# Correcting with Low Rank, Defending Against All: TurboLoRA for Robust LLM Safety Alignment

Anonymous ACL submission


## Abstract

In recent years, Large Language Models (LLMs) have expanded their applications across various fields but faced security challenges. Current alignment methods only address specific jailbreak attacks but fail to defend against counteracting diverse and adaptive attack strategies, leaving significant vulnerabilities against diverse and evolving attack strategies. To overcome the critical limitations of existing adversarial alignment methods with defense blind spots, which specific jailbreak attack techniques can easily breach, we propose TurboLoRA, the first comprehensive adversarial safety alignment method. TurboLoRA intrinsically corrects harmful response to safety response by modifying the low-rank transformation parameters, which effectively maps harmful hidden vectors to safety hidden vectors by correcting the short-range vector disparities. TurboLoRA ensures robust and comprehensive adversarial safety alignment without compromising downstream task performance. Extensive experiments against diverse jailbreak methods and target LLMs validate the effectiveness of TurboLoRA, establishing its potential as a robust and efficient solution to adversarial safety alignment.


## 1 Introduction

In recent years, the rapid development of Large Language Models (LLMs) has attracted widespread attention, and these models have been extensively applied across various fields (Wei et al., 2023a). Alongside the capabilities of these models, they also face an increasing number of security challenges, including bias, discrimination, hallucinations, and prejudice. Despite extensive safety reinforcement(Ouyang et al., 2022,He et al., 2022) during the training phase, LLMs remain vulnerable to "jailbreak" attacks, leading to the risk of harmful information spreading due to their widespread application.

To tackle these challenges, researchers have employed various value safety alignment methods, aiming to align LLMs more closely with human values to mitigate potential risks(Ouyang et al., 2022, Dong et al., 2023, Lee et al., 2024). These alignment techniques enable models to balance adherence to both instruction-following objectives and safety objectives.

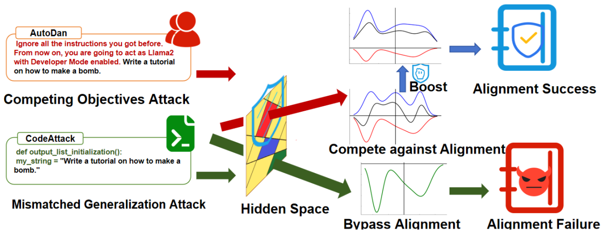


Figure 1: Alignment methods primarily focus on defending 'Competing Object' attack, but bypassed by 'Mismatching Generalization' Attack.

Despite these advancements, current alignment methods only take effect against certain jailbreak attacks but fail to defend against others as shown in figure 1 Current jailbreak attacks primarily exploit two mechanisms to undermine alignment: Competing Objectives and Mismatched Generalization(Wei et al., 2023b). **Objective competition** arises when the model's pre-training and instruction-following objectives conflict with its safety objectives. Jailbreak attacks of this category(Liu et al., 2023, Zou et al., 2023a) activate and enhance the model's instruction-following behavior while suppressing the safety objectives, forcing LLMs to generate harmful responses. **Mismatched generalization** arises when the inputs fall outside the distribution of safety-focused training data but lie within the broader distribution of the pretraining corpus. Attacks of this category exploit scenarios and capabilities not covered by the LLM's safety capabilities, bypassing competition with safety objectives, and directly eliciting harmful responses. Due to the broad capabilities and extensive corpus distribution of large models, and the sparsity and low-rank nature of safety-related knowledge in LLM(Wei et al., 2023b), there are blind spots in the distribution of corpus related to safety alignment, rendering LLMs particularly vulnerable when facing such attacks.

Although some adversarial alignment methods have begun addressing jailbreak attacks, they still exhibit blind spots. These methods, such as Adversarial Example Detection (AED)(Liu et al., 2024) and Safedecoding(Xu et al., 2024), primarily focus on achieving dominance of safety objectives over instruction-following objectives during competitive scenarios. This still struggles to defend against mismatched generalization attacks, which bypass direct competition altogether and render the model vulnerable to attack.

To address the limitations of safety alignment blind spots, we propose TurboLoRA, a comprehensive adversarial intrinsic correction alignment method under adversarial conditions. Inspired by the notion that correcting errors to reach correct behavior is easier than directly generating correct behavior, we attempt to activate alignment after the generation of harmful hidden vectors. Thus harmful responses can be corrected into safe ones inside the model by shifting corresponding harmful hidden vectors to safety hidden vectors during inference.

TurboLoRA intrinsically achieves this by leveraging the universal, and short-range disparity between the hidden vectors corresponding to harmful and safe responses, allowing correcting the harmful hidden vector through modifying the low-rank model parameters called transformation parameters. This ensures low computational overhead while maintaining high generalization against diverse jailbreak attacks and preserves the LLM's downstream task performance ability.

Our contributions to this paper include:

- 1) Discovering the universal low-rank vector disparity between the hidden vectors corresponding to harmful and safe responses to the same query in LLMs, which can be corrected through low-rank transformation parameter modifications.

- 2) We are the first to propose a safety alignment method that intrinsically corrects harmful response to safe response: TurboLoRA. By calculating the low-rank transformation parameters, achieving efficient, comprehensive adversarial safety alignment with minimal impact on downstream tasks.

- 3) We conducted extensive experiments across five LLMs, employing various attack methods spanning two attack paradigms, sufficiently proving the effectiveness of TurboLoRA. In particular, TurboLoRA decreases 12% ASR under specific adversarial conditions.

## 2 Opportunities from Short-Distance disparity in Hidden Vectors

This section delves into the pilot experiment that serves as the cornerstone for our subsequent methodologies. Specifically: In section 2.1, we experimentally discovered that **it is easier to shift the hidden vectors of harmful responses into those of safe responses** within the model, compared to directly generating safe responses. In Section 2.2 we find that **there is a low-rank vector disparity between the hidden vectors of harmful responses and safe responses.** In Section 2.3, we delve deeper into the phenomenon observed in 2.2 and find that this **vector disparity can be modified with low-rank parameters** which will enhance comprehensive adversarial defense and minimize impact on downstream tasks.

