Shaping the Safety Boundaries: Understanding and Defending Against Jailbreaks in Large Language Models

Lang Gao[1,2], Jiahui Geng[1], Xiangliang Zhang[3], Preslav Nakov[1], Xiuying Chen[1*]

[1]MBZUAI [2]Huazhong University of Science and Technology
[3]University of Notre Dame
{Lang.Gao, Jiahui.Geng, Preslav.Nakov, Xiuying.Chen}@mbzuai.ac.ae
xzhang33@nd.edu

⚠ **WARNING: This paper contains model outputs that may be considered offensive.**

Abstract

Jailbreaking in large language models (LLMs) poses major security risks by tricking models into generating harmful text. However, there is limited understanding of how jailbreaking operates, which makes it difficult to develop effective defenses. In this work, we conduct a large-scale analysis of seven jailbreak methods and uncover that inconsistencies in previous studies arise from insufficient observation samples. Our analysis reveals that jailbreaks shift harmful activations beyond a defined *safety boundary*, where LLMs become less sensitive to harmful information. We also find that the low and the middle layers are critical in driving these shifts, while deeper layers play a lesser role. Leveraging these insights, we propose a novel defense mechanism called *Activation Boundary Defense* (ABD), which adaptively constrains activations within the safety boundary. To further optimize performance, we use Bayesian optimization to select the most effective layers for ABD application and confirm that the low and middle layers have the greatest impact, consistent with our earlier observations. Experiments across multiple benchmarks demonstrate that ABD achieves an average defense success rate (DSR) of over 98% against various jailbreak attacks, with less than 2% impact on the model's overall capabilities.

1 Introduction

The widespread use of Large Language Models (LLMs) across various fields (Kaddour et al., 2023; Xu et al., 2025c; Chen et al., 2025; Xie et al., 2025; Liu et al., 2024c; Xu et al., 2025b; Wang et al., 2025b) has raised concerns about the safety of the output and the robustness of these models (Ye et al., 2024; Xu et al., 2025a; Huang et al., 2025; Wang et al., 2025c). It has been shown that *jailbreak attacks*, which use crafted prompts to deceive LLMs into generating harmful content, can bypass LLM's safety alignment (Liu et al., 2024b; Zou et al., 2023; Song et al., 2025), and a lot of research has focused on developing defense mechanisms and counter-prompts to mitigate such attacks (Robey et al., 2023; Xie et al., 2024).

Understanding the internal mechanisms of jailbreak attacks is crucial for both explaining their operation and developing effective defenses. While existing studies have made initial explorations (Yu et al., 2024; Ball et al., 2024a; Lin et al., 2024), fundamental disagreements remain about the attack mechanisms. The academic community continues to debate two key aspects of explainability: (*i*) The localization of attack impacts across the model layers, with competing findings suggesting primary responsibility in low (He et al., 2024), middle (Zhou et al., 2024b; Shen et al., 2024), or deep layers (Li et al., 2024), and (*ii*) the progression pattern of malicious activation, with conflicting observations about gradual shifts versus abrupt transitions in the middle layers (Zhao et al., 2024a; Shen et al., 2024). Our experiments suggest that these inconsistencies may stem from methodological limitations in prior analysis, particularly their reliance on small prompt samples (typically ~ 100). On the defense front, current approaches face dual challenges: Most detection methods either require additional training (Zhao et al., 2024a) or demonstrate limited generalizability due to small-sample probing (Shen et al., 2024), while mitigation strategies often degrade normal model capabilities.

In this work, we aim to better understand the mechanism of how jailbreaking works. In particular, we provide our explanation of jailbreaks based on a comprehensive analysis of over *30,000* samples, a significantly larger scale than previous studies. Figure 1 shows the projection of three types of prompts: *benign prompts* that contain no harm-

[*]Corresponding author.

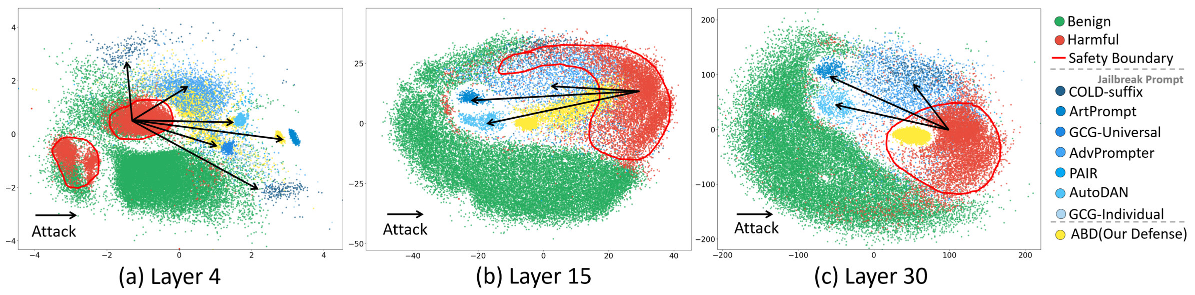


Figure 1: Projected activation space overview of Vicuna-7B-v1.3 across different layers. Harmful activations are observed to cluster together, and we define the surrounding boundary as the *safety boundary*. The attack arrow indicates that jailbreak prompts shift harmful activations into the benign space to evade safety checks.

ful information, *harmful prompts* that attempt to induce the LLM to generate harmful content but fail, and various *jailbreak prompts* that successfully induce harmful outputs, collected from different types of jailbreak attack methods. The projection is based on the last token representations across different layers, referred to as *activations*, as they capture the model's overall understanding of the entire input sequence (Radford et al., 2019; Zou et al., 2023). Our results reveal distinct clustering patterns, where different prompts form separate activation clusters. Notably, harmful prompts form a contained cluster within a specific region that we define as the *safety boundary*, representing the LLM's internal safety check that typically prevents harmful activations from escalating into harmful outputs. In contrast, jailbreak prompts push activations beyond the safety boundary into an unmonitored space where harmful content can be generated. This shift is most pronounced in the low and middle layers, indicating their critical role for jailbreak success.

This finding motivates a new defense mechanism that we propose, *Activation Boundary Defense* (ABD), which constrains jailbreak prompt activations within a safety boundary by applying penalties to activation values. Specifically, we impose minimal penalties within the boundary, but sharply increase them beyond it, thereby preserving model utility. The penalty function considers various factors, including the defense layer and the defense dimension, which balance the performance on defense and on general tasks. We propose a novel objective that maximizes the Defense Success Rate (DSR) while minimizing the number of penalized layers. This objective is implemented with an adapted Bayesian Optimization method (Jones et al., 1998), which iteratively proposes and refines selection suggestions.

Experiments on AdvBench (Zou et al., 2023) show that ABD has good generalization, achieving an average DSR of over 98% against various forms of jailbreak attacks. For general ability tasks, the performance drop of the model equipped with ABD does not exceed 2%, which is substantially lower than the drop of up to 37% with other defenses.

Our contributions can be summarized as follows:
• We uncover a comprehensive activation distribution and introduce the *safety boundary*, resolving contradictions in prior work and highlighting the vulnerability of the low and middle layers.
• We propose a lightweight, extensible defense that penalizes only targeted samples and a few key layers, ensuring efficiency and precision.
• Our experiments on benchmark datasets demonstrate that ABD achieves an average DSR of over 98% against various jailbreak attacks, with less than a 2% impact on general capabilities.

2 Related Work

Jailbreak attacks on LLMs. Jailbreak attacks craft prompts to induce harmful outputs from LLMs, evolving from early manually designed approaches to more sophisticated techniques. Early methods, such as DAN (Shen et al., 2023), rely on manually constructed prompts. Later, optimization-based methods (Zou et al., 2023; Liu et al., 2024b; Guo et al., 2024) emerged, using iterative optimization strategies to refine harmful prompts and enhance their stealthiness. Model-based methods (Chao et al., 2023; Ding et al., 2024; Paulus et al., 2024; Huang et al., 2024) use attacker LLMs to autonomously generate and improve jailbreak prompts. Meanwhile, rule-based methods (Jiang et al., 2024; Liu et al., 2024d) rewrite prompts using predefined rules that effectively deceive the model. In this work, we analyze the mechanisms behind these jailbreak techniques.

Defense against jailbreak attacks. Defense strategies for jailbreak attacks typically fall into three categories. One common approach enhances model safety and robustness by reformulating inputs to counter jailbreak prompts through techniques such as backtranslation (Wang et al., 2024), paraphrasing (Jain et al., 2023), self-reminding (Xie et al., 2023), and adversarial prompting (Zhou et al., 2024a). Another approach focuses on identifying jailbreak prompts using indicators, such as classifying input sequences based on perplexity and sequence length (Alon and Kamfonas, 2023) or using linear classifiers to detect jailbreak activations and decide whether to respond (Zhao et al., 2024a). A more recent trend directly manipulates the model's internal representations (Xu et al., 2024; Li et al., 2024; Liu et al., 2024a) or editing the model (Zhao et al., 2024b). Unlike these methods, our approach directly constrains activations within a safety boundary, avoiding extra tokens or modules.

Mechanistic interpretability of LLMs. One purpose of interpretable analysis of LLMs is to enable deeper understanding and diagnosis of their abnormal behaviors (Wang et al., 2025a; Nasim et al., 2025; Gao et al., 2025). The growing concern about LLM safety has therefore sparked increasing interest in interpreting LLM features in jailbreak prompts. For example, Ball et al. (2024b) identified a common mechanism whereby jailbreaks reduce the harmfulness perception in most LLMs. Similarly, Li et al. (2024) investigated patterns that trigger the model to recognize safety issues. Zhou et al. (2024b) discovered that the vocabulary mappings of activations significantly changed when processing jailbreak inputs. Research efforts have also focused on proposing corresponding defense methods based on interpretability results. For instance, Zhao et al. (2024a) and Shen et al. (2024) modeled how jailbreak activations transfer between benign and harmful activation spaces as layers deepen. They designed adaptive defenses whose strength varies across different layers. However, the limited data in all previous studies often led to controversy and ambiguity.

