Training Large Language Models for Reasoning through Reverse Curriculum Reinforcement Learning

Zhiheng Xi [1][*] **Wenxiang Chen** [1][*] **Boyang Hong** [1][*] **Senjie Jin** [1][*] **Rui Zheng** [1] **Wei He** [1] **Yiwen Ding** [1]
Shichun Liu [1] **Xin Guo** [1] **Junzhe Wang** [1] **Honglin Guo** [1] **Wei Shen** [1] **Xiaoran Fan** [1] **Yuhao Zhou** [1] **Shihan Dou** [1]
Xiao Wang [1] **Xinbo Zhang** [2] **Peng Sun** [2] **Tao Gui** [3] **Qi Zhang** [1] **Xuanjing Huang** [1]

https://github.com/WooooDyy/LLM-Reverse-Curriculum-RL

Abstract

In this paper, we propose **R**^{3}: Learning **R**easoning through **R**everse Curriculum **R**einforcement Learning (RL), a novel method that employs only outcome supervision to achieve the benefits of process supervision for large language models. The core challenge in applying RL to complex reasoning is to identify a sequence of actions that result in positive rewards and provide appropriate supervision for optimization. Outcome supervision provides sparse rewards for final results without identifying error locations, whereas process supervision offers step-wise rewards but requires extensive manual annotation. **R**^{3} overcomes these limitations by learning from correct demonstrations. Specifically, **R**^{3} progressively slides the start state of reasoning from a demonstration's end to its beginning, facilitating easier model exploration at all stages. Thus, **R**^{3} establishes a step-wise curriculum, allowing outcome supervision to offer step-level signals and precisely pinpoint errors. Using Llama2-7B, our method surpasses RL baseline on eight reasoning tasks by 4.1 points on average. Notably, in program-based reasoning, 7B-scale models perform comparably to larger models or closed-source models with our **R**^{3}.

1. Introduction

Large language models (LLMs) have made impressive advancements in complex, multi-step reasoning, by prompting

Table 1. Comparison of the proposed R^3 with other supervision methods in terms of three key features. **Golden** means whether the supervisory signals are based on golden labels (e.g., correctness) or human preference; **Human-Annotation-free** indicates that the method does not require detailed annotations for each intermediate step; **Step-level Sup.** means whether the method can provide step-by-step supervisory signals.

FEATURE	SUPERVISION METHOD		
	Outcome Sup.	Process Sup.	Ours
Golden	✔	✘	✔
Human-Annotation-free	✔	✘	✔
Step-level Sup.	✘	✔	✔

or learning to generate solutions in a step-by-step Chain-of-Thought manner (Wei et al., 2022; Kojima et al., 2022; Kim et al., 2023a). Training a language model specialized in reasoning is proved to be superior to prompting-based approaches (Uesato et al., 2022; Yu et al., 2023b). However, Supervised Fine-tuning (SFT) focuses on imitating human demonstrations, requiring large-scale, diverse annotations to achieve generalization (Lightman et al., 2023; Yuan et al., 2023; Shen et al., 2021). Reinforcement learning (RL) offers a viable alternative to improve reasoning via exploration and learning (Bai et al., 2022; Ouyang et al., 2022; Zheng et al., 2023; Luo et al., 2023).

When applying RL to complex reasoning tasks, the core challenge lies in identifying a sequence of actions that yield positive rewards and providing appropriate supervisory signals for optimization (Sutton et al., 1998). On one hand, as task difficulty increases, so does the complexity and length of the reasoning chain. LLMs struggle with the accumulation of errors and uncertainties across multiple intermediate steps (Lightman et al., 2023; Yu et al., 2023a; Zhang et al., 2023). The increase of reasoning steps leads to an exponential growth in the search space for reasoning, making it challenging to obtain correct final results (Xie et al., 2023). On the other hand, existing methods for supervised signals
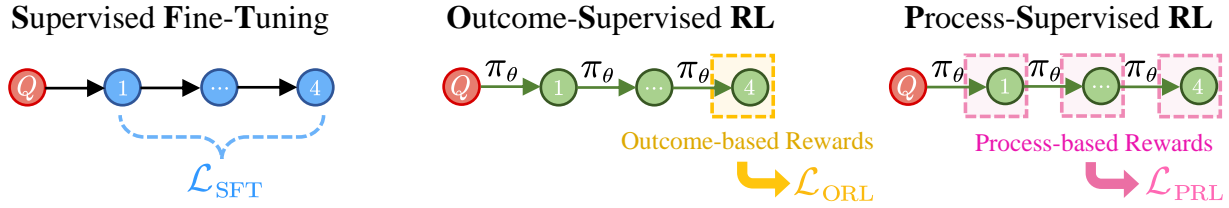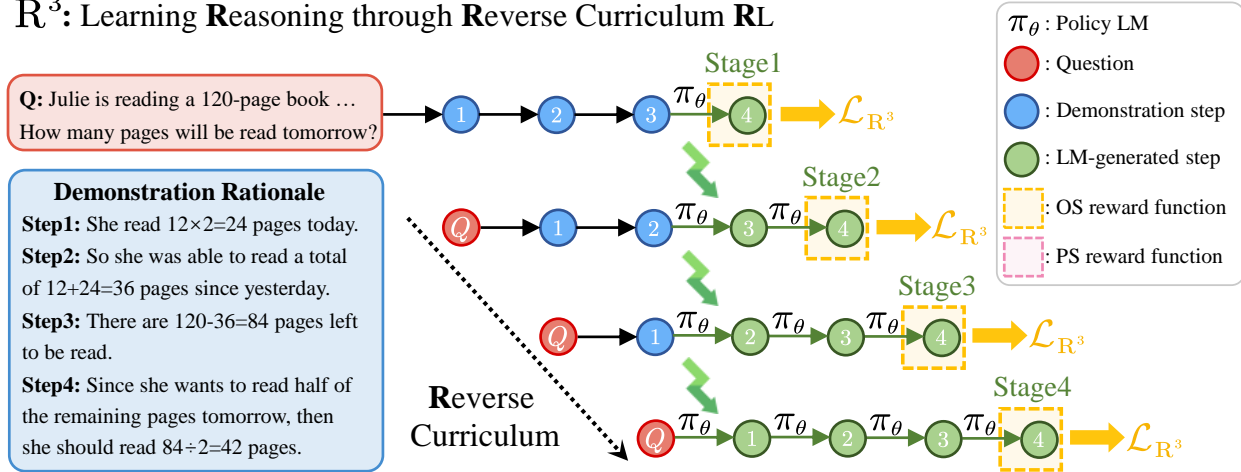
Figure 1. Schematic comparison between R^3 and other methods for training LLMs for reasoning. $\mathcal{L}_{(\cdot)}$ represents the optimization objective for each method. Supervised Fine-Tuning optimizes models using annotated rationales, without additional exploration. In RL, the model first generates a reasoning path and receives supervisory signals for optimization. Outcome-Supervised (OS) RL rewards the final result, while Process-Supervised (PS) RL rewards each reasoning step. The proposed R^3 provides approximately step-by-step supervisory signals similar to PS with only an OS reward function. Specifically, R^3 let the model begin reasoning from a state sampled from a correct demonstration, and provide feedback to supervise the generated actions with outcome supervision. It then gradually moves the start state from the end of the demonstration to the beginning, creating a curriculum with progressively increasing difficulty for exploration.

require a trade-off between feedback quality and annotation cost (Uesato et al., 2022). Outcome supervision (OS, Cobbe et al., 2021; Yu et al., 2023a) rewards only the final outcome (top center in Figure 1), but sparse rewards make it difficult to determine which actions led to success or failure (Wang et al., 2023b). Process supervision (PS, Uesato et al., 2022; Lightman et al., 2023) provides detailed feedback at every step of reasoning (top right in Figure 1), but this approach requires highly skilled annotators to select better reasoning paths, significantly increasing costs (Lightman et al., 2023).

In this work, we propose R^3: Learning **R**easoning through **R**everse Curriculum **R**einforcement Learning (bottom in Figure 1) to address the limitations. It employs only outcome supervision to achieve an effect similar to process supervision. Specifically, R^3 let the model begin reasoning from a state sampled from a correct demonstration, and provide feedback to supervise the generated actions with outcome supervision. By slowly moving the start state from the end of the demonstration to the beginning, the model faces an easy exploration problem at each point where it is

likely to succeed, since it has already learned to solve most of the remaining parts. In this way, a curriculum of gradually increasing exploration difficulty is created, and we can provide approximately step-by-step supervisory signals for the model.

This method facilitates the model's exploration as it shortens the reasoning chain and narrows the sampling space, aiding the model in gaining positive rewards more efficiently. We can interpret R^3 as a form of dynamic programming (Bertsekas, 2012). If N reasoning steps are required to obtain a reward, this reasoning can now be learned in a time that is linear in N, rather than exponential (Florensa et al., 2017; Salimans & Chen, 2018). To improve the training stability and model generalization, we mix the start states of various exploration difficulties for training. Thorough experiments on Llama2-7B demonstrate that R^3 outperforms both the SFT and RL baselines across eight reasoning tasks, achieving an average improvement of 5.4 points and 4.1 points, respectively. Notably, in program-based reasoning on GSM8K, it surpasses SFT and RL by an average of 11.4

points and 4.2 points, respectively. Moreover, Codellama-7B + R^3 outshines models that use extra annotated data like MAmmoTH-Coder (Yue et al., 2023) and Tora (Gou et al., 2023), and is comparable to larger or closed-source models such as GPT-3.5-Turbo.

