Breaking Barriers: Do Reinforcement Post Training Gains Transfer To Unseen Domains?

Chuxuan Hu[1] **Yuxuan Zhu**[1] **Antony Kellermann** **Caleb Biddulph**

Suppakit Waiwitlikhit **Jason Benn** **Daniel Kang**[1]*

Abstract

Reinforcement post training (RPT) has recently shown promise in improving the reasoning abilities of large language models (LLMs). However, it remains unclear how well these improvements generalize to new domains, as prior work evaluates RPT models on data from the same domains used for fine-tuning. To understand the generalizability of RPT, we conduct two studies. (1) Observational: We compare a wide range of open-weight RPT models against their corresponding base models across multiple domains, including both seen and unseen domains in their fine-tuning data. (2) Interventional: we fine-tune LLMs with RPT on single domains and evaluate their performance across multiple domains. Both studies converge on the same conclusion that, although RPT brings substantial gains on tasks similar to the fine-tuning data, the gains generalize inconsistently and can vanish on domains with different reasoning patterns.

1 Introduction

Large language models (LLMs) have achieved remarkable performance across a wide range of tasks, from structured reasoning domains such as math [15, 30, 39, 73] and code [4, 7, 11, 29, 46, 64, 93], to general reasoning domains including legal [24], finance [90], and medical [23, 33]. Recent advances in reinforcement post training (RPT) [61] have produced dramatic improvements, rivaling top human performers in programming competitions and mathematics contests [6, 16, 21, 22, 25, 40, 42, 43, 54, 60, 67, 69, 78, 79, 87, 91]. This raises an important question: does RPT provide generalizable improvements, as broadly as those achieved through pretraining?

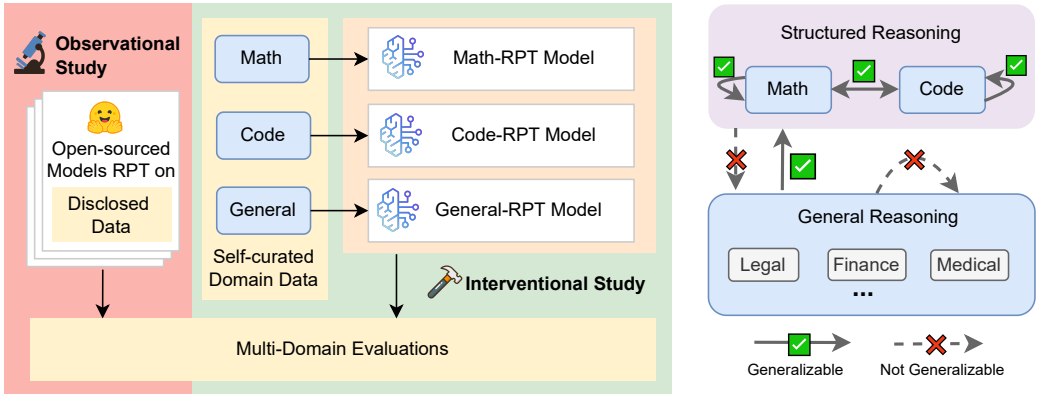
Existing evaluation frameworks and RPT setups provide limited evidence to answer this question. To address it systematically, we design a two-stage investigation pipeline (Figure 1a).

First, prior work evaluates RPT models within their fine-tuning domains [42, 43]. To overcome this limitation, we conduct an *observational study* in which we evaluate 14 recent open-weight RPT models with publicly disclosed fine-tuning data alongside their corresponding base models across a wide range of domains, including legal, financial, and medical benchmarks, spanning their seen and unseen domains. This study is designed to provide an initial view into the generalizability of RPT.

Additionally, we notice that these RPT models, as a representative selection of existing open-weight models such as DeepSeek R1 [18] and RLVR [66], are fine-tuned on mixed domain data. The presence of such confounding factors makes it difficult to isolate and interpret the generalizability of RPT at a finer granularity. To strengthen our findings, we conduct an *interventional study* in which we fine-tune LLMs via RPT on math, coding, and general reasoning data and evaluate their performance on both in-domain and out-of-domain tasks. We illustrate our methodology in detail in Section 3.

*[1]University of Illinois Urbana-Champaign. Correspondence to: Daniel Kang <ddkang@illinois.edu>.

Preprint. Under review.



(a) Overview of our two-stage investigation pipeline. (b) Summary of RPT generalizability.

Figure 1: The method (a) and key findings (b) of our work. Through a unified multi-domain evaluation framework combining observational and interventional studies, we find that RPT exhibits limited generalizability across domains.

As we summarize in Figure 1b, our findings show that gains from RPT on domains involving structured reasoning patterns (e.g., math, code) generalize well within and across structured domains, but fails to generalize to unstructured domains. In contrast, gains from RPT on unstructured domains (e.g., legal, finance) do not generalize well within unstructured domains, but show transferability to structured domains. We analyze these results comprehensively in Section 4.

Our findings suggest that RPT models are most effective when the target task shares reasoning patterns with the RPT data. Consequently, while RPT remains a powerful method for improving LLMs' performance, its benefits are largely limited to the domains represented in the fine-tuning data and do not generalize to a wide range of new, unseen domains.

2 Background

In this section, we introduce the motivation behind our study. We begin by demonstrating the strong performance of LLMs fine-tuned via RPT across various reasoning tasks, particularly in mathematics and coding. We then discuss the limitations of existing work in understanding the mechanisms and boundaries of RPT. We close by introducing the data-domain taxonomy grounds our investigations.

RPT models demonstrate promising performance across a wide range of tasks. RPT models have achieved remarkable improvements on complex reasoning benchmarks. For example, Gemini 2.5 Pro [21] achieves over 90% accuracy on AIME 2024 [30], a math competition benchmark. Grok 3 Beta [78] solves roughly 80% of the tasks in LiveCodeBench v5 [29], a coding benchmark, while Claude 3.7 Sonnet [6] reaches around 85% accuracy on GPQA Diamond [57], a benchmark for graduate-level scientific reasoning.