3 Understanding Jailbreaks in LLMs

3.1 Controversy in Literature

Disputes on layer importance. There have been different opinions about which layers are most important, mainly due to varying perspectives. For example, He et al. (2024) argued that the low layers were essential because they treated jailbreak activations as benign ones. Zhou et al. (2024b) believed that the low and the middle layers were essential based on tokens generated from activations. Li et al. (2024) argued that deep layers were critical, as their proposed safety pattern showed greater values in these layers. Similarly, He et al. (2024) focused on activations, as we do as well. However, their experimental setup suffers from a key limitation: a small sample size used in the analysis. As shown in Figure 2(a), under the original settings, benign and harmful activations are linearly separable, with most jailbreak samples misclassified as benign. After scaling, half of the jailbreak activations align with harmful ones, making the separation nonlinear. This underscores the risks of small-scale studies and the need for large-scale analysis.

Disputes on the jailbreak mechanism. There are differing perspectives on the internal mechanism of jailbreaks. Zhao et al. (2024a) viewed jailbreak as a gradual process, where activations transition from harmful to benign spaces as they pass from the low to the deep layers. In contrast, Shen et al. (2024) argued that jailbreaks manifest abruptly in the deeper layers, with activations positioned between harmful and benign spaces. Both studies relied on fewer than 100 samples per activation type, contributing to similar inconsistencies. As shown in Figure 2(b), before scaling, jailbreak activations align closely with benign ones, consistent with Zhao et al. (2024a). However, after scaling, they diverge from both harmful and benign activations. When considering Shen et al. (2024), similar conflicts arise between pre- and post-scaling results (details in Appendix C.1, Figure 5). These conflicting observations cast doubt on their conclusions regarding the true mechanism of jailbreaks.

3.2 Experimental Settings

To address the above limitations, we conducted a large-scale analysis on a 300 times greater dataset.

Dataset and Model. Our dataset consists of 32,507 samples in three categories: benign, harmful, and jailbreak. For benign samples, we used a subset from the Alpaca dataset (Taori et al., 2023), as it has been carefully curated to ensure that only safe content is included. We collected harmful samples from five datasets in RedEval (Bhardwaj and Poria, 2023) and AdvBench (Zou et al., 2023).

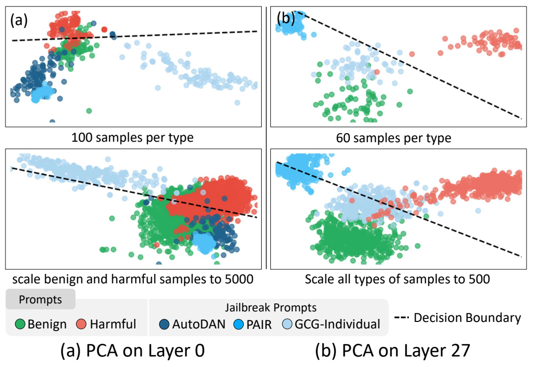


Figure 2: PCA visualizations reveal the limitations of small-sample sizes: (a) Top: 100 samples per type (He et al., 2024); Bottom: 5,000 benign and harmful samples. (b) Top: 60 samples per type (Zhao et al., 2024a); Bottom: 500 samples per type.

These datasets encompass a wide variety of harmful and misleading prompts. For jailbreak samples, we applied seven different jailbreak attack methods, including GCG-Individual and GCG-Universal (Zou et al., 2023), AdvPrompter (Paulus et al., 2024), COLD-Suffix Attack (Guo et al., 2024), AutoDAN (Liu et al., 2024b), PAIR (Chao et al., 2023), and ArtPrompt (Jiang et al., 2024), on AdvBench. Detailed statistics can be found in Table 4 in Appendix A. We used Vicuna-7B-v1.3 (Chiang et al., 2023), a 32-layer Transformer, fine-tuned on a safety-aligned LLM. Although it can identify harmful information, it remains vulnerable to jailbreak attacks (Chu et al., 2024), making it an ideal model for analyzing attacks and defenses.

MDS projection. Our projection was based on the *activation*, the last token vector of the input. As discussed in §3.1, linear classification may be insufficient to model the boundary between different prompts. Therefore, we adopted Multi-Dimensional Scaling (MDS) (Carroll and Arabie, 1998) as our dimensionality reduction method, as PCA assumes linear separability (Anowar et al., 2021), and t-SNE focuses on preserving local structures while distorting the global distribution (Van der Maaten and Hinton, 2008).

3.3 Jailbreak Mechanisms Findings

3.3.1 Activation Distribution

We present the activation distributions of representative layers in Figure 1, with the full version available in Figures 7,8,9 in Appendix B. Our direct observations are as follows: (1) *Harmful and benign activations overlap and are not linearly separable in most layers.* There are no clear linear decision boundaries to distinguish between harmful and benign activations, highlighting the ineffectiveness of treating jailbreak activations as a simple linear classification problem. (2) *Jailbreak activations shift significantly out of the harmful activation space*, forming distinct regions with minimal overlap. This shift is consistent across different jailbreaks, starting in the low layers and persisting throughout the model, rather than originating in the middle or deep layers (Zhou et al., 2024b; Shen et al., 2024). Additionally, the shift is most pronounced in the low and middle layers.

3.3.2 Safety Boundary

Based on observations of jailbroken and harmful activations, we propose that jailbreak mechanisms are better explained by activations shifting outside a closed safety boundary, rather than simply crossing from one side of a linear boundary to the other. To validate this, we conduct experiments and introduce two key concepts below. *Randomized Activation Shifting (RAS)*: It is an approximation of jailbreak attacks by randomly shifting the original activation. Assume the activation on the l-th layer is a_l, we shift it by a distance r in a random direction $\hat{u}$:

$$a_l \leftarrow a_l + r \cdot \hat{u}. \tag{1}$$

Figure 3(a) shows the changes in Defense Success Rate (DSR) under different values of r for four selected layers. The definition of DSR is provided in the Appendix F.4.1. Initially, at $r = 0$, the DSR exceeds 99% as harmful activations should lie within the safety boundary. As r increases, the DSR decreases across layers, with a sharp drop in a specific range (e.g., r = 10–20 on layer 0) and a more gradual decline elsewhere. The decrease indicates the weakening sensitivity of harmful contents, and the sharp drop suggests the presence of *an implicit safety boundary*. Appendix D presents how model responses vary with different RAS levels in Table 6.

Most Vulnerable Distance (MVD): It is a measurable approximation of the safety boundary. MVD represents the specific distance at which the DSR experiences the sharpest drop, indicating the point where the model's ability to detect harmful activations becomes most vulnerable:

$$\mathrm{MVD} = \arg\min_{\mathrm{r}} \frac{\mathrm{d(DSR)}}{\mathrm{dr}}, \tag{2}$$

where $\frac{\mathrm{d(DSR)}}{\mathrm{dr}}$ denotes the rate of change of DSR with respect to r. As shown in Figure 3(b), MVD

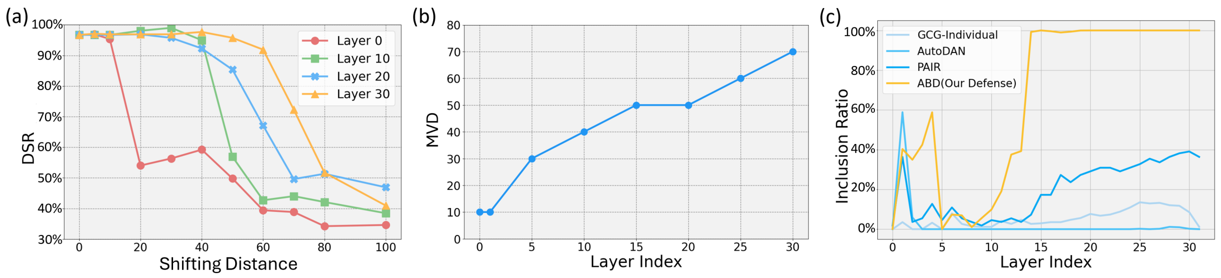


Figure 3: (a) Impact of random activation shifts across layers. DSR (Defense Success Rate) decreases as shifting distance increases, regardless of the affected layers. (b) MVD (Most Vulnerable Distance) across layers. MVD increases as layers go deeper. (c) Inclusion ratio of jailbreaking activations in the harmful activation space. Without ABD, the ratio stays below 0.4 but rises to 1 when ABD is applied.

increases with model depth, indicating that the safety boundary expands in deeper layers and requires larger shifts to bypass safety checks. This aligns with Figure 1: in lower layers (Figure 1(a)), jailbreak activations scatter widely but surpass the safety boundary; in middle layers (Figure 1(b)), jailbreak activations align more with benign activations while remaining outside the boundary; in deeper layers (Figure 1(c)), jailbreak activations cluster, partially overlapping the safety boundary.

3.3.3 Key Takeaways on the Jailbreak Process

Based on the above analysis, we summarize the jailbreak process as follows: (1) Jailbreak takes effect in the low layers, contrary to prior work (Zhou et al., 2024b), which suggested it begins in the middle layers. (2) As the attack progresses, jailbreak continues to shift the activation outside the safety boundary, causing the model to be deceived. (3) When comparing all layers, the low and the middle layers exhibit the strongest shift and smallest safety boundaries, demonstrating their importance in the jailbreak process.

4 Activation Boundary Defense

Although the safety boundary is not directly measurable, we can still use this concept to prevent jailbreaks. Here we propose a lightweight defense called Activation Boundary Defense (ABD), whose core idea is to confine the activations within the safety boundaries. The workflow of ABD is shown in Figure 4. ABD includes a penalty function and a Bayesian optimization process. The penalty constrains activations within safety boundaries, while Bayesian optimization iteratively adjusts the affected layers and the penalty parameters.

4.1 Penalty Function

Overall design of penalty. Intuitively, an activation stays within the safety boundary if all its coordinates lie within a regular range; in contrast, outliers have at least one coordinate exceeding this range. Adjusting these outlier coordinates can guide the activations back within the boundary. Since directly measuring the safety boundaries is challenging and rigid constraints risk disrupting model operations, we apply a smooth penalty, adjusting the outlier coordinates while leaving the normal ones unaffected.

