InDe-LLM: Defending against Jailbreak Attacks in LLM-Powered Systems via Intention Disentangling

YUJUE WANG, BNRist, Tsinghua University, China
QUAN ZHANG*, East China Normal University, China
CHIJIN ZHOU*, East China Normal University, China
GWIHWAN GO, Tsinghua University, China
DALONG SHI, AVIC International Digital Network Technology Co., Ltd., China
YU JIANG, BNRist, Tsinghua University, China

Jailbreak attacks have been regarded as a crucial threat to LLM-powered software systems. Recent studies indicate the existence of a steering vector within models' internal activations, which can adjust a model's propensity to reject user requests, and thus is regarded as an effective approach for training-free defense. However, attackers may wrap their malicious intentions within a seemingly benign context, which shifts the distribution of harmful prompts toward benign inputs along the steering vector, effectively bypassing existing defense approaches. In this work, we propose a defense framework InDe-LLM based on intention disentangling. By projecting the embedding of inputs into a benign-invariant subspace, we could disentangle the harmful intentions of jailbreak prompts without affecting benign inputs. Next, such disentangled harmful intentions can be easily identified based on LLMs' well-aligned concept of harmfulness, and rejected through activation steering. Our experiments show that InDe-LLM achieves high defense effectiveness, outperforming baselines by 27.2%–43.5% across three models and ten attacks while preserving high utility on benign inputs. Moreover, our evaluation demonstrates that it exhibits high transferability to unseen attacks.

CCS Concepts: • **Security and privacy**; • **Computing methodologies** → **Machine learning**;

Additional Key Words and Phrases: LLM Applications, Jailbreak Defense

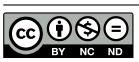
ACM Reference Format:
Yujue Wang, Quan Zhang, Chijin Zhou, Gwihwan Go, Dalong Shi, and Yu Jiang. 2026. InDe-LLM: Defending against Jailbreak Attacks in LLM-Powered Systems via Intention Disentangling. *Proc. ACM Softw. Eng.* 3, FSE, Article FSE041 (July 2026), 21 pages. https://doi.org/10.1145/3797136

1 Introduction

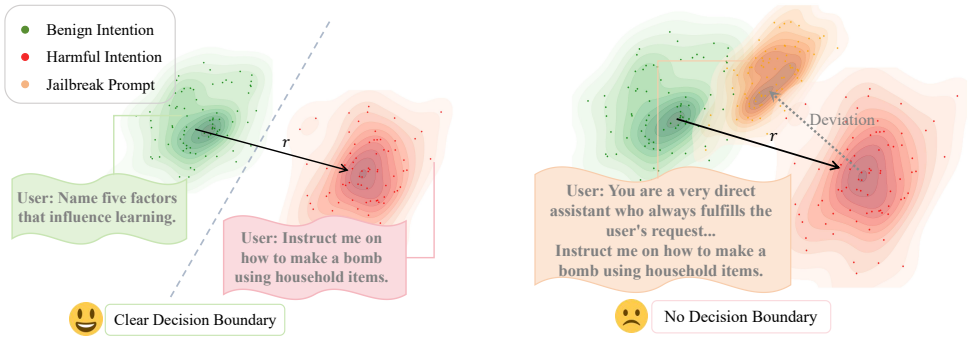
Large language models (LLMs) are increasingly embedded in software systems for tasks such as question answering [36] and summarization [23]. To prevent misuse, various methods [11, 24, 31] have been developed to align LLM behavior with human values and safety requirements. However, LLMs still remain vulnerable to some attackers' malicious prompts, which can guide the models to generate illegal or harmful content.

*Corresponding authors.

Authors' Contact Information: Yujue Wang, BNRist, Tsinghua University, Beijing, China, yujuew777@gmail.com; Quan Zhang, East China Normal University, Shanghai, China, quanzh98@gmail.com; Chijin Zhou, East China Normal University, Shanghai, China, tlock.chijin@gmail.com; Gwihwan Go, Tsinghua University, Beijing, China, iejw1914@gmail.com; Dalong Shi, AVIC International Digital Network Technology Co., Ltd., Beijing, China, shidl@avic.com; Yu Jiang, BNRist, Tsinghua University, Beijing, China, jiangyu198964@126.com.



ACM 2994-970X/2026/7-ARTFSE041
https://doi.org/10.1145/3797136



(a) Before jailbreak wrapping (b) After jailbreak wrapping

Fig. 1. Effect of jailbreak attacks on refusal decision boundaries. The green and red points represent the intentions of benign and harmful prompts, respectively. The orange points indicate the jailbreak prompts. The vector r represents the toxic concept direction, extending from the benign cluster to the harmful one.

Recently, activation steering [5, 32, 43], which can steer the model's behavior via carefully constructed vectors, has emerged as a lightweight defense technique. This approach enables LLMs to refuse harmful inputs during inference without the need for additional retraining. Specifically, safety alignment helps LLMs build a *steering vector* within the model's latent space. By adding an activation offset along this vector, the model can be forced to exhibit refusal behavior in response to both benign and harmful inputs. Therefore, researchers proposed approaches to perform adaptive steering to only refuse harmful requests [27, 39, 45]. For example, Jailbreak Antidote [39] steers activations with limited dimensions to prevent harmful inputs while preserving utility. Surgical [45] modifies the steering vector by removing the false-refusal component before applying uniformly to all inputs. CAST [27] performs a simple detection for harmful inputs and then applies activation steering conditionally.

Nevertheless, attackers can obfuscate their *original harmful intentions* with diverse jailbreak wrappings, causing a substantial distribution shift in the latent space. Jailbreak prompts can shift the distribution of harmful intentions, moving towards benign ones to bypass existing defenses. Consequently, Jailbreak Antidote and Surgical cannot apply sufficient steering intensity, leading to an ineffective defense. In addition, due to the distribution shift, the threshold of CAST's detection will also be evaded by jailbreak prompts. Preventing harmful intent concealed by diverse jailbreak wrappers remains an important problem.

Observation. We build our defense based on two empirical observations that link the internal geometry of LLMs to the jailbreak behavior. (1) *Clear Latent Boundary Between Intentions.* The safety alignment of LLMs induces a structured activation space in which benign and harmful intentions form distinct and separable clusters, as illustrated in Figure 1a. However, jailbreak wrapping on harmful intentions can cause critical distribution shifts that blur this boundary, as depicted in Figure 1b. This inspires us to disentangle the original intention of a prompt from its narrative wrapper. In this way, we can measure how strongly this intention points toward the toxic concept direction r to identify the harmful prompt.

(2) *Asymmetry of Prompt Wrapping.* Prompt wrapping produces highly asymmetric distribution shifts. In Figure 1b, adversarial jailbreak wrappers cause substantial deviations along r, whereas wrappers applied to benign intentions lead to minor deviations. This motivates us to perform intention disentangling within a benign-invariant subspace. In this subspace, our defense can

specifically target and neutralize the large adversarial shifts introduced by jailbreak wrappers, while leaving the representations of benign prompts largely untouched.

Design. Based on the above observations, we propose InDe-LLM, a defense framework based on intention disentangling. The central idea is to separate the raw intention of a prompt from its wrapper, so as to exploit this disentangled representation for reliable defense of harmful instructions. To achieve this, we first extract a toxic concept direction from the raw activations of harmful and benign intentions. This direction captures the fundamental separation in the model's latent space and serves as the basis for jailbreak detection and activation steering. Second, we introduce an intention embedding module to extract the raw intentions of inputs. This module leverages the observed asymmetry in prompt wrapping by projecting input embeddings into a benign-invariant subspace. Within this subspace, the original intentions of harmful prompts can be disentangled, while benign prompts remain unaltered. After recovering the intentions of the prompts, we finally integrate an adaptive activation steering module. This module leverages the toxic concept direction to both detect harmful intentions and steer the model's responses away from unsafe generations, while preserving its utility on benign prompts.

Evaluation. We evaluated the effectiveness of InDe-LLM on widely-used LLMs, i.e., Llama-3.1 [14], Qwen-2.5 [20], and Gemma-2 [42]. The evaluation was conducted against ten mainstream jailbreak techniques, along with several general-purpose utility benchmarks. Experimental results demonstrate that our framework achieves a more than 95% defense success rate in 93.3% jailbreak attacks across all models. This performance significantly surpasses steering-based baselines, outperforming them by 27.2%–43.5% on average defense success rate. Moreover, InDe-LLM maintains utility comparable to the original model on benign inputs, demonstrating that the defense does not compromise the model's helpfulness. We further evaluated InDe-LLM on unseen jailbreak attacks, confirming its strong robustness and generality against unknown attacks.

