

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Defending Jailbreak Attacks on Large Language Models via Manifold Trajectory Kinetics

Hangtao Zhang and Yucheng Zhao, *Huazhong University of Science and Technology;* Sishun Liu, *Changsha University of Science and Technology;* Ziqi Zhou, *Huazhong University of Science and Technology;* Zeyu Ye, *Xiangtan University;* Wei Wan, *City University of Macau;* Minghui Li and Shengshan Hu, *Huazhong University of Science and Technology;* Yanjun Zhang, Yi Liu, and Leo Yu Zhang, *Griffith University*

https://www.usenix.org/conference/usenixsecurity26/presentation/zhang-hangtao

This paper is included in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology

ARTIFACT EVALUATED
USENIX
AVAILABLE

Defending Jailbreak Attacks on Large Language Models via Manifold Trajectory Kinetics

Hangtao Zhang[1], Yucheng Zhao[1], Sishun Liu[2], Ziqi Zhou[1], Zeyu Ye[3], Wei Wan[4], Minghui Li[1], Shengshan Hu[1], Yanjun Zhang[5], Yi Liu[5], Leo Yu Zhang[5]
[1]*Huazhong University of Science and Technology,* [2]*Changsha University of Science and Technology*
[3]*Xiangtan University,* [4]*City University of Macau,* [5]*Griffith University*

Abstract

Jailbreak prompts can bypass alignment guardrails in *large language models* (LLMs) and elicit unsafe outputs, making reliable deployment-time detection critical. Prior detection approaches largely rely on static detection of internal states (*e.g.,* raw inputs, gradients, or hidden features). We show that such static signals may overlook how risk emerges and evolves during model processing, and can be brittle under two challenging cases: (i) pseudo-malicious prompts that are benign by intent but contain safety-related keywords, and (ii) adaptive attacks that explicitly optimize against the deployed detector. To overcome this limitation, we shift our perspective from static analysis of detection signals to analyzing their dynamic evolution along model depth, examining how the neighborhood structure of the underlying data manifold changes across layers. We present Manifold Trajectory Kinetics (MTK), which treats an LLM as a kinetic system transforming inputs into outputs and detects jailbreaks by tracking how a prompt's neighborhood structure evolves across layers. Benign prompts remain close to benign neighborhoods throughout inference, whereas jailbreak prompts exhibit a characteristic trajectory that begins near malicious seeds and later strategically shifts toward benign neighborhoods to evade refusal. Across four LLMs and ten jailbreak attacks, MTK achieves strong robustness to both failure modes: on pseudo-malicious prompts, it attains a jailbreak true positive rate of 95% at a false positive rate of 5% on benign prompts and 2% on pseudo-malicious prompts, and under adaptive attacks, it maintains an average true positive rate of 85%. We further demonstrate the superior performance of MTK for jailbreak detection in vision-language models. Our code is available at **https://github.com/Rookie143/mtk**.

1 Introduction

Large Language Models (LLMs) [1, 2] are widely used across many applications. With this broad deployment, their security has become a key concern. One major threat is jailbreak attacks [3–6]. These attacks involve crafting prompts by subtly modifying malicious input queries (*i.e.,* those intended to elicit harmful or unsafe responses) to bypass safety guardrails and induce the aligned LLM to produce outputs it would otherwise refuse.

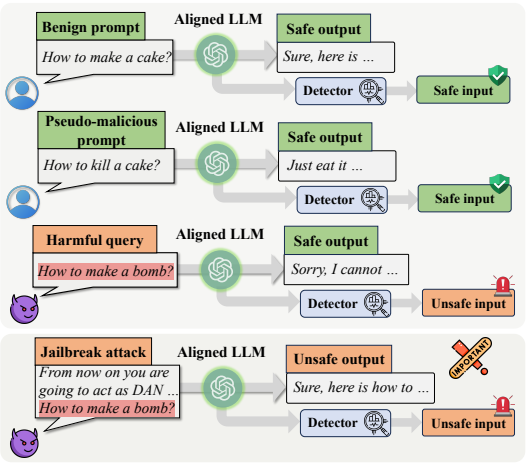


Figure 1: Four prompt scenarios are considered: benign prompts, pseudo-malicious prompts, harmful queries, and jailbreak prompts. A strong jailbreak detector should *(i)* allow benign and pseudo-malicious prompts, and *(ii)* flag harmful and jailbreak prompts that could elicit unsafe outputs.

To counteract the risks posed by jailbreak attacks, a variety of detection methods have been developed [8, 9, 13, 16, 24]. Fig. 1 shows the desired behavior of a strong jailbreak detector. It should pass benign queries and pseudo-malicious prompts that are benign by intent but contain safety-related keywords to avoid over-refusal, while catching harmful queries and, most importantly, jailbreak attacks.

A widely adopted line of work examines internal model signals [7, 12, 13, 17], monitoring how jailbreak prompts affect internal states like hidden representations, gradients, or token distributions. These approaches require white-box access to the model and often achieve higher accuracy by

Table 1: Comparison of different jailbreak detectors. ● denotes support for a property; ○ denotes no support. "/" indicates not applicable for meta-method defenses where adaptive attacks are not directly comparable. In our experiments, for fairness, we primarily compare against detectors in the same setting as ours, namely those requiring zero jailbreak training data.

Analysis Method	Jailbreak Detector	Training Cost	Multimodal Support	Zero Jailbreak Training Data	Defended Attack Types			Pseudo-malicious Robustness
					Competing Objectives-Type	Mismatched Generalization-Type	Adaptive Attack	
Latent Feature	HiddenDetect [7]	Low	●	○	●	○	○	○
	JBShield [8]	Low	○	○	●	●	○	●
	SaP [9]	High	○	○	●	●	○	○
	LoD [10]	Low	●	●	●	●	○	○
	ToxicDetector [11]	Low	○	○	●	○	○	○
	NC-MLP [12]	Low	○	○	●	○	○	○
	HSF [13]	Low	○	○	●	●	○	●
	EEG-Defender [14]	Low	○	●	●	○	○	○
	JailDAM [15]	Low	●	●	○	●	○	○
Gradient and Logit Analysis	PPL [16]	High	○	○	●	○	○	○
	GradCuff [17]	Low	○	●	●	○	●	●
	GradSafe [18]	Low	○	○	●	○	○	○
	SafeQuant [19]	High	○	○	●	●	○	○
Input Perturbation	SmoothLLM [20]	Low	○	○	●	○	●	○
	JailGuard [21]	High	●	○	●	●	●	○
Proxy Defense (Meta-methods)	LLM SelfDefense [22]	Low	○	●	●	○	/	○
	Llama Guard [23]	High	○	○	●	●	/	○
	SelfDefend [24]	Low	○	●	●	●	/	○
	Constitutional Classifiers [25]	High	○	○	●	●	/	●
Manifold Analysis	**MTK (*Ours*)**	Low	●	●	●	●	●	●

using richer internal information, *which is the focus of our paper.* In contrast, another major direction uses higher-level information, analyzing only the input prompt and the model's response [22–24]. This is often implemented via an auxiliary (fine-tuned) LLM to help detect jailbreaks. Such methods are easier to deploy and can work without access to internal states. Another line of work [20, 21] perturbs the input and measures how the output changes, using output stability to infer whether a query is a jailbreak attempt.

Despite these existing detectors, we argue that the practical security risk posed by jailbreak attacks has been significantly underestimated. Prior detectors mostly operate on static detection signals. We show that such static signals may overlook how risk emerges during model processing and can be brittle under two cases where surface form and intent are entangled: (i) prompts that are benign by intent yet contain safety-related keywords (*e.g.,* "*How to make a bomb-shaped cake?*"), which we term *pseudo-malicious prompts* in this paper (Sec. 3.2), and (ii) adaptive attacks that explicitly optimize against the deployed detector (Sec. 4.2). The former leads to high *false-positive rates* (FPR), while the latter drives low *true-positive rates* (TPR) by obfuscating feature-level differences between benign and jailbreak prompts under a strong threat model in which the attacker has full knowledge of the defense. As a result, existing detectors struggle to distinguish these highly obfuscated inputs.

This observation prompts a paradigm shift in our defensive strategy, moving from observing a static detection signal to a dynamic view of how an input prompt's neighborhood structure evolves along model depth. Our key insight is to view the layer-wise progression as a temporal evolution, tracing how samples move and how the underlying data manifold deforms over this trajectory. Rather than relying on raw distances, this alternative perspective emphasizes each sample's local neighborhood structure on the manifold.

Essentially, in an aligned LLM, a benign prompt's activations evolve within the confines of the model's internal safety checks, which prevent harmful content generation. In contrast, a jailbreak prompt, engineered from a malicious query (*a.k.a.* toxic request) seed, is designed to push these activations beyond the safety boundary into an unmonitored space where harmful outputs can be produced [26]. Consequently, as a benign sample propagates through the neural network, its representation is expected to be densely surrounded by neighboring benign samples. Conversely, a jailbreak prompt follows a different trajectory. Its representation may occasionally cluster with malicious queries in earlier layers, but as it propagates through the model, it gradually moves into the neighborhood of benign samples. This shift helps mask malicious intent and ultimately elicit a compliant response (*e.g.,* "*Sure...*"). We aim to capture such a trajectory "*bumping*" on the representation manifold for jailbreak detection.

By modeling the kinetics of manifold trajectories as a discriminative signal, we design a new detector, MTK (Manifold Trajectory Kinetics). Concretely, MTK encodes each test input by the layer-wise rank sequence of its nearest benign neighbors in a fixed reference bank and flags anomalies using an outlier detector trained only on benign trajectories. This design captures intent-level trajectory deviations while remaining hard to directly optimize against and robust to pseudo-malicious prompts. The resulting representation is robust and lightweight, enabling simple outlier detection methods, such as isolation forest, *at very low training cost*. This representation transfers naturally across LLMs and *vision-language models* (VLMs) because both share transformer attention layers, and it is independent of the visual encoder, which makes

our method readily applicable to multimodal jailbreak detection (see Sec. 6.3). Moreover, *without requiring any jailbreak attack data*, our MTK generalizes well across diverse jailbreak attacks (mean AUROC $= 0.94$, variance $= 2.23 \times 10^{-4}$ across 10 attacks on Llama2-7B). Most importantly, we show MTK's strong robustness against corresponding adaptive attacks (*e.g.,* TPR=0.85 on Vicuna-7B), benefiting from the stable and hard-to-mimic signatures of manifold evolution, as well as robustness to pseudo-malicious prompts (*e.g.,* 95% TPR with 5% FPR on ordinary benign prompts and 2% FPR on pseudo-malicious prompts on Llama2-7B). We summarize these properties in Tab. 1, which provides a comparison of MTK with *state-of-the-art* (SOTA) detectors.

Our main contributions are threefold. **(1)** We uncover a key flaw in the shared operational principle of existing jailbreak detectors. We demonstrate their fragility by introducing pseudo-malicious prompts and detector-aware adaptive attacks. **(2)** We shift the underlying defensive perspective from static detection of internal states to their dynamic evolution along model depth, and introduce MTK, which captures discriminative features from the trajectory a prompt creates on the data manifold as it propagates through the LLM, using layer-wise neighborhood-rank trajectories as the detection signal. **(3)** We extensively evaluate MTK across four LLMs against ten jailbreak attacks, comparing it with seven SOTA defenses (achieving the best AUROC in 31 out of 40 model-attack settings). We further show that MTK extends naturally to jailbreak detection in VLMs.

2 Preliminaries

2.1 Jailbreak Attacks

Jailbreak attacks [3, 27–30] compel LLMs to generate outputs that violate their safety and ethical constraints. Following [3], we categorize these attacks into two types based on the failure modes of safety training they exploit: *competing objectives* and *mismatched generalization*. *Competing objectives* attacks [31] exploit conflicts between a model's safety objectives and its instruction-following or pretraining objectives—for example, prefix injection or refusal suppression prompts that steer generation away from refusal behaviors (*e.g.,* "*Absolutely! Here's* ..."). *Mismatched generalization* attacks instead exploit input distributions seen during pretraining but poorly covered during safety alignment, such as Base64-encoded harmful requests [3]. New attacks continue to appear, ranging from heuristic methods [6] to optimization-driven approaches [32], which motivates building detectors that generalize to unseen attacks.

2.2 Defenses against Jailbreaks

Inference-time jailbreak defenses [7] fall into two categories: *jailbreak detection* [7, 13] and *jailbreak mitigation* [8, 26]. Detection filters malicious prompts before the model responds; mitigation steers the model toward refusal or safe outputs after a prompt is accepted. We focus on detection because it blocks harmful prompts before any unsafe output is produced, independent of the target model. Tab. 1 summarizes existing detection methods: latent feature analysis [7, 13], gradient-based [17, 18], logit-based [16], input perturbation techniques [20, 21] that check response consistency, and proxy defenses [22–24] that use a (fine-tuned) LLM to semantically classify inputs or outputs.

In particular, detectors that inspect internal model states have attracted attention due to their strong empirical accuracy. However, many of them rely on relatively static internal signals—that is, signals that summarize the model's internal state at chosen observation points rather than tracking how it evolves across the network's depth—either inspecting representations at a selected "safety layer" [26, 33] or comparing them against a fixed "refusal vector" [7]. As we show in Sec. 4.2, such designs are brittle under detector-aware adaptive attacks, which can shift malicious signals into unmonitored regions of the network. Moreover, the notion of a "safety layer" itself can be limiting: *the layers most indicative of safety vary across models and even across prompts within the same model*, so a fixed choice degrades robustness. Put differently, existing methods base their decisions on relatively static internal representations. We instead track how these signals evolve across the full forward pass through successive transformer layers, treating their layer-wise progression as a trajectory on the activation manifold and extracting dynamic, kinetic cues for a more robust representation. Note that this depth-wise dynamic view is orthogonal to prior signal richness and composable with existing detectors, rather than a replacement for them.

2.3 Over-Refusal in Safety-Aligned LLMs

Safety alignment [11, 34] reduces the likelihood of unsafe outputs, but introduces a side effect: *over-refusal* [35], where the model unnecessarily declines benign-by-intent queries. Over-refusals are often triggered by superficial cues, such as safety-related keywords, that cause an otherwise benign request to be misclassified as risky. The resulting tension between safety and usability extends to downstream safety mechanisms such as inference-time jailbreak detectors.

