CARFT: Boosting LLM Reasoning via Contrastive Learning with Annotated Chain-of-Thought-based Reinforced Fine-Tuning

Wenqiao Zhu[1,2,*], Ji Liu[1,*], Rongjunchen Zhang[1], Haipang Wu[1], Yulun Zhang[2]
[1]HiThink Research, [2]Shanghai Jiao Tong University

Abstract

Reasoning capability plays a significantly critical role in the the broad applications of Large Language Models (LLMs). To enhance the reasoning performance of LLMs, diverse Reinforcement Learning (RL)-based fine-tuning approaches have been proposed to address the limited generalization capability of LLMs trained solely via Supervised Fine-Tuning (SFT). Despite their effectiveness, two major limitations hinder the advancement of LLMs. First, vanilla RL-based approaches ignore annotated Chain-of-Thought (CoT) and incorporate unstable reasoning path sampling, which typically results in model collapse, unstable training process, and suboptimal performance. Second, existing SFT approaches generally overemphasize the annotated CoT, potentially leading to performance degradation due to insufficient exploitation of potential CoT. In this paper, we propose a Contrastive learning with annotated CoT-based Reinforced Fine-Tuning approach, i.e., CARFT, to enhance the reasoning performance of LLMs while addressing the aforementioned limitations. Specifically, we propose learning a representation for each CoT. Based on this representation, we design novel contrastive signals to guide the fine-tuning process. Our approach not only fully exploits the available annotated CoT but also stabilizes the fine-tuning procedure by incorporating an additional unsupervised learning signal. We conduct comprehensive experiments and in-depth analysis with three baseline approaches, two foundation models, and two datasets to demonstrate significant advantages of CARFT in terms of robustness, performance (up to 10.15%), and efficiency (up to 30.62%). Code is available at https://github.com/WNQzhu/CARFT.

1 Introduction

The reasoning capability of Large Language Models (LLMs) stands as a critical component, driving an extensive array of potential applications, which span mathematical problem (Wang et al., 2024; Luo et al., 2023), financial analysis (Yang et al., 2023; Zhang et al., 2023a), and medical applications (Singhal et al., 2022), etc. The advent of reasoning LLMs, e.g., OpenAI o1 (Jaech et al., 2024), OpenAI o3 (OpenAI, 2025), Llama-Nemotron (Bercovich et al., 2025), Claude 3.7 (Anthropic, 2024), and DeepSeek R1 (DeepSeek-AI, 2025), has significantly heightened the interest in exploring the reasoning capabilities of LLMs across both academic and industrial sectors. Additionally, given the straightforward verification of answers, the task of solving mathematical problems has emerged as a pivotal domain in the study of LLM reasoning capacities.

One of the conventional strategies for augmenting the reasoning capabilities of LLMs is Supervised Fine-Tuning (SFT). SFT entails fine-tuning LLMs with training samples that incorporate annotated Chain-of-Thought (CoT) (Wei et al., 2022). In a training dataset $\mathcal{D}_{\text{train}}$, each training sample is structured as a tuple $(\mathbf{x}, \mathbf{c}, \mathbf{y})$, where $\mathbf{x}$ represents the input question, $\mathbf{c}$ represents the annotated CoT, and $\mathbf{y}$ denotes the correct ground truth answer. CoT $\mathbf{c}$ in the training sample is generally written or labeled by experienced experts or high-end LLMs, which is highly valuable for the fine-tuning of LLMs.

SFT-based reasoning enhancement approaches only exploit a single annotated CoT for each question within the training dataset. However, multiple CoTs (Zhang et al., 2023b) exist for each question. Hence, conducting SFT with only a single annotated CoT in the training dataset may limit the generalization capability of LLMs.

To address the limitations of SFT-based methods, Reinforcement Learning (RL)-based fine-tuning approaches emerge (Luong et al., 2024; Shao et al., 2024; Liu et al., 2025b). A prominent and state-of-the-art RL-based fine-tuning approach is ReFT

*Corresponding authors: zhuwnq@outlook.com and jiliu-work@gmail.com

Method	A-CoT	SG-CoT
SFT	✔	✘
PPO-like (e.g., ReFT)	✘	✔
CARFT	✔	✔

Table 1: An overview of whether methods employ Annotated-CoT (A-CoT) or Self-Generated CoT (SG-CoT).

(Luong et al., 2024), which incorporates online RL approach, i.e., Proximal Policy Optimization (PPO) (Schulman et al., 2017), to dynamically sample CoTs at each training step. This mechanism enables ReFT to leverage multiple CoTs, thereby improving the generalization capability of LLMs.

Despite their effectiveness, two major limitations still exist with the existing RL-based fine-tuning approaches, which hinders the advancement of LLMs. First, existing RL-based approaches solely rely on on-policy sampled CoTs without considering the highly valued annotated CoTs while enhancing reasoning performance. Due to the reward hacking problem (Skalse et al., 2022), such sampled CoTs may not be valid or correct, potentially degrading model performance. Second, existing approaches suffer from unstable training. The inherent exploratory nature of RL can lead to model collapse, i.e., a phenomenon where the behavior of LLMs significantly deteriorates during training. This instability can severely impact the performance of LLMs and result in undesirable outputs.

To address these challenges, we propose a novel Contrastive learning with Annotated CoT-based Reinforced Fine-Tuning approach, i.e., CARFT, which effectively leverages the valuable annotated CoTs in the training dataset while sampling other potential CoTs so as to achieve superb performance. CARFT begins with learning a unified representation for each CoT, encompassing both high-quality annotated CoTs and on-policy sampled CoTs. Based on this representation, we design contrastive signals to improve both the reasoning performance and the stability of the fine-tuning process. Specifically, we propose exploiting a masked loss, e.g., InfoNCE (Chen et al., 2020), to utilize the unified representation to generate the contrastive signal. This contrastive signal serves as a guiding mechanism for the on-policy CoT sampling process, helping to stabilize the fine-tuning of LLMs while maximizing the utilization of informa-

tion from the annotated CoT. Table 1 and Figure 1 illustrate the working characteristic of CARFT. In addition, we propose embedding-enhanced partial reward to further improve the performance. The key contributions of this paper are summarized as follows:

- We propose a novel contrastive learning-based framework with an original contrastive signal construction method that fully exploits annotated CoTs to improve both the performance and the stability in the fine-tuning of LLMs.

- We design an embedding-enhanced partial reward so as to further improve the stability in the reinforced fine-tuning process and to achieve superb performance of LLMs.

- We conduct extensive experiments and thorough ablation studies to demonstrate the effectiveness of CARFT compared with three baseline approaches, two foundation models, and two datasets. Extensive experimental results demonstrate that CARFT significantly outperformances baselines in terms of effectiveness (up to 10.15%) and robustness.

