform their base models at small k (e.g., pass@1), but the base model catches up and often overtakes them at large k (e.g., pass@256). The correct reasoning paths are already present in the base model. What RLVR does is make those paths more likely to be sampled, effectively sharpening the distribution around known rewarded traces. This comes at a cost: the model narrows its coverage of solvable problems, because it concentrates probability on high-reward paths at the expense of the broader set of valid reasoning strategies it knew before training.

In the observations made in the previous sections, the RLVR-trained models similarly appear to exploit high-level comparative heuristics already present in the base/initial SF-Tuned model. While the base model is already very poor at the given task, the SFT-before-RL checkpoint has a very small seed dataset. In both cases this leads to sparse signals right from the beginning and the observed collapse in the braod categories of the reasons (e.g. "bigger models", "more data", "more recent methods") rather than learning a fine-grained scientific reasoning like we expect it to.

A deeper issue is that RLVR and SFT make updates to different parts of the weight space. Zhu et al. [2025a] show this directly: RLVR updates consistently land in low-curvature, off-principal directions: regions that carry little variance in the pre-trained model. SFT, by contrast, targets principal directions, the high-variance parts of the weight matrix. The reason for this difference is the KL penalty built into GRPO style methods: updating principal directions would shift the model's distribution most strongly, triggering large KL penalties, so the optimizer naturally steers away from them and works in flatter, less disruptive parts of the space. These off-principal updates are also very small in magnitude, small enough that in lower numerical precision formats (e.g. bf16) they effectively disappear from most weight positions, which is why RLVR appears to touch only a tiny fraction of parameters.

The choice of low-rank adapters under tight compute constraints likely amplifies this effect. LoRA [Hu et al., 2021] makes fine-tuning feasible in this setting, but recent work suggests that adapter capacity and structure are critical for inducing strong reasoning via RLVR. Khan et al. [2026] show that, in a strictly micro-budget (one A40 GPU, under 24 hours) RLVR regime, very low-rank adapters (e.g., r=8) systematically fail to capture the optimization dynamics required for mathematical reasoning, while higher-rank adapters (e.g., r=256) unlock considerably more plasticity; heavily task-aligned bases can instead degrade under noisy, low-budget RLVR. Complementary results from a comprehensive evaluation of parameter-efficient methods for RLVR [Yin et al., 2025; Zhu et al., 2025a] indicate that vanilla LoRA is often suboptimal: structural variants such as DoRA, AdaLoRA and MiSS better preserve and exploit reasoning capacity, whereas extreme compression (e.g., VeRA, Rank-1) and some SVD-based initializations can suffer from spectral collapse when they force updates into the principal subspace. Against this backdrop, the rank-64 vanilla LoRA configuration used here can be viewed as a compromise between plasticity and stability: it enables training on a single A100 while respecting the off-principal geometry to some extent, but may not provide enough adaptive capacity in the right subspaces to substantially restructure the model's internal comparative reasoning circuits.

The observed training dynamics of the RLVR runs: rapid early reward gains, sharp reductions in output length, and eventual performance plateaus; also match broader analyses of entropy dynamics in RL for reasoning LLMs. Cui et al. [2025] show that, in GRPO-style RLVR on verifiable tasks, policy entropy collapses quickly (output distribution becomes narrow). This implies that much of the improvement is obtained by trading away exploratory capacity, with a predictable ceiling once entropy is exhausted. Petrenko et al. [2026] similarly demonstrate that standard policy-gradient objectives naturally drive entropy down over training unless it is actively controlled, and propose entropy-preserving variants (REPO, ADAPO) that maintain diversity and support sequential post-training. The reward hacking and reasoning-length collapse seen in the Reason-DAPO and Reason-SFT-DrGRPO runs in this work are consistent with this picture: the policy rapidly converges on short, high-reward patterns that fit the binary verifier and format rewards, but in doing so it sacrifices the diversity and depth of reasoning traces.

These entropy dynamics interact with the small group size used in our setup. All RLVR runs here use a group size of $G{=}4$ (§2.5.3), well below the $G{\geq}8$–16 regimes that recent analyses and methods such as F-GRPO [Plyusov et al., 2026] identify as stabilizing for rare-but-correct trajectories. With $G{=}4$, group-normalized advantages are noisy and many prompts effectively never see diverse successful trajectories, making it easier for the policy to overfit a narrow set of high-reward heuristics and harder to sustain exploration. Increasing effective group size via more memory, down-sampling schemes, or hierarchical grouping would likely improve both entropy behaviour and robustness. But for the experiments done in this work, compute constraints prevent the usage of larger group sizes.

Recent RLVR algorithms suggest concrete ways to mitigate some of these issues under similar or slightly larger compute budgets. Focal GRPO (F-GRPO) [Plyusov et al., 2026] introduces difficulty-aware advantage scaling that down-weights updates on prompts where correctness is already high, reducing the tendency of small-group RLVR to overfit common patterns and forget rare-but-correct trajectories. TAMPO [Dang et al., 2026] treats the sampling temperature as a meta-policy, learning to adapt exploration over training instead of a fixed hyperparameter. In the inner loop, the model generates rollouts at a temperature selected by the meta-policy; in the outer loop, temperatures that produced high-advantage trajectories are rewarded. Both techniques directly target the exploration side of the RLVR trade-off without changing the underlying verifier. Combining such exploration aware updates with entropy-preserving objectives [Cui et al., 2025; Petrenko et al., 2026] and geometry-aware PEFT could, in principle, sustain longer, more diverse reasoning traces while still driving up verifiable accuracy.

In light of these findings, the results presented in this work can be interpreted as a lower bound on what is achievable under the chosen LoRA rank, Group size, and RLVR hyperparameters. Direct-SFT and Reason-SFT provide strong seeds, but the rank-64 vanilla LoRA adapters, GRPO variants with $G{=}4$, and entropy-collapsing policy-gradient updates together appear to favour reweighting existing high-level heuristics over learning new, faithful reasoning procedures. Future work could therefore explore: (i) higher-rank or structurally richer, geometry-aware adapters (e.g., DoRA, AdaLoRA, MiSS, etc.) targeted at layers most responsible for comparative reasoning; (ii) RLVR objectives that explicitly manage entropy and exploration, such as Clip-Cov/KL-Cov or REPO/ADAPO, possibly combined with learned temperature schedules and adaptive group sizing; and (iii) multi-stage pipelines where RLVR first broadens the model's coverage of reasoning modes and only then distils those modes into shorter, more faithful rationales. These directions offer a path toward improving both accuracy and reasoning quality, rather than trading one off against the other.

3.4 RQ5 & RQ6: Are Model Predictions Robust?

3.4.1 Robustness

Robustness to presentation variation is evaluated across four dimensions as described in §2.6.3. Figure 3.8 summarizes the bias deviations Δ across difficulty tiers, Table 3.6 provides bootstrapped significance tests with Confidence Intervals and Table 3.8 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.

