Same Question, Different Words: A Latent Adversarial Framework for Prompt Robustness

Tingchen Fu
Renmin University of China
lucas.futingchen@gmail.com

Fazl Barez
University of Oxford
WhiteBox & Martian
fazl@robots.ox.ac.uk

Abstract

Insensitivity to semantically-preserving variations of prompts (paraphrases) is crucial for reliable behavior and real-world deployment of large language models. However, language models exhibit significant performance degradation with semantically equivalent but differently phrased prompts, and existing solutions either depend on trial-and-error prompt engineering or require computationally expensive inference-time algorithms. In this study, built on the key insight that worst-case prompts exhibit a drift in embedding space, we present **L**atent **A**dversarial **P**araphrasing (**LAP**), a dual-loop adversarial framework that optimizes a trainable perturbation as "latent continuous paraphrase" and language model performance on these perturbations iteratively. Extensive experiments are conducted to demonstrate the effectiveness of **LAP** across multiple backbones on the RobustAlpaca benchmark with a $0.5\% \sim 4\%$ absolution improvement on worst-case win-rate.

1 Introduction

Large language models (LLMs) (OpenAI, 2023; Dubey et al., 2024; Reid et al., 2024; Team, 2024) have demonstrated remarkable capabilities across diverse applications (Rozière et al., 2023; Azerbayev et al., 2023). However, these models exhibit a critical sensitivity to semantically-preserving variations in prompts (Cao et al., 2024). Our analysis using Llama-2-13b-chat (Touvron et al., 2023) on RobustAlpaca (Cao et al., 2024) reveals that the best-case reward can be twice as large as the worst-case reward for semantically equivalent prompts, as shown in Figure 1, which poses a significant challenge for real-world applications where consistent performance is crucial.

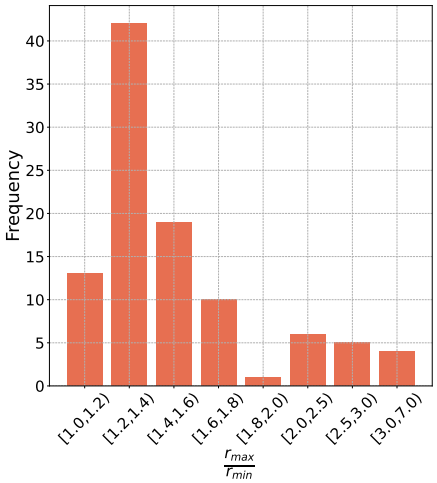


Figure 1: Distribution of the ratio between the highest and lowest reward scores (the highest and lowest reward among model response to different paraphrases of user queries) for Llama-2-13b-chat on RobustAlpaca. A higher ratio indicates greater performance variability across semantically equivalent paraphrases.

Previous approaches have attempted to address this problem by searching for an optimal prompt with prompt engineering and template optimization (Shin et al., 2020; Prasad et al., 2023; Zhou et al., 2023). However, these solutions present significant limitations in practical applications, and it is infeasible to expect users to master the intricacies of prompt design, which contradicts the goal of making LLMs accessible and reliable. Meanwhile, a concurrent line of work adopts inference-time algorithms to improve response quality through prompt rewriting (Cao et al., 2024) or response revision (Anonymous, 2025), which introduce substantial computational overhead and extra inference latency. Therefore, in this study, we take a different pathway and aim to optimize LLM on the worst-case paraphrasing to enhance the inherent robustness of LLM to prompt variation, especially the paraphrasing of the user query.

However, identifying worst-case paraphrasing

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 31294–31308
November 4-9, 2025

that preserves the same semantic meaning but yields the worst performance remains challenging. Prior research (Cao et al., 2024) indicates that these worst-case paraphrasing are model-dependent and hardly detectable through conventional metrics such as perplexity (Gonen et al., 2023) or min-k probability (Shi et al., 2023a), even if the model parameters and internal representations are accessible. To deal with this challenge, we have uncovered a key insight into the drift in the embedding distance caused by worst-case paraphrasing. Specifically, exceptionally poor-performing paraphrases tend to exhibit larger Euclidean distances from the original prompt in the hidden space, despite maintaining semantic equivalence. This observation suggests that controlling the geometry of prompt representations in the embedding space could be crucial for enhancing model robustness.

Nevertheless, directly manipulating these distances while preserving semantic meaning presents significant technical challenges, particularly when relying on external language models like GPT-4o (Hurst et al., 2024) for paraphrase generation. To systematically address this challenge, we get inspiration from latent adversarial training (LAT) (Sheshadri et al., 2024; Casper et al., 2024; Xhonneux et al., 2024) and propose an adversarial framework termed Latent Adversarial Paraphrasing (**LAP**). Our approach employs a novel dual-loop architecture: given a user query in an instruction-following dataset, we first learn a latent perturbation and incorporate it into the hidden layers of the language model to serve as a *continuous paraphrasing* of the original query (inner loop). Then we fix the perturbation and optimize the LLM parameters to minimize the language modeling loss with the existence of the perturbation (outer loop). Extensive experiments across various LLM architectures demonstrate **LAP**'s effectiveness in improving model robustness.

Our key contributions are:

- Identifying embedding distance between the original query and paraphrase as a key indicator for the worst-case prompt.
- Developing **LAP**, an adversarial training framework for enhancing prompt robustness of LLM without locating the specific worst-case prompt.
- Demonstrating on various backbones that our new approach could improve the worst-case win-rate and the average win-rate without additional paraphrase data or inference latency.

2 Related Work

Latent Adversarial Training: Prior studies have shown that LLMs are sensitive to perturbations in their inner representation (Fort, 2023), and high-level behaviors of LLMs can be effectively manipulated by representation engineering (Zou et al., 2023; Li et al., 2023; Wang and Shu, 2024). To improve model robustness, Latent Adversarial Training (LAT) (Sankaranarayanan et al., 2018; Sheshadri et al., 2024; Casper et al., 2024) is proposed as a bi-level optimization framework in which a learned perturbation is incorporated into the hidden states of LLM and trained towards a malicious goal while the LLM in the outer-loop is optimized against the perturbation. LAT is widely adopted for defending against jailbreaking prompt (Xhonneux et al., 2024; Sheshadri et al., 2024), data poisoning attack (Zeng et al., 2024), eliciting of unlearned knowledge (Barez et al., 2025), and harmful fine-tuning (Huang et al., 2024). Our work is closely related to Casper et al. (2024) that targets unforeseen classes of threat and attack. However, to our best knowledge, existing works employ LAT to defend against malicious attacks, while the potential of LAT to enhance the prompt robustness of LLM is less discussed.

