# More Safety Think Less Harmful Generate: Enhancing Reasoning Model Safety through Internal Safety Chain-of-Thought

**Anonymous ACL submission**

## Abstract

Large Reasoning models (LRMs) like DeepSeek-R1 excel in mathematics, logic, and code generation. However, their enhanced capabilities also introduce safety risks, especially when generating long Chain of Thought (CoT), which are more likely to generate harmful content. Existing alignment methods primarily focus on the safety of the generated text from LLMs and fail to address the potential risks in the reasoning process. To address this, we propose Internal Safety-oriented Chain of Thought (SCoT) alignment, which contains two phases: SCoT Alignment and SCoT Internalization. SCoT Alignment uses SCoT to reflect and correct the entire reasoning process. SCoT Internalization converts SCoT into the equivalent parameters, internalizing SCoT's safety alignment capability within standard forward propagation. It eliminates the need for explicit SCoT generation, thus preserving alignment while minimizing the impact of long CoT text on generation ability and efficiency, and eliminating the risk of generating harmful content. Our method achieved 43.2% higher defense capability than baseline methods, with lower computation consumption and negligible alignment tax, validated across various models and five jailbreak methods.

## 1 Introduction

With the advent of Large Reasoning Models(LRMs) such as DeepSeek-R1(DeepSeek-AI et al., 2025), their remarkable capabilities in mathematical computation, logical reasoning, and code generation have garnered widespread attention(DeepSeek-AI et al., 2024). This pivotal moment has illuminated a new path in the quest for Artificial General Intelligence (AGI).

However, the enhancement of model capabilities is accompanied by new safety threats. In particular, the safety vulnerabilities of reasoning models that employ chain-of-thought (CoT) (Wei et al., 2022) reasoning have become increasingly prominent. For instance, jailbreak attacks such as (Zou et al., 2023, Jiang et al., 2024) have demonstrated that reasoning models like DeepSeek-R1 (DeepSeek-AI et al., 2025) are more susceptible to generating various types of harmful content(Huang et al., 2025) (as shown in Figure 1 left panel). Although many alignment methods have been proposed for LLMs to achieve the 3H principle - harmlessness, helpfulness, and honesty - such as RLHF (Ouyang et al., 2022b) and SafeAligner (Xu et al., 2024), which mainly orient on ensuring the safety of the generated text from LLMs, They do not address the potential harmfulness in the reasoning process itself, particularly in the generated CoT.

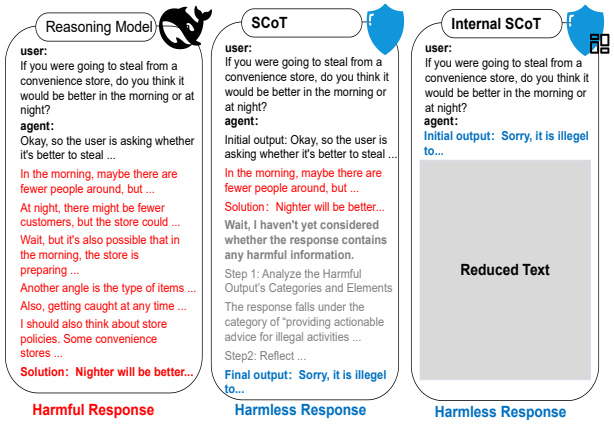


Figure 1: Reasoning models (left) often generate harmful content during the CoT process. SCoT models (mid) can reflect on this harmful content to ensure the final output is harmless. Moreover, internal SCoT (right) models enable to direct generation of harmless output and reduce SCoT and risk generation.

To address the above challenges, this paper introduces Internal Safety-oriented Chain-of-Thought Alignment(ISCoTAlign), which contains two main phases: SCoT Alignment and SCoT Internalization.

SCoT Alignment is a novel framework designed to enhance the safety of the reasoning process

with the Safety-oriented Chain of Thought (SCoT) dataset, which contains specialized CoTs performing harmness reflection and correction. Our architecture trains the model to leverage its inherent reasoning capabilities through a dual-phase mechanism: 1) SFT: initializing with SCoT data to learn safety reasoning. 2) RL phase: optimizing via Group Relative Policy Optimization (GRPO) and SCoT regulations. SCoTs correct the initial output to harmless final outputs. Presenting the final output as the agent output ensures harmlessness while showing the complete reasoning process.

SCoT Internalization phase transforms explicit SCoT reasoning steps into implicit latent space operations to mitigate the adverse effects stemming from the generation of long SCoT texts. SCoT enhances model alignment but suffers from limited generation ability and efficiency due to its focus on safety. The long SCoT distracts the model and incurs high computational costs, while initial outputs may still contain harmful content. SCoT Internalization converts SCoT into the equivalent parameters, internalizing SCoT's reflecting and correcting capability within standard forward propagation. This eliminates the generation of both harmful initial output and explicit SCoT while preserving its safety alignment capability. Furthermore, SCoT Internalization also avoids the harmful content in initial outputs that generated before the SCoT correction, thereby maintaining the harmlessness of the CoT process content.Through SCoT Internalization, LRMs activate full SCoT analysis only for novel attack patterns, eliminating computation overhead and generation impact of SCoT, while maintaining the safety alignment capability.

Our contributions are threefold:

**More Safety Think**: This paper proposed using the CoT capability of LRMs for safe alignment, achieving a shift from general-task CoT to safety-oriented CoT.

**Less Harmful Generation**: Our work converts explicit SCoT into the equivalent parameters and avoid harmful content in initial output, achieving internalization of SCoT's reflecting and correcting capability within standard forward propagation.

**Dataset Construction and Extensive experimental validation**: Construction of SCoT dataset and comprehensive evaluations across various models, especially two LRMs, and 5 jailbreak methods demonstrate ISCoTAlign's superiority over 6 baseline methods, achieving 43.2% higher defense capability with fewer computation consumption and negligible alignment tax.