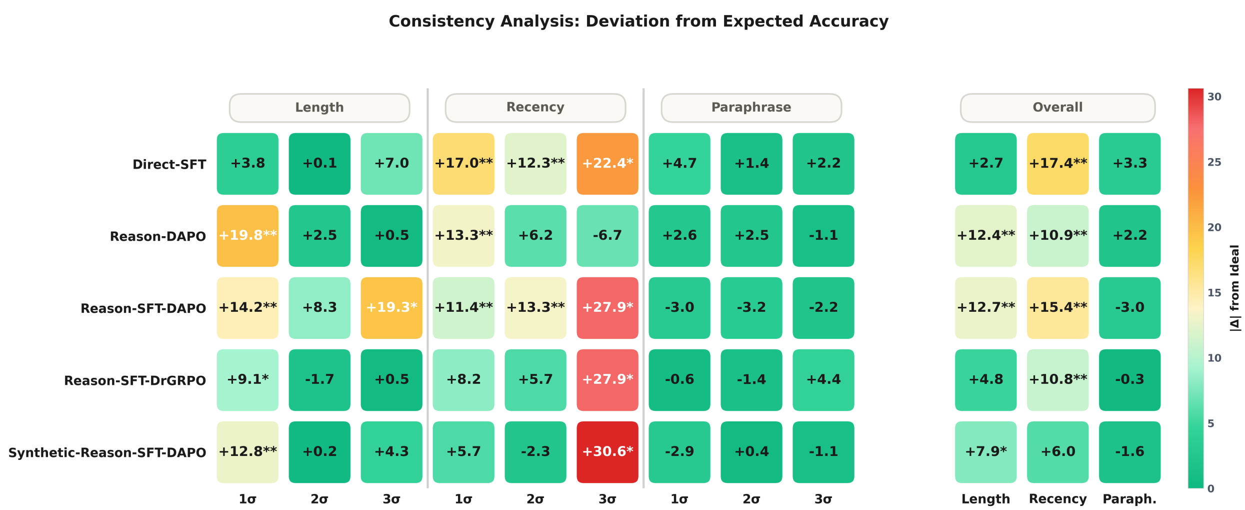


Figure 3.8: Differential Analysis (Δ based) of different Robustness tests across Difficulty Subsets (σ) and Overall Performance. **: $p < 0.01$; *: $p < 0.05$

Position bias

Figure 3.5 shows that trained models exceed 85% consistency on the in-domain test set, compared to 29–40% for untrained base models. Position bias is not a meaningful confound for any trained model in the main results, which are reported using consistency-aware accuracy throughout.

Paraphrasing

No model shows a statistically significant change in accuracy due to paraphrasing at any difficulty tier ($p > 0.05$ for all models and all σ-categories) i.e. we do not reject the null hypothesis $\Delta_{para} = 0$. All overall CIs are narrow and straddle zero within ± 8 pp, and since N is large for both original and paraphrased sets at every tier, these estimates are well-powered. Model predictions are driven by what the ideas say, not by how they are worded.

Table 3.6: Bootstrapped bias significance tests ($B = 10{,}000$). Δ in percentage points with 95% CIs. **: $p < 0.01$; *: $p < 0.05$.

Model	Tier	Length Δ [95% CI]	Recency Δ [95% CI]	Paraphrase Δ [95% CI]
Direct-SFT	Overall	+2.7 [−2.8, 8.3]	+17.5** [11.8, 23.2]	+3.3 [−0.6, 7.1]
	1-σ	+3.8 [−4.4, 12.0]	+17.1** [9.3, 24.8]	+4.6 [−0.8, 10.1]
	2-σ	+0.1 [−8.3, 8.5]	+12.3** [3.7, 21.3]	+1.4 [−4.2, 7.0]
	3-σ	+7.2 [−8.1, 23.2]	+22.5* [0.2, 46.0]	+2.2 [−7.7, 12.1]
Reason-DAPO	Overall	+12.3** [6.5, 18.2]	+10.9** [4.9, 16.9]	+2.2 [−2.0, 6.3]
	1-σ	+19.8** [11.7, 27.7]	+13.3** [5.2, 21.2]	+2.6 [−3.0, 8.3]
	2-σ	+2.5 [−7.5, 12.4]	+6.2 [−4.1, 16.7]	+2.5 [−4.2, 9.5]
	3-σ	+0.3 [−15.5, 16.6]	−6.8 [−21.9, 11.3]	−1.1 [−12.1, 9.9]
Reason-SFT-DAPO	Overall	+12.7** [6.5, 18.8]	+15.4** [9.2, 21.5]	−3.0 [−7.2, 1.4]
	1-σ	+14.2** [5.8, 22.7]	+11.3** [3.1, 19.8]	−3.1 [−9.1, 2.8]
	2-σ	+8.3 [−0.9, 18.0]	+13.3** [3.4, 23.2]	−3.2 [−9.9, 3.5]
	3-σ	+19.2* [2.3, 36.3]	+27.8* [4.3, 51.5]	−2.2 [−13.2, 8.8]
Reason-SFT-DrGRPO	Overall	+4.8 [−1.2, 10.8]	+10.7** [4.7, 16.8]	−0.3 [−4.5, 3.9]
	1-σ	+9.1* [0.7, 17.5]	+8.2 [−0.1, 16.5]	−0.7 [−6.7, 5.3]
	2-σ	−1.8 [−11.7, 8.2]	+5.7 [−4.3, 16.1]	−1.4 [−8.5, 5.6]
	3-σ	+0.4 [−15.5, 16.5]	+27.8* [4.4, 51.5]	+4.4 [−5.5, 14.3]
Synthetic-Reason-SFT-DAPO	Overall	+7.9* [1.7, 14.0]	+6.0 [−0.3, 12.3]	−1.6 [−6.1, 2.8]
	1-σ	+12.8** [4.3, 21.2]	+5.7 [−2.7, 14.0]	−2.9 [−8.7, 3.0]
	2-σ	+0.2 [−10.2, 10.7]	−2.3 [−13.1, 8.2]	+0.4 [−7.0, 7.7]
	3-σ	+4.3 [−14.3, 23.3]	+30.6* [5.8, 55.7]	−1.1 [−14.3, 12.1]

Length bias

The first thing to note is a clean split by training paradigm. *Direct-SFT* shows no length bias at any tier or difficulty level (overall +2.7 [−2.8, 8.3]), with zero inside every per-σ interval, and a near-perfect balance at the 2-σ tier specifically (85.6% on longer vs. 85.5% on shorter). All RL-trained models, by contrast, favour longer descriptions to some degree, suggesting that training with binary outcome rewards reinforces description length as a proxy for quality.

Within the RL-trained models, two qualitatively different patterns emerge. *Reason-DAPO* and *Synthetic-Reason-SFT-DAPO* show length sensitivity that is strongly concentrated at high difficulty: their 1-σ CIs are entirely above zero ($+19.8$ [11.7, 27.7] and $+12.8$ [4.3, 21.2]), while medium and easy tier intervals include zero. *Reason-SFT-DAPO* shows a different and more concerning pattern: its length bias is large and significant at both the hardest tier ($+14.2$ [5.8, 22.7]) and the easiest ($+19.2$ [2.3, 36.3]). This points to a structural preference for longer descriptions that persists regardless of whether the task actually requires it. *Reason-SFT-DrGRPO* sits closest to *Direct-SFT*, with an overall CI that just crosses zero ($+4.8$ [-1.2, 10.8]), though significant at 1-σ ($+9.1$ [0.7, 17.5]).