Out-of-domain generalizability of RPT models remains understudied. Despite impressive results, recent work has examined the limitations of RPT models and the opaque nature of their underlying reasoning capabilities [45, 70, 85, 88]. In particular, there is growing interest in the role of RPT data, especially the extent to which RPT algorithms rely on large, diverse corpora to achieve generalization [91].

While RPT models benefit from training on diverse, mixed-domain data, this diversity makes it difficult to directly assess their generalizability to unseen domains. As a result, prior work evaluates RPT models on tasks within the same domains as their training data [45, 70, 85, 88]. However, many reasoning tasks remain underrepresented or entirely absent from existing training corpora. Exploring the generalizability of LLMs trained with RPT to such tasks is therefore essential for identifying the boundaries of their applicability in real-world, complex scenarios.

Understanding RPT generalizability requires a systematic view of reasoning domains. Following the task suites defined in prior work [66], we focus on three major *domains* of interest: code, math, and general reasoning (Figure 1b). These domains are chosen to capture a broad spectrum of reasoning challenges commonly seen in language model evaluations. The general reasoning domain can be further divided into application-specific subdomains such as legal, finance, medical, etc. In this paper, we not only examine performance across the high-level domains, but also evaluate how reasoning patterns and generalization behaviors vary across subdomains.

Among the three, we consider math and code domain tasks to follow structured reasoning patterns, where solutions follow deterministic logical steps and require precise syntax and formal semantics [66]. In contrast, tasks within the general reasoning domain require more flexible and context-sensitive reasoning, referred to here as *unstructured* reasoning. We define unstructured reasoning as problem-solving processes that do not adhere to a fixed sequence of logical operations and often lack a well-defined intermediate representation or symbolic grounding. Such tasks typically demand broader world knowledge, interpretive judgment, and the ability to handle ambiguity or incomplete information. For instance, legal and financial question answering may involve interpreting lengthy documents, extracting relevant information from loosely connected statements, or evaluating conflicting evidence.

3 Study Design

We present our study design that aims to investigate the generalizability of RPT. We propose the following research questions (RQs) that have not been systematically examined in prior work.

- ***(RQ1)*** *Cross-domain generalization.* To what extent do the capabilities acquired through RPT transfer to tasks from domains not included in the training data?
- ***(RQ2)*** *Role of reasoning structure.* How does the structure of reasoning required by a task affect generalization? Do skills learned from highly structured domains (e.g., mathematics, code generation) transfer to less-structured domains (e.g., medical or legal reasoning), and vice versa?
- ***(RQ3)*** *Intra-domain generalization.* How effectively do RPT gains generalize across subdomains within the same domain?

To address these research questions, we design a two-stage pipeline. First, we perform an observational study by evaluating 14 RPT models, each compared against its corresponding base model across a diverse set of benchmarks, spanning their seen and unseen domains. Because existing RPT models are typically trained with different configurations (e.g., different RL algorithms and hyperparameters) on multi-domain data, it is challenging to isolate the effect of RPT itself from the advantages brought by specific configuration or data.

To mitigate confounding factors, we further conduct an interventional study, where we fine-tunedd three RPT models from the same base model with the same configuration, each on a disjoint single-domain dataset. We then evaluate these trained models using the same benchmarks as in the observational analysis. In the rest of this section, we describe the evaluation settings and experimental setup for both studies.

Benchmarks. For evaluation, we use 16 popular benchmarks, covering a wide range of domains and difficulty levels. We categorize these benchmarks into the following three representative domains:

- *Math*: For easier questions, we use GSM8K [15] and MATH-500 [39], while for more challenging problems, we select AIME 2024 [30] and AMC 2023 [73].
- *Code*: We use easy coding problems, including MBPP [7] and HumanEval [11], and relatively challenging problems, including BigCodeBench [93], LiveCodeBench [29], USACO [64], and Codeforces [46]. To test programming language generalization, we also include the multi-language benchmark Polyglot [4].
- *General reasoning*: We use high-quality benchmarks that are not mathematics nor programming problems for general reasoning, including PubMedQA [33] and MedQA [23] for medical reasoning, TabFact [13] for fact verification, LegalBench [24] for legal reasoning, and FinBench [90] for financial problem solving.

Table 1: Selected RFT models for observational analysis. The `RFT Domain(s)` refers to the domain(s) covered in the RFT training data.

***(Model ID)* Reasoning Model**	**Base Model**	**RPT Domain(s)**
(1) DeepScaleR-1.5B-Preview [43]	DeepSeek-R1-Distill-Qwen-1.5B [18]	Math
(2) DeepCoder-1.5B-Preview [42]	DeepSeek-R1-Distill-Qwen-1.5B [18]	Code
(3) Skywork-o1-Open-Llama-3.1-8B [51]	Llama-3.1-8B-Instruct [47]	Code, Math
(4) Eurus-2-7B-PRIME [16, 87]	Qwen2.5-Math-7B [84]	Code, Math
(5) Absolute_Zero_Reasoner-Coder-3b [91]	Qwen2.5-Coder-3B [28, 83]	Code
(6) Absolute_Zero_Reasoner-Coder-7b [91]	Qwen2.5-Coder-7B [28, 83]	Code
(7) ZR1-1.5B [94]	DeepSeek-R1-Distill-Qwen-1.5B [18]	Code, Math
(8) Llama-3.1-Nemotron-Nano-8B-v1 [9]	Llama-3.1-8B-Instruct [47]	Instruction Following
(9) Thespis-Llama-3.1-8B [41]	Meta-Llama-3.1-8B-Instruct-abliterated [49]	Chat
(10) STILL-3-1.5B-preview [31, 48, 71]	DeepSeek-R1-Distill-Qwen-1.5B [18]	Math
(11) Arcee-Maestro-7B-Preview [3]	DeepSeek-R1-Distill-Qwen-7B [18]	Code, Math
(12) Fino1-8B [56]	Llama-3.1-8B-Instruct [47]	Finance
(13) OREAL-7B [44]	OREAL-7B-SFT [44]	Math
(14) Open-RS3 [17]	DeepSeek-R1-Distill-Qwen-1.5B [18]	Math

Evaluation Configurations. For all benchmarks, we use a consistent sets of generation hyperparameters across all models. The maximum response length is set to $\min\{16192,\, C\}$, where C denotes the model's context window. For each model, we run the small benchmarks (i.e., AMC 2023 and AIME 2024) 16 times, while executing all other benchmarks once.