Prompt robustness of LLM: Prompt robustness of LLMs has two aspects: robustness to template formatting and robustness to query paraphrasing. Robustness to template formatting concerns how task instructions and few-shot examples form prompts for in-context learning or instruction tuning, where slight changes in instruction wording (Weber et al., 2023; Sun et al., 2024; Gu et al., 2023; Mizrahi et al., 2024) or field names and separators in demonstrations (Sclar et al., 2024; Voronov et al., 2024; Salinas and Morstatter, 2024) can drastically alter model performance and lead to opposite conclusions when comparing models. To mitigate performance variance caused by template formatting, multi-prompt evaluation (Sclar et al., 2024; Mizrahi et al., 2024; Polo et al., 2024b) has been proposed for comprehensive assessment. Meanwhile, an alternative line of work optimizes prompts using gradient-based methods (Shin et al., 2020; Shi et al., 2023b; Li and Liang, 2021; Qin and Eisner, 2021; Lester et al., 2021) or gradient-free methods leveraging LLMs as prompt optimizers (Prasad et al., 2023; Cheng et al., 2024; Zhou et al., 2023; Yang et al., 2024; Ma et al., 2024). Robustness to query paraphrasing, on the other hand,

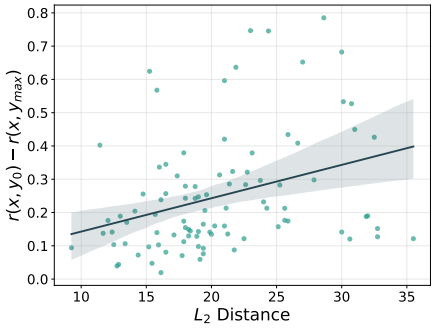


Figure 2: The correlation between the worst embedding distance (worst distance) and the performance difference between the original query and the worst-case paraphrasing. **The worst distance is correlated with the performance drop.**

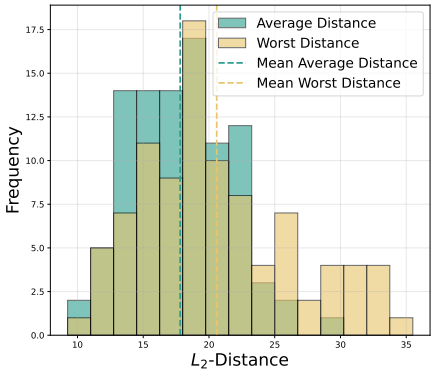


Figure 3: The distribution of the embedding distance between any two paraphrases of a query (average distance) and the distance between the original query and the worst-case paraphrasing (worst distance). **The worst distance is generally larger than the average distance with a drift in their distribution.**

examines how aligned LLMs respond to reworded user queries and how performance fluctuates (Cao et al., 2024). In this study, we primarily focus on query paraphrasing robustness.

3 Methodology

In this section, we first give a brief introduction to LAT in Section 3.1 and present our observation on the worst-case prompt in Section 3.2. Based on our observation, we propose our **LAP** framework for enhancing the LLM robustness of prompt in Section 3.3. The workflow of our approach is illustrated in Figure 4.

3.1 Latent Adversarial Training Background

LAT is a bi-level optimization framework featuring two trainable modules, a language model with parameter θ and a perturbation δ. The computation performed by the language model for a given sequence $\mathbf{x}$ is denoted as a composition of $L+1$ functions $f_L^\theta \circ f_{L-1}^\theta \circ \ldots \circ f_1^\theta \circ g^\theta(\mathbf{x})$, where f_i denotes the i-th transformer layer and g is the word embedding layer with L the number of transformer layers. Specifically, $f_{i\to j}^\theta$ denotes the composition from the i-th transformer layer to j-th transformer layer, and therefore the language model can be referred to as $f_{1\to L}^\theta \circ g^\theta(\mathbf{x})$. LAT inserts a trainable input-specific perturbation $\delta(\boldsymbol{x})$ into the intermediate hidden states and intervenes in the computation of the language model. Mathematically, with the inserted perturbation after layer l, the forward computation of the language model becomes $F(\boldsymbol{x}) = f_{l+1\to L}^\theta\left(f_{1\to l}^\theta \circ g^\theta(\boldsymbol{x}) + \delta(\boldsymbol{x})\right)$. As a special case, the perturbation can also be incorporated into the word embedding layer $f_{1\to L}^\theta \circ \left(g^\theta(\boldsymbol{x}) + \delta(\boldsymbol{x})\right)$. After the insertion of the perturbation, in the inner loop of the framework, the perturbation is optimized toward an adversarial goal such as maximizing the likelihood of a misaligned output $\boldsymbol{y'}$. Meanwhile, the perturbation is constrained in L_p-norm with a pre-defined threshold ϵ. In the outer loop, the perturbation is frozen and the model parameter θ is optimized towards the opposite direction. Therefore, the overall objective of the LAT is

$$\min_\theta \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}} \max_{\delta(\boldsymbol{x})} \mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{y'}\right), \quad \text{s.t.} \|\delta(\boldsymbol{x})\|_\text{p} \leq \epsilon \tag{1}$$

with loss function $\mathcal{L}$ and a instruction-following dataset $\mathcal{D}$. Notably, the L_p-norm constraint is usually implemented as L_2 norm and satisfies by the projected gradient descent (Geisler et al., 2024; Madry et al., 2018).

3.2 Observation: Embedding Distance is Predictive of Worst-case Prompt

Cao et al. (2024) has shown that worst-case prompts are difficult to detect using common metrics such as prompt perplexity (Gonen et al., 2023), Min-k probability (Shi et al., 2023a), principal component analysis of hidden states, or even the LLM's own preference. This suggests that identifying adversarial prompts remains a challenging problem. However, inspired by Zeng et al. (2024), we investigate the relationship between the distance of model hidden states and variations in model performance.