2 Related Work

Reinforcement Learning (RL)-based LLM Reasoning. Recent years have witnessed widespread application in Natural Language Processing (NLP), particularly in the domains of preference optimization (Stiennon et al., 2020; Rafailov et al., 2023; Gheshlaghi Azar et al., 2024; Zhu et al., 2025) and reasoning (Luong et al., 2024; Shao et al., 2024; Yu et al., 2025; Liu et al., 2025b). These methods typically follow a standard three-stage pipeline: (1) SFT, (2) reward modeling, and (3) RL-based optimization. A key distinction among these approaches lies in how the reward signal is obtained. In preference optimization, reward models are learned from human feedback, while in mathematical reasoning tasks, rule-based methods are typically exploited to construct reward signals, as ground-truth answers can be explicitly verified. Within the context of preference optimization, Direct Preference Optimization (DPO) (Rafailov et al., 2023) has emerged as an effective algorithm that avoids the need for explicit reward model training. However, due to its offline nature (Feng et al., 2024), DPO may struggle to explore diverse CoTs (Luong et al., 2024). As a result, on-policy approaches, e.g., GRPO (Shao et al., 2024), DAPO
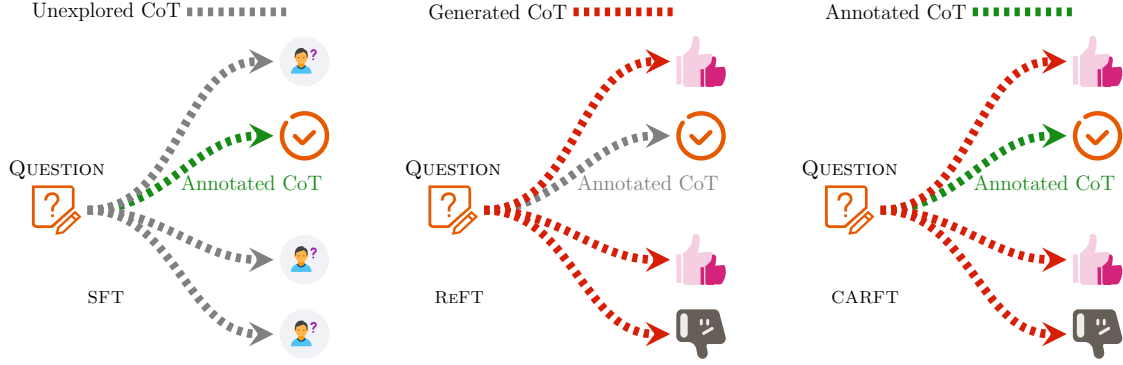
Figure 1: Comparison between SFT, ReFT, and CARFT on the exploration of CoT.

(Yu et al., 2025), Dr.GRPO (Liu et al., 2025b), and ReFT (Luong et al., 2024), are generally employed to better explore such diversity in reasoning.

On-policy approaches utilize multiple rollouts to estimate the Generalized Advantage Estimation (GAE). DAPO and Dr.GRPO are both improved variants of GRPO. Specifically, DAPO is designed for long-CoT scenarios and introduces four key techniques: higher clipping, dynamic sampling, token-level policy gradient loss, and overlong reward shaping. On the other hand, Dr.GRPO improves upon GRPO by eliminating the bias present in the original method. While these approaches are effective, they come with the trade-off of increased computational complexity. In contrast, ReFT (Luong et al., 2024) utilizes only a single on-policy sample per step, making it significantly computationally efficient.

Despite their strengths, these approaches rely solely on on-policy sampling, ignoring potentially valuable annotated CoTs already present in the training data. Moreover, model collapse occurs frequently within the reinforced fine-tuning process with the existing approaches.

Contrastive Learning. Contrastive learning has shown strong effectiveness in diverse fields such as multimodal pretraining (Radford et al., 2021), recommendation systems (Yang et al., 2022), graph embedding (Zhu et al., 2022), and report generation (Zhou and Wang, 2024). Theoretically, the contrastive loss can be decomposed into two components: an alignment term and a uniformity term (Wang and Isola, 2020). The alignment term reduces the distance between embeddings of positive pairs, while the uniformity term encourages embeddings of negative pairs to be uniformly dispersed

in the representation space. Inspired by this framework, we propose in this paper to utilize contrastive feedback to guide the online generation of CoT reasoning.

3 Method

In this section, we first present the preliminary of reinforced LLM fine-tuning. Then, we detail CARFT, including the contrastive learning-based framework with an original contrastive signal construction method and an embedding enhanced partial reward method.

3.1 Preliminary of Reinforced Fine-Tuning

Reinforced fine-tuning incorporates feedback signals derived from either a learned reward model or predefined rules to guide the training of LLMs. Given an input prompt x and the corresponding response y produced by a LLM, the objective is to maximize the expected cumulative reward, which is formally formulated as:

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} \left[r(x, y) \right]$$
$$- \beta \mathbb{D}_{\mathrm{KL}} \left[\pi_\theta(y|x) || \pi_{\mathrm{ref}}(y|x) \right], \quad (1)$$

where $r(x, y)$ denotes the reward function, θ represents the parameters of the LLM, π_θ refers to the learnable target policy, π_{ref} corresponds to a reference policy, i.e., typically the initially pre-trained LLM, exploited to stabilize training, and β denotes the coefficient of KL-divergence, which encourages the updated policy to stay close to the original distribution.

A commonly used algorithm in this setting is PPO (Schulman et al., 2017), which employs GAE (Schulman et al., 2015) for stable gradient updates.

Figure 2: The framework CARFT is composed of two sequential stages: (i) supervised fine-tuning (SFT), followed by (ii) contrastive feedback.

In particular, the advantage at time step t is calculated as:

$$\hat{A}_t = \sum_{l=0}^{L-t} (\gamma\lambda)^l \sigma_{t+l}, \qquad (2)$$

where L represents the maximum length of token sequence, $\sigma_{t'}$ is the Temporal Difference (TD) residual at timestep t', $\lambda \in (0,1]$ serves as the GAE discount factor, and $\gamma \in [0,1]$ controls the discounting of TD residuals. $\sigma_{t'}$ is formulated as:

$$\sigma_{t'} = -V_\phi(s_{t'}) + r_{\text{total}}(s_{t'}, a_{t'}, s_{t'+1}) + \gamma V_\phi(s_{t'}),$$

where $s_{t'}$ denotes the state at time step t', and $a_{t'}$ represents the action at time step t', γ is similar as defined in Formula 2, $r_{\text{total}}(\cdot)$ calculates the total reward, and $V_\phi(\cdot)$ estimates the state value from a given state with ϕ referring to a vector of policy parameters. Actions correspond to individual tokens selected from the vocabulary.

The total reward at the token level includes both the external reward signal and an internal regularization based on the Kullback–Leibler (KL) divergence between the current and reference policies, as defined as follows:

$$\begin{aligned} r_{\text{total}}(s_{t'}, a_{t'}, s_{t'+1}) = \ & r(s_{t'}, a_{t'}) \\ & + \beta \mathbb{D}_{\text{KL}}\left[\pi_\theta(\cdot|s_{t'})||\pi_{\text{ref}}(\cdot|s_{t'})\right], \end{aligned}$$

where π_θ represents the sampling actions policy with policy parameters ϕ, and π_{ref} corresponds to the sampling actions with a reference policy. Given GAE $\hat{A}_t$ and state value $V_\phi(s_t)$, we can estimate the reward as defined in Formula 3:

$$\hat{R}_t = \hat{A}_t + V_\phi(s_t). \qquad (3)$$

Under the PPO framework, the policy and value loss functions are separately formulated to ensure a stable and effective fine-tuning process. The policy loss is defined in Formula 4.

$$\begin{aligned} \mathcal{L}_{\text{policy}}(\theta) = -\mathbb{E}\Bigg[& \min\Bigg(\frac{\pi_\theta(a_t|s_t)}{\pi_\theta^{\text{old}}(a_t|s_t)}\hat{A}_t, \\ & \text{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_\theta^{\text{old}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_t\Bigg)\Bigg], \end{aligned}$$
$$(4)$$

where π_θ^{old} corresponds to the sampling process before update, ϵ represents a hyperparameter controlling the clipping range and preventing excessively large policy updates, and $\text{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_\theta^{\text{old}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_t$ modifies the surrogate objective by clipping the probability ratio, which removes the incentive for moving $\frac{\pi_\theta(a_t|s_t)}{\pi_\theta^{\text{old}}(a_t|s_t)}$ outside of the interval $[1 - \epsilon, 1 + \epsilon]$ (see details in (Schulman et al., 2017)). Then, the value loss is defined as:

$$\mathcal{L}_{\text{value}}(\phi) = \frac{1}{2}\mathbb{E}\Bigg[\max\Bigg(\|V_\phi(s_t) - \hat{R}_t\|^2, \\ \left\|\text{clip}\left(\hat{R}_t - V_\phi(s_t), \hat{A}_t - \epsilon, \hat{A}_t + \epsilon\right)\right\|^2\Bigg)\Bigg],$$

where ϵ is similar as defined in 4. Finally, the overall reinforcement learning loss combines both the policy loss and the value loss as defined in Formula 5:

$$\mathcal{L}_{\text{RL}} = \mathcal{L}_{\text{policy}} + \alpha\mathcal{L}_{\text{value}}, \qquad (5)$$

where α balances the relative importance of the policy and value losses within the reinforced fine-tuning process.