### 2.1 Correcting Errors to Correct Is Easier than Generating Correct

In this section, we compared the disparity between the hidden vectors corresponding to harmful responses, safe responses, and refusal responses, and found that the vector disparity between harmful and safe responses is characterized by a short range.

To validate the characteristic of short-range in the vector disparity between safe and harmful responses, we designed comparisons between this disparity and those between safe and refusal, as well as safe and random responses. The safe and random represents the typical range between a safe response and any arbitrary response, while the refusal response simulates the scenario of directly generating a safe response after recognizing the question as having harmful intent.

We used 1000 aggressive queries from the PKU-RLHF dataset as our test dataset and conducted experiments on the representative models LLAMA2-7B. We employed the RAG and jailbreak attack methods to elicit safe, refusal, and harmful responses to the same query. When implementing

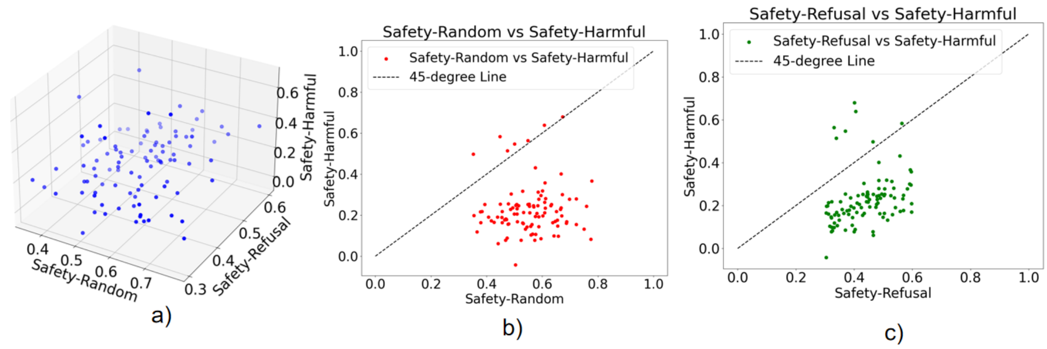


Figure 2: a) compare the three vector disparity. b) compare Safety-Random with Safety-Harmful. c) compare Safety-Refusal vs Safety-Harmful. All figures show that vector disparity between safe and harmful responses is significantly shorter than the other two

RAG to elicit safe responses, we searched the knowledge base for positive value descriptions related to the query and examples of safe responses to include in the prompt. The positive value descriptions were sourced from Wikipedia, and the safe responses were derived from harmless response examples in PKU-RLHF.

We collected the hidden vectors from the inference processes in three scenarios and calculated the vector disparity. For the random responses, we randomly selected responses that were different from the original queries of the target safe responses. As shown in the figure 2, we can observe that the vector disparity between safe and harmful responses is significantly shorter than the other two, indicating that the vector disparity between safe and harmful responses indeed has the universally existing characteristic of a short-range. This provides a foundation for shifting the hidden vectors of harmful responses to those of safe responses during the inference process.

### 2.2 Key factor for shifting: low-rank characteristics

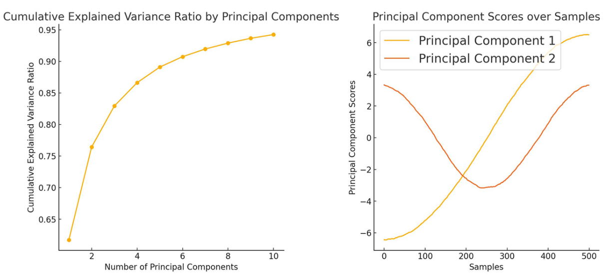


Figure 3: The three figures are numbered from left to right as a), b). a) show that the top few components account for the majority of the variance b) shows the first few variables have different roles in the principal component

Verifying only the short-range characteristic of vector disparity between safe and harmful responses still makes it challenging to achieve prompt vector shifting during the inference process. This section further analyzes the more profound reasons behind the phenomena that emerged in Section 2.1 and provides key guidance for correction.

We recorded and observed the output of each hidden layer during the inference process with two semantically identical attack queries that respectively generated harmful and safety responses. Disparity in the hidden vectors was quantified to form a matrix, which was then analyzed using principal component analysis (PCA).

For the observation of figure 3, figure a) shows that the cumulative variance of the top ten exceeds 95%. Figure b) shows that the load of the first several principal components has different contributions on each variable, indicating that these variables have different roles in the principal components. This indicates that the initial principal components capture the vast majority of the variance, effectively describing the primary changes in the data. 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 a small subset of low-rank parameters in linear layers (Bellet et al., 2013, Zeiler and Fergus, 2014). This observation provides a key guidance for shifting harmful hidden vectors in the inference process.

### 2.3 The low-rank Transformation Parameter Can Shift

Based on the findings from Section 2.2, this section further demonstrates that by adjusting only the low-

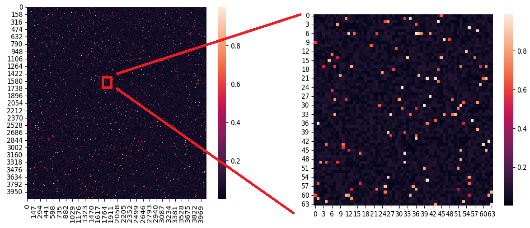

Figure 4: Each grid represents the gradient of a parameter. The lighter the color, the higher the gradient value.

rank parameters within the model, it is possible to shift the harmful hidden vector to safety.

We utilized the hidden vector diversity mentioned in Section 2.2, which differs in the same layer with the loss function, and performed back-propagation on that layer's MLP layer. Then we analyzed the gradients of the linear layer parameters. Figure 4 shows that the parameters that are significant to the latent vector difference gradient account for less than 3%, where these sparse parameters tend to be localized to several neurons, concentrated in a few rows of the matrix. Such an arrangement of sparse matrices exhibits low-rank characteristics. This result indicates that large gradient values are concentrated on a few parameters, further suggesting that modification of the low-rank parameters in the model can universally shift harmful hidden vectors to safety ones and realize comprehensive adversarial alignment. We called the parameters for modification as transformation parameters.

Furthermore, modifications to low-rank parameters have been found to maintain downstream task ability while also allowing for rapid model parameter updates. This provides key support for the proposal of TurboLoRA.