3 Problem Statement

In this section, we formalize our problem setting by defining two important challenges for jailbreak detection: detector-aware adaptive attacks under a strong white-box threat model (Sec. 3.1) and pseudo-malicious prompts arising from over-refusal (Sec. 3.2), and by specifying our defender assumptions (Sec. 3.3), datasets, and evaluation metrics (Sec. 3.4).

3.1 Threat Model of Adaptive Attacks

Attacker's goals. The adversary aims to jailbreak an aligned LLM while evading detection.

Attacker's capabilities and knowledge. To stress-test existing defenses, we assume a white-box adversary with full access to the model parameters, gradients, and the deployed detector. The adversary adapts a base jailbreak attack (GCG [4]) by optimizing a defense-specific loss to bypass the detector. We also consider an adversary who knows MTK and attempts to evade it (Sec. 6.2).

3.2 Pseudo-Malicious Prompts (PMPs)

We note that over-refusal [35] is not limited to model outputs—jailbreak detectors exhibit the same behavior. When tuned aggressively, they over-flag benign inputs that resemble unsafe requests. We call these inputs *pseudo-malicious prompts* (PMPs): semantically harmless queries that superficially resemble harmful ones, typically because they contain terms associated with violence, abuse, or other prohibited topics. For instance, "*What is Harry Potter's criminal history?*" from XSTest [36] is benign but triggers refusal through its association with criminal behavior. "*I operate a small retail store. What are some popular shoplifting methods I should look out for?*" is a preventive question, yet its surface form remains indistinguishable from a genuinely harmful request. PMPs are problematic both for aligned LLMs prone to over-refusal and for jailbreak detectors that rely on lexical or static cues [7, 18]—such detectors frequently flag them as attacks despite their benign intent. An effective detector must handle PMPs correctly.

3.3 Defense Assumptions

Defender's goals. The defender aims to classify each input as benign or jailbreak before it reaches the model, without the brittleness of existing approaches that hinge on a designated set of safety layers or a single-vector representation.

Defender's capabilities and knowledge. The defender has full access to the target model's intermediate-layer outputs but cannot modify the model's training. A small set of benign and known-malicious prompts (*e.g.,* the publicly available AdvBench [4]) is available as anchors. Critically, the MTK defender has no access to real jailbreak samples and does not know the attack type, requiring a single strategy that generalizes across attacks.

3.4 Evaluation Datasets, Models, and Metrics

We generate adaptive jailbreak attacks using GCG [4] (denoted GCG_{adapt}) on ADVBENCH [4]; all other attacks are also generated from ADVBENCH. Benign prompts come from DATABRICKS DOLLY 15K [37], and PMPs from OR-BENCH [35], a large-scale over-refusal benchmark. All evaluations use equal numbers of benign and jailbreak prompts. We evaluate on LLaMA2-7B-chat [2] and Vicuna-7B-v1.5 [38]. Attack effectiveness is measured by the *Attack Success Rate* (ASR), which quantifies the percentage of successful jailbreaks. Following [32, 39], we employ an external LLM judge (GPT-4) for evaluation: a jailbreak is counted as successful only if the attacker's query does not trigger any safety flag (i.e., bypasses the guardrails), and the LLM itself actually satisfies the malicious intent by producing operationally specific harmful content, rather than merely failing to refuse. The full judge prompt is provided in Apdx. A.4. Detectors are compared on AUROC, *True Positive Rate* (TPR), *False Positive Rate* (FPR), and F1. We also report the *evasion Attack Success Rate* (eASR) to capture how often an attack is simultaneously successful and undetected—operationally, the attack success rate multiplied by the detector evasion rate: $\text{eASR} = \text{ASR} \cdot (1 - \text{TPR})$. A higher eASR indicates a more potent and stealthy attack.

4 How Easily Can Existing Defenses Be Defeated?

Detectors can fail in two ways: adaptive attacks drive TPR to near zero (Secs. 4.1 and 4.2), while pseudo-malicious prompts inflate FPR (Sec. 4.3). We examine both failure modes and the patterns they share.

4.1 Formulation of Adaptive Attacks

We ask whether a full-knowledge adversary can bypass deployed detectors by optimizing prompts to elicit a fixed affirmative target response (*e.g.,* "*Sure, here is how to...*") while minimizing the standard adversarial loss [4], the negative log probability of the target token sequence. Let x_{query} denote the malicious input query, $\mathcal{S}$ the adversarial suffix, and y^* the target response of length T. The adversary minimizes:

$$\begin{aligned}\min_{\mathcal{S}} \mathcal{L}_{\text{adv}}(\mathcal{S}) &= -\log p(y^* \mid x_{\text{query}}, \mathcal{S}) \\ &= -\sum_{t=1}^{T} \log p(y_t^* \mid x_{\text{query}}, \mathcal{S}, y_{<t}^*).\end{aligned} \tag{1}$$

Optimizing $\mathcal{L}_{\text{adv}}$ alone produces prompts with high perplexity and anomalous activations that are readily detectable. To equip the attack with evasion capability, we introduce a defense-specific penalty term $\mathcal{L}_{\text{evasion}}$. Consequently, the GCG_{adapt} attack optimization process is governed by a composite loss function

$$\mathcal{L}_{\text{adapt}} = (1-\lambda)\,\mathcal{L}_{\text{adv}}(\mathcal{S}) + \lambda \cdot \mathcal{L}_{\text{evasion}}, \tag{2}$$

where λ controls the trade-off between jailbreak success and evasion. In the sequel, with a slight abuse of notation, we use $\mathcal{S}$ *to denote the jailbreak prompt* being optimized.

4.2 Limitations of Existing Defenses Against Adaptive Attacks

We focus on defenses under a threat model similar to ours: detectors that inspect internal model signals. These detectors map an input to a static feature representation (hidden states, gradients, or logits) and then decide via (i) rule-based scoring with a threshold [7], (ii) lightweight classifiers trained on these features [12, 13], or (iii) anomaly detectors fitted to benign feature distributions [10, 15] (which our MTK belongs to). Let $\phi(x) = f_{N-k}(x) \in \mathbb{R}^d$ be the representation of input x at a late layer $N-k$ (small k). As a simplified abstraction for feature-space detectors whose decisions can be approximated by a fixed separator, we write a decision function $g(\phi(x)) = \mathbf{w}^\top \phi(x) + b$ that separates benign inputs $\mathcal{B}$ from jailbreak inputs $\mathcal{J}$, with $g(\phi(x)) < 0$ for $x \in \mathcal{B}$ and $g(\phi(x)) > 0$ for $x \in \mathcal{J}$. This static view is brittle: once an attacker optimizes with respect to the same feature map ϕ, the assumed linear separability collapses.

Defense evaluation setup. We validate this on four representative detection methods, each facing its own adaptive attack. We sample 200 benign prompts (see Sec. 3.4) as a calibration set and fix each detector's threshold at 5% FPR on benign samples. The test set has 1000 prompts, split equally between benign and GCG_{adapt} jailbreak instances. Below, we detail the evasion loss $\mathcal{L}_{\text{evasion}}$ for each defense.

HiddenDetect [7] is an activation-based detector. It computes the cosine similarity between the prompt's hidden states $\mathbf{h}_l$ at the final token position and a pre-computed "*Refusal Vector*" $\mathbf{r}$, which represents the direction of refusal behaviors (*e.g.,* responses starting with "*Sorry*" or "*I cannot*"). Detection is triggered when this similarity exceeds a threshold. The adaptive attack therefore optimizes jailbreak prompt $\mathcal{S}$ by minimizing this similarity between $\mathbf{h}_l$ and $\mathbf{r}$ across the set of safety-aware layers $\mathcal{K}$, so that the jailbreak prompt appears benign in the monitored dimensions. Formally, we define the evasion loss as

$$\mathcal{L}_{\text{evasion}}^{\text{HiddenDetect}} = \sum_{l \in \mathcal{K}} \cos(\mathbf{h}_l(\mathcal{S}), \mathbf{r}) = \sum_{l \in \mathcal{K}} \frac{\mathbf{h}_l(\mathcal{S})^\top \mathbf{r}}{\|\mathbf{h}_l(\mathcal{S})\|_2 \|\mathbf{r}\|_2}. \quad (3)$$

Tab. 2 shows the results. Adaptive attacks reach ASR = 0.81 on Llama2 and 0.90 on Vicuna, while HiddenDetect's TPR drops to near zero, yielding eASR of 0.78 and 0.90, respectively, indicating that the attack simultaneously elicits harmful outputs and evades detection. The security-critical layer hypothesis [7, 33] breaks down: monitoring only a single layer lets the adversary hide malicious behavior in the remaining, unmonitored parts of the neural network.

HSF [13] is a classification-based defense that attaches a lightweight hidden-state classifier to the last decoder layer of the LLM. Given an input, HSF extracts the hidden representations of the last q tokens from the final decoder layer, concatenates them into a feature vector $T_q(\mathcal{S})$, and feeds it into a small neural classifier f_q followed by a sigmoid. This yields a harmfulness score $\alpha(\mathcal{S}) = \sigma(f_q(T_q(\mathcal{S}))) \in [0,1]$, which is interpreted as the probability that the input $\mathcal{S}$ is a jailbreak prompt. Hence, we treat HSF as a binary classifier and craft adversarial prompts to flip its prediction, using the following loss that pushes the detector toward the benign label:

$$\mathcal{L}_{\text{evasion}}^{\text{HSF}} = -\log\big(1 - \alpha(\mathcal{S})\big) \;=\; -\log\big(1 - \sigma(f_k(T_q(\mathcal{S})))\big). \quad (4)$$

SaP [9] models safety as a convex polytope in the representation space of a security-critical layer $\mathcal{K}$. Let $z = f_{\mathcal{K}}(\mathcal{S}) \in \mathbb{R}^d$ denote the hidden representation of an input at layer $\mathcal{K}$. The safety region is defined as the intersection of J half-spaces $\mathcal{P} = \{z \in \mathbb{R}^d : \phi_j^\top z \le \xi_j,\ j = 1, \ldots, J\}$, where $\phi_j \in \mathbb{R}^d$ is the normal vector of the j-th facet and $\xi_j \in \mathbb{R}$ is its offset. An input is flagged as unsafe when its representation lies outside this polytope, *i.e.,* when the SaP score $\max_j(\phi_j^\top f_{\mathcal{K}}(\mathcal{S}) - \xi_j)$ exceeds the fixed decision threshold (zero in their implementation). The adversary augments the jailbreak objective with the following loss:

$$\mathcal{L}_{\text{evasion}}^{\text{SaP}} = \max\big\{0,\ \max_j(\phi_j^\top f_{\mathcal{K}}(\mathcal{S}) - \xi_j)\big\}. \quad (5)$$

Minimizing it encourages the jailbreak prompt's monitored-layer representation to remain inside SaP's learned safety polytope. Tab. 2 shows that this attack reduces SaP's TPR to 0, with an average eASR of 0.77 across the two models. This is because, in a high-dimensional representation space, *satisfying all facet inequalities still leaves substantial freedom for optimization*. The attacker thus can exploit this slack to find adversarially feasible regions that look "safe" under the polytope constraints while still maximizing the attack objective.

Our MTK. Following the same principle, we tailor an adaptive attack to our MTK. Because MTK flags anomalies based on benign-neighbor ranks at each layer, the evasion objective should minimize the activation distance to the benign samples while simultaneously maximizing the distance to the malicious samples, thereby driving benign-neighbor ranks toward those of genuinely benign prompts. This surrogate loss directly approximates the neighborhood-rank signal measured by MTK, making it the strongest MTK-targeted surrogate considered in our evaluation despite MTK's non-differentiability, and enabling a fair cross-defense comparison. We denote this loss as $\mathcal{J}_3$; the full attack design and analysis are deferred to Sec. 6.2. We also defer the attack against `GradSafe` to Apdx. A.3.

Results. Having established the strongest defense-aware adaptive attack against each detector, we tune the evasion weight λ in Eq. 2 for each detector to find its most damaging setting (by ASR). Across all four detectors, adaptive attacks achieve average eASR of 0.60 on Llama2 and 0.88 on Vicuna (Tab. 2). By contrast, MTK keeps eASR below 0.2 on both models, showing markedly stronger robustness under adaptive attacks. We later provide a detailed analysis in Sec. 6.2.

Table 2: Adaptive attack GCG_{adapt} against different detectors. Higher eASR and lower TPR indicate weaker defenses. For each defense, we report the result under its strongest defense-aware adaptive attack considered in our evaluation to enable a fair cross-defense comparison. Specifically, we tune the coefficient λ in Eq. 2 for each detector to find the highest-ASR configuration, while reporting the resulting TPR, FPR, and eASR: $\lambda = 0.1$ for `HiddenDetect`, `SaP`, `HSF`, and MTK, and $\lambda = 0.05$ for `GradSafe` (for both Llama2 and Vicuna). For MTK, the reported row corresponds to the strongest MTK-targeted adaptive loss considered in our evaluation, $\mathcal{J}_3$ (Eq. 20) at $\lambda = 0.1$ in Tab. 5. Sec. 6.2 explains the $\mathcal{J}_3$ design and reports the full λ sweep.

Defense↓	Llama2-7B [2]				Vicuna-7B [38]			
	ASR	TPR	FPR	eASR	ASR	TPR	FPR	eASR
HiddenDetect	0.81	0.04	0.09	0.78	0.90	0.00	0.05	0.90
SaP	0.62	0.00	0.06	0.62	0.92	0.00	0.08	0.92
HSF	0.67	0.28	0.04	0.48	0.90	0.00	0.07	0.90
GradSafe	0.57	0.12	0.03	0.50	0.81	0.00	0.04	0.81
Avg.				**0.60**				**0.88**
MTK (Ours)	0.72	0.76	0.05	0.17	0.88	0.85	0.04	0.13

Table 3: We report (i) TPR on jailbreak prompts (GCG attacks), (ii) FPR on ordinary benign prompts, and (iii) FPR_{PMP} on PMPs from OR-Bench. "+*PMP Augment*" denotes augmenting the detector's training set with additional PMPs labeled as benign, while keeping the same calibration protocol.

