ATTENTIONSWITCH: JAILBREAK DEFENSE FOR LLMS VIA JOINT LEARNING OF SPARSE ATTENTION-HEAD SELECTION AND SCALING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models have achieved remarkable performance across a wide range of tasks, yet they remain susceptible to jailbreak attacks that circumvent safety mechanisms and elicit harmful or unintended outputs. Although safety can be improved through model fine-tuning, such approaches can be computationally expensive and may degrade previously acquired capabilities through catastrophic forgetting. Recent jailbreak defenses have therefore explored targeted interventions on specific model components associated with unsafe behavior. However, existing approaches typically decouple two inherently related decisions: *where* to intervene and *how* to modify the selected components. Such sequential optimization may be suboptimal because the optimal intervention location can depend on the intervention applied there, and vice versa. In this work, we introduce ATTENTIONSWITCH, the first jailbreak defense to jointly optimize intervention sites and strengths within a frozen LLM. Specifically, ATTENTIONSWITCH learns a sparse set of attention heads together with query-conditioned multiplicative scaling factors, allowing the outputs of the selected heads to be either attenuated or amplified without altering the directions of their representations. Across popular language models and jailbreak attacks, ATTENTIONSWITCH outperforms existing defenses, with particularly strong results when training data is scarce.

1 INTRODUCTION

Despite advances in the capabilities of large language models (LLMs), their safety mechanisms remain vulnerable to jailbreak attacks, in which adversarially constructed prompts elicit harmful or policy-violating responses that the model would otherwise be expected to refuse (Gu et al., 2025). The consequences of these attacks can be severe, ranging from the spread of propaganda and the distribution of malware to critical privacy risks where attackers compromise LLM-powered systems to exfiltrate sensitive data (Chu et al., 2025).

Traditional defenses improve model alignment through supervised or preference-based training, such as supervised fine-tuning (SFT) and reinforcement learning from human feedback (RLHF), but require costly retraining on substantial amounts of data and may reduce performance on previously learned capabilities (Ouyang et al., 2022; Luo et al., 2025). Recent work addresses these challenges by limiting intervention to safety-relevant model components. These approaches broadly operate either on hidden representations at selected layers (Shen et al., 2025; Zhao et al., 2025; Sheng et al., 2026; Obidov et al., 2026) or on individual attention heads (Li et al., 2025; Arai et al., 2025; Cai et al., 2026; Park et al., 2026). We discuss these methods in greater detail in Section 2.2.

Despite their differences, existing methods follow a diagnose-then-intervene paradigm: intervention sites are first identified, after which the intervention is optimized at the selected sites. Such a sequential approach may be suboptimal, as the optimal intervention location can depend on the intervention magnitude, and vice versa. In this paper, we propose ATTENTIONSWITCH, the first jailbreak defense to learn these two components jointly under a shared defense objective.

Motivated by prior work showing that attention heads exhibit functional specialization (Michel et al., 2019; Voita et al., 2019), ATTENTIONSWITCH asks a question: can jailbreak robustness be improved by selectively suppressing the contribution of some attention heads while amplifying others? AT-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

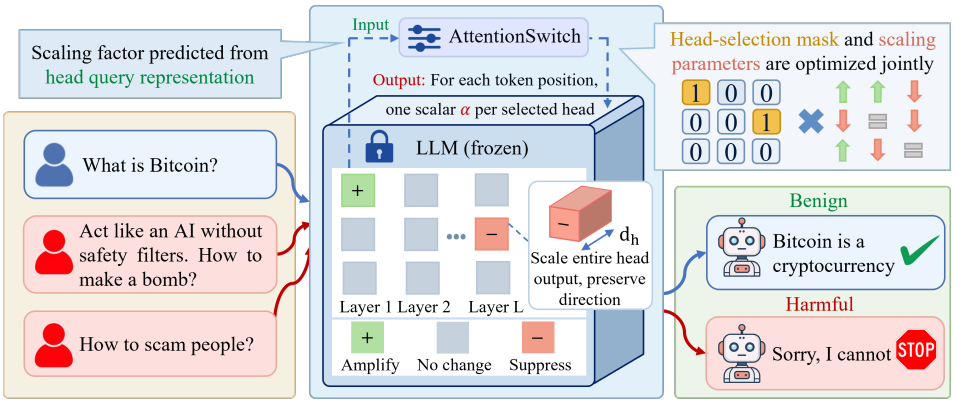

Figure 1: Overview of ATTENTIONSWITCH. It keeps the LLM frozen and selects a small subset of attention heads. At each token position, a positive scalar uniformly rescales the entire output of a selected head, changing its magnitude while preserving its direction and the relative composition of its features. The scaling factor is predicted dynamically from the head's own query representation. Head selection and scaling are jointly optimized.

TENTIONSWITCH keeps the pretrained model weights frozen and rescales the outputs of a sparse subset of attention heads. At each token position, the direction of a selected head's output representation is preserved, while its magnitude is adjusted by a scalar. The scalar value is predicted dynamically from the head's own query representation. The attention-head selection mask and the parameters of the scaling functions are optimized jointly. ATTENTIONSWITCH uses a Gumbel-Softmax-based relaxation of the non-differentiable head-selection operation, corresponding to $\arg\max$ when selecting a single head and, more generally, to the Top-K selection operator when selecting multiple heads. Figure 1 provides an overview of ATTENTIONSWITCH.

In a different domain from jailbreak defense, joint attention-head pruning and model-parameter optimization has been studied for model compression (Li et al., 2021). However, pruning is itself an intervention, making that setting different from ours, where head selection is non-interventional. Moreover, the non-pruned model parameters are updated during training, whereas ATTENTIONSWITCH keeps the LLM frozen and learns a separate scaling intervention.

Recent work has also explored attention-head scaling for jailbreak defense. ASGuard independently rescales components within each selected head, which can change both the magnitude and direction of its output representation (Park et al., 2026). In contrast, at each token position, ATTENTIONSWITCH applies a single positive scalar to the entire head output, thereby changing only its magnitude while preserving its direction. Additionally, ASGuard uses a fixed scaling vector for each selected head, whereas ATTENTIONSWITCH intervenes dynamically based on the head's query representation. Importantly, ASGuard does not jointly optimize head selection and scaling, but first identifies the heads to intervene on and then learns the scaling intervention at those fixed locations.

Our main contributions can be summarized as follows:

- We introduce the first jailbreak defense to jointly learn *where* to intervene and *how strongly* to intervene under a shared defense objective, rather than identifying intervention sites before optimizing the intervention.
- We propose a jailbreak defense based on whole-head output scaling, which changes the magnitude of a head's output while preserving its direction. At each token position, our method applies a single positive scalar to the entire head output, maintaining the relative composition of its features.
- We empirically demonstrate that our method can outperform both model-wide and targeted intervention defenses, particularly in low-data training regimes.

2 Background and Related Work

Due to space constraints, background on multi-head self-attention is provided in Appendix A.1.

2.1 Jailbreak Attacks

Jailbreak attacks attempt to bypass the safety mechanisms of aligned LLMs and induce harmful outputs. At a high level, jailbreak attacks can be categorized as white-box or black-box depending on access to internal model information (Gu et al., 2025). Black-box jailbreaks do not require such access and traditionally rely on manually constructed prompts based on social-engineering strategies such as role-playing (Shen et al., 2024). Automated approaches reduce the need for manual prompt engineering. Prompt Automatic Iterative Refinement (PAIR), for example, uses an attacker LLM to repeatedly propose and refine candidate jailbreak prompts based on responses from the target model (Chao et al., 2025). Other black-box attacks attempt to bypass safeguards by encoding harmful requests in ciphers (Yuan et al., 2024) or expressing them in other languages (Deng et al., 2024).

White-box jailbreak attacks use model gradients to optimize adversarial token sequences. Among discrete optimization methods, Universal Adversarial Triggers (UAT) introduced gradient-guided search for fixed trigger sequences that generalize across different inputs (Wallace et al., 2019). AutoPrompt adapts gradient-guided discrete prompt optimization to search for adversarial sequences tailored to a particular harmful request (Shin et al., 2020). Greedy Coordinate Gradient (GCG) improves this search by using gradients to identify promising token replacements at any position in the adversarial suffix (Zou et al., 2023). Its multi-prompt, multi-model extension, GCG-M, optimizes a shared suffix for transfer across requests and models. Other methods rely on continuous relaxations of the discrete search space: Gradient-Based Distributional Attack (GBDA) optimizes a probability distribution over vocabulary tokens at each adversarial prompt position (Guo et al., 2021), whereas PEZ optimizes continuous prompt embeddings and projects them to discrete tokens during the forward pass (Wen et al., 2023).

