AlignTree: Efficient Defense Against LLM Jailbreak Attacks

Gil Goren, Shahar Katz, Lior Wolf

Blavatnik School of Computer Science, Tel Aviv University
{gilgoren@mail, shaharkatz3@mail, wolf@cs}.tau.ac.il

Abstract

Large Language Models (LLMs) are vulnerable to adversarial attacks that bypass safety guidelines and generate harmful content. Mitigating these vulnerabilities requires defense mechanisms that are both robust and computationally efficient. However, existing approaches either incur high computational costs or rely on lightweight defenses that can be easily circumvented, rendering them impractical for real-world LLM-based systems. In this work, we introduce the AlignTree defense, which enhances model alignment while maintaining minimal computational overhead. AlignTree monitors LLM activations during generation and detects misaligned behavior using an efficient random forest classifier. This classifier operates on two signals: (i) the refusal direction—a linear representation that activates on misaligned prompts, and (ii) an SVM-based signal that captures non-linear features associated with harmful content. Unlike previous methods, AlignTree does not require additional prompts or auxiliary guard models. Through extensive experiments, we demonstrate the efficiency and robustness of AlignTree across multiple LLMs and benchmarks.

Introduction

LLMs have become integral to numerous applications across various domains, making their security a pressing concern. However, recent research has highlighted vulnerabilities such as using LLMs to generate phishing emails, malicious code, hate speech, and inadvertently exposing sensitive information (Wei, Haghtalab, and Steinhardt 2023; Gupta et al. 2023). Given the substantial incentives for adversaries to circumvent security measures and obtain responses to otherwise restricted queries, often referred to as "jailbreak" attacks, research on security alignment has gained momentum. Early efforts focused on training-time alignment (Glaese et al. 2022; Ouyang et al. 2022), where harmful prompts were introduced during training to adjust the model's behavior to refuse inappropriate requests. Another method involved aligning the model through system prompts, explicitly instructing it to reject harmful commands (Bai et al. 2022). While inducing no computational overhead during models' inference, multiple studies have demonstrated that these approaches are insufficient on their own, as simple prompt engineering techniques can effectively circumvent them (Wei, Haghtalab, and Steinhardt 2023; Qiu et al. 2023; Liu et al. 2024b). Furthermore, advanced adversarial techniques, such as suffix-based jailbreak attacks (Zou et al. 2023) and automatic LLM-assisted jailbreak prompt generation (Mehrotra et al. 2024; Chao et al. 2024b), continue to expose weaknesses in existing defenses.

To address these threats, LLM security research has evolved to include external defenses across all stages of the generation pipeline (Yao et al. 2024). We can categorize these defenses into three categories: (1) Pre-processing, which focuses on filtering harmful inputs before they are processed by the model (Jain et al. 2023; Zeng et al. 2024a), but at the cost of additional inference time, causing a delay in user-interface systems such as chat-based LLMs. (2) In-process defenses, which monitor and regulate activations and internal representations during inference (Xu et al. 2024; Zhang et al. 2025; Dong et al. 2025). This approach has relatively low computational overhead but is based on a limited number of identified features from the activation space, mostly binary ones, which makes them less robust to a wide range of attacks. (3) Post-processing, which filters and modifies outputs after generation (Phute et al. 2024; Zeng et al. 2024b) can identify not only harmful inputs but also misaligned models' output; however, it requires processing long segments of text and delays the LLMs' responses. In addition, different defense methods from all groups are built on additional models, mostly LLMs. These defenses not only increase inference time but also the compute requirement from a system that needs to execute an LLM such as Llamaguard (Llama Team and AI @ Meta 2024) as its external defenses.

Therefore, defending against sophisticated attacks remains a challenge, mostly in real-time deployed systems. To solve these challenges, we propose AlignTree, a lightweight and computationally efficient classifier that enhances the alignment of LLMs and assists in distinguishing between harmful and harmless prompts. Relying solely on base model activations, AlignTree achieves state-of-the-art (SOTA) performance in Attack success rate (ASR) and efficiency, without increasing the refusal rate. To this end, we rely on two complementary sources of signal: (i) activations projected onto the linear refusal direction following Arditi et al. (2024), and (ii) motivated by prior work suggesting that refusal behavior in LLMs is not entirely linear (Wollschläger et al. 2025; Hildebrandt et al. 2025), we train non-linear support vector

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Method	ASR ↓	Overhead	
		Additional LLM	Additional Inference
Baseline model	High	No	0
Llama Guard (Llama Team and AI @ Meta 2024)	Low	Yes	2
AutoDefense (Zeng et al. 2024b)	Low	Yes	20
SmoothLLM (Robey et al. 2024)	Medium	No	10
SelfDefense (Phute et al. 2024)	Medium	No	2
PerplexityDefense (Jain et al. 2023)	High	No	0
AlignTree (**Ours**)	Low	No	0

Table 1: LLMs jailbreak defense methods and their computational overheads.

machines (SVMs) with radial basis function (RBF) across tokens and layers' hidden state.

The two types of features are then used to train a Random Forest classifier, which assigns confidence scores reflecting the harmfulness of a prompt. The main advantage of the resulting classifier (AlignTree) is that, in contrast to prior methods, it does not rely on fine-tuning, additional inference passes, or auxiliary models. Instead, it leverages the LLM's internal activations to enhance model alignment through targeted probing.

We extensively evaluate AlignTree across nine different LLMs and multiple widespread harmfulness benchmarks. AlignTree outperforms existing state-of-the-art defenses by achieving a lower attack success rate (ASR), minimizing unnecessary refusal of harmless instructions, and significantly reducing computational overhead. By addressing the efficiency gaps overlooked in prior work and enabling a more complex defense strategy using confidence scores, AlignTree paves the way for scalable, real-time LLM alignment.

Related Work

Recent advancements in the field of LLMs have significantly enhanced the understanding of their vulnerabilities, defense mechanisms, and security alignment strategies. Yao et al. (2024) provided a comprehensive taxonomy of threats and corresponding defenses. LLM inference defenses are often categorized into three stages: **Pre-Process**, **In-Process**, and **Post-Process**, based on when the defense mechanisms are applied during the model's inference pipeline.