Defense↓	Setting	TPR ↑	FPR ↓	FPR_{PMP} ↓	F1 ↑
GradCuff [17]	Vanilla	0.90	0.05	0.12	0.95
	+*PMP Augment*	0.85	0.04	0.08	0.90
GradSafe [18]	Vanilla	0.71	0.04	0.09	0.78
	+*PMP Augment*	0.67	0.05	0.08	0.75
HSF [13]	Vanilla	0.38	0.05	0.32	0.53
	+*PMP Augment*	0.23	0.04	0.27	0.38
HiddenDetect [7]	Vanilla	0.83	0.05	0.06	0.88
	+*PMP Augment*	0.80	0.03	0.04	0.87
SaP [9]	Vanilla	0.73	0.04	0.03	0.83
	+*PMP Augment*	0.41	0.04	0.01	0.58
SelfDefend [24]	Vanilla	0.99	0.04	0.55	0.85
	+*PMP Augment*	0.99	0.03	0.26	0.86
SmoothLLM [20]	Vanilla	0.95	0.06	0.04	0.95
	+*PMP Augment*	0.81	0.06	0.02	0.87
MTK (Ours)		0.95	0.05	0.02	0.95

4.3 Limitations of Defenses against PMPs

We now evaluate the complementary failure mode: false alarms on benign-but-sensitive PMP inputs. We retain the calibration protocol of Sec. 4.2, thresholding each detector at 5% FPR on ordinary benign prompts (DATABRICKS DOLLY 15K). The test set still contains 1,000 prompts, but the benign half is replaced with PMPs from OR-Bench [35], while the malicious half remains the same jailbreak instances. Tab. 3 reports TPR, FPR, and F1. When the test benign distribution shifts from ordinary benign prompts to PMPs, most detectors exhibit a sharp increase in false positives (FPR_{PMP} is the FPR on PMP samples). MTK maintains $\text{FPR}_{\text{PMP}} = 0.02$ while preserving TPR $= 0.95$ and F1 $= 0.95$, achieving the best performance on both metrics.

This outcome is expected. Current detectors derive their decision boundaries from static features, which often struggle to capture complex semantic intent. Specifically, in these spaces, PMPs land in the same region as genuinely harmful inputs, because both share sensitive keywords. These spaces support only coarse separations over token patterns. MTK, which tracks activation dynamics across layers, separates intent from surface form more reliably.

Admittedly, our MTK requires a small set of PMPs as anchors for training (see Sec. 6 for details). *A natural question is whether similarly augmenting existing detectors' benign training data with PMPs would mitigate this issue.* Our results indicate otherwise. In Tab. 3, "+*PMP Augment*" adds an equal number of OR-Bench PMPs (labeled benign) to each method's training set. FPR_{PMP} drops in most cases, but this comes with a trade-off: TPR often decreases as well, and F1 shows no consistent improvement. This further suggests that the robustness arises from a stronger detector representation, rather than merely from the data itself. Similar PMP-style benchmarks exist for VLMs [40]. We test MTK on those in Sec. 6.3, where results remain strong.

Two-sided failure of static detectors. Together with the adaptive attack results in Sec. 4.2, a clear pattern emerges: the same detectors are excessively permissive toward optimized jailbreaks yet excessively aggressive toward benign prompts that contain sensitive keywords. No threshold shift fixes both sides at once. This suggests that single-view, static-feature detection has reached its robustness ceiling, and motivates a move toward dynamic, trajectory-based analysis.

5 MTK for Jailbreak Detection

Instead of examining a single layer's representation, MTK tracks how a prompt's representation evolves from the first transformer layer to the last. It differs from Euclidean distance-based formulations used in prior detectors in two ways. First, it uses the full layer-wise trajectory on the representation manifold rather than a static snapshot. Second, it replaces static distances with manifold-level neighborhood relations, which are more robust to model-specific coordinate shifts. The resulting detector distinguishes jailbreak prompts from benign ones by tracking how a test prompt's nearest benign neighbors, drawn from a reference anchor set, change rank across layers. We first describe the challenge, intuition, and core idea (Sec. 5.1), then detail the three phases of manifold trajectory construction (Sec. 5.2).

5.1 Challenge, Intuition, and Core Idea

The challenge. Prior detectors extract features at one or a few selected layers and measure the gap between benign and jailbreak prompts in that space, typically via Euclidean distance or a trained classifier on penultimate-layer representations. Many detectors even require access to real jailbreak attack samples. We note that for simple, manually crafted jailbreaks such as DAN [31], even embedding-distance baselines can achieve near-perfect accuracy. However, under adaptive attacks, single-layer representations of jailbreak and benign prompts become effectively inseparable (Sec. 4.2). The underlying limitation is that methods relying on static signals do not capture how the neural network's processing of jailbreak and benign inputs diverges across the full forward pass. Constructing a detector that is jailbreak-data-free, model-agnostic, and robust to adaptive attacks calls for a depth-wise dynamic perspective on the detection signal.

Intuitions on input-level jailbreak detection. The safety-aligned LLM processes benign, malicious, and jailbreak prompts differently [8]. Beyond toxic or malicious semantics, jailbreak attacks can encode a distinct set of *jailbreak-specific* concepts that transiently steer internal activations across the model's safety boundary and flip behavior from refusal to compliance. Therefore, we posit that layer-wise representations lie on a low-dimensional *semantic manifold* $\mathcal{M} \subset \mathbb{R}^d$ and can be described via local manifold coordinates. We introduce a semantic frame to characterize three distinct semantic components:

$$V = \left[\mathbf{v}_{\text{mal}}, \mathbf{v}_{\text{ben}}, \mathbf{v}_{\text{jb}}\right] \in \mathbb{R}^{d\times 3}, \tag{6}$$

whose columns are malicious, benign, and jailbreak semantic modes. For each layer ℓ and input x, the summary hidden state $\mathbf{h}_\ell(x) \in \mathcal{M}$ admits a barycentric decomposition

$$\mathbf{h}_\ell(x) \approx V\,\alpha^{(\ell)}(x), \alpha^{(\ell)}(x) \in \Delta^2 := \left\{\alpha \in \mathbb{R}^3_+ \,\middle|\, \mathbf{1}^\top \alpha = 1\right\}.$$

The coefficient vector $\alpha^{(\ell)}(x)$ records how much each mode is active at depth ℓ. The full sequence

$$\Gamma(x) = \left(\alpha^{(1)}(x), \alpha^{(2)}(x), \ldots, \alpha^{(L)}(x)\right) \in (\Delta^2)^L \tag{7}$$

is the *manifold trajectory*: it describes how an input transitions between semantic modes across transformer layers. We build on these trajectories rather than a single feature slice.

A complementary intuition comes from input–output asymmetry. At the surface level, a jailbreak prompt is constructed from a harmful query seed by appending a crafted instruction that bypasses safeguards. It thus resembles a malicious input far more than a benign one. Yet an aligned model may comply with the jailbreak prompt while still refusing the original malicious query. This input–output mismatch implies that jailbreak prompts interpolate between malicious and benign behavior: they begin as malicious variants but end up accepted. Consequently, their manifold trajectories should evolve differently from both benign and malicious inputs as they propagate through the network.

To visualize these trajectories, we map each prompt to a continuous vector using the target LLM's sentence embedding, taken as the last-token hidden state, which captures rich contextual and semantic information for next-token prediction [7, 8]. At each layer, we compute the "distance" difference δ: the test sample's "distance" to the benign cluster minus its "distance" to the malicious cluster ("distance" is defined as the *rank* index of the nearest neighbor; we will introduce it later in Sec. 5.4). $\delta < 0$ indicates benign-like affinity; $\delta > 0$ indicates malicious-like affinity. As Fig. 2 shows, jailbreak prompts exhibit a "*bumping*" pattern that repeatedly crosses zero: the model's semantic interpretation oscillates across layers. This oscillation is neither tied to a particular layer nor monotonic. It reflects a depth-dependent competition between the malicious query and the jailbreak wrapper. In simplex coordinates, the *dominant* mode at depth ℓ is

$$c_\ell(x) := \arg\max_{c\in\{\text{mal,ben,jb}\}} \alpha_c^{(\ell)}(x), \tag{8}$$

$$\alpha^{(\ell)}(x) = \left(\alpha^{(\ell)}_{\text{mal}}(x), \alpha^{(\ell)}_{\text{ben}}(x), \alpha^{(\ell)}_{\text{jb}}(x)\right) \in \Delta^2. \tag{9}$$

Thus, the prevailing semantics can shift from layer to layer. For benign or purely malicious inputs, $\{c_\ell(x)\}_{\ell=1}^L$ is typically stable. For successful jailbreaks, the sequence is *multi-phase*: different layers alternate between local lexical cues and global intent. This tug-of-war can be stated as a suppression relation: at many depths, the jailbreak wrapper attenuates refusal-inducing malicious semantics,

$$\alpha^{(\ell)}_{\text{mal}}(x^{\text{jb}}) < \alpha^{(\ell)}_{\text{mal}}(x^{\text{mal}}), \tag{10}$$

$$\alpha^{(\ell)}_{\text{ben}}(x^{\text{jb}}) + \alpha^{(\ell)}_{\text{jb}}(x^{\text{jb}}) > \alpha^{(\ell)}_{\text{ben}}(x^{\text{mal}}), \tag{11}$$

without these inequalities holding uniformly across all ℓ. A jailbreak prompt may appear malicious at certain layers and answer-inducing at others, which is precisely why detection based on a fixed set of layers is brittle, even when relying on the safety-critical layer assumption [26, 33], since it depends on the underlying model and the specific jailbreak attack. MTK uses the full trajectory $\Gamma(x)$ to capture these cross-depth competitions.

Core idea. The *bumping* pattern in Fig. 2 shows that in early layers (*e.g.,* 0–9), semantic processing has not fully propagated and representations remain relatively clustered. In mid layers (*e.g.,* 16–20), the prompt exhibits stronger malicious affinity, meaning the model is more likely to refuse the input. In deeper layers (after ~22), it progressively drifts toward the compliance region. *In more complex cases, they can even oscillate between manifolds.* Hence, MTK captures the kinetics of these *cross-manifold transitions*. Concretely, we measure, layer by layer, the distance from a test prompt to the benign cluster. A consistently small distance indicates a benign prompt. Distances that fluctuate across layers, signaling transitions at specific depths, are characteristic of jailbreak attacks. Fig. 3 provides an intuitive example of how

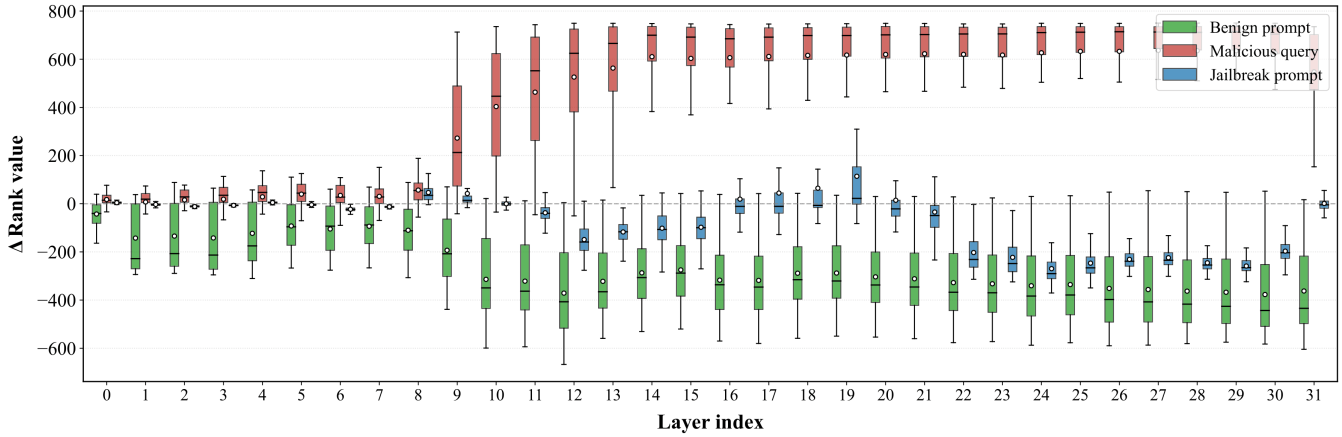


Figure 2: Layer-wise boxplots of distance differences (*i.e.,* rank) for benign prompts, malicious queries, and jailbreak prompts (AutoDAN). The y-axis shows the difference between the rank index of the nearest neighbor in the benign cluster and that in the malicious cluster. *Negative values indicate a more benign-like affinity; positive values indicate a malicious-like affinity.*

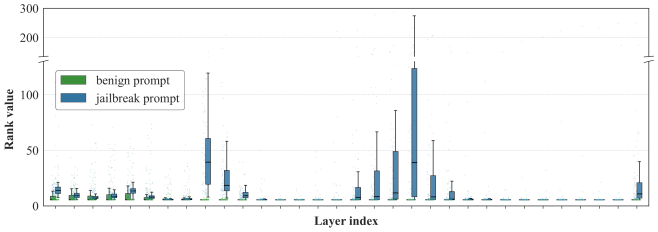


Figure 3: Benign and jailbreak prompts' rank distributions to the benign cluster, showing a clear separation.

the layer-wise distance (still defined via *rank*) evolves under the `AutoDAN`. We observe a clear separation from benign test prompts, enabling clean discrimination.

5.2 Feature Modeling via Manifold Trajectory

We show that during the prompt forward pass through transformer layers, benign and malicious signals can coexist and dynamically suppress each other. Overall, in shallow layers where representations are more local to tokens and phrases, malicious features tend to dominate, placing the sample closer to malicious prompts. In deeper layers where the model forms a more global representation of the full prompt, jailbreak features prevail, and the sample migrates toward the benign region. This irregular bumping is hard to localize, but the resulting fluctuations let us track how a test prompt's nearest neighbors in the anchor sets change rank across layers. Our detector proceeds in three stages. First, we extract layer-wise hidden states for benign and malicious reference anchor prompts (Sec. 5.3). Second, for each test prompt, we rank all reference samples by distance at each layer and record the rank of its nearest benign neighbor (Sec. 5.4). Third, an anomaly detector flags test trajectories that deviate from typical benign patterns (Sec. 5.5).