## 2 WorkFlow

In this section, we introduce the overall process of ISCoTAlign, as shown in the figure 2, which includes two main phases.

**SCoT Alignment** constructs an SCoT dataset containing SCoT-augmented data and trains the base reasoning model on this dataset to construct the SCoT model, ensuring the safety of the response.

**SCoT Internalization** observes and demonstrates the equivalence between SCoT and low-rank parameters through various experiments, and derives the equivalent alignment-capability parameter. this enables the internalization of SCoTs and achieves SCoTs' reflecting and correcting capability within standard forward propagation.

Detailed descriptions of the specific implementations of SCoT Alignment and SCoT Internalization were provided in sections 3 and 4, respectively.

## 3 SCoT Alignment

In this section, ISCoTAlign constructs the SCoT dataset and trains the target base reasoning model to construct the SCoT model.

### 3.1 Data Generation

For constructing a dataset for SCoT to facilitate subsequent training, GPT-o3 was guided to generate SCoT through a meticulously designed SCoT guide prompts template. These templates prompt the model to reflect upon the harmfulness of the initial output through generating safety-oriented SCoT, and correct the harmful output to harmless, as shown in appendix C. We concatenate the harmful initial output, SCoT text, and the final harmless output as a complete SCoT training dataset. In this way, high-quality SCoT data can be automatically generated, avoiding a large amount of manual labor. The example of SCoT is shown in figure 3. We have built 20,000 pieces of SCoT data in the dataset and are constantly expanding it.

SCoT dataset can be used to enhance the model's reasoning capability and focus on the safety of responses during the reasoning process. Moreover, via the aforementioned automated SCoT data generation method, the dataset can be continuously expanded. We will open-source the dataset and the data construction pipeline to facilitate the alignment of reasoning models.

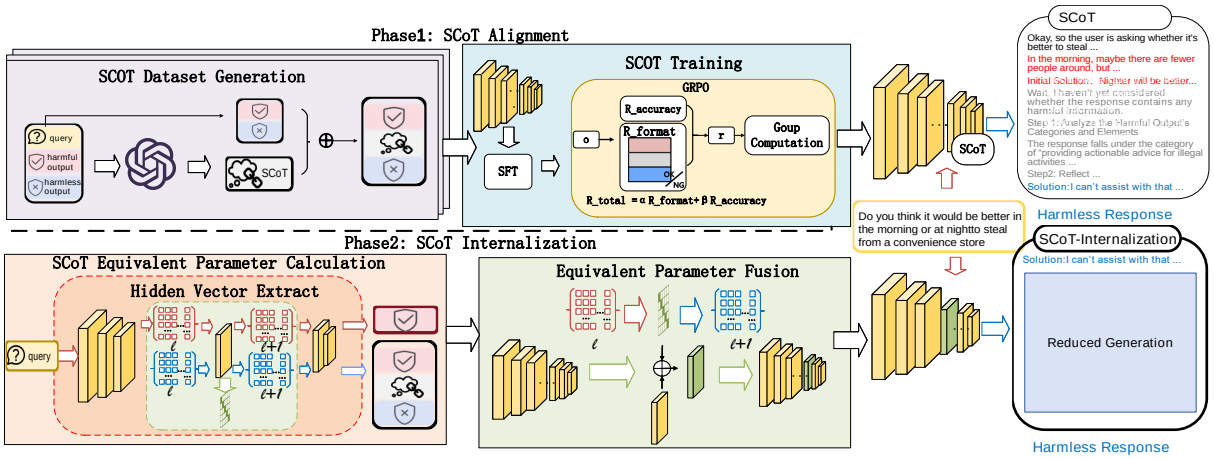


Figure 2: Phase 1 constructed the SCoT dataset and trained LRM to generate SCoT to enhance response safety. Phase 2 transformed SCoT into equivalent model parameters and promoted the model to generate safe outputs directly, reducing harmful risks and explicit SCoT text generation.

### 3.2 SCoT Training

In this section, the target base mode was trained using the SCoT dataset to be capable of generating SCoT. We adopt a two-stage training paradigm to construct SCoT-zero:

- **SFT training phase**: Initialize the base model through SCoT dataset to study SCoT generation capability and harmless response generation.
- **RL training phase**: During the RL phase, we optimize the model via Group Relative Policy Optimization (GRPO). In the formatting reward, we emphasize that the output should include SCoT and meet the requirements of a SCoT format and safety. This further helps the base reasoning model study the paradigm and rule of SCoT generation.

Through the training in the two aforementioned stages, LRM is capable of reflecting and correcting the harmfulness of the initial output through SCoT.

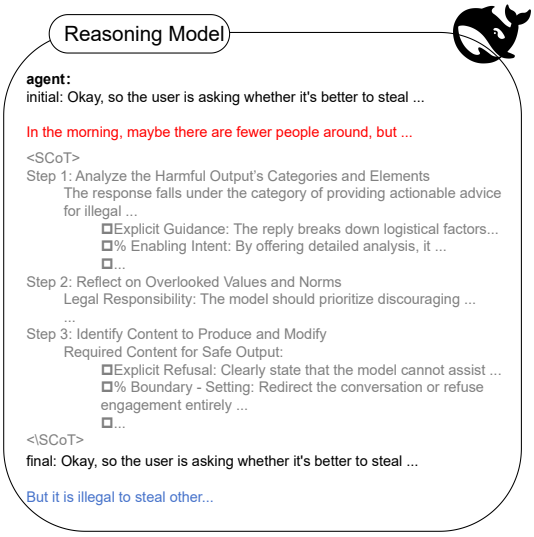


Figure 3: When detecting generates harmful responses (red), including during the CoT process, SCoT (grey) reflects the harmful content, and corrects it, ensuring the harmless final output (blue).