3.2 CARFT

While existing approaches either overemphasizes the annotated CoT (for SFT) or face challenges in achieving stable reinforced fine-tuning while ignoring annotated CoT (for existing RL-based approaches), we propose a novel contrastive learning-based approach, i.e., CARFT, to properly levarage the annotated CoTs so as to address this issue. In this section, we first present the overall workflow of CARFT. Then, we explain the CoT embeddings. Afterward, we propose masked contrastive signal construction approach in CARFT. Finally, we explain a novel embedding enhanced partial reward method.

3.2.1 Workflow

As shown in Figure 2, the overall workflow of CARFT consists of two sequential stages: the SFT stage and the reinforced fine-tuning stage.

SFT We assume that each training sample in the training dataset is a triplet $(\mathbf{x}, \mathbf{c}, \mathbf{y})$, where $\mathbf{x}$ denotes the input question, $\mathbf{c}$ represents the annotated CoT, and $\mathbf{y}$ is the ground-truth answer. We carry out SFT with a few epochs to improve the instruction-following ability of the LLM.

Contrastive Feedback Let $\mathbf{c}_1$ and $\mathbf{c}_2$ denote two distinct CoTs corresponding to the same input question $\mathbf{x}_1$, derived either from training examples or on-policy sampling, i.e., $(\mathbf{x}_1, \mathbf{c}_1, \mathbf{y}_1)$ and $(\mathbf{x}_1, \mathbf{c}_2, \mathbf{y}_1)$. We assume that $\mathbf{y}_1$ is the valid answer to $\mathbf{x}_1$. Since $\mathbf{c}_1$ and $\mathbf{c}_2$ pertain to the same input $\mathbf{x}_1$, we posit the existence of a conditional distribution $p_1(\mathbf{c} \mid \mathbf{h}_1)$ such that both $\mathbf{c}_1 \sim p_1(\mathbf{c} \mid \mathbf{h}_1)$ and $\mathbf{c}_2 \sim p_1(\mathbf{c} \mid \mathbf{h}_1)$, where $\mathbf{h}_1$ denotes a latent variable associated with $\mathbf{x}_1$. Given that $\mathbf{c}_1$ and $\mathbf{c}_2$ are sampled from the same distribution, there should exist a similarity metric $m(\cdot, \cdot)$ under which the distance between $\mathbf{c}_1$ and $\mathbf{c}_2$ is smaller than the distance between $\mathbf{c}_1$ and any $\mathbf{c}_i$ drawn from a different distribution $p_{i,i\neq1}(\mathbf{c} \mid \mathbf{h}_i)$, with a high probability:

$$m(\mathbf{c}_1, \mathbf{c}_2) \leq m(\mathbf{c}_1, \mathbf{c}_i), \quad \text{for } \mathbf{c}_i \sim p_{i,i\neq1}(\mathbf{c} \mid \mathbf{h}_i).$$

This insight provides two key advantages when incorporated as an unsupervised signal. First, it enables us to exploit the annotated CoTs in the training data in the reinforced fine-tuning process of LLMs. Second, it offers a guiding signal for CoT generation, helping to stabilize the reinforced fine-tuning process and to mitigate the risk of model collapse.

3.2.2 Chain-of-Thought Embeddings

Given a CoT $\mathbf{c}$ of length L, represented as:

$$\mathbf{c} = [a_1, a_2, \cdots, a_L],$$

we denote the corresponding token embeddings and state values as:

$$\mathbf{H} = [H_1, H_2, \cdots, H_L]$$

and

$$\mathbf{V}_\phi = [V_\phi(1), V_\phi(2), \cdots, V_\phi(L)],$$

respectively.

To obtain a compact representation of the entire CoT, we compute a weighted sum of the token embeddings using the softmax-normalized state values, as defined in Formula 6.

$$\mathbf{e} = \sum \text{Softmax}(\mathbf{V}_\phi) \odot \mathbf{H}, \qquad (6)$$

where $\odot$ denotes element-wise multiplication between the state values and the corresponding embedding vectors.

In practice, in order to reduce memory consumption, we first project each embedding H_i into a lower-dimensional space using a simple single-layer MultiLayer Perceptron (MLP), denoted by $\text{proj}(\cdot)$. The projected embeddings are then exploited in place of the original ones:

$$\mathbf{H} = [\text{proj}(H_1), \text{proj}(H_2), \cdots, \text{proj}(H_L)].$$

3.2.3 Masked Contrastive Signal Construction

In this section, we design two types of contrastive signals for reinforced fine-tuning, i.e., positive and negative. We denote the signal related to CoT that results in a correct answer by positive signal, and that results in a wrong answer by negative signal.

Positive Signal Given a batch of training samples $\{\mathbf{x}_i, \mathbf{c}_i^{\text{annotated}}, \mathbf{y}_i\}_1^B$ with B presenting the batch size, we conduct LLM self-generation to generate a batch of rollout CoTs, i.e., $\{\mathbf{x}_i, \mathbf{c}_i^{\text{rollout}}, \mathbf{y}_i\}_1^B$. By employing the CoT embedding module, we could get embeddings of the annotated CoTs $\{\mathbf{e}_i^{\text{annotated}}\}_1^B$ and rollout CoT $\{\mathbf{e}_i^{\text{rollout}}\}_1^B$ exploiting the approach presented in Section 3.2.2. We construct a contrastive feedback with InfoNCE (Chen et al., 2020) to provide the positive contrastive signal as defined in Formula 7. We describe the CARFT framework with positive signal in Algorithm 1. For each pair $(\mathbf{c}, \hat{\mathbf{c}})$, where $\mathbf{c}$ denotes

Algorithm 1: CARFT with Positive Signal

> **Input** : Tuples of (*question, CoT, answer*): $\mathcal{D}_{\text{train}} = \{(\mathbf{x}, \mathbf{c}, \mathbf{y})\}$, Number of RL steps: T, Number of updates per RL step: U, Initial policy: π_θ^0.
>
> **Output:** Final Policy: π_θ

1 **for** $i \leftarrow 1$ **to** T **do**
2     $\mathbf{x}, \mathbf{c}, \mathbf{y} \sim \mathcal{D}_{\text{train}}$ // Sample training data from $\mathcal{D}_{\text{train}}$
3     $\hat{\mathbf{c}} \sim \pi_\theta$ // On-policy CoT sampling
4     $\hat{\mathbf{y}} \leftarrow \text{EXTRACT}(\hat{\mathbf{c}})$ // Extract answer
5     $\mathbf{e}^{\text{annotated}} \leftarrow \mathbf{c}, \quad \hat{\mathbf{e}}^{\text{rollout}} \leftarrow \hat{\mathbf{c}}$ // Construct CoT Embeddings
6   Compute $\sigma_t, \hat{A}_t, \hat{R}_t, \mathcal{M}_1$
7   **for** $i \leftarrow 1$ **to** U **do**
8        $\theta, \phi \leftarrow \text{OPTIMIZATION\_STEP}(\mathcal{L})$ // Equation 9
9   **end**
10 **end**
11 **return** π_θ

---

the annotated CoT and $\hat{\mathbf{c}}$ represents the on-policy sampled CoT, we construct the corresponding embeddings for each CoT, resulting in $\mathbf{e}^{\text{annotated}}$ and $\hat{\mathbf{e}}^{\text{rollout}}$, respectively. Which is then utilized to guide the fine-tuning steps.

$$\mathcal{L}_{c_1} = \sum_{i=1}^{B} -\log \frac{\exp(\langle \mathbf{e}_i^{\text{annotated}}, \mathbf{e}_i^{\text{rollout}}\rangle / \tau) \odot \mathcal{M}_1}{\sum_{j=1}^{B} \exp(\langle \mathbf{e}_i^{\text{annotated}}, \mathbf{e}_j^{\text{rollout}}\rangle / \tau)},$$
(7)

where $\mathcal{M}_1$ represents a binary mask, in which each element takes the value 1 if the corresponding CoT leads to a correct answer, and 0 otherwise. The notation $\langle \cdot, \cdot \rangle$ denotes the inner product.