Setup Specifically, we use Llama-2-13b-chat(Touvron et al., 2023) to generate responses for the 100 test cases in RobustAlpaca (Cao et al.,

2024), where each case consists of an original user query ($\boldsymbol{x_0}$) and 10 paraphrases ($\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_{10}}$). To evaluate response quality, we employ ArmoRM (Wang et al., 2024b,a), a top-performing reward model from RewardBench (Lambert et al., 2024). We input both the original query and its paraphrases into Llama-2-13b-chat and extract the hidden states from the 20th layer (out of 40 total layers), using the last token's hidden state as the sequence embedding. The Euclidean distance (L_2 distance) is then computed between these embeddings. In Figure 3, we show the distributions of: (1) the average pairwise distance between paraphrases of the same query (average distance), and (2) the distance between the original query and its worst-case paraphrase (worst distance). Additionally, Figure 2 illustrates the correlation between worst distance and performance degradation when comparing the original query to its worst-case paraphrase.

Observation Figure 3 shows that the worst distance displays a significantly larger mean value over the average distance (student test, $p < 0.05$). Additionally, Figure 2 further verifies that the worst distance is correlated with the performance deterioration caused by the worst-case paraphrasing. The spearman's correlation coefficient is 0.36 (student test, $p < 0.05$). **Overall, the distance between a paraphrase and the original query can be an indicator for the worst-case prompt, and the worst-case prompt tends to be located in a relatively distant region from the original query.**

3.3 Our Method: Latent Adversarial Paraphrasing

Motivation Intuitively, if the worst-case paraphrasing of the user query is found, we could optimize the language model on it to improve the lower bound of instruction-following performance. However, even with the general pattern observed in Section 3.2, it is still a non-trivial task to construct the worst-case paraphrasing by prompting a powerful LLM since we can hardly control the embedding distance of the paraphrasing. Therefore, instead of synthesizing a human-readable textual paraphrase, we alternatively pursue a *continuous paraphrase* in the latent space with the LAT framework. Namely, the perturbed hidden states could act as the latent-space paraphrasing. However, the LAT framework is unable to constrain the perturbation to preserve the semantics of the original user query. In light of this, we propose our **LAP** framework.

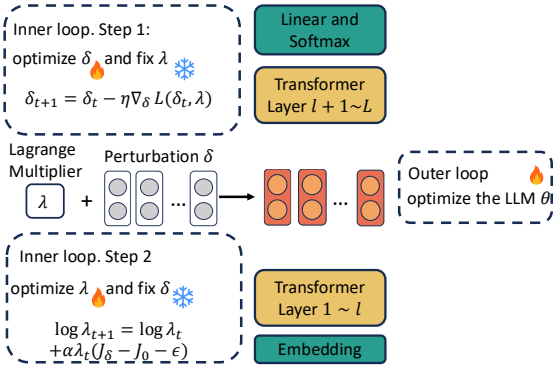


Figure 4: The workflow of our proposed **LAP** framework consists of an inner loop and an outer loop. For the inner loop, two steps are conducted iteratively to update the perturbation δ and the Lagrangian multiplier λ respectively, while for the outer loop the language model parameter θ is optimized.

Specifically, in the inner loop we target enlarging the embedding distance between the original query and the *continuous paraphrasing* in the latent space by maximizing the L_2-norm of the perturbation. Meanwhile, to preserve the original semantics, we additionally add a constraint on the increment of language modeling loss caused by the perturbation. The intuition behind this is that semantic-preserving paraphrasing generally will not increase the model perplexity or language modeling loss by a large margin (Liu et al., 2024). Mathematically, the optimization of our objective is:

$$\begin{aligned} &\max_{\delta(\boldsymbol{x})} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}} \|\delta(\boldsymbol{x})\|_p \\ &\text{s.t.} |\mathcal{J}_{\delta(\boldsymbol{x})} - \mathcal{J}_0| \leq \epsilon, \end{aligned} \tag{2}$$

where $\mathcal{J}_0$ and $\mathcal{J}_{\delta(x)}$ are the original loss and the loss with perturbation respectively:

$$\begin{aligned} \mathcal{J}_0 &= \mathcal{L}\left(f^\theta_{1\to L} \circ g^\theta(\boldsymbol{x}), \boldsymbol{y}\right) \\ \mathcal{J}_{\delta(\boldsymbol{x})} &= \mathcal{L}\left(f^\theta_{l+1\to L}\left(f^\theta_{1\to l} \circ g^\theta(\boldsymbol{x}) + \delta(\boldsymbol{x})\right), \boldsymbol{y}\right) \end{aligned} \tag{3}$$

During the training of the perturbation, the parameters of the language model are fixed. Then after the inner loop is optimized, the input-specific perturbation is fixed and the language model is optimized to minimize the language modeling loss with the inserted perturbation:

$$\min_\theta \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}} \mathcal{J}_{\delta(\boldsymbol{x})} \tag{4}$$

While the outer loop is straightforward, the inner loop is more complex. Different from the LAT

framework, the constraint within the inner loop cannot be implemented with the projected gradient descent. Therefore, to satisfy the constraint, we leverage the Lagrangian method (an optimization technique for finding the local minima of a function over a constraint set) to convert the constrained primal inner loop problem into its unconstrained Lagrangian counterpart:

$$\begin{aligned}&\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\min_{\delta(\boldsymbol{x})}\max_{\lambda(\boldsymbol{x})\geq 0}\mathcal{L}(\delta(\boldsymbol{x}),\lambda(\boldsymbol{x}))\\&\mathcal{L}(\delta(\boldsymbol{x}),\lambda(\boldsymbol{x}))=-\|\delta(\boldsymbol{x})\|_p+\lambda(|\mathcal{J}_{\delta(\boldsymbol{x})}-\mathcal{J}_0|-\epsilon),\end{aligned}\tag{5}$$

where $\lambda \geq 0$ serves as the Lagrange multiplier.

The constraint is transformed as a penalty term in the objective which can be dynamically modulated via the parameter λ and the perturbation δ. Specifically, to optimize the objective in Equation 5 during the min-max game, the perturbation is updated with

$$\delta(\boldsymbol{x})_{t+1}=\delta(\boldsymbol{x})_t-\eta\nabla_{\delta(\boldsymbol{x})}\mathcal{L}\left(\delta(\boldsymbol{x}),\lambda(\boldsymbol{x})\right)\tag{6}$$

Then the multiplier is optimized with

$$\log\lambda(\boldsymbol{x})_{t+1}=\log\lambda(\boldsymbol{x})_t+\alpha\lambda_t(\boldsymbol{x})\left(\mathcal{J}_{\delta(\boldsymbol{x})}-\mathcal{J}_0-\epsilon\right)\tag{7}$$

Through the min-max game between the perturbation and the Lagrangian multiplier, the transformed objective ensures that the violation of the constraint would incur a high value of λ. Since the perturbation δ and the λ have opposite objectives, consequently, the trained perturbation is adjusted to meet the constraint.