In summary, we make the following contributions:

1. We propose R^3, a novel method which employs outcome supervision to achieve an effect similar to process supervision, to enhance the reasoning ability of LLMs.

2. We conduct extensive experiments across eight reasoning tasks to highlight the effectiveness of our method. Furthermore, we showcase the superiority of R^3 in program-based reasoning through its application on three models for solving math problems.

3. We perform in-depth ablation and analysis to provide insights into the training dynamics of R^3 and how it works.

2. RL with Outcome and Process Supervision

We use RL notations to describe the language generation process. At each timestep t, the policy language model (LM) π_θ^{RL} parameterized by θ receives a state s_t, which consists of the input prompt and the generated text up to this point. Then, the policy's action a_{t+1} is to generate next token conditioned on the state, and the probability is as $\pi_\theta(a_{t+1}|s_t)$. After that, the environment returns a reward $r(s_t, a_{t+1})$, and the state is transitioned to s_{t+1} with the transition probability $p(s_{t+1}|s_t, a_{t+1})$. The goal of RL is to find an optimal policy to maximize the cumulative reward (i.e., return) over a trajectory $\tau = \{s_0, a_1, ..., s_T, a_T\}$ where s_0 is the initial state (i.e., the prompt) and T is the length of actions. The general form of the policy gradient is gaven as (Mnih et al., 2016):

$$\mathbb{E}_{\tau \sim \pi_\theta^{\mathrm{RL}}} \left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta^{\mathrm{RL}}(a_t|s_{t-1}) R(s_{t-1}, a_t) \right], \quad (1)$$

where $\mathbb{E}_{\tau \sim \pi_\theta^{\mathrm{RL}}}$ refers to the expectation under the distribution of trajectories sampled from the policy π_θ^{RL}. The return $R(s_{t-1}, a_t) = \sum_{t'=t}^{T} \gamma^{t'-t+1} r(s_{t'-1}, a_{t'})$ is the discounted sum of rewards from timestep t with factor $\gamma \in [0, 1)$. With this gradient, we can perform gradient ascent to optimize the model. If the return is favorable, the actions are "reinforced" by increasing their probability of being selected. Given a dataset $\mathcal{D} = \{(s_0^i, \mathbf{a}^i)\}_{i=1}^{N}$ of N pairs of input s_0 and human-generated output sequence $\mathbf{a}$, where $\mathbf{a} = (a_1, a_2, ..., a_T)$ and the whole trajectory is

$\tau = \{s_0, a_1, ..., s_{T-1}, a_T\}$. The policy gradient becomes:

$$\mathbb{E}_{s_0 \sim \mathcal{D}} \left[\mathbb{E}_{\tau \sim \pi_\theta^{\mathrm{RL}}(\cdot|s_0)} \left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta^{\mathrm{RL}}(a_t|s_{t-1}) R(s_{t-1}, a_t) \right] \right]. \quad (2)$$

2.1. Outcome Supervision and Process Supervision

Here we present the operating mechanisms of outcome supervision and process supervision, along with their advantages and limitations, as briefly summarized in Table 1.

Outcome supervision. In outcome supervision, only the final result of the sampled sequence is assigned a reward score, and the score for other tokens are 0 (Cobbe et al., 2021; Yu et al., 2023a):

$$r_o(s_{t-1}, a_t) = \begin{cases} rf_o(s_{t-1}, a_t), & t = T \\ 0, & t \neq T \end{cases}$$

where $rf_o(\cdot)$ is a reward function that returns 1 is the answer is correct else 0. In this paradigm, we don't require detailed annotations for each reasoning step or the training a reward model to allocate rewards. Instead, the golden answer to the question is enough. This supervision is solely based on the correctness, not on the preference of humans. Despite this simplicity, the supervisory signals are sparse, making it challenging for the policy LM to pinpoint reasoning errors accurately. The policy may fall into aimless exploration and struggle in obtaining positive rewards due to the large action space of the LM and the long decision-making chain.

Process supervision. In process supervision, a reward model $rm_p(\cdot)$ is trained to assign a reward score for each intermediate reasoning step (Uesato et al., 2022; Lightman et al., 2023):

$$r_p(s_{t-1}, a_t) = \begin{cases} rm_p(s_{t-1}, a_t), & t \in \mathcal{T}^{Delimiter} \\ 0, & t \notin \mathcal{T}^{Delimiter} \end{cases}$$

where $\mathcal{T}^{Delimiter}$ represents the set of timesteps that delimite each step (e.g., newline or some special symbols). In this paradigm, the rewards are dense, then provide more precise supervision. However, the training for reward model needs fine-grained annotations, which demands skilled annotators and can be very expensive (Lightman et al., 2023; Luo et al., 2023). Additionally, the reward model reflects human preferences, which might introduce bias, and may not always align perfectly with objective correctness or usefulness (Wang et al., 2024b; Pitis, 2023).
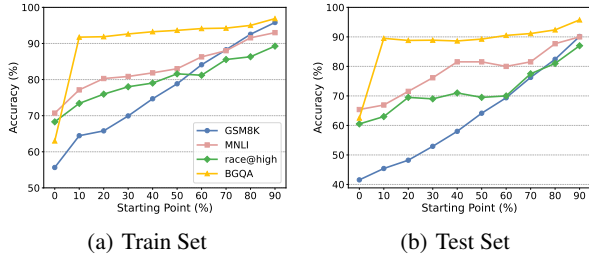
(a) Train Set   (b) Test Set

Figure 2. Accuracy v.s. different start state for exploration. The horizontal axis represents the start state for exploration, with the values indicating the percentage of given actions out of the total actions in the demonstration. The results demonstrate a trend that starting the reasoning from a position closer to the target state makes it easier for the model to obtain a positive reward.

3. Methodology

Motivation. From our previous analysis in Section 2, we seek to merge the benefits of outcome and process supervision while avoiding their drawbacks. We aim to develop a method that doesn't need fine-grained annotations for every step or training a reward model, avoids personal biases by using only golden outcome supervision, and still provides an effect akin to step-level supervision. Hence, we assume access only to the outcome-based reward function $rf_o(\cdot)$ and propose $\mathbf{R}^3$: Learning **R**easoning through **R**everse Curriculum **R**einforcement Learning.

3.1. Start Exploration from Intermediate States of Demonstrations

For a multi-hop reasoning problem, there is a golden answer that can be derived through different reasoning paths. We assume to have access to at least one demonstration, i.e., correct reasoning path that leads to a the golden answer as in supervised fine-tuning. When the model begins exploration from the initial start state s_0, it might face difficulty in obtaining positive rewards as discussed in Section 2.1.

Inspired by previous work in the area of RL with demonstrations (Kakade & Langford, 2002; Subramanian et al., 2016b; Florensa et al., 2017; Salimans & Chen, 2018), we define the set of intermediate states of a given demonstration as $S^{Inter} \subset \mathcal{S}$, and let the policy LM π_θ start exploration from an intermediate state $s_k \in \mathcal{S}^{Inter}$ close to the target state: $\pi_\theta(\mathbf{a}_{k+1:T}|s_k)$ where $\mathbf{a}_{k+1:T} = (a_{k+1}, ..., a_T)$. An outcome-based reward function then provides feedback for the final result, serving as a supervisory signal for actions taken after s_k. In this strategy, the trajectory preceding s_k in the demonstration (i.e., $\{s_0, a_1, s_1, ..., a_k\}$) can serve as a form of guidance, enabling the model to get positive rewards more easily and avoid getting stuck in directionless, inefficient exploration processes, as shown in Figure 2.

3.2. Reverse Curriculum Learning for Step-level Supervision

Once the policy learns to achieve the goal starting from the selected state close to the target, it can extend its training to more distant states (e.g., s_{k-1}), bootstrapping the knowledge it has already acquired. At each point, the model faces an easy exploration problem where it is likely to succeed, as it has already learned to solve most of the remaining parts. In this way, a curriculum of gradually increasing exploration difficulty is created, allowing us to provide approximately step-by-step supervisory signals for the model. Now the policy gradient can be written as:

$$
\mathbb{E}_{s_k \sim \mathcal{S}^{Inter}} \left[\mathbb{E}_{\tau \sim \pi_\theta^{\mathrm{RL}}(\cdot|s_k)} \left[\sum_{t=k+1}^{T} \nabla_\theta \log \pi_\theta^{\mathrm{RL}}(a_t|s_{t-1}) R_o(s_{t-1}, a_t) \right] \right], \tag{3}
$$

where $\mathcal{S}^{Inter}$ refers to the set of intermediate states of a demonstration sampled from dataset $\mathcal{D}$; k starts from $T-1$ and progressively slides back to 0. In the final step, the model begins rolling out from the initial state s_0, which is equivalent to the original outcome-supervised RL.

In multi-step reasoning, language models may generate a large number of actions (i.e., tokens), making it difficult to enumerate all possible intermediate states and explore from these states. Therefore, the number of start states in the reverse curriculum will affect training costs and final reasoning performance. In our method, we sample M intermediate states from demonstrations either at line breaks (if present) or uniformly, as start states for exploration. Thus, a reverse curriculum with M stages is created using these selected starting points[1]. We refer to this method in this paper as **vanilla staged RL**. In our experiments, M is typically 5 or 6 and in Section 5.1, we analyze the impact of the number of stages on reasoning performance.