For prompting, we apply each model's default chat template and system prompt. For model pairs whose base models are pretrained-only (i.e., not instruction-tuned), we evaluate under two prompting strategies: (1) the default prompt, and (2) the official CoT+few-shot template used for evaluating Qwen2.5-Math [84], using $\min\{4,\, N\}$ shots, where N is the maximum number of few-shot exemplars that can fit within the model's context window. For each model (the reasoning model and its corresponding base model), we report the higher score across the two prompting strategies, ensuring that both models are evaluated under their most favorable prompting conditions. We evaluate each model on each benchmark on a single 24 GB RTX A5000.

Metrics. To assess whether RPT improves the accuracy performance within or across domains, we report the aggregated accuracy improvement $\Delta_{i,j}^{(\mathcal{D})}$ of an RPT model i over its base model j for a given domain $\mathcal{D}$:

$$\Delta_{i,j}^{(\mathcal{D})} = \frac{\sum_{t\in\mathcal{D}} N_t R_t (A_{i,t} - A_{j,t})}{\sum_{t\in\mathcal{D}} N_t R_t},$$

where N_t is the number of problems in t, R_t is the number of repetitions we executed for t, $A_{i,t}$ is the accuracy of model i on benchmark t, and $A_{j,t}$ is the accuracy of model j on benchmark t.

In addition, to ensure statistical significance in our findings, we applied the Cochran–Mantel–Haenszel (CMH) test [1], a statistical test for analyzing stratified categorical data. We treat each benchmark as an independent stratum—that is, a random sample of distinct downstream tasks. Given an RPT model i, a base model j, and a domain $\mathcal{D}$ of benchmarks, we calculate the common odds ratio estimate ($\theta_{i,j,\mathcal{D}}$) that estimates the correlation between the RPT process and the accuracy improvement on $\mathcal{D}$:

$$\hat{\theta}_{i,j}^{(\mathcal{D})} = \frac{\sum_{t\in\mathcal{D}} N_t R_t A_{i,t}(1 - A_{j,t})}{\sum_{t\in\mathcal{D}} N_t R_t A_{j,t}(1 - A_{i,t})}$$

An odds ratio greater than 1 indicates improvement due to RPT; a value less than 1 indicates a decrease in accuracy due to RPT. We evaluate the statistical significance under the null hypothesis $H_0 : \theta_{i,j}^{(\mathcal{D})} = 1$ against the alternative hypothesis $H_1 : \theta_{i,j}^{(\mathcal{D})} \neq 1$, using the standard CHM test statistics,

$$\xi = \frac{\left(\sum_{t\in\mathcal{D}} N_t R_t (A_{i,t} - A_{j,t})\right)^2}{\sum_{t\in\mathcal{D}} N_t^2 R_t^2 A_{i,t} A_{j,t} (1 - A_{i,t})(1 - A_{j,t})(2N_t - 1)^{-1}}$$

which follows a chi-squared distribution asymptotically with 1 degree of freedom.

Observational Study. To ensure a comprehensive and representative evaluation of RPT model generalizability, we adopt a systematic approach to selecting models for our observational study:

- *Stage 1.* We collect the 466 models from Hugging Face applying the following filtering criteria: as of April 23rd, 2025: (1) the model supports `Text Generation` tasks, (2) its model card description contains the keyword `reasoning`, `chain-of-thought`, and/or `chain of thought` and (3) the model has received at least 10 likes.
- *Stage 2.* We use `o4-mini` [53] to prefilter models potentially trained with RPT, based on their model card descriptions. This automatic filtering is followed by manual verification, resulting in 31 models that we confirm to be RPT models.
- *Stage 3.* From the 31 RPT models, we manually select 12 that meet the following criteria: (1) the RPT datasets are publicly disclosed, (2) the model sizes range from 1.5B to 8B parameters, and (3) the base models are not purely pretrained models, ensuring they can generate coherent responses and follow basic instructions for evaluating reasoning capabilities.

Additionally, we include 2 recently released models, `Absolute_Zero_Reasoner-Coder-3B` and `Absolute_Zero_Reasoner-Coder-7B` [91], both fine-tunedd with limited RPT data and released on May 6th, 2025. These models demonstrate strong performance on math and code reasoning tasks and serve as representative cases for examining RPT generalizability.

We finalize our selection of 14 RPT models, with the details, including base models and RPT domains, presented in Table 1. For each RPT model and its corresponding base model, we compare performance across 16 benchmarks.

Interventional Analysis. To isolate the effect of RPT from other training configurations, including datasets, algorithms, and hyperparameters, we trained three RPT models based on `DeepSeek-R1-Distill-Qwen-1.5B` [18] on three disjoint datasets—math, code, and general reasoning—respectively. We curated these datasets based on existing datasets that leads to performant RPT models. Specifically,

- *Math*: we uniformly sampled 40,000 problems from a combination of the math split of Eurus-2-RL [16], which originates from the NuminaMath-CoT dataset [36].
- *Code*: we uniformly sampled 40,000 deduplicated problems from a combination of KodCode [82], DeepCoder-Preview [42], Apps [26], TACO [37], and the code split of Eurus-2-RL [16].
- *General Reasoning*: we selected 40,000 high-quality, non-math, and non-code data from the multi-subject RLVR dataset [67]. To achieve that, we applied o3-mini [53] to exclude math-related, code-related, or fact-recall questions.

To ensure fair evaluation, we clean this data to ensure that it does not overlap with our evaluation set.

We applied consistent settings for all three RPT training processes. In terms of the RL algorithm, we applied Group Relative Policy Optimization (GRPO) with the same setting as DeepCoder [42]. In terms of hyperparameters, we trained each of the dataset for one epoch with a batch size of 64 and a context length of 8,192. To stabilize the training process, we used a learning rate of 10^{-6} and an entropy coefficient of 0. We fine-tuned the models on 8 80GB H100 GPUs.