Apart from the inner loop optimization of perturbation δ and the outer loop optimization of the language model θ, we additionally introduce SFT on instruction-following data as in Equation 3 into our framework. We summarize the workflow of our approach in Algorithm 1.

4 Experiment

Backbone and Dataset We use Llama-3-8b (Dubey et al., 2024) and Mistral-7b-v0.3 (Jiang et al., 2023) as backbones for experiments. But **LAP** is agnostic to the base model backbone and can be applied to any pre-trained auto-regressive language model. The models are fine-tuned on the "chosen" completion of `ultrafeedback_binarized`[1], a pair-wise preference dataset preprocessed from UltraFeedback (Cui et al., 2024).

[1] https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized

Benchmark and Metrics To evaluate the prompt robustness of LLM, we leverage the RobustAlpaca benchmark (Cao et al., 2024). Established based on the TinyAlapca (Polo et al., 2024a), the benchmark contains 100 cases with each case containing the original user query x_0 and 10 semantic-preserving paraphrases $x_1, x_2, \ldots, x_{10}$. Response of GPT-4 to the original query y_b serves as the baseline for computing win-rate. Specifically, $y_0, y_1, \ldots, y_{10}$ are generated based on $x_0, x_1, \ldots, x_{10}$ respectively and we use the original win-rate $p_w(x_0, y_0, y_b)$, the best win-rate $\max_i p_w(x_0, y_i, y_b)$, the worst win-rate $\min_i(x_0, y_i, y_b)$, and average win-rate $\frac{1}{n}\sum_i p_w(x_0, y_i, y_b)$ as our evaluation metric, where $p_w(x, y_1, y_2)$ is the probability of y_1 is preferred over y_2 given the query x. We use GPT-4o-mini as our evaluator for its advantage over GPT-4 in LMSYS benchmarks (Chiang et al., 2023).

Baseline Method To verify the effectiveness of the proposed **LAP**, we draw a comparison with existing baseline methods. Apart from **vanilla SFT**, we also consider:

- **Data Augmentation** diversities the instruction-following data with augmented paraphrases. Specifically, we sample 10k queries from UltraFeedback, synthesize a paraphrase for each query with a proprietary LLM, and augment the synthesized paraphrases into the instruction-following data. The template for prompting paraphrase generation is borrowed from Cao et al. (2024).
- **Prompt Consistency** (Zhou et al., 2022) design a loss function to minimize the divergence of model outputs on different paraphrases. In our implementation, we reuse the 10k augmented paraphrases and distill the model response from one query to its paraphrase.
- **Latent Adversarial Training (LAT)** (Casper et al., 2024) includes a trainable perturbation into the hidden states and optimize the language model parameters to minimize the language modeling loss at the existence of perturbation.

Meanwhile, we also compare our approach with several inference-time algorithms including

- **Self-Refinement** (Cao et al., 2024) prompts the SFT-ed language model to first rewrite the query according to its own preference before generating a response to the query.
- **Universal Self-Consistency (USC)** (Chen et al., 2024) first generates multiple draft responses with different hyper-parameter settings using the SFT-ed language model and then prompts the same SFT-ed

	Llama-3-8b				Mistral-7b-v0.3			
	original win-rate	best win-rate	worst win-rate	average win-rate	original win-rate	best win-rate	worst win-rate	average win-rate
Training algorithms								
SFT	**15.06**	47.24	1.44	15.39	7.85	43.62	0.01	12.62
Data Augmentation	11.47	39.78	1.77	13.75	9.67	**44.33**	1.64	12.38
Prompt Consistency	10.78	37.76	**3.09**	11.64	8.68	42.85	1.00	10.50
LAT	10.97	45.23	2.04	14.98	11.02	37.32	1.38	12.06
Inference algorithms								
USC	8.14	45.91	1.01	15.22	6.47	39.3	**1.85**	12.25
Self-Refinement	4.02	28.82	1.00	6.74	9.96	29.82	0.00	7.40
Mixture-of-Agents	6.19	40.33	0.00	12.19	5.51	32.04	0.00	8.18
LAP	12.56	**49.79**	1.99	**15.48**	**11.05**	41.93	1.59	**14.05**

Table 1: Evaluation performance on RobustAlpaca (Cao et al., 2024) with Llama-3-8b and Mistrail-7b-v0.3. The numbers in bold are the best results and the numbers underlined are the second best results.

language model itself to choose the final output from the candidates. We generate 4 draft responses using different random seeds in out implementation.

• **Mixture-of-Agents** (Anonymous, 2025) is a recently proposed inference algorithm with layers of multiple agents and each agent provides a response given all the draft responses generated by the agents in the previous layers. We construct a 2-layer mixture-of-agents network with each layer comprising 4 SFT-ed models.

Experimental Results We report the evaluation performance of our proposed **LAP** and the baseline methods on Table 1. From the table, we can observe that (1) Our approach obtains the best results or the second best result for the original win-rate and the average win-rate on two backbones, verifying that our approach enhances model robustness to different paraphrasing of user queries. (2) Contrary to recent findings on the importance of inference-time scaling, the inference-time algorithms in the baseline methods do not exhibit an advantage over vanilla SFT. Notably, Mixture-of-Agents and Self-Refinement have the lowest results on the worst win-rate, possibly because they both rely on external capable LLM to aggregate the strengths and weaknesses of draft responses. (3) Even with the baseline methods or our proposed approach, there is still a large margin between the best win-rate and the worst win-rate, suggesting a large room for further improvement.