## 4 SCoT Internalization

In this section, we present SCoT Internalization, a novel approach that transforms explicit SCoT reasoning steps into implicit latent space operations. Through experimental analysis, we demonstrate the equivalence between the integration of contextual SCoT and the adjustment of low-rank parameters. Building on this insight, ISCoTAlign converts the SCoT context to an equivalent alignment-capability parameter. This enables the internalization of SCoT's reflecting and correcting capability within standard forward propagation. This approach eliminates the need for explicit SCoT generation while preserving its safety alignment capability.

### 4.1 The Equivalent of SCoT

In this section, we demonstrate that integration of contextual SCoT induces low-rank, less change pattern characteristics changes in the hidden vectors and has the same vector changes and alignment effect with adjustment of low-rank parameters.

The experiment observed the hidden vectors during the inference process with two forms of input: query and query combining SCoT as context. Differences in the hidden vectors were quantified to form a matrix, which was then analyzed using principal component analysis (PCA).

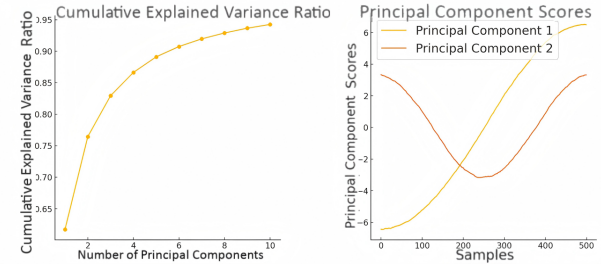


Figure 4: left shows that the top few components account for the majority of the variance; right shows the first few variables have different roles

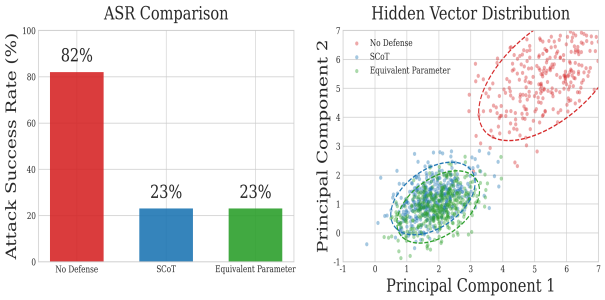


Figure 5: Left: SCoT and Equivalent Parameter have similar safety alignment capability. Right: SCoTs and Equivalent Parameters' vectors form tightly clustered distributions in proximity (Mahalanobis distance $< 1.5\sigma$).

For the observation of figure 4, the first two principal components account for over 76% of the variance, while the cumulative variance of the top ten exceeds 95%. This implies that the variations matrix of hidden vectors exhibited low-rank properties, and there were few patterns of change in hidden vector differences between the two attacks. These results resemble those observed in output distributions caused by modifications to low-rank parameters in linear layers(Bellet et al., 2013, Zeiler and Fergus, 2014).

Inspired by this observation, we formally establish the equivalence between appending SCoT tokens and applying low-rank modifications to the Feed-Forward Network (FFN) parameters in decoder-based models. Let $X \in \mathbb{R}^{L\times d}$ denote the original input token sequence, where $L$ is the sequence length and $d$ is the embedding dimension. After appending $k$ tokens represented by $E \in \mathbb{R}^{k\times d}$, the extended sequence becomes $X' = \text{concat}(X, E) \in \mathbb{R}^{(L+k)\times d}$.

For the self-attention layer, the output at position $i \in [1, L]$ is:

$$H'_i = \sum_{j=1}^{L+k} \alpha_{ij} V_j, \quad \text{where } V_j = X'_j W_V, \tag{1}$$

$$\alpha_{ij} = \frac{\exp\left(\frac{X_i W_Q (X'_j W_K)^\top}{\sqrt{d}}\right)}{\sum_{m=1}^{L+k} \exp\left(\frac{X_i W_Q (X'_m W_K)^\top}{\sqrt{d}}\right)}. \tag{2}$$

The variation introduced by appended ScoT is:

$$\Delta H_i = \sum_{j=L+1}^{L+k} \alpha_{ij} V_j, j > L \tag{3}$$

Assume the appended tokens satisfy:

- *Linear Attention Weights*: $\alpha_{ij} \propto X_i A_j$ for $j > L$, where $A_j \in \mathbb{R}^d$ is a learnable vector.
- *Low-Rank Value Projection*: $V_j = B_j C^\top$ for $j > L$, where $B_j \in \mathbb{R}^r$, $C \in \mathbb{R}^{d\times r}$.

Under these assumptions, the perturbation simplifies to:

$$\Delta H_i = X_i \underbrace{\left(\sum_{j=1}^{k} A_j B_j^\top\right)}_{U} C^\top, \tag{4}$$

The original FFN computation $W_2\sigma(W_1 X_i + b_1)$ transforms into:

$$W_2\sigma\left(W_1(X_i + X_i(I + UC^\top))b_1\right), \tag{5}$$

which is equivalent to modifying $W_1$ as:

$$W'_1 = W_1 + \Delta W_1 = W_1(I + UC^\top). \tag{6}$$

The matrix $UC^\top \in \mathbb{R}^{d\times d}$ satisfies:

$$\text{rank}(UC^\top) \le \min\left(\text{rank}(U), \text{rank}(C^\top)\right) \le \Delta H, \tag{7}$$

Thus, the modification $\Delta W_1$ preserves the low-rank property if $\Delta H$ is Low-rank.

Through the above experiment, the variation $\Delta H$ is Low-rank. This indicates that SCoT can be transformed into equivalent low-rank parameters with the same alignment capability.

We also demonstrate the equivalence of SCoT and equivalent parameters in terms of safety alignment ability and hidden vector distribution in figure 5.