4 Findings

In this section, we present the findings of our study based on results from observational and interventional studies. We summarize our findings as follows:

- ***(RQ1)*** RPT does not exhibit generalizability in arbitrary unseen domains (Section 4.1).
- ***(RQ2)*** RPT demonstrates cross-domain generalizability when reasoning patterns are similar, such as mutual transfer between math and code, but fails to generalize across distinct reasoning patterns, such as from math or code to general reasoning (Section 4.2).
- ***(RQ3)*** Intra-domain generalizability of RPT strongly depends on the structural similarity between subdomain tasks (Section 4.3).

4.1 RPT Gains Do Not Generalize to Arbitrary Unseen Domains

Existing RPT models fail to transfer beyond their training domains. We begin by analyzing our observational study, which evaluates a diverse set of existing RPT models using multi-domain

Table 2: Existing Open-sourced RPT models achieve significantly larger accuracy gains Δ (%) and odds ratios $\hat{\theta}$ on in-domain (*ID*) tasks compared to out-of-domain (*OOD*) tasks. An asterisk (*) denotes statistical significance at $p < 0.05$.

Metric	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(7)*	*(8)*	*(9)*	*(10)*	*(11)*	*(12)*	*(13)*	*(14)*	**Avg.**
$\Delta^{(ID)} \uparrow$	5.40	4.61	6.96	30.03	-6.27	30.12	2.82	7.07	-26.01	4.31	-4.27	-3.84	-0.41	-0.59	**3.57**
$\Delta^{(OOD)} \uparrow$	1.67	4.01	-2.64	46.32	2.55	-23.31	-5.47	13.22	-4.73	-1.33	-7.39	-27.89	-8.44	-0.34	**-1.48**
$\Delta^{(ID)} - \Delta^{(OOD)}$	3.73	0.60	33.37	-16.29	-8.82	53.43	8.30	-6.15	-2.13	5.64	3.12	24.05	-8.85	-0.25	**5.04**
$\hat{\theta}^{(ID)} \uparrow$	1.36*	1.45*	1.59*	7.13*	0.52*	22.47*	1.22*	1.34*	0.34*	1.28*	0.72*	0.83*	0.97	0.97	**3.01**
$\hat{\theta}^{(OOD)} \uparrow$	1.07*	1.18*	0.31*	14.09*	1.15*	0.41*	0.80*	1.98*	0.68*	0.95*	0.69*	0.30*	1.47*	0.99	**1.86**
$\hat{\theta}^{(ID)}/\hat{\theta}^{(OOD)}$	1.27	1.22	5.15	0.51	0.45	54.61	1.53	0.68	0.50	1.35	1.03	2.74	0.66	0.98	**5.19**

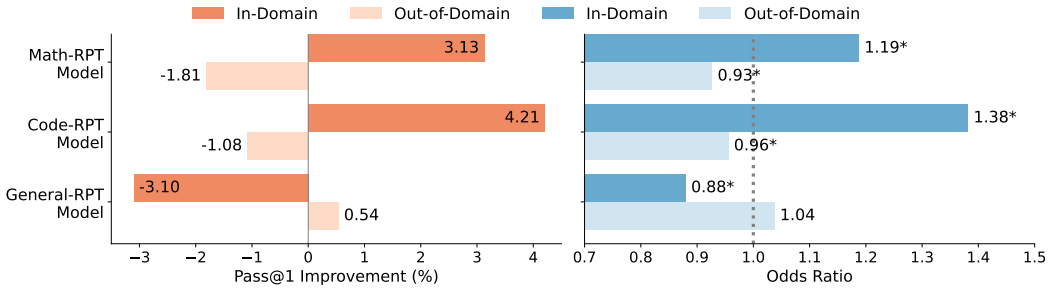


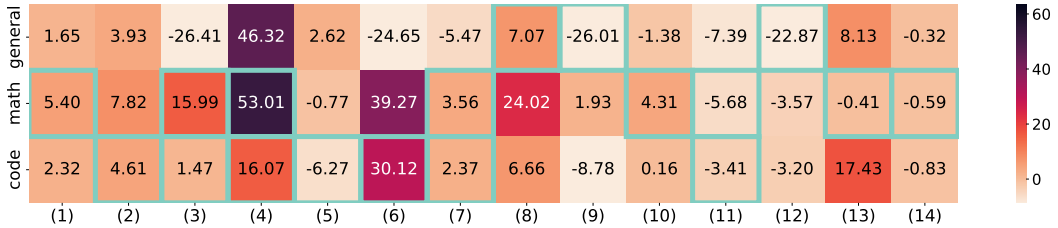
Figure 2: RPT models on single domains show significant pass@1 improvements over base models and higher odds ratios on in-domain tasks, but not on out-of-domain tasks. No single-domain model achieves statistically significant gains in out-of-domain tasks (* denotes $p < 0.05$).

tasks. Specifically, we compare the performance improvements of each model in tasks from the same domain as their training data (ID), and tasks that are out-of-domain with their training data (OOD). For instance, *(1)* `DeepScaleR-1.5B-Preview` is trained exclusively on math-related data. Therefore, ID tasks for this model include GSM8K, MATH500, AIME 2024, and AMC 2023, while all other tasks (e.g., legal, medical, coding) are OOD.