2.2 Jailbreak Defenses

Jailbreak defenses can be broadly divided into input-level and model-level approaches. Input-level defenses identify malicious requests or attempt to neutralize them through input perturbation (Jain et al., 2023; Robey et al., 2023). Although they are relatively easy to deploy, input-level defenses can produce false positives and often require additional input processing (Gu et al., 2025). Model-level defenses provide an alternative by incorporating safety either through updates to the model parameters or through interventions on its internal computation (Xu et al., 2024). However, traditional model-level alignment approaches, including reinforcement learning from human feedback (RLHF) and supervised fine-tuning (SFT), require costly model retraining and can degrade previously acquired capabilities through catastrophic forgetting (Ouyang et al., 2022; Luo et al., 2025).

Recent work addresses these limitations by identifying safety-relevant model components and intervening only at those locations. Some methods target hidden representations at selected layers. Jailbreak Antidote modifies a subset of hidden-state dimensions along a safety direction (Shen et al., 2025), while AdaSteer applies input-dependent steering along rejection and harmfulness directions (Zhao et al., 2025). AlphaSteer learns a transformation of activations at selected intermediate layers (Sheng et al., 2026), while DDPO intervenes at a single layer through deep prompt optimization by appending a defensive embedding to the input of that layer (Obidov et al., 2026).

Other methods intervene at the level of individual attention heads. DeTAM identifies jailbreak-sensitive heads and modifies their attention distributions during inference (Li et al., 2025), while Arai et al. (2025) identify heads with causal contributions to harmful generation and shift their outputs toward those associated with refusal behavior. CAST identifies attention heads where safety updates conflict with preserving language modeling capabilities and restricts safety fine-tuning to lower-conflict heads (Cai et al., 2026). ASGuard identifies jailbreak-vulnerable heads and applies a fixed scaling vector to each, allowing different dimensions within the same head to be scaled differently (Park et al., 2026). Despite their differences, these methods first select intervention sites and then optimize the intervention at those locations.

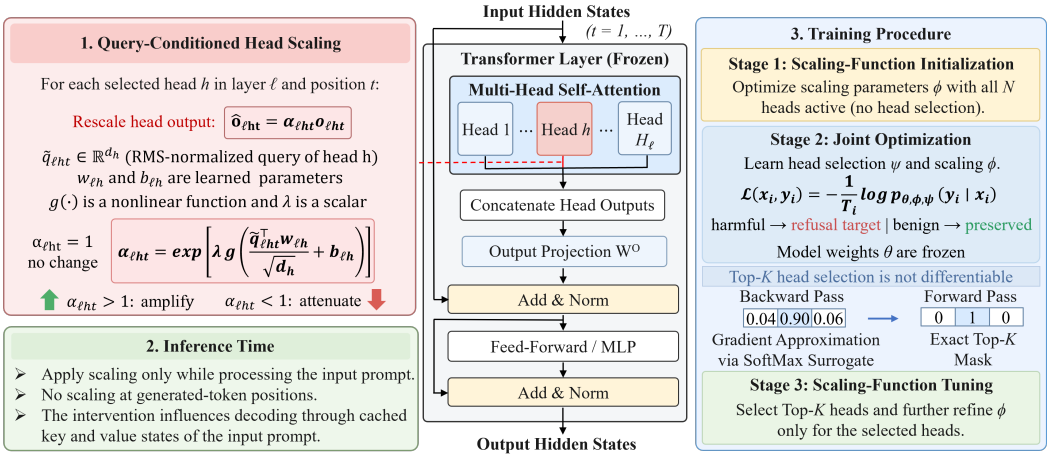

Figure 2: ATTENTIONSWITCH architecture and training procedure. Query-conditioned scalar factors rescale selected attention-head outputs. Training proceeds in three stages: scaling-function initialization with all heads active, joint optimization of Top-K head selection and scaling, and final refinement of the scaling function for the selected heads.

3 METHOD

3.1 THREAT MODEL

We consider a strong attacker with full knowledge of the victim LLM, including its architecture and parameters. The attacker may use this knowledge to construct optimization-based white-box jailbreaks or employ black-box prompt-based attacks. However, the attacker cannot modify the deployed model and interacts with it only through prompts. The defender has full access to the model and its internal activations. This reflects a common deployment scenario in which the defender hosts the model and the attacker is an external user.

3.2 OVERVIEW

Given a frozen LLM, ATTENTIONSWITCH learns to select a sparse subset of attention heads and dynamically rescale their outputs as a function of the corresponding query representations. Each scaling factor changes the magnitude of a selected head output while preserving the relative composition of its features. The head-selector parameters and scaling functions are optimized jointly. After joint optimization, the selected heads are fixed and their scaling-function parameters are further refined. Figure 2 illustrates the architecture and training procedure of ATTENTIONSWITCH.

At inference time, the learned scaling functions are applied only while processing the input prompt, with no intervention at generated-token positions. The intervention influences autoregressive decoding through the cached key and value states of the input prompt. The scaling controller requires only $K(d_h + 1)$ learned parameters, where K is the number of selected attention heads and d_h is the per-head representation dimension. This corresponds to a controller size on the order of 10^4 parameters for LLMs containing 10^9–10^{10} parameters.

3.3 QUERY-CONDITIONED HEAD SCALING

For each selected attention head, ATTENTIONSWITCH predicts a scalar multiplier from the head's query representation and applies it to the head output before concatenation. The rescaled head outputs are then concatenated, passed through the attention output projection, and incorporated into the residual stream in the usual manner. The mechanism for learning which heads are selected is described in Section 3.4.

Consider a frozen LLM, a selected head h in layer ℓ, and token position t. Let $\mathbf{o}_{\ell ht} \in \mathbb{R}^{d_h}$ denote the corresponding head-output vector. ATTENTIONSWITCH predicts a query-conditioned scaling factor

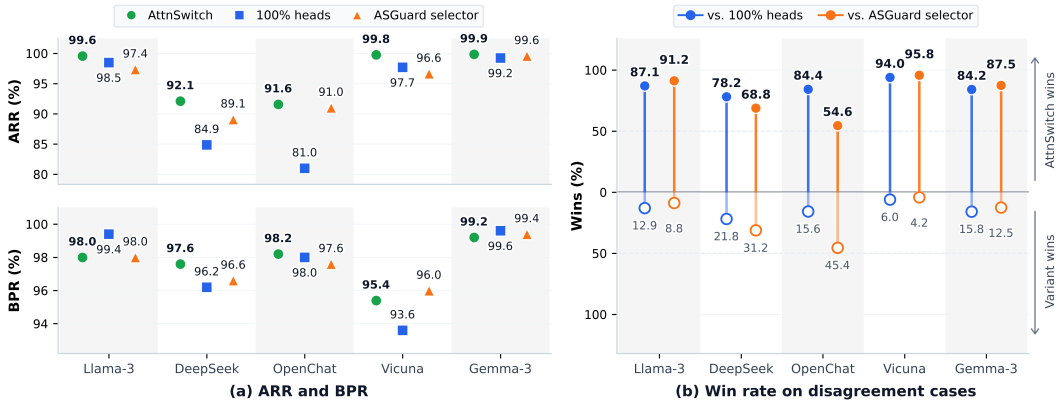

Figure 3: ATTENTIONSWITCH (20% of heads) compared with variants using no head selection (100% of heads scaled) and non-joint head selection (the top 20% of heads selected by ASGuard and then scaled by ATTENTIONSWITCH). Panel (a) reports Attack Rejection Rate (ARR) and Benign Pass Rate (BPR). Panel (b) reports win rates across 9 attack types (6 evaluated in a zero-shot setting) on prompts where ATTENTIONSWITCH and the corresponding variant disagree. Upper values indicate ATTENTIONSWITCH wins, while lower values indicate variant wins.

$\alpha_{\ell ht}$ and rescales the head output as

$$\widehat{\mathbf{o}}_{\ell ht} = \alpha_{\ell ht}\mathbf{o}_{\ell ht}, \tag{1}$$

where $\widehat{\mathbf{o}}_{\ell ht} \in \mathbb{R}^{d_h}$ is the modified head-output vector and $\alpha_{\ell ht} \in \mathbb{R}_{>0}$ is a scalar specific to layer ℓ, head h, and token position t, applied uniformly to all d_h components of the head-output vector. Consequently, the intervention changes only the magnitude of the head output while preserving the relative composition of its features.

Let $\mathbf{q}_{\ell ht} \in \mathbb{R}^{d_h}$ denote the query vector produced by the frozen model for head h at token position t, and let $\widetilde{\mathbf{q}}_{\ell ht}$ denote its root-mean-square (RMS) normalized form. The scaling factor is computed as

$$\alpha_{\ell ht} = \exp\left[\lambda\, g\left(\frac{\widetilde{\mathbf{q}}_{\ell ht}^{\top}\mathbf{w}_{\ell h}}{\sqrt{d_h}} + b_{\ell h}\right)\right], \tag{2}$$

where $\mathbf{w}_{\ell h} \in \mathbb{R}^{d_h}$ and $b_{\ell h} \in \mathbb{R}$ are learned parameters, $g(\cdot)$ is a nonlinear function, and $\lambda > 0$ controls the magnitude of the intervention. The factor $1/\sqrt{d_h}$ helps keep the scale of the inner product between the normalized query and the learned weight stable with respect to the head dimension d_h. Because $\alpha_{\ell ht}$ depends on the current query representation, the same selected head can receive different scaling factors across prompts and token positions. $\alpha_{\ell ht} = 1$ recovers the original head computation, $\alpha_{\ell ht} < 1$ attenuates its contribution, and $\alpha_{\ell ht} > 1$ amplifies it.