3.3. Mixing Start States for Generalization

As shown in preliminary experiments in Figure 3, staged RL may have potential limitations. Models might overfit to simple patterns presented in the early stages of the curriculum and fail to generalize effectively when the difficulty increases, leading to a degradation of previously acquired knowledge. Furthermore, our findings indicate that staged RL may struggle to adequately capture and model complex interactions and dependencies inherent within the data. To address this issue, we draw inspiration from the field of

---

[1]Please note that 'stage' here refer to training stages, where the intermediate states sampled in the first stage are those closest to the goal, while the states sampled in the last stage are those farthest from the goal.

Figure 3. Learning curves on test sets with 5 different difficulty level for Staged RL and R^3. The farther the starting point for exploration is from the target, the higher the difficulty level. The horizontal axis represents the training process. The vertical dashed lines indicate the transitions between training stages for staged RL. The experiments are conducted on GSM8K reasoning. Staged RL suffers significant performance drops when transitioning stages, while the performance of R^3 improves stably.

multi-task learning (Ruder, 2017; Zhang & Yang, 2022) and treat each stage as an independent task. In the final **R**^{3}, we adopt a mixed strategy to ensure smooth transitions and cooperative optimization between stages of different difficulty levels, stabilizing the training process and enhancing reasoning performance.

3.4. Reward Design and Policy Optimization

We employ proximal policy optimization (PPO, Schulman et al., 2017) as our basic policy gradient algorithm as it has proved effective in RLHF of LLMs. We apply partial reward ϵ (e.g., $\epsilon = 0.1$) on mathematical reasoning tasks when answer can be extracted and of numeric type to make the reward denser following (Zhong et al., 2017; Le et al., 2022):

$$rf_o(s_{T-1}, a_T) = \begin{cases} 1, & \text{answer correct} \\ \epsilon, & \text{answer not correct, but numeric} \\ 0, & \text{answer not correct} \end{cases}$$

We also design reward functions based on the exploration difficulty, which will be discussed in Section 5.3. Following (Lu et al., 2023), our total reward is the sum of reward function score and the Kullback-Leibler (KL) divergence between the learned RL policy and initial policy π_θ^{Init} scaled by a coefficient factor β:

$$r_{final}(s_{t-1}, a_t) = r_o(s_{t-1}, a_t) \\ - \beta \text{KL}\left(\pi_\theta^{RL}(\cdot|s_{t-1}), \pi_\theta^{Init}(\cdot|s_{t-1}) \right). \quad (4)$$

We calculate advantages with generalized advantage estimate (GAE) and perform optimization similar to Schulman et al. (2017).

Our algorithm is outlined in Algorithm 1. We first construct the curriculum datasets of different stages and describes procedures for vanilla staged RL and the final R^3.

4. Experiments

4.1. Experimental Setup

Datasets. Given that our work focuses on enhancing the reasoning capabilities of LLMs, we select various task types that require reasoning abilities, including logical reasoning, mathematical reasoning, reading comprehension, and natural language inference (NLI). We also consider program-based reasoning (i.e., P-CoT) for math problem solving following Gao et al. (2023), where we execute the generated Python program to obtain the answer.

Regarding mathematical reasoning, we choose GSM8K (Cobbe et al., 2021) and SVAMP (Patel et al., 2021), two widely used datasets. For the logical reasoning, we utilize the BoardgameQA (BGQA, Kazemi et al., 2023), which is a challenging reasoning task containing contradictory information from various sources. We select its "main" subset and "conflict" subset. For NLI, we select the commonly used datasets SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018), and acquire their rationales from CoT-Collection (Kim et al., 2023a). For reading comprehension, we choose race@Middle and race@High (Lai et al., 2017), two challenging reading comprehension tasks, and obtain their rationales from CoT-Collection (Kim et al., 2023a). More details can be seen in Appendix B.

Models and baselines. For CoT reasoning, we choose Llama2-Base-7B (Touvron et al., 2023) as our backbone model because it is widely used. We include few-shot CoT, SFT and RL as our baselines. For P-CoT reasoning, we choose Llama2-Base-7B (Touvron et al., 2023), Glactica (Taylor et al., 2022), and Codellama-7B (Rozière et al., 2023) as our backbone. We include few-shot P-CoT, SFT and RL as baselines. We also consider recently proposed methods/models that require data augmentation, including MAmmoTH-Coder (7B & 34B, Yue et al., 2023), Tora and Tora-coder (7B & 13B, Gou et al., 2023).

Table 2. Evaluating results on CoT Reasoning. The best results of each dataset is in **bold** and marked with underline, while the second is marked with underline. Generally, "Staged RL" represents RL with a reverse, staged manner, while R^3 represents the final method with mixed stages. While the vanilla staged RL is only slightly better than RL baseline, R^3 outperforms all other baselines significantly.

METHOD	MATH REASONING		LOGICAL REASONING		NL INFERENCE		READING COMPRE.		AVERAGE
	GSM8K	SVAMP	BGQA$_{main}$	BGQA$_{conflict}$	MNLI	SNLI	race@High	race@Middle	
Few-shot	15.13	39.62	39.73	34.97	47.69	28.96	38.00	39.20	35.41
SFT	41.55	58.40	62.50	57.25	65.38	68.00	60.50	68.00	60.19
RL	44.67	57.30	65.50	58.15	66.15	69.60	61.50	69.00	61.48
Staged RL	47.69	61.00	67.00	58.60	67.69	68.00	63.00	69.50	62.81
R^3	**50.49**	**64.40**	**67.75**	**59.35**	**72.31**	**72.80**	**68.50**	**71.50**	**65.62**

Table 3. Evaluating results of P-CoT reasoning on GSM8K. Our method is marked in blue and outperforms Few-shot, SFT, and RL. Even against methods needing data augmentation, Codellama + R^3 achieves better performance on a 7B model scale. **Note** that † indicates Tora and Tora-code are trained on additional data in SFT, but this data is not used for R^3 as it's not released.

P-COT METHOD	MODEL SIZE	AUG DATA	PERFOR.
Glactica + Few-shot	6.7B	-	18.6
Glactica + SFT	6.7B	-	57.1
Glactica + RL	6.7B	-	66.1
Glactica + R^3	6.7B	-	**69.3**
Llama2 + Few-shot	7B	-	18.3
Llama2 + SFT	7B	-	57.7
Llama2 + RL	7B	-	63.1
Llama2 + R^3	7B	-	**68.9**
Codellama + Few-shot	7B	-	32.7
Codellama + SFT	7B	-	63.3
Codellama + RL	7B	-	70.7
Codellama + R^3	7B	-	**74.2**
Models Using Extra Training Data			
MAmmoTH-Coder (Yue et al., 2023)	7B	260k	59.4
Tora (Gou et al., 2023)	7B	16k	68.8
Tora (Gou et al., 2023) + R^3	7B	16k†	**73.2**
Tora-code (Gou et al., 2023)	7B	16k	72.6
Tora-code (Gou et al., 2023) + R^3	7B	16k†	**76.3**
Larger Models / Close-sourced Models			
MAmmoTH-Coder (Yue et al., 2023)	13B	260k	64.7
MAmmoTH-Coder (Yue et al., 2023)	34B	260k	72.7
Codex (Chen et al., 2021)	N.A.	-	71.6
GPT-3.5-Turbo (Jie et al., 2023)	N.A.	-	78.0
GPT-4 (OpenAI, 2023; Zhou et al., 2023)	N.A.	-	97.0

Implementation details. Our training is done with eight A100-80GB GPUs and using DeepSpeed framework (Rasley et al., 2020). For few-shot CoT, we run five times with different demonstrations and report the average performance. For SFT, we set the learning rate to $2e-5$. For each RL-related method, we first perform SFT to warm-up and then perform RL. We set the partial reward ϵ to 0.1 for SVAMP and 0.2 for GSM8K. For CoT experiments, we set β to 0.05 in math reasoning and set β to 0.3 in other tasks; for P-CoT experiments, we set β to 0.01. For mathematical tasks, we perform 50 epochs for RL and report the best performance, including CoT and P-CoT. For other tasks, we perform 5 epochs for RL and report the best performance. More details can be seen in Appendix C.

4.2. Experimental Results

Results on CoT reasoning. The main results are demonstrated in Table 2. Generally, we can find that: (1) RL methods consistently perform better than prompt-based methods and SFT, showing that by continuously performing exploration and learning, models can refine their reasoning capabilities over time, similar to (Luo et al., 2023). (2) R^3 outperforms other baselines in all tasks, with an average improvement of 5.4 over SFT and 4.1 over RL, indicating that our method can provide stable and significant optimization. However, staged RL is only a bit better than the RL baseline, possibly due to overfitting and ineffective stage-to-stage adaptation mentioned before.

Specifically, our method can enhance different reasoning ability of models. For example, on mathematical tasks, R^3 shows significant improvements compared to SFT and RL Baselines, suggesting that our method effectively helps models to acquire and refine structured and formal reasoning abilities through exploration. Our method also allows models to handle reasoning tasks with contradictory information (BGQA), demonstrating a notable enhancement in their defeasible reasoning ability (i.e., reasoning with conflicting information guided by preference, Pollock, 1987; Hecham et al., 2018; Maher et al., 2020).