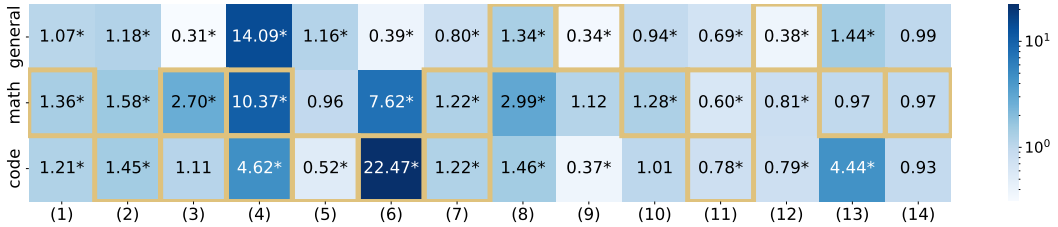
We present the results in Table 2. Across the table, RPT models exhibit considerably higher improvements on *ID* tasks compared to *OOD* tasks, with a 3.57% increase in pass@1 for ID tasks, but a 1.48% decrease for OOD tasks. For example, *(1)* DeepScaleR-1.5B-Preview shows a 5.1% gain in pass@1 on math domain tasks, but only 1.7% in others, representing a 3× drop. This lack of generalizability stems from the RPT algorithm itself rather than simply from overfitting to large-scale training data: notably, a similar trend is observed in *(6)* `Absolute_Zero_Reasoner-Coder-7B`, which was fine-tunedd on a near-zero amount of data. Despite its minimal training data exposure, this model experiences a 23.31% decrease in pass@1 accuracy on unseen domains, while achieving a 30.12% improvement within its RPT domain. We also observe that the generalizability of RPT algorithms is sensitive to the training data, implementation details, and finetuning strategy. For example, although all trained on math data, *(1)* `DeepScaleR-1.5B-Preview` demonstrates improvements in both ID and OOD tasks, whereas *(7)* `ZR1-1.5B` and *(10)* `STILL-3-1.5B-Preview` show statistically significant performance gains in ID tasks as well as statistically significant performance drops on OOD tasks. These findings suggest that the gains from RPT are largely domain-specific: models significantly improve on tasks similar to their training data, but fail to generalize robustly to other unseen domains.

Single-domain finetuning reinforces evidence of RPT's limited generalizability. To further dissect the generalizability limitations identified in our observational study, we conduct a more controlled investigation by isolating models fine-tuned exclusively on single domains. To do so, we analyze our interventional study results, where ID corresponds to the training domain, while OOD include all tasks from the remaining two domains in our evaluation.

As shown in Figure 2, none of the models fine-tunedd on a single domain exhibit statistically significant improvement on OOD tasks. Both the Math-RPT model and the Code-RPT model show performance drops on OOD tasks with statistical significance, in contrast to the statistically

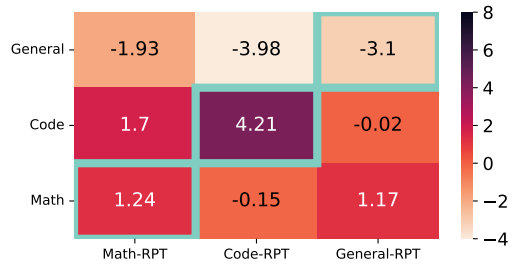


(a) Pass@1 improvement $\Delta^{(\mathcal{D})}$ in percentage across domains.

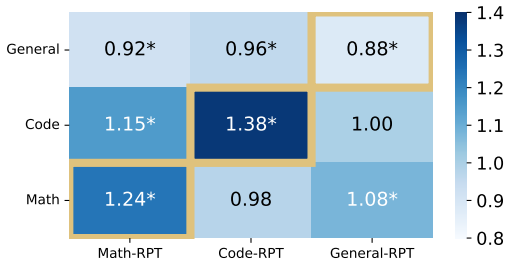


(b) Odds ratio $\hat{\theta}^{(\mathcal{D})}$ across domains. An asterisk $(*)$ denotes statistical significance at $p < 0.05$.

Figure 3: Multi-domain evaluation results of existing RPT models. We highlight in-domain results with frames. RPT shows mutual generalizability between math and code, one-way transfer from general reasoning to math and code, but no generalization from math or code to general reasoning.



(a) Pass@1 improvement $\Delta^{(\mathcal{D})}$ (%) across domains.



(b) Odds ratio $\hat{\theta}^{(\mathcal{D})}$ across domains.

Figure 4: Multi-domain evaluation results of RPT models on single domains. An asterisk $(*)$ denotes statistical significance at $p < 0.05$. We highlight in-domain results with frames. RPT demonstrates generalizability from math to code and from general reasoning to math, but shows no generalizability from math or code to general reasoning.

significant gains they achieve in-domain. The General-RPT model also demonstrates no statistically significant gains on its OOD tasks.

4.2 RPT Gains Generalize Across Domains with Similar Reasoning Patterns

Structured-to-structured generalization is effective. We observe that models fine-tuned on math and code data exhibit strong mutual generalizability. In our observational analysis (Figure 3), models fine-tuned exclusively on math or code demonstrate transferable performance gains across these two domains. For example, models fine-tuned on math domain data achieve an average improvement of 2.18% in pass@1 on math domain tasks and 4.77% on code domain tasks. Similarly, models fine-tuned on code domain data improve by 9.49% in pass@1 on code domain tasks and 15.44% on math domain tasks. In both cases, the improvement is even greater on the non-finetuned domain, suggesting that math and code tasks share common structured reasoning patterns that enable RPT to generalize effectively across these domains.

Structured reasoning patterns that are more foundational tend to exhibit stronger cross-domain transfer. Building on the findings from our observational study, we further examine the generalizability across structured reasoning domains using interventional study results from models fine-tuned on single domains (Figure 4). We observe that the generalizability from math to code is notably stronger and more consistent than the reverse. This aligns with the intuition that mathematical reasoning is a more fundamental form of structured thinking, serving as the backbone for coding tasks, and thus enables better cross-domain transfer when used as RPT data.

Structured-to-unstructured generalization is limited. Models trained on structured reasoning domains—such as math and code—exhibit substantially reduced improvements when evaluated on general-domain tasks. In our observational study (Figure 3), models fine-tuned on structured reasoning domains (i.e., math, code, or both) achieve an average improvement of -0.27% in pass@1 on general-domain tasks, compared to significant gains of 11.08% on math and 5.82% on code tasks. While the improvements in math and code domain tasks are statistically significant, the performance drops on general reasoning tasks, indicating a lack of generalizability to unstructured domains. For instance, in the *(1)* `DeepScaleR-1.5B-Preview` and *(2)* `DeepCoder-1.5B-Preview` pair, the observed gains in math and code domain tasks are both significantly higher than those in general-domain tasks. Our interventional study results further confirm this trend (Figure 4): while the Math-RPT model shows improvements in both math and code domain tasks, its performance drops notably on general-domain tasks. Similarly, the Code-RPT model shows a statistically significant drop in performance on general reasoning tasks. These results suggest that although structured reasoning skills generalize well across similarly structured domains, they fail to transfer effectively to domains that require less structured, more heterogeneous reasoning patterns.

