OPENRFT: ADAPTING REASONING FOUNDATION MODEL FOR DOMAIN-SPECIFIC TASKS WITH REINFORCEMENT FINE-TUNING

Yuxiang Zhang, Yuqi Yang, Jiangming Shu, Yuhang Wang, Jinlin Xiao & Jitao Sang[*]
School of Computer Science and Technology
Beijing Jiaotong University
Beijing, China

yuxiangzhang, yqyang, jiangmingshu, yhangwang, jinlinx,
jtsang@bjtu.edu.cn

ABSTRACT

OpenAI's recent introduction of Reinforcement Fine-Tuning (RFT) showcases the potential of reasoning foundation model and offers a new paradigm for fine-tuning beyond simple pattern imitation. This technical report presents *OpenRFT*, our attempt to fine-tune generalist reasoning models for domain-specific tasks under the same settings as RFT. OpenRFT addresses two key challenges of lacking reasoning step data and the limited quantity of training samples, by leveraging the domain-specific samples in three ways: question augmentation, synthesizing reasoning-process data, and few-shot ICL. The evaluation is conducted on SciKnowEval, where OpenRFT achieves significant performance gains with only 100 domain-specific samples for each task. More experimental results will be updated continuously in later versions. Source codes, datasets, and models are disclosed at: https://github.com/ADaM-BJTU/OpenRFT.

1 INTRODUCTION

OpenAI's o1 model has shown strong reasoning abilities in mathematics and programming, but its generalization to other tasks remains uncertain. The recent introduction of Reinforcement Fine-Tuning (RFT) (OpenAI, 2024) has provided a promising avenue for reasoning generalization. With only dozens of high-quality $(question, answer)$ pairs, RFT enables the creation of customized reasoning models excelling at domain-specific tasks.

The significance of RFT is at least two-fold: (1) It demonstrates the promise of using generalist reasoning models, like o1, as reasoning foundation models. By enabling the efficient creation of domain-specific reasoning models, RFT practically expands the applicability of reasoning models across diverse tasks. (2) It introduces a new paradigm for fine-tuning foundation models. Unlike Supervised Fine-Tuning (SFT), which merely mimics patterns in training data, RFT leverages reasoning capabilities to facilitate thinking and trial-and-error learning. This brings models closer to achieving human-like generalization, moving beyond mechanical imitation to extrapolate knowledge to new cases.

It is believed that the core techniques behind RFT are closely related to those of o1. Inspired by recent o1-replication efforts (ope, 2024; Team, 2024a; SimpleBerry, 2024; Zhao et al., 2024; Zhang et al., 2024), we attempt to develop an implementation under the same settings as the RFT demo, which we call *OpenRFT*. While this early exploration may not achieve optimal results, we hope it is beneficial to the community for clarifying the conceptual landscape and inspiring further advancements in this area.

[*] Corresponding author.

Realizing RFT requires addressing two key challenges: the absence of reasoning step data in the provided domain-specific samples, and the limited quantity of such samples. For the first challenge, lacking reasoning process supervision may lead to rollout data where the final outcome is correct, but the reasoning steps are flawed (example illustrated in Fig. 4 in *Appendix*). This will introduce incorrect reward signals, causing an imbalance between exploration and exploitation. OpenRFT addresses this challenge with two approaches. (1) Reasoning process synthesis and SFT: We begin by prompting the vanila model to roll out and fill in the missing reasoning steps in the domain-specific samples. These synthesized data are then used to fine-tune the original reasoning foundation model via SFT. This allows the policy model to adapt to the reasoning process of the domain task, providing a more robust starting point for the subsequent RL phase. (2) Incorporating a Process Reward Model (PRM) in RL: PRM helps supervise the rationality of the reasoning process, enhance the probability of correct reasoning process rollouts, and thus stabilize the RL training.

For the second challenge, Reinforcement Learning (RL) generates data by exploring the environment and adapting its learning distribution. This reduces reliance on the initial sample quantity. However, having only dozens or hundreds of samples may still be insufficient. OpenRFT addresses this with two approaches. (1) Data augmentation: This approach directly increases the data volume by rephrasing questions and shuffling options to generate new domain-specific samples. (2) Domain knowledge embedding: This approach aims to enhance the efficiency of RL exploration. We introduce a simple prompting-based technique: utilizing domain-specific samples in a few-shot In-Context Learning (ICL) setup to guide the policy model's exploration.

We selected reasoning foundation model and process reward model from the Skywork-o1 series (o1 Team, 2024). The evaluation is conducted on SciKnowEval (Feng et al., 2024), a newly-released scientific benchmark that encompasses five distinct abilities. We have chosen 8 specific tasks corresponding to the reasoning ability at level L3, spanning various disciplines including biology, chemistry, physics, and materials science. Experimental results show that using only 100 domain-specific samples, OpenRFT increases performance by an average of 11%. We also find that more data, a stronger reasoning foundation model, and better alignment of the action space all contribute to improved results.

It is important to note that, much like how an effective System-2 model (e.g., o1) relies on a powerful System-1 model (e.g., GPT-4o), the feasibility of RFT depends on having a strong generalist reasoning model and a corresponding PRM. Only with such models can the reasoning process for domain-specific tasks be understood and scored. Since there is currently no open-source, highly effective generalist reasoning model, we merely offer an early implementation of RFT. It is believed that when stronger general reasoning models emerge, the full potential of RFT will be further unlocked.