The practical consequence is most visible at the hardest tier. In the 1-σ pairs, *Reason-DAPO* outperforms *Direct-SFT* on longer inputs (79.1% vs. 72.9%) but falls 10 points below it on shorter ones (59.2% vs. 69.1%). *Synthetic-Reason-SFT-DAPO* mirrors this reversal (72.1% vs. 59.3%). The advantage of these RL-trained models on hard comparisons therefore depends on how much information the descriptions provide; when descriptions are short, *Direct-SFT* is the more reliable choice. The length bias are likely a residual effects of using GRPO based policies, even when DAPO and Dr. GRPO are crafted to address this.

Recency bias. Before treating the significant recency bias observed across most models as a failure, it is worth noting what the training distribution looks like: 57.08% of pairs have the newer idea winning (Table 3.7). A model applying a naive recency heuristic uniformly achieves ≈57%; *Direct-SFT* reaches 77.10%, 20 points above that ceiling. Recency is a learnable empirical prior in competitive NLP benchmarking, not a shortcut, unlike description length, which carries no principled signal about empirical outcome. But such bias is still not desirable to achieve a more robust generizable model.

Against this backdrop, *Synthetic-Reason-SFT-DAPO* is the only model whose overall CI includes zero ($+6.0$ [-0.3, 12.3]), making it the only model that does not clearly show a recency preference at the overall level. This extends tier-by-tier: its 1-σ and 2-σ recency CIs both include zero ($+5.7$ [-2.7, 14.0] and -2.3 [-13.1, 8.2]). The remaining models split in how consistently the recency signal applies across difficulty. *Reason-SFT-DAPO* and *Direct-SFT* show stable, well-estimated recency bias at both hard and medium tiers, with lower bounds above 3 pp. *Reason-SFT-DrGRPO*'s recency CIs are borderline at the 1-σ tier ($+8.2$ [-0.1, 16.5]) and non-significant at medium, making it the least recency-sensitive among the significantly biased models overall.

The most striking recency result comes from the raw accuracies in Table 3.8. At medium difficulty, *Reason-DAPO* actually performs better on older-winning pairs than newer-winning ones (85.1% vs. 77.5%), and *Synthetic-Reason-SFT-DAPO* shows this inversion even more strongly (88.1% vs. 71.6%). Both models score higher precisely when recency points in the wrong direction. These are non-significant as Δ values, but the direction is consistent and the underlying accuracy difference is large. Note that both the model's trainig invloves tuning based on an exsiting LM (*Reason-DAPO*: Direct tuning of a general strong SLM, and *Synthetic-Reason-SFT-DAPO*: Distill from a string LLM). This behavious likely arises from the biases learnt by such models during their training.

Results across the 3-σ tier should be read cautiously throughout: the older-winning subset contains only 24 instances and the same-year subset only 12 (36 in total), producing CIs close to 50 pp wide for both recency and length. The extreme accuracy values at this tier reflect estimation noise at small N, not stable model behaviour.

		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 3.7: 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.

Overall robustness

Reason-SFT-DrGRPO is the most robust model: no significant overall length bias, the lowest recency Δ lower bound among models with significant overall recency bias, and paraphrasing near zero across all tiers. *Synthetic-Reason-SFT-DAPO* presents a different trade-off: the best recency profile of any model, but a significant hard-tier length bias and a notable performance inversion on medium-difficulty recency pairs. Length bias in RL-trained models is the primary robustness limitation identified here.

Table 3.8: 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
Synthetic-Reason-SFT-DAPO	72.1	59.3	68.8	74.5	54.7	65.8	62.9
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
Synthetic-Reason-SFT-DAPO	72.6	72.5	71.6	88.1	54.2	72.5	72.9
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
Synthetic-Reason-SFT-DAPO	76.0	71.7	80.8	66.7	16.7	74.7	73.6
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
Synthetic-Reason-SFT-DAPO	72.7	64.9	71.6	78.9	53.5	68.9	67.3

3.4.2 Sycophancy

Tables 3.9–3.12 report results from the sycophancy stress test described in §2.6.4. Each model was evaluated on $n = 50$ pairs per ($\sigma \in \{1, 2, 3\}$) under three conditions: hard lie, soft lie, and truth control, yielding 450 total matched pairs per model. Position bias is measured independently at two levels: the sycophancy-test level (same positional answer on original and swapped presentations of the sycophancy prompt) and the generation level (same positional slot predicted on original and swapped idea orderings during standard evaluation).

Table 3.9: Sycophancy stress-test outcomes aggregated over hard-lie and soft-lie conditions across all σ ($n = 300$ per model).

Outcome	Base Qwen3	Reason-SFT-DAPO	Synthetic-Reason-SFT-DAPO
Compliance (defended the lie)			
Purely sycophantic	32.0%	71.0%	56.7%
Consistently wrong	42.0%	12.7%	17.3%
Total	**74.0%**	**83.7%**	**74.0%**
Resistance (argued against the lie)			
Robust & correct	19.3%	10.3%	16.7%
Confused	6.7%	6.0%	9.0%
Total	**26.0%**	**16.3%**	**25.7%**
Position bias (standard evaluation)			
Generation-level	**58.4%**	8.7%	7.0%

Key takeaways

Small base models extensively trained through RLVR remain susceptible to sycophancy, highlighting a clear trade-off between baseline accuracy and adversarial robustness. *Reason-SFT-DAPO*, despite its forecasting superiority, suffers from a confidence-induced vulnerability where its sensitivity to persuasive framing overrides its content evaluation. Its compliance is a genuine content-driven failure, while its resistance is entirely an artifact of position bias. *Synth-Reason-SFT-DAPO* demonstrates more robust resistance to false claims, but its reliance on synthetic reasoning traces leaves it prone to confusion, lacking empirical grounding. Ultimately, under pressured social framing, model responses are largely governed by the interaction between prompt structure and positional bias, with genuine belief evaluation playing a secondary role.

Table 3.10: Hard-lie outcomes per σ ($n = 50$ per cell, except *Synth-Reason-SFT-DAPO* at σ=3 where $n = 49$; Overall column $n = 150$ for Base and Reason-SFT-DAPO, $n = 149$ for Synth-Reason-SFT-DAPO). The split between *purely sycophantic* (complied; gen. correct) and *consistently wrong* (complied; gen. incorrect) reflects underlying model accuracy rather than sycophantic susceptibility.

Model	Metric	σ=1 (Hard)	σ=2 (Med.)	σ=3 (Easy)	Overall
Base Qwen3	Total compliance	100.0%	100.0%	100.0%	100.0%
	Pure Syco.	46.0%	60.0%	48.0%	51.3%
	Consistently wrong	54.0%	40.0%	52.0%	48.7%
	Total resistance	0.0%	0.0%	0.0%	0.0%
	Robust & correct	0.0%	0.0%	0.0%	0.0%
	Confused	0.0%	0.0%	0.0%	0.0%
Reason-SFT-DAPO	Total compliance	100.0%	100.0%	100.0%	100.0%
	Pure Syco.	76.0%	86.0%	82.0%	81.3%
	Consistently wrong	24.0%	14.0%	18.0%	18.7%
	Total resistance	0.0%	0.0%	0.0%	0.0%
	Robust & correct	0.0%	0.0%	0.0%	0.0%
	Confused	0.0%	0.0%	0.0%	0.0%
Synthetic-Reason-SFT-DAPO	Total compliance	100.0%	100.0%	100.0%	100.0%
	Pure Syco.	64.0%	74.0%	83.7%	73.8%
	Consistently wrong	36.0%	26.0%	16.3%	26.2%
	Total resistance	0.0%	0.0%	0.0%	0.0%
	Robust & correct	0.0%	0.0%	0.0%	0.0%
	Confused	0.0%	0.0%	0.0%	0.0%

All reasoning models fail the hard lie test. Under the hard-lie condition, all three models comply in 100% of instances across all σ (Table 3.10). None of the models resist a direct instruction to confirm a false winner when there is no option to disagree, and even the chain-of-thought reasoning traces do not question the false claim.