Unstructured-to-structured generalization is promising. RPT models trained on unstructured general-domain data still exhibit measurable gains in structured tasks. In our observational study (Figure 3), general-domain RPT models show substantially higher pass@1 improvements on math (21.40%) and code (12.16%) tasks compared to tasks within the general reasoning domain. Similarly, in our interventional analysis (Figure 4), the General-RPT model achieves statistically significant gains on math domain tasks and shows no noticeable degradation on code domain tasks, while underperforming on tasks within its own domain. This suggests that unstructured reasoning patterns encompass broader representational complexity and implicitly subsume the essential components of structured reasoning, functioning as a conceptual superset.

4.3 Intra-domain RPT Gains Depend on Structural Similarity among Subdomains

Structured reasoning patterns generalize effectively within domain. Consistent with prior work on math and code reasoning, our observational study shows that models fine-tunedd on structured domains generalize well across tasks within the same domain (Figure 3). On average, models trained on math data achieve a pass@1 improvement of 2.18% on math tasks, while models trained on code data show an average improvement of 9.49% on code tasks. Our interventional analysis further confirms this trend where structured-domain models (i.e., the Code-RPT model and the Math-RPT model) exhibit the largest gains on tasks from their corresponding training domain (Figure 4). These results suggest that data following consistent and structured reasoning templates facilitates reliable generalization within the same domain, as downstream tasks can leverage similar inductive patterns.

Unstructured reasoning patterns lack intra-domain consistency. In contrast, models trained on general-domain (unstructured) data demonstrate limited or negative transfer to other unstructured tasks from different domains in our observational study (Figure 3). For instance, `Fino1-8B` (model *(12)*), fine-tunedd on financial data, exhibits notable performance drops when evaluated on all unrelated general-domain tasks. Its pass@1 on `PubMedQA` (medical domain) declines from 3.26% to 1.28%, on `LegalBench` (legal domain) from 6.42% to 4.84%, and on `TabFact` declines from 64.18% to 48.39% (general tabular knowledge). Our interventional results reinforce this observation: the General-RPT model underperforms the base model on general-domain tasks, with the degradation in accuracy being statistically significant (Figure 4). This suggests that, unlike structured domains, unstructured reasoning tasks are highly diverse and domain-specific. They lack a shared logical template, making it difficult for RPT to generalize even within what is nominally the same domain.

5 Related Work

In this section, we review RPT from the following four aspects.

RPT algorithms. The dominant approach for RPT is Proximal Policy Optimization (PPO) [58, 59], which builds on classical policy-gradient methods such as REINFORCE [76] and Trust Region Policy Optimization (TRPO) [58] and has been widely used in the post-training of OpenAI's ChatGPT [52], Google's Gemini [68], and Anthropic's Claude [5]. However, PPO is an actor-critic-based algorithm, which requires an additional critic network that is typically also initialized from the same pretrained LLM. This introduces both computational overhead and algorithmic complexity compared to the vanilla REINFORCE algorithm [77]. In recognition of this, recent work has attempted to design simplified alternative approaches to PPO for LLM post-training. A line of work studies arguably simplest rejection sampling fine-tuning [20, 72], which iteratively generates multiple completions per prompt, filters out low-quality responses, and fine-tunes on the selected outputs. Another direction revisits Reinforce-style methods, such as GRPO [61], DAPO [86], VAPO [89], ReMax [38], RLOO [2, 34], Reinforce++ [27], and Reinforce-rej [80]. These algorithms discard the critic network and instead rely on Monte Carlo estimates from on-policy samples to update the policy. These methods primarily differ in how they estimate the advantage function. Among them, GRPO has garnered particular attention for its strong empirical performance in the post-training of DeepSeek-R1 [60].

RPT frameworks. RPT was first widely applied under the reinforcement learning from human feedback (RLHF) framework in the context of LLM post-training [8, 55]. In RLHF, a proxy reward is learned from human-annotated preference dataset and Bradley-Terry model [10], and the models are trained to optimize a KL-regularized reward objective to prevent overfitting the imperfect learned reward. RLHF has since become a standard technique in LLM training pipelines [5, 52, 68, 72]. More recently, RPT with verifiable reward has received significant attention in building powerful reasoning models, following the release of OpenAI o1 [54] and DeepSeek-R1 [18]. In this framework, a verifier is employed to check the correctness of the final answer for reasoning-related tasks, and serves as the reward signal for the RPT training. The major advantage is that this verifier score is much more reliable than the learned reward in RLHF, thus enabling large-scale RPT training in the post-training stage. This has also led to the development of increasingly capable and open-source RPT frameworks [42, 43, 63].

Limitations of RPT. Despite the recent successes of RPT in improving language model reasoning capabilities, the limitations of RPT in general have been widely studied. At the training phase, SFT with LLM reasoning traces, without RL, has been shown to be effective enough [70, 88]. At the inference phase, the quality of reasoning models depends crucially on their ability to scale under test-time compute constraints [50, 92]. Moreover, the effectiveness of LLMs' lengthy "thinking" processes has also been challenged: recent findings suggest that bypassing explicit multi-step reasoning through simpler prompting or parallelized sampling can achieve comparable or even superior results [45, 85]. Building on these observations, our work aims to examine the limitations of RPT at a finer granularity, specifically its data generalizability across the training and inference phases.