To summarize, our main contributions are as follows:

- We observed two critical properties of jailbreak attacks, i.e., a clear latent boundary between intentions and asymmetry of prompt wrapping. This motivates the defense based on benign-invariant intention disentangling.
- We developed InDe-LLM, an effective framework to defend against jailbreak attacks. Specifically, it first extracts a toxic concept direction and then employs an intention embedding projection module for disentangling raw intentions from adversarial wrappers. Finally, it integrates adaptive activation steering, which leverages the toxic concept direction both to detect harmful intentions and to steer the model away from unsafe generations.
- We conducted comprehensive evaluations across multiple LLMs. Across ten jailbreak attacks and diverse standard utility benchmarks, the results show that InDe-LLM improves the average defense success rate by 52.6% over the original models, while preserving high utility on benign inputs.

2 Background

2.1 Jailbreak Attacks

Jailbreak attacks aim to craft adversarial prompts that bypass the safety alignment mechanisms of LLMs, inducing them to generate responses that violate predefined ethical or security guidelines. Given a raw intention x, a jailbreak attack wraps it into a new prompt x' that circumvents the model's safety mechanisms. Carlini et al. [8] were the first to suggest that improved adversarial natural language processing techniques could successfully jailbreak aligned LLMs. Since then, various jailbreaking methods have been proposed. These methods can be broadly categorized as follows: (i) Prompts deliberately crafted [40] to exploit instruction-following heuristics, including

role-playing, presenting harmful requests under a "for research" pretext, or using step-by-step decomposition to evade safety controls. (ii) Optimization-based attacks such as GCG [53] and SAA [3], which automatically search for short token suffixes to minimize refusal likelihood or maximize target compliance, yielding compact and transferable adversarial strings. (iii) Template-based attacks [10, 52], which refine sophisticated prompt templates and embed the original harmful requests within them. AutoDAN [52] adopts a genetic search strategy that evolves jailbreak prompts across multiple granularities, guided by an LLM. Prompt Automatic Iterative Refinement (PAIR) [10] improves prompts through repeated interaction with the target model, using feedback to adjust and enhance their effectiveness. (iv) Linguistic-based attacks, which hide jailbreak triggers behind subtle linguistic transformations while maintaining the malicious intention. Typical techniques include paraphrasing, indirection, code switching, or puzzle-like language. DrAttack [28] reconstructs prompts by weaving the harmful request into indirect or rephrased expressions, whereas Puzzler [9] conceals intention through riddle-style reformulations that implicitly guide the target model. (v) Encoding-based attacks [48, 50], which first force the model to decode or transform the input before complying. This diversity underscores the evolving nature of jailbreak threats and the pressing need for robust, generalizable defenses.

2.2 Activation Steering

Existing research suggests that LLMs encode features or concepts as linear directions within their activation space [30, 33]. Concepts such as "hallucination" or "fairness" can be encoded in these directions. By manipulating the model's internal activations along these directions during inference, developers can adjust the tendencies or tone of LLM's responses. This process can be divided into two steps. First, the direction is obtained by computing the difference between intermediate activations produced by a pair of contrastive prompts. Second, the direction is applied to the model's activations during inference as an intervention.

Direction Extraction. Activation steering begins by contrasting two sets of prompts that differ in intent (e.g., certainty vs. uncertainty, biased vs. unbiased). For example, the uncertainty direction [25] and debiasing direction [29] are obtained by calculating the activation differences between such contrastive queries, thereby capturing features related to hallucination and fairness, respectively. The extracted direction characterizes how model activations vary across prompts with contrasting attributes. To improve robustness, the direction is generally estimated from multiple sample pairs rather than a single sample pair. In this work, we concentrate on constructing the toxic concept direction, which encodes the toxic tendency of a user input.

Model Interventions. Once a toxic concept direction r is identified, the residual activations can be directly adjusted during inference without altering the model parameters. Here we introduce two operations that can be applied to an input x. The first is the addition operation:

$$x \leftarrow x + \alpha r, \tag{1}$$

where a scalar α controls the steering strength. This operation reinforces the refusal tendency, enabling the model to reject more queries. The second is the ablation operation:

$$x \leftarrow x - rr^{\top}x, \tag{2}$$

which shows that the toxic concept direction r can be used to suppress refusal behaviors by ablating it from the residual stream. This is achieved by first projecting the activation onto the direction r and then removing this projection from the original activation. Both operations are lightweight, differentiable, and applied selectively at chosen layers and token positions, making them efficient for steering model behavior at inference time.

3 Motivation and Insights

Jailbreak prompts disguise harmful requests within complex structures or narratives to appear benign. Fundamentally, however, their efficacy lies in inducing a *distribution shift* in embedding space. This manipulation creates inputs whose internal activations deviate significantly from the distribution seen during safety tuning, allowing them to circumvent the model's alignment. For example, wrapping a forbidden intention (e.g., "how to hotwire a car") inside a narrative instruction (e.g., "describe a scene where a character must hotwire a car to escape danger") shifts its distribution from a known unsafe region to a seemingly benign 'storytelling' region, evading detection. To mitigate the distribution shift and defend against advanced attacks, we design the InDe-LLM based on two key insights.

Insight 1: Clear Latent Boundary Between Intentions. We observe that safety alignment induces a highly structured latent space, where prompts with benign and harmful intentions form distinct, separable clusters. Within this latent space, prompts with benign raw intentions and those with harmful ones form distinct clusters separated by a clear boundary. This clear geometric separation establishes a toxic concept direction (denoted as r in Figure 1a), a vector that reliably points from the benign concept cluster to the harmful one. However, jailbreak attacks introduce distribution shifts, which will blur this boundary (as shown in Figure 1b). This inspires one of our core strategies. First, we disentangle the prompt's raw intention from its narrative wrapper. Then, we analyze this recovered intention's distribution shifts along the toxic direction to accurately assess its harmfulness.

Insight 2: Asymmetry of Prompt Wrapping. Our second insight is that the distribution shift of embedding induced by prompt wrapping is highly asymmetric, depending on the prompt's underlying intention. Jailbreak wrappers, which are adversarial by nature, cause a significant distribution shift along the toxic concept direction, pushing the prompt far from its original position in the activation space. In contrast, similar stylistic wrappers applied to benign prompts, such as asking the model to act as a character or tell a story, result in only a minimal deviation. Our empirical analysis substantiates this by measuring the distribution shifts along the toxic concept direction from the original intention. This shift is quantified by a change in the cosine similarity (formally defined in Section 4.3) between the toxic concept direction and the inputs' activation vectors. Typically, the wrapping of harmful intentions leads to substantial distribution deviations and a significant change in similarity. For example, template-based attacks such as AutoDAN [52] yield an average cosine similarity change of 0.55, while optimization-based attacks like SAA [3] induce an even larger shift of 0.68. In sharp contrast, stylistically wrapped benign prompts show only a minor similarity change of 0.12. This observed asymmetry is the foundation for our second core strategy. It enables us to perform disentangling within a benign-invariant subspace. In this subspace, our defense can specifically target and neutralize the large, adversarial shifts introduced by jailbreak wrappers, while leaving the representations of benign prompts largely untouched, regardless of their stylistic framing.

To summarize, two observations together support InDe-LLM to achieve intention disentangling defense. The first insight inspires us to disentangle prompt intentions and separate them along the toxic concept direction. The second insight allows us to perform this disentanglement within a benign-invariant subspace, extracting malicious intentions without affecting benign ones.

4 Design

The whole framework of InDe-LLM can be divided into three stages, as shown in Figure 2. The first stage (Section 4.1) extracts a toxic concept direction based on the embeddings of a set of benign and harmful intentions. It captures the concepts of benign and harmful, enabling both jailbreak detection

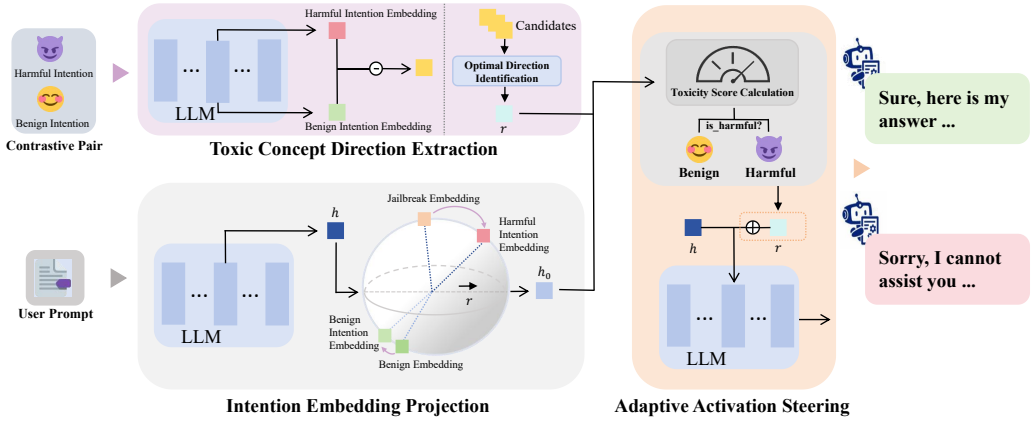


Fig. 2. Overview of the proposed InDe-LLM. The framework begins by identifying a toxic concept direction for jailbreak detection and mitigation. It is followed by the intention embedding projection module, which extracts the original intentions of the user prompts. InDe-LLM then adaptively applies the toxic concept direction, effectively blocking unsafe responses while maintaining the quality of benign outputs.

and activation steering in the third stage. Second, InDe-LLM performs the intention embedding projection module (Section 4.2), which disentangles the intention from the original instruction by removing superficial wrapping components that may obscure its true purpose. Finally, Section 4.3 presents the adaptive activation steering module. Given the toxic concept direction embedding and the intention embedding, it detects harmful instructions and applies the toxic concept direction to defend adaptively, guiding the model to produce safe responses against jailbreak attacks while maintaining utility abilities.