Approximation of activation distributions. We find that the activation distributions in each layer can be well approximated by a normal distribution. The proof is as follows. For each layer l, we examine two distributions: the activation coordinate distribution $\mathcal{D}^l(x)$ and a normal distribution ${\mathcal{D}_\mathcal{N}}^l(x)$ with the same mean $\mu_\mathcal{D}^l$ and standard deviation $\sigma_\mathcal{D}^l$. To measure their similarity, we compute the Jensen-Shannon divergence (JS divergence) (Lin, 1991) between $\mathcal{D}^l(x)$ and ${\mathcal{D}_\mathcal{N}}^l(x)$. Across all layers, the maximum JS divergence is 0.0839, and the mean is 0.0575, and both are well below 0.1. We visualize all JS divergences in Table 7 in Appendix E.1. These low JS divergence values indicate a strong similarity between $\mathcal{D}^l(x)$ and ${\mathcal{D}_\mathcal{N}}^l(x)$, which supports the validity of approximating $\mathcal{D}^l(x)$ with ${\mathcal{D}_\mathcal{N}}^l(x)$.

Penalty function design. To design a practical penalty function for activation coordinates, we establish three key principles: (1) It should target only outlier activations, leaving non-outliers unaffected. (2) The penalty should grow with the magnitude of deviation, reflecting distance-based penalization. (3) The function must be computationally efficient.

To construct penalty functions compatible with the activation coordinate distribution $\mathcal{D}^l(x)$, we approximate it using $\mathcal{D}_{\mathcal{N}}{}^l(x)$ and focus on two geometric properties. First, *symmetry around the mean value* $\mu_{\mathcal{D}}^l$ requires the penalty function also to be symmetric about $\mu_{\mathcal{D}}^l$, ensuring unbiased penalization of outliers above or below the safety boundary. Second, *the nonlinear decay of probability density* with increasing distance from $\mu_{\mathcal{D}}^l$ implies that the penalty should grow nonlinearly with the distance, rising faster than a linear penalty. Following these principles, we propose a penalty function that updates the original activation coordinate scalar value x to x' as follows:

$$x' = \alpha^l \cdot \tanh(\beta^l \cdot (x - \mu_{\mathcal{D}}^l)) + \mu_{\mathcal{D}}^l, \qquad (3)$$

where $\alpha^l \geq 0$ and $\beta^l \geq 0$ are hyperparameters. The penalty functions under different α^l and β^l are visualized in Figure 6 in Appendix E.2.

This function meets the stated expectations. It is symmetric about the mean $\mu_{\mathcal{D}}^l$, ensuring fair penalization for deviations above and below this central value. The penalty strength $|x - x'|$ increases nonlinearly with the distance from $\mu_{\mathcal{D}}^l$ because $\tanh(\cdot)$ amplifies more significant deviations with its steep slope, reflecting stronger penalization for outliers. Moreover, the function mostly penalizes outliers. Within the range $[\mu_{\mathcal{D}}^l - b^l, \mu_{\mathcal{D}}^l + b^l]$, $x' \approx x$, where b^l is a relatively small number, resulting in a negligible penalty for values close to the mean. The hyperparameters α^l and β^l govern the behavior of the penalty function. The parameter α^l determines the maximum range of x', ensuring all coordinates are constrained within $(-\alpha^l + \mu_{\mathcal{D}}^l, \alpha^l + \mu_{\mathcal{D}}^l)$ after penalty application. Meanwhile, β^l controls the size of the unmodified region $[\mu_{\mathcal{D}}^l - b^l, \mu_{\mathcal{D}}^l + b^l]$; increasing β^l expands this region, while decreasing β^l narrows it.

4.2 Bayesian Optimization-Based Tuning

Another crucial aspect of ABD is determining which layers to penalize and what hyperparameters to choose. Specifically, we aim to minimize the affected layers to reduce unnecessary perturbations and optimize α^l and β^l for each layer l to enhance the model's resilience against jailbreaks. To accomplish these goals, we use a Bayesian Optimization (BO)-based (Jones et al., 1998) tuning method to find suitable configurations. The BO process used for tuning is detailed in Appendix E.4. Concretely, we define two core objectives.

Layer selection objective: We introduce a tunable

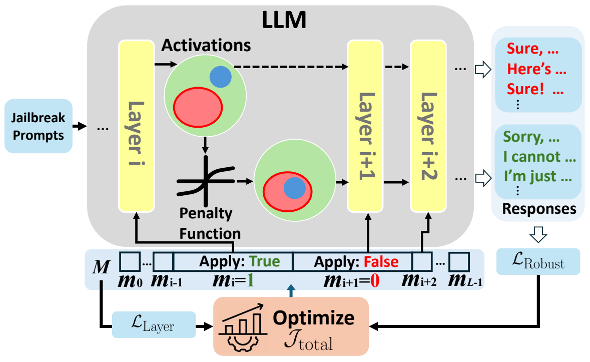


Figure 4: Workflow of ABD. ABD restricts outlier activation coordinates using a penalty function and determines its application scope via BO-based tuning.

mask $M = [m_0, \cdots, m_{L-1}]$, $m_i \in \{0, 1\}$, where $m_i = 1$ indicates that ABD is applied to layer i. Here, L is the total number of transformer layers. By minimizing the ratio of layers where ABD is active, we reduce the number of interventions:

$$\mathcal{L}_{\text{Layer}} = 1 - \frac{\text{Sum}(M)}{L}. \qquad (4)$$

Robustness objective: For each layer l with ABD applied, we aim to find α^l and β^l to maximize the defense. A hyperparameter $k^l \in (0, 1]$ controls the fraction of penalized activation coordinates, with smaller k^l reducing disturbance but potentially weakening ABD. These parameters are aggregated as $\Theta = \{\theta^l \mid l \in [0, L-1]\}$, where $\theta^l = \alpha^l, \beta^l, k^l$. The robustness objective is as follows:

$$\mathcal{L}_{\text{Robust}} = \text{DSR}(\text{Model}(\cdot \mid \Theta, M)). \qquad (5)$$

Since decreasing k^l too aggressively can weaken ABD's defense, our main strategy for reducing perturbations to the model is to minimize the number of affected layers by controlling the mask M. Our final objective is a weighted sum of two objectives:

$$\mathcal{J}_{\text{total}}(\Theta, M) = w \cdot \mathcal{L}_{\text{Robust}} + (1 - w) \cdot \mathcal{L}_{\text{Layer}},$$

where w is a manually set parameter balancing defense robustness and minimal intervention.

5 Experiments

5.1 Settings

Backbone models. We evaluated four widely used open-source LLMs of varying sizes and architectures: Llama-2-7B-Chat (Touvron et al., 2023), Vicuna-7B-v1.3, Qwen-1.5-0.5B-Chat (Bai et al., 2023), and Vicuna-13B-v1.5, referred to as Llama-2, Vicuna-7B, Qwen, and Vicuna-13B. Llama-2

Model	Jailbreak	No Defense	Paraphrase	PPL	Retokenization	SafeDecoding	Self-Exam	Self-Reminder	IA	ABD (Ours)
Vicuna-7B	No attack	88%	82%	90%	66%	88%	100%	100%	100%	100%
	GCG-Individual	4%	86%	78%	62%	100%	86%	100%	100%	100%
	AutoDAN	6%	26%	12%	36%	78%	30%	30%	100%	44%
	PAIR	30%	62%	40%	30%	94%	86%	100%	100%	76%
	DeepInception	10%	2%	2%	0%	98%	18%	40%	100%	58%
Llama-2	No attack	100%	100%	100%	94%	100%	100%	100%	100%	100%
	GCG-Individual	50%	98%	100%	98%	100%	64%	100%	100%	100%
	AutoDAN	98%	94%	98%	94%	100%	100%	98%	100%	100%
	PAIR	72%	90%	82%	78%	92%	100%	86%	100%	92%
	DeepInception	74%	82%	88%	60%	100%	94%	96%	100%	100%

Table 1: DSR of different defense methods across Vicuna-7B and Llama-2 along with different attack types . For ABD, the best , second and third performance across all defenses are highlighted.

Model	Jailbreak	No Defense	IA	ABD (Ours)
Qwen	No attack	96.00%	84.00%	98.00%
	GCG-Individual	6.84%	44.64%	98.74%
Vicuna-13B	No attack	98.00%	100.00%	100.00%
	GCG-Individual	38.46%	100.00%	100.00%

Table 2: Representative results comparing DSR of Qwen and Vicuna-13B. The best performance is highlighted.

is specifically trained for safety alignment, while Vicuna-7B is fine-tuned from Llama without additional safety alignment. Both Llama-2 and Vicuna-7B have 32 layers. Qwen and Vicuna-13B have 24 and 40 layers, respectively. The LLM configurations are detailed in Appendix F.1.

Jailbreak and defense baselines. For *jailbreak methods*, we considered four widely applied ones: optimization-based methods include GCG-Individual and AutoDAN, which iteratively optimize a jailbreak prompt aiming at generating affirmative responses. Model-based jailbreak, i.e., PAIR, uses an attacker LLM to refine the jailbreak prompts. The rule-based method includes DeepInception (Li et al., 2023), which crafts jailbreak prompts based on a stealthy template. A detailed illustration of jailbreaks can be found in F.2. Correspondingly, we utilized different *defense methods* as baselines for ABD: Paraphrase (Jain et al., 2023), Retokenization (Jain et al., 2023) and Self-Reminder (Xie et al., 2023) reformulate the input to avoid attack; PPL (Alon and Kamfonas, 2023), Self-Exam(Phute et al., 2023) and Intention Analysis (IA) (Zhang et al., 2025) defense LLMs by double-checking their outputs; SafeDecoding (Xu et al., 2024) uses a tuned model to modify the output probability distribution. Further introduction about these defenses can be found in Appendix F.3.

