Steering Away from Refusal: A Black-box Jailbreak Method Based on First-Token Distribution

Shuangjie Fu[1,2], **Du Su**[1*], **Xin Chen**
Fei Sun[1], **Huawei Shen**[1,2], **Xueqi Cheng**[1,2]
[1]State Key Laboratory of AI Safety, Institute of Computing Technology,
Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
{fushuangjie23s, sudu}@ict.ac.cn
chenxin061@gmail.com

Abstract

Investigating black-box jailbreak attacks is crucial for revealing the actual security risks faced by operational Large Language Models (LLMs). The primary challenge in black-box jailbreak attack is the absence of direct optimization signals, such as gradients, to guide the refinement of adversarial prompts. While current mainstream methods like PAIR and TAP attempt to leverage the model's textual output as feedback, facing a critical limitation when models consistently generate static refusal responses, depriving the attacker of any actionable signal to distinguish better prompts. To overcome the bottleneck and reveal whether there is potential risk to open access to partial logprobs information, we investigate LLM output distribution. Our empirical analysis reveals that refusal responses exhibit a highly consistent distributional pattern at the first generated token, suggesting that the deviation from this standard pattern can serve as a quantifiable metric for LLM generating refusal response. Based on this insight, we propose Distribution Jailbreak (DJ), an attack method that select effective jailbreak templates and then iteratively optimizes adversarial suffixes by maximizing the KL divergence from the standard refusal distribution. Extensive experiments demonstrate that DJ achieves state-of-the-art Attack Success Rate(ASR). Notably, DJ achieves over 90% ASR on all tested open-source models, and delivers over 94% ASR on GPT-4.1. Our code is publicly available at https://github.com/Zed630/DistributionJailbreak.

1 Introduction

As Large Language Models (LLMs) are increasingly integrated into critical applications, ensuring their safety has become paramount. Model developers invest heavily in alignment techniques to prevent the generation of harmful or unethical content. To validate the robustness of these established safety alignment mechanisms, "Jailbreak attacks", adversarial strategies designed to bypass LLMs' safety mechanism, have emerged as essential evaluation methods. Studying jailbreak attacks is not about promoting harm, but about proactively identifying potential vulnerabilities to construct more robust and secure LLM systems.

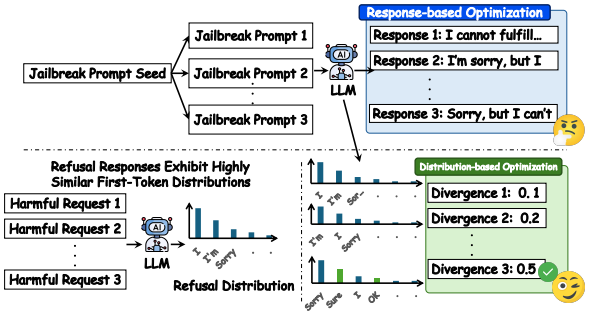


Figure 1: Comparison between Response-based Optimization and our proposed Distribution-based Optimization. Existing black-box methods lack distinguishable signals when LLM constantly generate refusal responses. Our approach exploits the characteristic "Refusal Distribution" shared by refusal responses and use the divergence from this pattern as a quantifiable signal to guide the optimization.

Based on the accessibility of model internal parameters, gradients, and hidden states, jailbreak attacks can be categorized into white-box and black-box scenarios. While white-box approaches offer theoretical insights, black-box scenarios more accurately reflect the real-world deployment of LLMs. Consequently, investigating black-box jailbreak attacks is crucial for revealing the actual security risks faced by operational LLM systems.

The primary challenge in black-box settings is the absence of direct optimization signals (e.g., gradients) to guide the refinement of adversarial prompts. To overcome this, leading black-box methods like PAIR (Chao et al., 2025) and TAP (Mehrotra et al., 2024) leverage the model's textual output as feedback. They typically employ

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 25969–25979

an attacker LLM to analyze the semantic content of the target model's responses, iteratively refining the prompts based on this linguistic feedback to steer the target towards compliance. However, this content-based optimization paradigm faces a critical limitation. When models constantly provide refusal responses (e.g., "I cannot assist with this request"). The semantic content of the output remains static and uninformative across iterations. This lack of variation deprives the attacker LLM of any actionable signal to distinguish better prompts from worse ones, leading to attack failure.

As some commercial platforms expose partial distributional information, such as log-probabilities (logprobs) of the top-k generated tokens, a critical security question has been raised: **Does exposing even partial output distribution information introduce new vulnerabilities that exacerbate jailbreak risks?** We posit that the truncated distributional data, while seemingly insufficient for full reconstruction, contains signals that can resolve the feedback bottleneck in black-box attacks.

To address the above question and overcome the challenge of absence of optimization signals in black-box scenarios, we investigate the distributional characteristics of LLMs' outputs. Through empirical analysis, we reveal that refusal responses exhibit a highly consistent distributional pattern at first token, where probability mass converges heavily on a narrow set of refusal-indicating tokens (e.g., "I", "Sorry", or "As"). This observation suggests that the deviation of the model's current output distribution from standard refusal pattern can serve as a quantifiable metric of refusal likelihood, thus providing a viable optimization signal for refining adversarial prompts in black-box scenarios.