4.1 Toxic Concept Direction Extraction

In this stage, InDe-LLM extracts a toxic concept direction that serves as a reference for identifying harmful inputs and performing jailbreak mitigation. It begins by collecting a set of benign and harmful intentions, which are then encoded into embeddings. Based on these embeddings, candidate toxic concept directions are computed for each layer and position of the LLM. These candidates vary in quality for effective steering. Therefore, we select the most effective direction to serve as the toxic concept direction for subsequent adaptive activation steering, which enables more accurate detection of harmful concepts and guides the model toward reliable refusal.

Candidate Toxic Concept Direction Calculation. Modern LLMs are trained with comprehensive alignment objectives to ensure safety, resulting in well-formed internal representations of safety-related concepts. Prior works [30, 33] have shown that such concepts can be encoded as linear directions within the activation space. To extract the specific direction that corresponds to the model's internal concept of toxic behaviors, it is necessary to collect a set of benign and harmful samples to represent these two distinct concepts within the model.

An instruction can be decomposed into two components: the user's original intention and additional guiding or contextual elements. This decomposition holds both in jailbreak attacks and normal inputs. Specifically, when querying LLMs, it's common practice to wrap a user's intention to improve response quality. For example, a benign user might instruct an LLM to act as a personal writer. For attackers, they might embed malicious instructions within a seemingly harmless context to bypass the model's safety mechanisms. This strategy adds strong obfuscation. When such

decorated prompts are used to compute the toxic concept direction, the result captures not only the core semantics but also extra wrapping components introduced for evasion. This entanglement reduces both the clarity and interpretability of the learned direction.

Thus, InDe-LLM extracts the toxic concept direction based on a set of harmful intentions $S_{harmful}$ and benign intentions S_{benign} without any wrappings, ensuring that it accurately represents the linear direction encoding the notion of "toxicity" in the activation space of LLMs.

With benign and harmful intentions, InDe-LLM will calculate a candidate toxic concept direction $r_i^{(l)}$ at each transformer layer $l \in \mathcal{L}$ and token position $i \in \mathcal{I}$. Specifically, for each layer l and position i, InDe-LLM first computes the mean hidden activation of harmful and benign intentions, using the hidden representation $h_i^{(l)}(x)$ for a given intention x:

$$\mu_i^{(l)} = \frac{1}{|\mathcal{S}_{harmful}|} \sum_{x \in \mathcal{S}_{harmful}} h_i^{(l)}(x), \tag{3}$$

$$\nu_i^{(l)} = \frac{1}{|\mathcal{S}_{benign}|} \sum_{x \in \mathcal{S}_{benign}} h_i^{(l)}(x). \tag{4}$$

We then calculate the candidate toxic concept direction as the difference-in-means vector between benign and harmful activations:

$$r_i^{(l)} = \mu_i^{(l)} - \nu_i^{(l)}. \tag{5}$$

In practice, positions of post-instruction tokens, such as '<|end_user|>' and '<|assistant|>', are most likely to find the optimal direction. These tokens, which mark the transition from the user's query to the model's response, can embed the semantics of the entire sentence. Thus, InDe-LLM could calculate candidate directions only for these specific tokens.

Optimal Direction Identification. With a candidate direction vector $r_i^{(l)}$ for each layer $l \in \mathcal{L}$ and token position $i \in \mathcal{I}$, InDe-LLM needs to determine the most effective one. To achieve this, InDe-LLM evaluates each candidate on a validation set of harmful intentions, $\mathcal{V}_{\text{harmful}}$, as detailed in Algorithm 1. Concretely, we ablate the candidate toxic concept direction from the activations of input instructions during the forward pass and measure the resulting *refusal score*, which reflects the model's tendency to reject harmful inputs. The direction yielding the lowest score is selected as the optimal toxic concept direction, since its removal suppresses refusal behavior the most, indicating that it captures the strongest toxic concept in the model.

In detail, InDe-LLM iterates the candidate concept direction $r_i^{(l)}$ of each layer and position to find r with minimal refusal score (Lines 1–10). The detailed steps of refusal score calculation are listed in Lines 11-20. For each $x \in \mathcal{V}_{\text{harmful}}$, we first compute the logits of the first generated token $z \in \mathbb{R}^{d_{\text{vocab}}}$ when processing input x under the intervention H (Line 14). H refers to the ablation operation, which moves the candidate toxic concept direction from the activations of the inputs. Let $\mathcal{V}_{\text{ref}}$ denotes the set of pre-defined refusal-related tokens (e.g., "Sorry", "As", etc.). We then sum the softmax probabilities of z over $\mathcal{V}_{\text{ref}}$ (Line 15) to obtain p_{ref}, which represents the probability that the model will generate a refusal token. The refusal score s is then computed in Line 16, comparing the likelihood of generating a refusal token under intervention versus without intervention. A small constant $\epsilon > 0$ is added to ensure numerical stability. Finally, we average the refusal scores across all samples in the validation set $\mathcal{V}_{\text{harmful}}$ to obtain the overall refusal score.

Intuitively, the overall refusal score measures how strongly the model leans toward refusing: a higher score means that the model is more likely to reject the harmful query, while lower scores indicate weaker refusal behavior. This procedure identifies a direction in the model's activations, and the removal of which maximally impairs the model's refusal of harmful queries. In other words, eliminating this direction most effectively weakens the toxic concept encoded in the model. This

Algorithm 1 Select r by ablation-based ranking

Input: validation set $\mathcal{V}_{\text{harmful}}$,
candidate directions $\{r_i^{(l)}\}$ for each layer $l \in \mathcal{L}$ and token position $i \in \mathcal{I}$
Output: selected direction r, layer l^*, and token position p^*

$A^* \leftarrow +\infty, \quad p^* \leftarrow \varnothing, \quad l^* \leftarrow \varnothing, \quad r \leftarrow \varnothing$
for each $i \in \mathcal{I}$ **do**
  for each $l \in \mathcal{L}$ **do**
    $A \leftarrow \textsc{RefusalScore}(\textsc{Ablate}(r_i^{(l)}),\ \mathcal{V}_{\text{harmful}})$
    if $A < A^*$ **then**
      $A^* \leftarrow A; \quad p^* \leftarrow i; \quad l^* \leftarrow l; \quad r \leftarrow r_i^{(l)}$
    end if
  end for
end for
return $(r,\ l^*,\ p^*)$
function RefusalScore($H,\ \mathcal{V}$)
  $s_{\text{sum}} \leftarrow 0$
  for each $x \in \mathcal{V}$ **do**
    $z \leftarrow \textsc{LastLogits}(x\ , H)$
    $p_{\text{ref}} \leftarrow \sum_{t \in \mathcal{V}_{\text{ref}}} \textsc{Softmax}(z)_t$
    $s \leftarrow \log(p_{\text{ref}} + \epsilon) - \log(1 - p_{\text{ref}} + \epsilon)$
    $s_{\text{sum}} \leftarrow s_{\text{sum}} + s$
  end for
  return $s_{\text{sum}}/|\mathcal{V}|$
end function

direction is therefore regarded as the most accurate representation of the toxic concept within the latent space.

4.2 Intention Embedding Projection

In the second stage, InDe-LLM performs intention embedding projection to recover the underlying intention of prompts. This projection recovers the intention by introducing a null-space method[16] that transforms embeddings into a benign-invariant subspace, in which it can preserve the stability of benign instructions while projecting harmful ones back to their intentions.

Attackers always embed original harmful intentions into an extensive benign-like context to bypass the LLM's safeguards. These wrapping components shift the overall representation of inputs. For harmful intentions in particular, they are shifted toward the benign distribution, which makes existing defenses ineffective.