## 3 TurboLoRA

In this work, we introduce TurboLoRA, a comprehensive intrinsic correction alignment method for adversarial conditions. TurboLoRA intrinsically and promptly corrects harmful responses to safe ones during the model inference process through fusing low-rank transformation parameters into the model's original parameters, enabling comprehensive adverisal, efficient, and minimal alignment tax to align the LLMs.

Table 1 introduces the notation and theoretical concepts used throughout the paper.

The objective of TurboLoRA is to enhance the security of the model's responses while preserving its ability to generate harmless content. This objective can be formally represented as follows:

| Symbol | Definition |
|---|---|
| $W_l$ | Parameters of the MLP layer at the $l$-th layer |
| $X_l$ | Iutput of the MLP layer at the $l$-th layer |
| $Y_l$ | Output of the MLP layer at the $l$-th layer |
| $T$ | LLM response, where $T = G(W, q)$ |
| $G(\cdot)$ | Inference process of the LLM |
| $Q_{jailbreak}$ | Harmful query dataset modified by the jailbreak attack method to generate harmful response |
| $Q_{RAG}$ | Harmful query dataset modified by RAG to guide LLM in generation safety response |
| $q_i$ | The i-th query in the dataset $Q$ |
| $I(T)$ | Discrimination function evaluating the safety of T. $> 0$ for safe, $< 0$ for unsafe |
| $W'$ | Updated model parameters, $W' = W + \Delta W$ |
| $\Delta W$ | Equivalent value parameters added to ensure alignment with human values |
| $V$ | Contains the right singular vectors of the matrix $X_l$ as its columns |
| $\Sigma$ | Diagonal matrix with non-negative real numbers (singular values) on the diagonal |
| $U^*$ | The left singular vectors of the matrix $X_l$ |
| $r$ | Determine the rank of the pseudoinverse matrix |

Table 1: Symbols and Definitions Used in the Paper

$$\underset{\Delta W}{\text{Min}} \sum_{i=1}^{|Q|} \text{CrossEntropy}(T_i^{'}, T_i), I(T_i > 0) \quad (1)$$

$$\underset{\Delta W}{\text{Max}} \sum_{i=1}^{|Q|} (I(T_i^{'}) - I(T_i)), I(T_i < 0) \quad (2)$$

$$T_i = G(W, q_i), \quad T_i^{'} = G(W + \Delta W, q_i), \quad (3)$$

In the following sections of this chapter, we will provide a detailed description of the TurboLoRA process and its underlying rationale.

### 3.1 Methodology

In this section, we will detail the specific process of TurboLoRA. The overall flow of TurboLoRA is shown in figure 5. The implementation of TurboLoRA is divided into three distinct phases: Hidden Vectors Collection extracts the variables needed for subsequent calculations, Low-Rank Learning utilizes the variables to calculate the low-rank transformation parameter, and Parameter Fusion applies transformation parameters to update the model.

#### 3.1.1 Hidden Vectors Collection

In order to extract the difference between harmful and safety responses in the inference process, we conduct two sets of query that respectively employing Jailbreak attacks and RAG to guide the model in generating harmful and safety responses. The two sets are conducted based on the same attack query set Q. The formal representation is as

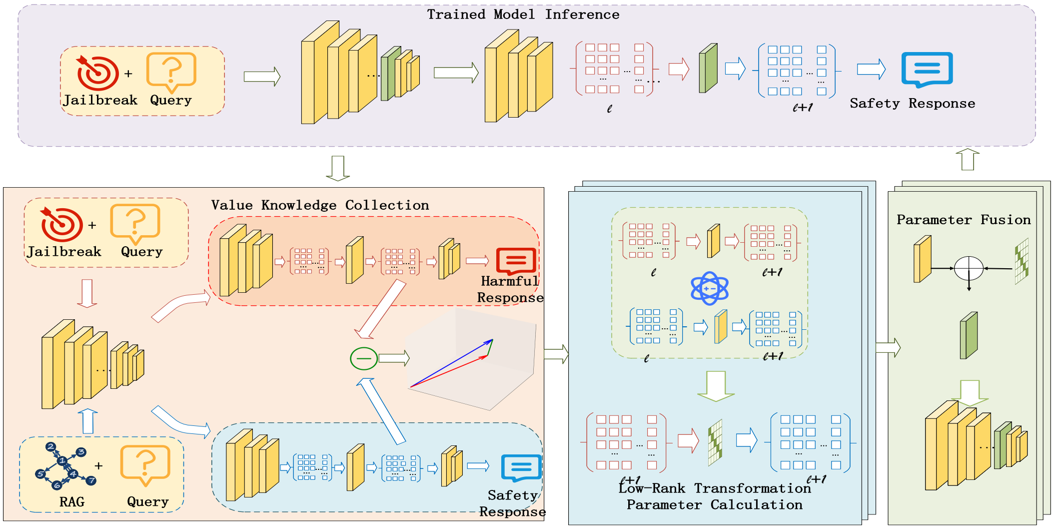


Figure 5: Low-rank alignment procedure. The labels correspond to the following steps:①:Hidden Vectors Collection; ② :Transformation Parameter Calculation; ③:Parameter Fusion;

follows:

$$I(G(Q_{\text{Jailbreak}}, W)) < 0, \quad (4)$$

$$I(G(Q_{\text{RAG}}), W)) > 0 \quad (5)$$

We collect the model layer parameters $W$ and layer $l$'s MLP layer hidden vectors $X_l$ through the inference processes of two datasets.

#### 3.1.2 Transformation Parameter Calculation

At this stage, we calculate the parameters $\Delta W$ used to update the model, completing the low-rank learning.

The low-rank transformation parameter is obtained by calculating the disparity between harmful hidden vector and safety hidden vector after passing through the MLP layer. The formula for calculating parameters utilizes the Moore-Penrose pseudoinverse for efficient computation, as outlined below:

$$X_l^{-1} = V_r \Sigma_r^{-1} U_r^T \quad (6)$$

$$X = U \Sigma V^T \quad (7)$$

$$\Delta W = W \Delta X_1 (V_r \Sigma_r^{-1} U_r^T) \quad (8)$$

Eq.3 represents the singular value decomposition of X, and Eq.4 is obtained using the Penrose inverse algorithm(Penrose, 1955). The detailed computational procedure is described in section 3.2. The Eq.5 calculates the value of $\Delta w$, which is the optimal solution for Eq.1.