2 METHOD

The key to Reinforcement Fine-Tuning lies in the effective utilization of limited domain-specific samples. Based on the way domain-specific samples are leveraged, as illustrated in Fig. 1, the framework of OpenRFT can be divided into three modules. (1) *Data augmentation*: By rewriting questions and shuffling options, we explicitly generate more domain-specific data. This helps explore a broader range of states and actions in the RL stage. (2) *SFT-based imitation*: Using a stronger reasoning foundation model as a teacher [1], the missing reasoning steps are synthesized for the provided domain-specific data. These enhanced samples are then used to pre-adapt the student policy model through SFT. (3) *RL-based exploration and self-improvement*: The domain-specific samples are incorporated into the policy model in a few-shot ICL manner. The policy model, under process supervision by the PRM, explores and continuously optimizes within an RL environment.

[1] Typically, a smaller reasoning foundation model (e.g., o1-mini) is desired to ensure efficiency in domain-specific applications. When synthesizing reasoning step data, it is ideal to use a stronger reasoning foundation model as the teacher model for distillation (e.g., o1). It is important to ensure that the action space of the teacher and student models remains consistent.

Due to the lack of a stronger reasoning model with consistent actions, in our reported experiments, the synthesis is instead performed by the policy model itself.

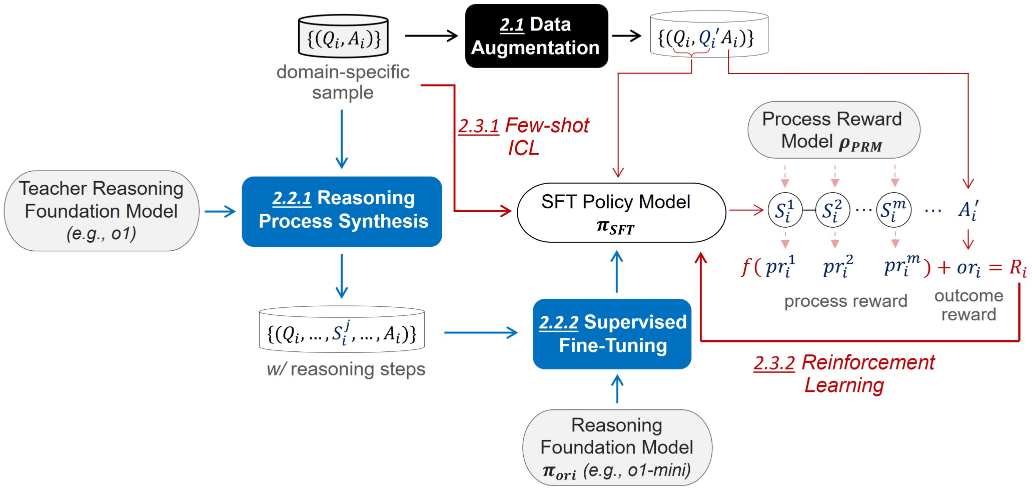
{(Q_i, A_i)}
domain-specific sample
2.1 Data Augmentation
{(Q_i, Q_i′ A_i)}
2.3.1 Few-shot ICL
Process Reward Model ρ_PRM
Teacher Reasoning Foundation Model (e.g., o1)
2.2.1 Reasoning Process Synthesis
SFT Policy Model π_SFT
S_i^1 S_i^2 ⋯ S_i^m ⋯ A_i′
f(pr_i^1 pr_i^2 pr_i^m) + or_i = R_i
process reward
outcome reward
{(Q_i, …, S_i^j, …, A_i)}
w/ reasoning steps
2.2.2 Supervised Fine-Tuning
2.3.2 Reinforcement Learning
Reasoning Foundation Model π_ori (e.g., o1-mini)

Figure 1: OpenRFT framework.

2.1 Data Augmentation

Data augmentation (DA) is a widely used technique for addressing data scarcity. In this framework, we propose a data augmentation method based on question rewriting. Specifically, we first utilize GPT-4o-mini to rewrite the question stem while preserving its original meaning, generating multiple variations. Subsequently, the options for each question are shuffled independently for these variations. Both the original question and the newly transformed question are used simultaneously in the PPO optimization.

Task Instructions for Data Augmentation

Task:
Given a paragraph, generate five distinct expressions while preserving the original meaning.

Instructions:

- The paragraph is a scientific multiple-choice question (without options).
- Keep the meaning unchanged; do not add extra or unrelated information.
- Adjust sentence structure if necessary, but the meaning must remain the same.
- Provide five variations, each separated by ”<sep >”.

Question:
`{question_paragraph}`

Figure 2: Task instructions for generating distinct expressions

2.2 SFT-based Imitation

2.2.1 Reasoning Process Synthesis

To enhance the understanding and response capabilities of a reasoning foundation model in specific domains, we adopt a teacher reasoning foundation model to synthesize reasoning process data and leverage this data for SFT of the reasoning foundational model. The teacher model is typically a stronger general reasoning foundation model capable of synthesizing high-quality reasoning samples tailored to domain-specific data.

Specifically, for each question Q_i in the domain dataset(Q_i, A_i), We designed a set of general prompts to leverage the reasoning capabilities of the teacher model, enabling it to generate initial reasoning processes and answers within its action space. To ensure the correctness and diversity of the generated data, we introduce a multi-sampling strategy to synthesize high-confidence reasoning process data $(Q_i, \ldots, S_i^j, \ldots, A_i')$, where S_i^j represents the j-th sampled intermediate reasoning step generated by the teacher model. By sampling multiple reasoning paths for each question Q_i, we select at least one data that can infer the correct answer A_i.

For each question Q_i, the final synthesized reasoning data (Q_i, S_i, A_i) includes the question, the intermediate reasoning steps S_i, and the corresponding answer A_i'. These data points are then aggregated to form the domain-specific reasoning dataset $\mathcal{D}_{\text{process}}$, which serves as a high-quality resource for SFT of the reasoning foundation model.

2.2.2 SFT OF POLICY MODEL

After completing the reasoning process synthesis tasks described in Section 2.2.1, we use each complete reasoning solution in the dataset to initialize the policy model π_{ori}. This step aims to let the policy model π_{ori} learn and generalize reasoning patterns and strategies embedded in $\mathcal{D}_{\text{process}}$, thereby enabling it to generate accurate and coherent reasoning solutions for new, unseen questions within the domain.