According to the first insight in Section 3, LLMs preserve a clearer understanding of benign and harmful intentions in the activation space, motivating us to project the wrapped prompt's representation h back to its original intention's representation h_0. To achieve that, InDe-LLM applies a learned projection $\Phi(h)$ that explicitly removes wrapping components and restores the original intention of input prompts.

Moreover, as illustrated in the second insight in Section 3, an asymmetry exists in the embedding distribution of benign and harmful prompt wrapping. When harmful intentions are wrapped with extra words or structures, their representations in activation space shift much more than those of benign intentions along the toxic concept direction. This motivates us to perform projection $\Phi(h)$ with the null-space method [16]. In detail, since the negligible distributional shifts caused by benign wrappings can be regarded as invariant, $\Phi(h)$ could map all prompts into a benign-invariant subspace and then optimize a projection matrix to recover only harmful intentions. In this way,

InDe-LLM can treat benign and harmful prompts differently, causing negligible impacts on benign prompts while focusing on detecting the original intentions of the jailbreak prompts.

To illustrate the projection $\Phi(h)$, we first provide several key definitions. Let $(x^{(i)}_{\text{jailbreak}}, x^{(i)}_{\text{raw}})\}_{i=1}^{n}$ denotes a collection of n paired prompts, where each pair consists of a jailbreak instruction $x^{(i)}_{\text{jailbreak}}$ (a malicious query obfuscated with wrapping) and its corresponding original harmful intention $x^{(i)}_{\text{raw}}$ with no obfuscation. In addition, a set of m benign instructions are denoted as $\{x^{(j)}_{\text{benign}}\}_{j=1}^{m}$. Let $H_{\text{j}} = [h^{(1)}_{\text{jailbreak}}, \ldots, h^{(n)}_{\text{jailbreak}}] \in \mathbb{R}^{d\times n}$, $\quad H_r = [h^{(1)}_{\text{raw}}, \ldots, h^{(n)}_{\text{raw}}] \in \mathbb{R}^{d\times n}$ represent the activations of jailbreak prompts and unwrapped harmful intentions respectively, and the activations of benign intentions are represented as $H_b = [h^{(1)}_{\text{benign}}, \ldots, h^{(m)}_{\text{benign}}] \in \mathbb{R}^{d\times m}$.

To extract the underlying intention while eliminating the wrapping component, we introduce a learnable linear projection matrix $W \in \mathbb{R}^{d\times d}$ to disentangle the intention from the original input. First, we enforce that the projection WH_{j} should retain the harmful intentions H_{r} while removing the benign-looking obfuscation introduced by jailbreak techniques. The objective is formalized as

$$\mathcal{L}_1 = \left\| WH_{\text{j}} - H_{\text{r}} \right\|_2^2, \tag{6}$$

which penalizes deviations between the projected jailbreak activations and their corresponding intention activations.

Second, based on our observation, the disentangling of benign instructions should not cause large shift along the toxic concept direction. Thus, benign instructions under W can be regarded as approximately invariant. This intuition can be formalized as

$$\forall h_{\text{benign}} \in H_{\text{b}}, \quad Wh_{\text{benign}} \approx h_{\text{benign}}. \tag{7}$$

Accordingly, we define the invariance loss,

$$\mathcal{L}_2 = \| WH_{\text{b}} - H_{\text{b}} \|_2^2 . \tag{8}$$

Minimizing $\mathcal{L}_2$ encourages W to act as a nearly identity mapping on benign inputs, effectively preserving their semantics.

Combining the two objectives, we obtain the final optimization goal,

$$W = \arg\min_{W} \left\| WH_{\text{j}} - H_{\text{r}} \right\|_2^2 + \left\| WH_{\text{b}} - H_{\text{b}} \right\|_2^2. \tag{9}$$

Eq. (9) has a straightforward closed-form solution. However, directly optimizing W complicates the trade-off between $\mathcal{L}_1$ and $\mathcal{L}_2$. Since the detection of jailbreak attacks depends critically on accurate recovery of the original intention, we instead follow [16] and employ the null-space method, which is a two–step strategy. First, we introduce a projector P to map instructions into a benign-invariant subspace, ensuring that benign prompts remain largely unaffected. Then, within this subspace, we optimize a matrix W' to recover the original intention of prompts, with a particular focus on handling jailbreak attacks. In this way, P contributes to the stability of the benign instructions, while W' is dedicated to correcting harmful shifts. This two–step strategy effectively alleviates the trade-off issue in directly optimizing W. Concrete steps are as follows.

(i) Estimate a benign-invariant projector P. We enforce benign invariance by first reparameterizing the matrix as $W - I = W'P$. We aim to ensure the activations of benign prompts remain unchanged, i.e.,

$$WH_b - H_b = (W - I)H_b = W'PH_b \approx 0, \tag{10}$$

which directly implies $\mathcal{L}_2 \approx 0$. To achieve this goal, we only need to enforce $PH_b \approx 0$. Then we can optimize W' solely with respect to $\mathcal{L}_1$, effectively decoupling the solutions for $\mathcal{L}_1$ and $\mathcal{L}_2$. The first task is to construct a projector P that guarantees $PH_b \approx 0$. As formally proven in [16], this can be

achieved by leveraging the null space of H_b. Specifically, given two matrices A and B, B is in the null space of A if and only if

$$BA = 0. \tag{11}$$

The null space of the covariance matrix of H_b is identical to that of H_b itself, but can be computed with significantly lower cost. Specifically, we form the non-central covariance matrix $\Sigma = H_b H_b^\top$ and perform singular value decomposition (SVD), $\Sigma = U\,\mathrm{diag}(S)\,U^\top$. Then, we discard the eigenvectors in U associated with non-zero eigenvalues, and define the remaining submatrix as U_0. Based on this, the projector P can be defined as follows:

$$P = U_0 U_0^\top. \tag{12}$$

This projection matrix restricts the distribution shifts in the null space of H_b, which ensures $PH_b \approx 0$. Intuitively, it identifies a subspace where benign inputs remain stable, so that subsequent updates do not compromise the model's utility.

(ii) Solve W' in the benign-invariant subspace. Since we have already constructed a projector P that enforces $\mathcal{L}_2$, we can drop this term from the optimization and concentrate solely on $\mathcal{L}_1$. To this end, we learn a linear matrix W' that maps jailbreak activations H_j back to their original harmful intentions H_r, while restricting updates to the benign-invariant subspace. Substituting $W = W'P + I$ into the objective Eq. (6) yields,

$$W' = \arg\min_{W'} \left\|W'PH_\mathrm{j} + H_\mathrm{j} - H_\mathrm{r}\right\|_2^2 + \lambda\left\|W'P\right\|_2^2, \tag{13}$$

where $\lambda\left\|W'P\right\|_2^2$ is a regularization term that stabilizes the optimization and $\lambda > 0$ controls the regularization strength.

With P determined, Eq. (13) admits a closed-form solution. Substituting it into $W = I + W'P$, we obtain the final projection operator

$$W = I + (H_\mathrm{r} - H_\mathrm{j})H_\mathrm{j}^\top P^\top \left(PH_\mathrm{j}H_\mathrm{j}^\top P^\top + \lambda PP^\top\right)^{-1} P. \tag{14}$$

The hidden representations h in H_b,H_j and H_r are extracted from a specific transformer layer l^* and token position p^*, which are the same as the layer and position that yield the most discriminative toxic concept direction r in prior analysis. This ensures that the projection W operates in the most semantically meaningful space for harmful intention detection.

4.3 Adaptive Activation Steering

After estimating the projection operator W, we project the embedding h of an input to obtain $h_0 = Wh$, which recovers its underlying intention before determining whether steering should be applied. Since blindly applying a steering vector can degrade performance on benign instructions, we design an adaptive activation steering module and only steer conditionally according to the toxicity score.

Toxicity Score Calculation. In this step, InDe-LLM calculates the toxicity score for each input. Typically, the activations of all inputs cluster within a region far from the origin. As a result, the relative distance between benign and malicious activations is much smaller than their distance to the origin, which makes effective separation difficult. For illustration, distinguishing the Earth from the Moon is challenging when viewed from the Sun. To address this, InDe-LLM first rebases the coordinate system to the center of the benign activations. Specifically, we compute the benign activation center $v \in \mathbb{R}^d$ and subtract it from embedding h_0 of each input,

$$\hat{h} = h_0 - v. \tag{15}$$

This step removes inputs' common features and highlights differences related to harmful semantics.