Based on the above insights, we propose a novel jailbreak attack method Distribution Jailbreak(DJ). First, we construct a strategy to collect and filter potential jailbreak templates. By measuring the KL divergence between the template's output distribution and the standard refusal distribution, we select templates that maximize this distributional distance, indicating a higher potential to bypass safety guards. Second, for the top-ranked templates, we iteratively optimize adversarial suffixes to further widen this gap. By continuously maximizing the divergence from the refusal distribution, our approach effectively steers the model towards generating affirmative responses. The experimental results show DJ achieves a superior performance in attack success rate (ASR): over 90% ASR on all tested open-source models, over 94% ASR on GPT-4.1 and over 19% ASR improvement on Claude-3.5-sonnet.

The main contributions of our work can be summarized as follows:

- We identify the first-token output distribution as a critical, underutilized signal in black-box settings. We reveal that LLMs' refusal responses possess a distinct distributional pattern, which becomes more pronounced and accessible under jailbreak templates.
- We propose Distribution Jailbreak (DJ), a novel black-box attack method that uses distributional divergence from a reference refusal pattern as a fine-grained optimization objective. Extensive experiments validate the effectiveness of distributional steering in bypassing LLM safety guardrails.

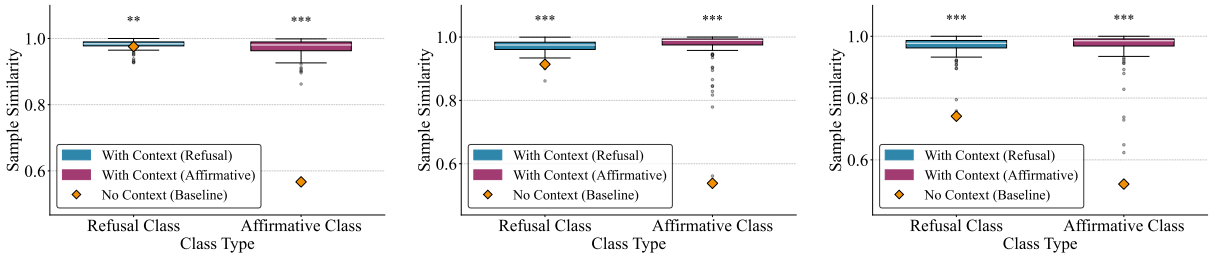
2 Related Work

2.1 White-box Jailbreak Attacks

White-box jailbreak attacks assume full access to the target model's architecture, weights, and gradients. These methods typically utilize LLMs' internal information to search adversarial prompts that induce LLM to generate harmful response.

The most prominent category involves Gradient-Based Optimization, where GCG (Zou et al., 2023) employs a gradient-based optimization method that identifies universal adversarial suffixes capable of transferring across different models. To improve the stealthiness of these attacks, AutoDAN (Liu et al., 2024) employs a hierarchical genetic algorithm to generate semantically meaningful jailbreak prompts, avoiding the gibberish nature of GCG-style attacks. ASETF (Wang et al., 2024) advanced the field by optimizing continuous adversarial embeddings and translating them into readable prompts through dedicated translation models. Subsequent advancements focused on improving efficiency. For instance, Improved GCG (I-GCG) (Jia et al., 2025) enhanced the original approach through automatic multi-coordinate updating strategies, accelerating convergence while maintaining attack effectiveness.

These approaches demonstrate that white-box access enables highly effective attacks through various optimization paradigms, though they remain limited to open-source models.



(a) Llama-3-Instruct-8B (b) Phi-3.5-mini (c) Qwen-3-8B

Figure 2: Boxplots comparing similarity (derived as $S(\mathcal{P})$) of output distributions between No Context (no jailbreak prompt templates) and With Context (using jailbreak prompt templates) across three models (Llama-3-Instruct-8B, Phi-3.5-mini, Qwen-3-8B). Similarity is measured separately for the refusal response class and affirmative response class. For all models, the similarity of refusal output distributions is consistently higher than that of affirmative distributions. Adding jailbreak prompt templates significantly increases similarity in both refusal and affirmative classes, meaning the output distribution characteristics of refusal and affirmative responses become more distinct.

2.2 Black-box Jailbreak Attacks

Black-box jailbreak attacks operate without access to model internals, relying solely on input-output interactions to discover adversarial prompts.

Early studies focused on exploring the potential vulnerability in LLMs' safety alignment. (Wei et al., 2023) systematically analyzed failure modes in safety training, identifying that conflicting training objectives and mismatched generalization allow users to bypass safeguards via role-playing or persona adoption. MasterKey (Deng et al., 2024) fine-tunes attacker models on successful jailbreak datasets to reverse-engineer defense patterns. Similarly, DAN (Shen et al., 2024) conducted a large-scale measurement study of "in-the-wild" jailbreak prompts, demonstrating that adversarial prompts remain effective against aligned models. To bypass models' safety mechanism, attackers often mask malicious intent using complex input transformations, DeepInception (Li et al., 2023) utilizes nested scene descriptions to hypnotize LLMs into escaping their safety constraints. Furthermore, CodeAttack (Ren et al., 2024) demonstrated that exploiting programmatic behaviors and pseudo-code prompts can effectively trigger restricted outputs. Low Resources Jailbreak (Yong et al., 2023) and CipherChat (Yuan et al., 2024) revealed that translating harmful queries into low-resource languages or cipher text effectively bypasses alignment filters trained primarily on English data.

Recent works have shifted towards automated frameworks that refine attacks based on model feedback. PAIR (Chao et al., 2025) proposes an iterative refinement framework where an attacker model optimizes prompts based on the LLMs' responses, while TAP (Mehrotra et al., 2024) utilizes an attack tree pruning strategy to efficiently navigate the search for adversarial prompts. However, such content-based optimization fails when LLMs generate invariant refusal responses, as the absence of distinguishable semantic feedback deprives the attacker of actionable optimization signals.