Results on P-CoT reasoning. The evaluating results on program-based reasoning is shown in table 3. We can find that: (1) R^3 outperforms other baselines on P-CoT reasoning across all three models. On average, it exceeds SFT by 11.4 points and surpasses the RL Baseline by 4.2 points. This demonstrates that our method is not only highly effective but also versatile and adaptable, capable of extending to various reasoning styles like programs. (2) Compared to other methods that require data augmentation, e.g., MAmmoTH (Yue et al., 2023), Tora and Tora-code (Gou et al., 2023), Codellama-7B + R^3 achieves the better results in 7B-sized models and matches up well with larger models and closed-source model GPT-3.5-Turbo. (3) When our method is applied to models like Tora and Tora-code, which were trained with additional data during SFT, it still yields signif-

Table 4. Ablation study on GSM8K CoT, by default $\beta = 0.05$, partial reward $\epsilon = 0.2$.

Method Setting	Performance
Llama2-Base 7B + R^3	**50.5**
- KL coefficient $\beta = 0$	46.6
- KL coefficient $\beta = 0.1$	44.1
- remove partial reward ϵ	44.6
- partial reward $\epsilon = 0.1$	45.9
- partial reward $\epsilon = 0.3$	45.4

Table 5. Performance when adopting different reward functions. The "Original" one is the basic reward function that returns 1 if the answer is correct else 0. Other functions assign various rewards according to the difficulty of exploration.

DATASET	REWARD FUNCTION				
	Original	R_{linear}	R_{square}	R_{sqrt}	$R_{discount}$
MNLI	72.3	68.5	68.5	70.0	68.5
race@High	68.5	65.0	65.5	66.0	66.0
GSM8K	50.5	43.7	41.9	44.1	45.1
Average	63.8	59.1	58.7	60.0	59.9

icant performance gain using only the original data in the reinforcement learning phase, demonstrating its adaptability and wide applicability.

5. Analysis and Discussion

5.1. Ablation Study

KL coefficient β and partial reward ϵ. We first conduct ablation study on GSM8K CoT to study the impact of β and ϵ, and the results are shown in Table 4.[2] If we set $\beta = 0$, the exploration space of the model becomes unconstrained, and we observe that R^3 can still perform well, which is different from the conclusions of previous RL methods where the model may collapse without KL penalty (Luong et al., 2024). This may be because R^3 does not require the model to constantly perform exploration from scratch, reducing the sampling space and making it easier to obtain rewards, thus facilitating training. If we set $\beta = 0.1$ to impose higher constraints, we observe a more significant drop in performance, indicating that overly strong KL constraints may hinder the model's optimization.

If we set a small partial reward ϵ or remove it, R^3 obtains a lower performance yet it still outperforms RL and SFT. On the other hand, if we set ϵ to a bigger value 0.3, the performance also drops as too large partial reward might lead the model to settle for obtaining simple rewards (outputting numbers) rather than striving for the correct answer.

Number of intermediate states selected M. As mentioned before, if we include all possible intermediate states as starting points, the cost can be extremely high. However, too small value of M might lead to large gaps between stages. Therefore, we need to find a balance and identify an appropriate M. We perform ablation experiments and the results in Figure 5 show that the performance converges when M reaches an appropriate value, such as 5 or 6, and larger M does not yield significant benefits.

5.2. R^3 Delivers Stable Reinforcement Learning

Figures 4(a) and 4(b) illustrate the training dynamics of vanilla RL and R^3 throughout the training process. We observe that RL encounters instability and fluctuations in training rewards, whereas our method is significantly more stable and yields higher returns. This can be attributed to R^3 providing denser, more detailed, and accurate supervisory signals, facilitating model's exploration and learning. The distinction is also evident in test performance, as shown in Figure 4(c), where our method achieves more stable improvements. We also provide case studies in Appendix F to intuitively show the superiority of our method.

5.3. Difficulty-based Reward Function Design

As mentioned before, when perform exploration from different states of the demonstration, the difficulty for the model to obtain a positive reward varies. This leads to an intuitive question: should we set different amounts of rewards for rollouts of varying difficulty, instead of setting them all to 1 when the final results are correct? Consequently, we use different variants of the reward function to observe their performance changes. Specifically, assuming the length of a demonstration τ is T: $\tau = (s_0, a_1, s_1, a_2...s_T)$, with the starting point as s_k, we approximately define the difficulty of the rolling out process as: $\mu = (T - k)/T$.

We then consider different reward functions related to the difficulty. These functions have different trends of change on the slope, including linear reward function $R_{linear} = \mu$, square reward function $R_{square} = \mu^2$, and square root reward function $R_{sqrt} = \mu^{1/2}$. Inspired by the conception of discount factor in RL, we also consider another discount reward function: $R_{discount} = \gamma^{(T-k)}$, where $\gamma = 0.9$. Experiments in Table 5 show counter-intuitive results that these modified reward functions do not bring performance improvements, but rather, performance decreases. This implies that we should treat each start state fairly.

5.4. Analysis of Training Data Construction

Scaling of training data. We first study the data efficiency of R^3, and the results are shown in Figure 6(a). Overall,

---

[2]See Appendix D.1 for more ablation results on other tasks.

(a) Mean Training Reward   (b) Mean Training Return   (c) Evaluation Accuracy
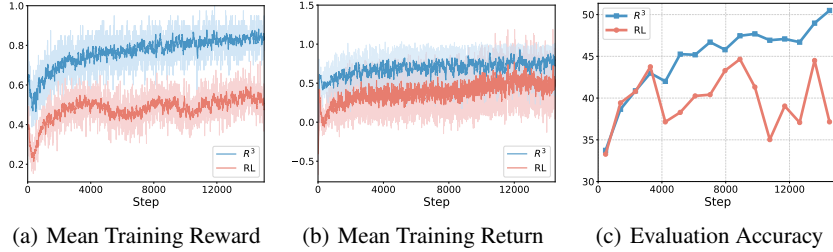
Figure 4. Training dynamics of RL and R³ on GSM8K CoT, including training reward, training return and evaluation accuracy.

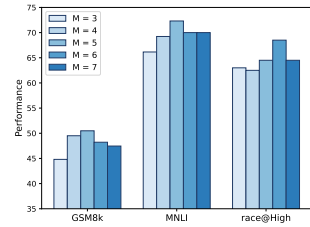*Figure 5.* Ablation study of different stage numbers M.

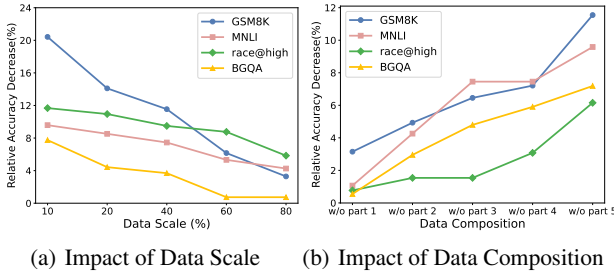(a) Impact of Data Scale   (b) Impact of Data Composition

Figure 6. Impact of data scale and composition. The vertical axis represents the percentage of performance decrease relative to training with full dataset. The horizontal axis of the left subfigure represents the amount of data used, while the horizontal axis of the right subfigure, labeled "w/o part j", indicates removing a part of training data corresponding to a specific difficulty level j.

as the amount of data decreases, the performance of R³ shows a decreasing trend. However, the sensitivity of R³ to data scale varies by task. For instance, on GSM8K, using a limited amount of data leads to a significant decline in performance. This may be because such tasks require a large amount of data to learn enough specialized mathematical knowledge to enable the model to generalize. In contrast, for BGQA, even with limited data scale, the model might still achieve better generalization performance by learning patterns and relationships in the language. Moreover, we demonstrate the absolute values of performance in Appendix D.2, and the results show that R³ can outperform the RL baseline with only a portion of the data.

Which part of data matters? Next we investigate which part of training data is crucial. We remove training data of varying difficulties (i.e., the farther the starting point is from the target, the greater the difficulty) and conduct experiments. Results in Figure 6(b) demonstrate a trend that removing the more difficult data results in poorer performance, highlighting the importance of challenging data. Conversely, removing the simplest data does not significantly degrade performance. We also provide the absolute performance values in the Appendix D.3.

6. Related Work

Reasoning with large language models. Multi-hop complex reasoning is considered one of the most challenging task for LLMs (Rae et al., 2021; Bommasani et al., 2021; Qiao et al., 2022), and researchers have developed several categories of methods, including prompting, supervised fine-tuning methods and reinforcement learning methods. Prompting, with chain-of-thought as a representative one, involves constructing demonstrations and instructions in the prompt to improve model's reasoning performance (Wei et al., 2022; Kojima et al., 2022; Xi et al., 2023; Chu et al., 2023). However, they proved to be sensitive to many factors and model-dependent (Shi et al., 2023; Zellers et al., 2018; Ye & Durrett, 2022). In SFT, models are trained with collected rationales, and their effectiveness largely relies on the scale and quality of the training data (Yuan et al., 2023; Yu et al., 2023b; Yue et al., 2023), necessitating considerable effort in gathering annotations. RL is also used in LLM reasoning, which will be discussed in detail in the next paragraph.

Reinforcement learning for large language models. RL has garnered much attention in LLM alignment (Askell et al., 2021; Bai et al., 2022; Ouyang et al., 2022; Zheng et al., 2023; Wang et al., 2024a), and has been applied in many other tasks like summarization (Ouyang et al., 2022; Stiennon et al., 2020), web navigation (Nakano et al., 2021; Qin et al., 2023) and machine translation (Gülçehre et al., 2023). There are also some work explores enhancing model's reasoning capabilities with RL, based on outcome supervision or process supervision (Lightman et al., 2023; Luo et al., 2023; Wang et al., 2023a; Luong et al., 2024). Furthermore, these two types of supervision are also utilized to perform answer reranking at inference time (Uesato et al., 2022; Cobbe et al., 2021; Yu et al., 2023a), which involves training a reward model based on either outcome or process supervision to rank multiple generated solutions and select the top one. These approaches are orthogonal to our method and can be seamlessly integrated for further improvement.