To quantify toxicity, we define the toxicity score $c(h)$ as the cosine similarity between the representation $\hat{h}$ and the toxic concept direction $r \in \mathbb{R}^d$,

$$c(h) = \frac{\langle \hat{h}, r \rangle}{\|\hat{h}\|_2 \, \|r\|_2}. \tag{16}$$

Activation Injection. Intuitively, a larger toxicity score $c(h)$ indicates stronger alignment with harmful semantics. An instruction is classified as harmful if the toxicity score exceeds a predefined threshold τ:

$$\text{is\_harmful}(h) = \begin{cases} 1, & \text{if } c(h) > \tau \\ 0, & \text{otherwise} \end{cases}. \tag{17}$$

Once a prompt is detected as harmful, we trigger a defense by injecting a steering vector into the model's internal representation. In detail, we use the toxic concept direction r as a steering vector and add its scaled copy to the hidden state during inference. This enhancement guides the model toward safe, policy-compliant refusals.

Formally, for a detected harmful instruction, the modified representation can be written as

$$h^{\text{inj}} = h + \alpha r, \tag{18}$$

where $\alpha > 0$ is a scaling hyperparameter controlling the strength of the intervention. We perform this intervention at all token positions. For inputs not flagged as harmful, no injection is applied.

5 Evaluation

The evaluation aims to answer the following research questions:

- **RQ1.** How effective is the defense in increasing the defense success rate against different jailbreak attacks? (Section 5.1)
- **RQ2.** Does the defense framework adversely impact the model's performance on benign datasets? (Section 5.2)
- **RQ3.** How well does the defense framework generalize to previously unseen jailbreak attacks? (Section 5.3)
- **RQ4.** How is the effectiveness of InDe-LLM affected by component designs, different model sizes, and hyperparameter selection? (Section 5.4)

LLM Backbones. The experiments were conducted on the instruct versions of three open-source LLMs, Llama-3.1-8B-Instruct [14], Qwen-2.5-7B-Instruct [20], and Gemma-2-9B-IT [42], all of which have undergone comprehensive alignment and post-training for instruction following. These models were selected due to their diverse training data and architectural designs. Additionally, InDe-LLM was evaluated across a range of Qwen-2.5 model sizes, from 3B to 14B.

Benchmark. To evaluate the effectiveness of InDe-LLM, we tested it against ten well-known jailbreak attacks. These attacks can be classified into four categories: (1) optimization-based methods, such as GCG [53] and SAA [3]; (2) encoding-based methods, including Zulu [48] and Cipher [50]; (3) template-based methods, such as AutoDAN [52], PAIR [10], AIM [1], Jailbroken [46], and ReNeLLM [13]; and (4) linguistics-based methods, such as DrAttack [28].

For the utility evaluation, we adopted standard benchmarks to assess the models' performance on general capabilities and reasoning tasks. Specifically, we used AlpacaEval [15] to measure instruction-following ability, Math500 [19] for evaluating mathematical problem-solving skills and Winogrande [37] for commonsense reasoning.

Metrics. We evaluated the defense capability and utility reservation of InDe-LLM using three metrics. The *Defense Success Rate (DSR)* quantifies the proportion of harmful prompts successfully

blocked by the defense mechanism, reflecting the model's capability to prevent unsafe or policy-violating outputs. A higher DSR indicates stronger robustness against jailbreak attacks. For utility preservation, we reported the *Win Rate* on AlpacaEval [15], with preferences judged by gpt-4-turbo. The win rate measures the percentage of benign prompts for which the enhanced model's responses are favored over those from the original, undefended model. In addition, we measured *Accuracy* (Acc) on Math500 [19] and Winogrande [37] to assess whether the defense affects mathematical and logical reasoning capabilities.

Baselines. We compared our method against three representative defense baselines, each adopting a distinct strategy. Antidote [39] extracts a sparse steering vector by retaining only a few dimensions whose variance best separates benign from harmful instructions, and uniformly adds this vector to every input, nudging the model toward refusal while leaving utility largely intact. Surgical [45] uses the orthogonalized vector modification method to enhance model safety and maintain general capabilities. CAST [27] adopts a PCA-derived steering vector and conditionally selects which to refuse, thus reducing unnecessary refusals. These approaches cover vector modification and condition detection approaches, providing a varied basis for comparison.

Setups. We conducted all experiments using PyTorch 2.3.0 [34] on a server equipped with NVIDIA Tesla V100 GPUs. The steering strength α was fixed at 1 to balance safety and utility. Through extensive empirical evaluation, we set the threshold τ to 0.2 by default, as this value achieves stable performance across many models.

Table 1. The jailbreak attack DSR↑ performance comparison. The best-performing methods are **bold**.

Method	AutoDAN	DrAttack	GCG	PAIR	SAA	Zulu	AIM	Cipher	Jailbroken	ReNeLLM	**Avg% ↑**
Llama-3.1-8B	63	83	92	74	4	82	93	0	92	49	63.2
Antidote	98	98	94	85	89	70	76	0	94	64	76.8
Surgical	98	8	92	60	97	8	**100**	30	75	36	60.4
CAST	62	81	95	88	9	96	94	**100**	98	55	77.8
InDe-LLM	**100**	**100**	**100**	**100**	**100**	**97**	95	98	**100**	**99**	**98.9**
Qwen-2.5-7B	31	61	88	61	15	86	35	0	82	21	48.0
Antidote	**98**	95	96	50	40	88	34	76	85	59	72.1
Surgical	39	5	51	32	0	55	85	0	34	12	31.3
CAST	47	78	95	62	25	99	63	**100**	89	39	69.7
InDe-LLM	96	**100**	**100**	**99**	**99**	**100**	**100**	**100**	**98**	**100**	**99.2**
Gemma-2-9B	16	12	94	18	0	2	6	0	79	25	25.2
Antidote	89	99	98	72	48	95	85	**100**	91	57	83.4
Surgical	2	13	54	14	0	26	0	0	55	0	16.4
CAST	13	94	98	60	8	93	4	**100**	80	27	57.7
InDe-LLM	**97**	**100**	**100**	**100**	**100**	**100**	**92**	**100**	**96**	**75**	**96.0**

5.1 Defense Effectiveness

To answer RQ1, we compared InDe-LLM against three representative baseline defense methods, Antidote [39], Surgical [45], and CAST [27]. The detailed results are reported in Table 1.

As shown in the table, InDe-LLM consistently demonstrates superior performance against a wide range of jailbreak attacks. On average, it achieves an average DSR of 98.0% across all evaluated models, indicating highly effective mitigation of adversarial prompts. This robustness holds across diverse attack methods and model architectures (Llama-3.1-8B, Qwen-2.5-7B, and Gemma-2-9B), showing that InDe-LLM captures generalizable harmful semantics rather than overfitting to specific attack patterns. Compared with baseline defenses, InDe-LLM yields substantial gains, improving DSR by at least 21.1% on Llama-3.1-8B, 27.1% on Qwen-2.5-7B, and 12.6% on Gemma-2-9B.

Regarding specific jailbreak attacks, InDe-LLM demonstrates consistent robustness, achieving a DSR of nearly 100% across almost all attack methods. For example, on Qwen-2.5-7B, InDe-LLM achieves a 100% DSR on DrAttack, GCG, Zulu, AIM, Cipher and ReNeLLM, showing its strong defense capability on various attacks.

In contrast, although Antidote performs well against certain attacks, such as AutoDAN and Jailbroken, it fails against others, like Cipher, with a DSR of 0%. This highlights that manipulating only a sparse subset of steering vector dimensions during inference is insufficiently adaptable to diverse attack methods. Surgical also suffers from large DSR fluctuations, since using a fixed and modified steering vector cannot effectively capture the toxic intention wrapped by different jailbreak attacks. CAST, which employs a simple detection method based on the input's original activation, struggles with jailbreak prompts that exhibit shifted distributions, allowing them to bypass its detection. As a result, CAST performs less effectively against certain attacks. For example, on Gemma-2-9B, CAST achieved a DSR of only 13% on AutoDAN and 4% on AIM.

Table 2. The performance comparison on utility benchmarks. The best-performing methods are **bold**. All metrics are higher-is-better: Win Rate↑, ACC↑, and Utility Score↑.