Pre-Process defenses operate on prompts before they are passed to the LLM for response generation. Jain et al. (2023) evaluated the effectiveness of different defenses, applying each defense independently to assess its impact. Perplexity filters, which use the model inference to compute the perplexity score with regard to its input, and potentially output, are designed to identify and filter out gibberish input, such as GCG (Greedy Coordinate Gradient) (Zou et al. 2023).

The use of LLM-as-a-judge has become a state-of-the-art approach (Gu et al. 2025). Security-aligned models, which are usually considered as small LLMs, such as LlamaGuard (Llama Team and AI @ Meta 2024) and ShieldGemma (Zeng et al. 2024a), have proven effective in detecting and assessing harmful inputs. However, this approach is computationally heavy, requiring storage and execution of an additional LLM and executing additional forward passes.

In-Process defenses analyze LLM intermediate results such as neuron activation and hidden states. Arditi et al. (2024) explored the existence of a refusal direction in hidden states, a single geometric space in the activation space, that can be leveraged to detect and block harmful prompts. Building on this, Zhang et al. (2025) utilized the refusal direction to identify harmful prompts and then reinforced the awareness of the LLM for the toxic concept via activation addition with the refusal direction. Similarly, Dong et al. (2025) trained a binary classifier on refusal direction activations to identify harmful prompts during response generation at every generated token, then steering the model toward producing harmless responses. Early work primarily treated refusal as a linear phenomenon, using linear directions to fine-tune models or guide their outputs. However, recent research has shown that refusal behavior in LLMs is not entirely linear (Hildebrandt et al. 2025; Wollschläger et al. 2025), suggesting that relying solely on linear signals may oversimplify the underlying dynamics and potentially degrade generation quality. In this work, we show that incorporating additional non-linear refusal signals can improve robustness and better mitigate harmful completions.

Other defenses, such as SmoothLLM (Robey et al. 2024), utilize a perturbation technique that copies the prompt and applies small changes to each copy, then generates multiple responses. Using majority voting, the prompt is classified as malicious or not. Similarly, Kumar et al. (2025) proposed the erase-and-check approach, which involves generating multiple copies of a prompt and randomly removing tokens. The model generates multiple responses, and majority voting is used to determine whether the prompt is malicious. Li et al. (2023) proposed RAIN, a method that enables models to rewind responses during generation if harmful content is detected. These kinds of approaches do not require additional LLM but suffer from a big latency caused by rerunning the base LLM multiple times, especially when considering the fact that in many systems, the ratio of harmful-harmless prompts is low

Post-Process defenses evaluate the LLM's generated response to harmful content. Phute et al. (2024) demonstrated how an LLM can act as a judge to review its responses for potential harm. Zeng et al. (2024b) built on this idea by employing a team of LLM agents that work together through dialogue to evaluate whether a prompt is harmful. Chen, Paliwal, and Yan (2023) implemented a multi-metric evaluation

system where several LLM judges calculate toxicity and quality metrics before reaching a consensus via majority voting. These approaches are as strong as the LLM they utilize for the classification of prompts' harmfulness, and require additional compute to host and run. In particular, systems that want to use multi-judges based methods, such as Chen, Paliwal, and Yan (2023); Zeng et al. (2024b), dramatically increase the computational requirement for deployed systems.

Table 1 provides an overview of several well-known defense methods and their associated overheads. Unlike other approaches, our method achieves state-of-the-art ASR results without introducing additional inference steps or requiring auxiliary models. In contrast, LlamaGuard and AutoDefense necessitate deploying extra models, leading to increased computational overhead. SmoothLLM and AutoDefense also depend on a large number of prompt variations, which is impractical in real-world scenarios. Self-Defense doubles the inference cost yet still fails to achieve low ASR in most cases. While PerplexityDefense is highly efficient, its simplicity limits its effectiveness against more sophisticated attacks.

Method

In this section, we introduce AlignTree, an efficient classifier for detecting harmful responses. AlignTree relies on two complementary signals: (i) scalar features derived from projecting activations onto the model's refusal direction, and (ii) non-linear features extracted by SVMs trained to identify malicious patterns in LLM activations. These signals are then combined and fed into a Random Forest classifier for the final prediction.

Obtaining Refusal Activations

Following Arditi et al. (2024), we extract a single linear refusal direction r^* that captures the model's internal representation of refusal. After determining r^*, we project hidden states onto this vector to obtain scalar Refusal Activations, which serve as one of the inputs to our classifier.

Difference-in-means. To detect the single refusal direction, we begin by constructing a set of **candidate** refusal directions using the difference-in-means method. Let D_{harmful} and D_{harmless} be the sets of harmful and harmless prompts, respectively. For each prompt t in these sets, we extract the hidden activation $x_i^{(l)}(t)$ at token position $i \in \mathcal{I}$ and layer $l \in [L]$ of the LLM, where L is the total number of layers. We then compute the average activation vectors for each token position and layer over the training subsets $D_{\text{harmful}}^{(\text{train})}$ and $D_{\text{harmless}}^{(\text{train})}$:

$$\mu_i^{(l)} = \frac{1}{|D_{\text{harmful}}^{(\text{train})}|} \sum_{t \in D_{\text{harmful}}^{(\text{train})}} x_i^{(l)}(t), \tag{1}$$

$$v_i^{(l)} = \frac{1}{|D_{\text{harmless}}^{(\text{train})}|} \sum_{t \in D_{\text{harmless}}^{(\text{train})}} x_i^{(l)}(t), \tag{2}$$

where $x_i^{(l)}(t)$ denotes the hidden activation at position i and layer l for prompt t. The difference-in-means vectors are then defined as:

$$r_i^{(l)} = \mu_i^{(l)} - v_i^{(l)}. \tag{3}$$

This yields a set of candidate directions $\{r_i^{(l)}\}$ across layers and token positions.

Selecting a Single Vector. We evaluate each candidate vector on held-out validation sets $D_{\text{harmful}}^{(\text{val})}$ and $D_{\text{harmless}}^{(\text{val})}$, following the procedure of Arditi et al. (2024). Each vector is assessed based on its ability to reduce refusal behavior when ablated, and to induce refusal behavior when added, while otherwise preserving the model's general functionality. The vector with the greatest effect under these criteria is selected as the single refusal direction, denoted r^*.