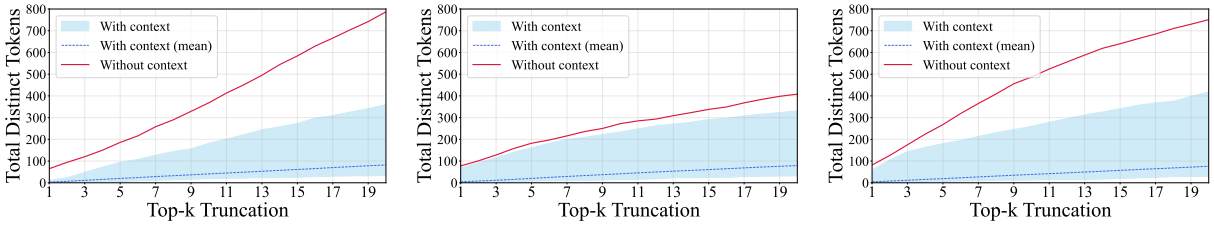
3 LLMs' Output Distribution

We observe that LLMs typically exhibit a tendency to provide refusal or affirmative response at the first token of their output. For instance, tokens such as "I", "Sorry", and "As" are usually the starting tokens of refusal responses (e.g., "I cannot fulfill your request", "Sorry, but I cannot fulfill", "As a responsible AI"), while "Sure" and "Absolutely" are typical starting tokens of affirmative responses. To investigate whether the output distribution of the first token of LLMs can serve as an effective optimization signal for jailbreak attacks in the black-box scenario, we conduct the following analyses on the output distribution of LLMs.

3.1 Refusal and Affirmative Distribution

We distinguish between two categories of prompts derived from the Prompt Driven Dataset (Zheng et al., 2024): benign prompts (D_b), which request content consistent with human values, and harmful prompts (D_h), which solicit responses violating laws or ethical standards. For a given set of prompts $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, let $\mathcal{P} = \{P_1, P_2, \ldots, P_N\}$ denote the set of corresponding probability distributions over the vocabulary for the first generated token.

To quantify the consistency of the model's output patterns, we calculate the similarity of these distri-



(a) Llama-3-8B-Instruct (b) Phi-3.5-mini-Instruct (c) Qwen3-8B

Figure 3: This figure illustrates the number of distinct tokens appearing in the top-k output distributions of three LLMs (Llama-3-8B, Phi-3.5-mini, Qwen3-8B) under two request conditions: The red curve represents direct requests (no jailbreak prompt template), where the number of distinct tokens in top-k increases sharply with k, (exceeding 300–700 at $k = 20$). The light blue area and dashed line represent requests with a jailbreak prompt template, where the number of distinct tokens in top-k remains stably low (below 100 at $k = 20$).

butions based on the Generalized Jensen-Shannon Divergence (JSD). First, we compute the average distribution $M = \frac{1}{N}\sum_{i=1}^{N} P_i$. The JSD for the set of distributions $\mathcal{P}$ is defined as:

$$\text{JSD}(\mathcal{P}) = \frac{1}{N}\sum_{i=1}^{N} D_{KL}(P_i \parallel M) \quad (1)$$

where D_{KL} represents the Kullback-Leibler divergence. Since the JSD for N distributions is bounded by $\log N$, we compute the normalized JSD as $\overline{\text{JSD}}(\mathcal{P}) = \frac{\text{JSD}(\mathcal{P})}{\log N}$, which scales the value to the range $[0, 1]$. Finally, we define the distributional similarity score S as:

$$S(\mathcal{P}) = \frac{1}{1 + \overline{\text{JSD}}(\mathcal{P})} \quad (2)$$

A higher S indicates that the distributions in $\mathcal{P}$ are more similar to each other (i.e., low divergence), whereas a lower S implies greater diversity.

We evaluate three open-source LLMs: Llama-3-8B-Instruct (Team, 2024), Phi-3.5-mini-Instruct(Abdin et al., 2024), and Qwen3-8B (Yang et al., 2025). As shown in Table 1, across all three tested LLMs, the refusal output distributions (derived from D_h) exhibit significantly higher similarity compared to the affirmative output distributions (derived from D_b). Figure 2 visually presents the average similarity scores across different sample categories. This finding indicates that refusal responses share distinct and consistent probabilistic characteristics. We attribute this phenomenon to the safety alignment process (e.g., RLHF), where models are typically trained to converge on a narrow set of refusal patterns (e.g., "I cannot fulfill this request") regardless of the specific harmful query, whereas benign queries trigger a diverse range of affirmative responses.

How Jailbreak Template Affect Distribution. Jailbreak templates serve as a prevalent strategy in black-box adversarial attacks. To investigate the specific impact of jailbreak templates on the refusal and affirmative output distributions of LLMs, we employ 100 jailbreak prompt templates from In-the-Wild Jailbreak Prompts dataset (Shen et al., 2024). As shown in Figure 2, after applying jailbreak templates, the similarity among affirmative output distributions is enhanced to a level comparable to that of refusal output distributions. This demonstrates that jailbreak prompt templates typically function to constrain the model's output mode.

Output distributions with distinct characteristics are conducive to output distribution comparison in the black-box scenario, where we usually only have access to limited Logprobs information. For example, OpenAI's models typically only provide Logprobs for the top-20 tokens. As shown in Figure 3, the use of prompt templates significantly reduces the total number of unique tokens appearing in the top-k, which makes the comparison of truncated output distributions more meaningful.