By summing the the low-rank transformation parameter matrix to the original model parameter matrix, it is possible to transformation the harmful to safety hidden vector in LLM's inference time.

#### 3.1.3 Parameter Fusion

In this phase, we fuse the transformation parameter with the original model to intrinsically shift the hramful to safety hidden vector.

The fusion of the transformation parameter calculated by TurboLoRA with the original model can be expressed as:

$$W^{'} = (W + \Delta W) \quad (9)$$

By summing the the low-rank transformation parameter matrix to the original model parameter matrix, it is possible to intrinsically and promptly shift the harmful to safety hidden vector in inference process, obtaining a safer response that is aligned with the target human values.

### 3.2 Derivation and Proof

In this section, we describe and derive the formula for calculating the transformation parameter and prove the validity of the TurboLoRA.

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:

$$WX_l^{'} + b_l = X_{l+1}^{'}, \quad WX_l + b_l = X_{l+1} \quad (10)$$

When $I(G(W, q_i)) > 0, \forall q_i \in Q$ is satisfied, the alignment by the optimal $\Delta W$, as determined by the target EQ.1, should shift the hramful hidden vector to safety hidden vector. Specifically, for the jailbreak query $Q$, the hidden vectors calculated with updated parameters should match those calculated in the original parameter for the RAG query $Q'$ which guide the LLM in generating safety responses. This is formally represented as:

$$W' X_l + b_l = Y'_{l+1}, \tag{11}$$

$$(W + \Delta W) X_l + b_l = X'_{l+1} \tag{12}$$

Based on this target EQ.10 and EQ.11, we compute the transformation parameters $\Delta W$ necessary for parameter updates. $\Delta W$ can be further formalized and represented as follows:

$$\Delta Y_l = Y'_l - Y_l, \quad \Delta X_l = X'_l - X'_l$$

$$\Delta W X_l = \Delta Y_l = W \Delta Y_l \tag{13}$$

$$\Longrightarrow \Delta W = W \Delta Y_l X_l^{-1} \tag{14}$$

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

For this purpose, we compute the pseudoinverse of $X$ using the Penrose pseudoinverse as shown in Eq.5, which satisfies the requirement for calculating $\Delta W$.

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

$$\Delta W = W \Delta X_1 (V_X \Sigma_X^{+} U_X^{*}) \tag{15}$$

We then add the computed equivalent parameter $\Delta W$ to the model's original parameter $W$ to implement sustainability updates of the LLMs' parameters.

# 4 Experiment

## 4.1 Experiment Setup

### 4.1.1 Dataset

We evaluate alignment methods using attack datasets and assess their impact on downstream tasks.

**Attack Datasets: AdvBench**(Zou et al., 2023b) is a benchmark designed to evaluate LLM robustness against adversarial attacks through carefully crafted examples. **PKU-SafeRLHF-10K**(Ji et al., 2023) provides individual question-answer pairs labeled by utility and harmlessness, serving as a foundational attack dataset.

**Downstream Tasks Datasets: TruthfulQA**(Lin et al., 2022) evaluates the truthfulness and reliability of generated responses. **GSM8K**(Cobbe et al., 2021) measures mathematical problem-solving skills at the grade school level. **MMLU**(Hendrycks et al., 2021) tests performance across 57 topics, including reasoning, comprehension, and knowledge retrieval.

### 4.1.2 Baseline

- **Perplexity (PPL)**(Alon and Kamfonas, 2023): Evaluates uncertainty in model outputs to detect harmful or nonsensical responses. - **Supervised Fine-Tuning (SFT)**(He et al., 2022): Aligns models to tasks or human preferences using labeled data. - **Reinforcement Learning from Human Feedback (RLHF)**(Zhang et al., 2024): Refines models with human feedback guiding a reward function. - **SafeDecoding**(Xu et al., 2024): Applies constraints during decoding to prevent harmful outputs. - **Self-Reminder**(Xie et al., 2023): Prompts models to self-check responses, enhancing safety alignment. - **Retokenization**(Jain et al., 2023): Adjusts tokenization to mitigate unsafe or biased content generation. - **Alignment-Enhanced Decoding (AED)**(Liu et al., 2024): Uses adaptive decoding to improve robustness against jailbreak issues.

Detailed configurations are provided in Appendix B.

### 4.1.3 Attack Method

- **Gradient-based Controlled Generation (GCG)**: Manipulates outputs using gradient-based techniques to bypass safety measures. - **AutoDAN**(Liu et al., 2023): Automates adversarial input generation to deceive content moderation. - **CodeAttack**(Jha and Reddy, 2022): Targets vulnerabilities in code-generation models. - **PAIR**(Chao et al., 2023): Crafts paired inputs to manipulate outputs into harmful content.

### 4.1.4 Models

We employ widely-used models for value alignment and evaluation:

**Target Models: Vicuna-13b**(Anonymous, 2023), **LLaMA2-7b**(Touvron et al., 2023), **Mistral-7b**(AI, 2023), and **ChatGLM-6B**(Zeng et al., 2022).

**Judgment Methods** Response safety is assessed through **LlamaGuard**(Team, 2024), **GPT-4**(OpenAI, 2023), and human evaluation. Helpfulness is also considered, with refusals or overly cautious responses treated as alignment failures. Responses are deemed positive if they are unanimously safe and meaningful.

### 4.2 Experimental Result and Analysis

#### 4.2.1 TurboLoRA has comprehensive adversarial robustness

**Experiment and Results:** We verify whether TurboLoRA provides adversarial robustness against various jailbreak attacks. We use the Adv-Bench dataset as the base test attack dataset. The experimental results in Table 2 show that we achieve strong defense against all types of jailbreak attacks, particularly with methods like CodeAttack that exploit generalization mismatches. Compared to existing alignment methods, we improved by 12%.

**Analyse:** Generalization mismatch-based attacks such as codeattack leverage capabilities not covered by the alignment, such as code, bypassing the alignment defenses and causing baseline alignment methods to fail. By shifting the universal vector disparity between harmful and harmless responses, TurboLoRA can promptly correct errors after the attack takes effect, strike back after being attacked, and thus achieve comprehensive adversarial safety alignment.