Given the question Q_i, the policy model π_{ori} utilizes the reasoning dataset $\mathcal{D}_{\text{process}}$ to predict intermediate reasoning steps and derive a final answer. Specifically, π_{ori} is trained to maximize the likelihood of producing the synthesized reasoning paths and correct answers from $\mathcal{D}_{\text{process}}$. The training objective is formulated as:

$$\mathcal{L}_{\text{SFT}} = -\sum\nolimits_{(Q_i, S_i, A_i) \in \mathcal{D}_{\text{process}}} \log P(S_i, A_i | Q_i; \pi_{ori}),$$

where $P(S_i, A_i | Q_i; \pi_{ori})$ represents the probability of the policy model generating the reasoning steps S_i and answer A_i conditioned on the question Q_i.

During training, the model learns to balance the trade-off between mimicking the high-quality reasoning paths synthesized by the teacher model and generalizing to new questions.

After the SFT process, we obtain the updated policy model π_{SFT}. This model incorporates the reasoning patterns, strategies, and domain-specific knowledge embedded in the dataset $\mathcal{D}_{\text{process}}$.

2.3 RL-BASED EXPLORATION AND SELF-IMPROVEMENT

2.3.1 FEW-SHOT ICL-BASED DOMAIN KNOWLEDGE EMBEDDING

In scenarios where there are only a few dozen training samples within a specific domain, the policy model lacking domain knowledge may face inefficiencies during the exploration stage. To address this, we employ a prompt-based in-context learning paradigm, with the aim of supplementing the policy model with knowledge and steering it toward reasoning and answering in the correct direction.

For each training task, we construct a corresponding vector database. For each training sample Q_i, we retrieve the top k most similar question-answer pairs $\{(Q_j, A_j) \mid j = 1, 2, \ldots, k\}$ based on vector similarity. These pairs are then concatenated with the task instructions to create an enhanced prompt enriched with domain-specific knowledge.

2.3.2 RL WITH PRM

The reward feedback based only on the outcome is easily influenced by the correct answer sampled in reinforcement learning, but with incorrect processes. This phenomenon is more likely to occur in long-range reasoning tasks. To mitigate the negative impact of this phenomenon to some extent, we integrate a process-based reward model ρ_{PRM} into the reward design of reinforcement learning.

In the reinforcement learning phase, the policy model π_θ improves itself using domain-specific datasets constructed under different data organization schemes. We model the entire sampling as a language-augmented Markov Decision Process(MDP), formally represented as $\mathcal{M} =$

$(\mathcal{V}, \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$(Wang et al., 2024; Carta et al., 2023). $\mathcal{V}$ represents the vocabulary, while the action space $\mathcal{A} \subseteq \mathcal{V}^N$ and state space $\mathcal{S} \subseteq \mathcal{V}^N$ are both made up of sequences of tokens. In this framework, s_0 represents the given question Q_i, and the action a_t is considered as either a reasoning step or the final answer prediction (referring to S_i^t as mentioned in Figure 1). Here, different actions a_t are separated by newline characters. The state transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ defines how the current state $s_t \in \mathcal{S}$ changes when an action $a_t \in \mathcal{A}$ is taken. Specifically, it concatenates the tokens in action a_t directly to the current state s_t to form a new state $s_{t+1} = \mathcal{T}(s_t, a_t)$. The reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^+$ consists of both outcome-based rewards and process-based rewards.

For the final action in the sampling process, where the policy model π_θ produces the answer prediction A_i', we compare it with the standard answer (Q_i, A_i) from the domain dataset to obtain the outcome reward score or_i. If A_i' matches A_i, the score is set to 1, otherwise it is set to 0. For all actions a_t except for the final answer prediction, we use the process reward model ρ_{PRM} to generate the process reward score $pr_i^t = \rho_{\text{PRM}}(s_{t-1}, a_t)$ for each reasoning step. We then combine the process rewards and the outcome reward or_i to get the final reward score R_i for the sample (Q_i, A_i) using the recomputing function $f(\cdot)$ and a linear combination:

$$R_i = \alpha \times or_i + (1 - \alpha) \times f(pr_i^1, pr_i^2, \cdots, pr_i^m)$$

Here, the function $f(\cdot)$ represents different reward aggregation strategies, such as mean, minimum, etc. We use the Proximal Policy Optimization (PPO)(Schulman et al., 2017) algorithm to train the policy model π_θ based on this reward function definition. The optimization objective is:

$$\max_\theta \mathbb{E}\left[\min\left(r(\theta)\hat{A}, \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}\right)\right]$$

where r calculates the probability ratio between the new and old policy. The advantage estimate $\hat{A}$ is crucial for guiding updates and is computed as the difference between the expected future return under the current policy and the baseline or value function. This advantage is directly influenced by the reward R, which makes the optimization process closely related to both the accuracy and rationality of the answer prediction and the reasoning process.

3 Experiments

3.1 Datasets

The experimental data in our study is sourced from the knowledge reasoning level (L3) of the dataset provided in SciKnowEval Feng et al. (2024), a newly-released scientific benchmark that encompasses five distinct abilities. In this work, we focus on 8 subsets, named *GB1-ftness-prediction, retrosynthesis, chemical-calculation, molecule-structure-prediction, high-school-physics-calculation, material-calculation, diffusion-rate-analysis, perovskite-stability-prediction* (denoted as *T1-T8*), covering four domains: *Biology, Chemistry, Physics, and Materials*. The questions in these subsets are in the form of multiple-choice questions (e.g., ABCD). To simulate the practical scenario of fine-tuning with limited training samples, we sample 100 training samples and 100 testing samples from each dataset. Not that *diffusion-rate-analysis* contains only 149 examples, so the number of testing samples in this case is limited to 49.