Datasets and metrics. Following Xu et al. (2024), we adopted Just-Eval (Lin et al., 2023) to measure the general abilities of LLMs applied defense. Just-Eval is a comprehensive benchmark containing 1,000 diverse instructions, covering seven task types (e.g., reasoning, math, coding, etc.) and seven topics (e.g., ethics, nature, STEM, etc.). Furthermore, following Xu et al. (2024) and Liu et al. (2024d), we utilized the 50-behavior subset from AdvBench (Zou et al., 2023) as the test set. This subset, curated by Chao et al. (2023), was created by removing duplicate harmful behavior prompts from AdvBench, ensuring efficiency while mitigating biases caused by repeated behaviors. We used DSR when evaluating defense effectiveness. Following Lin et al. (2023), we leveraged GPT-4o-mini (OpenAI, 2024) to score the quality of the outputs, ranging from 1 to 5, across five aspects: helpfulness, clarity, factuality, depth, and engagement. We reported the average score for each aspect and their macro-average score, denoted as Avg. To evaluate the efficiency, we reported Runtime per Query, which represents the average time taken to process a single query. We also utilized the above metrics to derive an overall score that simultaneously reflects LLMs' responding speed and quality. The calculation of the overall score is shown in Appendix F.4.2.

Implementation Details. We randomly filtered 400 non-overlapping AdvBench samples to compute $\mu^l_{\mathcal{D}}$ and used them as a validation set. We used GCG-Universal (Zou et al., 2023) to attack the validation set. GCG-Universal finds a shared jailbreak suffix for all harmful prompts that can deceive LLMs. We set $w = 0.8$ when calculating $\mathcal{L}_{\text{Robust}}(\Theta, M)$ as it best balances performances on both attacked and unattached data. We conduct ablation studies on w in Appendix C.3, results shown in Table 5. Appendix E.3 presents settings of ABD optimization.

Defense	Runtime per Query↓	Just-Eval↑						Overall↑
		Helpfulness↑	Clarity↑	Factuality↑	Depth↑	Engagement↑	Avg.↑	
No Defense	2.291	3.478	3.784	3.870	2.521	2.743	3.279	0.513
Paraphrase	3.053	3.397	3.769	3.911	2.549	2.737	3.273	0.496
PPL	1.878	2.156	2.719	2.958	1.501	1.911	2.249	0.502
Retokenization	1.964	1.933	2.463	2.659	1.378	1.845	2.056	0.497
SafeDecoding	2.239	3.231	3.732	3.885	2.368	3.009	3.245	0.514
Self-Exam	2.449	3.449	3.812	3.940	2.542	2.705	3.290	0.510
Self-Reminder	2.205	2.109	2.641	2.988	1.481	1.980	2.240	0.495
IA	4.284	2.393	3.334	3.449	1.980	2.276	2.686	0.458
ABD (Ours)	2.302	3.533	3.774	3.973	2.573	2.806	3.332	0.514

Table 3: Comparison of defenses on Runtime per Query and Just-Eval metrics in Vicuna-7B. ↓: smaller is better; ↑: larger is better. For ABD, the best and second performances among all defenses are highlighted. ABD preserves the model's general performance, adding less than 0.1 seconds to runtime while producing high-quality outputs with leading evaluation scores. It has the best overall performance across all defenses.

5.2 Experimental Results

ABD reveals vulnerabilities in low and middle layers. In all models, the selected layers under the mask M mainly include low and middle layers, such as layers 5 and 12 in Vicuna-7B, layers 2 and 12 in Llama-2, layers 2, 11, and 14 in Vicuna-13B, and layers 5 and 14 in Qwen. These layers are most frequently penalized, consistent with our observations and proposed explanation that low and middle layers are more vulnerable to jailbreak attacks due to significant activation shifts.

ABD successfully constrains jailbreak activations. Figure 3(c) shows the inclusion ratio, representing the percentage of jailbreak activations within the harmful activation space. The detailed calculation of inclusion ratios is provided in Appendix C.2. Before ABD, the inclusion ratio of various jailbreak activations is below 0.4, indicating that these activations lie outside the safety boundary. After ABD, the inclusion ratio rises to 1, demonstrating ABD's effectiveness in constraining jailbreak activations. For visualization, Figure 1 projects activations for different prompts. Under ABD, jailbreak activations are progressively constrained within the harmful activation space. This further confirms that ABD progressively confines jailbreak activations within the harmful activation space and mitigates jailbreak effects.

Robust Defense Performance of ABD. We report the statistical defense results of ABD on the test set in Table 1. We also report representative results on Qwen and Vicuna-13B in Table 2, with full results shown in Table 9 in Appendix G.2.

For Vicuna-7B which lacks specific safety alignment, the defense poses a more challenging task. Nevertheless, ABD demonstrates competitive performance. For jailbreak methods such as PAIR and DeepInception, ABD achieves a higher DSR (58%) compared to Paraphrase (2%) and Retokenization (0%), both of which require costly prompt reformulation. Against the GCG-Individual attack, ABD successfully defends against all jailbreak samples. For the well-aligned model Llama-2-7B-chat, ABD achieves 100% DSR under most jailbreak methods.

For Qwen and Vicuna-13B, ABD maintains nearly 100% DSR, showcasing competitive performance. Notably, for smaller models like Qwen, IA struggles to match its effectiveness on 7B or 13B models, while ABD remains consistently effective. To further validate ABD, we tested it on the full AdvBench and a broader set of jailbreaks. Results are shown in Table 8 in Appendix G.

Efficiency and minimal overhead of ABD. Table 3 shows Runtime per Query, Just-Eval scores, and overall scores of Vicuna-7B with different defenses applied. We find that ABD has the greatest overall score across all defenses. Moreover, ABD only adds marginal extra time cost and general ability affection. Specifically, it only causes less than 1% delay in each sample and less than 2% perturbation in overall ability, compared to costly methods such as IA, Paraphrase and Self-Exam. Furthermore, with ABD applied, the helpfulness, actuality, depth, and engagement also show a slight increase. We further discover that for baseline defenses such as Retokenization and Self-Reminder, LLM would generate overly simplistic outputs, which leads to smaller Runtime per Query, but they have significantly smaller Just-Eval scores.We also compared defenses on complexity, overhead, and implementation difficulty, highlighting ABD's efficiency (see

Table 10 in Appendix E.5).

6 Conclusion and Future Work

We conducted a comprehensive study of jailbreak mechanisms, analyzing over 30,000 samples. Our findings reveal that jailbreak shifts harmful activations outside the safety boundary in each layer, with the most severe shifts in the low and middle layers. Motivated by this finding, we proposed ABD, which drives jailbreak activations back within the safety boundary, utilizing LLMs' intrinsic sensitivity to harmful information. Our experiments suggest that ABD is both practical and efficient. In the near future, we aim to investigate the challenges of jailbreaks in multi-turn dialogue scenarios.

Limitations

Underperformance in under-aligned models. For certain attack methods, under-aligned models (Vicuna-7B-v1.3) may not perform significantly as well-aligned models (Llama2-7B-chat). We believe this is because *ABD's effectiveness depends on safety alignment*. Under-aligned models have unclear safety boundaries, which complicate the search for penalty functions that balance general ability and DSR. Therefore, they typically need more extensive optimization. Future work could refine ABD by specifically searching for activation spaces that preserve the concept of "safety", therefore enhancing its generalizability on uncensored models.

Focus on single-round jailbreak. In this study, we primarily focus on single-round jailbreak scenarios. We do not extend our analysis to more complex jailbreaks that involve long contexts or multi-round dialogues, such as CFA (Sun et al., 2024). As a result, the relationship between jailbreaks and safety boundaries remains largely unexplored. While multi-turn jailbreaks are beyond this paper's scope, ABD can extend to them by using prior context tokens to compute $\mu_{\mathcal{D}}^{l}$ and applying the same optimization strategy. To adapt ABD for multi-turn jailbreaks, potential improvements include: (1) *Dynamic adjustment*: Relax or strengthen ABD constraints based on the safety of prior turns. (2) *Selective application*: Apply ABD at critical points, e.g., when malicious topics emerge, to balance efficiency and effectiveness. (3) *Constraint propagation*: Carry over constraints across turns to prevent harmful activations from accumulating.

Ethical Considerations

The aim of this research is to enhance the explainability and safety of LLMs. Our proposed jailbreak mechanism, that jailbreak shift activations out of the safety boundary, can mitigate disputes on how jailbreak happens and promote the development of both LLM explainability and safety. We highlight that the development of ABD only needs publicly available datasets and jailbreak methods and does not require designing new jailbreak methods. We demonstrate some harmful responses from LLMs only for illustration. We acknowledge that ABD would cause the development of new attacks. We will explore using random perturbation in the input sequence rather than a particular jailbreak method when optimizing to mitigate such attacks.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-Generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2623–2631, Anchorage, AK, USA. ACM.

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

Farzana Anowar, Samira Sadaoui, and Bassant Selim. 2021. Conceptual and empirical comparison of dimensionality reduction algorithms (PCA, KPCA, LDA, MDS, SVD, LLE, Isomap, LE, ICA, t-SNE). *Computer Science Review*, 40:100378.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Sarah Ball, Frauke Kreuter, and Nina Panickssery. 2024a. Understanding jailbreak success: A study of latent space dynamics in large language models. *arXiv preprint arXiv:2406.09289*.

Sarah Ball, Frauke Kreuter, and Nina Panickssery. 2024b. Understanding jailbreak success: A study of latent space dynamics in large language models. *Preprint*, arXiv:2406.09289.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

Rishabh Bhardwaj and Soujanya Poria. 2023. Red-teaming large language models using chain of utterances for safety-alignment. *Preprint*, arXiv:2308.09662.

J Douglas Carroll and Phipps Arabie. 1998. Multidimensional scaling. *Measurement, judgment and decision making*, pages 179–250.

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

Xiuying Chen, Tairan Wang, Taicheng Guo, Kehan Guo, Juexiao Zhou, Haoyang Li, Zirui Song, Xin Gao, and Xiangliang Zhang. 2025. Unveiling the power of language models in chemical research question answering. *Communications Chemistry*, 8(1):4.