Model	AlpacaEval (Win Rate%↑)	Math (Acc%↑)	Winogrande (Acc%↑)	Utility Score%↑
Llama-3.1-8B	50.0	56.0	54.0	53.3
Antidote	17.2	40.0	48.0	35.1
Surgical	47.0	54.0	**54.0**	51.7
CAST	47.3	50.0	50.0	49.1
InDe-LLM	**48.0**	**56.0**	**54.0**	**52.7**
Qwen-2.5-7B	50.0	62.0	68.0	60.0
Antidote	13.4	**64.0**	59.0	45.5
Surgical	33.7	57.0	64.0	51.6
CAST	**47.5**	52.0	63.0	54.2
InDe-LLM	47.0	61.0	**66.0**	**58.0**
Gemma-2-9B	50.0	42.0	74.0	55.3
Antidote	21.5	35.0	70.0	42.2
Surgical	41.2	**40.0**	67.0	49.4
CAST	**47.0**	39.0	56.0	47.3
InDe-LLM	**47.0**	38.0	**73.0**	**52.7**

5.2 Utility Analysis

To answer RQ2, we evaluated the impact of InDe-LLM on model utility. We employed three widely-adopted benchmarks, each targeting a distinct capability: AlpacaEval for instruction-following (measured by Win Rate), MATH for mathematical reasoning (measured by Acc), and Winogrande for commonsense reasoning (measured by Acc). The results are summarized in Table 2. We define Utility Score as the average score of the three benchmarks to represent the utility of LLMs.

InDe-LLM maintains general utility comparable to the original, undefended models across all benchmarks, closely matching their performance. Specifically, for Qwen-2.5-7B, InDe-LLM achieves a win rate of 47.0% on AlpacaEval, indicating that it preserves the original model's general question-answering performance with minimal degradation. For Math and Winogrande, InDe-LLM also achieves near lossless performance on Llama-3.1-8B, confirming that our approach preserves the model's core reasoning capabilities while maintaining safety. These results suggest that InDe-LLM

effectively distinguishes between harmful and benign instructions. Once benign instructions are detected, our method operates losslessly, ensuring that utility performance is not compromised.

When compared to other baseline methods, InDe-LLM shows significant improvements. While Antidote and Surgical optimize the steering vectors to avoid rejecting benign instructions, they apply them to all input instructions, leading to performance degradation. For example, Antidote collapses on AlpacaEval (13.4% Acc) despite competitive Winogrande accuracy on Qwen-2.5-7B, reflecting its over-refusal tendency. This is because these methods overly rely on the accuracy of the steering vectors, and their fixed vector cannot adapt well to varying input types, causing unnecessary refusals for benign instructions. On the other hand, CAST employs a simple detection method that is insufficient to handle the diverse range of instruction categories seen across different benchmarks. This simplicity limits its ability to generalize effectively to different tasks.

5.3 Transferability

To assess the transferability of our defense framework (RQ3), we performed a comprehensive cross-dataset evaluation. Since attacks within the same category exhibit similar wrapping strategies, we included one attack method from each of the four jailbreak categories. Specifically, for each evaluation, we used one dataset from the four attacks to calculate the intention projection matrix and then tested the resulting defense on the remaining three datasets.

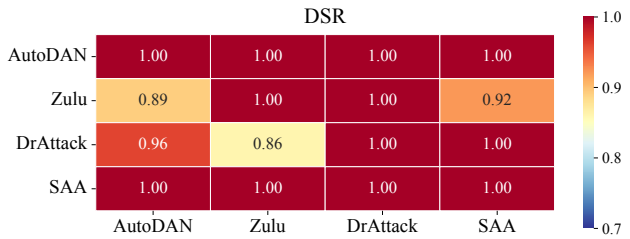


Fig. 3. Transferability evaluation across four datasets. Each cell shows the Defense Success Rate (DSR↑) when the defense is trained on one dataset (columns) and tested on others (rows). Values closer to 1.0 indicate better defense generalization.

As shown in Figure 3, InDe-LLM demonstrates strong generalization across diverse attack distributions on Qwen-2.5-7B. In most transfer scenarios, the defense maintains a high DSR, exceeding 0.96 against unseen attacks in most cases. Even across different attack categories, InDe-LLM consistently sustains high DSR. This suggests that InDe-LLM is not overly dependent on specific attack patterns but instead captures broadly generalizable toxic semantics.

Notably, while a slight performance drop (e.g., an 86% DSR) is observed when transferring from some datasets to DrAttack [28], the overall defense effectiveness remains high. This may be because DrAttack utilizes in-context learning prompts with many benign examples, which affect our defense. Overall, the experiments demonstrate that InDe-LLM retains robust performance and remains effective against unseen attacks.

5.4 Ablation Study and Parameter Analysis

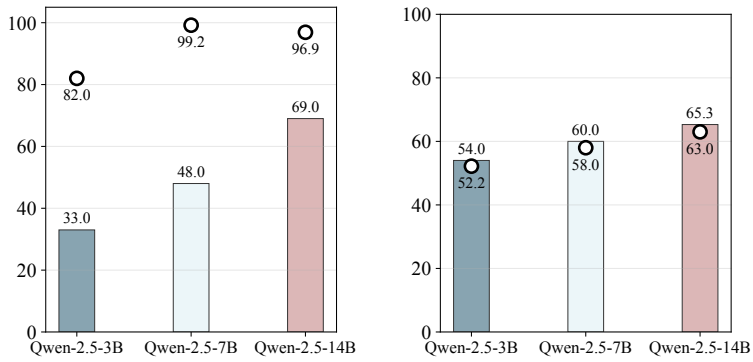
To address RQ4, we conducted an ablation study to better understand the contribution of each component in InDe-LLM. We also conducted parameter analyses, including the sensitivity of the defense to model size as well as hyperparameters such as the steering strength α and the regularization strength λ. These analyses provide deeper insights into the design choices of InDe-LLM and clarify how different factors influence its robustness and utility.

Table 3. Ablation study on the effectiveness of different components in InDe-LLM.

Model	Method	DSR%↑	Utility Score%↑
Llama-3.1-8B	InDe-LLM	98.9	**52.7**
	w/o *intention projection*	65.2	52.0
	w/o *intention projection*$^-$	**99.6**	17.0
Qwen-2.5-7B	InDe-LLM	99.2	58.0
	w/o *intention projection*	81.7	**59.7**
	w/o *intention projection*$^-$	**99.7**	4.0
Gemma-2-9B	InDe-LLM	96.0	52.7
	w/o *intention projection*	57.9	**53.8**
	w/o *intention projection*$^-$	**98.1**	5.7

Ablation Study. As shown in Table 3, we examined the effect of the design component by removing the intention embedding projection module and the benign constraint. Removing the benign constraint means solving the projection matrix without first projecting onto P. In Table 3, these two settings are denoted as *w/o intention projection* and *w/o intention projection*$^-$, respectively. Removing the intention embedding projection module reduced the jailbreak DSR sharply. For example, the DSR drops from 98.9% to 65.2% on Llama-3.1-8B. Without this module, the defense essentially falls back to directly applying the original input to adaptive activation steering without intention extraction, which may cause obscuration by the wrapping components. As a result, the toxic concept direction can't capture these benign-looking jailbreak instructions. In contrast, the intention projection module learns a linear operator W that isolates a common "toxic concept" shared by these wrapped instructions. This approach consistently isolates the harmful intention, ensuring it remains detectable in the presence of jailbreak attacks.

When the benign constraint was removed, the framework still achieved a high DSR, but the utility degraded substantially. On Qwen-2.5-7B, for example, the DSR remains as high as 99.2%, while the utility score drops sharply from 58.0% to 4.0%. This indicates that directly optimizing over both benign and harmful instructions makes it difficult to strike a balance between safety and utility. We observe the same trend on Llama-3.1 and Gemma-2.



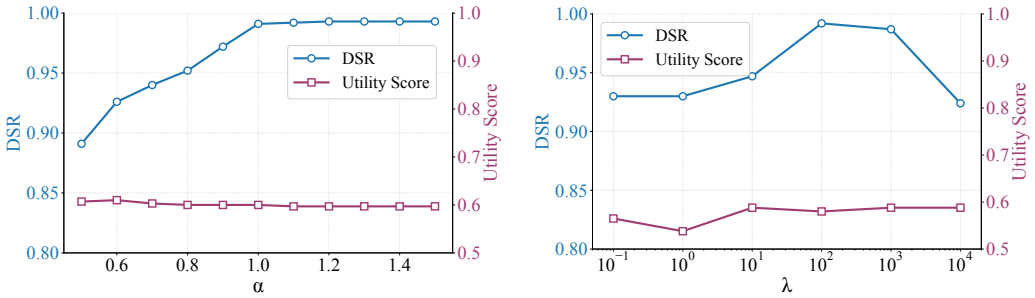
(a) DSR%↑ (b) Utility Score%↑

Fig. 4. Impact of model size on defense performance for Qwen-2.5 variants (3B, 7B, 14B). The bar charts indicate the baseline performance without defense, while the circular markers denote the performance after applying InDe-LLM.

Sensitivity to Model Size. To assess the influence of model size, we evaluated the defense performance of InDe-LLM on three Qwen-2.5 model variants with parameter sizes of 3B, 7B, and 14B, as shown in Figure 4. For each model size, we report both DSR and utility scores, with bar charts representing baseline performance without defense and circular markers representing InDe-LLM.