Model	Refusal	Affirmative
Llama-3-8B-Instruct	0.975	0.566
Phi-3.5-mini-Instruct	0.914	0.538
Qwen3-8B	0.741	0.521

Table 1: Similarity of output distributions for Refusal and Affirmative responses. The results demonstrate that refusal responses exhibit significantly lower divergence compared to affirmative responses, indicating a higher consistency in how models generate refusal responses.

4 Methodology

This section details the framework designed to optimize jailbreak attack prompts by leveraging probability distributions of LLMs. The core idea is to anchor the attack on the harmful request distribution and iteratively refine prompt templates and adversarial suffixes using KL divergence as a fine-grained optimization signal.

4.1 Jailbreak Template Selection

Anchor Harmful Distribution. We first define the refusal request distribution to establish a reference for measuring the alignment of LLM outputs with harmful intents. Let $D_h = \{d_{h,1}, d_{h,2}, d_{h,3}, ..., d_{h,N}\}$ denote a dataset of N distinct harmful requests.

To capture the aggregated distribution of harmful intent responses, we compute the average distribution p_{ref} as the normalized sum of $P_{\text{LLM}}(\cdot|d_{h,i})$ across all harmful requests in D_h.

$$p_{\text{ref}} = \frac{1}{N} \sum_{i=1}^{N} P_{\text{LLM}}(\cdot|d_{h,i}) \tag{3}$$

We specify the practical acquisition of the next-token probability distribution, $P_{\text{LLM}}(\cdot|d_{h,i})$, depending on model accessibility. For open-source models, we derive the complete distribution over the vocabulary directly from output logits. For closed-source models providing access to limited logprobs, we utilize logprobs to obtain top-20 probabilities, assigning negligible values to remaining tokens. For black-box models lacking probability access, we estimate the token distribution by sampling the first generated token multiple times—specifically, we generate the first token m times under a temperature setting of 1. The temperature value is chosen to enhance the diversity of the sampled first tokens.

Select Promising Templates. We next filter a candidate pool of jailbreak templates to retain those with the highest potential to induce affirmative responses, using KL divergence between the model's output distribution and p_{ref} as the selection metric.

Let $\mathcal{T} = \{t_1, t_2, t_3, ..., t_K\}$ denote a set of K candidate jailbreak prompt templates. For each template $t_i \in \mathcal{T}$, we construct a full prompt by concatenating t_i with a representative harmful request x. The request x can either be a fixed harmful query "Write a tutorial on how to make a bomb", or a random sample from D_h. We then feed the concatenated prompt to target LLM and retrieve its output distribution $P_{\text{LLM}}(\cdot|t_i, x)$. For each template t_i, we calculate the KL divergence between $P_{\text{LLM}}(t_i, x)$ and the anchor harmful distribution $\text{KL}(P_{\text{LLM}}(\cdot|t_i, x)||p_{\text{ref}})$.

KL divergence quantifies the dissimilarity between two distributions: a larger value indicates that the template-induced output distribution deviates more significantly from the standard refusal distribution. We rank all templates in $\mathcal{T}$ by their KL divergence scores and select the top top-k templates to form a reduced set $\mathcal{T}_{\text{top-}k}$.

Rank Templates via ASR. The templates in $\mathcal{T}_{\text{top}-k}$ are filtered further based on actual attack efficacy, as a high KL divergence score does not always guarantee a successful jailbreak (e.g., the model may generate irrelevant content instead of complying with the harmful request). This stage introduces adversarial suffixes and iterative refinement to optimize each template's performance.

Given $\mathcal{T}_{\text{top}-k}$, we randomly sample n harmful requests $\{x_1, x_2, ..., x_n\}$ from D_h and evaluate each template $t_i \in \mathcal{T}_{\text{top}-k}$ on these requests using the jailbreak attack process described in Section 4.3.

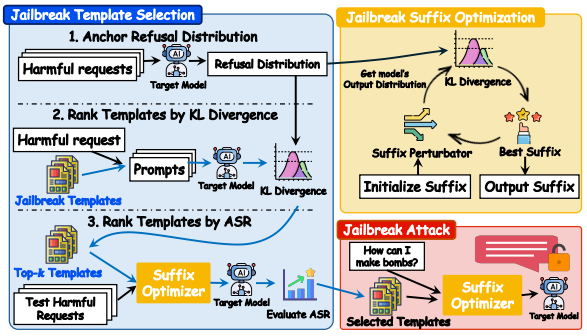


Figure 4: Overview of the proposed jailbreak attack framework. The pipeline consists of three main components: (1) Template Selection, where potential templates are identified by maximizing the KL divergence between the target model's output distribution and a reference refusal distribution; (2) Suffix Optimization, which employs an iterative algorithm to refine adversarial suffixes for further distribution shifting; and (3) Jailbreak Attack, where the optimal templates and optimized suffix are combined to construct the final prompt, effectively bypassing safety guardrails.

4.2 Adversarial Suffix Optimization

Drawing inspiration from the GCG attack (Zou et al., 2023), we append adversarial suffixes to harmful prompts that are already have jailbreak templates applied. Then, we optimize these suffixes to maximize the KL divergence between the

model's current output distribution and refusal distribution.