Applications of RPT across tasks. RPT has proven effective for a broad range of tasks with well-defined correctness, where LLMs are finetuned using reward feedback. These range from structured tasks such as math [16, 81] and coding [35, 62, 65, 75], to unstructured tasks like search engine use [32] and open-ended question answering [66]. However, all these models are trained and evaluated on tasks within a single domain or task type. Even works on general knowledge [66] remain confined to open-ended question answering tasks, without testing transfer across fundamentally different task types. In contrast, our work directly addresses this cross-domain generalization gap.

6 Conclusion

In this paper, we identify important limitations in the generalizability of RPT across domains. Through both observational and interventional studies, we consistently find that while RPT produces substantial improvements within training domains, its generalization to unseen domains is limited. In particular, while there is evidence of cross-domain transfer between structured domains like math and code, there is little evidence of transfer to unstructured domains. Our work emphasizes the need for a more nuanced understanding of cross-domain knowledge transfer in LLMs.

References

[1] Alan Agresti. *Categorical data analysis*. John Wiley & Sons, 2013.

[2] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

[3] Arcee AI. Arcee maestro-7b-preview. `https://huggingface.co/arcee-ai/Arcee-Maestro-7B-Preview`, 2024. Accessed: 2025-05-12.

[4] Aider-AI. Polyglot benchmark, 2025. URL `https://github.com/Aider-AI/polyglot-benchmark?tab=readme-ov-file`.

[5] Anthropic. Introducing claude. 2023. URL `https://www.anthropic.com/index/introducing-claude`.

[6] Anthropic. Claude 3.7 sonnet. `https://www.anthropic.com/claude/sonnet`, February 2025.

[7] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL `https://arxiv.org/abs/2108.07732`.

[8] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022. URL `https://arxiv.org/abs/2204.05862`.

[9] Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen, Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, Eric Chung, and Chris Alexiuk. Llama-nemotron: Efficient reasoning models, 2025. URL `https://arxiv.org/abs/2505.00949`.

[10] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[11] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. 2021.

[12] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 699–708, 2020.

[13] Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. Tabfact: A large-scale dataset for table-based fact verification. *arXiv preprint arXiv:1909.02164*, 2019.

[14] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

[15] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[16] Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards, 2025. URL `https://arxiv.org/abs/2502.01456`.

[17] Quy-Anh Dang and Chris Ngo. Reinforcement learning for reasoning in small llms: What works and what doesn't, 2025. URL `https://arxiv.org/abs/2503.16219`.

[18] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He,

Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[19] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

[20] Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment, 2023. URL https://arxiv.org/abs/2304.06767.

[21] Google DeepMind. Gemini 2.5 pro. https://deepmind.google/technologies/gemini/pro/, 2025.

[22] IBM Granite Embedding Team. Granite embedding models, 2024. URL https://github.com/ibm-granite/granite-embedding-models/.

[23] Maxime Griot, Coralie Hemptinne, Jean Vanderdonckt, and Demet Yuksel. Large Language Models lack essential metacognition for reliable medical reasoning. *Nature Communications*, 16(1):642, January 2025. ISSN 2041-1723. doi: 10.1038/s41467-024-55628-6. URL https://doi.org/10.1038/s41467-024-55628-6.

[24] Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Ré, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel N. Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, Jessica Wu, Joe Nudell, Joel Niklaus, John Nay, Jonathan H. Choi, Kevin Tobia, Margaret Hagan, Megan Ma, Michael Livermore, Nikon Rasumov-Rahe, Nils Holzenberger, Noam Kolt, Peter Henderson, Sean Rehaag, Sharad Goel, Shang Gao, Spencer Williams, Sunny Gandhi, Tom Zur, Varun Iyer, and Zehua Li. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models, 2023.

[25] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025.

[26] Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with apps. *NeurIPS*, 2021.

[27] Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

[28] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

[29] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code, 2024. URL https://arxiv.org/abs/2403.07974.

[30] Ming-Hui Jia. Aime 2024, 2024. URL https://huggingface.co/datasets/Maxwell-Jia/AIME_2024.

[31] Jinhao Jiang, Zhipeng Chen, Yingqian Min, Jie Chen, Xiaoxue Cheng, Jiapeng Wang, Yiru Tang, Haoxiang Sun, Jia Deng, Wayne Xin Zhao, Zheng Liu, Dong Yan, Jian Xie, Zhongyuan Wang, and Ji-Rong Wen. Enhancing llm reasoning with reward-guided tree search. *arXiv preprint arXiv:2411.11694*, 2024.

[32] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

[33] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*, 2019.

[34] Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! 2019.

[35] Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven C. H. Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning, 2022. URL https://arxiv.org/abs/2207.01780.

[36] Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-CoT](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf), 2024.

[37] Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. Taco: Topics in algorithmic code generation dataset. *arXiv preprint arXiv:2312.14852*, 2023.

[38] Ziniu Li, Tian Xu, Yushun Zhang, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv e-prints*, pages arXiv–2310, 2023.

[39] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

[40] 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. *arXiv preprint arXiv:2503.20783*, 2025.

[41] Locutusque. Thespis-llama-3.1-8b. https://huggingface.co/Locutusque/Thespis-Llama-3.1-8B, 2024.

[42] Michael Luo, Sijun Tan, Roy Huang, Xiaoxiang Shi, Rachel Xin, Colin Cai, Ameen Patel, Alpay Ariyak, Qingyang Wu, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. https://pretty-radio-b75.notion.site/DeepCoder-A-Fully-Open-Source-14B-Coder-at-O3-mini-Level-1cf81902c14680b3bee5eb349a512a51, 2025. Notion Blog.

[43] Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013
2025. Notion Blog.

[44] Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, Haian Huang, et al. Exploring the limit of outcome reward for learning mathematical reasoning. *arXiv preprint arXiv:2502.06781*, 2025.

[45] Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking, 2025. URL https://arxiv.org/abs/2504.09858.

[46] MatrixStudio. Codeforces python submissions, 2024. URL https://huggingface.co/datasets/MatrixStudio/Codeforces-Python-Submissions.

[47] Meta AI. Meta llama 3.1 8b instruct. https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct, 2024. Release date: July 23, 2024.

[48] Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413*, 2024.

[49] mlabonne. Meta-llama-3.1-8b-instruct-abliterated. https://huggingface.co/mlabonne/Meta-Llama-3.1-8B-Instruct-abliterated, 2024.