Negative Signal We devise a scheme to utilize the signal within the negative CoT as well. We denote the annotated CoTs and the associated negative CoTs by $\mathbf{c}'^{\text{annotated}}_i$ and $\mathbf{c}'^{\text{rollout}}_i$, respectively. Initially, we calculate the Longest Common Subsequence (LCS) of $\mathbf{c}'^{\text{annotated}}_i$ and $\mathbf{c}'^{\text{rollout}}_i$. Subsequently, based on the LCS and the parts of the sequence that exclude the LCS, we construct four embeddings, denoted $\mathbf{e}^{\text{annotated}}_{i,\text{LCS}}$, $\mathbf{e}^{\text{annotated}}_{i,\text{exc}}$, $\mathbf{e}^{\text{rollout}}_{i,\text{LCS}}$, and $\mathbf{e}^{\text{rollout}}_{i,\text{exc}}$, respectively. Then, the negative contrastive signal is formulated in Formula 8.

$$\mathcal{L}_{c_2} = \sum_{i=1}^{B} -\log \frac{\exp(\langle \mathbf{e}_{i,\text{LCS}}^{\text{rollout}}, \mathbf{e}_{i,\text{exc}}^{\text{annotated}}\rangle / \tau) \odot \mathcal{M}_2}{\sum_{j=1}^{B} \exp(\langle \mathbf{e}_{i,\text{LCS}}^{\text{rollout}}, \mathbf{e}_{j,\text{exc}}^{\text{rollout}}\rangle / \tau)},$$
(8)

where $\mathcal{M}_2$ represents a binary mask, in which each element takes the value 1 if the corresponding CoT leads to a wrong answer, and 0 otherwise. See details in Appendix.

Optimization We optimize the following reinforcement learning loss to learn the policy:

$$\mathcal{L} = \mathcal{L}_{\text{RL}} + c \left\{ \mathcal{L}_{c_1} \text{ or } \mathcal{L}_{c_2} \right\} \tag{9}$$

where c balances the relative importance of the PPO and contrastive losses during the reinforced fine-tuning process (see detailed algorithms in Appendix).

3.2.4 Embedding-enhanced Partial Reward

In order to further improve the stability and the performance of the contrastive signal, we propose an embedding-enhanced partial reward method.

ReFT (Luong et al., 2024) assigns a partial reward $r(x, y) = 0.1$ to the CoT when it is a negative CoT, from which a numerical answer can be extracted. Unlike the partial reward in ReFT, we introduce a fine-grained partial reward by leveraging our unified CoT Embedding, which provides a tool to measure CoT similarity.

$$r(x, y) = \langle \mathbf{e}^{\text{annotated}}, \mathbf{e}^{\text{rollout}} \rangle \times 0.1 + 0.2. \quad (10)$$

The inner product $\langle \mathbf{e}^{\text{annotated}}, \mathbf{e}^{\text{rollout}} \rangle$ ranges from -1 to 1, leading to a partial reward range of $[0.1, 0.3]$. When the CoTs are dissimilar, the inner product approaches -1, resulting in a reward close to 0.1; when they are similar, the reward approaches 0.3. This strategy encourages well-behaved CoT generation. By assigning differentiated rewards to negative CoTs, the embedding-enhanced partial reward method further improve the stability of the reinforced fine-tuning process and the final performance of LLMs.

Method	SVAMP	GSM8K
#Train Samples	3076	7465
#Test Samples	1000	1319

Table 2: Statics of the train and test datasets.

4 Experiments

In this section, we present the experimental results. First, we present our experiment setup. Then, we demonstrate the evaluation of CARFT compared with SFT, ReFT, and Dr.GRPO. Afterward, we present an ablation study.

4.1 Experimental Setup

We conduct experiments on two publicly available datasets: SVAMP (Patel et al., 2021) and GSM8K (Cobbe et al., 2021). Table 2 presents the key statistics of SVAMP and GSM8K. For the reasoning process, we leverage the CoT annotations from (Luong et al., 2024), which were generated based on few-shot prompting (Wei et al., 2022; Gao et al., 2023) with GPT-3.5-turbo (OpenAI, 2023). Our experiments are conducted based on two foundation models: CodeLlama-7B (Rozière et al., 2023) and Qwen2.5-7B-Instruct (Team, 2024). We evaluate CARFT in comparison with three baseline approaches: SFT, ReFT (Luong et al., 2024), and Dr.GRPO (Liu et al., 2025b). ReFT is a state-of-the-art RL approach for LLM fine-tuning. As an advanced extension of GRPO (Shao et al., 2024), Dr.GRPO demonstrates excellent performance on R1-like (DeepSeek-AI, 2025) tasks. See setup details in Appendix.

4.2 Evaluation of CARFT

As illustrated in Table 3, CARFT significantly outperforms SFT and ReFT across different models (up to 10.15% on average). With the SVAMP dataset, CARFT yields substantial accuracy enhancements compared with SFT. Precisely, the accuracy escalates from 62.3% to 64.8% and from 86.9% to 88.0%, with absolute increments of 2.5% and 1.1% for CodeLlama and Qwen2.5-Instruct, respectively. Moreover, with the GSM8K dataset, CARFT showcases remarkable improvements as well. The accuracy climbs from 43.82% to 50.95% and from 80.67% to 84.31%, corresponding to absolute boosts of 7.13% and 3.64% for CodeLlama and Qwen2.5-Instruct, respectively.

Figure 3: Accuracy curves of various methods on SVAMP dataset and Qwen2.5-7B-Instruct backbone.

Table 3 further reveals that ReFT can outperform the SFT baseline once the training process stabilizes. Nevertheless, the performance of ReFT remains inferior to CARFT. In addition, the experimental results demonstrate that ReFT is plagued by the model collapse issue, which significantly undermines its effectiveness. Furthermore, we find that when model collapse occurs, the performance of ReFT lags far behind that of SFT (up to 14.56%) and CARFT (up to 18.2%).

Furthermore, Figures 3 depicts the accuracy curves of different approaches on the SVAMP dataset, with Qwen2.5-7B-Instruct serving as the backbone model. These results indicate that ReFT undergoes model collapse after undergoing fine-tuning for just one epoch. In addition, we find that when using Qwen2.5-7B-Instruct, SFT is prone to an unstable tuning process, as evidenced by the decline in its accuracy as the fine-tuning process advances. In contrast, CARFT exhibits remarkable stability and superb performance throughout the entire training process. This outstanding performance can be attributed to our contrastive feedback mechanism as presented in Section 3.2.3, which offers reference signals for the generation of CoTs.

As demonstrated in Figure 4, CARFT consistently outperforms ReFT during the fine-tuning process and converges rapidly, swiftly reaching peak accuracy values. Nevertheless, the figure also suggests that CARFT is potentially vulnerable to unstable fine-tuning process. See additional experimental results in Appendix.