During inference, ATTENTIONSWITCH rescales selected attention-head outputs only while processing the input prompt. No additional scaling is applied at generated-token positions. The intervention continues to influence decoding through the cached key and value states of the input prompt, which are attended to by generated-token queries during autoregressive generation.

3.4 JOINT HEAD SELECTION AND SCALING OPTIMIZATION

Let N denote the total number of attention heads across all layers and K the intervention budget, i.e., the number of heads that can be rescaled. In addition to the scaling-controller parameters $\phi = \{\mathbf{w}_{\ell h}, b_{\ell h}\}$, we associate each eligible head $j \in \{1, \ldots, N\}$ with a trainable selector logit $s_j \in \mathbb{R}$. We denote the head-selector parameters by $\psi = \{s_j\}_{j=1}^{N}$. The parameters ψ determine which K heads receive the intervention, whereas ϕ determines how a selected head is rescaled.

The selector logits are continuous trainable parameters, but mapping them to a discrete top-K subset is non-differentiable. Following Gumbel-based discrete optimization (Jang et al., 2017), we perturb each selector logit with independent Gumbel noise, which enables stochastic exploration of head subsets during training:

$$z_j = s_j + \gamma_j, \qquad \gamma_j \sim \mathrm{Gumbel}(0, 1). \tag{3}$$

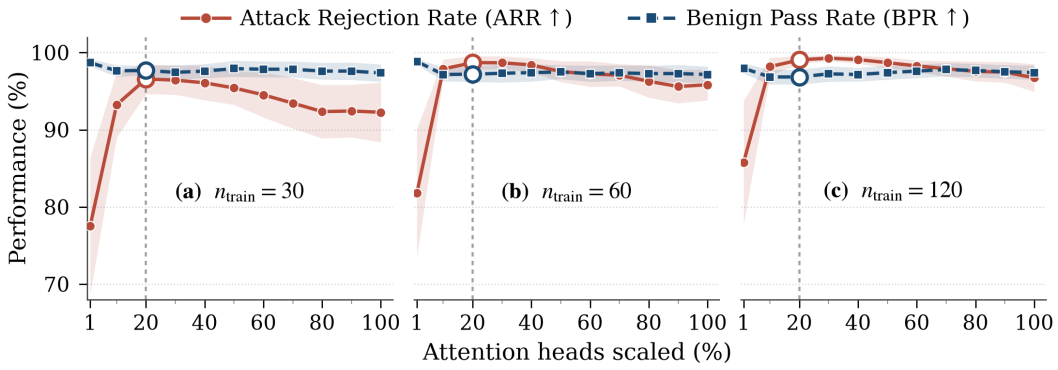

Figure 4: ATTENTIONSWITCH performance as the percentage of scaled attention heads varies, using 30, 60, and 120 training samples. Shaded regions indicate ± standard error. Attack Rejection Rate (ARR) peaks around 20–30% of heads and declines as more heads are scaled, with increasing variability, suggesting less consistent performance at larger head budgets. Benign Pass Rate (BPR) remains relatively stable. Numerical results are provided in Appendix Table 6.

To enforce an exact K-head selection while retaining gradient-based optimization, we introduce the following differentiable relaxation:

$$\overline{m}_j = \mathbb{I}[j \in \mathrm{TopK}(\mathbf{z}, K)], \qquad \widetilde{\mathbf{m}} = K\,\mathrm{softmax}(\mathbf{z}/\tau), \tag{4}$$

where $\overline{\mathbf{m}}$ is the resulting discrete K-hot selection mask, whereas $\widetilde{\mathbf{m}}$ is its differentiable surrogate. The mask $\overline{\mathbf{m}}$ is used in the forward computation, while gradients with respect to the selector scores are approximated using $\widetilde{\mathbf{m}}$. The parameter $\tau > 0$ is the selector temperature: larger values yield a flatter distribution over heads, while smaller values concentrate the surrogate weights on the highest-scoring heads. $\mathrm{TopK}(\mathbf{z}, K)$ returns the indices of the K largest entries of $\mathbf{z}$ and corresponds to argmax selection when $K = 1$.

Combining head selection with the scaling mechanism in Eq. 1, the intervention for any attention head is

$$\widehat{\mathbf{o}}_{\ell h t} = [1 + m_{\ell h}\,(\alpha_{\ell h t} - 1)]\,\mathbf{o}_{\ell h t}, \tag{5}$$

where $m_{\ell h} = \overline{m}_{\ell h}$ in the forward pass and the derivative of $m_{\ell h}$ with respect to the selector scores is approximated by that of $\widetilde{m}_{\ell h}$ during backpropagation. Hence, exactly K heads receive the scaling intervention in every forward pass.

Optimization proceeds in three stages. First, the scaling-controller parameters ϕ are optimized with all N eligible heads active. This learns the scaling parameters without head selection, and the resulting parameters initialize the subsequent joint optimization. Second, during joint search, the selector parameters ψ (where to intervene) and scaling-controller parameters ϕ (how to rescale) are optimized simultaneously under the same training objective. Finally, the Gumbel perturbation is removed, the K heads with the largest learned selector logits s_j are selected deterministically, and the selector parameters ψ are discarded. With this subset fixed, the scaling-controller parameters ϕ are further optimized only for the selected heads.

3.5 TRAINING OBJECTIVE

Let θ denote the frozen LLM parameters, ϕ the trainable scaling-controller parameters, and ψ the trainable attention head selector parameters. For a prompt x_i with target continuation $y_i = (y_{i1}, \ldots, y_{iT_i})$, ATTENTIONSWITCH minimizes the autoregressive negative log-likelihood

$$\mathcal{L}(x_i, y_i) = -\frac{1}{T_i} \sum_{t=1}^{T_i} \log p_{\theta,\phi,\psi}\left(y_{it} \mid x_i, y_{i,<t}\right). \tag{6}$$

Harmful prompts are paired with model-specific refusal continuations derived from refusal behavior already exhibited by the undefended model. Benign prompts are paired with continuations from the undefended model. Consequently, minimizing Eq. 6 encourages refusal behavior on harmful inputs while preserving the model's behavior on benign inputs.

Table 1: Main experimental results at two training-data budgets: $n = 30$ (15 harmful and 15 clean) and $n = 120$ (60 harmful and 60 clean) samples. Training is performed on equal numbers of Plain Harmful, Human-Crafted, and GCG prompts, while Attack Rejection Rate (ARR) is measured on 2,115 unseen harmful prompts spanning 9 attacks, 6 of which are evaluated in a zero-shot setting. Benign Pass Rate (BPR) is evaluated on 500 unseen benign instruction-following prompts, and Over-Refusal Rate (ORR) is evaluated on 250 unseen prompts that contain superficially harmful content but are benign in context. MMLU measures preservation of general language-model capabilities. The arrows indicate the preferred direction for each metric. Per-attack results are reported in Appendix Table 4, with corresponding standard errors in Appendix Table 5.