Reinforcement learning with reverse curriculum. In goal-oriented RL, reverse curriculum learning (Florensa et al., 2018a;b) effectively addresses the problem of sparse rewards (Ladosz et al., 2022). This method involves initially training the agent achieve the target from a starting point near the target, and subsequently relocating the starting point to more distant positions (Wu et al., 2021). Notably, methods that sample starting points from intermediate states of quality demonstrations (Subramanian et al., 2016a; Popov et al., 2017) and trajectories are commonly applied to tasks like the games (Hosu & Rebedea, 2016; Salimans & Chen, 2018) and robotics (Peng et al., 2018; Nair et al., 2018; Plappert et al., 2018). We employ such strategy to address the issue of sparse rewards in outcome supervision of LLM reasoning and provide an effect akin to process supervision.

7. Conclusion and Future Work

In this work, we rethink the existing supervision paradigms of reinforcement learning for large language model reasoning, and propose R^3 that employs only outcome supervision to achieve the benefits of process supervision via reverse curriculum reinforcement learning. We perform thorough experiments on natural language-based and program-based CoT to demonstrate the effectiveness of our method. Moreover, we conduct detailed ablation and analysis to showcase the stability and operating mechanism of our method. In the future, we will attempt to scale up the model size for better performance. Additionally, we will explore the impact of training data with larger scale and diversity on R^3.

Impact Statement

This work proposes a reinforcement learning-based method to enhance the reasoning capabilities of LLMs and achieve better performance. Therefore, it must be used responsibly and reasonably to prevent the training of models that could have negative societal impacts, such as toxic LLMs. Additionally, compared to supervised fine-tuning, RL-related methods may consume more computational resources, potentially leading to greater energy consumption. Future work needs to consider optimizations and improvements in efficiency.

Acknowledgements

References

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., Das-Sarma, N., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Kernion, J., Ndousse, K., Olsson, C., Amodei, D., Brown, T. B., Clark, J., McCandlish, S., Olah, C., and Kaplan, J. A general language assistant as a laboratory for alignment. *CoRR*, abs/2112.00861, 2021. URL https://arxiv.org/abs/2112.00861.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., Das-Sarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., Showk, S. E., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., Johnston, S., Kravec, S., Lovitt, L., Nanda, N., Olsson, C., Amodei, D., Brown, T. B., Clark, J., McCandlish, S., Olah, C., Mann, B., and Kaplan, J. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862, 2022. doi: 10.48550/ARXIV.2204.05862. URL https://doi.org/10.48550/arXiv.2204.05862.

Bertsekas, D. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific, 2012.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R. B., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N. S., Chen, A. S., Creel, K., Davis, J. Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K., Goodman, N. D., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D. E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Koh, P. W., Krass, M. S., Krishna, R., Kuditipudi, R., and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021. URL https://arxiv.org/abs/2108.07258.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. A large annotated corpus for learning natural language inference. In Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., and Marton, Y. (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pp. 632–642. The Association for Computational Linguistics, 2015. doi: 10.18653/V1/D15-1075. URL https://doi.org/10.18653/v1/d15-1075.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chu, Z., Chen, J., Chen, Q., Yu, W., He, T., Wang, H., Peng, W., Liu, M., Qin, B., and Liu, T. A survey of chain of thought reasoning: Advances, frontiers and future. *CoRR*, abs/2309.15402, 2023. doi: 10.48550/ARXIV.2309.15402. URL https://doi.org/10.48550/arXiv.2309.15402.

Cobbe, K., Kosaraju, V., Bavarian, M., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

Florensa, C., Held, D., Wulfmeier, M., Zhang, M., and Abbeel, P. Reverse curriculum generation for reinforcement learning. In *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, volume 78 of *Proceedings of Machine Learning Research*, pp. 482–495. PMLR, 2017. URL http://proceedings.mlr.press/v78/florensa17a.html.

Florensa, C., Held, D., Geng, X., and Abbeel, P. Automatic Goal Generation for Reinforcement Learning Agents, July 2018a. URL http://arxiv.org/abs/1705.06366. arXiv:1705.06366 [cs].

Florensa, C., Held, D., Wulfmeier, M., Zhang, M., and Abbeel, P. Reverse Curriculum Generation for Reinforcement Learning, July 2018b. URL http://arxiv.org/abs/1707.05300. arXiv:1707.05300 [cs].

Gao, L., Madaan, A., Zhou, S., Alon, U., Liu, P., Yang, Y., Callan, J., and Neubig, G. PAL: program-aided language models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10764–10799. PMLR, 2023. URL https://proceedings.mlr.press/v202/gao23f.html.

Gou, Z., Shao, Z., Gong, Y., Shen, Y., Yang, Y., Huang, M., Duan, N., and Chen, W. Tora: A tool-integrated reasoning agent for mathematical problem solving. *CoRR*, abs/2309.17452, 2023. doi: 10.48550/ARXIV.2309.17452. URL https://doi.org/10.48550/arXiv.2309.17452.

Gülçehre, Ç., Paine, T. L., Srinivasan, S., Konyushkova, K., Weerts, L., Sharma, A., Siddhant, A., Ahern, A., Wang, M., Gu, C., Macherey, W., Doucet, A., Firat, O., and de Freitas, N. Reinforced self-training (rest) for language modeling. *CoRR*, abs/2308.08998, 2023. doi: 10.48550/ARXIV.2308.08998. URL https://doi.org/10.48550/arXiv.2308.08998.

Hecham, A., Bisquert, P., and Croitoru, M. On a flexible representation for defeasible reasoning variants. In André, E., Koenig, S., Dastani, M., and Sukthankar, G. (eds.), *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pp. 1123–1131. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018. URL http://dl.acm.org/citation.cfm?id=3237863.

Hosu, I.-A. and Rebedea, T. Playing atari games with deep reinforcement learning and human checkpoint replay, 2016.

Jie, Z., Luong, T. Q., Zhang, X., Jin, X., and Li, H. Design of chain-of-thought in math problem solving. *CoRR*, abs/2309.11054, 2023. doi: 10.48550/ARXIV.2309.11054. URL https://doi.org/10.48550/arXiv.2309.11054.

Kakade, S. M. and Langford, J. Approximately optimal approximate reinforcement learning. In Sammut, C. and Hoffmann, A. G. (eds.), *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia, July 8-12, 2002*, pp. 267–274. Morgan Kaufmann, 2002.

Kazemi, M., Yuan, Q., Bhatia, D., Kim, N., Xu, X., Imbrasaite, V., and Ramachandran, D. Boardgameqa: A dataset for natural language reasoning with contradictory information. *CoRR*, abs/2306.07934, 2023. doi: 10.48550/ARXIV.2306.07934. URL https://doi.org/10.48550/arXiv.2306.07934.

Kim, S., Joo, S. J., Kim, D., Jang, J., Ye, S., Shin, J., and Seo, M. The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 12685–12708. Association for Computational Linguistics, 2023a. URL https://aclanthology.org/2023.emnlp-main.782.

Kim, S., Joo, S. J., Kim, D., Jang, J., Ye, S., Shin, J., and Seo, M. The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning. *arXiv preprint arXiv:2305.14045*, 2023b.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.

Ladosz, P., Weng, L., Kim, M., and Oh, H. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, September 2022.

ISSN 1566-2535. doi: 10.1016/j.inffus.2022.03.003. URL https://www.sciencedirect.com/science/article/pii/S1566253522000288.

Lai, G., Xie, Q., Liu, H., Yang, Y., and Hovy, E. H. RACE: large-scale reading comprehension dataset from examinations. In Palmer, M., Hwa, R., and Riedel, S. (eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pp. 785–794. Association for Computational Linguistics, 2017. doi: 10.18653/V1/D17-1082. URL https://doi.org/10.18653/v1/d17-1082.

Le, H., Wang, Y., Gotmare, A. D., Savarese, S., and Hoi, S. C. Coderl: Mastering code generation through pre-trained models and deep reinforcement learning. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. *CoRR*, abs/2305.20050, 2023. doi: 10.48550/ARXIV.2305.20050. URL https://doi.org/10.48550/arXiv.2305.20050.

Lu, X., Roy, B. V., Dwaracherla, V., Ibrahimi, M., Osband, I., and Wen, Z. Reinforcement learning, bit by bit. *Found. Trends Mach. Learn.*, 16(6):733–865, 2023. doi: 10.1561/2200000097. URL https://doi.org/10.1561/2200000097.

Luo, H., Sun, Q., Xu, C., Zhao, P., Lou, J., Tao, C., Geng, X., Lin, Q., Chen, S., and Zhang, D. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *CoRR*, abs/2308.09583, 2023. doi: 10.48550/ARXIV.2308.09583. URL https://doi.org/10.48550/arXiv.2308.09583.

Luong, T. Q., Zhang, X., Jie, Z., Sun, P., Jin, X., and Li, H. Reft: Reasoning with reinforced fine-tuning. *CoRR*, abs/2401.08967, 2024. doi: 10.48550/ARXIV.2401.08967. URL https://doi.org/10.48550/arXiv.2401.08967.