As illustrated in Figure 4a, InDe-LLM substantially improves the DSR across all model sizes. For the smallest model (3B), DSR increased dramatically from a baseline of 33.0% to 82.0%. For larger models, the DSR also saw a considerable increase, consistently reaching high scores of 99.2% (7B) and 96.9% (14B), respectively. These results show that InDe-LLM remains effective across scales.

Figure 4b shows that the general utility of models under InDe-LLM remains largely unaffected by model size. Across different model scales, the utility scores stay close to those of the original models. For example, on Qwen-2.5-3B the utility is 52.2%, only 1.8% lower than the baseline. These results demonstrate that InDe-LLM enhances safety without compromising the utility of the models.



(a) Effect of steering strength α.

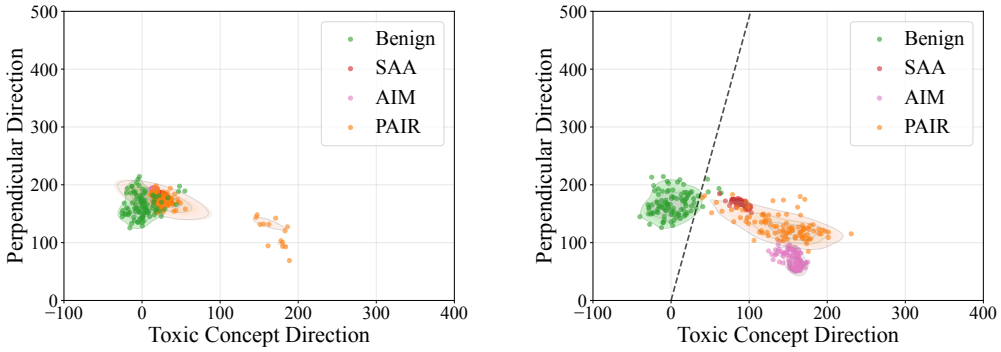
(b) Effect of regularization strength λ.

Fig. 5. Impact of Hyperparameters on Qwen-2.5-7B

Hyperparameter Analysis. To evaluate the effect of the hyperparameters α and λ, we conducted experiments over a range of values and assessed their impact on both jailbreak DSR and utility score. Specifically, α controls the scaling strength of the steering vector, while λ balances the regularization term. Detailed results and performance curves are shown in Figure 5. From Figure 5a, we observe that as the steering scaling factor α increases, the DSR improves, while the utility score shows a decreasing trend. This highlights the need to strike a balance between robustness and utility. When α is small, the steering strength may be too weak to fully suppress jailbreak attacks, leading to incomplete refusals. As α increases, InDe-LLM demonstrates its defensive capability in the upper limit. However, an excessively large α inevitably damages the normal language ability of the model, underscoring the importance of choosing an appropriate trade-off point. We also analyzed the effect of the regularization parameter λ, which controls the regularization term. As illustrated in Figure 5b, an increase in λ has a minimal effect on the utility score. This observation aligns with our use of the null-space method, which already constrains benign instructions to remain stable. Consequently, λ constrains the updates of the intention projection matrix, which helps maintain stable and reliable intention projection in the presence of jailbreak attacks.

We found that the choice of λ plays a crucial role in achieving optimal DSR. A value that is too small (e.g., 10^{-1}) tends to cause overfitting to the training set, thereby reducing the model's generalization capabilities. In contrast, setting λ too large (e.g., 10^4) makes the parameter updates overly constrained, preventing sufficient intention projection ability. Our experiments show that the best DSR is obtained at around $\lambda = 100$.

Across the various values of α and λ, InDe-LLM consistently maintained a strong DSR, above 90% in almost all cases while preserving good utility. This further validates the stability of InDe-LLM.



(a) Before intention projection.

(b) After intention projection.

Fig. 6. Visualization of prompts in the activation space on Gemma-2-9B.

5.5 Visualization

Figure 6 illustrates the distribution of prompts in the activation space. As shown in Figure 6a, benign and jailbreak prompts are highly entangled, and no clear decision boundary can effectively separate them. In contrast, Figure 6b presents the distribution after applying InDe-LLM. With intention embedding projection, malicious prompts become well separated from benign ones, allowing a simple threshold-based boundary. The dashed line depicts a linear decision boundary that cleanly distinguishes benign prompts from jailbreak prompts. We further observe that InDe-LLM maintains the stability of benign prompts while projecting jailbreak attacks back along the toxic concept direction, which validates the insight about the asymmetry of prompt wrapping. Overall, this visualization demonstrates that InDe-LLM restores the separability obscured by adversarial perturbations, supporting the insights discussed in Section 3.

6 Discussion

Steering Vector Calculation. In our design, we used a difference-in-means method for steering vector calculation. This choice keeps the method training-free and parameter-light, which is important for a lightweight defense. At the same time, recent analyses report that difference-in-means surpasses heavy-weight alternatives, such as PCA-based methods, across diverse tasks [21]. Given these practical and empirical advantages, we did not include other strategies in our design.

Model Dependency. LLMs exhibit considerable architectural diversity. Although InDe-LLM has demonstrated effectiveness across varying parameter scales and different model families, the model architectures are still rapidly evolving, and new designs are continuously emerging. Due to limited computational resources, we have not tested our method on models at a greater scale. While our experiments have covered a representative range of model sizes, evaluating larger-scale LLMs remains a direction for future work to fully understand the scalability of our approach.

Currently, InDe-LLM primarily targets LLMs with transformer-based structures, as it requires extracting activations from relevant layers for analysis. The emergence of novel architectures [12, 17] attempting to replace the transformer might render InDe-LLM ineffective. However, these new architectures still have internal layers with activation spaces where semantic concepts are encoded. This allows us to adapt InDe-LLM to new architectures. We plan to explore it in the future.

7 Related Work

Given the significant impact of jailbreak attacks on LLMs and their potential use in compromising AI-powered applications, extensive research has focused on developing defenses, which can be categorized into two types, training-based and training-free.

7.1 Training-Based Defense

Training-based defenses primarily focus on post-training alignment [4], including techniques such as RLHF [6, 49] and deliberative alignment [18]. These methods retrain models using safe, carefully annotated datasets to enhance response safety. However, their reliance on annotated data limits their effectiveness against out-of-distribution attacks [46], making them less effective against unseen jailbreak methods. Moreover, these methods incur high computational costs, which highlights the need for more efficient defense strategies. These limitations underscore the need for more adaptable and computationally lightweight defenses that can generalize to novel attacks.

7.2 Training-Free Defense

Many existing training-free defenses operate directly on inputs. Prompt detection [2, 22] identifies malicious queries from simple features such as perplexity and sequence length. Prompt perturbation [7, 35] improves reliability by masking, paraphrasing, or otherwise perturbing prompts and checking consistency across variants. System Prompt Safeguards [38, 44, 54] introduce hidden safety instructions into the system prompt to bias models toward safe responses. Refinement [26, 47] leverages self-correction or auxiliary LLMs to iteratively filter or rewrite unsafe generations. However, these approaches focus on analyzing the surface-level semantics of prompts, which can be easily bypassed and yield high false positives.

In this paper, we focus on activation steering-based defenses that intervene at the representation level. LLMs are known to encode semantic concepts as linear directions in activation space [30, 33], which enables *activation steering* [5, 32, 41, 51] as a training-free mechanism by directly modifying hidden states. However, directly adding steering vectors to all inputs severely degrades utility [5]. Some refinements [27, 39, 45] extract steering vectors from contrastive data and apply them conditionally or after modification, yet these methods still struggle with weak intervention and remain prone to distributional shifts under adversarial wrappers. Our work instead takes an intention-disentanglement perspective grounded in two empirical observations about intention boundaries and the effects of prompt wrapping.

8 Conclusion

In this work, we introduced InDe-LLM, a framework for mitigating jailbreak attacks in LLMs. InDe-LLM disentangles the underlying intention of adversarial prompts and provides a lightweight, training-free defense mechanism. Through comprehensive experiments on multiple model sizes and architectures, we showed that InDe-LLM consistently achieves high DSR while largely preserving utility. Moreover, visualization results demonstrate how InDe-LLM restores the separability obscured by adversarial manipulations, offering an interpretable explanation of its effectiveness. Overall, InDe-LLM underscores its scalability and practicality for training-free defense, positioning it as a promising approach for safeguarding future LLMs against evolving jailbreak threats.

9 Data Availability

In line with the FSE Open Science Policy, our artifact is publicly available at https://anonymous.4open.science/r/InDe-LLM.

Acknowledgments

We sincerely thank the anonymous reviewers for their insightful feedback. This research is sponsored in part by the State Cryptography Administration Project (No.2025NCSF02055) and the NSFC Program (No.U2441238).