As shown in Table 4, CARFT with embedding-enhanced partial reward enabled significantly outperforms all baseline approaches in terms of both

Method	Size	SVAMP	GSM8K	Average
CodeLlama + SFT	7B	62.3%	43.82 %	53.06%
CodeLlama + ReFT (Luong et al., 2024)	7B	62.5%	50.27%	56.39%
CodeLlama + CARFT	7B	**64.8%**	**50.95%**	**57.88%**
Qwen2.5-Instruct + SFT	7B	86.9%	80.67%	83.79%
Qwen2.5-Instruct + ReFT (Luong et al., 2024)	7B	85.9%	66.11%	76.01%
Qwen2.5-Instruct + CARFT	7B	**88.0%**	**84.31%**	**86.16%**

Table 3: Evaluation Accuracy of Various Methods on the SVAMP and GSM8K Datasets.

Figure 4: Accuracy curves of various methods on SVAMP dataset and CodeLlama-7B backbone.

Method	Accuracy	Time Cost(hours)
ReFT	62.5%	**14.12**
Dr.GRPO	63.7%	24.49
CARFT	**64.2%**	16.99

Table 4: Evaluation Accuracy of Various Methods on the SVAMP Datasets, based on CodeLlama-7B.

accuracy (up to 0.5% compared with Dr.GRPO and 1.7% compared with ReFT) and efficiency (up to 30.62% compared with Dr.GRPO). Interestingly, Dr.GRPO also surpasses ReFT in terms of performance metrics (1.2%), which is accompanied with a considerable increase in computational time. Specifically, Dr.GRPO relies on significant computing resources due to the generation of a larger number of CoTs.

4.3 Ablation Study

Positive Signal versus Negative Signal We conduct an ablation study to show the impact of positive and negative contrastive signals. As shown in Figure 5, CARFT outperformances ReFT with both

Figure 5: Accuracy of CARFT with positive signal and negative signal, based on the SVAMP dataset and with the CodeLlama-7B as the backbone model.

Figure 6: Accuracy Curve of CARFT with positive signal and negative signal, based on the SVAMP dataset and with the CodeLlama-7B as the backbone model.

positive (2.30% higher) and negative contrastive (0.4% higher) signals. Notably, the positive signal demonstrates a more pronounced performance gain (1.9% higher) compared to its negative counterpart. Due to its excellent performance, we employ the positive signal in our experiments.

Robustness To assess the robustness of the proposed method, we perform an ablation study on the contrastive loss coefficient c. As illustrated in Figure 7, by systematically varying the value of c within the range from 5×10^{-4} to 1.5×10^{-3},

Figure 7: Accuracy of CARFT with different c, based on the SVAMP dataset and with the CodeLlama-7B as the backbone model.

we observe that CARFT consistently outperforms the SFT and ReFT baseline across all tested values. This consistent superiority in performance strongly validates the robustness of CARFT and demonstrates its resilience to changes in the contrastive loss coefficient.

Stability To further enhance the stability of the reinforced fine-tuning process, we propose an embedding-enhanced partial reward method, as described in Section 3.2.4. As shown in Figure 8, this approach effectively improves training stability. The tuning process of CARFT using the embedding-enhanced partial reward method achieves a final accuracy that is 0.5% higher than that of the baseline without the method. Moreover, CARFT with this enhancement exhibits a more stable accuracy improvement curve. CARFT with embedding-enhanced partial reward enabled achieves a peak accuracy of 64.2%, which also corresponds to significant improvements over ReFT, with gains of up to 1.7%.

4.4 Complexity

Simpler methods like SFT require significantly less computational overhead as they do not involve the rollout process used in RL-based methods. However, the performance of SFT is lower compared to RL-based approaches. Our proposed CARFT method achieves the highest performance among all the considered methods.

Let N denote the number of parameters in the LLM, and L represent the length of the on-policy sampled CoT. The computational complexity of a single forward pass in ReFT is $O((L + 1) \cdot N)$, while that of SFT is $O(N)$. Since Dr.GRPO requires multiple rollouts, let G be the number of rollouts. Accordingly, the computational complexity of Dr.GRPO becomes $O((G \cdot L + 1) \cdot N)$. As for CARFT, which incorporates a contrastive signal, let M represent the size of the projector and

Figure 8: Accuracy Curve of CARFT Partial Rewards, based on the SVAMP dataset and with the CodeLlama-7B as the backbone model.

d denote the hidden dimension of the LLM. The computational complexity of CARFT is then given by $O((L + 1) \cdot N + M \cdot d \cdot L)$, where $M \cdot d$ is significantly smaller than N.

The space complexity of SFT is O(N). In contrast, ReFT incurs an additional overhead of O(Ld) due to caching the sequence of on-policy CoT trajectories, resulting in a total space complexity of $O(N + L \cdot d)$. Similarly, Dr.GRPO has a space complexity of $O(N + G \cdot L \cdot d)$. For CARFT, if we denote the projected hidden size as d_2, its space complexity becomes $O(N + L \cdot d + M + L \cdot d_2)$. Here, M is significantly smaller than N, and d_2 (the compressed hidden dimension) is much smaller than the original hidden size d.

5 Conclusions

In this paper, we propose a novel contrastive learning-based framework with annotated CoTs, i.e., CARFT, to enhance the reasoning capabilities of LLMs. We propose generating contrastive signals from both positive and negative CoTs while incorporating annotated CoTs. In order to further improve the stability of the reinforced fine-tuning process, we propose a novel embedding-enhanced partial reward method. Extensive experimental results demonstrate significant advantages of CARFT in terms of performance (up to 10.15%) and efficiency (up to 30.62%). In addition, CARFT corresponds to better stability during reinforced fine-tuning compared with existing approaches.

Limitations

CARFT requires additional computational overhead to compute the embeddings of CoTs, to achieve excellent. As a result, it consumes a longer computational time compared to the ReFT and SFT. However, CARFT needs less computational time than Dr.GRPO as CARFT needs less on-policy sampled CoTs. In addition, CARFT is designed to exploit a centralized annotated CoT dataset. The annotated datasets may be stored in multiple data centers or devices, which may hinder the application of CARFT with decentralized data. In addition, we have restricted the context to fewer than 1024 tokens in this work. We plan to explore improving reasoning in long-context scenarios (Zhu et al., 2024). In addition, while CARFT exploits centralized data, we will investigate decentralized datasets, e.g., federated learning (Liu et al., 2024e,d; Jia et al., 2024; Liu et al., 2024a,c; Che et al., 2023; Liu et al., 2023b, 2022; Zhang et al., 2022; Zhou et al., 2022; Jia et al., 2025; Liu et al., 2025a; Chen et al., 2025; Liu et al., 2024b) and distributed machine learning (Liu et al., 2023a).

References

Anthropic. 2024. Claude 3.7.

Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, and 1 others. 2025. Llama-nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*.

Tianshi Che, Ji Liu, Yang Zhou, Jiaxiang Ren, Jiwen Zhou, Victor S. Sheng, Huaiyu Dai, and Dejing Dou. 2023. Federated learning of large language models with parameter-efficient prompt tuning and adaptive optimization. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7871–7888.

Chunlu Chen, Ji Liu, Haowen Tan, Xingjian Li, Kevin I-Kai Wang, Peng Li, Kouichi Sakurai, and Dejing Dou. 2025. Trustworthy federated learning: privacy, security, and beyond. *Knowledge and Information Systems*, 67(3):2321–2356.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

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

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. 2024. Towards analyzing and understanding the limitations of dpo: A theoretical perspective. *arXiv preprint arXiv:2404.04626*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *AISTATS*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Juncheng Jia, Ji Liu, Chao Huo, Yihui Shen, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2025. Efficient federated learning with timely update dissemination: J. jia et al. *Knowledge and Information Systems*, pages 1–35.