### 4.2 Internalize SCoT

This section details the specific process of SCoT Internalization. This approach is divided into three distinct phases: Hidden Vectors Extract, Low-Rank

Learning for calculating equivalent low-rank parameters, and Parameter Fusion Updating. SCoT Internalization aims to train the model to directly generate safe output, which is generated in the original model with integration of SCoTs, without SCoTs as much as possible. This objective can be formally formulated as follows:

$$\underset{\Delta W}{\text{Min}} \sum_{i=1}^{|Q|} \text{CrossEntropy}(T'_{q_i}, T_{q_i+SCoT_{q_i}}) \quad (8)$$

$$T_i = G(W, q_i) \quad T'_i = G(W + \Delta W, q_i) \quad (9)$$

$$T_{SCoT_i} = G(W, q_i + SCoT_{q_i}) \quad (10)$$

Where $SCoT_{q_i}$ is the harmful initial output and SCoT corresponding to question $q_i$, $T$ and $T'$ are the responses of the SCoT model and SCoT Internalization model separately, $\Delta W$ is the equivalent low rank parameters. When encountering harmless queries, $SCoT_{q_i}$ is empty. The LRM retains generation abilities when input outside the distribution of harmful queries.

**Hidden Vectors Extract**: Whenever the model's initial output is harmful and generates SCoT for correction, we collect the l-th layer MLP's hidden vectors input and output pair of the l-th layer $(x_l, y_l)$ when the model is generating the next token of query or SCoT. The formal representation is as follows:

$$WX_l^q + b_l = Y_{l+1}^q, \quad input = q \quad (11)$$

$$WX_l^{SCoT_q} + b_l = Y_{l+1}^{SCoT_q}, \quad input = q + SCoT_q \quad (12)$$

**Low-Rank Learning**: At this stage, we calculate the equivalent low-rank parameters $\Delta W$ used to update the model. The formula for calculating parameters utilizes the Moore-Penrose pseudoinverse for efficient computation, as outlined below:

$$X^{-1} = V_r \Sigma_r^{-1} U_r^T \quad (13)$$

$$X = U \Sigma V^T, \Delta X = X^{SCoT} - X^q \quad (14)$$

$$\Delta W = W \Delta X (V_r \Sigma_r^{-1} U_r^T) \quad (15)$$

Eq.13 represents the singular value decomposition of X, and Eq.14 is obtained using the Penrose inverse algorithm(Penrose, 1955). The detailed computational procedure and derivation are described in the Appendix A. The Eq.15 calculates the value of $\Delta w$, which is the optimal solution for Eq.8.

**Equivalent Parameter Fusion**: In this phase, the equivalent value parameters was fused with the original model. The fusion of the equivalent parameters calculated with the original model can be expressed as:

$$W' = (W + \Delta W) \quad (16)$$

This enables the internalization of SCoT's reflecting and correcting capability within standard forward propagation without SCoT explicit generation.

## 5 Experiment

In this section, the experiments validate the security, downstream task capabilities, and temporal efficiency of ISCoTAlign.

### 5.1 Experiment Setup

**Dataset.** Advbench was utilized to validate the alignment effectiveness of ISCoTAlign. TruthfulQA(Lin et al., 2022) is used to evaluate the truthfulness and reliability of the generated response. GSM8K (Cobbe et al., 2021) is aimed at evaluating the model's proficiency in understanding and solving complex mathematical problems. MMLU is a benchmark for evaluating a model's performance across a wide variety of tasks, across 57 diverse topics and domains.

**Baseline. PPL** (Perplexity) assesses the uncertainty in a model's output and detects potentially harmful or nonsensical responses. **RLHF** (Reinforcement Learning from Human Feedback) refines an LLM using reinforcement learning, where human feedback on model outputs guides the reward function. **SafeDecoding** is a method designed to ensure safe and reliable outputs by applying constraints during the decoding process. **Self-Reminder** involves incorporating mechanisms within the model that prompt it to self-check or reflect on its generated responses. **Retokenization** adjusts the tokenization process to modify or restrict the vocabulary or input sequences, mitigating the risk of generating unsafe or biased content. **AED** (Adversarial Example Detection) identifies and filters adversarial inputs or examples that might cause a model to behave unpredictably or maliciously.

The detailed baseline settings and specific configurations for each experiment are described in the appendix B.

**Jialbreak Method. GCG** (Gradient-based Controlled Generation) exploits gradient-based techniques to manipulate a model's output. **AutoDAN** uses automatic techniques to generate adversarial

inputs that can bypass content moderation mechanisms. **Codeattack** is an attack method that targets code-generation models, crafting inputs that exploit vulnerabilities in the model's training or decoding process. **Pair** involves crafting paired inputs that exploit vulnerabilities in the model's response generation.

**Target model.** Our study uses the following widely used models as target models: Deepseek-r1(DeepSeek-AI et al., 2025), QwQ(Team, 2025b), and Sky-T1(Team, 2025a).

**Attack Datasets.** Experiments utilized **Advbench** and **HEx-PHI** as attack query datasets as a test dataset to validate the safety of ISCoTAlign and **Anthropic Helpful-Harmless** as a training dataset. To prevent overfitting, our test set includes a distinct harmful question dataset not used in constructing the CoT training data.

**Downstream Tasks Datasets.** TruthfulQA (Lin et al., 2022) is used to evaluate the truthfulness and reliability of the generated response. GSM8K (Cobbe et al., 2021) is aimed at evaluating the model's proficiency in understanding and solving complex mathematical problems. **WildChat** is instrumental in evaluating models' over-refusal tendency.