Refusal Activations. To measure the alignment of a hidden state $h \in \mathbb{R}^{d_{\text{model}}}$ with the refusal direction, we compute its projection onto r^*:

$$\text{proj}_{r^*}(h) = \frac{h \cdot r^*}{\|r^*\|} \in \mathbb{R} \tag{4}$$

This scalar value, referred to as the Refusal Activation, measures the degree to which the hidden state aligns with the direction associated with refusal behavior. We collect activations from the final token position across multiple layers, resulting in a set of scalar features that together constitute the Refusal Activations.

Extracting Non-linear Malicious Signals

While a single linear refusal direction captures some aspects of harmful prompt detection, prior work (Hildebrandt et al. 2025; Wollschläger et al. 2025) suggests that the geometry of refusal in LLMs may be inherently non-linear. To capture richer indicators of harmfulness, we train a large set of Support Vector Machines (SVMs) with radial basis function (RBF) kernels.

For each layer of the model, $l \in [L]$, and each token position i among the first 3 and last 5 tokens of the prompt, we train a separate SVM classifier $\text{SVM}_i^{(l)}$. Each classifier $\text{SVM}_i^{(l)}$ is trained to distinguish between harmful and harmless prompts using the hidden activations $x_i^{(l)}(t) \in \mathbb{R}^{d_{\text{model}}}$, taken from a labeled training set.

In total, we train $8 \times L$ SVMs, one for each combination of the 8 selected token positions and all L layers. We used the same training set for both model training and Refusal Activation extraction. After training, we evaluate all $8L$ SVMs on a held-out validation set based on accuracy. We then select the top-performing $L/2$ SVMs to use in our classifier.

Probabilistic Feature Extraction. For each SVM, we use 5-fold cross-validation on the designated training set to generate out-of-fold harmfulness probabilities. To obtain probabilities from the raw SVM, we follow the algorithm by Platt (2000), which fits a sigmoid to map decision values to probabilities. This results in a single confidence score per training example for each SVM, enabling us to represent its non-linear signal as a normalized scalar feature used by the final classifier. We denote this calibrated output as $P_{\text{harmful}}(x_i^{(l)})$, representing the harmfulness probability predicted by the $\text{SVM}_l^{(l)}$ associated with feature i at layer l.

Let $\mathcal{S}$ denote the set of $L/2$ selected classifiers. For a new prompt t, we compute the calibrated harmfulness probabilities of each SVM in $\mathcal{S}$, resulting in a feature vector of confidence scores that encodes non-linear harmfulness signals:

$$\text{SVMFeatures}(t) = \left[P_{\text{harmful}}(x_i^{(l)}(t))\right]_{(i,l)\in\mathcal{S}}. \quad (5)$$

AlignTree

We train a Random Forest classifier using two types of input signals for each prompt t: (i) Refusal activations, computed by projecting the final token activations from each layer $l \in [1..L]$ onto the selected refusal direction r^*; and (ii) Harmfulness probability estimates, generated by a selected set $\mathcal{S}$ of nonlinear SVM classifiers.

The complete input feature vector F is constructed by concatenating these components:

$$\text{F}(t) = \left[\text{proj}_{r^*}(x_{-1}^{(l)}(t))\right]_{l=1}^{L} \oplus \left[P_{\text{harmful}}(x_i^{(l)}(t))\right]_{(i,l)\in\mathcal{S}} \quad (6)$$

where $x_{-1}^{(l)}(t)$ denotes the activation at the final token position in layer l, $x_i^{(l)}(t)$ is the activation at token position i in layer l, and $\oplus$ denotes vector concatenation.

To ensure computational efficiency, we employ a lightweight Random Forest model consisting of a small number of shallow decision trees, trained on a curated dataset.

Threshold selection We define a harmfulness threshold τ to decide whether a prompt is accepted or blocked. Prompts with predicted harmfulness below τ are passed to the LLM, while those above are rejected as malicious. To avoid excessive refusals while minimizing missed harmful prompts, τ is selected to maximize precision while balancing recall. This trade-off is optimized using the following F_β score:

$$F_\beta = (1+\beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}} \quad (7)$$

To emphasize precision, we set $\beta = 0.2$. For each model, we select the final threshold as the one that maximizes the F_β score on the validation set. See Figure 1 for the generalized F-score curves and the selected threshold for Qwen2.5-7B-Instruct. Additional experiments validating the threshold selection are provided in the Appendix.

Experiments

Refusal and SVM Datasets. In our experiments, we compile two datasets for training the refusal vectors and SVMs: (i) $D_{harmful}$: Prompts labeled as harmful, drawn from Advbench (Zou et al. 2023), MaliciousInstruct (Huang et al. 2023), TDC2023 (Mazeika et al. 2023), StrongReject (Souly et al. 2024) and HarmBench (Mazeika et al. 2024). (ii) $D_{harmless}$: a collection of benign prompts sampled from ALPACA (Taori et al. 2023). Additionally, we included the white-box targeted attack from Andriushchenko, Croce, and Flammarion (2025) to further evaluate AlignTree's robustness, with results detailed in the Appendix.

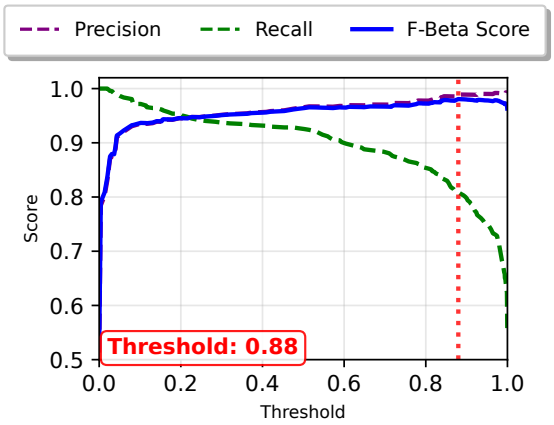

Figure 1: Qwen2.5-7B-Instruct threshold selection based on the generalized F_β score. $\tau = 0.88$.

Random Forest Datasets. In addition, we construct an additional dataset for training the Random Forest classifier, combining harmful datasets: Jailbreakbench (Chao et al. 2024a), PAIR (Chao et al. 2024b), and AutoDAN (Liu et al. 2024a) attack samples sourced from Zhang et al. (2025), along with the harmless datasets ALPACA Taori et al. (2023) and XSTest (Röttger et al. 2024). Samples from ALPACA are present in both our Random Forest training set and the Refusal and SVM datasets; however, we made sure that there is no overlap between these datasets.