3.2 Comparison Methods

The following methods are introduced as our comparative approaches. In addition to these, we also report the results of o1-mini and gpt-4o-mini, which can be regarded as a strong ceiling performance.

- **Vanilla.** The policy model, i.e., Skywork-o1-Open-Llama-3.1-8B, which is used without any modifications or adjustments.
- **SFT.** The experimental setup involves distilling process data with limited training samples using the policy model, and then performing SFT on the policy model to obtain the final model.
- **SFT+.** SFT with distillation data from a stronger model, QwQ-32B-Preview Team (2024b); Yang et al. (2024).

- **ReFT.** The method is derived from Luong et al. (2024), but without the warm-up stage. The model is obtained by directly training the vanilla policy model and the outcome reward model through PPO reinforcement learning on a limited set of training samples.
- **ReFT + PRM.** The method is built upon the ReFT, with the addition of a process reward model to enhance the reward signal.
- **SFT + RL(PRM).** The policy model is first initialized using SFT with the data from self-distillation, followed by reinforcement learning with process rewerd.
- **SFT + RL(PRM) + DA.** This method builds upon the *SFT + RL(PRM)* by incorporating data augmentation during reinforcement learning, thereby expanding the scale of the reinforcement training dataset.
- **SFT+RL(PRM)+DA+ICL.** This implements all the proposed modules. First, we use data augmentation techniques to increase the number of available training samples. Then, we initialize the policy model by performing SFT on the process data during self-distillation. Finally, we finetune the model using process supervision reinforcement learning, and employ ICL techniques to guide RL exploration.

3.3 Experimental Setup

We conduct experiments using the foundation models from the Skywork o1 Open series o1 Team (2024). These models demonstrate strong cognitive reasoning abilities in tasks involving mathematics, code, and other reasoning tasks. Specifically, we employ Skywork-o1-Open-Llama-3.1-8B as the policy model and Skywork-o1-Open-PRM-Qwen-2.5-7B as the process reward model [2].

Hyper-parameters

In all experiments, training is conducted on devices equipped with NVIDIA H20-96GB GPUs, utilizing OpenRLHF Hu et al. (2024) for reinforcement learning. Both SFT and RL experiments employ LoRA fine-tuning with a rank of 4.

For SFT training, the batch size, learning rate, number of epochs, and maximum sequence length are set to 8, 5e-5, 8, and 2048, respectively. In the PPO setup, the actor and critic learning rates are set to 3e-5 and 6e-5, respectively. Additionally, the maximum generated sequence length is set to 1536, and the KL coefficient is fixed at 0.01. The weighted coefficient, α, for combining the process and outcome reward models is set to 0.7.

Regarding data augmentation, the original dataset is expanded by a factor of six, yielding a total of 600 samples. For synthetic reasoning process data, roll-out sampling is performed on each training sample until a response that correctly matches the true answer is obtained. This reasoning data is then treated as the ground truth for the corresponding training sample. In cases where 64 sampling attempts do not yield the correct answer, particularly when using the QwQ-32B-Preview model, the true answer is directly used as the response in the training data, bypassing the reasoning process data. This strategy is also applied in the Skywork-O1 experiments.

For in-context learning (ICL), Sentence-BERT is used to calculate the similarity between the current question and existing examples. The top 3 most similar examples are then selected as context for the current query.

Finally, for model generation, the temperature coefficient is set to 0.6 for all models, except for o1-mini, where the official guidelines specify a temperature coefficient of 1. All other generation parameters are set to their default values.

Evaluation

[2] In proprietary implementations, such as OpenAI, the policy model and PRM are expected to be aligned, sharing the same action space and state-transition. However, in typical experimental settings, it is more common that only an open-source generalist reasoning model is available acting as the policy model, while the corresponding PRM is often inaccessible.

In our current implementation, both the policy model and the PRM are from the Skywork-o1 series (o1 Team, 2024). However, the model provider has not explicitly stated that their action spaces are aligned, which may potentially impact the performance of RFT.

Model/ Method	Biology	Chemistry			Physics	Materials			*Avg.*
	T1	T2	T3	T4	T5	T6	T7	T8	
GPT-4o-mini	0.37	0.69	0.84	0.32	0.53	0.49	0.90	0.525	0.583
o1-mini	0.35	**0.86**	**0.87**	0.23	**0.73**	**0.70**	**0.87**	0.50	0.639
Vanilla	0.28	0.55	0.52	0.23	0.45	0.34	0.41	0.41	0.403
ReFT	0.27	0.50	0.52	0.23	0.44	0.33	0.41	0.50	0.402
ReFT+PRM	0.30	0.57	0.49	0.23	0.44	0.36	0.37	0.48	0.405
SFT	0.33	0.53	0.49	0.20	0.45	0.37	0.43	0.49	0.415
SFT+RL(PRM)	0.29	0.59	0.52	0.24	0.47	0.36	0.46	0.57	0.437
SFT+RL(PRM)+DA	0.29	0.63	0.53	0.21	0.47	0.38	0.48	0.59	0.447
SFT+RL(PRM)+DA+ICL	0.33	0.57	0.52	0.28	0.46	0.36	0.49	0.53	0.443

Table 1: Accuracy of different models/methods. **Bold** indicates the highest value, while underline indicates the highest value among the different methods based on the open-source Skywork-o1.

The answer output format is explicitly defined in the prompt to facilitate the extraction of answers from the model's output using predefined rules. This approach allows us to directly compare the model's predictions with the ground truth. We set the maximum sampling length to 2048. For samples where the answer cannot be identified within this length, we consider the prediction as incorrect in the calculation. We report accuracy values for both methods across all datasets. Since the GPT-4o-mini and o1-mini models are sufficiently robust, we report the results based on a single evaluation. For other methods, we perform three evaluations and report the average accuracy.