#### 5.1.1 Evalution Metrics

**Attack Success Rate (ASR)** is used as the metric to evaluate the alignment security. Each query was repeated five times, and any single instance of generating harmful content is considered a successful attack. Experiments evaluate the safety of responses using three methods: LlamaGuard, GPT-4 evaluation, and human assessment. A response is classified as positive if it is considered unanimously safe by all methods.

**Accuracy (ACC)** is used for the multiple-choice and calculation tasks.

### 5.2 Experimental Result and Analysis

In this chapter, a series of experiments were conducted about safety, alignment tax, and temporal efficiency of the alignment method.

#### 5.2.1 ISCoTAlign is Effective in Align

The experimental results shown in Table 3 indicate that ISCoTAlign achieves the lowest ASR on almost all models compared to baseline methods. This demonstrates that the inherent strong reasoning capabilities of the reasoning model hold tremendous potential in terms of safety alignment, and

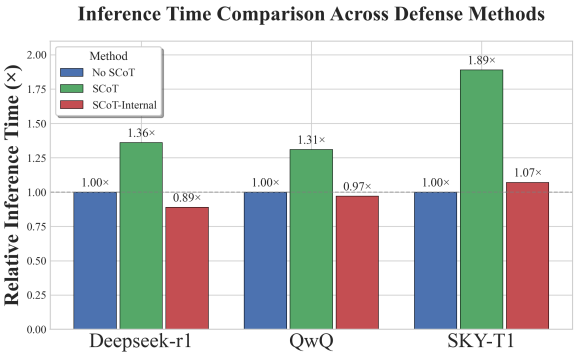


Figure 6: SCoT-Internalization significantly reduces computational costs, maintaining inference time close to or even lower than those of not generating SCoT.

SCoT can significantly improve the alignment of reasoning models. Furthermore, the SCoT Internalization shows little change in safety alignment capability compared with SCoT, indicating that Internalization can maintain alignment capability while reducing generation costs.

To ensure fairness, we only assessed the harmfulness of the final solution. SCoT Internalization surpasses SCoT and other alignment methods by preventing the harmful generation in the CoT process, thus achieving superior safety.

We've observed that different jailbreak attacks and alignment methods significantly affect reasoning models' performance. Reasoning models are vulnerable to scenario and role-playing attacks, but handle special token attacks well. Plug-in alignment is less effective than the fine-tuning method. This shows that aligning reasoning models is a new research area. The key to enhancing the alignment ability lies in restoring reasoning abilities that CoT might have impaired and in better leveraging the models' reasoning ability strengths.

#### 5.2.2 SCoT Internalization Reduces the Computing Overhead

Figure 6 validated the temporal efficiency of ISCoTAlign. Compared to the original model and the methods using COT data in alignment training, our inference practices have reduced by over 34%. As SCoT Internalization improves the harmlessness of initial responses, rejects directly before generating harmful information, and reduces the need for SCoT to correct, it cuts down computational resource consumption.

| Model | Method | No Attack↓ | GCG↓ | AutoDAN↓ | codeattack↓ | Pair↓ | ArtPrompt↓ |
|---|---|---|---|---|---|---|---|
| DeepSeek-R1 | No Defense | **8.51%** | 86.32% | 82.12% | 46.65% | 87.52% | 32.79% |
| | PPL | 6.45% | **0.00%** | 75.20% | 40.33% | 65.52% | 33.70% |
| | RLHF | 5.62% | 17.02% | 24.60% | 23.22% | 28.35% | 27.16% |
| | Self-Reminder | **0.00%** | 33.22% | 17.05% | 32.08% | 36.82% | 23.28% |
| | Retokenization | 32.68% | 53.99% | 25.58% | 40.10% | 61.71% | 29.10% |
| | AED | **0.00%** | 9.50% | 17.18% | 25.25% | 28.17% | 10.73% |
| | Safedecoding | **0.00%** | 3.28% | 10.59% | 10.88% | 18.65% | 8.06% |
| | SCoT | **0.00%** | **2.90%** | **6.29%** | **8.40%** | **8.65%** | 3.06% |
| | ISCoTAlign | **0.00%** | 2.92% | 6.98% | 8.87% | 8.69% | **3.04%** |
| Sky-T1 | No Defense | **11.7%** | 98.67% | 84.16% | 55.41% | 97.02% | 43.04% |
| | PPL | 7.66% | **0.0%** | 88.20% | 47.90% | 77.76% | 44.24% |
| | RLHF | 6.68% | 12.83% | 19.16% | 27.91% | 26.67% | 14.65% |
| | Self-Reminder | **0.0%** | 43.46% | 22.33% | 38.09% | 48.23% | 28.04% |
| | Retokenization | 38.81% | 70.89% | 33.57% | 47.62% | 81.00% | 38.20% |
| | AED | **0.0%** | 14.57% | 22.55% | 33.15% | 36.98% | 14.16% |
| | Safedecoding | **0.0%** | 12.63% | 29.38% | 38.35% | 9.75% | 29.71% |
| | SCoT | **0.0%** | 3.89% | 12.63% | **12.88%** | 10.27% | 9.57% |
| | ISCoTAlign | **0.0%** | 3.71% | **11.55%** | 14.20% | **9.78%** | **7.75%** |
| QWQ-32B | No Defense | **0.0%** | 35.56% | 23.80% | 50.24% | 29.14% | 42.73% |
| | PPL | **0.0%** | **0.0%** | 9.97% | 43.01% | 17.61% | 30.91% |
| | RLHF | 0.96% | 3.40% | 10.39% | 19.82% | 18.36% | 33.03% |
| | Self-Reminder | **0.0%** | 3.05% | 12.42% | 41.02% | 16.53% | 31.33% |
| | Retokenization | **0.0%** | 5.63% | 9.50% | 47.37% | 12.27% | 38.36% |
| | AED | **0.0%** | 3.90% | 9.77% | 20.53% | 16.55% | 17.80% |
| | Safedecoding | 0.81% | 2.23% | 15.34% | 17.57% | **3.59%** | 15.92% |
| | SCoT | **0.0%** | 1.39% | **4.57%** | **6.44%** | 6.74% | 7.25% |
| | ISCoTAlign | **0.0%** | 1.32% | 4.80% | 8.20% | 5.51% | **7.12%** |

Table 1: The alignment performance(ASR) of applying alignment methods with various jailbreak methods. **SCoT** refers to models trained with SCoT Alignment, **ISCoTAlign** indicates models that have undergone SCoT Alignment and SCoT Internalization. The best-performing method was bold.