Juncheng Jia, Ji Liu, Chendi Zhou, Hao Tian, Mianxiong Dong, and Dejing Dou. 2024. Efficient asynchronous federated learning with sparsification and quantization. *Concurrency and Computation: Practice and Experience*, 36(9):e8002.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Hoi. 2022. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems*.

Ji Liu, Tianshi Che, Yang Zhou, Ruoming Jin, Huaiyu Dai, Dejing Dou, and Patrick Valduriez. 2024a. Aedfl: efficient asynchronous decentralized federated learning with heterogeneous devices. In *SIAM Int. Conf. on Data Mining (SDM)*, pages 833–841. SIAM.

Ji Liu, Chunlu Chen, Yu Li, Lin Sun, Yulun Song, Jingbo Zhou, Bo Jing, and Dejing Dou. 2024b. Enhancing trust and privacy in distributed networks: a comprehensive survey on blockchain-based federated learning. *Knowledge and Information Systems*, 66(8):4377–4403.

Ji Liu, Juncheng Jia, Tianshi Che, Chao Huo, Jiaxiang Ren, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2024c. Fedasmu: Efficient asynchronous federated learning with dynamic staleness-aware model update. In *AAAI Conference on Artificial Intelligence*, volume 38, pages 13900–13908.

Ji Liu, Juncheng Jia, Beichen Ma, Chendi Zhou, Jingbo Zhou, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2022. Multi-job intelligent scheduling with cross-device federated learning. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 34(2):535–551.

Ji Liu, Juncheng Jia, Hong Zhang, Yuhui Yun, Leye Wang, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2024d. Efficient federated learning using dynamic update and adaptive pruning with momentum on shared server data. *ACM Transactions on Intelligent Systems and Technology*, 15(6):1–28.

Ji Liu, Beichen Ma, Qiaolin Yu, Ruoming Jin, Jingbo Zhou, Yang Zhou, Huaiyu Dai, Haixun Wang, Dejing Dou, and Patrick Valduriez. 2025a. Efficient federated learning with heterogeneous data and adaptive dropout. *ACM Transactions on Knowledge Discovery from Data*.

Ji Liu, Jiaxiang Ren, Ruoming Jin, Zijie Zhang, Yang Zhou, Patrick Valduriez, and Dejing Dou. 2024e. Fisher information-based efficient curriculum federated learning with large language models. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10497–10523.

Ji Liu, Zhihua Wu, Danlei Feng, Minxu Zhang, Xinxuan Wu, Xuefeng Yao, Dianhai Yu, Yanjun Ma, Feng Zhao, and Dejing Dou. 2023a. Heterps: Distributed deep learning with reinforcement learning based scheduling in heterogeneous environments. *Future Generation Computer Systems*, 148:106–117.

Ji Liu, Xuehai Zhou, Lei Mo, Shilei Ji, Yuan Liao, Zheng Li, Qin Gu, and Dejing Dou. 2023b. Distributed and deep vertical federated learning with big data. *Concurrency and Computation: Practice and Experience*, 35(21):e7697.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025b. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

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

OpenAI. 2023. Gpt-3.5 turbo fine-tuning and api updates.

OpenAI. 2025. Introducing openai o3 and o4-mini.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: memory optimizations toward training trillion parameter models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '20. IEEE Press.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, and 7 others. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and P. Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *CoRR*, abs/1506.02438.

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

5932

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

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguera y Arcas, Dale Webster, and 11 others. 2022. Large language models encode clinical knowledge. *arXiv preprint arXiv:2212.13138*.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and characterizing reward hacking. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. In *NeurIPS*.

Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alvaro Bartolome, Alexander M. Rush, and Thomas Wolf. The Alignment Handbook.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl.

Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. 2024. Mathcoder: Seamless code integration in LLMs for enhanced mathematical reasoning. In *The Twelfth International Conference on Learning Representations*.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. 2023. Fingpt: Open-source financial large language models. *FinLLM Symposium at IJCAI 2023*.

Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 1434–1443.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Boyu Zhang, Hongyang Yang, tianyu Zhou, Ali Babar, and Xiao-Yang Liu. 2023a. Enhancing financial sentiment analysis via retrieval augmented large language models. *ACM International Conference on AI in Finance (ICAIF)*.

Hong Zhang, Ji Liu, Juncheng Jia, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2022. Fedduap: Federated learning with dynamic update and adaptive pruning using shared data on the server. In *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2776–2782.

Mengxue Zhang, Zichao Wang, Zhichao Yang, Weiqi Feng, and Andrew Lan. 2023b. Interpretable math word problem solution generation via step-by-step planning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6858–6877, Toronto, Canada. Association for Computational Linguistics.

Chendi Zhou, Ji Liu, Juncheng Jia, Jingbo Zhou, Yang Zhou, Huaiyu Dai, and Dejing Dou. 2022. Efficient device scheduling with multi-job federated learning. In *AAAI Conf. on Artificial Intelligence*, volume 36, pages 9971–9979.

Yuanpin Zhou and Huogen Wang. 2024. Divide and conquer radiology report generation via observation level fine-grained pretraining and prompt tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Wenqiao Zhu, Ji Liu, Lulu Wang, Jun Wu, and Yulun Zhang. 2025. Sgdpo: Self-guided direct preference optimization for language model alignment. In *Findings of the Association for Computational Linguistics: ACL 2025*.

Wenqiao Zhu, Chao Xu, Lulu Wang, and Jun Wu. 2024. Psc: Extending context window of large language models via phase shift calibration. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–13.

Wenqiao Zhu, Yesheng Xu, Xin Huang, Qiyang Min, and Xun Zhou. 2022. Spherical graph embedding for item retrieval in recommendation system. CIKM '22, page 4752–4756.

A Appendix

A.1 Experimental Settings

All experiments are conducted on an ensemble of 8 H100-80GB GPUs. Given that reinforced fine-tuning for reasoning tasks is inherently time-intensive, we utilize FlashAttention (Dao et al., 2022; Dao, 2024) and DeepSpeed Zero stage 3 (Rajbhandari et al., 2020; Rasley et al., 2020) to expedite the fine-tuning process. These technologies enable us to scale up the batch size, thereby enhancing computational efficiency. Additionally, we utilize the HuggingFace Alignment Handbook (Tunstall et al.) and the TRL library (von Werra et al., 2020) as methodological guides to streamline the fine-tuning implementation.

To ensure consistency and comparability across experiments, we adopt a structured hyperparameter configuration strategy. During the warmup phase, we initialize training with a batch size of 64 and a learning rate of 1e-5. This learning rate is then adjusted to 3e-7 during the reinforcement fine-tuning stage to stabilize the optimization process. We maintain a batch size of 64 for all models on the SVAMP dataset. On GSM8K, we tailor the batch size to each model's computational characteristics: 64 for Qwen2.5-7B-Instruct and 96 for CodeLlama-7B, balancing memory efficiency and training throughput. In reinforcement learning components, we set the KL divergence coefficient to 0.05 to regulate policy updates and employ a temperature parameter (τ) of 0.2 and $c = 1e - 3$ in the contrastive learning loss to control embedding similarity. We set the dimension of the projected embedding to 64. For PPO optimization, we configure $\lambda = 1$, $\gamma = 0.95$, $\alpha = 5$, $\epsilon = 0.2$, and $U = 2$. For Dr.GRPO, to ensure a fair comparison, we set the parameter G in Dr.GRPO to 2, which matches the maximum number of CoTs in CARFT at each step. We set the reward $r(x, y)$ to 1 if the answer is correct, and 0 otherwise. We also adopt a partial reward scheme (Le et al., 2022), setting the reward to 0.1 in cases where a numerical answer can be extracted but is incorrect.