3.4 Results

The main results are summarized in Table 1. Key observations include: (1) *o1-mini* demonstrated the strongest reasoning capabilities, yet *GPT-4o-mini* showed a competitive performance in certain tasks. As the representative of System-1 models, *GPT-4o-mini* excels in versatility, outperforming *o1-mini* on tasks where domain knowledge is crucial. (2) The contribution of *ReFT* is trivial. Designed to enhance general reasoning abilities, it fails to address the distribution discrepancy between the provided domain samples and the policy model to be fine-tuned. (3) With PRM to supervise the reasoning process, *ReFT+PRM* contributes to increasing the likelihood of sampling correct reasoning processes, although the improvement is not significant. (4) After fine-tuning with self-synthesized reasoning process data, *SFT* slightly outperformed *Vanilla*. This indicates that with such a limited number of samples, relying solely on SFT is far from sufficient. However, it can provide a good exploration starting point for subsequent RL. (5) Compared with *SFT*, *SFT+RL(PRM)* shows obvious improvement, highlighting the necessity of RL in fully leveraging the limited domain-specific samples. (6) *SFT+RL(PRM)+DA* achieves consistent improvement over *SFT+RL(PRM)* in different tasks, validating the contribution of domain-specific samples in synthesizing new samples. It achieves the best performance among the methods initialized with Skywork-o1, an average improvement of 11% compared to *Vanilla*. (7) By further incorporating few-shot ICL, there was no improvement but a slight decrease in the performance. This is possibly due to the inconsistent prompts during the SFT and RL stages. It is interesting to see that few-shot ICL benefits the most challenging task (T4: *molecule-structure-prediction*), which is precisely where GPT-4o-mini excels. This somewhat underscores the effectiveness of domain knowledge. We are experimenting alternative ways for domain knowledge embedding.

3.5 Discussions

More domain-specific data contributes to better RFT results. From Table 1, we observe that data augmentation techniques significantly enhance model performance when training data is limited. This suggests that more domain-specific datasets could further improve the *ReFT* method. Motivated by this, we explored the impact of data size on *ReFT* by comparing the effects of different data sizes on *ReFT* and two variants of *OpenRFT* using the T8 dataset. In all experiments with the *OpenRFT* variants, the models are initialized with the same set of 100 samples across all conditions.

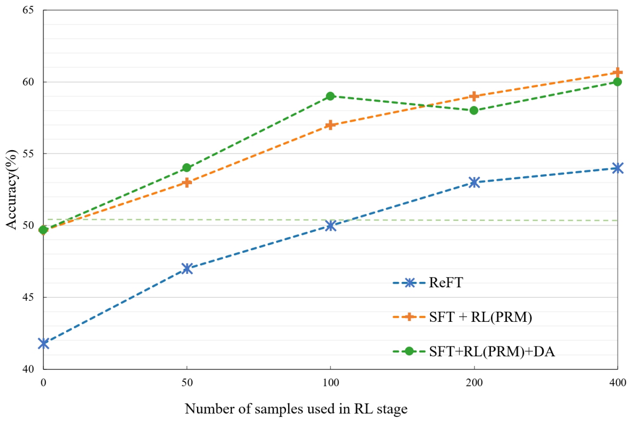

Figure 3: Performance with different sizes of domain-specific data. The light green dashed line represents the performance of SFT with 100 samples.

Model	Biology	Chemistry			Physics	Materials			*Avg.*
	T1	T2	T3	T4	T5	T6	T7	T8	
Vanilla	0.28	**0.55**	**0.52**	**0.23**	**0.45**	0.34	0.41	0.41	0.40
SFT	**0.33**	0.53	0.49	0.20	**0.45**	**0.37**	**0.43**	**0.49**	**0.41**
SFT+	0.27	0.45	0.44	0.12	0.34	0.25	0.28	0.30	0.31

Table 2: Analysis of teacher-student policy alignment. *SFT* and *SFT+* indicate synthesizing reasoning process by the student policy itself and a stronger reasoning model *QwQ-32B*, respectively.

As shown in Fig. 3, the performance of all three methods improves as the number of training samples increases. The influence of data augmentation is most pronounced when the training dataset is small. However, as the size of the data set increases, the benefit of data augmentation diminishes. This reduction in effectiveness may be due to the inherent errors introduced by LLM-based data augmentation, particularly when handling complex molecular formulas or similarly challenging scenarios.

Teacher policy model should align with the student policy model. As introduced in Section 2, it is ideal to use a stronger reasoning foundation model as the teacher model to synthesize the reasoning process for the domain-specific samples. We employed the more powerful QwQ-32B as the teacher model, and then performed SFT on the student policy model using the synthesized reasoning process data.

Table 2 displays the performance of the student policy model when generating its own process data (referred to as *SFT*) and when using the process data synthesized by QwQ-32B-Preview (Team, 2024b) (referred to as *SFT+*). It can be observed that *SFT+* significantly underperforms compared to *SFT*, even falling below the baseline of vanilla models.

Although QwQ-32B-Preview is stronger than the student policy model to be fine-tuned (Skywork o1 Open-Llama-3.1-8B), and the synthesized reasoning process data are likely of higher quality, the discrepancy in teacher-student policy action spaces leads to a degradation in training when using these inconsistent reasoning step data for fine-tuning. This validates the importance of ensuring that the action space of the teacher and student models should be well aligned.

4 Related Work

This section presents a review of related work, framed within the lens of System-1 and System-2 inference. Building on the previously introduced notation, we begin with a straightforward definition of System-1 and System-2 inference.