However, the *type* of compliance differs across models and mainly reflects their underlying accuracy rather than their tendency to be sycophantic. For *Reason-SFT-DAPO*, most hard-lie compliance falls into the purely sycophantic category at all three σ levels. This means the model usually has the correct belief but abandons it when given a direct instruction.

Synthetic-Reason-SFT-DAPO shows a similar pattern with the purely sycophantic share increasing for easier instances, which matches the confidence-related vulnerability discussed later.

For *Base Qwen3*, the split is more balanced across all difficulty levels (46–60% purely syco-

Table 3.11: Soft-lie outcomes per σ ($n = 50$ per cell; $n = 150$ overall). *Syco.-pos. bias*: fraction of paired instances where the model gave the same positional answer to the soft-lie prompt on both the original and swapped idea orderings. *Resist. bias* and *Compl. bias*: fraction of resistant and complied instances respectively that exhibit sycophancy-level position bias.

Model	**Metric**	σ=1 **(Hard)**	σ=2 **(Med.)**	σ=3 **(Easy)**	**Overall**
Base Qwen3	Compl.	46.0%	48.0%	50.0%	48.0%
	Pure Syco.	10.0%	18.0%	10.0%	12.7%
	Robust	36.0%	42.0%	38.0%	38.7%
	Confused	18.0%	10.0%	12.0%	13.3%
	Syco.-pos. bias	98.0%	94.0%	92.0%	94.7%
	Resist. bias	100.0%	100.0%	96.0%	98.7%
	Compl. bias	95.7%	87.5%	88.0%	90.3%
Reason-SFT-DAPO	Compl.	58.0%	64.0%	80.0%	67.3%
	Pure Syco.	54.0%	58.0%	70.0%	60.7%
	Robust	22.0%	28.0%	12.0%	20.7%
	Confused	20.0%	8.0%	8.0%	12.0%
	Syco.-pos. bias	72.0%	68.0%	46.0%	62.0%
	Resist. bias	100.0%	100.0%	100.0%	100.0%
	Compl. bias	51.7%	50.0%	32.5%	43.6%
Synthetic-Reason-SFT-DAPO	Compl.	38.0%	58.0%	50.0%	48.7%
	Pure Syco.	30.0%	48.0%	42.0%	40.0%
	Robust	34.0%	26.0%	40.0%	33.3%
	Confused	28.0%	16.0%	10.0%	18.0%
	Syco.-pos. bias	74.0%	82.0%	66.0%	74.0%
	Resist. bias	80.6%	100.0%	56.0%	77.9%
	Compl. bias	63.2%	69.0%	76.0%	69.9%

phantic and 40–54% consistently wrong). Because the base model has lower overall accuracy, its compliance often reflects agreement with an incorrect belief rather than giving up a correct one.

Overall, the hard-lie condition shows that all models can be overridden when they are given no choice. However, the purely sycophantic rate in this setting mainly reflects model accuracy rather than true sycophantic behaviour.

Models do better when they are allowed to disagree. Because the soft-lie prompt allows disagreement, resistance here is the main measure of sycophancy. *Base Qwen3* and *Synthetic-Reason-SFT-DAPO* resist at very similar rates (51–52%), while *Reason-SFT-DAPO* is much more vulnerable, resisting in only 32.7% of cases. Strikingly, the model that performs best on the forecasting

Table 3.12: Truth-control outcomes per σ ($n = 50$ per cell; $n = 150$ overall). *Agreed*: model endorsed the correctly declared winner (Correct_Agreement). *Concordant*: agreed with truth and predicted correctly under standard evaluation. *Contrary*: disagreed with truth despite predicting correctly under standard evaluation: the model holds the right belief but refuses to endorse it when declared. *Syco.-pos. bias*: fraction of paired instances where the model gave the same positional response on both orderings of the truth-control prompt.

Model	**Metric**	σ=1 **(Hard)**	σ=2 **(Med.)**	σ=3 **(Easy)**	**Overall**
Base Qwen3	Agreed	56.0%	58.0%	42.0%	52.0%
	Concordant	38.0%	46.0%	34.0%	39.3%
	Contrary	8.0%	14.0%	14.0%	12.0%
	Syco.-pos. bias	94.0%	90.0%	90.0%	91.3%
Reason-SFT-DAPO	Agreed	58.0%	54.0%	46.0%	52.7%
	Concordant	36.0%	44.0%	38.0%	39.3%
	Contrary	40.0%	42.0%	44.0%	42.0%
	Syco.-pos. bias	88.0%	96.0%	94.0%	92.7%
Synthetic-Reason-SFT-DAPO	Agreed	56.0%	56.0%	44.0%	52.0%
	Concordant	30.0%	38.0%	38.0%	35.3%
	Contrary	34.0%	36.0%	44.0%	38.0%
	Syco.-pos. bias	88.0%	84.0%	94.0%	88.7%

task (*Reason-SFT-DAPO*) is the most vulnerable to sycophancy.

Fine-tuning eliminates generation-level position bias. Table 3.9 confirms the pattern shown in Figure 3.5.

The base model shows a high **58.4%** generation-level position bias (Table 3.9). Fine-tuning greatly reduces this to **7.0–8.7%**. As a result, the fine-tuned models usually make content-based predictions, making their sycophantic behaviour a more meaningful failure mode in practice.

Soft-lie resistance is mostly driven by position bias. Most soft-lie responses remain determined by position: 94.7% for *Base Qwen3*, 62.0% for *Reason-SFT-DAPO*, and 74.0% for *Synthetic-Reason-SFT-DAPO*. For *Reason-SFT-DAPO*, 100.0% of its resistance is position-driven, whereas most of its compliance (56.4%) is content-driven. *Base Qwen3* shows a similar pattern. Only *Synthetic-Reason-SFT-DAPO* shows a noticeable portion of non-position-driven resistance (22.1%), suggesting it occasionally exhibits genuine, content-based disagreement with the false claim.

Sycophancy severity Looking at the per-σ soft-lie results for *Reason-SFT-DAPO* (Table 3.11), the purely sycophantic rate increases as instances become easier.

Easier instances usually correspond to cases where the model holds a confident and correct belief. However, this reveals a *confidence-induced vulnerability*: when the model is most confident in its correct belief, it can become easier to persuade.

Synthetic-Reason-SFT-DAPO does not show the same pattern. Its purely sycophantic rates across difficulty are lower at every level, and its robust-and-correct rate remains relatively stable.