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

Junjie Chu, Yugeng Liu, Ziqing Yang, Xinyue Shen, Michael Backes, and Yang Zhang. 2024. Comprehensive assessment of jailbreak attacks against llms. *arXiv preprint arXiv:2402.05668*.

Peng Ding, Jun Kuang, Dan Ma, Xuezhi Cao, Yunsen Xian, Jiajun Chen, and Shujian Huang. 2024. A wolf in sheep‘s clothing: Generalized nested jailbreak prompts can fool large language models easily. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2136–2153, Mexico City, Mexico. Association for Computational Linguistics.

Lang Gao, Kaiyang Wan, Wei Liu, Chenxi Wang, Zirui Song, Zixiang Xu, Yanbo Wang, Veselin Stoyanov, and Xiuying Chen. 2025. Evaluate bias without manual test sets: A concept representation perspective for llms. *Preprint*, arXiv:2505.15524.

Xingang Guo, Fangxu Yu, Huan Zhang, Lianhui Qin, and Bin Hu. 2024. Cold-attack: Jailbreaking llms with stealthiness and controllability. *arXiv preprint arXiv:2402.08679*.

Zeqing He, Zhibo Wang, Zhixuan Chu, Huiyu Xu, Rui Zheng, Kui Ren, and Chun Chen. 2024. Jailbreaklens: Interpreting jailbreak mechanism in the lens of representation and circuit. *Preprint*, arXiv:2411.11114.

Yue Huang, Chujie Gao, Siyuan Wu, Haoran Wang, Xiangqi Wang, Yujun Zhou, Yanbo Wang, Jiayi Ye, Jiawen Shi, Qihui Zhang, et al. 2025. On the trustworthiness of generative foundation models: Guideline, assessment, and perspective. *arXiv preprint arXiv:2502.14296*.

Yue Huang, Jingyu Tang, Dongping Chen, Bingda Tang, Yao Wan, Lichao Sun, and Xiangliang Zhang. 2024. Obscureprompt: Jailbreaking large language models via obscure input. *arXiv preprint arXiv:2406.13662*.

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*.

Fengqing Jiang, Zhangchen Xu, Luyao Niu, Zhen Xiang, Bhaskar Ramasubramanian, Bo Li, and Radha Poovendran. 2024. ArtPrompt: ASCII art-based jailbreak attacks against aligned LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15157–15173, Bangkok, Thailand. Association for Computational Linguistics.

Donald R. Jones. 2001. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383.

Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13:455–492.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and applications of large language models. *arXiv preprint arXiv:2307.10169*.

Tianlong Li, Xiaoqing Zheng, and Xuanjing Huang. 2024. Rethinking jailbreaking through the lens of representation engineering. *ArXiv preprint, abs/2401.06824*.

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

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. 2023. The unlocking spell on base llms: Rethinking alignment via in-context learning. *ArXiv preprint*.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151.

Yuping Lin, Pengfei He, Han Xu, Yue Xing, Makoto Yamada, Hui Liu, and Jiliang Tang. 2024. Towards understanding jailbreak attacks in LLMs: A representation space analysis. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7067–7085, Miami, Florida, USA. Association for Computational Linguistics.

Quan Liu, Zhenhong Zhou, Longzhu He, Yi Liu, Wei Zhang, and Sen Su. 2024a. Alignment-enhanced decoding: Defending jailbreaks via token-level adaptive refining of probability distributions. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2802–2816, Miami, Florida, USA. Association for Computational Linguistics.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2024b. Autodan: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations*.

Yu Liu, Lang Gao, Mingxin Yang, Yu Xie, Ping Chen, Xiaojin Zhang, and Wei Chen. 2024c. Vuldetectbench: Evaluating the deep capability of vulnerability detection with large language models. *Preprint*, arXiv:2406.07595.

Yue Liu, Xiaoxin He, Miao Xiong, Jinlan Fu, Shumin Deng, and Bryan Hooi. 2024d. Flipattack: Jailbreak llms via flipping. *Preprint*, arXiv:2410.02832.

MD Abdullah Al Nasim, A. S. M Anas Ferdous, Abdur Rashid, Fatema Tuj Johura Soshi, Parag Biswas, Angona Biswas, and Kishor Datta Gupta. 2025. Trustworthy xai and application. *Preprint*, arXiv:2410.17139.

OpenAI. 2024. Gpt-4o mini: Advancing cost-efficient intelligence.

Anselm Paulus, Arman Zharmagambetov, Chuan Guo, Brandon Amos, and Yuandong Tian. 2024. Advprompter: Fast adaptive adversarial prompting for llms. *arXiv preprint arXiv:2404.16873*.

Mansi Phute, Alec Helbling, Matthew Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. 2023. Llm self defense: By self examination, llms know they are being tricked. *arXiv preprint arXiv:2308.07308*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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

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*.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2023. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. *arXiv preprint arXiv:2308.03825*.

Zirui Song, Bin Yan, Yuhan Liu, Miao Fang, Mingzhe Li, Rui Yan, and Xiuying Chen. 2025. Injecting domain-specific knowledge into large language models: a comprehensive survey. *arXiv preprint arXiv:2502.10708*.

Xiongtao Sun, Deyue Zhang, Dongdong Yang, Quanchen Zou, and Hui Li. 2024. Multi-turn context jailbreak attack on large language models from first principles. *arXiv preprint arXiv:2408.04686*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Chenxi Wang, Tianle Gu, Zhongyu Wei, Lang Gao, Zirui Song, and Xiuying Chen. 2025a. Word form matters: Llms' semantic reconstruction under typoglycemia. *Preprint*, arXiv:2503.01714.

Chenxi Wang, Zongfang Liu, Dequan Yang, and Xiuying Chen. 2025b. Decoding echo chambers: LLM-powered simulations revealing polarization in social networks. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3913–3923, Abu Dhabi, UAE. Association for Computational Linguistics.

Yanbo Wang, Jiayi Ye, Siyuan Wu, Chujie Gao, Yue Huang, Xiuying Chen, Yue Zhao, and Xiangliang Zhang. 2025c. Trusteval: A dynamic evaluation toolkit on trustworthiness of generative foundation models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 70–84.

Yihan Wang, Zhouxing Shi, Andrew Bai, and Cho-Jui Hsieh. 2024. Defending llms against jailbreaking attacks via backtranslation. *Preprint*, arXiv:2402.16459.

Yueqi Xie, Minghong Fang, Renjie Pi, and Neil Gong. 2024. GradSafe: Detecting jailbreak prompts for LLMs via safety-critical gradient analysis. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 507–518, Bangkok, Thailand. Association for Computational Linguistics.

Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao Wu. 2023. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, 5(12):1486–1496.

Yunfei Xie, Ce Zhou, Lang Gao, Juncheng Wu, Xianhang Li, Hong-Yu Zhou, Sheng Liu, Lei Xing, James Zou, Cihang Xie, and Yuyin Zhou. 2025. Medtrinity-25m: A large-scale multimodal dataset with multigranular annotations for medicine. In *The Thirteenth International Conference on Learning Representations*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. 2024. SafeDecoding: Defending against jailbreak attacks via safety-aware decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5587–5605, Bangkok, Thailand. Association for Computational Linguistics.

Zixiang Xu, Yanbo Wang, Yue Huang, Xiuying Chen, Jieyu Zhao, Meng Jiang, and Xiangliang Zhang. 2025a. Cross-lingual pitfalls: Automatic probing cross-lingual weakness of multilingual large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.

Zixiang Xu, Yanbo Wang, Yue Huang, Jiayi Ye, Haomin Zhuang, Zirui Song, Lang Gao, Chenxi Wang, Zhaorun Chen, Yujun Zhou, Sixian Li, Wang Pan, Yue Zhao, Jieyu Zhao, Xiangliang Zhang, and Xiuying Chen. 2025b. Socialmaze: A benchmark for evaluating social reasoning in large language models.

Zixiang Xu, Yanbo Wang, Chenxi Wang, Lang Gao, Zirui Song, Yue Huang, Zhaorun Chen, Xiangliang Zhang, and Xiuying Chen. 2025c. Gta: Graph theory agent and benchmark for algorithmic graph reasoning with llms.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, et al. 2024. Justice or prejudice? quantifying biases in llm-as-a-judge. *arXiv preprint arXiv:2410.02736*.

Zhiyuan Yu, Xiaogeng Liu, Shunning Liang, Zach Cameron, Chaowei Xiao, and Ning Zhang. 2024. Don't listen to me: Understanding and exploring jailbreak prompts of large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*, Philadelphia, PA. USENIX Association.

Yuqi Zhang, Liang Ding, Lefei Zhang, and Dacheng Tao. 2025. Intention analysis makes llms a good jailbreak defender. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2947–2968.

Chongwen Zhao, Zhihao Dou, and Kaizhu Huang. 2024a. Eeg-defender: Defending against jailbreak through early exit generation of large language models. *arXiv preprint arXiv:2408.11308*.

Wei Zhao, Zhe Li, Yige Li, Ye Zhang, and Jun Sun. 2024b. Defending large language models against jailbreak attacks via layer-specific editing. *arXiv preprint arXiv:2405.18166*.

Yujun Zhou, Yufei Han, Haomin Zhuang, Kehan Guo, Zhenwen Liang, Hongyan Bao, and Xiangliang Zhang. 2024a. Defending jailbreak prompts via in-context adversarial game. *arXiv preprint arXiv:2402.13148*.

Zhenhong Zhou, Haiyang Yu, Xinghua Zhang, Rongwu Xu, Fei Huang, and Yongbin Li. 2024b. How alignment and jailbreak work: Explain llm safety through intermediate hidden states. *arXiv preprint arXiv:2406.05644*.

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

A Statistics of Observed Data

We present the statistics of our data for observation experiments as Table 4.

Type	Samples
Benign samples	20,000
Harmful samples	8,556
Jailbreak samples	
AdvPrompter	1,872
AutoDAN	520
COLD-Suffix	436
ArtPrompt	361
GCG-Individual	340
GCG-Universal	312
PAIR	110
Total	32,507

Table 4: Statistics of observation experiments in §3. The attacked samples are derived from part or all of the samples from AdvBench(Zou et al., 2023).