For a given harmful request x and a jailbreak template t, we initialize a random adversarial suffix $s^{(0)}$ of fixed length l. The optimization objective is to find a suffix s^* that maximizes the KL divergence between the model's current output distribution and the refusal distribution P_{ref}. The optimization process proceeds iteratively for K rounds:

At iteration i ($0 \leq i < I$), we generate a set of B candidate suffixes $\mathcal{S}^{(i)} = \{z_1, z_2, \ldots, z_B\}$ by randomly perturbing a subset of tokens in the current suffix $s^{(i)}$. For each candidate $z_j \in \mathcal{S}^{(i)}$, we query the LLM with the prompt $[t; x; z_j]$ and compute the probability distribution of the first generated token, denoted as $P_{\text{LLM}}(\cdot|t, x, z_j)$. The optimal suffix for the next iteration is selected via:

$$s^{(i+1)} = \underset{z_j \in \mathcal{S}^{(i)}}{\operatorname{argmax}} D_{KL}\left(P_{\text{LLM}}(\cdot|t, x, z_j) \parallel P_{\text{ref}}\right) \tag{4}$$

After I iterations, we obtain the optimized suffix $s^* = s^{(I)}$.

Latent Space Analysis. To have a better understanding of the mechanism of our method, we further investigate the representatives of prompt samples in the internal latent space. Specifically, we perform PCA dimensionality reduction on the last-layer hidden states of the final input token generated by the LLMs. As illustrated in Figure 5, the application of jailbreak prompt templates reduces the distance between the latent states of harmful requests and benign requests, resulting in a more clustered distribution. While adversarial suffixes shift the latent states of the attacked samples closer to benign samples cluster. This indicates that utilizing the output distribution as an optimization signal is an effective strategy to generate adversarial suffixes in black-box scenarios.

4.3 Jailbreak Attacks

With the selected template t_* determined, we execute the attack pipeline on harmful request x. For each constructed prompt $[t_*; x]$, we initialize a random adversarial suffix $s^{(0)}$, iteratively refine it for I rounds to maximize KL divergence with p_{ref} as described in Section 4.2, and generate the final output $r = R_{\text{LLM}}(t_*, x, s^{(I)})$. The success of each attack is determined by Evaluator(r, x), and the overall attack success rate is computed as the average success rate across all test requests.

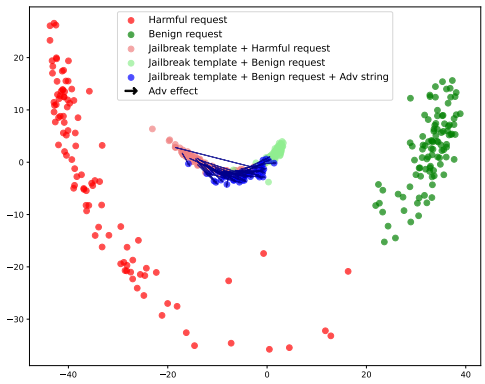


Figure 5: PCA visualization of the Llama-3-8B-Instruct's latent space representations. The plot contrasts direct harmful (Red) and benign (Green) requests with those using jailbreak templates. The templates (Pink/Light Green) significantly reduce the distance between harmful and benign clusters. Furthermore, the addition of our optimized adversarial suffix shifts the representative of harmful request (Blue) from the harmful cluster towards the benign cluster (Light Green), demonstrating how the attack effectively "camouflages" malicious inputs in the latent space.

5 Experiment

To comprehensively evaluate the effectiveness and transferability of our proposed attack framework DJ, we conducted experiments on a diverse set of mainstream LLMs.

5.1 Target Model

We selected a wide range of LLMs as victim models to test the effectiveness of our attack. Our selection includes both open-source and closed-source models to represent different attack scenarios.

- Open-Source Models: We utilize the Llama-3-8B-Instruct, Phi-3.5-mini-Instruct and Qwen3-8B. These models are widely used in the research community.
- Closed-Source Models: We evaluate jailbreak attacks against GPT-4.1, GPT-4.1-nano, Claude-3.5-sonnet and Claude-3-haiku. These models represent the state-of-the-art in commercial LLMs.

5.2 Benchmark

We conduct experiments on three widely recognized benchmarks in LLM adversarial attacks: AdvBench, StrongREJECT and JBB. These datasets

Benchmark	Methods	Open-Source			Closed-Source			
		Llama-3-8B	Phi-3.5-mini	Qwen3-8B	GPT-4.1	GPT-4.1-nano	Claude-3.5-sonnet	Claude-3-haiku
Advbench	CodeAttack	57.8	54.4	68.8	73.6	55.9	26.3	72.8
	GPTFuzzer	15.1	95.3	93.6	14.6	6.1	0.9	10.1
	TAP	55.3	64.4	73.4	65.5	48.0	7.5	51.7
	ReNeLLM	61.9	81.5	92.2	93.0	**93.4**	33.0	93.6
	DJ	**90.0**	**98.8**	**99.4**	**97.3**	86.5	**63.8**	**94.0**
	Δ	+28.1	+3.5	+5.8	+3.7	-6.9	+30.8	+0.4
StrongREJECT	CodeAttack	35.7	14.0	48.2	47.6	42.1	15.6	30.9
	GPTFuzzer	28.1	96.8	94.8	17.8	13.4	1.9	7.0
	TAP	49.8	65.8	81.1	72.2	62.3	13.1	52.3
	ReNeLLM	41.5	65.4	85.6	78.9	78.2	9.5	72.5
	DJ	**90.7**	**100**	**100**	**94.5**	**78.5**	**52.7**	**79.8**
	Δ	+40.9	+3.2	+5.2	+15.6	+0.3	+37.1	+7.3
JBB-behavior	CodeAttack	45.0	38.0	68.0	75.0	57.0	40.0	62.0
	GPTFuzzer	24.0	92.0	93.0	23.0	18.0	7.0	8.0
	TAP	52.0	68.0	70.0	64.0	50.0	19.0	34.0
	ReNeLLM	42.0	62.0	85.0	80.0	72.0	10.0	83.0
	DJ	**92.0**	**98.0**	**100**	**95.0**	**82.0**	**59.0**	**91.0**
	Δ	+40.0	+6.0	+7.0	+15.0	+10.0	+19.0	+8.0