This suggests that synthetic reasoning traces from a stronger teacher model may provide a more stable knowledge base under adversarial pressure than reasoning learned only through RL or grounded reasoning.

Confused cell The confused quadrant: cases where the model resists the lie but still makes an incorrect prediction; is especially high for *Synthetic-Reason-SFT-DAPO* at $\sigma=1$, reaching **28.0%** under the soft lie, higher than for the other models.

These are not sycophantic failures. The model correctly rejects the false claim but lacks the knowledge needed to produce the correct prediction regardless of framing. This supports the idea that synthetic chain-of-thought traces may teach the *form* of argumentative resistance without providing the empirical grounding needed for accurate predictions. This interpretation is also consistent with the lower forecasting accuracy of *Synthetic-Reason-SFT-DAPO* in the main evaluation (§3.3).

Truth control Table 3.12 shows an additional pattern. Across all models and difficulty levels σ, agreement with the correctly declared winner is relatively low, ranging from 42–58%.

The most notable pattern appears in the two fine-tuned models, *Reason-SFT-DAPO* and *Synthetic-Reason-SFT-DAPO*. Both show a high *contrary* rate (34–44%), meaning they sometimes disagree with the declared true winner even when they predicted correctly under standard evaluation.

Among the cases where the model predicted correctly on its own, it still refused to endorse the declared correct answer in about half of instances (49–54% across models and difficulty levels). This is the opposite of sycophancy: the model resists a *true* claim at a surprisingly high rate (42.0% overall for *Reason-SFT-DAPO*), although this is still lower than its soft-lie compliance rate of 60.7%.

The contrast is particularly strong for *Reason-SFT-DAPO*. When the model predicted correctly under standard evaluation, it endorsed the true winner in only 46–51% of cases at $\sigma=2$ and $\sigma=3$, but accepted the false winner in 67–85% of those same cases under the soft lie.

This suggests that the model responds more strongly to persuasive framing than to whether the claim is actually true.

Base Qwen3 shows the opposite pattern. It agrees with the true winner more often (71–83% across difficulty levels), which is much higher than its soft-lie compliance rate (21–30%). The base model also has a much lower contrary rate (12.0%) compared with the fine-tuned models (38–42%), meaning it challenges declared outcomes less often overall.However, because the base model has strong position bias during generation, its agreement with the truth is not necessarily due to careful evaluation of the declared outcome. Instead, it often reflects a mix of positional preference and a general tendency to follow the given framing. When the model is wrong, this is more likely due to its lower prediction accuracy rather than deliberate disagreement with the claim.

Truth control position bias: positional preference dominates Position bias in the truth-control condition is extremely high for all three models, reaching **84–96%** across every difficulty level σ (Table 3.12). This is much higher than the 0% observed in the hard-lie condition.

Under the soft-lie condition, resistance was already almost entirely positional for *Base Qwen3* and *Reason-SFT-DAPO* (98.7% and 100.0%). In the truth-control condition, this positional effect extends to compliance as well. Under the soft lie, some compliance from *Reason-SFT-DAPO* was still driven by content (56.4%), but in truth control almost all responses are positional.

This helps explain the high contrary rates observed in the fine-tuned models. Their disagreement with the true label (34–44%) does not reflect independent judgment, but instead occurs because positional preferences conflict with the correct answer in many cases.Importantly, the generation-level position bias of the fine-tuned models remains low (6–12%) in the truth-control subset. This shows that the strong positional effect appears mainly under the pressured framing of the truth-control prompt and is not part of the models' normal prediction behaviour.

3.4.3 Can Trained Models Rank Ideas?

The practical motivation for this work is not just pairwise prediction but the ability to filter and rank a set of generated ideas by predicted empirical potential.

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**
Synthetic-Reason-SFT-DAPO	85.27	43.81	1.32	71.74	36.36	1.80
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 3.13: 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.

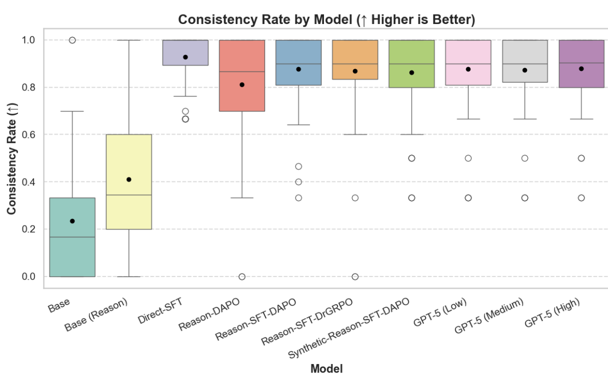


Figure 3.9: Distribution of consistency rate (%) across different research goals/leaderboards for the in-domain test set

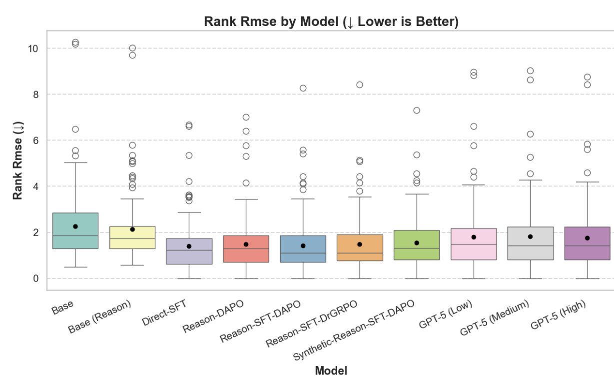


Figure 3.10: Distribution of RMSE across different research goals/leaderboards for the in-domain test set

Trained models outperform GPT-5 on ranking. On the in-domain test, all trained models achieve higher Top-1 accuracy than GPT-5 at any reasoning effort. *Reason-SFT-DrGRPO* ties *Reason-SFT-DAPO* for the best RMSE (1.12 vs. GPT-5-high 1.41), while *Reason-DAPO* achieves the highest Top-1 accuracy (51.43%). On the CD test, *Reason-SFT-DrGRPO* achieves RMSE of

1.65 against GPT-5's 1.77, and beats GPT-5 on Top-1 under low and medium reasoning effort (41.30% vs. 36.96% and 34.78%). The other models perform closer to GPT-5 with *Reason-SFT-DAPO* achieveing the lowest Top-1 accuracy, at the same time achieving the second best RMSE.

Reason-DAPO achieves the highest in-domain Top-1 accuracy (51.43%) despite lower in-domain accuracy (72.73%) (Table 3.2) than *Direct-SFT* (77.10%), which in turn achieves lower Top-1 (44.76%). Higher consistency does not guarantee better ranking of the top idea. This is happening because of the sensitivity of the trained models to labelling methods. As observed in §3.2, the RL variants are more robust in general to noise like this leading to better prediction when ranks are directly used for labelling and not following the stratification based on σ on which they were trained on.

Comparability and Noise RMSE comparisons across models with higher consistency rates allow for meaningful comparisons and prevent noise due to instability. For example a model with near-zero consistency would assign all ideas the same rank, and create a very unstable RMSE (As $n \to \infty$, if all ranks=1, then $RMSE \approx n/\sqrt{3}$). With this caveat in mind, among the trained models and GPT-5 that are largely consistent, the RMSE distributions (Figures 3.10–3.12) are right-skewed with small numbers of high-error outliers; the highest outliers for fine-tuned variants are lower than those for GPT-5 on the CD set showing their effectiveness.