AlignTree Hyperparameters. The Random Forest classifier uses the hyperparameters $n\_estimators = 50$, $max\_depth = 6$, and $min\_samples\_split = 5$. A grid search exploring hyperparameter sensitivity is detailed in the Appendix. The SVMs employ an RBF kernel; the use of linear kernels is explored in our Ablation Study. Feature importance for each signal is detailed in the Appendix.

Evaluation datasets. For evaluating adversarial robustness, we primarily leveraged **harmful datasets** from Garak (Derczynski et al. 2024), including: PromptInject, an implementation of Perez and Ribeiro (2022), designed to disrupt model alignment using directives such as *"IGNORE PREVIOUS INSTRUCTIONS"*, and MalwareGen, a collection of prompts aimed at eliciting malware-generating code from the model. Additionally, we incorporated two attack datasets from JBShield (Zhang et al. 2025) that bypassed the safeguards of Llama-3.1-8B: PAIR, an algorithm that crafts semantic jailbreaks using only black-box access to an LLM (Chao et al. 2024b), and AutoDAN, a dataset of adversarial attacks generated via genetic algorithms, requiring only black-box access to an LLM (Liu et al. 2024a). Samples from PAIR and AutoDAN are included in both our Random Forest training dataset and evaluation datasets; however, we ensured that there is no overlap between them.

To ensure that AlignTree does not degrade performance or lead to excessive refusals of **harmless** responses, we evaluated it on four benign, commonsense reasoning datasets:

PIQA (Bisk et al. 2020) -– assessing physical commonsense reasoning; ARC-Challenge (Clark et al. 2018) — test-

Model	Strategy	ASR ↓				Refusal ↓			
		MalwareGen	PromptInject	PAIR	AutoDAN	PIQA	OpenBookQA	SIQA	ARC
Qwen2.5	Baseline	91.0	50.0	51.0	48.0	**0**	**0**	**0**	**0**
-0.5B	AutoDefense	5.0	**0**	13.0	**0**	**0**	6.0	3.0	8.0
-Instruct	SelfDefense-Input	43.0	13.0	8.0	17.0	80.0	35.0	33.0	46.0
	SelfDefense	42.0	16.0	11.0	13.0	72.0	37.0	37.0	41.0
	PerplexityDefense	84.0	50.0	50.0	47.0	**0**	**0**	**0**	**0**
	SmoothLLM	77.0	43.0	49.0	44.0	**0**	**0**	**0**	**0**
	AlignTree (**Ours**)	**4.0**	41.0	**6.0**	**0**	**0**	**0**	**0**	**0**
Llama	Baseline	9.0	43.0	14.0	**0**	2.0	**0**	**5.0**	**0**
-3.1-8B	AutoDefense	**5.0**	**0**	16.0	**0**	2.0	1.0	7.0	4.0
-Instruct	SelfDefense-Input	8.0	32.0	**8.0**	**0**	55.0	47.0	30.0	52.0
	SelfDefense	8.0	28.0	**8.0**	**0**	51.0	55.0	34.0	49.0
	PerplexityDefense	8.0	42.0	15.0	**0**	2.0	**0**	**5.0**	**0**
	SmoothLLM	8.0	37.0	13.0	**0**	2.0	**0**	**5.0**	**0**
	AlignTree (**Ours**)	**5.0**	18.0	9.0	**0**	**1.0**	**0**	**5.0**	**0**
gemma	Baseline	24.0	50.0	36.0	6.0	**0**	**0**	**0**	**0**
-3-12b	AutoDefense	**7.0**	**5.0**	19.0	**1.0**	**0**	7.0	**0**	2.0
-it	SelfDefense-Input	23.0	35.0	28.0	3.0	1.0	**0**	1.0	**0**
	SelfDefense	18.0	**5.0**	33.0	4.0	2.0	17.0	58.0	11.0
	PerplexityDefense	16.0	52.0	35.0	5.0	**0**	**0**	**0**	**0**
	SmoothLLM	25.0	55.0	37.0	7.0	**0**	**0**	**0**	**0**
	AlignTree (**Ours**)	10.0	40.0	**10.0**	**1.0**	**0**	**0**	**0**	**0**

Table 2: Attack Success Rate (ASR) for each harmful dataset and model, as well as Refusal rates for harmless datasets. The full results (nine LLMs from three families) are in the Appendix and show similar patterns.

ing scientific reasoning; OpenBookQA (Mihaylov et al. 2018) — evaluating advanced question answering; and SIQA (Social Interaction QA) (Sap et al. 2019) –- measuring social commonsense understanding.

Baselines To assess the effectiveness of our efficient classifier, we compare it against eight defense strategies, including several state-of-the-art methods. We focus on methods that do not require deploying auxiliary models. These strategies include: (i) Baseline which relies solely on the model's native alignment; (ii) AutoDefense (Zeng et al. 2024b), (iii) SmoothLLM (Robey et al. 2024), (iv) SelfDefenseInput and (v) SelfDefense (Phute et al. 2024), which query the main model on the harmfulness of the prompt and response, respectively; (vi) PerplexityDefense (Jain et al. 2023). For these defenses, we chose the hyperparameters per their original papers and are described in the Appendix, as well as additional implementation details.

We evaluate three families of instruction-tuned LLMs: Qwen2.5 (0.5B, 3B, 7B) (Team 2024), Llama3 (1B, 3B, 8B) (Grattafiori et al. 2024), and Gemma3 (1B, 4B, 12B) (Gemma Team et al. 2025). In this section, we present results for a single model per family to maintain clarity, selecting different sizes to ensure diversity: Qwen2.5-0.5B-Instruct, Llama-3.1-8B-Instruct, and Gemma-3-12B-It. Complete results for all nine models are included in the Appendix and exhibit the same patterns as those presented in the main text.

Following prior work (Mazeika et al. 2024; Arditi et al. 2024; Zhang et al. 2025), we adopt the Attack Success Rate (ASR) metric, which measures the proportion of harmful completions that bypass refusal mechanisms. To evaluate both harmfulness and refusals, we rely on ChatGPT-4o (OpenAI 2024), using its responses and a set of refusal-related keywords. In addition to adversarial evaluation, we conduct a complementary experiment on benign datasets to measure over-refusal and execution time.