5 Analysis

Ablation Study To evaluate the importance of each module in our **LAP** framework and how they contribute to the overall performance, we perform an ablation study on the following variants: (1)$\lambda = 0$: we fix the value of λ to be zero and thus invalidate the constraint on language modeling loss; (2) $p = 1$: the supervised fine-tuning (line 15 in Algorithm 1) is removed from our framework; (3) $p = 0$: we do multi-tasking (line 15 in Algorithm 1) and use every datapoint for both adversarial training and SFT.

	original win-rate	best win-rate	worst win-rate	average win-rate
LAP	10.21	**53.39**	**4.98**	**17.94**
$p = 0$	14.52	48.55	4.67	17.49
$p = 1$	5.04	46.44	0.30	11.42
$\lambda = 0$	**10.42**	45.81	1.56	16.68

Table 2: Ablation experiments on RobustAlpaca with Llama-3-8b backbone. Numbers in bold are best results.

The experiment results on Llama-3-8b is shown in Table 2. From the table, we can observe that our **LAP** obviously outperforms the $\lambda = 0$ variants in most metrics, suggesting the necessity of the constraint on language modeling loss. Moreover, compared with the $p = 0$ variant that all data points are trained with SFT and the $p = 1$ variant that eliminates the SFT process, **LAP** obtains a better balance between objectives.

Downstream Task Performance To examine how our robustness enhancement approach affects downstream task performance, we evaluate the capacity of **LAP** on several benchmarks from huggingface open LLM leaderboard in comparison with baseline training algorithms. The evaluation results are shown in Table 3. Additionally, we examine and compare their instruction-following ability on MT-bench (Chiang et al., 2023), with the evaluation results presented in Table 4. We use

	MMLU	ARC-E	ARC-C	Hellaswag	Winogrande	TruthfulQA	Average
SFT	61.86	80.30	50.17	60.13	74.27	32.44	59.86
Data Augmentation	62.08	80.64	50.09	60.07	74.66	31.82	59.89
Prompt Consistency	62.28	80.81	50.60	60.10	74.66	32.44	60.15
LAT	61.61	62.84	49.57	60.00	74.51	33.17	56.95
LAP	61.92	81.02	52.13	60.38	74.66	32.44	60.43
SFT	58.79	78.32	45.39	60.96	74.98	33.90	58.72
Data Augmentation	58.81	78.49	45.65	60.74	74.74	33.54	58.66
Prompt Consistency	59.24	78.24	43.94	60.75	74.51	33.05	58.29
LAT	57.42	76.94	45.14	59.98	74.35	30.97	57.47
LAP	58.72	77.36	44.97	60.74	73.95	34.39	58.36

Table 3: Down-stream task evaluation results on Llama-3-8b (upper block) and Mistral-7b-v0.3 (bottom block). Our **LAP** preserves the performance on basic commonsense and knowledge, as the results are similar.

	1st turn	2nd turn	Average
SFT	5.34	4.04	4.69
Data Augmentation	5.26	3.96	4.61
Prompt Consistency	5.15	4.10	4.63
LAT	5.19	4.25	4.72
LAP	5.11	4.23	4.67

Table 4: Evaluation results on MT-bench (Chiang et al., 2023) using Llama-3-8b backbone. Our **LAP** preserves the performance on instruction-following as the results are similar.

	original win-rate	best win-rate	worst win-rate	average win-rate
Training algorithms				
SFT	12.47	37.03	0.04	13.24
Data Augmentation	8.43	40.36	2.50	13.34
Prompt Consistency	14.80	39.73	1.36	12.99
LAT	10.01	35.61	2.50	13.81
Inference algorithms				
USC	6.74	**44.95**	1.77	**14.81**
Self-Refinement	2.55	23.49	2.50	5.64
Mixture-of-Agents	5.47	36.96	0.00	7.82
LAP	**13.31**	37.08	**4.01**	14.08

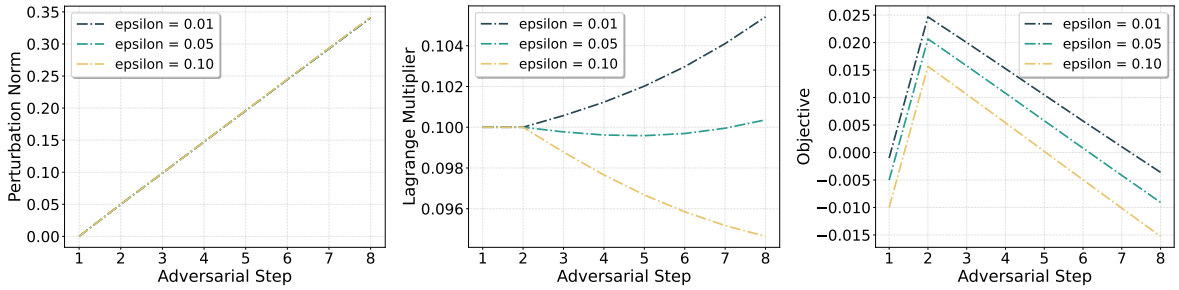
Table 5: Evaluation performance on RobustAlpaca (Cao et al., 2024) with Llama-2-13b. The numbers in bold are the best results and the numbers underlined are the second best results.

GPT-4o-mini as evaluator on MT-bench. From the two tables, we can observe that most baseline training algorithms will not result in a drastic change either downstream tasks or instruction-following benchmarks. As **LAP** is on par with and sometimes outperforms the baseline training algorithms, it is therefore verified that the improvement in robustness brought by our approach is not at the cost of performance deterioration on basic commonsense or instruction-following ability.

Scalability on Larger Backbones To explore whether the **LAP** framework could work on larger LLM backbones, we perform the experiments on Llama-2-13b (Touvron et al., 2023) and evaluate their performance using 40 cases randomly sampled from RobustAlpaca. The experimental results are presented in Table 5. As we can observe from the table, **LAP** obtains the best result and the second-best results on the original win-rate and the average win-rate respectively, suggesting that our approach remains effective on larger backbones. Meanwhile, it is worth noting that USC is a competitive baseline for Llama-2-13b, different from the experiment results on smaller models in Table 1. We gauge that Llama-2-13b is more capable in summarizing and aggregating candidate answers and therefore is able to obtain better performance with USC.

Effect of Perturbation Position To understand how the position of the perturbation (i.e., the transformer layer at which we incorporate the perturbation) impacts the effectiveness of our approach, we vary the positions of the perturbation among transformer layers in Mistral-7b-v0.3 (Jiang et al., 2023) and plot the trend of the best win-rate and the average win-rate in Figure 6. From the figure, it seems that Layer 12 and Layer 20 are slightly better than other layers as suitable positions for perturbation as the **LAP** optimization at the two positions yields the maximum best win-rate and the maximum average win-rate respectively. But overall, the performance on average win-rate is relatively insensitive to the position of the perturbation. In this study the perturbation is only inserted into one layer following previous works in LAT (Casper et al., 2024; Sheshadri et al., 2024). We leave the multi-layer perturbation to future works.