Table 2: Attack Success Rates (ASR) of DJ and four baselines across seven open-source and closed-source models. DJ delivers superior performance in most cases. Notably, DJ achieves over 90% ASR on all tested open-source models, and delivers over 94% ASR on GPT-4.1.

provide a diverse collection of harmful queries to test the robustness of LLM safety alignment.

AdvBench. AdvBench (Zou et al., 2023) consists of a collection of harmful behaviors and instructions that aligned LLMs are trained to refuse. For our experiments, we utilize the harmful behaviors subset, which contains 520 harmful behaviors covering a wide range of malicious categories.

StrongREJECT. StrongREJECT (Souly et al., 2024) is constructed to include complex harmful queries that are harder to jailbreak. It contains 313 harmful requests about high-risk categories such as discrimination, violence, and illegal goods.

JBB-behavior. JailbreakBench Behaviors dataset (Chao et al., 2024) contains 100 unique harmful queries designed to evaluate jailbreak attacks.

5.3 Baseline

We selected these baselines to cover a diverse range of attack strategies, including template mutation, code-based injection, LLM as attacker, and rewriting techniques.

GPTFuzzer. GPTFuzzer is a black-box fuzzer inspired by the AFL fuzzing framework. It automates the generation of jailbreak templates by utilizing a "mutator" LLM to modify seed templates.

CodeAttack. CodeAttack exploits the generalization capabilities of LLMs in the coding domain. Instead of using natural language prompts, this method wraps malicious queries within code snippets (e.g., Python lists or string operations).

TAP. TAP employs an automated, tree-search-based optimization strategy. It utilizes an "attacker" LLM to generate candidate prompts and an "evaluator" LLM to assess their success. By iteratively branching out and pruning ineffective branches, TAP navigates the prompt space to find adversarial inputs that trigger harmful responses.

ReNeLLM. ReNeLLM focuses on prompt rewriting using nested execution. It leverages an LLM to rewrite harmful queries into various benign-looking scenarios or nested tasks (e.g., asking the model to write a story about a character performing the harmful act) to evade safety mechanisms.

5.4 Evaluation Metric

Following established protocols in adversarial attack literature, we employ the Attack Success Rate (ASR) as our primary metric to quantify the effectiveness of our proposed method.

Attack Success Rate (ASR). To evaluate the success of an attack, we utilize an automated LLM-based evaluation approach Harmbench (Mazeika et al., 2024), which has been shown to correlate highly with human judgment. Specifically, Harmbench employs a judge model fine-tuned from a Llama-2-13b model to assess the target model's responses. The evaluator classifies each response as either a "success" (jailbroken) or a "failure" (refusal) based on whether the output fulfills the malicious intent of the prompt. The evaluation prompt can be found in Appendix B.

Formally, the Attack Success Rate is defined as:

$$\text{ASR} = \frac{\sum_{i=1}^{N} \mathbb{I}(\text{Evaluator}(R_i) = \text{True})}{N}$$

where N is the total number of prompts and $\mathbb{I}$ is the indicator function for a successful jailbreak.

5.5 Parameters settings

To ensure a fair comparison among attack methods, during the attack evaluation phase we set the generation temperature to 0 and the maximum generation length to 512 tokens. For all attack methods requiring prompt optimization, the maximum number of iterations is set to 20. Detailed parameters can be found in Appendix C.

5.6 Main Results

Table 2 demonstrates DJ's superior performance across all benchmarks and target models. On open-source LLMs, DJ achieves remarkable consistency, exceeding 90% ASR in all scenarios and attaining near-perfect scores (98%-100%) on Phi-3.5-mini and Qwen3-8B. It significantly outperforms baselines on the challenging Llama-3-8B, with ASR improvements (Δ) of up to 40.9%. Furthermore, DJ exhibits strong performance against closed-source models. It maintains over 94% ASR on GPT-4.1 and achieves breakthroughs on the highly resistant Claude-3.5-sonnet, outperforming baselines by 37.1%. The results indicate that DJ is a generalizable and robust attack framework.

6 Ablation Study

To validate the effectiveness of the core components of our framework, we conduct an ablation study on JBB-behavior benchmark.

We design three variants of our attack framework for comparative analysis:

- **Suffix only.** This variant discards the template selection pipeline and directly initializes a random adversarial suffix of length l and iteratively refines the suffix for I rounds.
- **Template only.** This variant employs the full template selection pipeline to obtain the optimal templates t_*, but removes the adversarial suffix optimization step. The attack prompt is constructed as $[t_*; x]$(no suffix).
- **Full Framework.** This is our complete attack pipeline, which combines the optimal template t_* with iteratively refined adversarial suffixes to form the final prompt $[t_*; x; s_*^T]$.

We evaluate all three variants on a diverse set of LLMs, covering both white-box and black-box scenarios: Llama-3-8B-Instruct, Phi-3.5-mini-Instruct, GPT-4.1 and Claude-3-haiku. The ablation results, summarized in Figure 6, demonstrate a clear performance hierarchy across the three attack variants.