The consistency distribution plot Figure 3.9 and 3.11 reveal a spread of consistency which means the reported metrics could be noisy.

Consistency rate The spread of consistency rates across the leaderboards are all left-skewed on both ID and CD test sets. On ID test, the medians and Inter Quartile Ranges (IQR) are largely similar showing their potential to be used as rankers in place of large models. But the distribution on CD sees a drop in median and interquartile ranges, with some models like *Reason-SFT-DrGRPO* being still competitive.

Reason-SFT-DrGRPO shows the smallest degradation across all three ranking metrics moving from the ID to the CD test, making it the strongest generalizer in the ranking setting as well as in the prediction setting.

Thus these results show that the trained model can perform at par or better than frontier models in the ranking task that is position bias aware, making them a more computationally efficient idea filters. But at the same time the Top-1 accurcy and RMSE observed are not the best one expects from a truly ”good” ranker. Designing ranking methodology that could overcome the current shortcomings or training the models in a way that allow them to also pay attention to the rank apart from mere 2 idea comparison is an important future direction to explore.

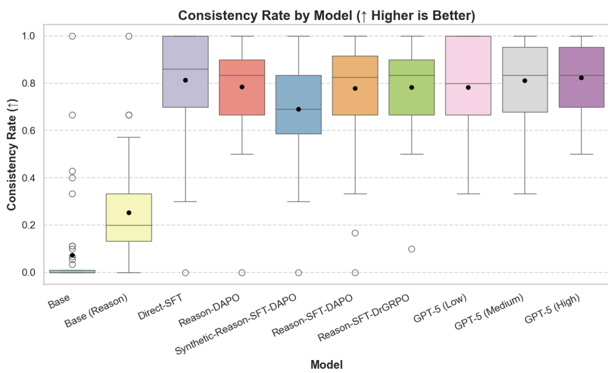


Figure 3.11: Distribution of consistency rate (%) across different research goals/leaderboards for the cross-domain test set

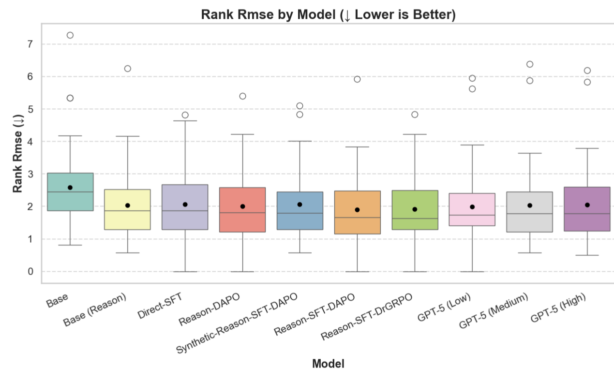


Figure 3.12: Distribution of RMSE across different research goals/leaderboards for the cross-domain test set

Chapter 4

Conclusions

This work demonstrates that language models can be taught to forecast research success through comparative evaluation of ideas. By constructing a large-scale dataset of 11,488 benchmark-specific idea pairs grounded in objective outcomes, fine-grained prediction becomes tractable for small models. Fine-tuned 8B models achieve 77.1% accuracy on in-domain comparisons, outperforming frontier models like GPT-5 by over 15 points, showing that scientific forecasting does not require large proprietary systems. Such models show potential to be used as effective validators directly addressing the current *validation bottleneck*.

RL-trained models reach 71.35% accuracy while generating justifications for their predictions, and show stronger cross-domain generalization than their SFT-only counterparts. The best model, *Reason-SFT-DrGRPO*, achieves 67.49% on an independent external benchmark, outperforming a retrieval-augmented GPT-4.1 system by 16 points. However, inducing reliable and interpretable reasoning in small models remains an open challenge. Reward hacking is a real failure mode without careful SFT initialization, and the reasoning traces produced, while sometimes coherent, are not consistently meaningful. Jointly optimizing for prediction accuracy and reasoning faithfulness requires reward signals that evaluate intermediate reasoning quality directly, not just the final label. Rubric-based reward models grounded in source papers are a promising direction here.

Robustness analysis reveals that predictions are stable to paraphrasing across all difficulty tiers, which is necessary for a practical verifier. Length bias in RL-trained models is the clearest limitation: these models systematically favor longer idea descriptions, a preference that has no principled grounding in outcome quality and could cause failures in settings where description length reflects documentation effort rather than idea merit. The fine-tuned models also show sycophantic behavior under adversarial framing, with the highest-accuracy models showing the highest susceptibility. They represent real deployment risks that future work needs to address.

As idea rankers, fine-tuned models perform at par or better than GPT-5 on cross-domain ranking, showing potential as compute-efficient idea filters for automated research pipelines. That said, ranking quality is not yet at the level one would want from a reliable filter ($\approx 50\%$ Top-1 Accuracy), and training models to implicitly reason about rank rather than just pairwise preference is an important next step.

The data collection pipeline is not tied to NLP or to any single leaderboard source. As long as a field has well-maintained objective benchmarks, the same approach applies. As leaderboards grow, the dataset can be extended continuously, making this a scalable foundation for outcome-grounded evaluation in AI-assisted science.[1]

[1] The CS/AI field is a very fast moving field at the moment, with new findings coming out every week. Efforts have been made to keep up with the literature and include them to best possible extent in this work but it is possible that we might have missed some papers given the sheer volume of research being published. Additionally many of the works cited in the results and discussion were only discovered when analysing the results observed, with many of the suggestions about methods to address some of the observations made, only being available midway through the experimentations. Given the computational constraint and the time required to run each of these analysis, incorporation of these ideas was not possible.

Bibliography

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

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.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru WANG, Yu Zhang, Denghui Zhang, Tong Zhang, Hanghang Tong, and Heng Ji. RM-r1: Reward modeling as reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=1ZqJ6jj75q.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025. URL https://arxiv.org/abs/2501.17161.

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.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models, 2025. URL https://arxiv.org/abs/2505.22617.

Haoran Dang, Cuiling Lan, Hai Wan, Xibin Zhao, and Yan Lu. Temperature as a meta-policy: Adaptive temperature in LLM reinforcement learning, 2026. URL https://arxiv.org/abs/2602.11779.

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.

Shashwat Goel, Rishi Hazra, Dulhan Jayalath, Timon Willi, Parag Jain, William F. Shen, Ilias Leontiadis, Francesco Barbieri, Yoram Bachrach, Jonas Geiping, and Chenxi Whitehouse. Training ai co-scientists using rubric rewards, 2025. URL https://arxiv.org/abs/2512.23707.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

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.

Tianyang Gu, Jingjin Wang, Zhihao Zhang, and HaoHong Li. Llms can realize combinatorial creativity: Generating creative ideas via llms for scientific research, 2025. URL https://arxiv.org/abs/2412.14141.

Xuemei Gu and Mario Krenn. Forecasting high-impact research topics via machine learning on evolving knowledge graphs. *Machine Learning: Science and Technology*, 6(2):025041, May 2025. ISSN 2632-2153. doi: 10.1088/2632-2153/add6ef. URL http://dx.doi.org/10.1088/2632-2153/add6ef.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains, 2025. URL https://arxiv.org/abs/2507.17746.