Training epoch limits are determined based on empirical convergence behavior. For the SFT baseline, we cap training at 60 epochs due to its tendency to be unstable; beyond this point, additional epochs yield diminishing returns. To ensure a fair comparison across methods, for ReFT, CARFT, and Dr.GRPO, we fine-tune the base model for 4 epochs and select the best checkpoint for rein-

Batch Size	Accuracy
64	**51.48%**
96	50.95%

Table 5: CARFT with different batch size, based on CodeLlama-7B model and GSM8K dataset.

Model	Model	Accuracy
Qwen2.5-Instruct-14B	SFT	86.7%
Qwen2.5-Instruct-14B	CARFT	88.9%

Table 6: Accuracy curves of various methods on SVAMP dataset and Qwen2.5-14B-Instruct backbone.

forced fine-tuning. All of these approaches are then trained for 70 epochs, allowing sufficient iterations for convergence while maintaining experimental rigor.

A.2 More Experiments

Batch Size In our experiments, we utilized FlashAttention (Dao et al., 2022; Dao, 2024) and DeepSpeed Zero stage 3 (Rajbhandari et al., 2020; Rasley et al., 2020) to accelerate the fine-tuning process with a large batch size. To evaluate how batch size affects model performance, we conducted a systematic ablation study. As shown in Table 5, increasing the batch size can degrade the performance of large language models (LLMs). Specifically, enlarging the batch size from 64 to 96 led to a drop in accuracy from 51.48% to 50.95%.

This suggests that reducing the batch size may be a viable strategy for achieving better performance. It is worth emphasizing that all experiments were carried out with consistent batch size configurations to ensure a fair and valid comparison.

Larger Model In addition to the two 7B models, we conducted experiments on Qwen 2.5 14B, as well. The models were fine-tuned for 100 epochs, and the results are summarized in Table 6. These results highlight the effectiveness and strong generalization of CARFT across diverse model scales and datasets.

Explains of the Accuracy Curve Figure 9 indicates that ReFT also suffers from model collapse, which yields poor results. CARFT shows strong stability across the whole fine-tuning process and outperforms both SFT and ReFT significantly.
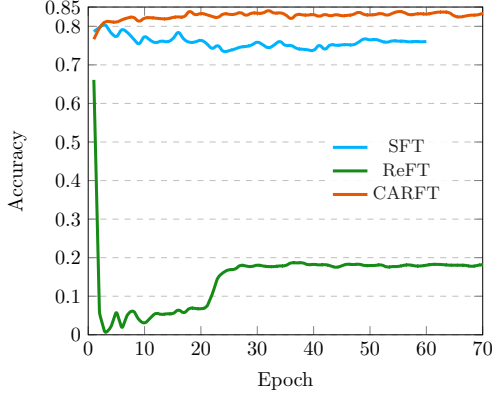
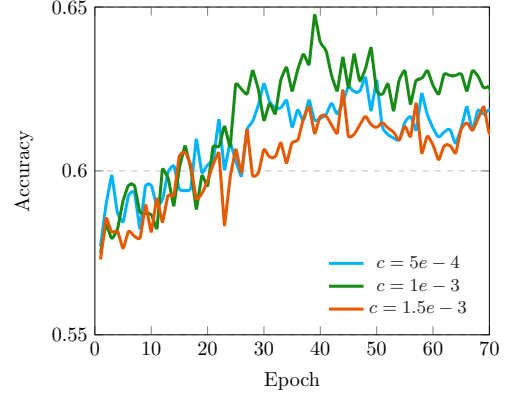Figure 9: Accuracy curves of various methods on GSM8K dataset and Qwen2.5-7B-Instruct backbone.

Figure 11: Accuracy Curve of CARFT with different c, based on the SVAMP dataset and with the CodeLlama-7B as the backbone model.
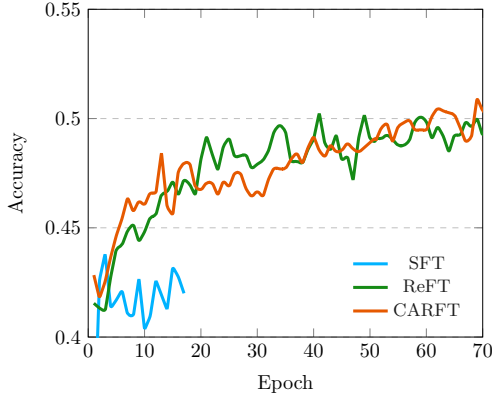

Figure 10: Accuracy curves of various methods on GSM8k dataset and CodeLlama-7B backbone.

Figure 12: RL loss curve for CARFT with the GSM8K dataset and CodeLlama-7B serving as the backbone model.

Figure 10 presents the accuracy of various methods on the GSM8K dataset using the CodeLlama-7B model as the backbone. We observed that further training did not lead to performance improvements in the SFT (Supervised Fine-Tuning) phase, so we terminated the training early. The figure also demonstrates that CARFT outperforms ReFT with a higher convergency accuracy.

Figure 11 illustrates the accuracy curves of CARFT under different values of parameter c. It can be observed that CARFT attains the optimal performance when $c = 1e - 3$.

Loss We present the RL (Reinforcement Learning) and contrastive learning loss curves for CARFT and ReFT models in Figures 12, 13, 14, 15, 16, and 17. These results are based on the GSM8K dataset and utilize the CodeLlama-7B and Qwen2.5-7B-Instruct base models. We make the
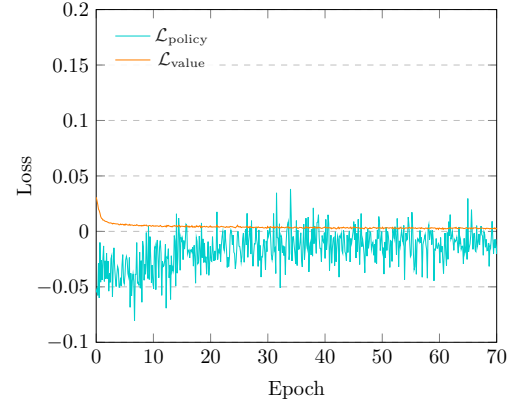
following observations: (1) As shown in Figure 12 and Figure 14, when using CodeLlama-7B as the base model, CARFT and ReFT exhibit similar loss curves. (2) In contrast, when Qwen2.5-7B-Instruct is used as the base model, the loss curves of CARFT and ReFT differ significantly. In particular, ReFT displays a fluctuating pattern, suggesting instability during fine-tuning. (3) Furthermore, as seen in Figure 13 and Figure 16, the contrastive loss varies across models, indicating differences in the CoT embedding spaces learned by each model.

A.3 CARFT with Negative Signal

We describe the CARFT framework with negative signal in Algorithm 2. For each pair $(\mathbf{c}, \hat{\mathbf{c}})$, where $\mathbf{c}$ denotes the annotated CoT and $\hat{\mathbf{c}}$ represents the on-policy sampled CoT, we first compute the longest

Algorithm 2: CARFT with Negative Signal

Input : Tuples of (*question, CoT, answer*): $\mathcal{D}_{\text{train}} = \{(\mathbf{x}, \mathbf{c}, \mathbf{y}\}$, Number of RL steps: T, Number of updates per RL step: U, Initial policy: π_θ^0.