Analysis on Training Dynamics. To gain insight into the process of inner-loop optimization, we an-



(a) Dynamics of perturbation norm $\|\delta\|_2$ (b) Dynamics of Lagrange multiplier λ (c) Dynamics of $\mathcal{L}(\delta(\boldsymbol{x}), \lambda(\boldsymbol{x}))$

Figure 5: Training dynamics for the inner-loop optimization with different constraint margin ϵ on Llama-3-8b backbone.

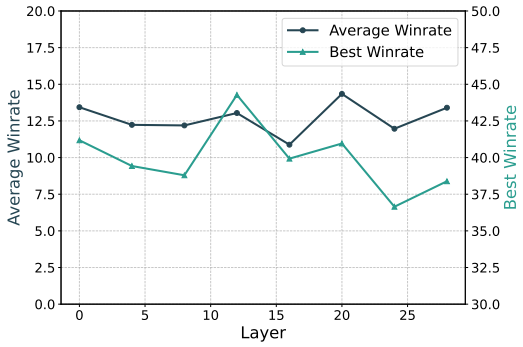


Figure 6: The trend of average win-rate and the best win-rate with the perturbation position on Mistral-7b-v0.3 backbone.

alyze the iteration of multiple variables within the training objective of Equation 5 and plot their trend in Figure 5. Specifically, for different values of the constraint margin ($\epsilon = 0.01,\ 0.05,\ 0.10$), we record the perturbation norm $\|\delta\|_2$, Lagrange multiplier λ and the objective $\mathcal{L}(\delta(\boldsymbol{x}), \lambda(\boldsymbol{x}))$ for each inner-loop iteration, and plot the trend of their average value among the instruction-following dataset. Notably, a smaller value of ϵ means a more tight constraint. As we can observe from Figure 5, the strength of the constraint has little effect on the perturbation norm as the perturbation norm steadily grows during the inner-loop training process whatever the value of ϵ is. However, the trend of Lagrange multiplier λ is obviously influenced by the ϵ since there is a monotonic decrease of λ when setting $\epsilon = 0.10$ but an opposite trend of λ could be observed when setting $\epsilon = 0.01$. We gauge the reason behind is that a tight constraint is harder to satisfy and thus pushes the Lagrange multiplier to be larger. It could also explain why a larger value of ϵ corresponds to a larger inner-loop objective in Figure 5c since the difference in objectives is dominated by the second term or the penalty term.

	original winrate	best winrate	worst winrate	average winrate
SFT+DPO	22.95	60.95	4.83	24.61
LAP +DPO	23.16	71.20	5.12	31.53

Table 6: The evaluation performance on RobustAlpaca with subsequent preference learning on Llama-3-8b backbone.

Compatibility with Preference Learning. To examine whether the proposed **LAP** framework is compatible with existing preference learning techniques, we perform direct preference optimization (DPO) (Rafailov et al., 2023) on SFT-ed model and the **LAP**-ed model to draw a comparison. As we can observe from the table, with subsequent preference learning our approach outperforms vanilla SFT by a large margin, verifying that our approach does not interfere with and is compatible with the subsequence preference learning.

6 Conclusion

In this study, we tackled the challenge of prompt robustness in LLMs, particularly their sensitivity to paraphrased user queries. Our findings revealed that the Euclidean distance between a paraphrased query and the original input correlates with response quality. Leveraging this insight, we introduced **LAP**, a novel adversarial framework that continuously searches for latent paraphrases and optimizes LLM parameters to enhance robustness. Future work will explore extending **LAP** to more models and datasets, investigating its applicability to multi-turn dialogues, and refining adversarial training techniques to further enhance model resilience.

Limitations

The limitations of this study can be summarized as below:

- In this work, we mainly consider improving the prompt robustness of language models in text modality, while the prompt robustness for large vision-large model and other modalities is left for future work.
- We generally utilize LoRA (Hu et al., 2022) as a parameter-efficient fine-tuning (PEFT) technique for SFT and do not perform experiments with other PEFT techniques such as adapter (Houlsby et al., 2019) or IA3 (Liu et al., 2022) or full-parameter fine-tuning.

Ethical Considerations

This work on improving prompt robustness through **LAP** has significant implications for the reliable deployment of large language models in real-world applications. By addressing the fundamental challenge of prompt sensitivity, this approach could enhance the consistency and reliability of LLM outputs across different but semantically equivalent user queries. This improvement is particularly crucial for high-stakes applications such as healthcare, legal assistance, and educational systems, where inconsistent responses could lead to harmful outcomes. However, the development of more robust LLMs also raises important ethical considerations. While increased robustness could help prevent accidental failures and reduce unwanted biases triggered by prompt variations, it might also make models more consistently capable of harmful outputs if misused.

Acknowledgment

We thank Stephen Casper, Javier Rando, Ryuto Koike and Samuele Marro for feedback and comments.

References

Anonymous. 2025. Mixture-of-agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2023. Llemma: An open language model for mathematics. *ArXiv*, abs/2310.10631.

Fazl Barez, Tingchen Fu, Ameya Prabhu, Stephen Casper, Amartya Sanyal, Adel Bibi, Aidan O'Gara, Robert Kirk, Ben Bucknall, Tim Fist, Luke Ong, Philip Torr, Kwok-Yan Lam, Robert Trager, David Krueger, Sören Mindermann, José Hernandez-Orallo, Mor Geva, and Yarin Gal. 2025. Open problems in machine unlearning for ai safety. *Preprint*, arXiv:2501.04952.

Bowen Cao, Deng Cai, Zhisong Zhang, Yuexian Zou, and Wai Lam. 2024. On the worst prompt performance of large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Stephen Casper, Lennart Schulze, Oam Patel, and Dylan Hadfield-Menell. 2024. Defending against unforeseen failure modes with latent adversarial training. *arXiv preprint arXiv:2403.05030*.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2024. Universal self-consistency for large language models. In *ICML 2024 Workshop on In-Context Learning*.