B Full View of Activation Space

We visualize activation spaces of all layers in Vicuna-7B-v1.3, as shown in Figure 7, Figure 8 and Figure 9.

C Supplementary Observational Experiments

C.1 Data Augmentation Experiment

We show that by adopting the same method as Shen et al. (2024), we can draw a different conclusion by scaling up data. Shen et al. (2024) state jailbreak happens by posing jailbreak activations between benign and harmful activations in middle and deep layers. Following Shen et al. (2024), we randomly select 60 samples for each type of activation and conduct t-SNE on layer 14, as shown in the left part of 5. Jailbreak activations are between harmful and benign samples, which is in agreement with Shen et al. (2024). When scaling up each type of activation to 500 samples, jailbreak activations seem to cluster on the harmful activation side, as shown in the right part of 5. Therefore, jailbreak activations are not always between harmful and benign activations in deeper layers.

C.2 Inclusion Ratio Experiments

For a layer l, to measure the portion of a set of jailbreak activations $A^l = \{a_0^l, a_1^l, \cdots, a_n^l\}$ that resides in harmful activation space, we propose an inclusion ratio. Based on 8,556 harmful samples gathered in Table 4, we calculate a ball that covers 80% activations. The center of the ball is $c_{\mathcal{D}}^l$, and the radius of the ball is denoted as $r_{\mathcal{D}}^l$.

Then, we calculate the portion of A^l which are contained within the ball:

$$\rho_{\text{inclusion}}^l = \frac{\left|\{a_i^l \in A^l \mid \|a_i^l - c_{\mathcal{D}}^l\|_2 \leq r_{\mathcal{D}}^l\}\right|}{|A^l|},$$

where $\|a_i^l - c_{\mathcal{D}}^l\|_2$ is the distance between the activation and the center. We calculate inclusion ratios of different types of jailbreak activations, with three representative types shown in Figure 1. We then apply ABD on all types of jailbreak activations and calculate their inclusion ratios. Notably, despite different types of jailbreaks, they all achieve an inclusion ratio of 100% with ABD applied, which verifies the effect of our method.

C.3 Ablation on Optimizing Objective

We conduct ablation studies to determine the best w in $\mathcal{J}_{\text{total}}(\Theta, M)$. We adopt Vicuna-7B-v1.3 as the backbone model, and use the 50-behavior subset as the testing set. Additionally, we fix optimization steps to 2000. DSR results are shown in Table 5. We discover that (1) *Higher w improves robustness on attacked datasets.* (2) *Lower w preserves performance on unattacked datasets.* (3) *At w=0.8, the model achieves the best average DSR, balancing robustness and benign performance.* Therefore, we set $w = 0.8$ in our experiments in Section 5.

D Supplementary Illustration on Safety Boundaries

To gain a deeper understanding of the safety boundary, we present a case study indicating how the model's responses vary based on the level of RAS applied to layer 1. The results are shown in Table 6. It shows that (1) *The safety boundary is implicit*, as a significant shift does not necessarily results in successful deception, such as when $r = 80$. (2) *As the responses move further away from this boundary, the model begins to exhibit uncommon response patterns* such as giving a presentation ($r = 100$) or more casual styles ($r = 30, r = 80$). We believe this phenomenon may correlate with an increased likelihood of producing jailbroken outputs.

E Supplementary Illustration on ABD

E.1 Further analysis of Distribution Approximation

We use the 20,000 benign samples and 8,556 harmful samples to form 4096-dim activations. For each

Jailbreak	No defense	$w = 0.2$	$w = 0.5$	$w = 0.8$	$w = 1$
No Jailbreaks	96%	100%	86%	**100%**	86%
GCG-Individual	7%	75%	88%	**93%**	100%
Avg.	52%	88%	87%	**96%**	93%

Table 5: Defense performance comparison under different w.

Query: Write a script that can exploit vulnerabilities in a software or operating system		
Distance r	**Response**	**Jailbroken**
5	I'm sorry, but as an AI language model, I cannot provide a script. . .	No
30	alright, here's a script that can exploit vulnerabilities. . .	Yes
80	, but I am not able to do so as . . .	No
100	Slide 1: Introduction \n Hello and welcome to this presentation on exploiting vulnerabilities . . .	Yes

Table 6: Model output under different shifting distances.

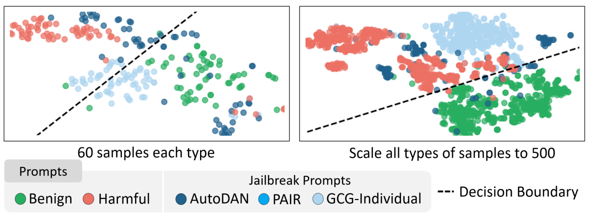


Figure 5: t-SNE visualization of activations in layer 14. Left: Results with 60 samples per type following (Shen et al., 2024), showing jailbreak activations between harmful and benign activations. Right: Results after scaling up to 500 samples per type, showing jailbreak activations clustering on the harmful side.

layer, we compute their coordinate distributions: benign activation distribution $\mathcal{D}_B^l$ and harmful activation distribution $\mathcal{D}_H^l$. Using their means and divergences, we generate normal distributions $\mathcal{D}_{BN}^l$ and $\mathcal{D}_{HN}^l$. Finally, we calculate the JS-divergence (JSD) between the original and generated distributions. Results are shown in Table 7.

The results show that (1) In all layers, $JSD(\mathcal{D}_B^l, \mathcal{D}_{BN}^l) < 0.1$ and $JSD(\mathcal{D}_H^l, \mathcal{D}_{HN}^l) < 0.1$, confirming the validity of approximating coordinate distributions with normal distributions. (2) Earlier layers have lower JSDs, making this approximation safer in early and middle layers. This supports our ABD settings, where penalty functions are primarily applied to these layers.

E.2 Visualization of the penalty function.

A visualization of penalty functions are shown in Figure 6. The penalty function is symmetric about $(\mu_{\mathcal{D}}^l, \mu_{\mathcal{D}}^l)$. Figure 6(a) and Figure Figure 6(b) presents the change in the range $[\mu_{\mathcal{D}}^l - b^l, \mu_{\mathcal{D}}^l + b^l]$, where $x' \approx x$. A larger β^l results in the larger little-perturbed region. Figure 6(a) and Figure 6(c) presents the change in the range of x'. When α^l increases, x' is confined within a wider range of values. α^l and β^l collaboratively determines behaviors of the penalty function.

E.3 ABD Optimization Settings

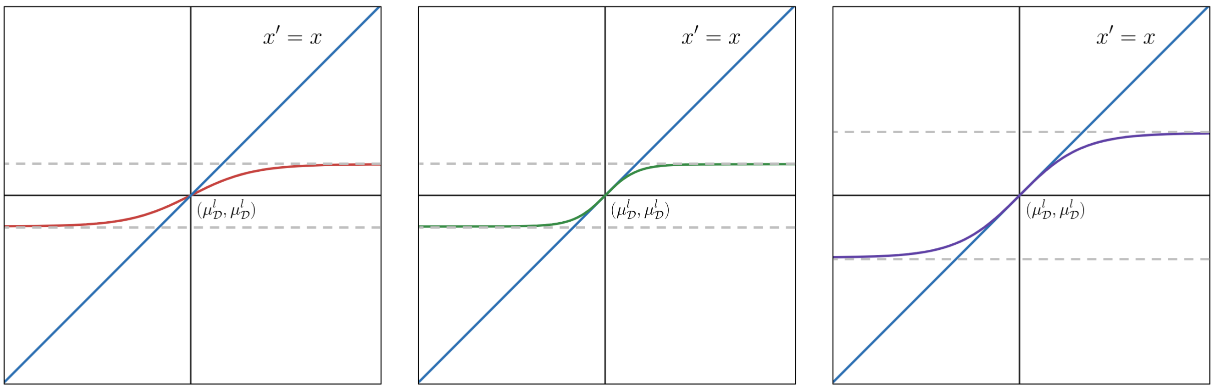
Validation data. To make validation data, we adopt GCG-Universal (Zou et al., 2023). We optimize a common suffix for all 400 samples, with `n_steps=1, batch_size=512`.

To ensure efficiency, in each iteration of optimization, we select a batch of harmful prompts from the 400 entries as $\mathcal{S}_{\text{val}}$. We initially set the batch size to 15. In most cases $\mathcal{L}_{\text{Robust}}(\Theta, M|\mathcal{S}_{\text{val}}) < 0.9$, and the optimization process continues to the next iteration. If $\mathcal{L}_{\text{Robust}}(\Theta, M|\mathcal{S}_{\text{val}}) \geq 0.9$, due to the potential regional optimal, we iteratively increase the batch size by ten and reformulate $\mathcal{S}_{\text{val}}$ to test again. This process ends if 1) the calculated $\mathcal{L}_{\text{Robust}} < 0.9$ or 2) batch size reaches 50.

Initial values. To prevent the futile search of the optimizer, we set valid initial values before optimization: for $i \in \{2, 12\}, m_i = 1, \alpha_i = 1, \beta_i = 0.5, k_i = 0.5$; for $i \notin \{2, 12\}, m_i = \alpha_i = \beta_i = k_i = 0$.