Output : Final Policy: π_θ

1 **for** $i \leftarrow 1$ **to** T **do**
2     $\mathbf{x}, \mathbf{c}, \mathbf{y} \sim \mathcal{D}_{\text{train}}$ // Sample training data from $\mathcal{D}_{\text{train}}$
3     $\hat{\mathbf{c}} \sim \pi_\theta$ // On-policy CoT sampling
4     $\hat{\mathbf{y}} \leftarrow \text{EXTRACT}(\hat{\mathbf{c}})$ // Extract answer
5     $\mathbf{e}_{\text{LCS}}^{\text{annotated}}, \mathbf{e}_{\text{exc}}^{\text{annotated}} \leftarrow \mathbf{c}, \quad \hat{\mathbf{e}}_{\text{LCS}}^{\text{rollout}}, \hat{\mathbf{e}}_{\text{exc}}^{\text{rollout}} \leftarrow \hat{\mathbf{c}}$ // Construct CoT Embeddings
6   Compute $\sigma_t, \hat{A}_t, \hat{R}_t, \mathcal{M}_2$
7   **for** $i \leftarrow 1$ **to** U **do**
8        $\theta, \phi \leftarrow \text{OPTIMIZATION\_STEP}(\mathcal{L})$ // Equation 9
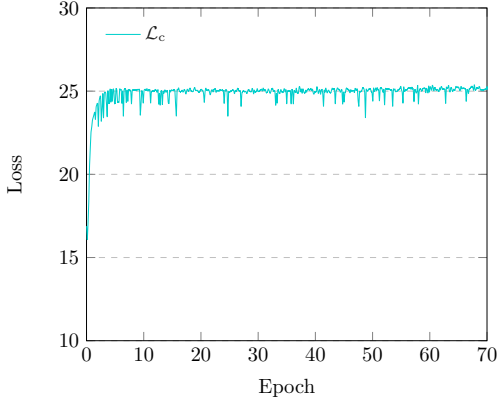9   **end**
10 **end**
11 **return** π_θ

Figure 13: Constrastive loss curve for CARFT with the GSM8K dataset and CodeLlama-7B serving as the backbone model.
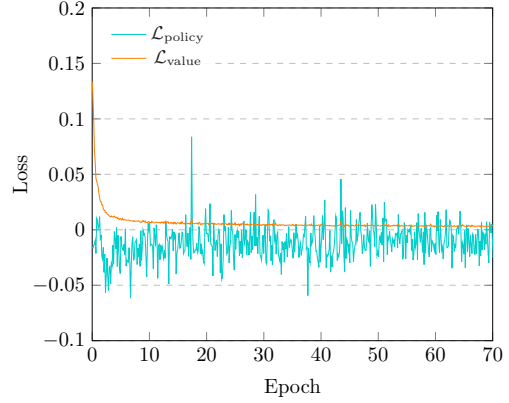

Figure 14: RL loss curve for ReFT with the GSM8K dataset and CodeLlama-7B serving as the backbone model.

common subsequence (LCS) between the two sequences. Using the LCS tokens, we construct corresponding LCS embeddings for both sequences, resulting in $\mathbf{e}_{\text{LCS}}^{\text{annotated}}$ and $\hat{\mathbf{e}}_{\text{LCS}}^{\text{rollout}}$, respectively. The remaining tokens—that is, those not included in the LCS—are used to generate two additional embeddings: $\mathbf{e}_{\text{exc}}^{\text{annotated}}$ and $\hat{\mathbf{e}}_{\text{exc}}^{\text{rollout}}$.

The masked InfoNCE loss then leverages $\mathbf{e}_{\text{LCS}}^{\text{rollout}}$, $\mathbf{e}_{\text{exc}}^{\text{annotated}}$, and $\hat{\mathbf{e}}_{\text{exc}}^{\text{rollout}}$ to provide feedback for training.

A.3.1 Illustration

We provide an example to illustrate how negative signals are constructed. Let

$$A = [a_1, a_2, \ldots, a_n]$$

denote the annotated CoT that leads to a correct solution. Let

$$B = [b_1, b_2, \ldots, b_m]$$

represent the on-policy sampled CoT that results in an incorrect solution.

Step 1: Compute Longest Common Subsequence (LCS) We first compute the LCS of A and B, denoted as:

$$C = [c_1, c_2, \ldots, c_k]$$

Step 2: Extract Sub-sequences Next, we compute the embeddings of the following components:
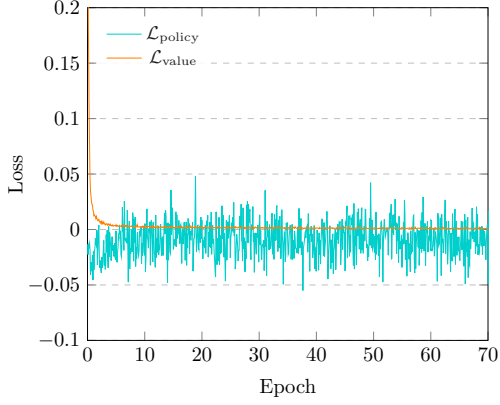
- C: the common subsequence,

Figure 15: RL loss curve for CARFT with the GSM8K dataset and Qwen2.5-7B-Instruct serving as the backbone model.
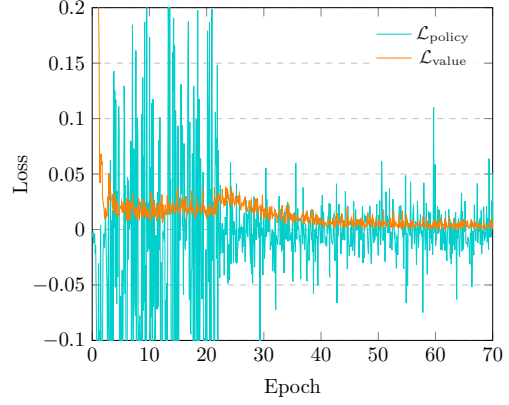

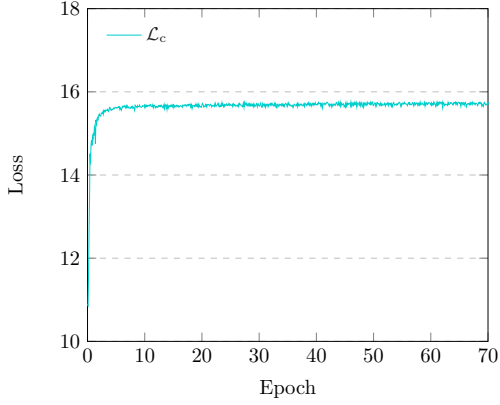Figure 17: RL loss curve for ReFT with the GSM8K dataset and Qwen2.5-7B-Instruct serving as the backbone model.

$M \to \infty$, the constrative loss converges to

$$
\lim_{M \to \infty} \mathcal{L}_{\text{contrastive}}(f; \tau, M) - \log M
$$
$$
= -\frac{1}{\tau} \mathbb{E}_{(x,y) \sim p_{\text{pos}}}[f(x)^\top f(y)]
$$
$$
+ \mathbb{E}_{x \sim p_{\text{data}}}\big[\log \mathbb{E}_{x^- \sim p_{\text{data}}}[e^{f(x^-)^\top f(x)/\tau}]\big]. \quad (11)
$$

Hence, the contrastive learning signal can be decomposed into two components: the *alignment term* and the *uniformity term*. The *alignment term* minimizes the distance between embeddings of positive pairs, while the *uniformity term* encourages negative embeddings to be more uniformly distributed. In CARFT, the contrastive learning signal helps align the on-policy sampled CoT distribution with the annotated CoT. This effectively leverages the information contained in the annotated CoT and enhances training stability.

Figure 16: Contrastive loss curve for CARFT with the GSM8K dataset and Qwen2.5-7B-Instruct serving as the backbone model.

- $A \setminus C$: parts of the correct CoT not in C,

- $B \setminus C$: parts of the incorrect CoT not in C.

Step 3: Apply Contrastive Signal We minimize the contrastive loss to generate the contrastive signal through backpropagation. The motivation behind the negative signal is to align the embedding of $A \setminus C$ with that of C, while increasing the distance between the embeddings of $B \setminus C$ and C, since $B \setminus C$ leads to an incorrect solution. As noted in (Wang and Isola, 2020), contrastive learning has the net effect of pulling positive pairs together while scattering negative examples apart.

B Asymptotics of $\mathcal{L}_{\textbf{contrastive}}$

According to Theorem 1 in (Wang and Isola, 2020), for fixed $\tau > 0$, as the number of negative samples