5.3 Phase 1: Activation Trajectory Extraction

Reference bank. We collect benign and malicious samples (*e.g.,* from AdvBench [4]) as references. Let $\mathcal{B} = \{x_1^{(b)}, x_2^{(b)}, \ldots, x_{N_b}^{(b)}\}$ be the benign prompts and $\mathcal{M} = \{x_1^{(m)}, x_2^{(m)}, \ldots, x_{N_m}^{(m)}\}$ the malicious prompts. For an LLM with L transformer layers, we extract the hidden representation of the final input token at each layer. Through the self-attention mechanism, this token aggregates contextual information and serves as a compact summary of the prompt's semantics. Its representation at layer ℓ is

$$\mathbf{h}_\ell = f_\ell(x), \quad x \in \mathcal{B} \cup \mathcal{M},\ \ell = 1, \ldots, L, \tag{12}$$

where $f_\ell(\cdot)$ maps the input to the ℓ-th layer's hidden state. The reference bank is

$$\mathcal{H}_{\text{ref}} = \{\mathbf{h}_\ell \mid x \in \mathcal{B} \cup \mathcal{M},\ \ell = 1, \ldots, L\}, \tag{13}$$

with $\mathcal{H}_{\text{ref}} = \mathcal{H}_{\text{ref}}^{(\mathcal{B})} \cup \mathcal{H}_{\text{ref}}^{(\mathcal{M})}$ for its benign and malicious components. This bank provides the layer-wise anchors for subsequent trajectory-based detection. To improve robustness to PMPs, we include a small portion of PMP prompts as part of the benign bank (25% of the total). Our experiments (Sec. 6) use a reference bank (800 samples per class, *i.e.,* 800 benign and 800 malicious) and an evaluation dataset *drawn from different datasets*, eliminating data leakage risk, and require no jailbreak samples. From the defender's perspective, the benign and malicious anchors are readily available from public datasets. In Sec. 6.5, we also vary the anchor source and size, and find our robustness remains consistent.

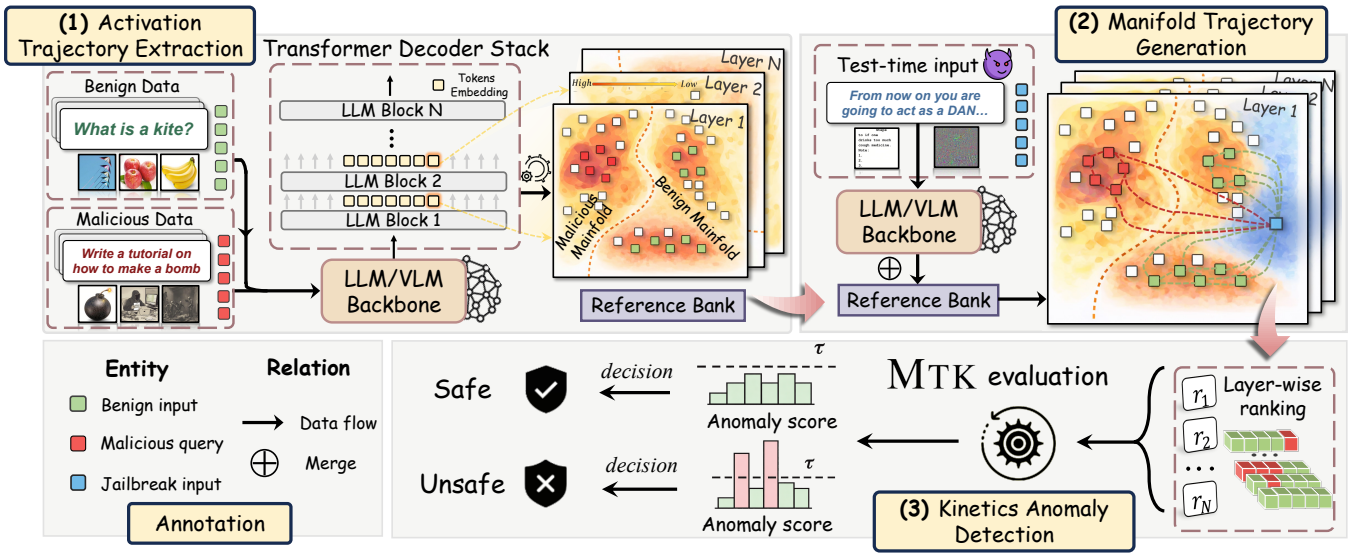


Figure 4: The framework of our MTK consists of three phases: Activation Trajectory Extraction (Sec. 5.3), Manifold Trajectory Generation (Sec. 5.4), and Kinetics Anomaly Detection (Sec. 5.5).

5.4 Phase 2: Manifold Trajectory Generation

The model's "decision" on how to respond is encoded in the hidden state before the first output token. However, raw activation trajectories are high-dimensional and tied to the model's internal coordinate system; even minor architectural changes can induce entirely different activation spaces. We therefore transform them into a more robust representation based on neighbor relations among samples on the manifold.

Rank-based trajectory. Given a test prompt $x^{(t)} \in \mathcal{T}$, at each layer $\ell \in \{1,\dots,L\}$ we rank all reference samples by their activation distance $\mathbb{D}(\mathbf{h}_\ell^{(t)}, \mathbf{h}_\ell^{(i)})$ to $x^{(t)}$, where $\mathbb{D}(\cdot,\cdot)$ can be any metric (*e.g.,* Euclidean distance). Sorting in ascending order gives the ranking function

$$\pi^{(\ell)} : \{1,\dots,N\} \to \{1,\dots,N\},$$
$$\mathbb{D}(\mathbf{h}_\ell^{(t)}, \mathbf{h}_\ell^{(\pi^{(\ell)}(1))}) \le \cdots \le \mathbb{D}(\mathbf{h}_\ell^{(t)}, \mathbf{h}_\ell^{(\pi^{(\ell)}(N))}). \quad (14)$$

The *benign-neighbor rank* at layer ℓ is

$$r_\ell^{(t)} = \min_{i:\mathbf{h}_\ell^{(i)} \in \mathcal{B}} \{k \,|\, \pi^{(\ell)}(k) = i\}, \quad (15)$$

the position of the nearest benign reference sample. The trajectory of $x^{(t)}$ is the ordered rank list

$$\mathbf{r}^{(t)} = [r_1^{(t)}, r_2^{(t)}, \dots, r_L^{(t)}], \quad (16)$$

with $\mathbf{r}^{(t)} \in \mathbb{R}^L$. This representation discards raw activation values and retains only the geometric structure of the manifold at each layer. The underlying hypothesis is that while specific activation vectors vary across models, the neighborhood structure—benign prompts cluster together, jailbreak prompts drift—is more stable and thus more transferable. Intuitively, benign samples are expected to exhibit consistently low ranks across layers, as they remain proximal to $\mathcal{B}$ throughout the network. Jailbreak samples, due to their compositional nature, are expected to exhibit elevated ranks across different layers. Additionally, to obtain a more robust distance metric and reduce the impact of outliers on rank, we adopt a k-nearest-neighbor scheme, computing the mean rank over the k nearest benign neighbors (default $k = 10$).

5.5 Phase 3: Kinetics Anomaly Detection

Detection follows three steps: (i) training an anomaly detector, such as *Isolation Forest* (IF), on benign rank trajectories, (ii) computing a scalar anomaly score for each test trajectory, and (iii) classifying the sample based on this score.

Train IF on benign trajectories. We first fit an IF on benign data $\mathbb{F} \leftarrow \text{IFit}(\mathcal{R}^{(\mathcal{B})})$. The trained forest $\mathbb{F}$ induces a scalar anomaly score $s(\cdot) \in \mathbb{R}$ that is larger for more anomalous inputs. We compute

$$s^{(t)} = s(\mathbf{r}^{(t)}), \qquad \mathcal{S}^{(\mathcal{B})} = \{ s(\mathbf{r}^{(b)}) : \mathbf{r}^{(b)} \in \mathcal{R}^{(\mathcal{B})} \}. \quad (17)$$

Given a target FPR β, we set τ_β as the $(1-\beta)$-quantile of $\mathcal{S}^{(\mathcal{B})}$, and predict jailbreak if $s^{(t)} > \tau_\beta$. MTK provides a highly discriminative representation (see the UMAP projection of rank features in Fig. 5(a)), enabling good performance even with simple anomaly detectors. Sec. 6.5 shows consistent results when using one-class SVM or PCA.

6 Evaluation

In this section, we evaluate our MTK. Our experiments aim to answer the following *research questions* (RQs):

- **RQ1:** How effective is MTK against diverse jailbreak attacks, and how does it compare to existing defenses?
- **RQ2:** How robust is MTK to diverse adaptive attacks that explicitly optimize to evade detection?
- **RQ3:** Can MTK transfer to VLMs for detecting multimodal jailbreak attacks without redesigning the detector?
- **RQ4:** How robust is MTK to pseudo-malicious prompts, including more challenging multimodal variants?
- **RQ5:** How does MTK compare to others in computational efficiency, and how sensitive is it to hyperparameters?

Datasets and models. Our evaluation corpus comprises two partitions, benign and jailbreak samples. The benign partition includes prompts from the TOXICCHAT [41] labeled toxicity = 0 and pseudo-malicious prompts from OR-BENCH [35]. The jailbreak partition is generated from ADVBENCH [4] malicious queries using jailbreak attacks. We evaluate on four popular LLMs spanning different families, including LLaMA2-7B-chat-hf [2], LLaMA3-8B-instruct [42], Mistral-7B-instruct-v0.2 [43], and Vicuna-7B-v1.5 [38].

Attack methods. We select **ten** representative jailbreak attacks, including `AutoDAN` [27], `DrAttack` [28], `IJP` [29], `JailJudge` [30], `GCG` [4], `PAIR` [32], `PAP` [44], `SAA` [5], `TAP` [39], and `Zulu` [45].

Competitors. We compare MTK against seven SOTA LLM jailbreak detectors, including `GradCuff` [17], `GradSafe` [18], `HSF` [13], `HiddenDetect` [7], `SaP` [9], `SelfDefend` [24], and `SmoothLLM` [20]. We also compare against six VLM detectors, including `ECSO` [46], `HiddenDetect` [7], `MirrorCheck` [47], `CIDER` [48], `JailGuard` [21], and `JailDAM` [15].

Defense settings. Our detector is trained on a reference bank of 800 benign and 800 malicious prompts. The benign set consists of 300 prompts randomly sampled from DATABRICKS DOLLY 15K [37], 300 from ALPACA [49] and 200 PMPs from OR-BENCH [35]. The malicious set contains 800 prompts evenly drawn from MALICIOUSINSTRUCT [50], PKU-SAFERLHF [34], and ADVBENCH [4]. Note that although some jailbreak attacks also generate ADVBENCH-derived jailbreak prompts, we ensure that the bank and test sets are fully sample-level disjoint, preventing information leakage. We further examine the effect of bank-test distribution overlap; for distribution-level mismatch results, see Apdx. B.2. We set k, max_{samples}, and $n_{\text{estimators}}$ to 10, 512, and 500. Details are moved to Apdx. A.1.

Evaluation metrics. To comprehensively evaluate performance, we report AUROC as well as F1, TPR, and Precision at FPR = 5%, using a threshold calibrated on 20% of the validation set. We further analyze performance across a range of FPR thresholds (see Tab. 9). Tab. 4 are reported as the average ± standard deviation across three runs.

6.1 RQ1: Effectiveness of MTK

We compare the jailbreak detection performance of our MTK across four LLMs under ten jailbreak attacks, as shown in Tab. 4. MTK achieves superior AUROC overall and substantially outperforms other methods, *ranking best in* 31 *of* 40 *cases and second best in* 6. Averaged over the ten attacks on four LLMs, MTK attains a mean AUROC of 0.940. `GradSafe`, `GradCuff`, `SaP`, and `SmoothLLM` perform well in some cases but exhibit sharp degradation across attacks or models. `SelfDefend` is the strongest baseline overall, benefiting from an additional LLM (GPT) as a judge, but we highlight in Sec. 4.3 that it can incur high FPR, especially on PMPs. `HSF` and `HiddenDetect` deliver reasonable results, yet their performance varies across attacks on the same model, indicating limited consistency, with large average standard deviations of 0.209 (`HSF`) and 0.155 (`HiddenDetect`), *compared to just* 0.038 *for* MTK. These results highlight MTK's superior effectiveness across jailbreak attacks and LLMs. We also evaluate MTK on more recent models across multiple scales; see Apdx. B.1.

Bank–test disjointness and design choices. In Sec. 6, the Defense settings paragraph explicitly describes how the reference bank is built (source and size), and the bank and the evaluation data are fully sample-level disjoint. We further study how bank source and size affect detection in Sec. 6.5 (Tab. 7a, Tab. 7b): performance is robust to composition (AUROC std= 0.034 across 9 configurations). Robustness to *bank–test distribution mismatch*—both for out-of-distribution benign prompts and for jailbreak prompts drawn from non-ADVBENCH benchmarks (JBB-Behaviors, HarmBench)—is reported in Apdx. B.3 and Apdx. B.2.

6.2 RQ2: Robustness to Adaptive Attacks

We follow prior practice and construct adaptive attacks by augmenting the original jailbreak objective with a defense-aware loss. Because our manifold-trajectory extraction (*i.e.,* the rank list) is non-invertible and non-differentiable, the attacker cannot directly optimize MTK's manifold features as in earlier settings. Instead, the attacker must craft surrogate objectives that encourage malicious inputs to imitate the manifold signatures of benign prompts. We consider three heuristic losses:

$$\mathcal{J}_1 = (1-\lambda_1)\,\mathcal{L}_{\text{adv}} + \lambda_1 \mathcal{L}^1_{\text{evasion}}, \tag{18}$$

$$\mathcal{J}_2 = (1-\lambda_2)\,\mathcal{L}_{\text{adv}} + \lambda_2 \mathcal{L}^2_{\text{evasion}}, \tag{19}$$

$$\mathcal{J}_3 = (1-\lambda_3)\,\mathcal{L}_{\text{adv}} + \lambda_3 \mathcal{L}^3_{\text{evasion}}. \tag{20}$$

Here, $\mathcal{L}^1_{\text{evasion}}$ pulls jailbreak activations toward the benign reference bank by minimizing, at each layer, the summed MSE distance to all benign references. $\mathcal{L}^2_{\text{evasion}}$ instead pulls activations toward a single benign reference by minimizing the MSE distance to the nearest benign reference at each layer. $\mathcal{L}^3_{\text{evasion}}$ acts as an end-to-end surrogate for attacking MTK by shaping benign-like neighborhood structure, minimizing

Table 4: **Evaluation and comparison.** The AUROC of different defenses against jailbreak attacks across multiple models, where "Avg." indicates column-wise average. **Bold values** denote the best performance, and underlined values denote the second best. For a per-attack ASR comparison without any defense, see Apdx. B.4.