Training Stages	**Pre-Training**		**Fine-Tuning**	
	Training data	Learning method	Training data	Learning method
System-1	(Q)	Self-supervised learning	(Q, A)	SFT
System-2	(Q, A)	RL + Self-Play	$(Q, \ldots, S^j, \ldots, A)$ [3]	RFT

Table 3: System-1 v.s. System-2: relied training data and used learning method in the pre-training and fine-tuning stages

- **System-1 inference:** This involves directly inferring the answer A_i from the question Q_i, represented as $p(A_i|Q_i)$. It operates in a straightforward, single-step manner.
- **System-2 inference:** System 2 inference, i.e., reasoning, involves multiple intermediate inference steps before deriving an answer. Specifically, it first infers the reasoning steps $\{S_i^1, \ldots, S_i^j, \ldots, S_i^m\}$ from the question Q_i, and then infer the final answer A_i. Formally, this can be expressed as:

$$p(A_i|Q_i) = \sum_{S_i^j} p(\{S_i^1, \ldots, S_i^j, \ldots, S_i^m\}|Q_i) \cdot p(A_i|Q_i, \{S_i^1, \ldots, S_i^j, \ldots, S_i^m\})$$

4.1 Acquiring System-2 Capability from System-1 Model

Recently, there has been a surge of efforts aimed at endowing models with System-2 reasoning capabilities. These approaches typically assume a pre-trained System-1 language model. Based on how the System-1 model is utilized, related work can be categorized into three types. (1) *Prompting-based*: Examples akin to the XoT family, such as CoT (Chain-of-Thought), ToT (Tree-of-Thought), and GoT (Graph-of-Thought), manually design reasoning schema to guide the System-1 model toward multi-step inference. (2) *SFT-based*: Examples including the recent reproduction works of o1, such as OpenO1 (ope, 2024) and Macro-O1 (Zhao et al., 2024), involve collecting training data containing reasoning steps through annotation or distillation and then applying supervised fine-tuning (SFT) to the System-1 model. (3) *RL-based*: Examples including Open-R (Team, 2024a), LLaMA-O1 (SimpleBerry, 2024), and o1-Coder (Zhang et al., 2024), allow the model to explore the underlying reasoning steps autonomously and iteratively optimize its policy.

ReFT (Luong et al., 2024) lies between the second and third types. In the warm-up stage, it uses SFT to acquire a reasoning format, and in the second stage, it applies RL for iterative optimization. However, ReFT differs from the reinforcement fine-tuning (RFT) discussed in this paper. In terms of positioning, ReFT aims to acquire System-2 capabilities from a System-1 model, whereas RFT assumes the existence of a System-2 foundation model and aims to fine-tune it into a domain-specific System-2 model. Methodologically, ReFT relies on training data that contains reasoning steps. When such reasoning steps are unavailable, the RL policy model exhibits a distribution gap with the provided fine-tuning data. Our experiments have shown that ReFT cannot be directly applied to the standard RFT setting.

4.2 Fine-Tuning Foundation Models

Fine-tuning foundation models to obtain domain-specific models usually results in improved domain performance. Moreover, by focusing on specific tasks, the derived models tend to be smaller, leading to more efficient inference. For instance, fine-tuning smaller o1-mini can achieve comparable or even superior domain-specific performance to that of larger o1.

Previous fine-tuning approaches have primarily relied on Supervised Fine-Tuning (SFT), which can be seen as targeting System-1 foundation models. However, SFT heavily depends on annotated data and is prone to overfitting.

In contrast, when fine-tuning System-2 foundation models, the base models already possess reasoning capabilities, enabling them to think, explore, and learn through trial and error. This allows more effective utilization of expert-provided training examples, leading to a deeper and more essential understanding of domain-specific tasks. This is also why many recent works have begun modeling in a System-2 manner, such as machine translation (Zhao et al., 2024), safety alignment (Wang & Sang, 2024), and retrieval-augmented generation (RAG) (Li et al., 2024), achieving results that outperform traditional System-1 methods.

Drawing an analogy to humans, the ability to draw inferences from limited examples is closely tied to the level of intelligence: those with average intelligence may memorize but struggle to deduce, whereas those with higher intelligence are adept at reasoning and can generalize well. Therefore, System-2 fine-tuning can be seen as a natural progression of model development, built upon a strong foundation model that has achieved a certain "intelligence" level.

Table 3 compares System-1 and System-2 w.r.t the pre-training and fine-tuning stages, in terms of their reliance on training data and used learning methods. System-1 directly infers answer from question. Pre-training uses large-scale unlabeled data (without answers) and learns through self-supervised learning. For fine-tuning, System-1 adjusts its parameters using SFT on a small amount of labeled data with answers.

System-2, by contrast, derives reasoning steps from the question before arriving at the answer. In pre-training, reasoning steps are not explicitly provided. Reinforcement Learning (RL) combined with Self-Play is used to explore and generate plausible reasoning steps, iteratively enhancing performance. Regarding the training data for fine-tuning, reasoning steps are optional, as RL is capable of balancing exploration and exploitation as showcased in the proposed OpenRFT. However, if samples containing reasoning steps are available, which is acceptable with just dozens of samples, will lead to better fine-tuning results. These expert-labeled reasoning steps help determine the action space and thus allow for a more effective adaptation of the policy model [4].

4.3 Employing Reinforcement Learning for Fine-Tuning

Fine-tuning generative models with RL has already been explored in previous works, such as RLHF (Reinforcement Learning with Human Feedback (Ouyang et al., 2022; Fan et al., 2024)) and Reinforcement Learning-based Knowledge Distillation (Ashok et al., 2017; Alwani et al., 2022), but not focused on fine-tuning a foundation model to create a domain-specific model. As shown in Table 4, different methods can be distinguished based on the source of the reward model and the policy model to be fine-tuned. For instance, in RLHF, the reward model derives from human preferences, and the policy model is often a base model or an SFT model, aiming to align with human values. In contrast, RFT utilizes domain-specific samples as the source of the reward model, and the policy model is a reasoning foundation model, aimed at acquiring a specialized reasoning model.