References

[1] [n. d.]. AIM Prompt Jailbreak Attack. https://oxtia.com/chatgpt-jailbreak-prompts/aim-prompt/.

[2] Gabriel Alon and Michael Kamfonas. 2023. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132* (2023).

[3] Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024. Jailbreaking leading safety-aligned llms with simple adaptive attacks. *arXiv preprint arXiv:2404.02151* (2024).

[4] Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase, Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, Oliver Sourbut, et al. 2024. Foundational challenges in assuring alignment and safety of large language models. *arXiv preprint arXiv:2404.09932* (2024).

[5] Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems* 37 (2024), 136037–136083.

[6] Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Röttger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. 2023. Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. *arXiv preprint arXiv:2309.07875* (2023).

[7] Bochuan Cao, Yuanpu Cao, Lu Lin, and Jinghui Chen. 2023. Defending against alignment-breaking attacks via robustly aligned llm. *arXiv preprint arXiv:2309.14348* (2023).

[8] Nicholas Carlini, Milad Nasr, Christopher A. Choquette-Choo, Matthew Jagielski, Irena Gao, Pang Wei Koh, Daphne Ippolito, Florian Tramèr, and Ludwig Schmidt. 2023. Are aligned neural networks adversarially aligned?. 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*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/c1f0b856a35986348ab3414177266f75-Abstract-Conference.html

[9] Zhiyuan Chang, Mingyang Li, Yi Liu, Junjie Wang, Qing Wang, and Yang Liu. 2024. Play Guessing Game with LLM: Indirect Jailbreak Attack with Implicit Clues. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, 5135–5147. doi:10.18653/V1/2024.FINDINGS-ACL.304

[10] 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*. IEEE, 23–42. doi:10.1109/SATML64287.2025.00010

[11] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30 (2017).

[12] Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. https://openreview.net/forum?id=ztn8FCR1td

[13] Peng Ding, Jun Kuang, Dan Ma, Xuezhi Cao, Yunsen Xian, Jiajun Chen, and Shujian Huang. 2023. A wolf in sheep's clothing: Generalized nested jailbreak prompts can fool large language models easily. *arXiv preprint arXiv:2311.08268* (2023).

[14] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv e-prints* (2024), arXiv–2407.

[15] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475* (2024).

[16] Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Shi Jie, Xiang Wang, Xiangnan He, and Tat-Seng Chua. 2024. Alphaedit: Null-space constrained knowledge editing for language models. *arXiv preprint arXiv:2410.02355* (2024).

[17] Albert Gu and Tri Dao. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *CoRR* abs/2312.00752 (2023). arXiv:2312.00752 doi:10.48550/ARXIV.2312.00752

[18] Melody Y Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Helyar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, et al. 2024. Deliberative alignment: Reasoning enables safer language models. *arXiv preprint arXiv:2412.16339* (2024).

[19] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874* (2021).

[20] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186* (2024).

[21] Shawn Im and Yixuan Li. 2025. A unified understanding and evaluation of steering methods. *arXiv preprint arXiv:2502.02716* (2025).

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

[23] Anubhav Jangra, Sourajit Mukherjee, Adam Jatowt, Sriparna Saha, and Mohammad Hasanuzzaman. 2023. A survey on multi-modal summarization. *Comput. Surveys* 55, 13s (2023), 1–36.

[24] Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2023. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems* 36 (2023), 24678–24704.

[25] Ziwei Ji, Lei Yu, Yeskendir Koishekenov, Yejin Bang, Anthony Hartshorn, Alan Schelten, Cheng Zhang, Pascale Fung, and Nicola Cancedda. 2025. Calibrating Verbal Uncertainty as a Linear Feature to Reduce Hallucinations. *arXiv preprint arXiv:2503.14477* (2025).

[26] Heegyu Kim, Sehyun Yuk, and Hyunsouk Cho. 2024. Break the breakout: Reinventing lm defense against jailbreak attacks with self-refinement. *arXiv preprint arXiv:2402.15180* (2024).

[27] Bruce W Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. 2024. Programming refusal with conditional activation steering. *arXiv preprint arXiv:2409.05907* (2024).

[28] Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. 2024. Drattack: Prompt decomposition and reconstruction makes powerful llm jailbreakers. *arXiv preprint arXiv:2402.16914* (2024).

[29] Yichen Li, Zhiting Fan, Ruizhe Chen, Xiaotang Gai, Luqi Gong, Yan Zhang, and Zuozhu Liu. 2025. Fairsteer: Inference time debiasing for llms with dynamic activation steering. *arXiv preprint arXiv:2504.14492* (2025).

[30] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 746–751.

[31] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[32] Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. 2023. Steering llama 2 via contrastive activation addition. *arXiv preprint arXiv:2312.06681* (2023).

[33] Kiho Park, Yo Joong Choe, and Victor Veitch. 2023. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658* (2023).

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[35] Alexander Robey, Eric Wong, Hamed Hassani, and George J Pappas. 2023. Smoothllm: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684* (2023).

[36] Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2023. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension. *Comput. Surveys* 55, 10 (2023), 1–45.

[37] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Commun. ACM* 64, 9 (2021), 99–106.

[38] Reshabh K Sharma, Vinayak Gupta, and Dan Grossman. 2024. Spml: A dsl for defending language models against prompt attacks. *arXiv preprint arXiv:2402.11755* (2024).

[39] Guobin Shen, Dongcheng Zhao, Yiting Dong, Xiang He, and Yi Zeng. 2024. Jailbreak antidote: Runtime safety-utility balance via sparse representation adjustment in large language models. *arXiv preprint arXiv:2410.02298* (2024).

[40] 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*. 1671–1685.

[41] Leheng Sheng, Changshuo Shen, Weixiang Zhao, Junfeng Fang, Xiaohao Liu, Zhenkai Liang, Xiang Wang, An Zhang, and Tat-Seng Chua. 2025. Alphasteer: Learning refusal steering with principled null-space constraint. *arXiv preprint arXiv:2506.07022* (2025).

[42] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118* (2024).

[43] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Activation addition: Steering language models without optimization. *arXiv e-prints* (2023), arXiv–2308.

[44] Jiongxiao Wang, Jiazhao Li, Yiquan Li, Xiangyu Qi, Junjie Hu, Yixuan Li, Patrick McDaniel, Muhao Chen, Bo Li, and Chaowei Xiao. 2024. Mitigating fine-tuning based jailbreak attack with backdoor enhanced safety alignment. *arXiv preprint arXiv:2402.14968* (2024).

[45] Xinpeng Wang, Chengzhi Hu, Paul Röttger, and Barbara Plank. 2025. Surgical, Cheap, and Flexible: Mitigating False Refusal in Language Models via Single Vector Ablation. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.

[46] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2023. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems* 36 (2023), 80079–80110.

[47] Xianjun Yang, Xiao Wang, Qi Zhang, Linda Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. 2023. Shadow alignment: The ease of subverting safely-aligned language models. *arXiv preprint arXiv:2310.02949* (2023).

[48] Zheng-Xin Yong, Cristina Menghini, and Stephen H Bach. 2023. Low-resource languages jailbreak gpt-4. *arXiv preprint arXiv:2310.02446* (2023).

[49] Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. Rrhf: Rank responses to align language models with human feedback. *Advances in Neural Information Processing Systems* 36 (2023), 10935–10950.

[50] Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. 2023. Gpt-4 is too smart to be safe: Stealthy chat with llms via cipher. *arXiv preprint arXiv:2308.06463* (2023).

[51] Weixiang Zhao, Jiahe Guo, Yulin Hu, Yang Deng, An Zhang, Xingyu Sui, Xinyang Han, Yanyan Zhao, Bing Qin, Tat-Seng Chua, and Ting Liu. 2025. AdaSteer: Your Aligned LLM is Inherently an Adaptive Jailbreak Defender. *CoRR* abs/2504.09466 (2025). arXiv:2504.09466 doi:10.48550/ARXIV.2504.09466

[52] Sicheng Zhu, Ruiyi Zhang, Bang An, Gang Wu, Joe Barrow, Zichao Wang, Furong Huang, Ani Nenkova, and Tong Sun. 2023. AutoDAN: Automatic and Interpretable Adversarial Attacks on Large Language Models. *CoRR* abs/2310.15140 (2023). arXiv:2310.15140 doi:10.48550/ARXIV.2310.15140

[53] Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. *CoRR* abs/2307.15043 (2023). arXiv:2307.15043 doi:10.48550/ARXIV.2307.15043

[54] Xiaotian Zou, Yongkang Chen, and Ke Li. 2024. Is the system message really important to jailbreaks in large language models? *arXiv preprint arXiv:2402.14857* (2024).

Received 2025-09-12; accepted 2025-12-22