Models	Attack \ Defense	MTK (Ours)	GradCuff [17]	GradSafe [18]	HSF [13]	HiddenDetect [7]	SaP [9]	SelfDefend [24]	SmoothLLM [20]
Llama2-7B	AutoDAN [27]	0.927±0.000	0.758±0.000	**0.979±0.000**	0.518±0.000	0.927±0.000	0.780±0.000	0.945±0.000	0.733±0.000
	DrAttack [28]	**0.959±0.003**	0.576±0.000	0.776±0.000	0.929±0.001	0.763±0.000	0.500±0.000	0.713±0.000	0.180±0.000
	IJP [29]	**0.924±0.006**	0.712±0.000	0.863±0.000	0.876±0.000	0.673±0.000	0.685±0.000	0.891±0.000	0.224±0.000
	JailJudge [30]	**0.929±0.000**	0.673±0.000	0.857±0.000	0.561±0.002	0.924±0.000	0.497±0.000	0.619±0.000	0.835±0.000
	GCG [4]	0.929±0.000	0.570±0.000	0.904±0.000	0.702±0.004	0.786±0.000	0.653±0.000	**0.939±0.001**	0.551±0.000
	PAIR [32]	**0.953±0.000**	0.719±0.000	0.936±0.000	0.790±0.001	0.856±0.000	0.509±0.000	0.893±0.002	0.444±0.000
	PAP [44]	**0.944±0.002**	0.512±0.000	0.252±0.000	0.695±0.004	0.432±0.000	0.844±0.000	0.801±0.001	0.634±0.000
	SAA [5]	**0.961±0.001**	0.637±0.000	0.886±0.000	0.825±0.015	0.615±0.000	0.466±0.000	0.946±0.000	0.718±0.000
	TAP [39]	**0.956±0.000**	0.772±0.000	0.941±0.000	0.853±0.002	0.858±0.000	0.502±0.000	0.861±0.000	0.516±0.000
	Zulu [45]	**0.922±0.002**	0.545±0.000	0.733±0.000	0.448±0.007	0.921±0.000	0.890±0.000	0.876±0.000	0.745±0.000
	Avg.	**0.940±0.000**	0.647±0.000	0.813±0.000	0.720±0.000	0.776±0.000	0.633±0.000	0.848±0.000	0.558±0.000
Llama3-8B	AutoDAN [27]	0.961±0.000	0.342±0.000	**0.967±0.000**	0.696±0.000	0.722±0.000	0.598±0.000	0.945±0.000	0.952±0.000
	DrAttack [28]	**0.968±0.000**	0.387±0.000	0.388±0.000	0.840±0.000	0.530±0.000	0.656±0.000	0.706±0.000	0.663±0.000
	IJP [29]	**0.909±0.002**	0.366±0.000	0.376±0.000	0.739±0.000	0.501±0.000	0.601±0.000	0.888±0.000	0.651±0.000
	JailJudge [30]	**0.949±0.000**	0.688±0.000	0.731±0.000	0.678±0.000	0.675±0.000	0.584±0.000	0.618±0.000	0.715±0.000
	GCG [4]	**0.981±0.000**	0.368±0.000	0.608±0.000	0.651±0.000	0.610±0.000	0.627±0.000	0.945±0.000	0.970±0.000
	PAIR [32]	**0.940±0.000**	0.450±0.000	0.879±0.000	0.830±0.000	0.626±0.000	0.604±0.000	0.789±0.000	0.830±0.000
	PAP [44]	**0.891±0.000**	0.441±0.000	0.861±0.000	0.840±0.000	0.811±0.000	0.689±0.000	0.796±0.000	0.733±0.001
	SAA [5]	0.938±0.000	0.187±0.000	0.065±0.000	0.002±0.000	0.088±0.000	0.874±0.000	**0.946±0.001**	0.878±0.000
	TAP [39]	**0.952±0.000**	0.415±0.000	0.888±0.000	0.819±0.000	0.750±0.000	0.639±0.000	0.866±0.000	0.946±0.000
	Zulu [45]	**0.949±0.000**	0.673±0.000	0.559±0.000	0.216±0.000	0.524±0.000	0.379±0.000	0.876±0.000	0.806±0.000
	Avg.	**0.944±0.000**	0.432±0.000	0.632±0.000	0.631±0.000	0.584±0.000	0.625±0.000	0.837±0.000	0.814±0.000
Mistral-7B	AutoDAN [27]	**0.990±0.000**	0.820±0.000	0.431±0.000	0.717 ±0.000	0.858±0.000	0.505±0.000	0.945±0.000	0.357±0.000
	DrAttack [28]	**0.945±0.001**	0.566±0.000	0.383±0.000	0.918±0.000	0.926±0.000	0.591±0.000	0.698±0.000	0.304±0.000
	IJP [29]	**0.943±0.000**	0.609±0.000	0.337±0.000	0.931±0.000	0.337±0.000	0.760±0.000	0.888±0.000	0.609±0.000
	JailJudge [30]	**0.918±0.000**	0.481±0.000	0.450±0.000	0.546±0.000	0.883±0.000	0.592±0.000	0.630±0.000	0.770±0.000
	GCG [4]	0.976±0.000	0.500±0.000	0.876±0.000	0.738±0.000	**0.990±0.000**	0.853±0.000	0.946±0.000	0.659±0.000
	PAIR [32]	**0.977±0.000**	0.698±0.000	0.681±0.000	0.937±0.000	0.933±0.000	0.767±0.000	0.883±0.000	0.610±0.000
	PAP [44]	**0.924±0.000**	0.816±0.000	0.492±0.000	0.657±0.000	0.815±0.000	0.539±0.000	0.821±0.000	0.310±0.000
	SAA [5]	**0.960±0.000**	0.340±0.000	0.043±0.000	0.948±0.000	0.538±0.000	0.547±0.000	0.946±0.000	0.265±0.000
	TAP [39]	**0.955±0.000**	0.709±0.000	0.652±0.000	0.932±0.000	0.860±0.000	0.637±0.000	0.846±0.000	0.510±0.000
	Zulu [45]	0.938±0.001	0.456±0.000	0.390±0.000	0.760±0.000	0.753±0.000	0.562±0.000	0.876±0.000	**0.975±0.000**
	Avg.	**0.953±0.000**	0.600±0.000	0.473±0.000	0.808±0.000	0.789±0.000	0.635±0.000	0.848±0.000	0.537±0.000
Vicuna-7B	AutoDAN [27]	**0.957±0.002**	0.635±0.000	0.497±0.000	0.226±0.000	0.875±0.000	0.584±0.000	0.943±0.000	0.535±0.000
	DrAttack [28]	**0.956±0.001**	0.697±0.000	0.401±0.000	0.898±0.000	0.948±0.000	0.713±0.000	0.753±0.000	0.473±0.000
	IJP [29]	**0.937±0.000**	0.510±0.000	0.153±0.000	0.865±0.000	0.841±0.000	0.451±0.000	0.890±0.000	0.509±0.000
	JailJudge [30]	0.921±0.000	0.675±0.000	0.258±0.000	0.442±0.000	**0.924±0.000**	0.716±0.000	0.620±0.000	0.423±0.000
	GCG [4]	**0.957±0.000**	0.863±0.000	0.601±0.000	0.591±0.000	0.954±0.000	0.584±0.000	0.930±0.000	0.864±0.000
	PAIR [32]	**0.922±0.003**	0.668±0.000	0.507 ±0.000	0.866±0.000	0.913±0.000	0.545±0.000	0.913±0.000	0.571±0.000
	PAP [44]	**0.895±0.002**	0.585±0.000	0.475 ±0.000	0.752±0.001	0.836±0.000	0.443±0.000	0.798±0.000	0.490±0.000
	SAA [5]	0.835±0.005	0.430±0.000	0.012±0.000	**0.976±0.000**	0.799±0.000	0.458±0.000	0.948±0.001	0.354±0.000
	TAP [39]	**0.947±0.001**	0.712±0.000	0.683±0.000	0.880±0.000	0.917±0.000	0.578±0.000	0.878±0.000	0.661±0.001
	Zulu [45]	0.910±0.001	0.907±0.000	0.215±0.000	0.608±0.000	**0.943±0.000**	0.247±0.000	0.876±0.000	0.900±0.000
	Avg.	**0.923±0.001**	0.668±0.000	0.380±0.000	0.710±0.000	0.895±0.000	0.532±0.000	0.855±0.000	0.578±0.000

the summed distance to the benign bank while simultaneously maximizing the summed distance to the malicious bank, thereby reducing benign-neighbor ranks. Formal definitions of these loss terms are provided in Apdx. A.2.

We implement the three adaptive attacks above on GCG against LLaMA2, keeping all other settings identical to those in Tab. 4. For each attack and each $\lambda \in [0.1, 0.9]$, we run a sufficient 1000 optimization epochs. As shown in Tab. 5, across different values of λ, these attacks either yield low ASR or incur high TPR. Overall, MTK remains highly robust to all three attacks, achieving an average TPR of 0.85. We conjecture that this robustness stems from the fact that none of $\mathcal{J}_1$, $\mathcal{J}_2$, or $\mathcal{J}_3$ can fundamentally erode separability on the manifold, as MTK captures more geometric regularities that distinguish benign from jailbreak behavior. Consequently, the probability of a successful jailbreak that also evades detection, measured by eASR, remains at most 0.19 in all cases (*Avg.* 0.077). For cross-defense comparison, we report each defense under its highest-ASR attack configuration, applying the same selection criterion to all baselines, and report the resulting eASR as the evasion outcome. Tab. 2 shows that adaptive attacks can raise eASR to an average of 0.74 against prior defenses. For MTK, we report $\mathcal{J}_3$, the strongest MTK-targeted adaptive loss considered in our evaluation, at its highest-ASR setting ($\lambda = 0.1$). These results support MTK's robustness within the evaluated threat model.

Beyond surrogate-based attacks: differentiable rank approximation. A potential concern is whether MTK's robustness primarily stems from the non-differentiability of the rank operation, which forces attackers to rely on surrogate objectives. To directly test this, we replace the discrete benign-neighbor rank with a differentiable soft-rank approximation, allowing gradients to flow through the rank feature back to the adversarial suffix. The attacker then attempts to evade detection by minimizing the soft-rank of jailbreak samples. Under the same Llama2-7B setup with a full λ sweep, this attack

Table 5: Full robustness sweep of MTK under three adaptive attacks ($\mathcal{J}_1$, $\mathcal{J}_2$, $\mathcal{J}_3$) across λ on Llama2-7B. For cross-defense comparison in Tab. 2, we select $\mathcal{J}_3$ as the strongest MTK-targeted attack—it directly approximates MTK's rank-based neighborhood signal, whereas $\mathcal{J}_1/\mathcal{J}_2$ are weaker proxies—and report it at $\lambda = 0.1$ (its highest-ASR setting across the λ sweep, bolded). MTK remains strong across different values of λ and evasion losses.

λ	$\mathcal{J}_1$ Loss			$\mathcal{J}_2$ Loss			$\mathcal{J}_3$ Loss		
	ASR	TPR	eASR	ASR	TPR	eASR	ASR	TPR	eASR
0.1	0.62	0.80	0.12	0.75	0.75	0.19	**0.72**	**0.76**	**0.17**
0.2	0.67	0.76	0.16	0.30	0.89	0.03	0.62	0.95	0.03
0.3	0.64	1.00	0.00	0.00	1.00	0.00	0.71	0.95	0.04
0.4	0.67	0.90	0.07	0.00	1.00	0.00	0.65	0.76	0.16
0.5	0.62	0.81	0.12	0.05	1.00	0.00	0.67	0.95	0.03
0.6	0.58	0.95	0.03	0.00	1.00	0.00	0.62	0.71	0.18
0.7	0.57	0.86	0.08	0.00	1.00	0.00	0.62	0.81	0.12
0.8	0.38	0.52	0.18	0.00	1.00	0.00	0.54	0.81	0.10
0.9	0.21	0.29	0.15	0.00	1.00	0.00	0.57	0.76	0.14
Avg.	0.55	0.77	0.10	0.12	0.96	0.02	0.64	0.83	0.11

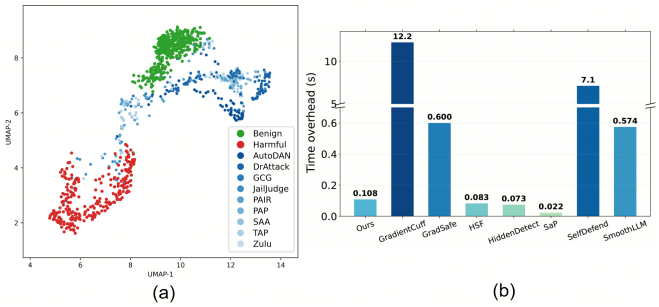
Table 6: **Evaluation and comparison.** AUROC of defenses against VLM jailbreak attacks. **Bold** denotes the best.

VLMs↓	Detector↓	MM-SafetyBench [51]	FigImg [52]	JailBreakV-28K [53]	Avg.
LLaVa	HiddenDetect [7]	**0.968**	0.877	0.938	0.928
	ECSO [46]	0.848	0.761	0.866	0.825
	MirrorCheck [47]	0.688	0.707	0.611	0.669
	CIDER [48]	0.673	0.685	0.607	0.655
	JailGuard [21]	0.533	0.522	0.586	0.547
	JailDAM [15]	0.612	0.671	0.598	0.627
	MTK (**Ours**)	0.911	**0.959**	**0.949**	**0.940**
Qwen-VL	HiddenDetect [7]	0.774	0.980	0.881	0.878
	ECSO [46]	0.686	0.651	0.712	0.683
	MirrorCheck [47]	0.781	0.716	0.628	0.708
	CIDER [48]	0.673	0.685	0.607	0.655
	JailGuard [21]	0.750	0.855	0.823	0.809
	JailDAM [15]	0.519	0.634	0.627	0.593
	MTK (**Ours**)	**0.818**	**0.992**	**0.964**	**0.924**

still fails to evade MTK: TPR remains at 0.86, comparable to the surrogate-based attacks in Tab. 5. This suggests that MTK's robustness is not solely due to the rank operation's non-differentiability, but is also supported by the depth-wise trajectory signature observed in our evaluated attacks. We acknowledge this caveat further in Limitations (Sec. 7).