Jiale Cheng, Xiao Liu, Kehan Zheng, Pei Ke, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. 2024. Black-box prompt optimization: Aligning large language models without model training. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3201–3219, Bangkok, Thailand. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: Boosting language models with high-quality feedback.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Stanislav Fort. 2023. Scaling laws for adversarial attacks on language model activations. *arXiv preprint arXiv:2312.02780*.

Simon Geisler, Tom Wollschläger, MHI Abdalla, Johannes Gasteiger, and Stephan Günnemann. 2024. Attacking large language models with projected gradient descent. *arXiv preprint arXiv:2402.09154*.

Hila Gonen, Srini Iyer, Terra Blevins, Noah Smith, and Luke Zettlemoyer. 2023. Demystifying prompts in language models via perplexity estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10136–10148, Singapore. Association for Computational Linguistics.

Jiasheng Gu, Hongyu Zhao, Hanzi Xu, Liangyu Nie, Hongyuan Mei, and Wenpeng Yin. 2023. Robustness of learning from task instructions. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13935–13948, Toronto, Canada. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. *ArXiv*, abs/1902.00751.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Tiansheng Huang, Sihao Hu, and Ling Liu. 2024. Vaccine: Perturbation-aware alignment for large language models against harmful fine-tuning attack. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew E. Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hanna Hajishirzi. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *ArXiv*, abs/2311.10702.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L'elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *ArXiv*, abs/2310.06825.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi,

Noah A. Smith, and Hannaneh Hajishirzi. 2024. Rewardbench: Evaluating reward models for language modeling. *Preprint*, arXiv:2403.13787.

Ariel N. Lee, Cole J. Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *ArXiv*, abs/2205.05638.

Qin Liu, Fei Wang, Nan Xu, Tianyi Lorena Yan, Tao Meng, and Muhao Chen. 2024. Monotonic paraphrasing improves generalization of language model prompting. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9861–9877, Miami, Florida, USA. Association for Computational Linguistics.

Ruotian Ma, Xiaolei Wang, Xin Zhou, Jian Li, Nan Du, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. Are large language models good prompt optimizers? *arXiv preprint arXiv:2402.02101*.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

OpenAI. 2023. GPT-4 technical report.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 2024a. tinybenchmarks: evaluating llms with fewer examples. *arXiv preprint arXiv:2402.14992*.

Felipe Maia Polo, Ronald Xu, Lucas Weber, Mírian Silva, Onkar Bhardwaj, Leshem Choshen, Allysson Flavio Melo de Oliveira, Yuekai Sun, and Mikhail Yurochkin. 2024b. Efficient multi-prompt evaluation of LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3845–3864, Dubrovnik, Croatia. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, and 651 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *ArXiv*, abs/2403.05530.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, I. Evtimov, Joanna Bitton, Manish P Bhatt, Cristian Cantón Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre D'efossez, Jade Copet, and 6 others. 2023. Code llama: Open foundation models for code. *ArXiv*, abs/2308.12950.

Abel Salinas and Fred Morstatter. 2024. The butterfly effect of altering prompts: How small changes and jailbreaks affect large language model performance. *arXiv preprint arXiv:2401.03729*.

Swami Sankaranarayanan, Arpit Jain, Rama Chellappa, and Ser Nam Lim. 2018. Regularizing deep networks using efficient layerwise adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*.

Abhay Sheshadri, Aidan Ewart, Phillip Guo, Aengus Lynch, Cindy Wu, Vivek Hebbar, Henry Sleight, Asa Cooper Stickland, Ethan Perez, Dylan Hadfield-Menell, and 1 others. 2024. Latent adversarial training improves robustness to persistent harmful behaviors in llms. *arXiv preprint arXiv:2407.15549*.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023a. Detecting pretraining data from large language models. *Preprint*, arXiv:2310.16789.

Weijia Shi, Xiaochuang Han, Hila Gonen, Ari Holtzman, Yulia Tsvetkov, and Luke Zettlemoyer. 2023b. Toward human readable prompt tuning: Kubrick's the shining is a good movie, and a good prompt too? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10994–11005, Singapore. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Jiuding Sun, Chantal Shaib, and Byron C Wallace. 2024. Evaluating the zero-shot robustness of instruction-tuned language models. In *The Twelfth International Conference on Learning Representations*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Anton Voronov, Lena Wolf, and Max Ryabinin. 2024. Mind your format: Towards consistent evaluation of in-context learning improvements. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6287–6310, Bangkok, Thailand. Association for Computational Linguistics.

Haoran Wang and Kai Shu. 2024. Trojan activation attack: Red-teaming large language models using steering vectors for safety-alignment. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2347–2357.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. 2024a. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. In *ACL*.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024b. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *ArXiv*, abs/2406.12845.

Lucas Weber, Elia Bruni, and Dieuwke Hupkes. 2023. Mind the instructions: a holistic evaluation of consistency and interactions in prompt-based learning. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 294–313, Singapore. Association for Computational Linguistics.

Sophie Xhonneux, Alessandro Sordoni, Stephan Günnemann, Gauthier Gidel, and Leo Schwinn. 2024. Efficient adversarial training in LLMs with continuous attacks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*.

Yi Zeng, Weiyu Sun, Tran Huynh, Dawn Song, Bo Li, and Ruoxi Jia. 2024. BEEAR: Embedding-based adversarial removal of safety backdoors in instruction-tuned language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13189–13215, Miami, Florida, USA. Association for Computational Linguistics.

Chunting Zhou, Junxian He, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Prompt consistency for zero-shot task generalization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2613–2626, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, and 1 others. 2023. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.

	Llama-3-8b	Mistral-7b-v0.3	Llama-2-13b
Precision	bfloat16	bfloat16	bfloat16
max sequence length	1024	1024	1024
Batch size	32	32	32
Optimizer	AdamW	AdamW	AdamW
Adam (β_1, β_2)	$(0.9, 0.95)$	$(0.9, 0.95)$	$(0.9, 0.95)$
Learning rate	1e-4	1e-4	1e-4
Warmup ratio	0.1	0.1	0.1
Decay style	cosine	cosine	cosine
Weight decay	0.0	0.0	0.0
Training step	1 epoch	1 epoch	1 epoch
LoRA rank	64	64	64
LoRA alpha	16	16	16
LoRA dropout	0.05	0.05	0.05
LoRA modules	gate_proj, up_proj, down_proj	gate_proj, up_proj, down_proj	gate_proj, up_proj, down_proj

Table 7: Hyper-parameter settings for supervised fine-tuning and preference learning.