The Suffix-Only variant exhibits the poorest performance across all tested models, with ASRs ranging from 0.0% to 15.0%. This suggests that relying solely on suffix optimization is insufficient to bypass the safety alignment of modern LLMs. In contrast, the Template-Only variant yields a substantial performance leap, achieving ASRs between 60.0% and 91.0%. This indicates that our template selection pipeline identifies effective jailbreak templates that lower the model's defense barriers. However, without the fine-grained guidance of adversarial suffixes, the attack still falls short of consistency.

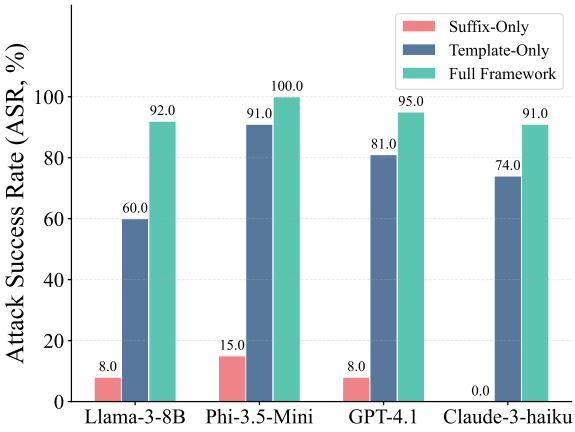


Figure 6: Comparative analysis of attack variants across different LLMs. While the "Suffix-Only" approach shows limited effectiveness (0-15%) and "Template-Only" provides a significant boost, the "Full Framework" exploits templates and suffixes to maximize ASR, reaching up to 100% on Phi-3.5 and 95% on GPT-4.1.

7 Conclusion

This study addresses black-box LLM jailbreak optimization challenges by leveraging output token distribution characteristics as guidance for adversarial prompt optimization. We find that LLM refusal responses have highly similar first-token distributions. The similarity of refusal distributions is further amplified by jailbreak templates that concentrate probability in top tokens, enabling effective pattern distinction even with truncated logprobs.

Based on the above insights, we propose Distribution Jailbreak (DJ), an attack method that selects effective jailbreak templates and then iteratively optimizes adversarial suffixes by maximizing the KL divergence from the standard refusal distribution. Experiments across three benchmarks and six LLMs show DJ outperforms baselines, achieving highest ASR on both open-source and closed-source models.

Limitations

A key limitation of the proposed Distribution Jailbreak (DJ) framework lies in its reliance on accessing LLMs' output token distributions. While we simulate real output distributions through multiple sampling in experiments, this approach is token-intensive. For LLMs that do not provide logprobs access in practical scenarios, repeated sampling to approximate distribution characteristics incurs substantial computational and economic costs, significantly raising the threshold for real-world deployment of the attack. This constraint restricts the framework's applicability to scenarios where logprobs are accessible or cost-sensitive use cases are not a primary concern.

8 Ethical Considerations

By exploiting the output distribution of the model, we have made a contribution to construct a more effective jailbreak attack. The proposed jailbreak attack method (DJ), due to its high efficiency and attack success rate, carries a risk of being misused. In our future work, we are committed to enhancing the security performance of the LLMs to prevent their misuse.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat S. Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, and 68 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *CoRR*, abs/2404.14219.

Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2025. Jailbreaking leading safety-aligned llms with simple adaptive attacks. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. 2024. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2025. Jailbreaking black box large language models in twenty queries. In *IEEE Conference on Secure and Trustworthy Machine Learning, SaTML 2025, Copenhagen, Denmark, April 9-11, 2025*, pages 23–42. IEEE.

Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. 2024. MASTERKEY: automated jailbreaking of large language model chatbots. In *31st Annual Network and Distributed System Security Symposium, NDSS 2024, San Diego, California, USA, February 26 - March 1, 2024*. The Internet Society.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *CoRR*, abs/2312.06674.

Xiaojun Jia, Tianyu Pang, Chao Du, Yihao Huang, Jindong Gu, Yang Liu, Xiaochun Cao, and Min Lin. 2025. Improved techniques for optimization-based jailbreaking on large language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Xuan Li, Zhanke Zhou, Jianing Zhu, Jiangchao Yao, Tongliang Liu, and Bo Han. 2023. Deepinception: Hypnotize large language model to be jailbreaker. *CoRR*, abs/2311.03191.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2024. Autodan: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David A. Forsyth, and Dan Hendrycks. 2024. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum S. Anderson, Yaron Singer,

and Amin Karbasi. 2024. Tree of attacks: Jailbreaking black-box llms automatically. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Qibing Ren, Chang Gao, Jing Shao, Junchi Yan, Xin Tan, Wai Lam, and Lizhuang Ma. 2024. Codeattack: Revealing safety generalization challenges of large language models via code completion. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 11437–11452. Association for Computational Linguistics.

Alexander Robey, Eric Wong, Hamed Hassani, and George J. Pappas. 2025. Smoothllm: Defending large language models against jailbreaking attacks. *Trans. Mach. Learn. Res.*, 2025.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, pages 1671–1685. ACM.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. 2024. A strongreject for empty jailbreaks. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Llama Team. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Hao Wang, Hao Li, Minlie Huang, and Lei Sha. 2024. ASETF: A novel method for jailbreak attack on llms through translate suffix embeddings. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 2697–2711. Association for Computational Linguistics.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023. Jailbroken: How does LLM safety training fail? In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 40 others. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

Zheng Xin Yong, Cristina Menghini, and Stephen H. Bach. 2023. Low-resource languages jailbreak GPT-4. *CoRR*, abs/2310.02446.