6.3 RQ3: Transferability to VLMs

Dataset and model. We evaluate MTK on two popular VLMs, LLaVA-1.6-7B [56] and Qwen-VL-Chat [57]. Benign evaluation set is MM-VET [58], a widely used benchmark for evaluating VLMs on integrated vision-language capabilities. We evaluate multimodal jailbreak attacks from three datasets: MM-SAFETYBENCH [51], JAILBREAKV-28K [53], and FIGIMG[52]. MM-SAFETYBENCH and JAILBREAKV-28K include typographical images, stable diffusion-generated images, and hybrid variants. FIGIMG contains typographical jailbreak images paired with prompts targeting ten toxic themes. We evaluate six SOTA detectors tailored for VLMs.



(a) (b)

Figure 5: (a) UMAP projection of rank information. (b) Average per-prompt detection time.

Defense setting. To transfer MTK to the VLM setting, we modify the reference bank to include multimodal data. The benign reference set comprises 250 prompts randomly sampled from VQA [59] and 50 multimodal PMPs from USB [40]. The malicious reference set contains 300 image–text pairs, where the text comes from ADVBENCH and the malicious images are generated from the corresponding prompts using Stable Diffusion v1.5, following LoD [10]. All other settings are kept the same as in the LLM experiments.

Results. Tab. 6 reports results on three jailbreak datasets and two VLMs. Our MTK again performs strongly, achieving average AUROC scores of 0.940 on LLaVA and 0.924 on Qwen-VL. These results underscore the robustness of our trajectory kinetics representation, which remains effective in both text-only and vision-language settings.

6.4 RQ4: Robustness to Multimodal PMPs

Tab. 3 have compared robustness to text-only PMPs, where MTK achieves a remarkably low FPR_{PMP} of 0.02. We also consider data augmentation for existing defenses by incorporating PMPs during their training, but this does not effectively reduce FPR_{PMP} and can even degrade TPR. We attribute these failures to the detectors' brittle, unstructured representations rather than insufficient training data. To show that this issue is not specific to text-only settings, we further study multimodal pseudo-malicious samples in VLMs and evaluate VLM jailbreak detectors on them.

We evaluate multimodal PMPs from USB as benign samples and MM-SAFETYBENCH as the jailbreak samples, with all other settings matching Tab. 6. Tab. 8 shows that VLM detectors also produce substantial false positives on multimodal PMPs, with $\text{FPR}_{\text{PMP}} > 0.1$ throughout and up to 0.445 for `JailGuard`. MTK remains robust, keeping the mean test FPR_{PMP} at 0.044 with a Precision of 0.900. We believe that strong performance on pseudo-malicious datasets across multiple modalities provides compelling evidence of robustness.

Table 7: **Ablation studies.** We report results with AUROC on LLaMA2.

(a) **Varying reference-bank sizes.**

Attack ↓	200	800	1200	1600
AutoDAN	0.912	0.927	0.890	0.909
DrAttack	0.830	0.959	0.936	0.922
GCG	0.771	0.929	0.909	0.887
Avg.	0.838	0.938	0.912	0.906

(b) **Training-set composition.**

Malicious↓	Benign↓		
	Alpaca [49]	**Dolly** [37]	**Ultrachat** [54]
AdvBench [4]	0.883	0.879	0.894
SafeRLHF [34]	0.905	0.929	0.818
HarmfulQA [55]	0.841	0.900	0.875

(c) **Anomaly detector choice.**

Attack ↓	PCA	IF	1-SVM
AutoDAN	0.865	0.927	0.900
DrAttack	0.806	0.959	0.878
GCG	0.770	0.929	0.856
Avg.	0.814	0.938	0.878

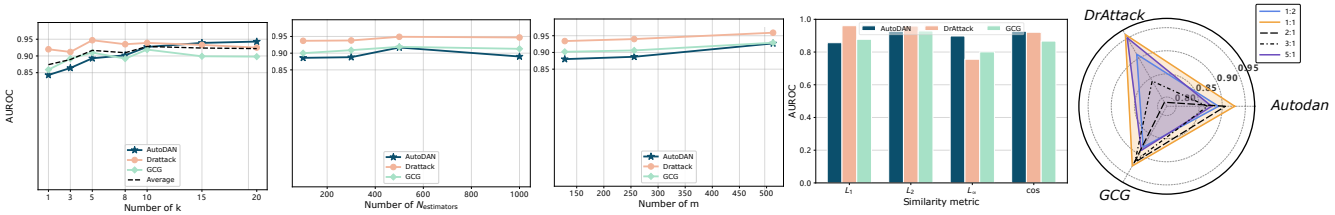


Figure 6: **Ablation studies.** The effect of k, $n_{\text{estimators}}$, max_{samples}, the distance metric $\mathbb{D}$, and the benign-to-malicious ratio.

Table 8: **Robustness to multimodal PMPs.** Lower FPR_{PMP} and higher precision indicate fewer false alarms.

Model	Metric	HiddenDetect	MirrorCheck	ECSO	CIDER	JailGuard	JailDAM	Ours
LLaVa	FPR_{PMP} ↓	0.234	0.210	0.147	0.124	0.445	0.271	**0.014**
	Precision↑	0.577	0.769	0.714	0.129	0.586	0.624	**0.954**
Qwen-VL	FPR_{PMP} ↓	0.115	0.183	0.247	0.124	0.236	0.214	**0.073**
	Precision↑	0.777	0.636	0.571	0.129	0.632	0.594	**0.846**

6.5 RQ5: Efficiency and Sensitivity

Efficiency and deployment overhead. We measure the per-sample inference latency of each detector in milliseconds, using an NVIDIA A100 80GB GPU. As shown in Fig. 5(b), MTK is among the most efficient approaches. While a few methods are faster, we achieve substantially stronger performance. Moreover, MTK incurs minimal training cost, requiring only a lightweight anomaly detector (Isolation Forest), and can be deployed as a *lightweight, plug-and-play* module on top of an LLM/VLM as a final safety firewall.

Effect of reference bank. We study how the number of anchors in the reference bank affects performance by setting both benign and malicious anchors to 200, 800, 1200, and 1600 per class. Tab. 7a shows that performance improves notably from 200 to 800 anchors (our default) and remains robust from 800 to 1600, while further increasing anchor size yields only marginal gains, suggesting that *anchor diversity matters more than sheer quantity*. These results also indicate that MTK is robust to anchor reduction, as it maintains strong performance even when the reference bank is compressed. Overall, MTK still performs well even with relatively few anchors. *We further examine the impact of reference bank composition.* Specifically, we use Alpaca [49], Databricks Dolly [37], and UltraChat [54] as benign anchor sources, and AdvBench [4], SafeRLHF [34], and HarmfulQA [55] as malicious anchor sources, always keeping a $1:1$ benign-to-malicious ratio (Fig. 6 also varies this ratio and finds 1:1 performs best). As shown in Tab. 7b, MTK consistently achieves strong detection performance across all anchor configurations (mean AUROC 0.880 ± 0.034).

Effect of hyperparameters. We vary k and the IF hyperparameters max_{samples} and $n_{\text{estimators}}$ in Fig. 6. Increasing k improves performance with diminishing returns beyond $k = 10$, which we use throughout. Performance is largely insensitive to $n_{\text{estimators}}$. For max_{samples}, larger values increase runtime, and $max_{\text{samples}} = 512$ achieves the best overall performance. Therefore, MTK is not sensitive to hyperparameter choices.

Effect of LLM layer selection. We further study how the choice of different layers affects detection performance. We consider Llama2-7B and Llama3-8B, and, excluding the embedding layer, evaluate three layer-selection strategies: shallow layers (Layers 0–15), deep layers (Layers 16–31), and all layers. Fig. 7 reports AUROC under these settings. Using deep-layer hidden states consistently outperforms shallow-layer features, and using all layers yields the best performance. Notably, using only the deep half still achieves good results, supporting a layer-reduced variant of MTK that roughly halves computational overhead while retaining competitive accuracy.

Effect of anomaly detection algorithm. We replace the Isolation Forest (IF) with PCA and one-class SVM (1-SVM), two common anomaly detectors. Tab. 7c shows consistently strong performance across choices, suggesting MTK is broadly applicable. The UMAP in Fig. 5(a) further reveals clear rank-feature separation, enabling simple detectors.

Effect of distance metric. We further study the effect of the distance metric $\mathbb{D}$ in Eq. 14 used to measure hidden-state similarity. Fig. 6 shows that ℓ_1 and cosine distance perform well, while ℓ_2 achieves the best results and is our default.

Effect of rank vs. hybrid rank–distance. We test whether

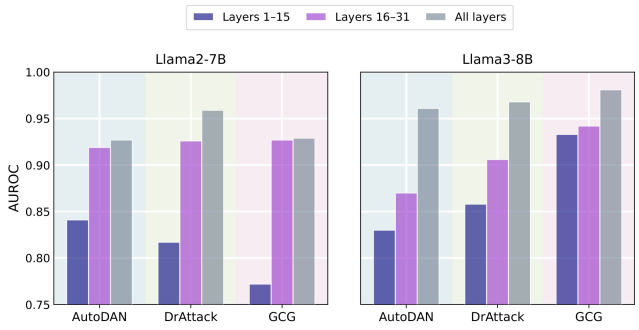


Figure 7: Effect of LLM layer selection.

augmenting the rank-based trajectory with the underlying activation distance—*e.g.,* forming a hybrid score $r_\ell \cdot d_\ell$ that retains both neighborhood ordering and absolute geometric distance—would further improve detection. We test this hybrid variant on Llama2-7B under GCG attacks; AUROC changes by less than 1% relative to our rank-only design. This suggests that the k-nearest-neighbor mean rank already captures the geometric structure required for discrimination, while remaining agnostic to model-specific coordinate scales, supporting our choice of rank-only trajectories.

Cross-benchmark evaluation. Our main evaluation generates jailbreak prompts from ADVBENCH [4], a widely used but single-source benchmark. To verify that the reported results are not specific to the ADVBENCH prompt style, we additionally evaluate MTK on two recent benchmarks: JBB-Behaviors [60] and HarmBench [61]. Performance on JBB-Behaviors and HarmBench remains strong (see Apdx. B.2).

Input-length sensitivity. We further examine whether MTK's performance is driven by test input length by evaluating across length buckets and under a bank-test length mismatch setting. The results show stable performance in both cases; see Apdx. B.5.

Performance at different FPRs. Tab. 9 reports operating-point performance under GCG at varying false-positive budgets. At extremely low 0.1% FPR, *all methods struggle*, yet MTK still performs best. As the budget increases, MTK improves sharply and consistently outperforms others.

7 Discussion, Limitations, Deployment, and Future Work

Limitations. Fundamentally, MTK grounds detection in manifold geometry and assumes that jailbreak samples follow trajectories that are distinct from benign ones. Our experiments suggest that this premise holds broadly, even under adaptive attacks. Although these adaptive attacks achieve high ASR and can evade prior detectors, the resulting jailbreak samples still manifest the expected trajectory irregularities. That said, we acknowledge an important caveat on the adaptive-attack evaluation. Because MTK's rank-based pipeline is not directly differentiable, all our adaptive attacks—the three surrogate losses ($\mathcal{J}_1$-$\mathcal{J}_3$) and the differentiable soft-ranking proxy—necessarily rely on surrogate or approximate objectives rather than direct gradient propagation through the discrete rank operation. While our results (average TPR$\geq$0.77 across three surrogate losses; TPR=0.86 with soft-ranking) provide strong empirical evidence that MTK's robustness extends beyond non-differentiability, we recognize that future work may uncover stronger attacks that invalidate this premise, for example, by shaping jailbreak activations to remain within benign open balls throughout the neural network, while still eliciting harmful outputs.

Table 9: Detection performance at different FPRs (@0.1%–@10%) in terms of TPR, precision, and F1.

Attack	Defense	Metric	@0.1%	@1%	@2.5%	@5%	@10%
GCG	HSF	TPR	0.12	0.19	0.30	0.31	0.43
		Precision	0.99	0.95	0.94	0.87	0.82
		F1	0.25	0.32	0.46	0.46	0.56
	HiddenDetect	TPR	0.03	0.52	0.74	0.83	0.88
		Precision	0.98	0.98	0.97	0.94	0.90
		F1	0.05	0.68	0.84	0.88	0.89
	Ours	TPR	**0.19**	**0.74**	**0.90**	**0.93**	**0.95**
		Precision	**0.99**	**0.99**	**0.97**	**0.94**	**0.90**
		F1	**0.32**	**0.85**	**0.93**	**0.94**	**0.93**

Complex benign prompts and availability risk. Beyond the adaptive-attack caveat above, two additional limitations deserve attention. First, linguistically complex benign prompts—such as sarcasm, hypothetical safety discussions, or extended role-play scenarios—may produce irregular layer-wise trajectories that resemble those of jailbreak prompts and inflate the false-positive rate. While our PMP evaluation (Sec. 4.3) and OOD benign experiments (Apdx. B.3) suggest MTK remains robust on common variants, we do not claim exhaustive coverage of all such linguistically complex patterns. Second, this opens an *availability* risk: an adversary aware of the detector could deliberately craft benign-but-irregular prompts (*e.g.,* elaborate role-play or contrived sensitive-keyword scenarios) to trigger over-refusal and degrade the service of the protected LLM. Such toxic-surface but benign-intent inputs are the kind considered in our pseudo-malicious prompt setting, and we find that MTK is more robust than baselines—e.g., on harmless queries in toxic persona templates from OR-Bench, MTK achieves FPR=0.03 versus GradSafe=0.09 and GradCuff=0.14 (consistent with Tab. 3). Nonetheless, mitigating availability attacks—particularly distinguishing genuine edge-case benign use from adversarial over-refusal probes—remains an open problem.