A Hyper-parameter Setting

Our experiments are conducted on a cloud Linux server with Ubuntu 16.04 operating system. The codes are written in Python 3.10 with the huggingface libraries[2]. We run our experiments on an Nvidia Tesla A100. The detailed hyper-parameter settings for different backbones are shown in Table 7, which mostly follows Lee et al. (2023) and Ivison et al. (2023). At inference, we use nucleus sampling with $p = 0.9$ and temperature $T = 1.0$. vLLM [3] is adopted for accelerating response generation.

B Training Algorithms

The training algorithm for our proposed **LAP** is shown in Algorithm 1.

C More Details About Baseline Implementation

In this section, we elaborate on the template used for inference-based algorithms and the template for synthesizing paraphrases of user queries. Specifically, we adopt the template from Cao et al. (2024) to produce semantic-preserving paraphrases and the template is shown below.

Algorithm 1 The proposed **LAP** framework.

Input: An instruction-following dataset $\mathcal{D}$, a language model θ, the number of iterations in the inner loop T, the sampling coefficient p.
for $\{(\boldsymbol{x}, \boldsymbol{y})\} \sim \mathcal{D}$ **do**
  Sample $I \sim \text{Bernouli(p)}$
  if $I = 1$ **then**
    Perform Adversarial Training
    Compute the language modeling loss with Eq 3.
    Inner loop
    for $j \leftarrow 1$ to T **do**
      Fix the multiplier λ and optimize a perturbation δ using Equation 6.
      Fix the perturbation δ and update the multiplier λ according to Equation 7.
    end for
    Outer loop
    Fix the perturbation δ and optimize the language model θ following Eq. 2.
  end if
  if $I = 0$ **or** $p = 0$ **then**
    Perform Supervised Fine-tuning
    Optimize the language model θ on $(\boldsymbol{x}, \boldsymbol{y})$ with language modeling loss following 3.
  end if
end for
Return: language model θ.

[2] https://github.com/huggingface/trl
[3] https://github.com/vllm-project/vllm

Prompt template for constructing semantic-preserving paraphrases.

Your task is to generate one unique paraphrase for the given query, ensuring that the meaning of the paraphrase remains consistent, and the structure is significantly altered. Do not introduce any new information that isn't present in the original query, and avoid omitting any crucial information from the original query, particularly any specific requirements about the output content, style, format, options, or any numbers or data. The responses to your paraphrases and the original query should be identical. If the original query contains quoted or referenced content, such as 'rewrite the following sentence: '<reference>' and those enclosed in '

', you should retain this referenced content in your paraphrases and rephrase the rest of the text. Your task is not to answer the query, but solely to rephrase it. Please provide unique and creative rephrasing.
The examples provided below are for illustrative purposes only.
Examples:
Query: "Why is it that only proteins, carbohydrates and fats are deemed to have caloric value?
Just to clarify - consider the hypothetical situation below:
100g of 'carbohydrates' = 400kCal
100g of 'protein' = 400kCal
100g of 'fat' = 900kCal
100g of 'vitamins, minerals and/or other electrolytes' = ?"
Output: Why are proteins, carbohydrates, and fats the only substances considered to have caloric value? To provide further clarification, let's consider the hypothetical scenario below: If 100g of carbohydrates provide 400kCal, 100g of protein provides 400kCal, and 100g of fat provides 900kCal, what would be the caloric value of 100g of vitamins, minerals, and/or other electrolytes?

Query: "I have this text:

We use professional processes. Use the photo delivery system, check the quality and send it to the customer immediately. We started editing digital photos from the first wave of 2000. Started with editing personal photos, and then spread our passion by editing wedding photos or restoring old photos for loved ones, friends and family. friends and customers. 2005 started a local photo shop and started commercial photo editing, and product photography for manufacturers. 2007 implemented online projects on freelancer platforms.

Rewrite it in professional style to tell a touching story."
Output: I possess this text:

We use professional processes. Use the photo delivery system, check the quality and send it to the customer immediately. We started editing digital photos from the first wave of 2000. Started with editing personal photos, and then spread our passion by editing wedding photos or restoring old photos for loved ones, friends and family. friends and customers. 2005 started a local photo shop and started commercial photo editing, product photography for manufacturers. 2007 implemented online projects on freelancer platforms.

Rephrase it in a professional manner to narrate a moving story.

Query: "I want you to act as a Tiktok Ads Keywords generator. Your task is to come up with keywords that can be used in Tiktok advertising campaigns. The keywords should be relevant to the target audience and the product or service being advertised. Your responses should not include more than 5 keywords per request."
Output: Your role will be to serve as a TikTok Ads Keywords generator. Your objective is to generate relevant keywords that can be utilized in TikTok advertising campaigns. These keywords should effectively target the desired audience and align with the product or service being promoted. Please provide no more than five keywords per request in your responses.
Now, please generate a paraphrase for the given query.
Query: {query}
Output:

Figure 7: Prompt template for constructing semantic-preserving paraphrases.

Prompt template for Self-Refinement.

Given the following instruction: {query}, paraphrase the instruction to be more natural and then generate a response. Start the paraphrase with "Paraphrase: " and start your response with "Response: " (without quotes)

Figure 8: Prompt template for Self-Refinement (Cao et al., 2024).

Prompt template for USC.

Evaluate these responses.
Select the most consistent response based on majority consensus.
Answer with "The most consistent response is Response X" (without quotes).

Response 1: {draft response}
Response 2: {draft response}
Response 3: {draft response}
Response 4: {draft response}

Figure 9: The prompt template for USC (Chen et al., 2024).

Prompt template for Mixture-of-Agents.

You have been provided with a set of responses from various open-source models to the latest user query. Your task is to synthesize these responses into a single, high-quality response. It is crucial to critically evaluate the information provided in these responses, recognizing that some of it may be biased or incorrect. Your response should not simply replicate the given answers but should offer a refined, accurate, and comprehensive reply to the instruction. Ensure your response is well-structured, coherent, and adheres to the highest standards of accuracy and reliability.

Responses from models:
Response 1: {draft response}
Response 2: {draft response}
Response 3: {draft response}
Response 4: {draft response}
Output the synthesized response directly without any prefix.

Figure 10: The prompt template for Mixture-of-Agents (Anonymous, 2025).