We compare RL-based fine-tuning and SFT from two perspectives. (1) *Training objective*: SFT aims to minimize prediction error by directly adjusting the model parameters. In contrast, RL-based fine-tuning focuses on optimizing the policy, aiming to maximize cumulative rewards. This approach allows for adaptive exploration and optimization, accommodating environmental changes and task demands. The flexibility enables the model to respond to uncertainties, not relying on fixed training data or rules, thereby enhancing the model's long-term performance and adaptability. (2) *Data dependency*: SFT relies on a fixed labeled dataset, where training is bound by the availability of labeled examples. On the other hand, RL-based fine-tuning generates new experience data through continuous interaction with the environment. Compared to SFT, RL can achieve effective learning with a small amount of high-quality data, such as human feedback, expert demonstrations, or simulated data.

In addition to the above-mentioned common characteristics, RFT differs from other RL-based fine-tuning methods in two significant ways, which introduce unique challenges. (1) *Inconsistent tasks*: Both RLHF and RL-based knowledge distillation belong to single-task RL methods, where the fine-tuned model and the target application are within the same task. For instance, in RL-based knowledge distillation, the teacher and student models address the same task. In RLHF, value alignment can be seen as an enhancement of the task for the SFT model. However, RFT requires fine-tuning the foundation model to be applied to different downstream domain tasks. Task and domain generalization is thus crucial. (2) *Inconsistent action mode*: RLHF and RL-based knowledge distillation assume consistency between the policy model's action mode and the data source format of the reward model. For example, in RLHF, the preference data of the reward model aligns with the generation

[4] It remains to be seen whether OpenAI's RFT will provide options that accept training data with reasoning steps in its formal release.

Methods	**Reward Model**	**Policy Model**	**Target**
RLHF	Human preference	Base model/ SFT model	Value Alignment
RL-based Knowledge Distillation	Teacher model	Student model	Model Compression
RFT	Domain samples	Foundation reasoning model	Specialized Reasoning

Table 4: Different methods of RL-based fine-tuning.

actions of the policy model, as are the actions in reinforcement distillation. However, in RFT, the provided domain samples lack reasoning step data, while the policy model is a reasoning model acting as multi-step inference. These differences prevent direct application of existing RL fine-tuning methods to RFT.

5 Conclusion & Future Works

This work presents a simple implementation for fine-tuning generalist reasoning models to solve domain-specific tasks. The provided limited domain-specific samples are exploited in question augmentation, synthesizing reasoning-process data, and few-shot ICL, which are integrated within a RL framework supervised by a general PRM. We are working towards updating PRM synchronously, instead of relying on a static generalist PRM. With iteratively refined training data, the PRM and policy model can engage in Self-Play manner to continuously improve performance.

Future work can be divided into two main directions: (1) *Domain knowledge embedding*. Further exploration is needed to enhance both the policy model and PRM, e.g., investigating effective representations of domain knowledge, adapting PRM by comparing differences between the provided expert samples and random samples, etc. (2) *Domain data augmentation*: While RL can search and explore diverse reasoning processes, the current question size of only dozens remains insufficient. In addition to rephrasing questions while retaining the original answers, generating new questions by leveraging unlabeled training data presents significant potential for exploration. Additionally, exploring more challenging samples through adversarial learning is also worth considering. Maintaining a pool of unsolved questions ensures continuous improvement through self-play.

Generalization remains the fundamental challenge for RFT. Beyond further improving the general reasoning capabilities of foundation models, continued exploration is needed in reward calculation for open-ended problems and the effective adaptation of policy actions. For instance, when the problem format extends beyond multiple-choice questions to data like long-form technical reports, it becomes crucial to design appropriate reward functions and define action spaces for efficient search. This is essential for enabling the model to quickly learn the reasoning processes of domain experts.

Acknowledgments

We sincerely thank the helpful discussions from Shangxi Wu, Chao Kong, Yifei Gao, and Xinyan Wen.

References

Open o1: A model matching proprietary power with open-source innovation. https://github.com/Open-Source-O1/Open-O1/, 2024.

Manoj Alwani, Yang Wang, and Vashisht Madhavan. Decore: Deep compression with reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12349–12359, June 2022.

Anubhav Ashok, Nicholas Rhinehart, Fares Beainy, and Kris M Kitani. N2n learning: Network to network compression via policy gradient reinforcement learning. *arXiv preprint arXiv:1709.06030*, 2017.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Learning*, pp. 3676–3713. PMLR, 2023.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Kehua Feng, Keyan Ding, Weijie Wang, Xiang Zhuang, Zeyuan Wang, Ming Qin, Yu Zhao, Jianhua Yao, Qiang Zhang, and Huajun Chen. Sciknoweval: Evaluating multi-level scientific knowledge of large language models, 2024.

Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

Huayang Li, Pat Verga, Priyanka Sen, Bowen Yang, Vijay Viswanathan, Patrick Lewis, Taro Watanabe, and Yixuan Su. Alr2: A retrieve-then-reason framework for long-context question answering. *arXiv preprint arXiv:2410.03227*, 2024.

Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024. URL `https://arxiv.org/abs/2404.03592`.

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

OpenAI. Openai's reinforcement fine-tuning research program. `https://openai.com/form/rft-research-program/`, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

SimpleBerry. Llama-o1: Open large reasoning model frameworks for training, inference and evaluation with pytorch and huggingface. `https://github.com/SimpleBerry/LLaMA-O1`, 2024. Accessed: 2024-11-25.

OpenR Team. Openr: An open source framework for advanced reasoning with large language models. `https://github.com/openreasoner/openr`, 2024a.

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024b. URL `https://qwenlm.github.io/blog/qwq-32b-preview/`.