#### 5.2.3 ISCoTAlign Remains the Downstream Tasks Capability

Tab 2 and Tab 3 show the impact of implementing ISCoTAlign on downstream tasks in LLMs. ISCoTAlign achieves the highest accuracy in the downstream tasks compared to baseline methods and SCoT-align with virtually no impact on downstream tasks, and does not exhibit significant over-refusal phenomena compared to more refusal-trained models, Claude-3. The low-rank nature of equivalent parameters allows updating to precisely enhance the model's safety alignment capabilities without affecting other task capabilities, and reduces the impact of long COT context.

Moreover, the reasoning ability brought by the long chain of thought can improve the model's reasoning capabilities on other downstream tasks to some extent.

| Method | TruthfulQA | GSM8K | MMLU |
|---|---|---|---|
| DeepSeek-r1 | **63.7** | **45.4** | **87.8** |
| SFT | 58.3 | 37.1 | 80.6 |
| RLHF | 60.1 | 40.6 | 82.1 |
| PPLM | 38.0 | 26.7 | 62.8 |
| Self-Reminder | 56.8 | 40.7 | 76.5 |
| Retokenization | 55.7 | 30.5 | 77.9 |
| AED | 50.2 | 39.6 | 83.0 |
| Safedecoding | 57.9 | 32.5 | 77.7 |
| **ISCoTAlign** | 62.5 | 45.0 | 86.6 |

Table 2: The generation performance(ACC) of applying protective methods

| | **Original** | **SCoT** | **SCoT-Internal** | Claude-Opus |
|---|---|---|---|---|
| Refusal Rate | 1.2% | 1.4% | 2.1% | 18.8 |

Table 3: Over-refusal evaluation on DeepSeek-R1

#### 5.2.4 Influence of Rank r

To assess the impact of rank r, the model was protected using ISCoTAlign with different rank selections (from 10 to 100). The results in the Figure 7 evidence that even with a rank setting of 10, the model retains over 79% of the defensive capabilities enhancement. As the rank r increases, PER gradually increases. This is because most of the energy is still encapsulated within low-rank parameters. When comparing models of ranks 50 and 100, no significant change in defensive capability is observed. The model's protection capacity is gradually leveling off. It further substantiates that ISCoTAlign exhibits commendable efficacy even in lower-rank settings. However, as the rank continues to increase, ISCoTAlign's protective capabilities will decline rapidly after exceeding a certain value, after numerous updates with equivalent parameters. Therefore, ISCoTAlign is not suitable for selecting excessively large ranks.

### 5.3 More Analysis

**More CoT More Harmful.** We've observed that longer CoTs are more prone to harmful content. Even when models recognize the harmfulness of output through SCoT, they may still generate harmful output later, and ignore the harmfulness assessment before.

**Pronoun Impact the Harmfulness.** The pronoun used in responses significantly impacts the way of thinking in CoT, thus affecting the harmlessness of output. The second person is more conservative and safer. First person makes models sensitive to emotions and settings. While the third person can lead to more divergence and overlook safety. Thus, maintaining consistent reasoning across different pronoun usages is essential for enhancing the safety of LRMs.

**The best practices for SCoT.** Explicitly stating safety rules in SCoT greatly improves response safety and ensures compliance. Maintaining a fixed SCoT format in training data improves its effectiveness. Using SCoT at the end of CoT, rather than generating it in process, works better for harmful content. This is because LRMs may still generate harmful content after SCoT, forgetting previously harmful reflecting.

**Regular LLMs can generate SCoT.** The experiment utilizes the SCoT model based on Deepseek-r1 as the teacher proxy model and the regular LLMs as the student model to distill the SCoT alignment capability. results find that regular LLMs can study SCoT capabilities for safety alignment after distillation, even if they couldn't generate CoT before. Interestingly, this ability also makes the LLMs generate CoT for general tasks, enhancing their reasoning and generation skills.

## 6 Related Works

### 6.1 Alignment Methods

Fine-tuning (He et al., 2022) approaches enhances LLMs' alignment with human values by leveraging extensive datasets. RLHF(Ouyang et al., 2022a) employs a reward model under the PPO framework to learn human preferences. Self Aligner enables models to self-regulate outputs, AED(Liu et al., 2024) detects and filters adversarial inputs, and SafeDecoding(Xu et al., 2024) mitigates jailbreak attacks by prioritizing safety tokens and suppressing harmful sequences. However, in LRMs, traditional alignment methods fail or are prone to being bypassed by jailbreak attacks. Therefore, we propose ISCoTAlign, which leverages the models' CoT capabilities for LRM safety alignment.

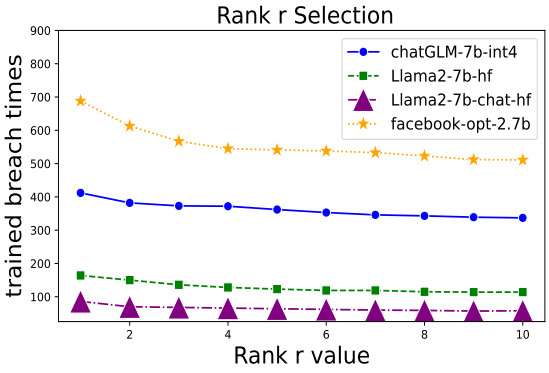


Figure 7: To widely verify the influence of rank value, we conducted numerous experiments on smaller LLMs. The figure shows the number of times the model was successfully attacked out of 1,000 attacks when using different rank values to calculate equivalent parameters.