Maher, M. J., Tachmazidis, I., Antoniou, G., Wade, S., and Cheng, L. Rethinking defeasible reasoning: A scalable approach. *Theory Pract. Log. Program.*, 20(4):552–586, 2020. doi: 10.1017/S1471068420000010. URL https://doi.org/10.1017/S1471068420000010.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Balcan, M. and Weinberger, K. Q. (eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1928–1937. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/mniha16.html.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Overcoming Exploration in Reinforcement Learning with Demonstrations, February 2018. URL http://arxiv.org/abs/1709.10089. arXiv:1709.10089 [cs].

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., and Schulman, J. Webgpt: Browser-assisted question-answering with human feedback. *CoRR*, abs/2112.09332, 2021. URL https://arxiv.org/abs/2112.09332.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774. URL https://doi.org/10.48550/arXiv.2303.08774.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

Patel, A., Bhattamishra, S., and Goyal, N. Are NLP models really able to solve simple math word problems? In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tür, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y. (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pp. 2080–2094. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.168. URL https://doi.org/10.18653/v1/2021.naacl-main.168.

Peng, X. B., Abbeel, P., Levine, S., and van de Panne, M. DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills. *ACM Transactions on Graphics*, 37(4):1–14, August 2018. ISSN 0730-0301, 1557-7368. doi: 10.1145/3197517.3201311. URL http://arxiv.org/abs/1804.02717. arXiv:1804.02717 [cs].

Pitis, S. Failure modes of learning reward models for llms and other sequence models. In *ICML 2023 Workshop The Many Facets of Preference-Based Learning*, 2023.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., Kumar, V., and Zaremba, W. Multi-goal reinforcement learning: Challenging robotics environments and request for research, 2018.

Pollock, J. L. Defeasible reasoning. *Cognitive science*, 11 (4):481–518, 1987.

Popov, I., Heess, N., Lillicrap, T., Hafner, R., Barth-Maron, G., Vecerik, M., Lampe, T., Tassa, Y., Erez, T., and Ried-miller, M. Data-efficient Deep Reinforcement Learning for Dexterous Manipulation, April 2017. URL http://arxiv.org/abs/1704.03073. arXiv:1704.03073 [cs].

Qiao, S., Ou, Y., Zhang, N., Chen, X., Yao, Y., Deng, S., Tan, C., Huang, F., and Chen, H. Reasoning with language model prompting: A survey. *CoRR*, abs/2212.09597, 2022. doi: 10.48550/arXiv.2212.09597. URL https://doi.org/10.48550/arXiv.2212.09597.

Qin, Y., Cai, Z., Jin, D., Yan, L., Liang, S., Zhu, K., Lin, Y., Han, X., Ding, N., Wang, H., Xie, R., Qi, F., Liu, Z., Sun, M., and Zhou, J. Webcpm: Interactive web search for chinese long-form question answering. In Rogers, A., Boyd-Graber, J. L., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 8968–8988. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.499. URL https://doi.org/10.18653/v1/2023.acl-long.499.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, H. F., Aslanides, J., Henderson, S., Ring, R., Young, S., Rutherford, E., Hennigan, T., Menick, J., Cassirer, A., Powell, R., van den Driessche, G., Hendricks, L. A., Rauh, M., Huang, P., Glaese, A., Welbl, J., Dathathri, S., Huang, S., Uesato, J., Mellor, J., Higgins, I., Creswell, A., McAleese, N., Wu, A., Elsen, E., Jayakumar, S. M., Buchatskaya, E., Budden, D., Sutherland, E., Simonyan, K., Paganini, M., Sifre, L., Martens, L., Li, X. L., Kun-coro, A., Nematzadeh, A., Gribovskaya, E., Donato, D., Lazaridou, A., Mensch, A., Lespiau, J., Tsimpoukelli, M., Grigorev, N., Fritz, D., Sottiaux, T., Pajarskas, M., Pohlen, T., Gong, Z., Toyama, D., de Masson d'Autume, C., Li, Y., Terzi, T., Mikulik, V., Babuschkin, I., Clark, A., de Las Casas, D., Guy, A., Jones, C., Bradbury, J., Johnson, M. J., Hechtman, B. A., Weidinger, L., Gabriel, I., Isaac, W., Lockhart, E., Osindero, S., Rimell, L., Dyer, C., Vinyals, O., Ayoub, K., Stanway, J., Bennett, L.,

Hassabis, D., Kavukcuoglu, K., and Irving, G. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446, 2021. URL https://arxiv.org/abs/2112.11446.

Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deep-speed: System optimizations enable training deep learning models with over 100 billion parameters. In Gupta, R., Liu, Y., Tang, J., and Prakash, B. A. (eds.), *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pp. 3505–3506. ACM, 2020. doi: 10.1145/3394486.3406703. URL https://doi.org/10.1145/3394486.3406703.

Rozière, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Remez, T., Rapin, J., Kozhevnikov, A., Evtimov, I., Bitton, J., Bhatt, M., Canton-Ferrer, C., Grattafiori, A., Xiong, W., Défossez, A., Copet, J., Azhar, F., Touvron, H., Martin, L., Usunier, N., Scialom, T., and Synnaeve, G. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950, 2023. doi: 10.48550/ARXIV.2308.12950. URL https://doi.org/10.48550/arXiv.2308.12950.

Ruder, S. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017. URL http://arxiv.org/abs/1706.05098.

Salimans, T. and Chen, R. Learning Montezuma's Revenge from a single demonstration, 2018.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Shen, J., Yin, Y., Li, L., Shang, L., Jiang, X., Zhang, M., and Liu, Q. Generate & rank: A multi-task framework for math word problems. In Moens, M., Huang, X., Specia, L., and Yih, S. W. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pp. 2269–2279. Association for Computational Linguistics, 2021. doi: 10.18653/V1/2021.FINDINGS-EMNLP.195. URL https://doi.org/10.18653/v1/2021.findings-emnlp.195.

Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, E. H., Schärli, N., and Zhou, D. Large language models can be easily distracted by irrelevant context. *CoRR*, abs/2302.00093, 2023. doi: 10.48550/arXiv.2302.00093. URL https://doi.org/10.48550/arXiv.2302.00093.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D. M., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F.

Learning to summarize from human feedback. *CoRR*, abs/2009.01325, 2020. URL https://arxiv.org/abs/2009.01325.

Subramanian, K., Isbell, C. L., and Thomaz, A. L. Exploration from Demonstration for Interactive Reinforcement Learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, AAMAS '16, pp. 447–456, Richland, SC, May 2016a. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-1-4503-4239-1.

Subramanian, K., Jr., C. L. I., and Thomaz, A. L. Exploration from demonstration for interactive reinforcement learning. In Jonker, C. M., Marsella, S., Thangarajah, J., and Tuyls, K. (eds.), *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pp. 447–456. ACM, 2016b. URL http://dl.acm.org/citation.cfm?id=2936990.

Sutton, R. S., Barto, A. G., et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. Alpaca: A strong, replicable instruction-following model. https://crfm.stanford.edu/2023/03/13/alpaca.html, 2023.

Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., Poulton, A., Kerkez, V., and Stojnic, R. Galactica: A large language model for science. *CoRR*, abs/2211.09085, 2022. doi: 10.48550/ARXIV.2211.09085. URL https://doi.org/10.48550/arXiv.2211.09085.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Canton-Ferrer, C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/ARXIV.2307.09288. URL https://doi.org/10.48550/arXiv.2307.09288.

Uesato, J., Kushman, N., Kumar, R., Song, H. F., Siegel, N. Y., Wang, L., Creswell, A., Irving, G., and Higgins, I. Solving math word problems with process- and outcome-based feedback. *CoRR*, abs/2211.14275, 2022. doi: 10.48550/ARXIV.2211.14275. URL https://doi.org/10.48550/arXiv.2211.14275.

Wang, B., Zheng, R., Chen, L., Liu, Y., Dou, S., Huang, C., Shen, W., Jin, S., Zhou, E., Shi, C., Gao, S., Xu, N., Zhou, Y., Fan, X., Xi, Z., Zhao, J., Wang, X., Ji, T., Yan, H., Shen, L., Chen, Z., Gui, T., Zhang, Q., Qiu, X., Huang, X., Wu, Z., and Jiang, Y. Secrets of RLHF in large language models part II: reward modeling. *CoRR*, abs/2401.06080, 2024a. doi: 10.48550/ARXIV.2401.06080. URL https://doi.org/10.48550/arXiv.2401.06080.

Wang, B., Zheng, R., Chen, L., Liu, Y., Dou, S., Huang, C., Shen, W., Jin, S., Zhou, E., Shi, C., et al. Secrets of rlhf in large language models part ii: Reward modeling. *arXiv preprint arXiv:2401.06080*, 2024b.

Wang, P., Li, L., Chen, L., Song, F., Lin, B., Cao, Y., Liu, T., and Sui, Z. Making large language models better reasoners with alignment. *CoRR*, abs/2309.02144, 2023a. doi: 10.48550/ARXIV.2309.02144. URL https://doi.org/10.48550/arXiv.2309.02144.

Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. Math-shepherd: A label-free step-by-step verifier for llms in mathematical reasoning. *arXiv preprint arXiv:2312.08935*, 2023b.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

Williams, A., Nangia, N., and Bowman, S. R. A broad-coverage challenge corpus for sentence understanding through inference. In Walker, M. A., Ji, H., and Stent, A. (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pp. 1112–1122. Association for Computational Linguistics, 2018. doi: 10.18653/V1/N18-1101. URL https://doi.org/10.18653/v1/n18-1101.