[50] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

[51] Skywork o1 Team. Skywork-o1 open series. https://huggingface.co/Skywork, November 2024. URL https://huggingface.co/Skywork.

[52] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023.

[53] OpenAI. Openai o3 and o4-mini system card, 2025. URL https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf.

[54] OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor

Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

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

[56] Lingfei Qian, Weipeng Zhou, Yan Wang, Xueqing Peng, Jimin Huang, and Qianqian Xie. Fino1: On the transferability of reasoning enhanced llms to finance. *arXiv preprint arXiv:2502.08127*, 2025.

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

[58] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017. URL https://arxiv.org/abs/1502.05477.

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

[60] 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.

[61] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Y Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[62] Wei Shen and Chuheng Zhang. Policy filtration in rlhf to fine-tune llm for code generation, 2024. URL https://arxiv.org/abs/2409.06957.

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

[64] Quan Shi, Michael Tang, Karthik Narasimhan, and Shunyu Yao. Can language models solve olympiad programming? *arXiv preprint arXiv:2404.10952*, 2024.

[65] Parshin Shojaee, Aneesh Jain, Sindhu Tipirneni, and Chandan K. Reddy. Execution-based code generation using deep reinforcement learning, 2023. URL https://arxiv.org/abs/2301.13816.

[66] Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. Crossing the reward bridge: Expanding rl with verifiable rewards across diverse domains, 2025. URL https://arxiv.org/abs/2503.23829.

[67] Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. Expanding rl with verifiable rewards across diverse domains. *arXiv preprint arXiv:2503.23829*, 2025.

[68] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[69] NovaSky Team. Unlocking the potential of reinforcement learning in improving reasoning models. https://novasky-ai.github.io/posts/sky-t1-7b, 2025. Accessed: 2025-02-13.

[70] Open Thoughts Team. Open Thoughts, January 2025.

[71] RUCAIBox STILL Team. Still-3-1.5b-preview: Enhancing slow thinking abilities of small models through reinforcement learning. 2025. URL https://github.com/RUCAIBox/Slow_Thinking_with_LLMs.

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

[73] Lewis Tunstall and Li Jia. Amc 2023, 2024. URL https://huggingface.co/datasets/AI-MO/aimo-validation-amc.

[74] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022. URL https://arxiv.org/abs/2206.07682.

[75] Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I. Wang. Swe-rl: Advancing llm reasoning via reinforcement learning on open software evolution, 2025. URL https://arxiv.org/abs/2502.18449.

[76] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

[77] Ronald J Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

[78] xAI. Grok 3 beta — the age of reasoning agents. https://x.ai/news/grok-3, February 2025.

[79] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.

[80] Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

[81] Wei Xiong, Hanning Zhang, Chenlu Ye, Lichang Chen, Nan Jiang, and Tong Zhang. Self-rewarding correction for mathematical reasoning, 2025. URL https://arxiv.org/abs/2502.19613.

[82] Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. *arXiv preprint arXiv:2503.02951*, 2025.

[83] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[84] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[85] Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning, 2025. URL https://arxiv.org/abs/2502.03387.

[86] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, 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, Weinan Dai, 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.

[87] Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

[88] 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.

[89] Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks, 2025. URL https://arxiv.org/abs/2504.05118.

[90] Zhihan Zhang, Yixin Cao, and Lizi Liao. Finbench: Benchmarking llms in complex financial problem solving and reasoning.

[91] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data, 2025. URL https://arxiv.org/abs/2505.03335.

[92] Eric Zhao, Pranjal Awasthi, and Sreenivas Gollapudi. Sample, scrutinize and scale: Effective inference-time search by scaling verification, 2025. URL https://arxiv.org/abs/2502.01839.

[93] Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, Thong Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, Prateek Yadav, Naman Jain, Alex Gu, Zhoujun Cheng, Jiawei Liu, Qian Liu, Zijian Wang, Binyuan Hui, Niklas Muennighoff, David Lo, Daniel Fried, Xiaoning Du, Harm de Vries, and Leandro Von Werra. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions, 2025. URL https://arxiv.org/abs/2406.15877.

[94] Zyphra. Zyphra zr1-1.5b. https://huggingface.co/Zyphra/ZR1-1.5B, 2024. Accessed: 2025-05-12.

A Limitations

Our work has several important limitations:

- **Use of RL algorithms:** The RPT techniques we experiment with (e.g., PPO, reward-modulated finetuning) may differ from those used in frontier models such as o3 [53] and Gemini [21]. As such, our findings may not directly apply to LLMs with proprietary optimization strategies.
- **Scale limitations:** Emergent behaviors often arise at larger model or dataset scales [74]. While our experiments are conducted on widely used open-weight models, it is possible that scaling effects in models with more parameters or trained on larger datasets could lead to different generalization dynamics or failure modes.

- **Synthetic data design:** Our use of structured or domain-specific data for reward-based finetuning is necessarily limited in scope. Other forms of synthetic data, such as instruction-augmented samples [14], adversarial examples [12], or data curated through human-AI collaboration [19], may lead to improved generalization that we do not capture in this study.

B Broader Impacts

This paper encourages the development of more robust evaluation suites and transparent training paradigms for reasoning-capable language models. By analyzing how reasoning finetuning (RPT) affects model generalization across structured and unstructured domains, our work contributes to understanding the risks of narrow overfitting and domain brittleness.

We also highlight the importance of open-weight baselines and reproducible experimental pipelines, which are essential for aligning model behavior with desirable generalization properties. However, as with any finetuning method, RPT may unintentionally amplify domain-specific biases or incentivize brittle heuristics if not evaluated rigorously. We urge future work to incorporate fairness, calibration, and robustness criteria into such assessments.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Sections 3 and 4

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix A

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Sections 3 and 4

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will open-source our code once on acceptance.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 3

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Section 4

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 3

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: All sections.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix B

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA]

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: [NA]

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: [NA]

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA]

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We used o3-mini and o4-mini to select models and filter data. We illustrated the process in detail in Section 3.