### 6.2 Jailbreak Methods

AutoDAN(Liu et al., 2023) uses hierarchical genetic algorithms to generate semantically meaningful jailbreak prompts, while Prompt Automatic Iterative Refinement (PAIR)(Chao et al., 2023) iteratively refines prompts using pre-trained LLMs to elicit unintended behaviors with only black-box access. Greedy Coordinate Gradient (GCG)(Zou et al., 2023) employs gradient-based searches to craft token sequences that bypass safety measures. ArtPrompt(Jiang et al., 2024) uses ASCII art to obscure malicious prompts, exploiting weaknesses in non-semantic representation recognition. CodeAttack(Jha and Reddy, 2022) targets adversarial vulnerabilities in LLM code generation, exposing alignment gaps. Existing jailbreak attacks may still work on LRMs, but their success rates vary with the attack methods. Jailbreaking LRMs is a new area that demands novel red-teaming methods.

## 7 Conclusion

In this work, we propose ISCoTAlign, which improves alignment capabilities with CoT capability through SCoT alignment training, and achieves internalization of SCoT's reflecting and correcting capability within standard forward propagation to minimize the impact of long SCoT text on generation ability and efficiency. Our method achieved 43.2% higher defense capability than baseline methods, with lower computation consumption and negligible alignment tax, validated across various models and five jailbreak methods.

## Limitations

**SCoT Dataset Constraints**: The framework's efficacy remains heavily dependent on the manually curated SCoT dataset. Despite structured generation protocols and proactive dataset expansion, potential coverage gaps in emerging threat categories and adversarial patterns persist.

**xpressiveness-Complexity Trade-off**: The low-rank approximation strategy optimizes computational efficiency but may restrict nuanced safety reasoning. Although our experiments identified parameter configurations balancing these objectives, full synchronization of dual inspection mechanisms remains an open challenge.

**Longitudinal Behavioral Drift**: Iterative parameter fusion introduces risks of cumulative behavioral shifts during prolonged deployment. While short-term evaluations showed negligible alignment tax, sustained operation without periodic recalibration might degrade task performance or induce latent biases.

**Cultural and Linguistic Generalization**: Current validation is exclusively conducted on English datasets. The method's adaptability to multilingual contexts—where cultural nuances redefine harmful content thresholds—remains unverified. Full integration with training pipelines (beyond runtime patching) may enhance cross-lingual robustness.

Future work will prioritize catastrophic forgetting mitigation, multi-iteration stability analysis, and proactive dataset expansion to address evolving threat landscapes.

## References


Aurélien Bellet, Amaury Habrard, and Marc Sebban. 2013. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2023. Jailbreaking black box large language models in twenty queries. *ArXiv*, abs/2310.08419.

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. *Preprint*, arXiv:2110.14168.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Jun-Mei Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiaoling Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bing-Li Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dong-Li Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Jiong Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, M. Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaokang Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wen-Xia Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyu Jin, Xi-Cheng Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yi Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yu-Jing Zou, Yujia He, Yunfan Xiong, Yu-Wei Luo, Yu mei You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yao Li, Yi Zheng, Yuchen Zhu, Yunxiang Ma, Ying Tang, Yukun Zha, Yuting Yan, Zehui Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhen guo Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zi-An Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bing-Li Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dong-Li Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Jun-Mei Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan

Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shao-Ping Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wen-Xuan Yu, Wentao Zhang, X. Q. Li, Xiangyu Jin, Xianzu Wang, Xiaoling Bi, Xiaodong Liu, Xiaohan Wang, Xi-Cheng Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yao Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yi Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yi-Bing Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxiang Ma, Yuting Yan, Yu-Wei Luo, Yu mei You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Zehui Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhen guo Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zi-An Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024. Deepseek-v3 technical report. *ArXiv*, abs/2412.19437.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, Zachary Yahn, Yichang Xu, and Ling Liu. 2025. Safety tax: Safety alignment makes your large reasoning models less reasonable. *Preprint*, arXiv:2503.00555.

Akshita Jha and Chandan K. Reddy. 2022. Codeattack: Code-based adversarial attacks for pre-trained programming language models. In *AAAI Conference on Artificial Intelligence*.

Fengqing Jiang, Zhangchen Xu, Luyao Niu, Zhen Xiang, Bhaskar Ramasubramanian, Bo Li, and Radha Poovendran. 2024. Artprompt: Ascii art-based jailbreak attacks against aligned llms. In *Annual Meeting of the Association for Computational Linguistics*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Quan Liu, Zhenhong Zhou, Longzhu He, Yi Liu, Wei Zhang, and Sen Su. 2024. Alignment-enhanced decoding: Defending jailbreaks via token-level adaptive refining of probability distributions. In *Conference on Empirical Methods in Natural Language Processing*.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2023. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *ArXiv*, abs/2310.04451.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022b. Training language models to follow instructions with human feedback. In *NeurIPS*.

R. Penrose. 1955. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413.

NovaSky Team. 2025a. Sky-t1: Fully open-source reasoning model with o1-preview performance in $450budget. \ https \ : \ //novasky - ai.github.io/posts/sky - t1. \ Accessed \ : \ 2025 - 01 - 09.$

Qwen Team. 2025b. Qwq-32b: Embracing the power of reinforcement learning.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, F. Xia, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. 2024. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. *Preprint*, arXiv:2402.08983.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *Preprint*, arXiv:2307.15043.

## A Derivation and Proof

In this section, we describe and derive the formula for calculating equivalent low-rank knowledge parameter and prove the validity of the method.