Wu, J., Zhang, D., Zhong, S., and Qiao, H. Trajectory-based split hindsight reverse curriculum learning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3971–3978. IEEE Press, 2021. doi: 10.1109/IROS51168.2021.9636842. URL https://doi.org/10.1109/IROS51168.2021.9636842.

Xi, Z., Jin, S., Zhou, Y., Zheng, R., Gao, S., Liu, J., Gui, T., Zhang, Q., and Huang, X. Self-polish: Enhance reasoning in large language models via problem refinement. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pp. 11383–11406. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.findings-emnlp.762.

Xie, Y., Kawaguchi, K., Zhao, Y., Zhao, X., Kan, M.-Y., He, J., and Xie, Q. Self-evaluation guided beam search for reasoning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Ye, X. and Durrett, G. The unreliability of explanations in few-shot in-context learning. *CoRR*, abs/2205.03401, 2022. doi: 10.48550/arXiv.2205.03401. URL https://doi.org/10.48550/arXiv.2205.03401.

Yu, F., Gao, A., and Wang, B. Outcome-supervised verifiers for planning in mathematical reasoning. *CoRR*, abs/2311.09724, 2023a. doi: 10.48550/ARXIV.2311.09724. URL https://doi.org/10.48550/arXiv.2311.09724.

Yu, L., Jiang, W., Shi, H., Yu, J., Liu, Z., Zhang, Y., Kwok, J. T., Li, Z., Weller, A., and Liu, W. Metamath: Bootstrap your own mathematical questions for large language models. *CoRR*, abs/2309.12284, 2023b. doi: 10.48550/ARXIV.2309.12284. URL https://doi.org/10.48550/arXiv.2309.12284.

Yuan, Z., Yuan, H., Li, C., Dong, G., Tan, C., and Zhou, C. Scaling relationship on learning mathematical reasoning with large language models. *CoRR*, abs/2308.01825, 2023. doi: 10.48550/ARXIV.2308.01825. URL https://doi.org/10.48550/arXiv.2308.01825.

Yue, X., Qu, X., Zhang, G., Fu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. Mammoth: Building math generalist models through hybrid instruction tuning. *CoRR*, abs/2309.05653, 2023. doi: 10.48550/ARXIV.2309.05653. URL https://doi.org/10.48550/arXiv.2309.05653.

Zellers, R., Bisk, Y., Schwartz, R., and Choi, Y. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J. (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 93–104. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-1009. URL https://doi.org/10.18653/v1/d18-1009.

Zhang, Y. and Yang, Q. A survey on multi-task learning. *IEEE Trans. Knowl. Data Eng.*, 34(12):5586–5609, 2022. doi: 10.1109/TKDE.2021.3070203. URL https://doi.org/10.1109/TKDE.2021.3070203.

Zhang, Y., Li, Y., Cui, L., Cai, D., Liu, L., Fu, T., Huang, X., Zhao, E., Zhang, Y., Chen, Y., et al. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.

Zheng, R., Dou, S., Gao, S., Hua, Y., Shen, W., Wang, B., Liu, Y., Jin, S., Liu, Q., Zhou, Y., Xiong, L., Chen, L., Xi, Z., Xu, N., Lai, W., Zhu, M., Chang, C., Yin, Z., Weng, R., Cheng, W., Huang, H., Sun, T., Yan, H., Gui, T., Zhang, Q., Qiu, X., and Huang, X. Secrets of RLHF in large language models part I: PPO. *CoRR*, abs/2307.04964, 2023. doi: 10.48550/ARXIV.2307.04964. URL https://doi.org/10.48550/arXiv.2307.04964.

Zhong, V., Xiong, C., and Socher, R. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103, 2017. URL http://arxiv.org/abs/1709.00103.

Zhou, A., Wang, K., Lu, Z., Shi, W., Luo, S., Qin, Z., Lu, S., Jia, A., Song, L., Zhan, M., and Li, H. Solving challenging math word problems using GPT-4 code interpreter with code-based self-verification. *CoRR*, abs/2308.07921, 2023. doi: 10.48550/ARXIV.2308.07921. URL https://doi.org/10.48550/arXiv.2308.07921.

A. Algorithm

Algorithm 1 R^3

Input: Policy language model π_θ, training data $\mathcal{D}$ with N data points, maximum rollout length T, number of stages M, outcome-based reward function $rf_o(\cdot)$.

Initialize policy model π_θ;

Procedure Construct reverse curriculum datasets:

    $\mathcal{D}_{list} \leftarrow [\,]$;

    for stage $m \leftarrow 1...M$ **do**

        $\mathcal{D}^m \leftarrow \emptyset$;

        for data point $(s_0, \mathbf{a}) = \{s_0, a_1, ..., s_{T-1}, a_T\}$ in $\mathcal{D}$ **do**

            Select an intermediate state s_k as the start state for stage m;

            $\mathcal{D}^m \leftarrow \mathcal{D}^m \cup (s_k, \mathbf{a}_{k+1:T})$;

        end

    end

    $\mathcal{D}_{list}$.append($\mathcal{D}^m$)

Procedure Rinforcement learning in a reverse, staged manner:

    for $\mathcal{D}^m$ in $\mathcal{D}_{list}$ **do**

        Perform Reinforcement Learning with π_θ and $rf_o(\cdot)$ on $\mathcal{D}^m$;

    end

Procedure Rinforcement learning with mixed stages:

    $\mathcal{D}^{mixed} \leftarrow \bigcup\limits_{m=1}^{M} \mathcal{D}_{list}[m]$;

    Perform Reinforcement Learning with π_θ and $rf_o(\cdot)$ on $\mathcal{D}^{mixed}$;

B. Evaluation Datasets

Category	Dataset	#Train Samples	#Test Samples
Math Reasoning	GSM8K-CoT (Cobbe et al., 2021)	7473	1319
	GSM8K-P-CoT (Gao et al., 2023)	7356	1319
	SVAMP (Patel et al., 2021)	3067	1000
Logical Reasoning	BGQA$_{main}$ (Kazemi et al., 2023)	3000	800
	BGQA$_{conflict}$ (Kazemi et al., 2023)	2000	800
NL Inference	MNLI (Bowman et al., 2015)	700	130
	SNLI (Williams et al., 2018)	750	125
Reading Comprehension	race@High (Lai et al., 2017)	782	200
	race@Middle (Lai et al., 2017)	734	200

Table 6. Statistics of the evaluation datasets.

Table 6 presents detailed information on the datasets utilized in the main result. Notably, we obtain CoT rationales for the MNLI, SNLI, race@High, and race@Middle datasets from the CoT-Collection (Kim et al., 2023b). Within the CoT-Collection, the annotated rationales are initially extracted based on the 'task' key. Subsequently, the data are filtered according to the 'prompt' and 'source' keys to prevent duplication.

C. Experimental Settings

In Table 7, we provide comprehensive experimental details of the experiment. Here, ϵ refers to the partial reward, and β denotes the KL coefficient. It's worth noting that for mathematical tasks, we perform 50 epochs for RL and report the best performance, including CoT and P-CoT.

Table 7. Implementation details in R^3

Datasets	Few-shot Samples	Actor Model lr	Critic Model lr	β	ϵ
GSM8K-CoT	8	$3e-7$	$3e-7$	0.05	0.2
GSM8K-P-CoT	8	$3e-7$	$3e-7$	0.01	0.2
SVAMP	8	$3e-7$	$3e-7$	0.05	0.1
BGQA$_{main}$	3	$2e-6$	$1e-6$	0.3	0
BGQA$_{conflict}$	3	$2e-6$	$1e-6$	0.3	0
MNLI	8	$2e-6$	$1e-6$	0.3	0
SNLI	8	$2e-6$	$1e-6$	0.3	0
race@High	3	$2e-6$	$1e-6$	0.3	0
race@Middle	3	$2e-6$	$1e-6$	0.3	0

D. Additional Experiments

D.1. Ablation Study

In Table 8, we conduct supplementary ablation studies on Section 5.1, providing results on BGQA$_{main}$, MNLI and race@High datasets. We can observe that if we set $\beta = 0.4$, imposing a stronger KL constraint, there will be a noticeable decrease in performance. If we set β to 0 or 0.1, the performance loss is not as pronounced but still falls below the optimal result.

Table 8. Ablation study on BGQA$_{main}$, MNLI and race@High, by default $\beta = 0.3$.

Dataset	Performance
BGQA$_{main}$	**67.8**
- KL coefficient $\beta = 0$	66.3
- KL coefficient $\beta = 0.2$	66.5
- KL coefficient $\beta = 0.4$	65.5
MNLI	**72.3**
- KL coefficient $\beta = 0$	70.0
- KL coefficient $\beta = 0.2$	70.0
- KL coefficient $\beta = 0.4$	68.4
race@High	**68.5**
- KL coefficient $\beta = 0$	64.5
- KL coefficient $\beta = 0.2$	66.5
- KL coefficient $\beta = 0.4$	64.5

D.2. Experimental Results of Data Scale

As a supplement to Section 5.4, Table 9 presents detailed values of performance. The table illustrates that R^3 achieves performance comparable to full-data training of SFT and RL baselines, using only a fraction of the available data.

Table 9. Impact of data scale

Dataset	R^3 with Data Scaling (%)						Baseline (Full Train Set)	
	10	20	40	60	80	100	SFT	RL
GSM8K	40.2	43.4	44.7	47.4	48.8	50.5	41.6	44.7
MNLI	65.4	66.2	66.9	68.5	69.2	72.3	65.4	66.2
race@High	60.5	61.0	62.0	62.5	64.5	68.5	60.5	61.5
BGQA	62.5	64.8	65.3	67.3	67.3	67.8	62.5	65.5

D.3. Impact of different parts of data.