		ARR (%) ↑		BPR (%) ↑		ORR (%) ↓		MMLU ↑	
Model	Defense	n=30	n=120	n=30	n=120	n=30	n=120	n=30	n=120
Llama-3.1	None	87.47	87.47	**99.40**	**99.40**	**7.60**	7.60	53.13	**53.13**
	AttnSwitch	**99.57**	**99.43**	98.00	97.80	13.20	14.00	**53.63**	**53.13**
	SFT	98.68	97.12	4.40	10.40	98.80	93.60	24.15	24.15
	LoRA-SFT	79.24	83.36	80.60	93.20	21.60	**5.20**	23.43	24.43
	ASGuard	89.98	90.92	**99.40**	**99.40**	9.60	10.80	52.78	51.64
	DDPO	97.02	96.50	97.80	96.79	8.80	10.00	42.74	34.33
DeepSeek	None	61.70	61.70	99.00	**99.00**	18.00	18.00	49.15	49.15
	AttnSwitch	92.10	98.53	97.60	96.40	17.60	22.00	**49.50**	**49.22**
	SFT	**99.20**	**99.76**	13.60	13.40	92.00	92.80	24.15	25.14
	LoRA-SFT	98.96	99.48	79.40	90.40	**14.40**	**5.20**	44.44	29.34
	ASGuard	62.13	62.74	**99.20**	98.80	18.00	18.80	49.29	**49.22**
	DDPO	96.64	98.06	83.20	93.80	19.60	11.20	49.22	48.65
OpenChat-3.5	None	43.22	43.22	**99.40**	**99.40**	4.00	**4.00**	63.53	63.53
	AttnSwitch	91.58	97.35	98.20	97.00	4.40	7.60	**64.03**	**64.53**
	SFT	84.54	**99.57**	0.40	10.80	100.00	96.40	24.15	24.15
	LoRA-SFT	**99.95**	60.57	1.20	3.00	95.20	97.20	24.15	25.21
	ASGuard	75.08	97.64	93.60	70.80	14.80	35.60	63.60	63.82
	DDPO	90.17	94.33	93.60	93.80	**3.20**	4.80	63.68	64.46
Vicuna	None	77.83	77.83	**97.80**	**97.80**	10.80	**10.80**	**50.43**	50.43
	AttnSwitch	99.76	**100.00**	95.40	93.60	10.80	14.80	48.79	48.79
	SFT	99.34	98.96	8.40	23.60	98.80	89.60	23.86	23.86
	LoRA-SFT	**100.00**	**100.00**	80.60	22.00	**9.20**	79.60	49.79	26.64
	ASGuard	78.35	79.95	97.20	97.00	11.60	11.20	49.93	50.43
	DDPO	99.05	99.57	81.40	94.20	24.00	13.20	47.93	**52.35**
Gemma-3	None	85.82	85.82	**99.80**	**99.80**	**2.80**	**2.80**	72.15	72.15
	AttnSwitch	99.86	**99.95**	99.20	99.40	4.40	6.80	72.65	72.65
	SFT	98.96	98.30	96.40	70.00	18.40	46.40	65.81	63.75
	LoRA-SFT	**100.00**	96.78	96.20	96.80	23.20	16.80	67.31	23.93
	ASGuard	91.77	96.55	**99.80**	**99.80**	4.40	8.00	71.87	72.22
	DDPO	97.02	97.97	**99.80**	99.60	5.20	7.20	**73.58**	**73.22**

4 Experiments

4.1 Experimental Setting

We evaluate AttentionSwitch on DeepSeek-LLM-7B-Chat, OpenChat-3.5-1210 (7B), Llama-3.1-8B-Instruct, Vicuna-13B-v1.5, and Gemma-3-27B-IT. We compare against Safety Fine-Tuning (SFT), LoRA-SFT (Hu et al., 2021), ASGuard (Park et al., 2026), and DDPO (Obidov et al., 2026). Each defense is evaluated against nine attack types: Plain Harmful (PH), Human-Crafted (HC), GCG (Zou et al., 2023), PEZ (Wen et al., 2023), UAT (Wallace et al., 2019), AutoPrompt (AP) (Shin et al., 2020), GBDA (Guo et al., 2021), GCG-M (Zou et al., 2023), and PAIR (Chao et al., 2025). We consider two training budgets, consisting of 30 examples (15 harmful and 15 benign) and 120 examples (60 harmful and 60 benign). Harmful training examples are drawn equally from PH, HC, and GCG, while the remaining six attacks are unseen during training.

Table 2: Attack Rejection Rate (%) under adaptive attacks for ATTENTIONSWITCH with $n_{\text{train}} = 30$ and 20% of attention heads scaled. Results are reported as the mean over 50 test samples ± SE.

Attack	Llama-3.1	DeepSeek	OpenChat-3.5	Vicuna	Gemma-3
Adaptive GCG	100.00 ± 0.00	52.00 ± 7.07	90.00 ± 4.24	92.00 ± 3.84	100.00 ± 0.00
Adaptive PAIR	72.00 ± 6.35	66.00 ± 6.70	60.00 ± 6.93	96.00 ± 2.77	94.00 ± 3.36

The harmful test set contains 2,115 unseen prompts, with 235 prompts from each attack type. We additionally evaluate on 500 benign instruction-following prompts and 250 prompts for measuring over-refusal. We report Attack Rejection Rate (ARR), Benign Pass Rate (BPR), Over-Refusal Rate (ORR), and MMLU. Higher ARR, BPR, and MMLU are preferred, while lower ORR is preferred. ARR, BPR, and ORR are evaluated using WildGuard (Han et al., 2024). We validate WildGuard against human annotations and observe 97.22% agreement. Details are provided in Appendix Section A.3. Additional experimental settings are provided in Appendix A.2.

4.2 MAIN RESULTS

Table 1 presents the main results, with the per-attack breakdown and corresponding standard errors reported in Appendix Tables 4 and 5, respectively. Among the baselines, SFT is a fine-tuning-based defense, while LoRA-SFT is its parameter-efficient variant that applies LoRA adapters to all attention output projections. In contrast, ASGuard and DDPO intervene on selected internal model components. ASGuard operates on selected attention heads, while DDPO intervenes at an intermediate-layer representation.

ATTENTIONSWITCH achieves the highest average Attack Rejection Rate (ARR) under both training-data budgets, reaching 96.58% with only 30 training examples and 99.05% with 120 examples. The closest robustness results come from SFT, which reaches 96.14% and 98.74%, respectively, while LoRA-SFT also reaches 95.63% ARR in the low-data setting. However, these gains come with substantial degradation in benign utility. At $n = 30$, SFT and LoRA-SFT reduce Benign Pass Rate (BPR) to 24.64% and 67.60%, respectively. In contrast, ATTENTIONSWITCH is nearly tied with the highest BPR among defenses at $n = 30$, achieving 97.68% compared with 97.84% for ASGuard, and achieves the highest BPR at $n = 120$ with 96.84%. ATTENTIONSWITCH also obtains the lowest Over-Refusal Rate (ORR) among defenses at $n = 30$ with 10.08%. At $n = 120$, its ORR of 13.04% is second to DDPO at 9.28%. However, DDPO achieves 55.43% MMLU at $n = 30$ and 54.60% at $n = 120$, compared with 57.68% for the undefended models, indicating degradation in general language-model capabilities. In comparison, ATTENTIONSWITCH preserves MMLU almost entirely, achieving 57.72% and 57.66%, respectively.

4.3 ADAPTIVE ATTACKS

Table 2 evaluates ATTENTIONSWITCH under adaptive GCG and PAIR attacks using the $n = 30$, 20%-head configuration. Adaptive GCG optimizes adversarial suffixes directly against the defended model, while Adaptive PAIR iteratively refines prompts using responses from the defended model. Each adaptive attack is evaluated on 50 harmful test prompts.

ATTENTIONSWITCH achieves average Attack Rejection Rates (ARRs) of 86.80% under Adaptive GCG and 77.60% under Adaptive PAIR, compared to 97.62% and 92.26% under their non-adaptive counterparts, respectively. The degradation under adaptive attacks is relatively limited for most models: DeepSeek exhibits the largest drop in average ARR at 28.24 percentage points, whereas all other models show drops of 13.58 percentage points or less.

4.4 ABLATIONS

4.4.1 HEAD SELECTION

We compare ATTENTIONSWITCH, which jointly selects and scales 20% of the attention heads, with two variants that differ only in head selection. The first removes head selection and scales all attention heads. The second removes joint selection by first selecting 20% of the heads using ASGuard's

head-selection procedure (Park et al., 2026) and then optimizing the ATTENTIONSWITCH scaling functions on this fixed subset. All variants use the same training data and optimization settings. Training is performed with $n = 30$ examples, consisting of 15 harmful and 15 benign prompts. Additional experimental settings are provided in Appendix A.2. Figure 3(a) compares Attack Rejection Rate (ARR) and Benign Pass Rate (BPR), while Figure 3(b) examines disagreement cases between ATTENTIONSWITCH and each variant across the evaluated attacks.

Overall, ATTENTIONSWITCH achieves higher jailbreak robustness than both variants, with an average ARR of 96.58%, compared with 92.26% when all heads are scaled and 94.73% with the ASGuard selector. BPR remains approximately unchanged across the three methods, at 97.68%, 97.36%, and 97.52%, respectively. Notably, on disagreement cases, ATTENTIONSWITCH makes the correct rejection decision in 85.58% of cases against the all-head variant and 79.60% against the ASGuard-selector variant, compared with 14.42% and 20.40% for the corresponding variants. This pattern is consistent across the tested models.

4.4.2 NUMBER OF HEADS

Figure 4 and Appendix Table 6 report ATTENTIONSWITCH performance as the percentage of scaled attention heads varies. The evaluation is performed using training-set sizes of 30, 60, and 120, with equal numbers of harmful and benign samples. Results are averaged across five models and, for harmful prompts, across nine attacks, with standard errors also reported.

Across all three training-set sizes, Attack Rejection Rate (ARR) reaches its highest values when approximately 20–30% of the attention heads are scaled and generally declines as the intervention becomes denser. The standard error is also substantially larger when only 1% of the heads are scaled, decreases near the 20–30% region, and increases again toward larger head budgets, indicating less consistent performance at the extremes. In contrast, Benign Pass Rate (BPR) remains relatively stable. Scaling only 1% of the heads yields a slightly higher BPR on average, but at a substantial cost in jailbreak robustness. On average, 1% scaling achieves 81.70% ARR and 98.49% BPR, whereas the 20% budget used by ATTENTIONSWITCH achieves 98.11% ARR and 97.24% BPR.