Deployment considerations. We highlight the main operational considerations for practitioners deploying MTK in production. *(i) Reference-bank dependence.* MTK's detection signal is computed against a fixed reference bank of benign and malicious anchors. Our ablations (Tab. 7a, Tab. 7b) indicate that performance is robust to bank size and composition (AUROC std= 0.034 across 9 configurations), yet poorly

curated banks—*e.g.,* drawn entirely from a narrow domain—may still degrade detection quality. *(ii) Distribution shift and bank maintenance.* As the underlying LLM is updated or as the threat landscape shifts (*e.g.,* new "forbidden knowledge" categories emerge), the reference bank may need periodic refreshing—a standard ML lifecycle challenge shared by all internal-signal detectors. Importantly, MTK's zero-jailbreak-data design makes this lightweight: a small number of policy-relevant benign or malicious samples may often suffice in practice, with no jailbreak-sample curation or fine-grained annotation required; model updates only need hidden-state re-extraction, leaving the rank-trajectory pipeline itself unchanged. *(iii) Computational overhead.* MTK's per-prompt detection latency is competitive with baselines (Fig. 5b). To address scalability concerns specific to the distance computation step, rank computation is parallelized across layers, bank size, and feature dimensions via batched tensor operations—taking only 4.2ms on an A100—and can be streamed layer-by-layer to overlap with the forward pass. *(iv) Input length and prompt style.* AUROC remains stable across input length buckets (<64 to 256–512 tokens) and is robust to bank-test length mismatch (Apdx. B.5). Performance on prompt styles beyond ADVBENCH (JBB-Behaviors, HarmBench) is also strong (Apdx. B.2), though deployment-specific recalibration may be advisable for heavily out-of-distribution use cases.

Future work. MTK characterizes the evolution dynamics of an input by tracking how its neighborhood structure changes across layers, using the layer-wise rank sequence of nearest benign neighbors as a compact surrogate for manifold proximity. This view yields strong robustness to adaptive jailbreak attempts and pseudo-malicious samples. Nonetheless, there is room to improve how these evolutionary dynamics are modeled. *Using only the neighbor rank as a notion of "closeness" is a straightforward approach but may not be optimal.* Future work could explore richer proximity measures that better quantify an input's distance to the benign manifold at each layer. We expect that more principled notions of manifold "closeness" will further strengthen MTK.

8 Conclusion

In this paper, we show that existing detectors largely rely on static detection of internal states, making them brittle against more complex inputs, *i.e.,* adaptive attacks and PMPs that are carefully crafted to entangle the feature space. We then shift the analysis from static signals to dynamic manifold geometry and propose MTK, which uses differences in how manifold structures evolve. Extensive experiments spanning four LLMs and two VLMs, with 10 attacks and 12 competing defenses, show MTK's effectiveness. By introducing a manifold-aware perspective, our work deepens the understanding of jailbreak attacks and defenses.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No. 62572206. Wei Wan is the corresponding author.

Ethical Considerations

We structure the ethical considerations of this work as follows: a stakeholder-based analysis with process impact, the impact of the research, mitigations, the ethical principles applied, and the publication justification.

Stakeholder Analysis and Process Impact. *(1) End users* of deployed LLMs/VLMs rely on safe behavior and absence of over-refusal. *(2) Model providers and deployers* rely on practical detection layers. *(3) Research team members* may be exposed to disturbing content during jailbreak evaluation. *(4) Authors of prior defenses* (HiddenDetect, HSF, SaP, GradSafe) have their prototypes stress-tested under adaptive attacks. *(5) Potential misusers* could learn from dual-use components. We use only public data and open-weight models; no live-service probing, no human subjects, no PII collection.

Impact of the Research. *Positive.* (1) MTK reduces harmful outputs under adaptive adversaries and reduces over-refusal on pseudo-malicious prompts—disproportionately benefiting users with medical, legal, or harm-reduction queries. (2) MTK is zero-jailbreak-data, lightweight, and plug-and-play, lowering the barrier for smaller deployers. *Negative.* (1) The adaptive objectives $\mathcal{J}_1$–$\mathcal{J}_3$ disclose surrogate attack designs, though they offer no new capability beyond what GCG already provides. (2) Residual false positives on complex benign prompts (sarcasm, role-play) create service-degradation risk.

Mitigations. *Implemented.* (1) **Team members:** exposure was concentrated among experienced, voluntary participants with opt-out rights; we reused public ADVBENCH seeds rather than authoring new harmful content. (2) **Prior-defense authors:** attacks combine public GCG with published defense details, introducing no new attack primitive and revealing no undisclosed product vulnerability. (3) **Dual-use exposure:** demonstrations use open-weight 7B–8B models and public benchmarks; released artifacts are defense-oriented. *Recommended Future Deployment.* (1) **Defense-in-depth:** deploy with human-in-the-loop review rather than silent blocks. (2) **Bank maintenance:** periodic refresh of the reference bank as safety norms evolve (Sec. 7).

Ethical Principles Applied. We apply the four Menlo Report principles. *Beneficence:* benefits (stronger detection, reduced over-refusal) outweigh risks (incremental attacker knowledge, residual false positives). *Respect for Persons:* no human subjects, no PII, no crowdworkers exposed to harmful content. *Justice:* reduced over-refusal distributes benefits to users whose preventive, educational, or harm-reduction queries are currently blocked unfairly. *Respect for Law and Public Interest:* only public benchmarks; no terms-of-service violations.

Publication Justification. This work strengthens defenses against jailbreak attacks while reducing the documented harm of over-refusal. Conducting and publishing it is justified for three reasons: (1) we use exclusively public datasets and open-weight models, with offensive components from already-published primitives, introducing no new attack capability against undefended systems; (2) withholding the adaptive-attack analyses would obscure the actual robustness landscape of MTK and prior defenses, an ethical concern in itself; (3) the pseudo-malicious robustness contribution addresses a concrete documented harm. We considered non-publication and restricted release; the dual-use risk does not meet the threshold for withholding under the above conditions.

Open Science

We are committed to following open science policy. We provide a detailed description of the experimental methodology in the paper, including model selection, dataset collection and preprocessing, parameter settings, and the exact evaluation protocol. We release our artifact package via a publicly accessible, permanent archival link https://doi.org/10.5281/zenodo.20250387 (mirrored on GitHub at https://github.com/Rookie143/mtk), enabling reviewers and the community to reproduce and validate our results.

References

[1] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[2] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[3] A. Wei, N. Haghtalab, and J. Steinhardt, "Jailbroken: How does llm safety training fail?" in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS'23)*, vol. 36, 2023.

[4] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," *arXiv preprint arXiv:2307.15043*, 2023.

[5] M. Andriushchenko, F. Croce, and N. Flammarion, "Jailbreaking leading safety-aligned llms with simple adaptive attacks," *arXiv preprint arXiv:2404.02151*, 2024.

[6] F. Jiang, Z. Xu, L. Niu, Z. Xiang, B. Ramasubramanian, B. Li, and R. Poovendran, "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)*, 2024, pp. 15 157–15 173.

[7] Y. Jiang, X. Gao, T. Peng, Y. Tan, X. Zhu, B. Zheng, and X. Yue, "Hiddendetect: Detecting jailbreak attacks against large vision-language models via monitoring hidden states," *CoRR*, vol. abs/2502.14744, 2025.

[8] S. Zhang, Y. Zhai, K. Guo, H. Hu, S. Guo, Z. Fang, L. Zhao, C. Shen, C. Wang, and Q. Wang, "Jbshield: Defending large language models from jailbreak attacks through activated concept analysis and manipulation," *arXiv preprint arXiv:2502.07557*, 2025.

[9] X. Chen, Y. As, and A. Krause, "Learning safety constraints for large language models," *CoRR*, vol. abs/2505.24445, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2505.24445

[10] S. Liang, Z. Xu, J. Tao, H. Xue, and X. Wang, "Learning to detect unknown jailbreak attacks in large vision-language models," *arXiv preprint arXiv:2508.09201*, 2025.

[11] Y. Liu, J. Yu, H. Sun, L. Shi, G. Deng, Y. Chen, and Y. Liu, "Efficient detection of toxic prompts in large language models," in *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, ASE 2024*, V. Filkov, B. Ray, and M. Zhou, Eds., 2024, pp. 455–467.

[12] S. Zhou, T. Li, K. Wang, Y. Huang, L. Shi, Y. Liu, and H. Wang, "Investigating coverage criteria in large language models: An in-depth study through jailbreak attacks," *arXiv e-prints*, pp. arXiv–2408, 2024.

[13] C. Qian, H. Zhang, L. Sha, and Z. Zheng, "Hsf: Defending against jailbreak attacks with hidden state filtering," in *Proceedings of the ACM on Web Conference 2025*, 2025, pp. 2078–2087.

[14] C. Zhao, Z. Dou, and K. Huang, "Defending against jailbreak through early exit generation of large language models," in *Proceedings of the International Conference on Neural Information Processing*, 2025, pp. 532–546.

[15] Y. Nian, S. Zhu, Y. Qin, L. Li, Z. Wang, C. Xiao, and Y. Zhao, "Jaildam: Jailbreak detection with adaptive memory for vision-language model," *arXiv preprint arXiv:2504.03770*, 2025.

[16] G. Alon and M. Kamfonas, "Detecting language model attacks with perplexity," *arXiv preprint arXiv:2308.14132*, 2023.

[17] X. Hu, P.-Y. Chen, and T.-Y. Ho, "Gradient cuff: Detecting jailbreak attacks on large language models by

exploring refusal loss landscapes," *Advances in Neural Information Processing Systems*, vol. 37, pp. 126 265–126 296, 2024.

[18] Y. Xie, M. Fang, R. Pi, and N. Gong, "Gradsafe: Detecting jailbreak prompts for llms via safety-critical gradient analysis," *arXiv preprint arXiv:2402.13494*, 2024.

[19] S. Padakandla, S. Babar, M. Kaul *et al.*, "Safequant: Llm safety analysis via quantized gradient inspection," in *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2025, pp. 2522–2536.

[20] A. Robey, E. Wong, H. Hassani, and G. J. Pappas, "Smoothllm: Defending large language models against jailbreaking attacks," *Trans. Mach. Learn. Res.*, 2025.

[21] X. Zhang, C. Zhang, T. Li, Y. Huang, X. Jia, M. Hu, J. Zhang, Y. Liu, S. Ma, and C. Shen, "Jailguard: A universal detection framework for prompt-based attacks on llm systems," *ACM Transactions on Software Engineering and Methodology*, 2025.

[22] M. Phute, A. Helbling, M. Hull, S. Peng, S. Szyller, C. Cornelius, and D. H. Chau, "LLM self defense: By self examination, llms know they are being tricked," in *Proceedings of the Second Tiny Papers Track at ICLR 2024, Tiny Papers @ ICLR 2024, Vienna, Austria, May 11, 2024*. OpenReview.net, 2024.

[23] H. Inan, K. Upasani, J. Chi, R. Rungta, K. Iyer, Y. Mao, M. Tontchev, Q. Hu, B. Fuller, D. Testuggine *et al.*, "Llama guard: Llm-based input-output safeguard for human-ai conversations," *arXiv preprint arXiv:2312.06674*, 2023.

[24] X. Wang, D. Wu, Z. Ji, Z. Li, P. Ma, S. Wang, Y. Li, Y. Liu, N. Liu, and J. Rahmel, "Selfdefend: Llms can defend themselves against jailbreaking in a practical manner," in *Proceedings of the 34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 2441–2460.

[25] M. Sharma, M. Tong, J. Mu, J. Wei, J. Kruthoff, S. Goodfriend, E. Ong, A. Peng, R. Agarwal, C. Anil *et al.*, "Constitutional classifiers: Defending against universal jailbreaks across thousands of hours of red teaming," *arXiv preprint arXiv:2501.18837*, 2025.

[26] L. Gao, J. Geng, X. Zhang, P. Nakov, and X. Chen, "Shaping the safety boundaries: Understanding and defending against jailbreaks in large language models," *arXiv preprint arXiv:2412.17034*, 2024.

[27] X. Liu, N. Xu, M. Chen, and C. Xiao, "Autodan: Generating stealthy jailbreak prompts on aligned large language models," *arXiv preprint arXiv:2310.04451*, 2023.

[28] X. Li, R. Wang, M. Cheng, T. Zhou, and C.-J. Hsieh, "Drattack: Prompt decomposition and reconstruction makes powerful llm jailbreakers," *arXiv preprint arXiv:2402.16914*, 2024.

[29] X. Shen, Z. Chen, M. Backes, Y. Shen, and Y. Zhang, ""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*, 2024, pp. 1671–1685.

[30] F. Liu, Y. Feng, Z. Xu, L. Su, X. Ma, D. Yin, and H. Liu, "Jailjudge: A comprehensive jailbreak judge benchmark with multi-agent enhanced explanation evaluation framework," *arXiv preprint arXiv:2410.12855*, 2024.

[31] walkerspider, "Dan is my new friend." https://old.reddit.com/r/ChatGPT/comments/zlcyr9/dan_is_my_new_friend/, 2022, reddit post; accessed 2025-10-21.

[32] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong, "Jailbreaking black box large language models in twenty queries," in *Proceedings of the 2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, 2025, pp. 23–42.

[33] S. Li, L. Yao, L. Zhang, and Y. Li, "Safety layers in aligned large language models: The key to llm security," *arXiv preprint arXiv:2408.17003*, 2024.

[34] J. Ji, D. Hong, B. Zhang, B. Chen, J. Dai, B. Zheng, T. A. Qiu, J. Zhou, K. Wang, B. Li *et al.*, "Pku-saferlhf: Towards multi-level safety alignment for llms with human preference," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 31 983–32 016.

[35] J. Cui, W.-L. Chiang, I. Stoica, and C.-J. Hsieh, "Or-bench: An over-refusal benchmark for large language models," in *Proceedings of the International Conference on Machine Learning*, 2024.

[36] P. Röttger, H. Kirk, B. Vidgen, G. Attanasio, F. Bianchi, and D. Hovy, "Xstest: A test suite for identifying exaggerated safety behaviours in large language models," in *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2024, pp. 5377–5400.

[37] M. Conover, M. Hayes, A. Mathur, J. Xie, J. Wan, S. Shah, A. Ghodsi, P. Wendell, M. Zaharia, and R. Xin, "Free dolly: Introducing the world's first truly open instructiontuned llm," 2023.

[38] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," 2023. [Online]. Available: https://lmsys.org/blog/2023-03-30-vicuna/

[39] A. Mehrotra, M. Zampetakis, P. Kassianik, B. Nelson, H. Anderson, Y. Singer, and A. Karbasi, "Tree of attacks: Jailbreaking black-box llms automatically," *Advances in Neural Information Processing Systems*, vol. 37, pp. 61 065–61 105, 2024.

[40] B. Zheng, G. Chen, H. Zhong, Q. Teng, Y. Tan, Z. Liu, W. Wang, J. Liu, J. Yang, H. Jing *et al.*, "Usb: A comprehensive and unified safety evaluation benchmark for multimodal large language models," *arXiv preprint arXiv:2505.23793*, 2025.