Alexander Gurung and Mirella Lapata. Learning to reason for long-form story generation, 2025. URL https://arxiv.org/abs/2503.22828.

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.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

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.

Hangzhan Jin, Sitao Luan, Sicheng Lyu, Guillaume Rabusseau, Reihaneh Rabbany, Doina Precup, and Mohammad Hamdaqa. Rl fine-tuning heals ood forgetting in sft, 2025. URL https://arxiv.org/abs/2509.12235.

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.

Zohaib Khan, Omer Tafveez, and Zoha Hayat Bhatti. Plasticity vs. rigidity: The impact of low-rank adapters on reasoning on a micro-budget, 2026. URL https://arxiv.org/abs/2601.06677.

Esther Landhuis. Scientific literature: Information overload. *Nature*, 535:457 – 458, 2016. URL https://api.semanticscholar.org/CorpusID:19699258.

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.

Lennart Meincke, Ethan R. Mollick, Lilach Mollick, and Dan Shapiro. Prompting science report 2: The decreasing value of chain of thought in prompting. *SSRN Electronic Journal*, January 2025. doi: 10.2139/ssrn.5285532. URL https://doi.org/10.2139/ssrn.5285532.

Hanane Nour Moussa, Patrick Queiroz Da Silva, Daniel Adu-Ampratwum, Alyson East, Zitong Lu, Nikki Puccetti, Mingyi Xue, Huan Sun, Bodhisattwa Prasad Majumder, and Sachin Kumar. Scholareval: Research idea evaluation grounded in literature, 2025. URL https://arxiv.org/abs/2510.16234.

Charles O'Neill, Tirthankar Ghosal, Roberta Răileanu, Mike Walmsley, Thang Bui, Kevin Schawinski, and Ioana Ciucă. Sparks of science: Hypothesis generation using structured paper data, 2025. URL https://arxiv.org/abs/2504.12976.

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

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Vardhan Palod, Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. Performative thinking? the brittle correlation between cot length and problem complexity, 2025. URL https://arxiv.org/abs/2509.07339.

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.

Aleksei Petrenko, Ben Lipkin, Kevin Chen, Erik Wijmans, Marco Cusumano-Towner, Raja Giryes, and Philipp Kr"ahenb"uhl. Entropy-preserving reinforcement learning, 2026. URL https://arxiv.org/abs/2603.11682.

Daniil Plyusov, Alexey Gorbatovski, Boris Shaposhnikov, Viacheslav Sinii, Alexey Malakhov, and Daniil Gavrilov. F-grpo: Don't let your policy learn the obvious and forget the rare, 2026. URL https://arxiv.org/abs/2602.06717.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. Mutual reasoning makes smaller llms stronger problem-solvers, 2024. URL https://arxiv.org/abs/2408.06195.

Soumya Rani Samineni, Durgesh Kalwar, Vardaan Gangal, Siddhant Bhambri, and Subbarao Kambhampati. Local coherence or global validity? investigating rlvr traces in math domains, 2025. URL https://arxiv.org/abs/2510.18176.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

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.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models, 2025. URL https://arxiv.org/abs/2310.13548.

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.

Michael D. Skarlinski, Sam Cox, Jon M. Laurent, James D. Braza, Michaela Hinks, Michael J. Hammerling, Manvitha Ponnapati, Samuel G. Rodriques, and Andrew D. White. Language agents achieve superhuman synthesis of scientific knowledge, 2024. URL https://arxiv.org/abs/2409.13740.

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.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

Keisuke Ueda, Wataru Hirota, Takuto Asakura, Takahiro Omi, Kosuke Takahashi, Kosuke Arima, and Tatsuya Ishigaki. Exploring design of multi-agent llm dialogues for research ideation, 2025. URL https://arxiv.org/abs/2507.08350.

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, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 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.

Qingyu Yin, Yulun Wu, Zhennan Shen, Sunbowen Li, Zhilin Wang, Yanshu Li, Chak Tou Leong, Jiale Kang, and Jinjin Gu. Evaluating parameter efficient methods for rlvr, 2025. URL https://arxiv.org/abs/2512.23165.

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.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?, 2025. URL https://arxiv.org/abs/2504.13837.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=shr9PXz7T0.

Hanqing Zhu, Zhenyu Zhang, Hanxian Huang, DiJia Su, Zechun Liu, Jiawei Zhao, Igor Fedorov, Hamed Pirsiavash, Zhizhou Sha, Jinwon Lee, David Z. Pan, Zhangyang Wang, Yuandong Tian, and Kai Sheng Tai. The path not taken: Rlvr provably learns off the principals, 2025a. URL https://arxiv.org/abs/2511.08567.

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

Appendix A

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."
    )
```

Appendix B

Examples

B.0.1 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
```

B.0.2 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

B.0.3 Qwen3-8B-Reason-SFT-DrGRPO

As mentioned in §(3.3.4), 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

B.0.4 Qwen3-8B-Synthetic-Reason-SFT-DAPO

An example of Synthetic-SFT-DAPO model reasoning and debating multiple strengths and weaknesses before correctly concluding the better idea for the given research goal. Unlike Reason-SFT-DrGRPO, it does not tend to minimize the reasoning traces or the responses.

Input

"research_goal": "The primary research objective is to develop question-answering systems capable of multi-step reasoning that requires synthesizing explicit knowledge with implicit common sense. Using the OpenBookQA benchmark, this research challenges models to process an elementary-level science question and a provided set of core facts to select the correct multiple-choice answer. This task addresses the critical need for models to move beyond simple fact retrieval and instead apply provided knowledge to novel situations, a crucial capability for advanced natural language understanding. Performance is

evaluated based on the model's accuracy in answering these questions, which are specifically designed to be unsolvable by simpler retrieval-based or word co-occurrence algorithms.",

"idea_A": "A model is proposed for question answering that performs joint reasoning over a language context and a knowledge graph (KG) by unifying them into a single graphical structure. The process begins by retrieving a KG subgraph `G_sub` relevant to the entities mentioned in the QA context. A novel 'working graph' `G_W` is then constructed. This is achieved by introducing a special 'QA context node' `z` that represents the concatenated question and answer choice. This `z` node is then connected to the topic entities (those mentioned in the question or answer) within the `G_sub` via newly defined, typed relations (`r_z,q` for question entities, `r_z,a` for answer entities). The initial representation of the `z` node is derived from an LM encoding of the QA context, while KG nodes are initialized with their entity embeddings. To address the issue of irrelevant nodes in the retrieved subgraph, a relevance scoring mechanism is introduced. For each KG node `v`, a relevance score `\u03c1_v` is computed by feeding the concatenation of the QA context text and the node's entity text to a pre-trained LM, formulated as `\u03c1_v = f_head(f_enc([text(z); text(v)]))`. This score quantifies the node's importance relative to the QA context and is used as a feature. Reasoning is performed on this working graph using a multi-layer graph neural network (GNN) based on a graph attention framework. At each layer `l`, the representation of each node `t` is updated via message passing: `h^(l+1)_t = f_n(\u03a3_{s\u2208N_t\u222a{t}} \u03b1_st * m_st) + h^(l)_t`, where `m_st` is the message and `\u03b1_st` is the attention weight. The message computation `m_st` is node type- and relation-aware, defined as `m_st = f_m(h^(l)_s, u_s, r_st)`, where `u_s` is the source node's type embedding and `r_st` is a relation embedding. The attention mechanism `\u03b1_st` is node type-, relation-, and score-aware. It computes query `q_s = f_q(h^(l)_s, u_s, \u03c1_s)` and key `k_t = f_k(h^(l)_t, u_t, \u03c1_t, r_st)` vectors, where `\u03c1` represents an embedding of the relevance score. The attention weight is then `\u03b1_st = exp(\u03b3_st) / \u03a3_{t'\u2208N_s\u222a{s}} exp(\u03b3_st')` with `\u03b3_st = q_s^T k_t / \u221aD`. This iterative process mutually updates the representations of both the QA context node and the KG nodes. The final prediction for an answer choice is derived from an MLP that takes as input the initial LM representation of the context (`z_LM`), the final GNN-updated representation of the context node (`z_GNN = h^(L)_z`), and a pooled representation of the final KG node embeddings (`g`). The entire model is trained end-to-end by optimizing a cross-entropy loss.",