Jun Wang, Meng Fang, Ziyu Wan, Muning Wen, Jiachen Zhu, Anjie Liu, Ziqin Gong, Yan Song, Lei Chen, Lionel M Ni, et al. Openr: An open source framework for advanced reasoning with large language models. *arXiv preprint arXiv:2410.09671*, 2024.

Yuhang Wang and Jitao Sang. Don't command, cultivate: An exploratory study of system-2 alignment, 2024.

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.

Yuxiang Zhang, Shangxi Wu, Yuqi Yang, Jiangming Shu, Jinlin Xiao, Chao Kong, and Jitao Sang. O1-coder: An o1 replication for coding, 2024. URL `https://arxiv.org/abs/2412.00154`.

Yu Zhao, Huifeng Yin, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. Marco-o1: Towards open reasoning models for open-ended solutions, 2024.

A Appendix

At $25^\circ\mathrm{C}$, for the reaction

$$\mathrm{N_2O_4}(g) \rightleftharpoons 2\mathrm{NO_2}(g)$$

the equilibrium constant is $K_c = 5.85 \times 10^{-3}$. $20.0\,\mathrm{g}$ of $\mathrm{N_2O_4}$ were added to a $5.00\,\mathrm{L}$ container. What is the molar concentration of $\mathrm{NO_2}$ at equilibrium?

A. $0.0146\,\mathrm{M}$ B. $0.0292\,\mathrm{M}$ C. $0.0218\,\mathrm{M}$ D. $0.00728\,\mathrm{M}$

To solve the problem, we need to determine the molar concentration of $\mathrm{NO_2}$ at equilibrium given the initial conditions and the equilibrium constant K_c.

Step 1: Write the balanced chemical equation

The balanced chemical equation for the reaction is:

$$\mathrm{N_2O_4}(g) \rightleftharpoons 2\mathrm{NO_2}(g)$$

Step 2: Define the initial concentrations

The initial moles of $\mathrm{N_2O_4}$ can be calculated using the molar mass of $\mathrm{N_2O_4}$:

$$\text{Molar mass of } \mathrm{N_2O_4} = 14 + 4 \times 16 = 80\,\mathrm{g/mol}.$$

So, the initial moles of $\mathrm{N_2O_4}$ are:

$$\text{Initial moles of } \mathrm{N_2O_4} = \frac{20.0\,\mathrm{g}}{80\,\mathrm{g/mol}} = 0.25\,\mathrm{mol}.$$

The initial molar concentration of $\mathrm{N_2O_4}$ in a $5.00\,\mathrm{L}$ container is:

$$\text{Initial concentration of } \mathrm{N_2O_4} = \frac{0.25\,\mathrm{mol}}{5.00\,\mathrm{L}} = 0.050\,\mathrm{M}.$$

Step 3: Define the change in concentrations

Let x be the change in concentration of $\mathrm{N_2O_4}$ at equilibrium. Since 1 mole of $\mathrm{N_2O_4}$ produces 2 moles of $\mathrm{NO_2}$, the change in concentration of $\mathrm{NO_2}$ will be $2x$.

Step 4: Write the equilibrium concentrations

The equilibrium concentration of $\mathrm{N_2O_4}$ will be: $\mathrm{N_2O_4} = 0.050 - x$.

The equilibrium concentration of $\mathrm{NO_2}$ will be: $\mathrm{NO_2} = 2x$.

Step 5: Write the equilibrium constant expression

The equilibrium constant expression for the reaction is: $K_c = \frac{[\mathrm{NO_2}]^2}{[\mathrm{N_2O_4}]}$.

Substitute the equilibrium concentrations into the expression:

$$5.85 \times 10^{-3} = \frac{(2x)^2}{0.050 - x} = \frac{4x^2}{0.050 - x}.$$

Step 6: Solve the quadratic equation

Rearrange the equation to form a standard quadratic equation:

$$5.85 \times 10^{-3}(0.050 - x) = 4x^2 \rightarrow 4000x^2 + 5.85x - 5.275 = 0$$

Step 7: Use the quadratic formula

The quadratic formula is given by:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a},$$

where $a = 4000$, $b = 5.85$, and $c = -5.275$. Substitute these values into the formula:

$$x = \frac{-5.85 \pm \sqrt{(5.85)^2 - 4 \times 4000 \times (-5.275)}}{2 \times 4000},$$

This gives us two possible solutions:

$$x_1 = \frac{322.333}{8000} = 0.040283625 \qquad x_2 = \frac{-333.283}{8000} = -0.041607875$$

Since a change in concentration cannot be negative, we discard the negative solution. Therefore, $x = 0.040283625\,\mathrm{M}$.

Step 8: Calculate the equilibrium concentration

The equilibrium concentration of $\mathrm{NO_2}$ is:

$$[\mathrm{NO_2}] = 2x = 2 \times 0.040283625 = 0.04056725\,\mathrm{M}.$$

Rounding to four significant figures, we get:

$$[\mathrm{NO_2}] = 0.0406\,\mathrm{M}.$$

However, this value does not match any of the given options. Let's re-evaluate the quadratic solution more carefully. The correct solution should be:

$$x = \frac{5.85 \times 10^{-3}}{8} = 0.007275\,\mathrm{M}.$$

The equilibrium concentration of $\mathrm{NO_2}$ is:

$$[\mathrm{NO_2}] = 2x = 2 \times 0.00728 = 0.0146\,\mathrm{M}.$$

Step 9: Verify the correct answer

The correct answer is:

$$\boxed{\mathrm{A}}$$

Figure 4: Reasoning Flaw in Domain-Specific Data: This example shows a chemical equilibrium problem. The final answer is correct, but the red font highlights errors in reasoning. The incorrect calculation of molar mass in Step 2 leads to a wrong intermediate result, although the final answer is still obtained. If process supervision is used, the above problems can be effectively avoided.