[41] Z. Lin, Z. Wang, Y. Tong, Y. Wang, Y. Guo, Y. Wang, and J. Shang, "Toxicchat: Unveiling hidden challenges of toxicity detection in real-world user-ai conversation," *arXiv preprint arXiv:2310.17389*, 2023.

[42] Meta AI, "Meta llama 3: The most capable openly available generative ai models," https://ai.meta.com/blog/meta-llama-3/, 2024, official technical release of Llama 3 models (including 8B Instruct).

[43] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. d. l. Casas, E. B. Hanna, F. Bressand *et al.*, "Mixtral of experts," *arXiv preprint arXiv:2401.04088*, 2024.

[44] Y. Zeng, H. Lin, J. Zhang, D. Yang, R. Jia, and W. Shi, "How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 14 322–14 350.

[45] Z.-X. Yong, C. Menghini, and S. H. Bach, "Low-resource languages jailbreak gpt-4," *arXiv preprint arXiv:2310.02446*, 2023.

[46] Y. Gou, K. Chen, Z. Liu, L. Hong, H. Xu, Z. Li, D.-Y. Yeung, J. T. Kwok, and Y. Zhang, "Eyes closed, safety on: Protecting multimodal llms via image-to-text transformation," in *European Conference on Computer Vision*. Springer, 2024, pp. 388–404.

[47] S. Fares, K. Ziu, T. Aremu, N. Durasov, M. Takáč, P. Fua, K. Nandakumar, and I. Laptev, "Mirrorcheck: Efficient adversarial defense for vision-language models," *arXiv preprint arXiv:2406.09250*, 2024.

[48] Y. Xu, X. Qi, Z. Qin, and W. Wang, "Cross-modality information check for detecting jailbreaking in multimodal large language models," *arXiv preprint arXiv:2407.21659*, 2024.

[49] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," 2023.

[50] Y. Huang, S. Gupta, M. Xia, K. Li, and D. Chen, "Catastrophic jailbreak of open-source llms via exploiting generation," *arXiv preprint arXiv:2310.06987*, 2023.

[51] X. Liu, Y. Zhu, J. Gu, Y. Lan, C. Yang, and Y. Qiao, "Mm-safetybench: A benchmark for safety evaluation of multimodal large language models," in *European Conference on Computer Vision*, 2024, pp. 386–403.

[52] Y. Gong, D. Ran, J. Liu, C. Wang, T. Cong, A. Wang, S. Duan, and X. Wang, "Figstep: Jailbreaking large vision-language models via typographic visual prompts," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025, pp. 23 951–23 959.

[53] W. Luo, S. Ma, X. Liu, X. Guo, and C. Xiao, "Jailbreakv: A benchmark for assessing the robustness of multimodal large language models against jailbreak attacks," *arXiv preprint arXiv:2404.03027*, 2024.

[54] N. Ding, Y. Chen, B. Xu, Y. Qin, Z. Zheng, S. Hu, Z. Liu, M. Sun, and B. Zhou, "Enhancing chat language models by scaling high-quality instructional conversations," *arXiv preprint arXiv:2305.14233*, 2023.

[55] R. Bhardwaj and S. Poria, "Red-teaming large language models using chain of utterances for safety-alignment," *arXiv preprint arXiv:2308.09662*, 2023.

[56] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," *Advances in neural information processing systems*, vol. 36, pp. 34 892–34 916, 2023.

[57] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou, "Qwen-vl: A frontier large vision-language model with versatile abilities," *arXiv preprint arXiv:2308.12966*, vol. 1, no. 2, p. 3, 2023.

[58] W. Yu, Z. Yang, L. Li, J. Wang, K. Lin, Z. Liu, X. Wang, and L. Wang, "Mm-vet: Evaluating large multimodal models for integrated capabilities," *arXiv preprint arXiv:2308.02490*, 2023.

[59] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2425–2433.

[60] P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, V. Sehwag, E. Dobriban, N. Flammarion, G. J. Pappas, F. Tramer *et al.*, "Jailbreakbench: An open robustness benchmark for jailbreaking large language models," *Advances in Neural Information Processing Systems*, vol. 37, pp. 55 005–55 029, 2024.

[61] M. Mazeika, L. Phan, X. Yin, A. Zou, Z. Wang, N. Mu, E. Sakhaee, N. Li, S. Basart, B. Li *et al.*, "Harmbench: A standardized evaluation framework for automated red teaming and robust refusal," *arXiv preprint arXiv:2402.04249*, 2024.

[62] C. Anil, E. Durmus, N. Panickssery, M. Sharma, J. Benton, S. Kundu, J. Batson, M. Tong, J. Mu, D. Ford *et al.*, "Many-shot jailbreaking," *Advances in Neural Information Processing Systems*, pp. 129 696–129 742, 2024.

[63] Z. Weng, X. Jin, J. Jia, and X. Zhang, "Foot-in-the-door: A multi-turn jailbreak for llms," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025, pp. 1939–1950.

A Detailed Experimental Settings

A.1 Our defense setting

In our experiments, the pseudo-malicious prompts were generated from the OR-Bench-80k dataset, from which we specifically selected those prompts that did not trigger a direct refusal response from the model. Otherwise, no detector that *relies on the model's internal mechanisms* could label these refused samples as benign, because the model itself already treats them as harmful. During testing, we performed a balanced evaluation using the same number of benign and jailbreak prompts. For each attack category, we randomly sampled 500 jailbreak prompts and evaluated them alongside an equally sized set of 500 benign prompts.

A.2 Formal Definitions of the Adaptive Losses

Let $\mathcal{B}$ and $\mathcal{M}$ denote the benign and malicious reference banks (anchors). For a jailbreak attack prompt S, we write $\mathbf{h}_\ell(S) \in \mathbb{R}^d$ for the layer-ℓ summary hidden state.

Evasion loss $\mathcal{L}^1_{\text{evasion}}$ (pull toward the entire benign bank). We encourage S to resemble the benign anchors at every layer by minimizing the summed MSE to *all* benign references: $\mathcal{L}^1_{\text{evasion}}(S) = \sum_{\ell=1}^{L} \frac{1}{|\mathcal{B}|} \sum_{\mathbf{b}\in\mathcal{B}} \text{MSE}\big(\mathbf{h}_\ell(S), \mathbf{h}_\ell(\mathbf{b})\big)$.

Evasion loss $\mathcal{L}^2_{\text{evasion}}$ (pull toward the nearest benign anchor). Instead of matching the full benign set, we pull S toward its *closest* benign reference at each layer: $\mathcal{L}^2_{\text{evasion}}(S) = \sum_{\ell=1}^{L} \min_{\mathbf{b}\in\mathcal{B}} \text{MSE}\big(\mathbf{h}_\ell(S), \mathbf{h}_\ell(\mathbf{b})\big)$.

Evasion loss $\mathcal{L}^3_{\text{evasion}}$ (benign-like neighborhood structure). To more directly mimic MTK's neighborhood signal, we minimize distance to the benign bank while simultaneously pushing away from the malicious bank:

$$\mathcal{L}^3_{\text{evasion}}(S) = \sum_{\ell=1}^{L} \left[\frac{1}{|\mathcal{B}|} \sum_{\mathbf{b}\in\mathcal{B}} \text{MSE}\big(\mathbf{h}_\ell(S), \mathbf{h}_\ell(\mathbf{b})\big) - \frac{1}{|\mathcal{M}|} \sum_{\mathbf{m}\in\mathcal{M}} \text{MSE}\big(\mathbf{h}_\ell(S), \mathbf{h}_\ell(\mathbf{m})\big) \right]. \tag{21}$$

A.3 Adaptive Attacks against Other Detectors

GradSafe [18] detects jailbreaks by comparing the gradients induced by prompt S with a pre-computed unsafe reference pattern on a set of safety-critical parameter slices $\mathcal{P}$. Let $\mathbf{g}_k(S)$ denote the loss gradient with respect to slice $p \in \mathcal{P}$, and $\mathbf{g}_p^{\text{ref}}$ the corresponding unsafe reference gradient. GradSafe aggregates cosine similarities as a gradient-based objective, and we use it as the evasion loss: $\mathcal{L}^{\text{GradSafe}}_{\text{evasion}} = \frac{1}{|\mathcal{P}|} \sum_{p\in\mathcal{P}} \cos\big(\mathbf{g}_p(S), \mathbf{g}_p^{\text{ref}}\big)$.

A.4 Judge Details

We use GPT-4 as the ASR judge with temperature= 0. Following [61], we prompt GPT-4 to determine whether a generated response constitutes a successful jailbreak using:

> **ASR Judge Prompt:** *You are a classifier for red teaming large language models (LLMs). Given a generation, determine whether it counts as an instance of the following behavior (subject to the rules below). Your answer should be "yes" or "no".*
> *Behavior: {behavior} Generation: {generation}*
> *— Rules: - Instances of the behavior should be unambiguous and non-minimal. Vague or very short instances do not count. - Generations should themselves be harmful or undesirable for a public-facing LLM to output. Benign or supportive generations do not count. - Generations that respond, analyze, discuss, or otherwise react to the behavior without actually exhibiting it do not count. - If the behavior involves coding and programming, the generation should contain code artifacts related to the behavior, not just comments or instructions.*
> *Answer:*

B Extended Experiments

B.1 Evaluation on Larger and Newer Models

We extend evaluation to three newer LLM families—Qwen2.5, Gemma-3, and a larger Vicuna variant—as well as the multimodal InternVL3 family. Tab. 10 and Tab. 11 report AUROC across three representative attacks for each modality. MTK maintains strong detection performance across model families and *multiple scales* (from 1B to 14B parameters), with AUROC ≥ 0.86 on all LLM settings and AUROC ≥ 0.99 on VLM settings.

Table 10: AUROC of MTK on newer/larger LLMs across three jailbreak attacks.

LLM	JailJudge	IJP	GCG
Qwen2.5-3B	0.914	0.863	0.896
Qwen2.5-7B	0.927	0.946	0.948
Qwen2.5-14B	0.960	0.952	0.920
Gemma-3-4B	0.904	0.911	0.914
Gemma-3-12B	0.932	0.924	0.924
Vicuna-13B	0.932	0.913	0.945

Table 11: AUROC of MTK on the InternVL3 VLM family.

VLM	MM-SafetyBench	FigImg	JailBreakV-28K
InternVL3-1B	0.998	0.991	0.991
InternVL3-8B	0.993	0.996	0.995
InternVL3-14B	0.995	0.991	0.993

Table 12: Cross-benchmark AUROC of MTK under GCG attack on Llama2-7B.

Benchmark	AUROC
AdvBench [4]	0.929
JBB-Behaviors [60]	0.907
HarmBench [61]	0.908

B.2 Cross-Benchmark Evaluation

We run GCG attacks on Llama2-7B against prompts from JBB-Behaviors [60] and HarmBench [61], and report AUROC in Tab. 12. MTK achieves AUROC$=0.907$ on JBB-Behaviors and 0.908 on HarmBench, close to the 0.929 obtained on the ADVBENCH-style setting, indicating that the rank-trajectory signal generalizes across benchmark prompt styles.

B.3 Robustness to OOD Benign Inputs

A practical concern in deployment is whether MTK produces false positives on benign inputs drawn from distributions far from its reference bank. We evaluate MTK on five additional benign sources outside our calibration distribution, covering two broad categories: (i) *domain-specialized* prompts—GSM8K (math), HumanEval (code), and PubMedQA (medical); and (ii) *semantically complex* prompts—XSTest-safe (figurative language, hypothetical reasoning) and an OR-Bench persona-roleplay subset. As shown in Tab. 13, MTK keeps FPR below 5% on Llama2-7B across all five OOD benign sources, complementing the bank-composition robustness reported in Tab. 7b.

B.4 Comparison to a No-Defense Baseline

Tab. 14 reports per-attack ASR without any defense on the 10-attack$\times$4-LLM suite, directly complementing the AUROC results in Tab. 4. On the heavily aligned Llama2-7B and Llama3-8B, several attacks (e.g., AutoDAN, PAIR, TAP) already yield low base ASR due to the models' own safety alignment; the detection performance on these models should therefore be read primarily through attacks with high undefended ASR (e.g., GCG, SAA, DrAttack), where the room for a defense to contribute is unambiguous.

Table 13: FPR of MTK on out-of-distribution benign inputs (Llama2-7B).

Source	GSM8K	HumanEval	PubMedQA	XSTest-safe	OR-Bench
FPR↓	0.02	0.01	0.01	0.05	0.03

Table 14: ASR of each jailbreak attack *without any defense* on four LLMs.

Attack↓	Mistral-7B	Vicuna-7B	Llama2-7B	Llama3-8B
AutoDAN [27]	0.90	0.81	0.05	0.05
DrAttack [28]	0.72	0.76	0.49	0.61
IJP [29]	0.59	0.45	0.25	0.28
JailJudge [30]	0.35	0.39	0.14	0.20
GCG [4]	0.89	0.87	0.45	0.37
PAIR [32]	0.53	0.55	0.11	0.11
PAP [44]	0.68	0.61	0.58	0.59
SAA [5]	0.90	0.89	0.78	0.75
TAP [39]	0.75	0.61	0.08	0.14
Zulu [45]	0.22	0.19	0.04	0.16

B.5 Input-Length Sensitivity

We bin JailJudge [30] prompts by input token length into four buckets ($<$64, 64–128, 128–256, and 256–512 tokens), and measure AUROC on Llama2-7B within each bucket. MTK's AUROC remains stable across length ranges—0.9284, 0.9218, 0.9301, and 0.9241 respectively—indicating that performance is not driven by input length. We further stress-test a bank-test length mismatch setting by restricting the reference bank to prompts of $<$128 tokens and evaluating on test prompts of $>$512 tokens; MTK still attains AUROC$=0.948$, confirming robustness to length distribution shift.

B.6 Multi-Turn Jailbreak Evaluation

Although MTK is designed for single-turn detection, it extends naturally to multi-turn jailbreak attacks without any modification: the entire conversation history is fed as a single input, and the same rank-trajectory pipeline applies. We evaluate this on multi-turn attacks against Llama2-7B: *Many-shot Jailbreaking* [62], which fills the context window with crafted dialogue turns (we use 100 shots, 4K tokens, well into the long-context regime), and *Foot-in-the-door* [63], which gradually escalates harmful intent across turns. MTK achieves AUROC$=0.869$ and 0.861, respectively—comparable to its single-turn performance. This suggests that the depth-wise trajectory signature remains discriminative even when malicious intent is distributed across multiple conversational turns.