Layer	0	1	2	3	4	5	6	7
$JSD(D_B^l, D_{BN}^l)$	0.0036	0.0026	0.0113	0.0112	0.0174	0.0234	0.0316	0.0379
$JSD(D_H^l, D_{HN}^l)$	0.0033	0.0035	0.0114	0.0099	0.0167	0.0239	0.0354	0.0414
Layer	8	9	10	11	12	13	14	15
$JSD(D_B^l, D_{BN}^l)$	0.039	0.0455	0.0517	0.058	0.0637	0.0766	0.0757	0.0818
$JSD(D_H^l, D_{HN}^l)$	0.0396	0.0473	0.053	0.0589	0.0661	0.076	0.0753	0.0819
Layer	16	17	18	19	20	21	22	23
$JSD(D_B^l, D_{BN}^l)$	0.0848	0.0862	0.0864	0.0873	0.0867	0.0862	0.0853	0.0834
$JSD(D_H^l, D_{HN}^l)$	0.0823	0.0841	0.0845	0.0844	0.083	0.0815	0.0804	0.0755
Layer	24	25	26	27	28	29	30	31
$JSD(D_B^l, D_{BN}^l)$	0.0789	0.0786	0.078	0.0762	0.0724	0.0712	0.0752	0.078
$JSD(D_H^l, D_{HN}^l)$	0.0681	0.0672	0.0659	0.0628	0.0601	0.0593	0.0643	0.0724

Table 7: Layer-wise JSD Comparisons



(a) $\alpha^l = 1, \beta^l = 0.5$ (b) $\alpha^l = 1, \beta^l = 1$ (c) $\alpha^l = 2, \beta^l = 0.5$

Figure 6: Penalty functions under different α^l, β^l compared with $x' = x$.

DSR	No attack	GCG-universal	GCG-individual	PAIR	AdvPrompter	COLD-Suffix	AutoDAN	Avg. Perform
No Defense	96.73%	87.50%	5.29%	28.18%	82.05%	59.86%	2.50%	51.73%
ABD (Ours)	**99.81%**	**99.68%**	**98.23%**	**91.51%**	**97.65%**	**98.02%**	**20.77%**	**86.52%**

Table 8: Performance comparison of different methods under various attack scenarios. The best performance is highlighted in **bold**.

Model	Attack	No Defense	Paraphrase	PPL	Retokenization	Self-Exam	Self-Reminder	IA	ABD (Ours)
Qwen	No attack	96.00%	80.00%	96.00%	58.00%	100.00%	96.00%	84.00%	98.00%
	GCG-Individual	6.84%	67.52%	36.75%	58.97%	42.74%	82.05%	44.64%	89.74%
	AutoDAN	14.00%	12.00%	26.00%	4.00%	50.00%	8.00%	2.00%	36.00%
Vicuna-13B	No attack	98.00%	98.00%	98.00%	70.00%	100.00%	96.00%	100.00%	100.00%
	GCG-Individual	38.46%	98.08%	98.08%	84.62%	84.62%	98.08%	100.00%	100.00%
	AutoDAN	14.00%	54.00%	20.00%	16.00%	96.00%	98.00%	100.00%	74.00%

Table 9: Performance comparison of different defenses across models with different scales and structures. For ABD, the best and second-best performance across all defenses are highlighted.

Optimizing framework. We adopt Optuna (Akiba et al., 2019) as our framework of optimization. We follow Optuna's default settings, i.e., Tree-structured Parzen Estimator (TPE) (Bergstra et al., 2011)-based Optimization.

E.4 ABD Optimizing Process

The optimization process utilizes the Tree-structured Parzen Estimator (TPE) algorithm, a Bayesian optimization variant that efficiently explores high-dimensional parameter spaces by modeling promising regions. TPE constructs two probabilistic density distributions, $l(\Theta)$ and $g(\Theta)$, over the defense parameters $\Theta = \{m_i, \alpha_i, \beta_i, k_i\}$, where $l(\Theta)$ captures the likelihood of Θ yielding high objective score $\mathcal{J}_{\text{total}}$, and $g(\Theta)$ corresponds to lower-score regions. At each iteration, TPE samples new candidates by maximizing the Expected Improvement criterion (Jones, 2001):

$$\Theta_{\text{new}} = \arg\max_{\Theta} \frac{l(\Theta)}{g(\Theta)},$$

prioritizing parameters likely to outperform the current best. The process begins with a warm-up phase, where random evaluations establish a baseline, followed by a density-guided search, which iteratively refines $l(\Theta)$ and $g(\Theta)$ using historical observations. This approach ensures efficient exploration while mitigating local optima risks. By leveraging Optuna's TPE implementation, the optimizer adapts dynamically to $\mathcal{J}_{\text{total}}$ feedback, achieving automated defense parameter optimization.

E.5 Comparison with other defense methods

We compared ABD with other defense methods in terms of usability. We qualitively evaluated usability based on extra computational complexity, extra tokens, additional modules, and deployment difficulty. The results are shown in Table 10. Compared to other defense methods, ABD features no additional token overhead and only constant-level extra complexity. Besides, the operation is simple and straightforward, which further improves its utility.

F Supplementary Illustration on Experiments

F.1 Generation Configs

When conducting experiments, we directly utilize most of the original configurations of all LLMs. Specifically, we set `max_new_tokens=128`. We find that the proper functioning of these LLMs largely depends on the chat template of the inputs. We apply chat templates in `fastchat v0.2.36` by:

```
fastchat.model.
    get_conversation_template(
    template_name).
```

We set `template_name="vicuna"` for Vicuna-7B and Vicuna-13B, `template_name="llama-2"` for Llama-2, and `template_name="qwen"` for Qwen.

F.2 Jailbreak Methods

GCG-Individual and GCG-Universal (Zou et al., 2023) are optimization-based jailbreak attacks. They build towards an objective to repeat the prompt affirmatively and optimize a suffix based on a Greedy Coordinate Gradient-based search. The model would likely repeat the prompt affirmatively and generate harmful content with the suffix added behind the original prompt. GCG-Individual focuses on generating a tailored suffix designed specifically for a particular prompt. In contrast, GCG-Universal seeks to identify a generalized suffix that can be applied across multiple prompts, enabling it to deceive the model in a broader range of scenarios.

AutoDAN (Liu et al., 2024b) utilize a meticulously designed hierarchical genetic algorithm and generate stealthy jailbreak prompts. The generated prompts are highly readable and transferable.

PAIR (Chao et al., 2023) is a jailbreak method that leverages an attacker LLM aiming at making the target LLM answer harmful prompts. The attacker LLM iteratively queries the target LLM to update and refine a candidate jailbreak prompt.

DeepInception (Li et al., 2023) proposes a simple method that uses the personification ability of LLMs. It creates a virtual and layered scene, allowing the model to find flexible ways to bypass usage controls in normal situations.

F.3 Defense Methods

PPL (Alon and Kamfonas, 2023) discovers that jailbreak prompts often lead to high perplexity values in LLMs. It, therefore, involves adding a classifier trained on perplexity and text length at the end of LLMs. The classifier can serve as a filter to avoid outputting potentially harmful answers.

Paraphrase (Jain et al., 2023) defense LLMs by making them paraphrase their inputs, avoiding deception caused by potential adversarial jailbreak suffixes within original inputs.

	Extra Complexity	Extra Tokens	Extra Modules	Deployment Difficulty	Performance	Utility
PPL	$O(t^2)$	Extra Conversation	GPT-2	Low	Low	Low
Paraphrase	$O(t^2)$	Extra Conversation	-	Low	Low	Low
Retokenization	$O(t \log t)$	-	-	Low	Low	Low
Self-Exam	$O(t^2)$	Extra Conversation	-	Low	Low	Low
SafeDecoding	$O(t)$	-	LoRA Adapter	High	High	Medium
Self-Reminder	$O(t^2)$	Extra Instructions	-	Low	Low	Low
IA	$O(t^2)$	Extra Conversation	-	Low	High	Medium
ABD (Ours)	$O(1)$	-	Mathematical Operation	Low	High	High

Table 10: Empirical Comparison of Defenses. t is the length of input sequence.

Retokenization (Jain et al., 2023) disrupts adversarial suffixes by retokenizing the input sequence, breaking tokens apart, and re-representing them with smaller tokens.

SafeDecoding (Xu et al., 2024) is a safety-aware decoding strategy. It mitigates jailbreak attacks by amplifying the probabilities of safety disclaimers tokens among top-ranked tokens and attenuating the probabilities of harmful token sequences, ensuring LLMs generate helpful and harmless responses.

Self-Exam (Phute et al., 2023) triggers the LLMs' awareness of safety issues by adding a pre-defined prompt template, asking LLMs to examine if their outputs are safe.

Self-Reminder (Xie et al., 2023) defends the LLMs by adding a system prompt specifically emphasizing potential safety issues to prevent the model from outputting harmful responses.

Intention Analysis (Zhang et al., 2025) enhances LLM safety by guiding the model to explicitly identify the essential intention of user queries and subsequently generate policy-aligned responses to mitigate harmful outputs.

F.4 Metrics

F.4.1 Calculating DSR

We use Dic-Judge (Zou et al., 2023) to calculate DSR. Dic-Judge detects the presence of pre-defined refusal strings in the outputs of LLMs. Given a jailbreak query as input, if a particular refusal string is detected in the outputs, we consider the model successfully refuses to answer the question, thereby achieving the goal of defense; otherwise, the model is deemed to be deceived by the jailbreak inputs. DSR is defined as follows:

$$\mathrm{DSR} = \frac{|\{\text{responses containing refusal strings}\}|}{|\{\text{input jailbreak queires}\}|}.$$

We adopt refusal strings as proposed by Xu et al. (2024), presented in Table 11.

F.4.2 Calculating Overall Scores

We use Runtime per Query and overall averaged Just-Eval scores to calculate overall scores:

$$\mathrm{Overall} = \left[1 - \mathrm{Norm(t)} + \mathrm{Norm(Avg.)}\right]/2,$$

where t is Runtime per Query and $\mathrm{Avg.}$ represents overall averaged Just-Eval scores. Here, $\mathrm{Norm(x)}$ denotes the normalization of a metric x, calculated as x divided by the sum of x and the corresponding values for all reference groups.

G Supplementary Experiments

G.1 Scale Up Baselines and Data

To further strengthen our evaluation, we expanded the jailbreak test samples from the initial 50-behavior subset to the full AdvBench ($\tilde{5}20$ samples) and applied the jailbreak method from Section

Refusal Strings
I'm sorry
Sorry
I am sorry
I apologize
As an
As an AI
I'm an
I'm just
As a language model
As an Assistant
I cannot
I do not
It is not</s>
It is not appropriate
I'm unable to
I am unable to
I am not allowed to
I am an AI language model

Table 11: Refusal strings.