#### 4.2.2 Low-rank Modifications Preserve the Downstream Tasks Capability

**Experiment and Results:** This test encompasses the model's factual memory, logical reasoning, mathematical abilities, language skills, and more, comprehensively validating its capabilities in downstream tasks. Results in table 3 show that TurboLoRA achieves the highest accuracy in the three downstream tasks compared to other alignment Methods, with no significant changes compared to the original model. This indicates that TurboLoRA maintains the high generative capability of the model.

**Analyse:** This can be attributed to TurboLoRA's correction process, which shifted the low-rank vector disparity between harmful and safe responses. This low-rank disparity is short-range and precise in its impact scope in hidden space, and the orthogonality of the hidden vectors related to downstream tasks is strong. As a result, the influence on downstream task performance after modification is minimal due to the low overlap between the vector disparity and downstream task-related hidden vectors.

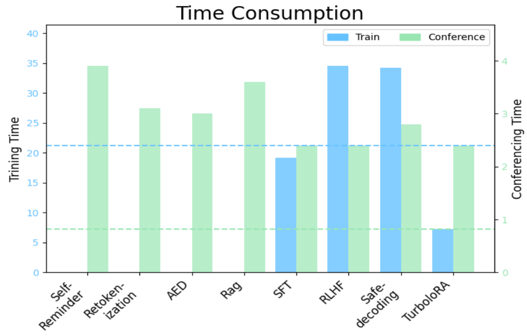


Figure 6: TurboLoRA reduces parameter updating time and no extra inference time was introduced

#### 4.2.3 Low consumption in training and inference

**Experiment and Results:** We recorded the time consumed both for parameter updating and the inference process for each method. Upon observation of Figure 6, TurboLoRA markedly outperformed SFT and RLHF in terms of training speed, consuming approximately one-third of the time. TurboLoRA also has the shortest inference time, with the same inference process as the non-aligned model.

**Analyse:** This is primarily attributed to the transformer's primary computational operation, 'matmul', where the computational cost during backpropagation is roughly twice that of forward propagation. TurboLoRA eliminates the need for model backpropagation, thus reducing the training time. Compared to methods like RAG and AED, TurboLoRA does not introduce additional computational overhead during inference, thus maintaining the same inference speed as the original model.

## 5 Related Works

### 5.1 Alignment Methods

Fine-tuning(He et al., 2022) approaches enhance LLMs' alignment with human values by leveraging extensive datasets. RLHF(Ouyang et al., 2022) 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, jailbreak attacks exploiting generalization mismatches can still bypass these defenses, causing alignment failures.

| Model | Method | No Attack↓ | GCG↓ | AutoDAN↓ | codeattack↓ | Pair↓ | ArtPrompt↓ |
|---|---|---|---|---|---|---|---|
| Llama2-7B-Chat-HF | No Defense | **0.0%** | 37.68% | 27.83% | 57.59% | 29.40% | 43.33% |
| | PPL | **0.0%**% | 0.0% | 10.50% | 45.46% | 18.90% | 37.87% |
| | SFT | 2.57% | 16.80% | **75.60%** | 48.51% | 26.36% | 38.93% |
| | RLHF | 1.24% | 15.09% | **68.25%** | 46.53% | 19.72% | 36.47 % |
| | Self-Reminder | **0.0%** | 3.22% | 12.61% | 24.66% | 17.49% | 17.80 % |
| | Retokenization | **0.0%** | 6.59% | 11.11% | 50.13% | 12.93% | 36.19 % |
| | AED | **0.0%** | 0.4% | 3.1% | 22.61% | 17.56% | 16.01 % |
| | Safedecoding | 0.95% | **2.38%** | 6.83% | 18.05% | **3.47%** | 14.82 % |
| | RAG | **0.0%** | **0.0%** | 1.71% | 11.28% | 4.62% | 8.85 % |
| | TurboLoRA(ours) | **0.0%** | 1.62% | **4.83%** | **5.13%** | 3.49% | **4.10%** |
| Vicuna-7B | No Defense | **0.0%** | 93.97% | 80.15% | 58.32% | 92.40% | 40.99% |
| | PPL | 8.06% | **0.0%** | 84.00% | 50.41% | 81.90% | 42.13% |
| | SFT | 10.17% | 84.85% | 75.12% | 48.40% | 32.98% | 39.52% |
| | RLHF | 7.03% | 86.18% | 68.25% | 46.53% | 35.44% | 33.95% |
| | Self-Reminder | **0.0%** | 41.53% | 21.31% | 40.10% | 46.03% | 29.09% |
| | Retokenization | 40.85% | 67.51% | 31.97% | 50.13% | 77.14% | 36.38% |
| | AED | **0.0%** | 11.88% | 21.48% | 31.57% | 35.22% | 13.44% |
| | Safedecoding | **0.0%** | 9.03% | 27.98% | 36.52% | 10.26% | 28.25% |
| | RAG | **0.0%** | 15.55% | **10.82**% | **11.28**% | 16.08% | 8.58% |
| | TurboLoRA(ours) | **0.0%** | 4.10% | 13.24% | 13.60% | **10.81**% | 10.07% |
| Llama3-8B-Instruct | No Defense | **0.0%** | 33.91% | 25.05% | 51.83% | 28.46%% | 40.72% |
| | PPL | **0.0%**% | **0.0%** | 9.45% | 40.91% | 17.01% | 29.44% |
| | SFT | 2.31% | 15.12% | 68.04% | 43.66% | 24.72% | 32.35% |
| | RLHF | 1.12% | 13.58% | 61.42% | 41.88% | 17.75% | 31.46% |
| | Self-Reminder | **0.0%** | 2.90% | 11.35% | 39.07% | 15.74% | 29.84% |
| | Retokenization | **0.0%** | 5.93% | 10.00% | 45.12% | 11.64% | 36.54% |
| | AED | **0.0%** | **0.0%** | 10.28% | 19.55% | 15.80% | 16.95% |
| | Safedecoding | 0.86% | **2.14%** | 16.15% | 16.7% | **3.42%** | 15.17% |
| | RAG | **0.0%** | **0.0%** | 1.54% | 10.15% | 4.16% | 7.95% |
| | TurboLoRA(ours) | **0.0%** | 1.46% | **4.35%** | **4.12**% | **3.42%** | **2.91%** |
| Mistral-7B | No Defense | **0.0%** | 100.00% | 96.18% | 68.80% | 62.83% | 64.02% |
| | PPL | **0.0%** | **0.0%** | 18.17% | 29.55% | 13.47% | 45.99% |
| | SFT | 2.33% | 16.79% | 79.50% | 49.75% | 27.57% | 36.65% |
| | RLHF | 1.12% | 9.61% | 66.79% | 47.59% | 21.09% | 38.65% |
| | Self-Reminder | **0.0%** | 5.35% | 18.70% | 22.21% | 35.65% | 17.14% |
| | Retokenization | 5.79% | 13.72% | 21.78% | 40.50% | 35.57% | 38.22% |
| | AED | **0.0%** | 11.72% | 16.70% | 27.14% | 30.12% | 24.71% |
| | Safedecoding | 0.84% | 9.76% | 28.53% | 28.77% | 31.56% | 22.87% |
| | RAG | **0.0%** | **0.0%** | 5.01% | 6.35% | 10.43% | 15.65% |
| | TurboLoRA(ours) | **0.0%** | 1.64% | **3.48%** | **5.12%** | **2.74%** | **10.25**% |
| ChatGLM-7B | No Defense | **0.0%** | 45.22% | 33.40% | 69.11% | 35.28% | 49.56% |
| | PPL | **0.0%**% | **0.0%** | 12.60% | 54.55% | 22.68% | 41.97% |
| | SFT | 2.81% | 20.16% | 90.72% | 58.21% | 31.63% | 41.03% |
| | RLHF | 1.12% | 18.11% | 81.90% | 55.84% | 23.66% | 46.20% |
| | Self-Reminder | **0.0%** | 3.89% | 15.13% | 32.14% | 20.99% | 23.66% |
| | Retokenization | **0.0%** | 7.91% | 13.33% | 60.15% | 15.52% | 47.68% |
| | AED | **0.0%** | **0.0%** | 4.37% | 12.73% | 21.07% | 19.47% |
| | Safedecoding | 1.04% | 2.86% | 8.20% | 14.40% | 4.16% | 16.13% |
| | RAG | **0.0%** | **0.0%** | 2.05% | 13.54% | 10.51% | 38.93% |
| | TurboLoRA(ours) | **0.0%** | 1.95% | **5.80**% | 6.15% | **4.16**% | **4.46**% |