First, we assess the trade-off between **attack success rate (ASR) and refusal behavior** using both harmful and harmless prompt datasets. We evaluate each defense's ability to block harmful prompts while minimizing refusals of harmless ones, ensuring practical real-world applicability. Table 2 presents ASR and refusal results for representative model families and sizes; full results in the Appendix exhibit the same trends. AlignTree demonstrates robust performance across all evaluated models and datasets, achieving substantial reductions in ASR compared to the no-defense baseline while maintaining the lowest refusal rates. In all tested scenarios, it delivers state-of-the-art refusal performance, showing the lowest rates across datasets and model families.

Across most datasets, AlignTree matches or exceeds the ASR performance of existing defenses, including more complex approaches such as AutoDefense and SelfDefense. For instance, on Gemma-3-12B, it attains the lowest ASR for the PAIR dataset; on Qwen2.5-0.5B, it records the lowest ASR among MalwareGen, PAIR, and AutoDAN. It also performs competitively on Llama-3.1-8B, closely matching or surpassing other defenses across all datasets. However, there are cases where AlignTree shows higher ASR than other defenses, for example, on PromptInject with Qwen2.5-0.5B and Gemma-3-12B, SelfDefense and AutoDefense achieve lower ASR at the cost of higher refusal rates, which frequently block benign inputs and cause over-refusal behavior. In contrast, AlignTree provides strong protection while minimizing

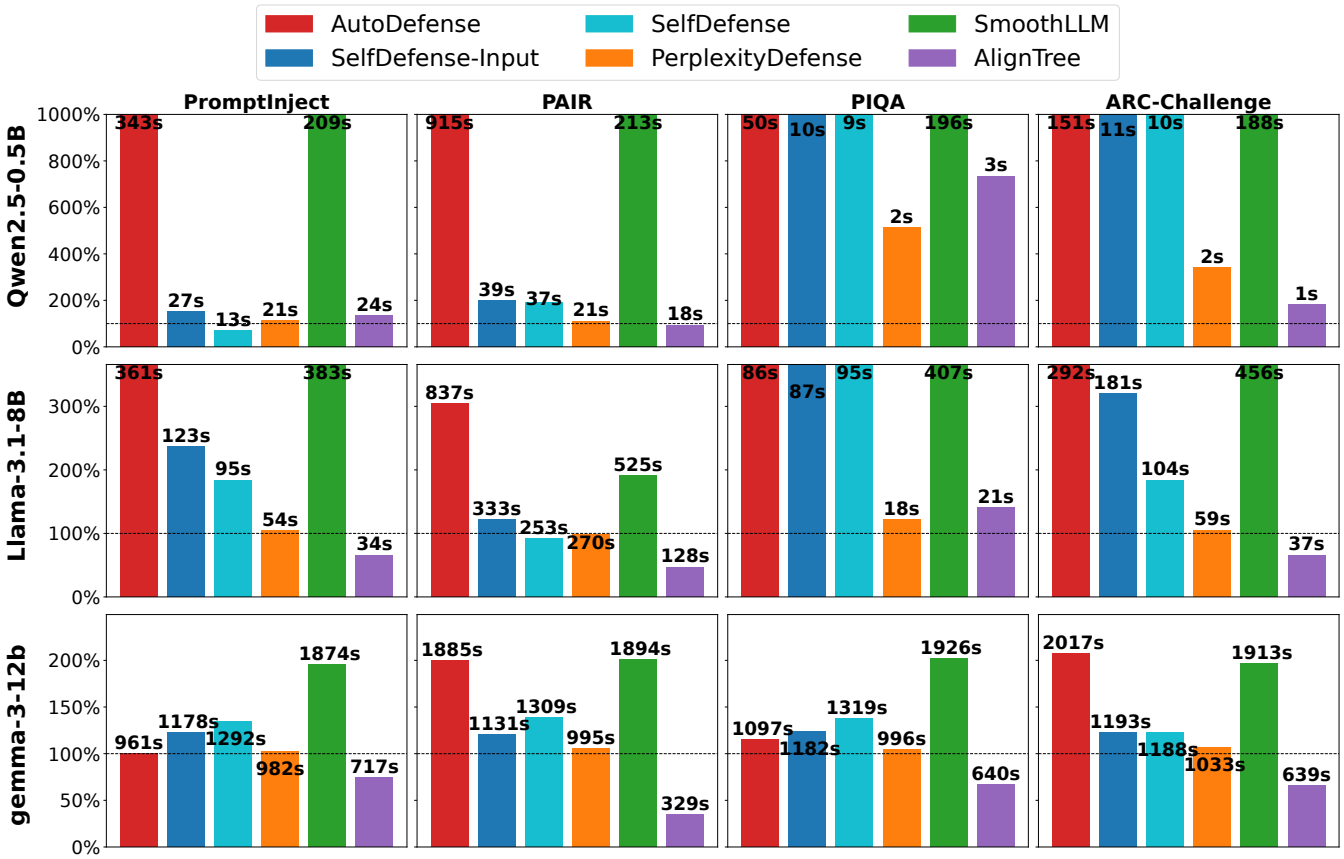

Figure 2: Execution time per method relative to running the baseline LM (dashed line) (Lower is better). Charts are capped at 1000% of baseline time. The full results (nine LLMs from three families) are in the Appendix and show similar patterns.

unnecessary refusals.

Secondly, we evaluate **defenses' efficiency**, defining execution time as the total duration required to process 100 prompts from a given task. Figure 2 shows that for most models and datasets, AlignTree achieves the lowest execution time. The only exceptions are a few cases where PerplexityDefense is marginally faster; however, PerplexityDefense incurs a higher ASR. Notably, AlignTree's execution time remains highly competitive with the baseline methods, introducing only negligible overhead compared to other defenses.

In summary, AlignTree delivers the strongest overall performance by combining substantial ASR reductions with the lowest refusal rates, while also achieving state-of-the-art execution times across most models and datasets. This balance of robustness, low refusal, and computational efficiency makes AlignTree a dependable and practical defense across diverse models and threat environments.