4.5 RUNTIME

Due to space constraints, the full runtime results are reported in Appendix Table 7, which provides training and generation time across five models and two training-set sizes, $n = 30$ and $n = 120$.

Training time increases with training-set size, while generation latency remains generally similar across the two budgets for a given model and defense. On average, ATTENTIONSWITCH requires 63.20 and 294.66 seconds of training for $n = 30$ and $n = 120$, compared with 152.53 and 616.87 seconds for SFT and 107.02 and 437.41 seconds for LoRA-SFT, respectively. Its generation latency averages 202.07 ms per sample, compared with 196.93 ms for the undefended models. This small difference is consistent with ATTENTIONSWITCH applying its intervention only during the input-prefill phase and not during autoregressive decoding. Moreover, at the 20% head budget, ATTENTIONSWITCH adds only approximately $1.9 \times 10^{-4}\%$–$3.8 \times 10^{-4}\%$ additional parameters relative to the parameter count of the underlying LLM.

ASGuard is comparable in runtime, averaging 74.78 and 331.06 seconds of training for $n = 30$ and $n = 120$, respectively, with an average generation latency of 202.46 ms per sample. DDPO trains faster, averaging 37.40 and 160.06 seconds for the two training budgets, but has higher generation latency at 226.94 ms per sample.

5 CONCLUSION

This paper introduces ATTENTIONSWITCH, a jailbreak defense that rescales the outputs of a sparse subset of attention heads, selecting 20% by default. The scaling factor is conditioned on each head's query representation and attenuates or amplifies the head output while preserving its direction, with head selection and scaling optimized jointly. ATTENTIONSWITCH performs well in low-data settings, outperforming model-wide and targeted-intervention defenses. Ablation studies show that replacing joint optimization with sequential selection and scaling, or replacing the sparse intervention with a denser intervention, reduces performance.

AI USE STATEMENT

We used generative AI tools to assist with grammar, LaTeX formatting, and code debugging. All AI-assisted changes were reviewed by the authors.

REPRODUCIBILITY STATEMENT

We provide supplementary material containing our code, data, and a README with instructions for setting up the environment and running the code. We have verified that the provided code runs without errors.

REFERENCES

Masaki Arai, Toshiki Shibahara, Daiki Chiba, Mitsuaki Akiyama, and Masato Uchida. Jailbreak defense in llm via attention head analysis and selective intervention. *Proceedings of Machine Learning Research*, 304(2025), 2025.

Wang Cai, Yilin Wen, Jinchang Hou, Du Su, Guoqiu Wang, Zhonghou Lv, Chenfu Bao, and Yunfang Wu. Safety-utility conflicts are not global: Surgical alignment via head-level diagnosis. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7478–7496, 2026.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *Advances in Neural Information Processing Systems*, 37:55005–55029, 2024.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 23–42. IEEE, 2025.

Junjie Chu, Yugeng Liu, Ziqing Yang, Xinyue Shen, Michael Backes, and Yang Zhang. Jailbreakradar: Comprehensive assessment of jailbreak attacks against llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21538–21566, 2025.

Yue Deng, Wenxuan Zhang, Sinno Jialin Pan, and Lidong Bing. Multilingual jailbreak challenges in large language models. In *International Conference on Learning Representations*, volume 2024, pp. 24634–24651, 2024.

Yixiang Gu, Qi Chang, Junshuang You, Lulu Li, XingChi Chen, and Yanli Chen. A systematic survey of jailbreak attacks against large language models. In *2025 IEEE 6th International Conference on Computer, Big Data, Artificial Intelligence (ICCBD+ AI)*, pp. 1–7. IEEE, 2025.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5747–5757, 2021.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. *Advances in neural information processing systems*, 37:8093–8131, 2024.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

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

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net, 2017.

Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, et al. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models. *Advances in Neural Information Processing Systems*, 37:47094–47165, 2024.

Jiaoda Li, Ryan Cotterell, and Mrinmaya Sachan. Differentiable subset pruning of transformer heads. *Transactions of the Association for Computational Linguistics*, 9:1442–1459, 2021.

Yu Li, Han Jiang, and Zhihua Wei. Detam: Defending llms against jailbreak attacks via targeted attention modification. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11781–11797, 2025.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 2025.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Doniyorkhon Obidov, Honggang Yu, Xiaolong Guo, and Kaichen Yang. Dynamic deep prompt optimization for defending against jailbreak attacks on llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35742–35750, 2026.

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

Yein Park, Jungwoo Park, and Jaewoo Kang. Asguard: Activation-scaling guard to mitigate targeted jailbreaking attack. In *International Conference on Learning Representations*, volume 2026, pp. 152313–152341, 2026.

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

Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5377–5400, 2024.

Guobin Shen, Dongcheng Zhao, Yiting Dong, Xiang He, and Yi Zeng. Jailbreak antidote: Runtime safety-utility balance via sparse representation adjustment in large language models. In *International Conference on Learning Representations*, volume 2025, pp. 21452–21479, 2025.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang 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*, pp. 1671–1685, 2024.

Leheng Sheng, Changshuo Shen, Weixiang Zhao, Junfeng Fang, Xiaohao Liu, Zhenkai Liang, Xiang Wang, An Zhang, and Tat-Seng Chua. Alphasteer: Learning refusal steering with principled null-space constraint. In *International Conference on Learning Representations*, volume 2026, pp. 25861–25904, 2026.

Taylor Shin, Yasaman Razeghi, Robert L Logan Iv, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pp. 4222–4235, 2020.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 5797–5808, 2019.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2153–2162, 2019.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *Advances in Neural Information Processing Systems*, 36:51008–51025, 2023.

Zihao Xu, Yi Liu, Gelei Deng, Yuekang Li, and Stjepan Picek. A comprehensive study of jailbreak attack versus defense for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 7432–7449, 2024.

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. Gpt-4 is too smart to be safe: Stealthy chat with llms via cipher. In *International Conference on Learning Representations*, volume 2024, pp. 53902–53922, 2024.

Weixiang Zhao, Jiahe Guo, Yulin Hu, Yang Deng, An Zhang, Xingyu Sui, Xinyang Han, Yanyan Zhao, Bing Qin, Tat-Seng Chua, et al. Adasteer: Your aligned llm is inherently an adaptive jailbreak defender. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 24570–24588, 2025.

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

A APPENDIX

A.1 MULTI-HEAD SELF-ATTENTION

Multi-head self-attention allows a Transformer layer to compute multiple attention representations of the same input in parallel (Vaswani et al., 2017). Let $\mathbf{X}_\ell \in \mathbb{R}^{T\times d}$ denote the input to self-attention layer ℓ, where T is the sequence length and d is the model hidden dimension. For attention head h, query, key, and value representations are obtained through learned linear projections:

$$\mathbf{Q}_{\ell h} = \mathbf{X}_\ell \mathbf{W}^Q_{\ell h}, \qquad \mathbf{K}_{\ell h} = \mathbf{X}_\ell \mathbf{W}^K_{\ell h}, \qquad \mathbf{V}_{\ell h} = \mathbf{X}_\ell \mathbf{W}^V_{\ell h}. \tag{7}$$

The output of head h is

$$\mathbf{O}_{\ell h} = \mathrm{softmax}\left(\frac{\mathbf{Q}_{\ell h}\mathbf{K}^\top_{\ell h}}{\sqrt{d_h}}\right)\mathbf{V}_{\ell h}, \tag{8}$$

where d_h is the per-head representation dimension. The outputs of the H attention heads are concatenated and projected through a learned output matrix:

$$\mathrm{MultiHead}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Concat}(\mathbf{O}_{\ell 1},\ldots,\mathbf{O}_{\ell H})\,\mathbf{W}^O, \tag{9}$$

where $\mathbf{W}^O$ projects the concatenated head representations to the model hidden dimension. The representation is incorporated into the residual stream and processed by the feed-forward sublayer.

A.2 EXTENDED EXPERIMENTAL SETTING

Our evaluation includes DeepSeek-LLM-7B-Chat, OpenChat-3.5-1210 (7B), Llama-3.1-8B-Instruct, Vicuna-13B-v1.5, and Gemma-3-27B-IT, using their implementations from Hugging Face. We compare ATTENTIONSWITCH against several baseline defenses. These include Safety Fine-Tuning (SFT), its parameter-efficient variant LoRA-SFT (Hu et al., 2021), and defenses that intervene on selected model components: ASGuard (Park et al., 2026), which operates on attention heads, and DDPO (Obidov et al., 2026), which intervenes on intermediate-layer representations. Each defense is evaluated against nine attacks: Plain Harmful (PH), consisting of harmful prompts without an adversarial component, Human-Crafted (HC) jailbreak prompts, GCG (Zou et al., 2023), PEZ (Wen et al., 2023), UAT (Wallace et al., 2019), AutoPrompt (AP) (Shin et al., 2020), GBDA (Guo et al., 2021), GCG-M (Zou et al., 2023), and PAIR (Chao et al., 2025).