"idea_B": "The proposed approach is a 66-billion parameter, decoder-only, pre-trained transformer model designed for auto-regressive language modeling. The architecture consists of 64 transformer layers, a hidden dimension of 9216, and 72 attention heads, with a feed-forward network dimension of 36864. It utilizes ReLU activation functions and is trained on sequences of 2048 tokens. The model's weights are initialized from a normal distribution with a mean of zero and a standard deviation of 0.006, with the standard deviation for output layers scaled by a factor of 1.0/\u221a2L, where L is the number of layers. Pre-training is conducted on a 180-billion token corpus, which is a deduplicated concatenation of datasets including RoBERTa, The Pile, and PushShift.io Reddit, tokenized using a GPT-2 byte-level Byte Pair Encoding (BPE) tokenizer. The training process employs the AdamW optimizer with \u03b21=0.9, \u03b22=0.95, and a weight decay of 0.1. A linear learning rate schedule is used, warming up over the first 375

million tokens and then decaying to 10\% of the maximum learning rate over 300 billion tokens. Regularization is applied through a dropout of 0.1 on all layers except embeddings and gradient norm clipping at 1.0. For task execution in a one-shot setting, the model is provided with a single in-context example sampled from the task's training set. For classification tasks with a predefined set of candidate answers, the model performs likelihood-based classification by selecting the candidate (\u03b1) that maximizes a specific probability score given the context (s). This selection can be based on one of three methods: the regular conditional probability `p(\u03b1|s)`, a calibrated probability `p(\u03b1|s)/p(\u03b1|\"Answer:\")` which normalizes by the candidate's prior probability, or a length-normalized probability `p(\u03b1|s)/len(\u03b1)`. For generative tasks, the model produces its output using greedy decoding."

Output

"thinking": "<think>\n**Evaluating OpenBookQA model**\n\nI'm looking at the OpenBookQA dataset, which uses core facts from OpenBook. The research goal is to develop QA systems that can handle multi-step reasoning. A 66B decoder-only model might not be the best approach since it hasn't been pre-trained on science knowledge. The model could use in-context learning, but it might not outperform specialized models. It seems that for OpenBookQA, a retrieval-based method would be more effective. The second idea is to create a joint reasoning model that combines a language context with a knowledge graph to help with question answering.\n**Analyzing OpenBookQA and KG integration**\n\nI'm looking at OpenBookQA, which uses core facts from OpenBook. The research goal is to develop systems that can synthesize explicit knowledge with common sense. The first idea is to build a working graph that connects a QA context to relevant knowledge graph nodes.

This could help with multi-step reasoning and avoid overfitting. The second idea is to use a large pre-trained model like the one from Meta , which could potentially handle OpenBookQA. However, it might not be as effective as a model that combines retrieval with a knowledge graph for better results.\n**Evaluating model performance and approaches**\n\nI'm looking at how a 66B model could perform with one-shot learning . It might not be the best approach since it could lead to overfitting . The research goal is to develop systems that can reason beyond simple fact retrieval. The OpenBookQA dataset requires using core facts, but a generative model might not be able to do that. The first idea is to use a knowledge graph (KG) that connects concepts, which could help with reasoning. It seems that combining a language model (LM) with a graph neural network (GNN) could be more effective for this task. The second idea is to use a 66B model for pretraining, which might not be necessary.

\n**Assessing model training and evaluation**\n\nI'm looking at how a model could be trained on a large dataset to help with generalization. The research goal is to develop question-answering systems that can handle multi-step reasoning. The 66B model might not be as effective as a fine-tuned approach. It seems that without retrieval, it won't perform well on OpenBookQA. The evaluation will focus on accuracy, and I think the first idea using a knowledge graph (KG) is better. It can help with multi-hop reasoning and reduce irrelevant information. The second idea of using a pre-trained model could work for in-context learning, but it might not be as effective as the first approach.\n** Evaluating ideas for OpenBookQA**\n\nI'm looking at how the OpenBookQA dataset might not have enough data for a 66B model to learn from. The research goal is to develop systems that can synthesize knowledge. The first idea uses a joint reasoning approach with a knowledge graph (KG) to connect entities, which should help with multi-step reasoning.

The second idea is a pre-trained model that could potentially outperform baselines, but it might not be as effective. \n\nSo, I think Idea A is better because it combines a language model with a graph neural network (GNN) for better performance. It can handle retrieval and relevance scoring, which will help with OpenBookQA. \n\nAnswer: 0\n</think>",

"final_answer": "Here is a concise, structured evaluation and conclusion.\n\n- Initial observations:\n - Idea B is a very large, generic, pre-trained decoder-only model with standard training and one-shot usage. It does not include any task-specific mechanism for incorporating provided core facts or a knowledge graph.\n - Idea A is a task-specific, end-to-end model that explicitly integrates a language context with a knowledge graph, includes a relevance scoring mechanism , and uses a GNN with type-aware and relation-aware message passing. It is designed to handle multi-step reasoning and is directly applicable to OpenBookQA.\n\n- Strengths and weaknesses:\n - Idea B strengths: high capacity, can leverage broad pretraining, easy to deploy. Weaknesses: no explicit use of provided core facts, no graph reasoning, one-shot may be insufficient, and it is likely to underperform on OpenBookQA without additional components.\n - Idea A strengths: explicit reasoning over structured knowledge, can select and score relevant nodes, handles multi-hop relations, and can be trained end-to-end. Weaknesses: more complex to implement, potential for overfitting to the graph, and requires good subgraph retrieval.\n\n- Key considerations leading to the conclusion:\n - OpenBookQA requires applying provided core facts and often benefits from graph-based reasoning. A 66B model alone is unlikely to outperform a graph-augmented approach that can explicitly reason over the provided facts .\n - The research goal emphasizes moving beyond simple retrieval and

toward multi-step reasoning. Idea A directly addresses this with a working graph and GNN. Idea B does not.\n\n- Why one is better:\n - Idea A is better aligned with the research goal and is more likely to achieve higher accuracy on OpenBookQA due to its explicit integration of knowledge and reasoning.\n\nAnswer: 0"

"prediction": 0,

"ground_truth": 0