Ablation Study

In this experiment, we evaluate the contribution of each signal by independently training separate Random Forest classifiers under four configurations. Our goal is to verify that combining these signals yields superior performance compared to any individual component: (i) RefusalClassifier — trained solely on the activations from a single refusal vector without SVM signals; (ii) SVMClassifier — trained only on non-linear SVM decision boundaries without incorporating refusal activations; (iii) MultiRefusalsClassifier — leveraging activations from multiple top-performing refusal vectors across layers and tokens; and (iv) AlignTreeLinear — using a single refusal vector with SVMs constrained to linear decision boundaries.

The complete AlignTree method delivers the most consistent performance across all evaluated models and datasets, striking a strong balance between low ASR and efficient execution time, without increasing refusal rates. For instance, on Qwen2.5-0.5B, AlignTree achieves the lowest ASR across all datasets while maintaining a competitive runtime. On Llama-3.1-8B, it attains the lowest ASR on PromptInject and closely matches the top results on MalwareGen. The primary exception is Gemma-3-12b, where AlignTreeLinear outperforms both AlignTree and all other defenses in terms of ASR. Despite this isolated advantage, AlignTreeLinear exhibits significantly worse performance in other settings, such as an ASR of 61.0 on MalwareGen for Qwen2.5-0.5B, compared to just 4.0 ASR with AlignTree. While it benefits from slightly faster execution, its reliance on linear classifiers limits expressiveness and leads to inconsistent results. The

Model	Strategy	MalwareGen		PromptInject		PIQA		ARC-Challenge	
		ASR ↓	Time ↓	ASR ↓	Time ↓	Refusal ↓	Time ↓	Refusal ↓	Time ↓
Qwen2.5	RefusalClassifier	89.0	27.18s	52.0	27.34s	**0**	1.44s	**0**	1.79s
-0.5B	SVMClassifier	33.0	21.85s	46.0	27.05s	**0**	1.46s	**0**	1.46s
-Instruct	MultiRefusalsClassifier	29.0	**17.45**s	53.0	18.32s	**0**	**0.58**s	**0**	**0.89**s
	AlignTreeLinear	61.0	22.67s	43.0	**16.57**s	**0**	0.95s	**0**	1.09s
	AlignTree	**4.0**	19.01s	**41.0**	24.8s	**0**	3.16s	**0**	1.27s
Llama	RefusalClassifier	5.0	145.54s	44.0	57.68s	2.0	20.55s	**0**	59.84s
-3.1-8B	SVMClassifier	**2.0**	66.58s	20.0	42.02s	**1.0**	17.02s	**0**	43.9s
-Instruct	MultiRefusalsClassifier	4.0	**62.49**s	32.0	**31.67**s	**1.0**	**11.66**s	**0**	**35.79**s
	AlignTreeLinear	7.0	101.99s	**18.0**	40.61s	**1.0**	14.38s	**0**	36.74s
	AlignTree	5.0	87.37s	**18.0**	34.2s	**1.0**	21.88s	**0**	37.44s
gemma	RefusalClassifier	21.0	619.41s	54.0	957.3s	**0**	981.92s	**0**	983.69s
-3-12b	SVMClassifier	26.0	738.11s	37.0	708.4s	5.0	965.68s	**0**	969.49s
-it	MultiRefusalsClassifier	25.0	509.97s	53.0	**602.4**s	**0**	**640.47**s	**0**	**606.22**s
	AlignTreeLinear	**8.0**	**496.35**s	**29.0**	800.97s	**0**	959.48s	**0**	949.66s
	AlignTree	10.0	591.11s	40.0	717.22s	**0**	988.4s	**0**	978.93s

Table 3: This table reports ASR, refusal rates, and execution time for each dataset, illustrating the impact of ablating individual components of AlignTree. The full results across eight datasets and nine LLMs from the three families are in the Appendix.

SVMClassifier, although leveraging non-linear signals, fails to generalize across datasets and exhibits excessive refusal rates, particularly on Gemma-3-12b.

Some variants, such as the RefusalClassifier and the MultiRefusalsClassifier, exhibit substantial ASR variability: for example, they perform strongly on Llama-3.1-8B-Instruct but poorly on Qwen2.5-0.5B-Instruct. We attribute this inconsistency to differences in the base models' pretrained alignment behavior, which we discuss further in the Appendix. Nevertheless, the MultiRefusalsClassifier outperforms its single-classifier counterpart, reinforcing the hypothesis that refusal mechanisms are multidimensional phenomena.

Most classifiers manage to avoid over-refusal, preserving usability on benign datasets such as PIQA and ARC. Notable exceptions include the SVMClassifier on PIQA for Gemma-3-12b and the RefusalClassifier on PIQA for Llama-3.1-8B, both of which demonstrate elevated refusal rates.

In summary, AlignTree emerges as the most reliable and general-purpose defense, consistently achieving a favorable trade-off between robustness, efficiency, and usability. While other classifier-based defenses leveraging model activations may be suitable in certain contexts, AlignTree demonstrates the most stable and dependable performance overall, making it a strong candidate for real-world deployment. Additional ablation results across all models and datasets are provided in the Appendix, demonstrating similar trends.

Conclusions

We introduced AlignTree, an efficient defense that enhances model alignment while maintaining minimal computational overhead. In order to build this lightweight deference, we trained a Random Forest classifier that integrates the linear refusal direction with a novel SVM-based signal designed to capture non-linear features associated with harmful content. Our results show that AlignTree consistently outperforms existing defenses in terms of ASR, refusal, and computational efficiency while introducing a non-negligible increase in execution time over the baseline. Moreover, our results demonstrate that leveraging non-linear harmfulness signals leads to improved alignment performance compared to relying solely on a single linear refusal vector, which we believe is essential for advancing alignment strategies. In future work, we plan to extend AlignTree by introducing an additional "suspicious" threshold, one that distinguishes borderline prompts from clearly benign or harmful ones. It will allow identifying prompts that warrant further analysis without immediate rejection and can be used jointly with additional defenses.

Limitations

While AlignTree represents a meaningful advancement in improving the alignment of LLMs, several limitations remain.

ASR evaluations in this work were conducted using another LLM, following methodologies similar to those in Arditi et al. (2024) and related defense studies. While practical, this approach may occasionally introduce evaluation inaccuracies due to model-based judgment.

Another limitation is that AlignTree requires training a separate classifier for each model, and its effectiveness depends heavily on the level of the base model's initial alignment and the quality of the data. Finally, while this work combined linear and non-linear signals, further research could explore more direct approaches to characterizing and utilizing non-linear refusal properties; for instance, by identifying additional semantic directions or better modeling the refusal manifold in latent space.