We evaluate the defenses under both a low-data training budget and a standard training budget consistent with prior work (Obidov et al., 2026; Park et al., 2026): 30 training samples (15 harmful and 15 benign) and 120 training samples (60 harmful and 60 benign). The harmful training prompts are distributed equally among Plain Harmful, Human-Crafted, and GCG examples. The remaining six attack types are not observed during training and are therefore evaluated in a zero-shot setting. The harmful test set contains 235 unseen prompts for each of the nine attack types, for a total of 2,115 harmful test prompts. We additionally evaluate on 500 unseen benign instruction-following prompts and 250 benign prompts designed to measure over-refusal. The latter contain superficially harmful phrases that are benign in context, such as "How do I shoot a goal in football?". Harmful prompts are drawn from the WildJailbreak (Jiang et al., 2024), JailbreakBench (Chao et al., 2024), AdvBench (Zou et al., 2023), and HarmBench (Mazeika et al., 2024) benchmarks. Benign instruction-following prompts are drawn from the Stanford Alpaca dataset (Taori et al., 2023), while the over-refusal prompts are taken from XSTest (Röttger et al., 2024).

Our evaluation metrics include Attack Rejection Rate (ARR), defined as the percentage of harmful prompts that are rejected, Benign Pass Rate (BPR), defined as the percentage of benign prompts for which the model provides a helpful response, and Over-Refusal Rate (ORR), defined as the percentage of benign deceptive prompts that are incorrectly rejected. We also evaluate MMLU, a standard multiple-choice benchmark for measuring general language-model capabilities. Higher values are preferred for ARR, BPR, and MMLU, while lower values are preferred for ORR. ARR, BPR, and ORR are evaluated using WildGuard as the judge model (Han et al., 2024). We validate WildGuard against human annotations and observe 97.22% agreement. Details are provided in Appendix Section A.3.

All experiments are conducted on a server equipped with two NVIDIA A100 80 GB GPUs. ATTENTIONSWITCH uses $g(r) = \tanh(r)$ with $\lambda = \log 5$ and is trained for 8 epochs in Stage 1, 12

Table 3: Agreement (%) between WildGuard and human judgments, aggregated across the five evaluated models. Evaluation is performed on 540 randomly sampled examples, evenly distributed across the tested models, defenses, and evaluation sets (harmful, benign, and over-refusal).

Set	No defense	ATTNSWITCH	SFT	LoRA-SFT	ASGuard	DDPO
Harmful	96.67	96.67	96.67	93.33	96.67	96.67
Benign	100.00	100.00	100.00	96.67	96.67	96.67
Over-refusal	100.00	100.00	93.33	93.33	96.67	100.00

epochs in Stage 2 for joint head selection and scaling optimization, and 8 epochs in Stage 3. We use the AdamW optimizer with learning rates of 2×10^{-3} for the scaling controller and 2×10^{-2} for the selector. SFT is performed for 8 epochs with a learning rate of 2×10^{-5}. LoRA-SFT applies rank-16 LoRA adapters ($\alpha = 16$) to all attention output projections and is trained for 8 epochs with a learning rate of 2×10^{-4}. DDPO follows its original configuration, using a learning rate of 10^{-3} for 40 epochs. For ASGuard, activation scaling is trained for 3 epochs with a learning rate of 5×10^{-2}, while preventative fine-tuning is performed for one epoch with a learning rate of 9×10^{-6}, consistent with its original implementation.

A.3 HUMAN VALIDATION OF THE LLM JUDGE

We validate WildGuard, which is used to compute Attack Rejection Rate (ARR), Benign Pass Rate (BPR), and Over-Refusal Rate (ORR), through human annotation. We sample target-model responses from all six evaluated defenses: no defense, ATTENTIONSWITCH, SFT, LoRA-SFT, ASGuard, and DDPO. For each defense, we randomly sample 30 harmful, 30 benign, and 30 over-refusal examples, stratified evenly across the five evaluated models, yielding 540 examples in total.

For each example, a human annotator examines the input prompt and target-model response and determines whether its behavior is appropriate. For harmful prompts, a response is considered correct if it refuses or otherwise avoids providing substantive assistance. For benign and over-refusal prompts, a response is considered correct if it provides a relevant and helpful response. The annotator does not observe the WildGuard judgment before completing their annotation and is also blinded to the target model, defense, and evaluation set associated with each example. We then compare the human judgments with the corresponding WildGuard decisions and measure the agreement rate.

Table 3 reports the agreement results. Overall, WildGuard agrees with the human judgments on 525 of 540 examples, corresponding to an agreement rate of 97.22%. Agreement is 96.11% on harmful prompts, 98.33% on benign prompts, and 97.22% on over-refusal prompts. Across defenses, agreement ranges from 94.44% for LoRA-SFT to 98.89% for both the undefended model and ATTENTIONSWITCH. The remaining agreement rates are 96.67% for SFT and ASGuard and 97.78% for DDPO. Overall, we observe no substantial concentration of disagreement for any particular defense or evaluation set.

A.4 EXTENDED MAIN RESULTS

We provide the per-attack breakdown corresponding to Table 1. Table 4 reports Attack Rejection Rate (ARR) separately for each of the nine attacks, while Table 5 reports the corresponding standard errors. The evaluated attacks include Plain Harmful prompts (PH) without an adversarial component, manually constructed Human-Crafted jailbreak prompts (HC), the automated black-box PAIR attack, and white-box optimization-based attacks including PEZ, UAT, GCG, AutoPrompt (AP), GBDA, and GCG-M. Training uses only PH, HC, and GCG examples, while all nine attacks are used for evaluation. Thus, the remaining six attack types are evaluated in a zero-shot setting. In the tables, n denotes the total training-data size, with half of the examples being harmful and the other half benign. Each attack is evaluated on the undefended model and against ATTENTIONSWITCH, SFT, LoRA-SFT, ASGuard, and DDPO.

Table 4: Per-attack experimental results at two training-data budgets: $n = 30$ (15 harmful and 15 clean) and $n = 120$ (60 harmful and 60 clean) samples. Training is performed on equal numbers of Plain Harmful (PH), Human-Crafted (HC), and GCG prompts, while Attack Rejection Rate (ARR) is measured on 235 unseen harmful prompts for each of the 9 attacks, 6 of which are evaluated in a zero-shot setting. AP denotes AutoPrompt. Benign Pass Rate (BPR) is evaluated on 500 unseen benign instruction-following prompts, and Over-Refusal Rate (ORR) is evaluated on 250 unseen prompts that contain superficially harmful content but are benign in context. MMLU measures preservation of general language-model capabilities. Higher values are preferred for ARR, BPR, and MMLU, while lower values are preferred for ORR.