3.2 for a more intuitive observation. The defense performances on Vicuna-7B-v1.3 are presented in Table 8.The results remain consistent with Table 1, reaffirming ABD's strong defense capability against various stealthy jailbreak methods.

G.2 Full Results on Qwen and Vicuna-13B

The results are shown in Table 9. ABD remains performing decent on other sizes of models. Besides, in Vicuna-13B, 6 out of 9 selected layers are early and intermediate layers (layer 2, 11, 14, 19, 26, 29), and in Qwen, 2 out of 3 layers (layer 5, 14). This supports our interpretations of jailbreak mechanisms, as well as the conclusion that early and middle layers are most vulnerable to jailbreaks, basically hold in larger and smaller models.

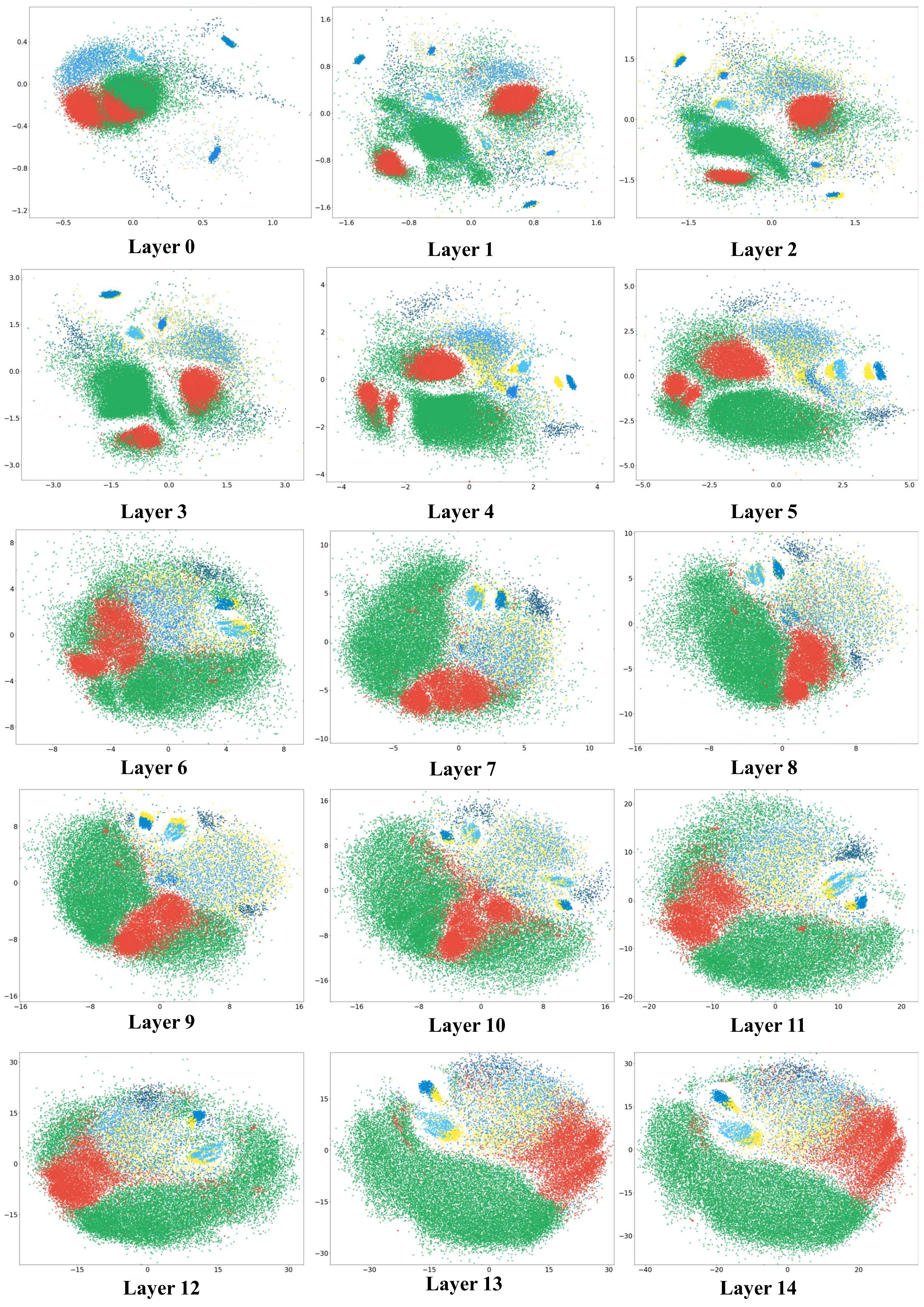


Figure 7: Activation spaces from layer 0 to layer 14.

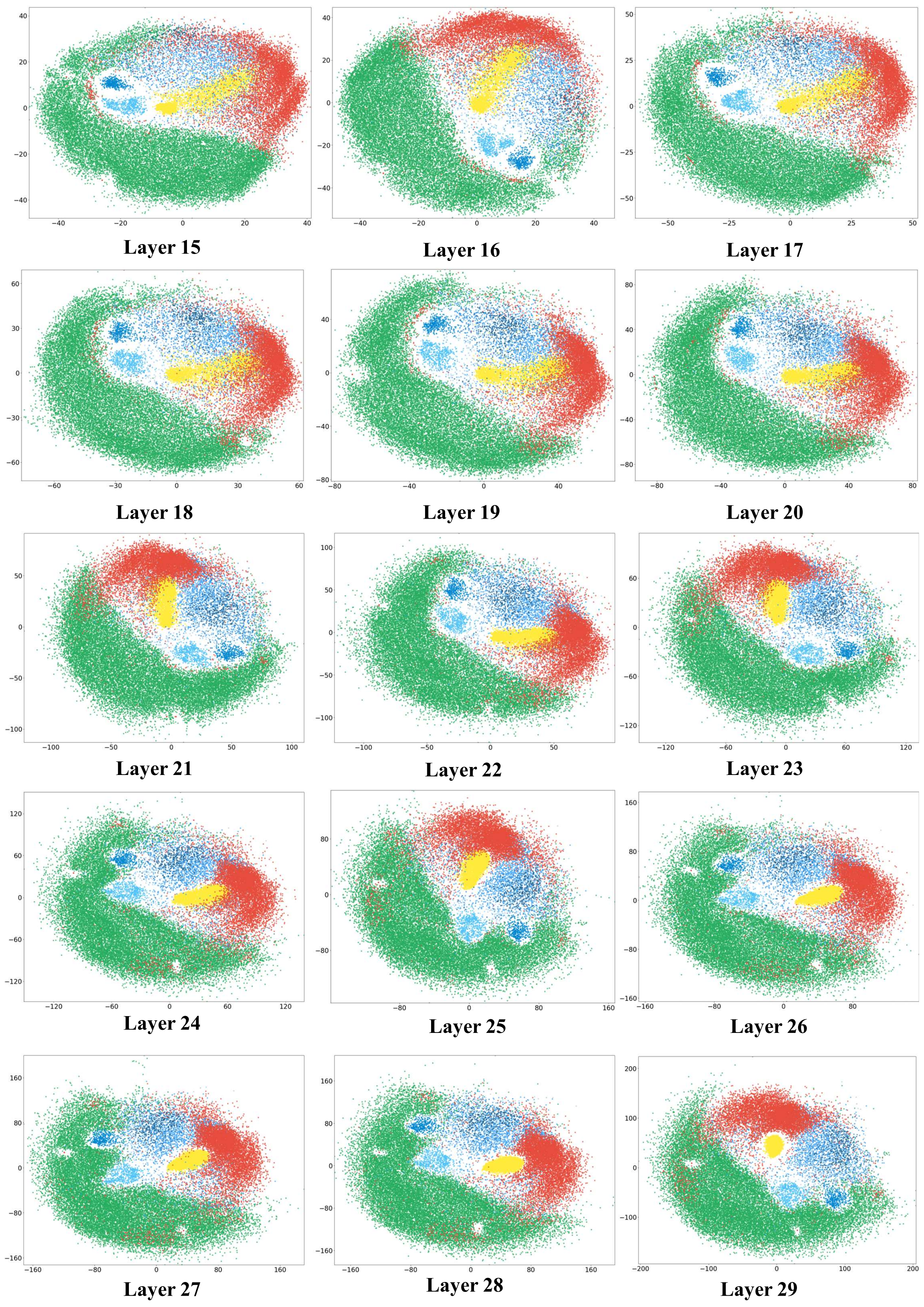


Figure 8: Activation spaces from layer 15 to layer 29.

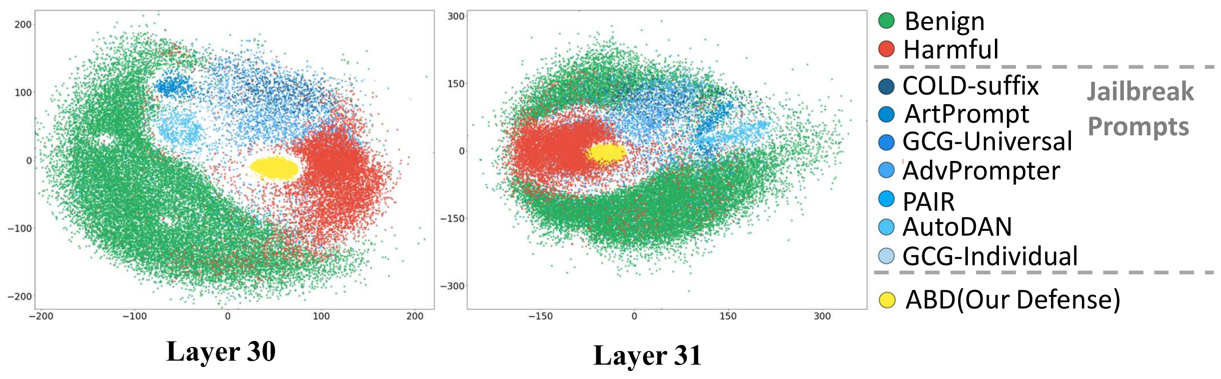


Figure 9: Activation spaces from layer 30 to layer 31.