Finally, AlignTree relies on a limited set of input signals and lightweight classifiers to reduce the risk of overfitting. Future work could explore the use of more complex models and larger training datasets to further enhance performance.

Ethics Statement

This work aims to enhance language models by introducing a novel method to improve their safe usage through efficient and robust defenses. We recognize the potential of such technologies and emphasize the importance of their responsible use. While our contributions are intended to support the development of more aligned models, we stress the need to prevent misuse, such as generating harmful content. Future research should focus on promoting more efficient defense strategies that align with societal benefits.

Acknowledgements

This work was supported by a Tel Aviv University Center for AI and Data Science (TAD) grant. This research was also supported by the Ministry of Innovation, Science & Technology, Israel (1001576154) and the Michael J. Fox Foundation (MJFF-022407). The contribution of SK is part of a PhD thesis research conducted at Tel Aviv University.

References

Andriushchenko, M.; Croce, F.; and Flammarion, N. 2025. Jailbreaking Leading Safety-Aligned LLMs with Simple Adaptive Attacks. arXiv:2404.02151.

Arditi, A.; Obeso, O.; Syed, A.; Paleka, D.; Panickssery, N.; Gurnee, W.; and Nanda, N. 2024. Refusal in Language Models Is Mediated by a Single Direction. arXiv:2406.11717.

Bai, Y.; Kadavath, S.; Kundu, S.; Askell, A.; Kernion, J.; Jones, A.; Chen, A.; Goldie, A.; Mirhoseini, A.; McKinnon, C.; Chen, C.; Olsson, C.; Olah, C.; Hernandez, D.; Drain, D.; Ganguli, D.; Li, D.; Tran-Johnson, E.; Perez, E.; Kerr, J.; Mueller, J.; Ladish, J.; Landau, J.; Ndousse, K.; Lukosuite, K.; Lovitt, L.; Sellitto, M.; Elhage, N.; Schiefer, N.; Mercado, N.; DasSarma, N.; Lasenby, R.; Larson, R.; Ringer, S.; Johnston, S.; Kravec, S.; Showk, S. E.; Fort, S.; Lanham, T.; Telleen-Lawton, T.; Conerly, T.; Henighan, T.; Hume, T.; Bowman, S. R.; Hatfield-Dodds, Z.; Mann, B.; Amodei, D.; Joseph, N.; McCandlish, S.; Brown, T.; and Kaplan, J. 2022. Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073.

Bisk, Y.; Zellers, R.; Bras, R. L.; Gao, J.; and Choi, Y. 2020. PIQA: Reasoning about Physical Commonsense in Natural Language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Chao, P.; Debenedetti, E.; Robey, A.; Andriushchenko, M.; Croce, F.; Sehwag, V.; Dobriban, E.; Flammarion, N.; Pappas, G. J.; Tramer, F.; Hassani, H.; and Wong, E. 2024a. JailbreakBench: An Open Robustness Benchmark for Jailbreaking Large Language Models. *https://arxiv.org/abs/2404.01318*.

Chao, P.; Robey, A.; Dobriban, E.; Hassani, H.; Pappas, G. J.; and Wong, E. 2024b. Jailbreaking Black Box Large Language Models in Twenty Queries. arXiv:2310.08419.

Chen, B.; Paliwal, A.; and Yan, Q. 2023. Jailbreaker in Jail: Moving Target Defense for Large Language Models. arXiv:2310.02417.

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. arXiv:1803.05457.

Derczynski, L.; Galinkin, E.; Martin, J.; Majumdar, S.; and Inie, N. 2024. garak: A Framework for Security Probing Large Language Models. arXiv:2406.11036.

Dong, W.; Li, P.; Tian, Y.; Zeng, X.; Li, F.; and Wang, S. 2025. Feature-Aware Malicious Output Detection and Mitigation. arXiv:2504.09191.

Gemma Team; Kamath, A.; Ferret, J.; et al. 2025. Gemma 3 Technical Report. arXiv:2503.19786.

Glaese, A.; McAleese, N.; Trębacz, M.; Aslanides, J.; Firoiu, V.; Ewalds, T.; Rauh, M.; Weidinger, L.; Chadwick, M.; Thacker, P.; Campbell-Gillingham, L.; Uesato, J.; Huang, P.-S.; Comanescu, R.; Yang, F.; See, A.; Dathathri, S.; Greig, R.; Chen, C.; Fritz, D.; Elias, J. S.; Green, R.; Mokrá, S.; Fernando, N.; Wu, B.; Foley, R.; Young, S.; Gabriel, I.; Isaac, W.; Mellor, J.; Hassabis, D.; Kavukcuoglu, K.; Hendricks, L. A.; and Irving, G. 2022. Improving alignment of dialogue agents via targeted human judgements. arXiv:2209.14375.

Grattafiori, A.; Dubey, A.; Jauhri, A.; et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783.

Gu, J.; Jiang, X.; Shi, Z.; Tan, H.; Zhai, X.; Xu, C.; Li, W.; Shen, Y.; Ma, S.; Liu, H.; Wang, S.; Zhang, K.; Wang, Y.; Gao, W.; Ni, L.; and Guo, J. 2025. A Survey on LLM-as-a-Judge. arXiv:2411.15594.

Gupta, M.; Akiri, C.; Aryal, K.; Parker, E.; and Praharaj, L. 2023. From ChatGPT to ThreatGPT: Impact of Generative AI in Cybersecurity and Privacy. arXiv:2307.00691.

Hildebrandt, F.; Maier, A.; Krauss, P.; and Schilling, A. 2025. Refusal Behavior in Large Language Models: A Nonlinear Perspective. arXiv:2501.08145.

Huang, Y.; Gupta, S.; Xia, M.; Li, K.; and Chen, D. 2023. Catastrophic Jailbreak of Open-source LLMs via Exploiting Generation. arXiv:2310.06987.

Jain, N.; Schwarzschild, A.; Wen, Y.; Somepalli, G.; Kirchenbauer, J.; yeh Chiang, P.; Goldblum, M.; Saha, A.; Geiping, J.; and Goldstein, T. 2023. Baseline Defenses for Adversarial Attacks Against Aligned Language Models. arXiv:2309.00614.