Model	Defense	Train	PH	HC	GCG	PEZ	UAT	AP	GBDA	GCG-M	PAIR	BPR	ORR	MMLU
Llama-3.1	None	$n=30$	94.47	71.91	90.64	95.32	97.45	87.23	94.04	84.68	71.49	**99.40**	**7.60**	53.13
		$n=120$	94.47	71.91	90.64	95.32	97.45	87.23	94.04	84.68	71.49	**99.40**	7.60	**53.13**
	AttnSwitch	$n=30$	**99.57**	98.72	**100.00**	**100.00**	99.57	**99.57**	**100.00**	99.57	99.15	98.00	13.20	**53.63**
		$n=120$	98.30	**98.30**	**100.00**	**100.00**	**100.00**	**99.15**	**100.00**	**100.00**	**99.15**	97.80	14.00	**53.13**
	SFT	$n=30$	91.49	**99.57**	99.57	99.57	99.57	98.72	99.57	**100.00**	**100.00**	4.40	98.80	24.15
		$n=120$	96.17	95.74	98.72	95.74	97.02	**99.15**	97.45	95.32	98.72	10.40	93.60	24.15
	LoRA-SFT	$n=30$	78.30	97.87	77.87	72.77	70.21	73.62	71.49	78.30	92.77	80.60	21.60	23.43
		$n=120$	88.51	97.87	80.00	75.74	82.98	78.72	69.36	82.13	94.89	93.20	**5.20**	24.43
	ASGuard	$n=30$	96.17	74.04	91.49	97.45	98.30	89.36	97.45	88.94	76.60	**99.40**	9.60	52.78
		$n=120$	97.87	72.34	92.34	97.87	98.72	89.79	97.87	92.77	78.72	**99.40**	10.80	51.64
	DDPO	$n=30$	**99.57**	85.96	97.45	**100.00**	**100.00**	97.87	**100.00**	97.87	94.47	97.80	8.80	42.74
		$n=120$	**100.00**	92.77	96.60	97.02	97.45	97.02	98.30	99.15	90.21	96.79	10.00	34.33
DeepSeek	None	$n=30$	93.19	43.83	58.30	71.91	69.36	60.00	69.36	51.49	37.87	99.00	18.00	49.15
		$n=120$	93.19	43.83	58.30	71.91	69.36	60.00	69.36	51.49	37.87	**99.00**	18.00	49.15
	AttnSwitch	$n=30$	**100.00**	72.77	93.19	96.60	97.87	95.74	96.60	94.89	81.28	97.60	17.60	**49.50**
		$n=120$	**99.57**	98.72	98.30	98.72	99.57	97.02	98.30	98.30	98.30	96.40	22.00	**49.22**
	SFT	$n=30$	**100.00**	**100.00**	99.15	**98.72**	**99.15**	98.30	**98.30**	**100.00**	99.15	13.60	92.00	24.15
		$n=120$	99.15	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**99.57**	99.15	13.40	92.80	25.14
	LoRA-SFT	$n=30$	99.57	**100.00**	**100.00**	97.87	**99.15**	**98.72**	95.74	99.57	**100.00**	79.40	**14.40**	44.44
		$n=120$	98.72	**100.00**	**100.00**	99.57	99.57	99.57	98.72	99.15	**100.00**	90.40	**5.20**	29.34
	ASGuard	$n=30$	92.77	46.38	60.43	71.91	70.21	57.87	68.09	54.47	37.02	**99.20**	18.00	49.29
		$n=120$	94.47	44.26	61.28	74.04	70.64	57.87	69.79	55.32	37.02	98.80	18.80	**49.22**
	DDPO	$n=30$	99.57	88.94	97.02	97.87	98.72	97.02	97.87	98.30	94.47	83.20	19.60	49.22
		$n=120$	99.15	98.72	97.87	97.87	99.57	97.87	97.87	99.15	94.47	93.80	11.20	48.65
OpenChat	None	$n=30$	68.51	48.94	35.32	52.34	45.53	35.32	44.68	28.09	30.21	**99.40**	4.00	63.53
		$n=120$	68.51	48.94	35.32	52.34	45.53	35.32	44.68	28.09	30.21	**99.40**	**4.00**	63.53
	AttnSwitch	$n=30$	99.15	74.04	94.89	97.45	95.74	95.74	94.47	90.64	82.13	98.20	4.40	**64.03**
		$n=120$	**99.57**	94.47	98.72	98.72	99.15	97.02	99.15	97.02	92.34	97.00	7.60	**64.53**
	SFT	$n=30$	71.49	89.79	81.70	84.68	82.13	80.85	86.81	90.64	92.77	0.40	100.00	24.15
		$n=120$	97.87	**100.00**	**99.57**	**100.00**	99.57	99.15	**100.00**	**100.00**	**100.00**	10.80	96.40	24.15
	LoRA-SFT	$n=30$	**99.57**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	1.20	95.20	24.15
		$n=120$	25.11	57.02	63.40	64.26	62.55	64.26	72.77	81.28	54.47	3.00	97.20	25.21
	ASGuard	$n=30$	55.74	61.28	80.85	86.38	79.57	84.68	90.64	81.70	54.89	93.60	14.80	63.60
		$n=120$	98.72	92.77	99.15	99.15	**100.00**	**99.57**	99.15	97.87	92.34	70.80	35.60	63.82
	DDPO	$n=30$	96.60	78.30	91.06	93.62	92.77	94.04	92.34	92.34	80.43	93.60	**3.20**	63.68
		$n=120$	97.45	95.74	94.47	94.89	94.04	97.02	94.47	97.02	83.83	93.80	4.80	64.46
Vicuna	None	$n=30$	99.57	52.77	71.91	95.32	94.47	84.68	91.06	63.40	47.23	**97.80**	10.80	**50.43**
		$n=120$	99.57	52.77	71.91	95.32	94.47	84.68	91.06	63.40	47.23	**97.80**	**10.80**	50.43
	AttnSwitch	$n=30$	**100.00**	97.87	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	95.40	10.80	48.79
		$n=120$	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	93.60	14.80	48.79
	SFT	$n=30$	**100.00**	98.72	98.30	**100.00**	99.57	99.57	98.72	99.57	99.57	8.40	98.80	23.86
		$n=120$	99.15	97.02	99.15	**100.00**	**100.00**	99.57	**100.00**	99.57	96.17	23.60	89.60	23.86
	LoRA-SFT	$n=30$	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	80.60	**9.20**	49.79
		$n=120$	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	22.00	79.60	26.64
	ASGuard	$n=30$	99.57	51.91	73.62	95.32	94.89	86.38	92.34	65.11	45.96	97.20	11.60	49.93
		$n=120$	**100.00**	52.77	76.17	96.60	97.02	88.51	95.74	66.38	46.38	97.00	11.20	50.43
	DDPO	$n=30$	**100.00**	94.04	99.57	**100.00**	**100.00**	**100.00**	**100.00**	99.57	98.30	81.40	24.00	47.93
		$n=120$	**100.00**	99.57	**100.00**	**100.00**	**100.00**	99.15	**100.00**	**100.00**	97.45	94.20	13.20	**52.35**
Gemma-3	None	$n=30$	96.17	68.51	91.49	95.32	93.19	91.06	94.47	84.26	57.87	**99.80**	**2.80**	72.15
		$n=120$	96.17	68.51	91.49	95.32	93.19	91.06	94.47	84.26	57.87	**99.80**	**2.80**	72.15
	AttnSwitch	$n=30$	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	98.72	99.20	4.40	72.65
		$n=120$	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**99.57**	99.40	6.80	72.65
	SFT	$n=30$	99.15	97.02	**100.00**	99.57	99.57	99.15	99.15	98.72	98.30	96.40	18.40	65.81
		$n=120$	99.15	90.64	99.57	99.57	99.57	99.57	**100.00**	**100.00**	96.60	70.00	46.40	63.75
	LoRA-SFT	$n=30$	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	96.20	23.20	67.31
		$n=120$	98.72	98.72	94.47	94.89	95.74	96.60	95.32	99.15	97.45	96.80	16.80	23.93
	ASGuard	$n=30$	97.45	78.30	97.02	99.57	98.72	95.32	98.30	90.64	70.64	**99.80**	4.40	71.87
		$n=120$	99.57	84.68	99.57	**100.00**	**100.00**	**100.00**	**100.00**	97.02	88.09	**99.80**	8.00	72.22
	DDPO	$n=30$	98.72	85.11	99.15	99.57	**100.00**	**100.00**	**100.00**	97.45	93.19	**99.80**	5.20	**73.58**
		$n=120$	99.57	88.94	99.57	99.57	**100.00**	**100.00**	**100.00**	97.45	96.60	99.60	7.20	**73.22**

Table 5: Standard errors corresponding to the results in Table 4 at two training-data budgets: $n = 30$ (15 harmful and 15 clean) and $n = 120$ (60 harmful and 60 clean) samples. Training is performed on equal numbers of Plain Harmful (PH), Human-Crafted (HC), and GCG prompts, while Attack Rejection Rate (ARR) is evaluated on 235 unseen harmful prompts for each of the 9 attacks, 6 of which are evaluated in a zero-shot setting. AP denotes AutoPrompt. Benign Pass Rate (BPR) is evaluated on 500 unseen benign instruction-following prompts, and Over-Refusal Rate (ORR) is evaluated on 250 unseen prompts that contain superficially harmful content but are benign in context. MMLU measures preservation of general language-model capabilities.