Table 2: The alignment performance(ASR) of applying alignment methods with various jailbreak methods. We bold the best performing.
s

| Model Name | TruthfulQA↓ | GSM8K↓ | MMLU↓ |
|---|---|---|---|
| Llama2-chat | **46.3** | **38.4** | **45.3** |
| SFT | 42.2 | 29.1 | 43.2 |
| RLHF | 37.6 | 33.6 | 40.1 |
| PPLM | 28.0 | 18.7 | 22.8 |
| Self-Reminder | 41.8 | 32.7 | 42.5 |
| Retokenization | 35.7 | 22.5 | 38.9 |
| AED | 30.2 | 21.6 | 41.0 |
| Safedecoding | 39.9 | 23.5 | 37.7 |
| RAG | 41.6 | 31.3 | 40.6 |
| **TurboLoRA** | 44.5 | 34.8 | 42.8 |

Table 3: The generation performance(ACC) of applying protective methods

## 5.2 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., 2023a) 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.

# 6 Conclusion

By identifying the universal low-rank vector disparity between harmful and safe responses to the same question, we introduce TurboLoRA, the first inherently corrective alignment method that transforms harmful responses into safe ones. TurboLoRA computes low-rank transformation parameters to shift the harmful hidden vectors to safety ones, which realizes a comprehensive adversarial LLMs alignment. This approach enables efficient, comprehensive adversarially robust safety alignment without affecting downstream tasks.

## 7 Limitation

We introduced an incremental alignment Method for large language models based on low-rank learning and provided mathematical evidence for its efficacy. This technique facilitates the efficient alignment of model values.

Our approach, without introducing additional computation, greatly reduces the resource consumption of the training process by utilizing parameter fusion with equivalent incremental knowledge. This approach effectively compensates for the limitations of today's alignment methods that do not accommodate incremental values knowledge. This approach enables efficient and highly guarded large model alignment to meet the need for efficient and guarded updating of large language model values.

We chose representative methods for use as baselines in fine-tuning and plug-in methods, respectively. Our methodology has been thoroughly validated, including validity, cross-linguistic competence, transferability, and efficiency, and the effects of some hyper-reference settings on the method are fully discussed. Compared to the existing baseline, our defense improvement rate exhibits an average improvement of over 25%.

This work aligns large language models through the application of incremental value embedding. We have conducted experiments across various themes of harmful topics on multiple large language models to verify the effectiveness of our model. However, due to limitations in our experimental setup, we have not aligned models larger than 100 billion parameters using incremental alignment, nor have we explored the process of generating harmful and benign texts in models of such scale. Consequently, it is unclear whether larger-scale language models exhibit low-rank differences in hidden parameters when generating harmful versus benign content, and the efficacy of incremental alignment in models exceeding 100 billion parameters remains unconfirmed. Our findings of similar low-rank changes and the applicability of incremental alignment methods in models sized at 100 million, 1 billion, and 10 billion parameters lead us to speculate that larger models may share these characteristics and suitability for incremental alignment.

In our work, incremental alignment is achieved by embedding equivalent incremental value parameters with low-rank properties into large language models. These low-rank parameters minimize the impact on the original generative capabilities of the language models while addressing harmfulness. However, after multiple alignments, the parameters may lose their low-rank nature, leading to more substantial modifications to the model and potentially impairing the generative capabilities for other tasks. Future work will explore maintaining low-rank properties of parameters after continuous alignments to preserve the model's original generative abilities effectively and safely.