Kumar, A.; Agarwal, C.; Srinivas, S.; Li, A. J.; Feizi, S.; and Lakkaraju, H. 2025. Certifying LLM Safety against Adversarial Prompting. arXiv:2309.02705.

Li, Y.; Wei, F.; Zhao, J.; Zhang, C.; and Zhang, H. 2023. RAIN: Your Language Models Can Align Themselves without Finetuning. arXiv:2309.07124.

Liu, X.; Xu, N.; Chen, M.; and Xiao, C. 2024a. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. arXiv:2310.04451.

Liu, Y.; Deng, G.; Xu, Z.; Li, Y.; Zheng, Y.; Zhang, Y.; Zhao, L.; Zhang, T.; Wang, K.; and Liu, Y. 2024b. Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. arXiv:2305.13860.

Llama Team and AI @ Meta. 2024. The Llama 3 Family of Models.

Mazeika, M.; Phan, L.; Yin, X.; Zou, A.; Wang, Z.; Mu, N.; Sakhaee, E.; Li, N.; Basart, S.; Li, B.; Forsyth, D.; and

Hendrycks, D. 2024. HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal. arXiv:2402.04249.

Mazeika, M.; Zou, A.; Mu, N.; Phan, L.; Wang, Z.; Yu, C.; Adam Khoja, F. J.; O'Gara, A.; Sakhaee, E.; Xiang, Z.; Rajabi, A.; Hendrycks, D.; Poovendran, R.; Li, B.; ; and Forsyth, D. 2023. TDC 2023 (LLM edition): the Trojan Detection Challenge.

Mehrotra, A.; Zampetakis, M.; Kassianik, P.; Nelson, B.; Anderson, H.; Singer, Y.; and Karbasi, A. 2024. Tree of Attacks: Jailbreaking Black-Box LLMs Automatically. arXiv:2312.02119.

Mihaylov, T.; Clark, P.; Khot, T.; and Sabharwal, A. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *EMNLP*.

OpenAI. 2024. ChatGPT-4o. https://chat.openai.com/.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C. L.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; Schulman, J.; Hilton, J.; Kelton, F.; Miller, L.; Simens, M.; Askell, A.; Welinder, P.; Christiano, P.; Leike, J.; and Lowe, R. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155.

Perez, F.; and Ribeiro, I. 2022. Ignore Previous Prompt: Attack Techniques For Language Models.

Phute, M.; Helbling, A.; Hull, M.; Peng, S.; Szyller, S.; Cornelius, C.; and Chau, D. H. 2024. LLM Self Defense: By Self Examination, LLMs Know They Are Being Tricked. arXiv:2308.07308.

Platt, J. 2000. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Adv. Large Margin Classif.*, 10.

Qiu, H.; Zhang, S.; Li, A.; He, H.; and Lan, Z. 2023. Latent Jailbreak: A Benchmark for Evaluating Text Safety and Output Robustness of Large Language Models. arXiv:2307.08487.

Robey, A.; Wong, E.; Hassani, H.; and Pappas, G. J. 2024. SmoothLLM: Defending Large Language Models Against Jailbreaking Attacks. arXiv:2310.03684.

Röttger, P.; Kirk, H.; Vidgen, B.; Attanasio, G.; Bianchi, F.; and Hovy, D. 2024. XSTest: A Test Suite for Identifying Exaggerated Safety Behaviours in Large Language Models. In Duh, K.; Gomez, H.; and Bethard, S., eds., *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 5377–5400. Mexico City, Mexico: Association for Computational Linguistics.

Sap; Maarten; Rashkin; Hannah; Chen; Derek; Bras, L.; Ronan; Choi; and Yejin. 2019. Social IQa: Commonsense Reasoning about Social Interactions. In Inui; Kentaro; Jiang; Jing; Ng; Vincent; Wan; and Xiaojun, eds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 4463–4473. Hong Kong and China: Association for Computational Linguistics.

Souly, A.; Lu, Q.; Bowen, D.; Trinh, T.; Hsieh, E.; Pandey, S.; Abbeel, P.; Svegliato, J.; Emmons, S.; Watkins, O.; and Toyer, S. 2024. A StrongREJECT for Empty Jailbreaks. arXiv:2402.10260.

Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; and Hashimoto, T. B. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Team, Q. 2024. Qwen2.5: A Party of Foundation Models.

Wei, A.; Haghtalab, N.; and Steinhardt, J. 2023. Jailbroken: How Does LLM Safety Training Fail? arXiv:2307.02483.

Wollschläger, T.; Elstner, J.; Geisler, S.; Cohen-Addad, V.; Günnemann, S.; and Gasteiger, J. 2025. The Geometry of Refusal in Large Language Models: Concept Cones and Representational Independence. arXiv:2502.17420.

Xu, Z.; Jiang, F.; Niu, L.; Jia, J.; Lin, B. Y.; and Poovendran, R. 2024. SafeDecoding: Defending against Jailbreak Attacks via Safety-Aware Decoding. arXiv:2402.08983.

Yao, Y.; Duan, J.; Xu, K.; Cai, Y.; Sun, Z.; and Zhang, Y. 2024. A survey on large language model (LLM) security and privacy: The Good, The Bad, and The Ugly. *High-Confidence Computing*, 4(2): 100211.

Zeng, W.; Liu, Y.; Mullins, R.; Peran, L.; Fernandez, J.; Harkous, H.; Narasimhan, K.; Proud, D.; Kumar, P.; Radharapu, B.; Sturman, O.; and Wahltinez, O. 2024a. ShieldGemma: Generative AI Content Moderation Based on Gemma. arXiv:2407.21772.

Zeng, Y.; Wu, Y.; Zhang, X.; Wang, H.; and Wu, Q. 2024b. AutoDefense: Multi-Agent LLM Defense against Jailbreak Attacks. arXiv:2403.04783.

Zhang, S.; Zhai, Y.; Guo, K.; Hu, H.; Guo, S.; Fang, Z.; Zhao, L.; Shen, C.; Wang, C.; and Wang, Q. 2025. JBShield: Defending Large Language Models from Jailbreak Attacks through Activated Concept Analysis and Manipulation. arXiv:2502.07557.

Zou, A.; Wang, Z.; Carlini, N.; Nasr, M.; Kolter, J. Z.; and Fredrikson, M. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. arXiv:2307.15043.