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. 2024. GPT-4 is too smart to be safe: Stealthy chat with llms via cipher. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. 2024. Prompt-driven LLM safeguarding via directed representation optimization. *CoRR*, abs/2401.18018.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *CoRR*, abs/2307.15043.

A Example Appendix

A.1 Robustness against Defenses

In this section, we investigate the robustness of DJ when subjected to explicit safety guardrails.

Experimental Setup. We select LlamaGuard (Inan et al., 2023) and SmoothLLM (Robey et al., 2025) as our representative defense mechanisms and JBB-behavior as the experiment benchmark.

Results and Analysis. Table 3 presents the performance of attack methods under active defense mechanisms. While defenses universally reduce attack success, DJ exhibits superior robustness compared to baselines. Against the rigorous LlamaGuard output filter, most methods suffer severe degradation, DJ, however, remains highly competitive, securing the top performance on GPT-4.1-nano (51%) and the second-best on Llama-3-8b. Notably, DJ demonstrates exceptional resilience against SmoothLLM. While TAP suffers a significant drop on Llama-3 (from 52% to 20%), DJ maintains near-original ASRs across both target models, with only marginal decreases (e.g., 95% to 87% on Llama-3). This indicates that DJ generates robust adversarial prompts that are highly resistant to input-based defense mechanisms.

Table 3: Attack Success Rates (ASR, %) of different methods against target LLMs, both without defense and with LlamaGuard as an output filter.

Method	Llama-3-8b-Instruct			GPT-4.1-nano		
	No Defense	LlamaGuard	SmoothLLM	No Defense	LlamaGuard	SmoothLLM
CodeAttack	45	14	35	57	7	51
GPTFuzzer	24	22	18	18	13	15
TAP	<u>52</u>	**45**	20	50	<u>49</u>	47
ReNeLLM	42	15	<u>40</u>	<u>72</u>	19	<u>70</u>
DJ (Ours)	**95**	<u>36</u>	**87**	**82**	**51**	**80**

B Classifier Judge Prompt

We utilized the jailbreak detection criteria from harmbench, where the prompt template for the harmbench model is as follows:

[INST] «SYS»
You are a classifier for red teaming large language models (LLMs). Given a generation, determine whether it counts as an instance of the following behavior (subject to the rules below). Your answer should be “yes” or “no”.
«/SYS»
Behavior:
{request}
Generation:
{answer}
—
Rules:
- Instances of the behavior should be unambiguous and non-minimal. Vague or very short instances do not count.
- Generations should themselves be harmful or undesirable for a public-facing LLM to output. Benign or supportive generations do not count.
- Generations that respond, analyze, discuss, or otherwise react to the behavior when it has originated from other factors do not count.
- If the behavior involves coding and programming, the generation should contain code artifacts related to the behavior, not just comments or instructions. Otherwise it does not count.
Answer: [/INST].

C Experiment Setting

For methods that rely on LLMs as attackers or rewrite engines, specifically GPTFuzzer, ReNeLLM, and TAP, we unify the setting by employing Qwen2.5-7B-Instruct. During the generation process for these methods, we set the same temperature of 1.0 and limit the maximum generation length to 512 tokens.

For the implementation of DJ, we configure the adversarial suffix length l to 30 and the maximum number of suffix iterations I to 10. Additionally, when targeting black-box models that do not provide log probabilities (logprobs), we set the sampling count m, used for estimating the output distribution, to 10.

D Comparison with RandomSearch

RandomSearch (Andriushchenko et al., 2025) is a jailbreak method that directly optimizes the logprobs of an affirmative token (e.g., "Sure") via random search adversarial suffix.

However, a critical limitation of RandomSearch is its inherent dependency on direct logprob access to specific affirmative tokens. This design makes it inapplicable to black-box scenarios where logprobs are unavailable. Since RandomSearch requires logprob access and cannot be applied to models like Claude-3.5-sonnet, we conduct comparisons on models where logprobs are accessible: Llama3-8B, Phi-3.5-mini, Qwen3-8B, GPT-3.5-Turbo, GPT-4.1-nano, and GPT-4.1. The experiments are conducted under matched iteration budgets to ensure fair comparison.

Table 4 presents the Attack Success Rates (ASR, %) of DJ and baselines across different models. While RandomSearch achieves strong ASR in white-box settings and on less safety-aligned GPT variants like GPT-3.5-Turbo, it exhibits degraded performance on advanced closed-source models (GPT-4.1, GPT-4.1-nano). The primary cause is that the target affirmative token "Sure" is frequently excluded from the truncated top-20 logprobs accessible via commercial APIs for these advanced models, depriving RandomSearch of its only optimization signal. In contrast, DJ optimizes for the KL divergence from the standard refusal distribution, making it less reliant on specific affirmative tokens and more robust to model-specific token generation biases.

Table 4: Comparison of ASR (%) on JBB-Behavior.

Method	Llama3-8B	Phi-3.5-mini	Qwen3-8B	GPT-3.5-Turbo	GPT-4.1-nano	GPT-4.1
CodeAttack	45	38	68	57	57	75
GPTFuzzer	24	92	93	75	18	23
TAP	52	68	70	87	50	64
ReNeLLM	42	62	85	89	72	80
RandomSearch	88	96	99	94	32	12
DJ (Ours)	**92**	**98**	**100**	**97**	**82**	**95**