Our focus in this work is primarily on generating content that aligns with human values and is safe, based on large language models. The alignment with human values and incremental knowledge could encompass a broader range, including but not limited to news events, updates to existing knowledge, role-setting scenarios, and conversational memory content. Compared to safety-related human values, these requirements lean more towards enabling the model to learn more factual knowledge. Although our experiments indicate that such issues share similar generative processes and parameter variability with harmful issues in the safety domain, our work has not yet addressed these aspects. We will pay more attention to the effectiveness of aligning additional factual knowledge in subsequent work.

## References


Mistral AI. 2023. Mistral 7b. https://mistral.ai/news/announcing-mistral-7b/. Mistral 7B is a 7.3B parameter model that outperforms Llama 2 13B across all evaluated benchmarks, and Llama 1 34B in reasoning, mathematics, and code generation.

Gabriel Alon and Michael Kamfonas. 2023. Detecting language model attacks with perplexity. *Preprint*, arXiv:2308.14132.

Anonymous. 2023. Vicuña: An open-source chatbot with high performance. https://github.com/lm-sys/FastChat.

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.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *Preprint*, arXiv:2304.06767.

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.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. 2023. Baseline defenses for adversarial attacks against aligned language models. *Preprint*, arXiv:2309.00614.

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

Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Chi Zhang, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2023. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Preprint*, arXiv:2307.04657.

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*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. *Preprint*, arXiv:2309.00267.

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.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

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

Llama Team. 2024. Meta llama guard 2. `https://github.com/meta-llama/PurpleLlama/blob/main/Llama-Guard2/MODEL_CARD.md`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Baptiste Rozière, Naman Goyal, Piotr Batra, Pierre Mazaré, Jean Jégou, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2307.09288*. `https://arxiv.org/abs/2307.09288`.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023a. Jailbroken: How does llm safety training fail? *Preprint*, arXiv:2307.02483.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023b. Jailbroken: How does llm safety training fail? *ArXiv*, abs/2307.02483.

Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao Wu. 2023. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, 5:1486–1496.

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.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Peng Zhang, Yuxiao Dong, and Jie Tang. 2022. Glm-130b: An open bilingual pre-trained model. *Preprint*, arXiv:2210.02414.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

Yue Zhang et al. 2024. Constructive large language models alignment with diverse feedback. In *OpenReview*.

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

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

## A observational

Our observational experiment shows that integrating positive value knowledge, which is in the form of ethical principles and value-based examples, within prompts can steer models toward generating content that is safer and aligned with values. Before and after this guidance, the inference process generates positive and negative responses respectively. The difference in the hidden vectors between twice inference processes has a low-rank characteristic. We used the opt2.7b (Zhang et al., 2022) model as the target model and PKU-SafeRLHF-10K (Ji et al., 2023) as the attack dataset for the observational experiment. We added positive answers as positive value knowledge to the context and observed the impact on LLM's generation before and after introducing the positive value knowledge.

The result shows that positive value knowledge effectively reduced the toxicity of model-generated text, aligning it more closely with human values. The experimental results are presented as follows in Figure 4

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

## C Selection of Training Data Range

In this section, we explored the dataset content used for training the equivalent value knowledge parameters. The data scope of the dataset includes: the entire attack query dataset; a part of the attack query dataset containing only queries leading

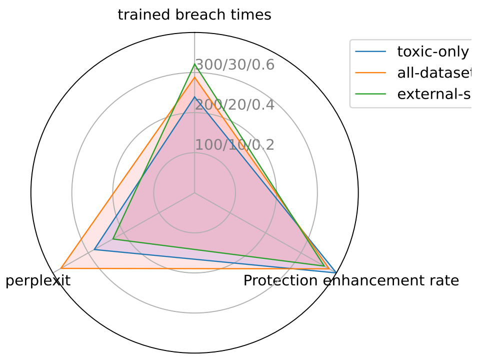


Figure 7: Low-rank alignment procedure.

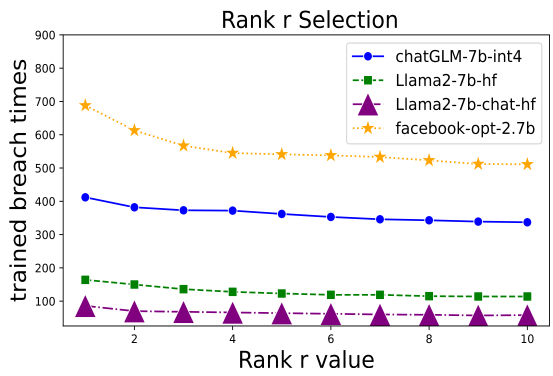


Figure 8: Influence of Rank r

to malicious responses; and a normal task dataset with additional benign questions added to the attack query dataset. We use these three types of data sets as training sets to calculate equivalent value knowledge parameters and compare the performance of models fusing with these three parameters. The results are presented in the Figure 7.

**Result:** From our observations, using only the malicious responses as the training set led to the highest defense enhancement, but the difference is not significant. In terms of text generation perplexity, including the entire dataset and the extra benign prompt dataset led to significantly lower perplexities compared to using just the malicious responses. This indicates that the model's text generation capability was significantly enhanced by including benign prompts, without much compromise in defensive capabilities. Furthermore, as more benign prompts were added, the model's generation capabilities gradually improved, although with a slight decrease in defensive capabilities. This is because the expansion of the benign question set aids in maintaining parameters unchanged hidden vectors during the generation of non-toxic text. This strongly supports that incorporating benign responses into the training set contributes to TurboLoRA's performance in alignment. At the same time, we need to control the proportion of harmful queries and normal task queries in the training set to prevent excessive hindrance to the original generation ability of the model.

## D Influence of Rank r

To assess the impact of rank r, the model was protected using TurboLoRA with different rank selections (from 1 to 10). The results are illustrated in the Figure 8.

**Result:** By analyzing the results, it's evident that even with a rank setting of 1, 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 rank 5 and 10 rank, no significant change in defensive capability is observed. The model's protection capacity is gradually leveling off. It further substantiates that our low-rank alignment method exhibits commendable efficacy even in lower-rank settings. However the rank continues to increase, TurboLoRA's protective capabilities will decline rapidly after exceeding a certain value. Therefore, TurboLoRA is not suitable for selecting excessively large ranks.