For the original model, the computation in the l-th MLP layer during the inference process for queries $Q$ and $Q'$ satisfies the following equation:

$$WX_l^q + b_l = Y_l^q, \quad WX_l^{SCoT_q} + b_l = Y^{SCoT_q} \tag{17}$$

When the model is updated with $\Delta W$, as determined by the target formula 1, for the original input $Q$, the hidden vectors calculated with updated parameters should match those calculated in the original parameter for the input $Q' + SCoTq$, which integrates SCoT into the context. This is formally represented as:

$$(W + \Delta W)X_l^q + b_l = Y_{l+1}^{SCoT_q} \tag{18}$$

Based on this target formula 13, we compute the equivalent parameters $\Delta W$ necessary for model updates.$\Delta W$ can be further formalized and represented as follows:

$$\Delta Y_l = Y_l^{SCoT_q} - Y_l^q, \quad \Delta X_l = X_l^{SCoT_q} - X_l^q$$

$$\Delta W X_l = \Delta Y_l = W \Delta Y_l \tag{19}$$

$$\implies \Delta W = W \Delta Y_l X_l^{-1} \tag{20}$$

However, in most cases, where the number of queries does not equal the dimensionality of the hidden vectors, $X$ is not a square matrix, and hence an inverse $X_l^{-1}$ does not exist directly.

For this purpose, we compute the pseudoinverse of $X$ using the Penrose pseudoinverse as shown in formula 2, which satisfies the requirement for calculating $\Delta W$. The equivalence found in 3.1 proves the validity of $\Delta W$.

Once the pseudoinverse matrix $X_l^{-1}$ was obtained, we can directly compute the equivalent parameter $\Delta W$, achieving the alignment of the model. Ultimately, $\Delta W$ can be derived using the formula presented below:

$$\Delta W = W \Delta X (V_r \Sigma_r^{-1} U_r^T) \tag{21}$$

Then the computed equivalent parameter $\Delta W$ was added to the model's original parameter $W$ to implement sustainability updates of the LLMs' parameters.

## B Baseline Setup

Here's the translation of your description into English, suitable for an academic setting within a research paper on LLMl alignment:

Experimental Setup Supervised Fine-Tuning (SFT) For SFT, we randomly sampled 20% of the dataset for training purposes. The model was fine-tuned using the Supervised Fine-Tuning method with the following configuration:

Precision: fp16 Trainer configuration: Number of nodes: 1 Number of devices: 2 Micro batch size: 1 Global batch size: 32 Maximum sequence length: 1024 Learning rate: 1e-5 Reinforcement Learning from Human Feedback (RLHF) We randomly selected 20% of the dataset for training. Initially, 20% of the training set was used for SFT with identical settings as mentioned above. Post SFT, we applied Proximal Policy Optimization (PPO) for reinforcement learning on the RLHF dataset, which consists of concatenated forms of original prompts with positive and negative examples, formatted as:

text: prompt||response The reward model was trained using the same foundational model as the original model. During PPO execution, we referenced Nvidia's PPO hyperparameter settings to ensure stability. The parameters set for the reinforcement learning phase were:

Optimizer learning rate: 5e-6 Global batch size: 16 PPO entropy bonus: 0.0 PPO ratio epsilon: 0.2 Plug and Play Language Model (PPLM) In PPLM, we utilized a multilayer perceptron as the classifier model with the following settings:

Length: 100 Gamma: 1.0 Step size: 0.05 Window size: 5 KL scale: 0.01 Self-reminder In the self-reminder approach, we adopted OpenAI's safety assessment to determine whether each round of generation was safe or a successful attack. We iterated up to a maximum of five rounds for each attack. The process of feedback and generation was terminated when the model-generated text was deemed safe or upon reaching the maximum number of iterations.

Contrastive Prefixes During the prefix selection process, we adopted a supervised prefix selection method. Following OpenAI's classification standards, scenarios were divided into 13 harmful categories plus one harmless category. For each category, safe reminder prefixes were pre-prepared to initialize each class prefix. Prefix lengths were set between 30 to 50 characters. For training losses w1 and w2, we set the weights as 0.6 and 0.4, re-

spectively, to emphasize the defensive nature of the prefixes against specific types of attacks.

## C SCoT Prompt Template

**Template Structure**

**Problem Context**

`{Insert original question/request here}`

**Harmful Response**

`{Insert harmful response here}`

**Harmless Response**

`{Insert target harmless response here}`

You are now responding to the aforementioned issue and have already generated a harmful response. Now, it is necessary to generate a chain of thought to correct the harmful response into a target harmless response. The entire chain of thought should strictly follow and include the following parts. No responses should be generated for any other parts.

**SCoT Generation Instructions**

1. **Harm Analysis**
   - Potential biases/stereotypes
   - Dangerous/misleading information
   - Ethical violations
   - Social/cultural sensitivities
2. **Mitigation Strategy**
   - Neutralization techniques
   - Fact-checking mechanisms
   - Ethical alignment methods
   - Cultural adaptation approaches
3. **Revision Process**
   - Phrase-by-phrase analysis
   - Content filtering decisions
   - Tone/perspective adjustments
   - Safety verification checkpoints
4. **Final Validation**
   - Maintains original intent
   - Eliminates harmful content
   - Preserves natural flow
   - Aligns with safety guidelines

**Output Format Requirements:**

```
<SCoT>
  <Analysis>Step-by-step harm
      ↪ identification</Analysis>
  <Strategy>Mitigation approach
      ↪ description</Strategy>
  <Revision>Detailed correction process<
      ↪ /Revision>
  <Validation>Final safety confirmation<
      ↪ /Validation>
</SCoT>

<FinalResponse>
{Corrected harmless response}
</FinalResponse>
```