Table 10 and Table 11 present the accuracy achieved when training the model without specific data parts. Notably, columns 1 through 5 (For race@High, columns1 through 6) signify the ascending difficulty levels of excluded training data, with higher part numbers indicating greater difficulty. The "All Parts" column reflects accuracy when utilizing the entire dataset. Furthermore, based on the results in Section 5.1, we can conclude that for the race@High dataset, optimal performance can be achieved when the number of intermediate states M is set to 6. Therefore, we supplement experiments with race@High containing 6 data parts in Table 11.

Table 10. Comparison of accuracy in training on different data parts

Dataset	w/o Part					All Parts	Baseline (Full Train Set)	
	1	2	3	4	5		SFT	RL
GSM8K	48.9	48.0	47.2	46.9	44.7	50.5	41.6	44.7
MNLI	71.5	69.2	66.9	66.9	65.4	72.3	65.4	66.2
race@High	64.5	64.0	64.0	63.0	61.0	65.0	60.5	61.5
BGQA	67.4	65.8	64.5	63.8	62.9	67.8	62.5	65.5

Table 11. Performance for race@High with 6 intermediate states

Dataset	w/o Part						All Parts	Baseline (Full Train Set)	
	1	2	3	4	5	6		SFT	RL
race@High	65.5	63.5	63.5	63.0	61.0	62.0	68.5	60.5	61.5

E. Prompts

We follow the Alpaca (Taori et al., 2023) prompts format in our experiments. The specific prompts are as follows.

Listing 1. Prompts used in R^3 experiments

```
Below is an instruction that describes a task.
Write a response that appropriately completes the request.

### Instruction:
{instruction}

### Response:
```

F. Case Study

We provide case studies of R^3 and vanilla RL on GSM8K-CoT, GSM8K-P-CoT and MNLI Datasets. Wrong reasoning steps are highlighted in **red**, and reasoning steps corrected by the R^3 method are indicated in **green**. It is evident that the model trained by R^3 has clearer logic and more accurate reasoning when facing complex reasoning tasks, often achieving better task completion.
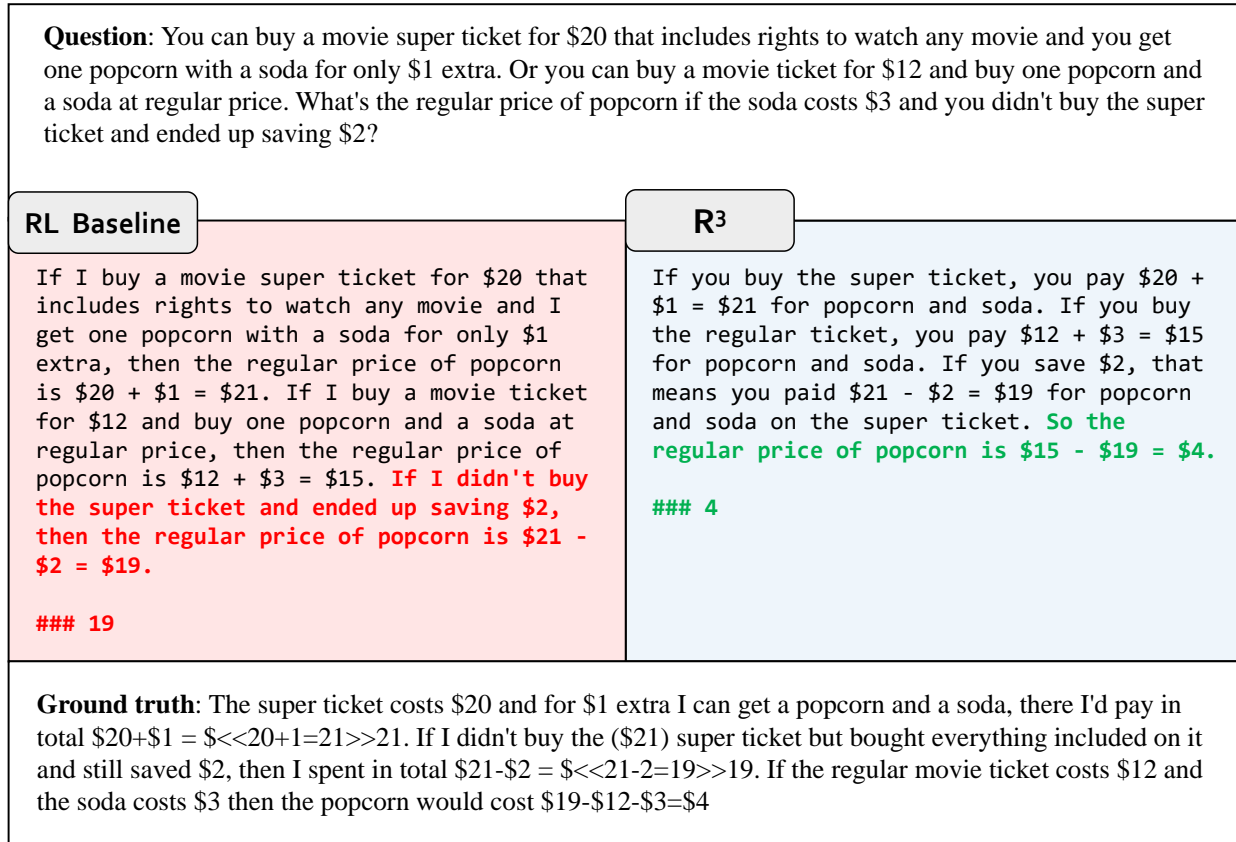
Question: You can buy a movie super ticket for $20 that includes rights to watch any movie and you get one popcorn with a soda for only $1 extra. Or you can buy a movie ticket for $12 and buy one popcorn and a soda at regular price. What's the regular price of popcorn if the soda costs $3 and you didn't buy the super ticket and ended up saving $2?

RL Baseline

```
If I buy a movie super ticket for $20 that
includes rights to watch any movie and I
get one popcorn with a soda for only $1
extra, then the regular price of popcorn
is $20 + $1 = $21. If I buy a movie ticket
for $12 and buy one popcorn and a soda at
regular price, then the regular price of
popcorn is $12 + $3 = $15. If I didn't buy
the super ticket and ended up saving $2,
then the regular price of popcorn is $21 -
$2 = $19.

### 19
```

R³

```
If you buy the super ticket, you pay $20 +
$1 = $21 for popcorn and soda. If you buy
the regular ticket, you pay $12 + $3 = $15
for popcorn and soda. If you save $2, that
means you paid $21 - $2 = $19 for popcorn
and soda on the super ticket. So the
regular price of popcorn is $15 - $19 = $4.

### 4
```

Ground truth: The super ticket costs $20 and for $1 extra I can get a popcorn and a soda, there I'd pay in total $20+$1 = $<<20+1=21>>21. If I didn't buy the ($21) super ticket but bought everything included on it and still saved $2, then I spent in total $21-$2 = $<<21-2=19>>19. If the regular movie ticket costs $12 and the soda costs $3 then the popcorn would cost $19-$12-$3=$4

Figure 7. Comparison of RL Baseline and **R**^{3} on GSM8K-CoT.

Question: Stephen placed an online order for groceries. His final bill came to $40.00. Because this was through a delivery vendor, they tacked on a 25% fee to his final total and charged him $3.00 in delivery fees. Stephen also added a $4.00 tip. After the extra fees, what was the final price of Stephen's groceries?

RL Baseline

```python
def solution( ):
    groceries_cost = 40.00
    delivery_fee = 3.00
    tip = 4.00
    delivery_fee_percent = 0.25
    total_cost = groceries_cost + delivery_fee + tip
    delivery_fee_amount = total_cost *
                          delivery_fee_percent
    total_cost_with_fee = total_cost +
                          delivery_fee_amount
    result = total_cost_with_fee
    return result
```

🐍 58.75

R³

```python
def solution( ):
    bill_initial = 40
    fee_percent = 0.25
    fee_amount = bill_initial * fee_percent
    bill_after_fee = bill_initial + fee_amount
    delivery_fee = 3
    bill_after_delivery_fee = bill_after_fee +
    delivery_fee
    tip = 4
    bill_after_tip = bill_after_delivery_fee + tip
    result = bill_after_tip
    return result
```

🐍 57.00

Ground truth: 57.00

Figure 8. Comparison of RL Baseline and **R**^{3} on GSM8K-P-CoT.

Question:
<Premise>
Limits on IDA balances range from $4,000 in Virginia to $10,000 in South Carolina and $50,000 in Missouri.

<Hypothesis>
The limit on a IDA balance in Missouri is $125

Does the premise entail the hypothesis?
OPTIONS:
- yes
- it is not possible to tell
- no

RL Baseline

```
The premise states that the limit on a
IDA balance in Missouri is $50,000. The
hypothesis also mentions this same number,
but with a different unit of measurement
(dollars instead of thousands). So the
answer is "yes". ### yes
```

R³

```
The premise states that the limit on a
IDA balance in Missouri is $50,000. The
hypothesis suggests a limit of $125,
which is much lower than what was
mentioned in the premise. Hence, it can
be concluded that the answer is "no".
### no
```

Ground truth: The premise mentions that the limit on IDA balance in Missouri is $50,000. Hence, it does not entail the hypothesis of a limit being $125 for this location. ### no

Figure 9. Comparison of RL Baseline and **R**³ on MNLI.