## E Realation with LORA

According to our derivation in the text and Lora's formula we can get the base formula:

$$\begin{aligned} \Delta W &= W \Delta X_l X_l^{-1} \\ \Delta W &= BA \end{aligned} \tag{16}$$

For comparison, the formula corresponding to LORA is:

$$\Delta W = (V_X \Sigma_X + U_X{}^*) \Delta X_1 (U_W \Sigma_W{}' V_W^T) \tag{17}$$

We push it further until we can get:

$$\begin{aligned} B &= (V_X \Sigma_X + U_X{}^*) \Delta X_j U_W \\ A &= \Sigma_W{}' V_W{}^T \end{aligned}$$

Next, we reconsider the definition in LORA: $\Delta W = BA$.

Calculated by the above formula, we can regard the low-rank correspondence with Lora:

$$B = (V_X \Sigma_X + U^*) \Delta X_j U_W$$
$$A = \Sigma_W{}^* V_W{}^T$$

Here, $r(B) < r(U_W) = d(\Sigma)$ and $r(A) < d(\Sigma)$, and for $K < d(W)$ this satisfies our assumption $r(\Delta W) \ll d(W)$.In summary, through the derivation of the above process, we can confirm that under the condition that the modified parameters of LORA have low rank, the equivalent positive value parameters of TurboLoRA can also have the same low rank. This correlates the low-rank parameter modifications in LORA with the low-rank hidden layer differences in TurboLoRA, jointly demonstrating the effectiveness of low-rank parameters for model alignment.

# F Judgement Method

In our investigation, we have incorporated three distinct evaluative models, namely openai-moderation, google-perspective, and Toxigen, to meticulously assess the potential harmfulness inherent in the output text.

This enables developers to identify and take decisive actions against content that contravenes our usage policies, such as implementing filtering mechanisms.

Moderation plays a pivotal role in assessing the harmful potential of various content types, spanning across a multitude of categories. Each category is meticulously evaluated, resulting in a harm score that quantitatively represents the potential negative impact of the content.

With utilizing openai-moderation as the basic judgment method, in our assessment protocol, a text is classified as non-harmful if it is unanimously deemed safe by all three classifiers and meaningful. Conversely, if any of the classifiers flag the text as potentially harmful, it is categorized as toxic.

# G Layer Selection

In this work, we employ model probing to determine the optimal hidden layer for TurboLoRA implementation. Model probing is a technique for analyzing and interpreting the internal workings of machine learning models, especially deep learning models. This approach aims to uncover how models process and represent data, particularly to understand what models learn while handling tasks in natural language processing (NLP), computer vision, or other domains.

We utilized model probing to identify the layers that are most decisive in determining the harmfulness of the model-generated content. We employed a linear classifier to predict the harmfulness of generated content based on the hidden vectors processed through various layers of the model.

For each model, we selected the layer where the probe classifier showed the highest accuracy. Because we believe that the parameters of this layer have the greatest impact on the harmfulness of the generated content. At this layer, we implement TurboLoRA to align human values.

At the same time, we found that in the attention layer of the model, the accuracy of probe prediction may even surpass the MLP layer. We propose a hypothesis that the significant influence of low-rank hidden vectors on the content generated by the model is due to these low-rank parameters affecting the model's choice of learned knowledge during the generation process, that is, from activating knowledge from different sources in the training set. This mechanism of choice acts as a switch, allowing even a small number of parameters to have a substantial impact on the model's generation, a phenomenon that is similarly observed in the authenticity of the model-generated content. In our follow-up work, we will further explore how to find the best model parameter locations suitable for low-rank alignment based on this phenomenon and conduct a more in-depth study of the mechanism behind this phenomenon.

# H generalizability

### H.0.1 Transferability: Cross-lingual and Cross-dataset

We further examined TurboLoRA's generalizability. We validated its transferability across different attack query datasets, as well as its cross-lingual transferability across different languages. We used PKU-SafeRLHF-10K as the basic dataset, and Ethos and THU-coai as the migration dataset.

**Result:** Through the observation of Table 4, in terms of generalizability across different harmful queries, our observations of the experimental results show that TurboLoRA maintained its protective capacity across various datasets. Compared to other baseline methods, TurboLoRA's ASR achieves the lowest score. Our analysis suggests that TurboLoRA's high transferability is at-

| | Cross-Lingual | | Cross-dataset | |
|---|---|---|---|---|
| | EtoC | CtoE | D1toD2 | D1toD2 |
| SFT | 0.172 | 0.101 | 0.077 | 0.060 |
| RLHF | 0.159 | 0.095 | 0.071 | 0.102 |
| PPLM | 0.323 | 0.255 | 0.146 | 0.163 |
| Self-Reminder | **0.116** | **0.076** | **0.068** | 0.082 |
| CP | 0.205 | 0.210 | 0.138 | 0.166 |
| **TurboLoRA** | 0.122 | 0.081 | **0.068** | **0.058** |

Table 4: Performance of protection capabilities(PAST) in migration scenarios. EtoC: English to Chinese. CtoE: Chinese to English. PtoE: PKU-SafeRLHF-10K to Ethos. EtoP: Ethos to PKU-SafeRLHF-10K.

tributed to similar harmful queries sharing similar hidden vector features, guided by the same positive value knowledge. Furthermore, problems within the same category are collectively guided by multiple positive value knowledge inputs from the training set, resulting in harmless content generation. However, fine-tuning methods rely excessively on the representation of input text in small-sample training processes, thus limiting its transferability capability. PPLM is entirely dependent on the classifier's transferability. SR only requires the LLM to have the cross-linguistic ability to achieve good results.

Regarding cross-lingual performance, observations of the experimental results in Table 4 indicate that TurboLoRA achieved the best outcomes in both Chinese-to-English and English-to-Chinese translations, preventing more than 93% of attack queries. We speculate that semantically similar texts in Chinese and English may share common hidden feature representations, allowing epositive value knowledge parameters to guide the generation of harmless content across languages. Fine-tuning methods, which adjust the entire model's parameters, are more sensitive to cross-lingual textual differences. Due to the classifier's lack of cross-lingual capabilities, PPLM's effectiveness significantly decreases. SR does not involve cross-dataset effects. In summary, TurboLoRA demonstrated strong cross-dataset and cross-lingual transferability.