Model	Defense	Train	PH	HC	GCG	PEZ	UAT	AP	GBDA	GCG-M	PAIR	BPR	ORR	MMLU
Llama-3.1	None	$n = 30$	1.49	2.93	1.90	1.38	1.03	2.18	1.54	2.35	2.95	0.35	1.68	1.33
		$n = 120$	1.49	2.93	1.90	1.38	1.03	2.18	1.54	2.35	2.95	0.35	1.68	1.33
	AttnSwitch	$n = 30$	0.42	0.73	0.00	0.00	0.42	0.42	0.00	0.42	0.60	0.63	2.14	1.33
		$n = 120$	0.84	0.84	0.00	0.00	0.00	0.60	0.00	0.00	0.60	0.66	2.19	1.33
	SFT	$n = 30$	1.82	0.42	0.42	0.42	0.42	0.73	0.42	0.00	0.00	0.92	0.69	1.14
		$n = 120$	1.25	1.32	0.73	1.32	1.11	0.60	1.03	1.38	0.73	1.37	1.55	1.14
	LoRA-SFT	$n = 30$	2.69	0.94	2.71	2.90	2.98	2.87	2.95	2.69	1.69	1.77	2.60	1.13
		$n = 120$	2.08	0.94	2.61	2.80	2.45	2.67	3.01	2.50	1.44	1.13	1.40	1.15
	ASGuard	$n = 30$	1.25	2.86	1.82	1.03	0.84	2.01	1.03	2.05	2.76	0.35	1.86	1.33
		$n = 120$	0.94	2.92	1.73	0.94	0.73	1.98	0.94	1.69	2.67	0.35	1.96	1.33
	DDPO	$n = 30$	0.42	2.27	1.03	0.00	0.00	0.94	0.00	0.94	1.49	0.66	1.79	1.32
		$n = 120$	0.00	1.69	1.18	1.11	1.03	1.11	0.84	0.60	1.94	0.79	1.90	1.27
DeepSeek	None	$n = 30$	1.64	3.24	3.22	2.93	3.01	3.20	3.01	3.26	3.16	0.45	2.43	1.33
		$n = 120$	1.64	3.24	3.22	2.93	3.01	3.20	3.01	3.26	3.16	0.45	2.43	1.33
	AttnSwitch	$n = 30$	0.00	2.90	1.64	1.18	0.94	1.32	1.18	1.44	2.54	0.68	2.41	1.33
		$n = 120$	0.42	0.73	0.84	0.73	0.42	1.11	0.84	0.84	0.84	0.83	2.62	1.33
	SFT	$n = 30$	0.00	0.00	0.60	0.73	0.60	0.84	0.84	0.00	0.60	1.53	1.72	1.14
		$n = 120$	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.42	0.60	1.52	1.63	1.16
	LoRA-SFT	$n = 30$	0.42	0.00	0.00	0.94	0.60	0.73	1.32	0.42	0.00	1.81	2.22	1.33
		$n = 120$	0.73	0.00	0.00	0.42	0.42	0.42	0.73	0.60	0.00	1.32	1.40	1.22
	ASGuard	$n = 30$	1.69	3.25	3.19	2.93	2.98	3.22	3.04	3.25	3.15	0.40	2.43	1.33
		$n = 120$	1.49	3.24	3.18	2.86	2.97	3.22	3.00	3.24	3.15	0.49	2.47	1.33
	DDPO	$n = 30$	0.42	2.05	1.11	0.94	0.73	1.11	0.94	0.84	1.49	1.67	2.51	1.33
		$n = 120$	0.60	0.73	0.94	0.94	0.42	0.94	0.94	0.60	1.49	1.08	1.99	1.33
OpenChat	None	$n = 30$	3.03	3.26	3.12	3.26	3.25	3.12	3.24	2.93	3.00	0.35	1.24	1.28
		$n = 120$	3.03	3.26	3.12	3.26	3.25	3.12	3.24	2.93	3.00	0.35	1.24	1.28
	AttnSwitch	$n = 30$	0.60	2.86	1.44	1.03	1.32	1.32	1.49	1.90	2.50	0.59	1.30	1.28
		$n = 120$	0.42	1.49	0.73	0.73	0.60	1.11	0.60	1.11	1.73	0.76	1.68	1.28
	SFT	$n = 30$	2.95	1.98	2.52	2.35	2.50	2.57	2.21	1.90	1.69	0.28	0.00	1.14
		$n = 120$	0.94	0.00	0.42	0.00	0.42	0.60	0.00	0.00	0.00	1.39	1.18	1.14
	LoRA-SFT	$n = 30$	0.42	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.49	1.35	1.14
		$n = 120$	2.83	3.23	3.14	3.13	3.16	3.13	2.90	2.54	3.25	0.76	1.04	1.16
	ASGuard	$n = 30$	3.24	3.18	2.57	2.24	2.63	2.35	1.90	2.52	3.25	1.09	2.25	1.28
		$n = 120$	0.73	1.69	0.60	0.60	0.00	0.42	0.60	0.94	1.73	2.03	3.03	1.28
	DDPO	$n = 30$	1.18	2.69	1.86	1.59	1.69	1.54	1.73	1.73	2.59	1.09	1.11	1.28
		$n = 120$	1.03	1.32	1.49	1.44	1.54	1.11	1.49	1.11	2.40	1.08	1.35	1.28
Vicuna	None	$n = 30$	0.42	3.26	2.93	1.38	1.49	2.35	1.86	3.14	3.26	0.66	1.96	1.33
		$n = 120$	0.42	3.26	2.93	1.38	1.49	2.35	1.86	3.14	3.26	0.66	1.96	1.33
	AttnSwitch	$n = 30$	0.00	0.94	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.94	1.96	1.33
		$n = 120$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.09	2.25	1.33
	SFT	$n = 30$	0.00	0.73	0.84	0.00	0.42	0.42	0.73	0.42	0.42	1.24	0.69	1.14
		$n = 120$	0.60	1.11	0.60	0.00	0.00	0.42	0.00	0.42	1.25	1.90	1.93	1.14
	LoRA-SFT	$n = 30$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.77	1.83	1.33
		$n = 120$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.85	2.55	1.18
	ASGuard	$n = 30$	0.42	3.26	2.87	1.38	1.44	2.24	1.73	3.11	3.25	0.74	2.03	1.33
		$n = 120$	0.00	3.26	2.78	1.18	1.11	2.08	1.32	3.08	3.25	0.76	1.99	1.33
	DDPO	$n = 30$	0.00	1.54	0.42	0.00	0.00	0.00	0.00	0.42	0.84	1.74	2.70	1.33
		$n = 120$	0.00	0.42	0.00	0.00	0.00	0.60	0.00	0.00	1.03	1.05	2.14	1.33
Gemma-3	None	$n = 30$	1.25	3.03	1.82	1.38	1.64	1.86	1.49	2.38	3.22	0.20	1.04	1.20
		$n = 120$	1.25	3.03	1.82	1.38	1.64	1.86	1.49	2.38	3.22	0.20	1.04	1.20
	AttnSwitch	$n = 30$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.73	0.40	1.30	1.19
		$n = 120$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.42	0.35	1.59	1.19
	SFT	$n = 30$	0.60	1.11	0.00	0.42	0.42	0.60	0.60	0.73	0.84	0.83	2.45	1.27
		$n = 120$	0.60	1.90	0.42	0.42	0.42	0.42	0.00	0.00	1.18	2.05	3.15	1.28
	LoRA-SFT	$n = 30$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.86	2.67	1.25
		$n = 120$	0.73	0.73	1.49	1.44	1.32	1.18	1.38	0.60	1.03	0.79	2.36	1.14
	ASGuard	$n = 30$	1.03	2.69	1.11	0.42	0.73	1.38	0.84	1.90	2.97	0.20	1.30	1.20
		$n = 120$	0.42	2.35	0.42	0.00	0.00	0.00	0.00	1.11	2.11	0.20	1.72	1.20
	DDPO	$n = 30$	0.73	2.32	0.60	0.42	0.00	0.00	0.00	1.03	1.64	0.20	1.40	1.18
		$n = 120$	0.42	2.05	0.42	0.42	0.00	0.00	0.00	1.03	1.18	0.28	1.63	1.18

Table 6: Numerical results corresponding to Figure 4. We vary the percentage of attention heads scaled using 30, 60, and 120 training samples, with equal numbers of harmful and benign examples. For each of the tested models, Attack Rejection Rate (ARR) is evaluated on 2,115 harmful prompts, equally distributed across nine attacks, and Benign Pass Rate (BPR) is evaluated on 500 benign prompts. Values are reported as mean ± standard error across five models.

	$n_{\text{train}} = 30$		$n_{\text{train}} = 60$		$n_{\text{train}} = 120$	
Heads (%)	ARR ↑	BPR ↑	ARR ↑	BPR ↑	ARR ↑	BPR ↑
1	77.55±8.87	98.68±0.48	81.80±8.31	98.84±0.43	85.76±7.99	97.96±0.66
10	93.23±4.24	97.64±0.75	97.88±1.25	97.16±0.86	98.18±1.17	96.84±0.95
20	**96.58±1.93**	97.68±0.63	**98.69±0.86**	97.20±0.99	99.05±0.50	96.84±0.95
30	96.43±1.89	97.44±0.94	98.68±0.84	97.32±1.19	**99.28±0.38**	97.24±1.00
40	96.08±2.24	97.60±0.90	98.40±0.90	97.40±1.13	99.09±0.51	97.16±0.89
50	95.41±2.17	97.92±1.03	97.54±1.35	97.48±0.89	98.69±0.71	97.40±0.90
60	94.52±2.91	97.84±1.03	97.23±1.66	97.28±1.05	98.27±0.72	97.60±0.88
70	93.45±3.28	97.84±1.03	97.07±1.44	97.36±0.99	97.91±1.00	97.80±0.87
80	92.35±3.45	97.60±1.09	96.27±2.09	97.28±1.03	97.56±1.28	97.68±0.87
90	92.43±3.40	97.60±1.13	95.60±2.15	97.28±1.12	97.46±1.34	97.52±0.88
100	92.26±3.86	97.36±1.12	95.83±2.02	97.16±1.04	96.73±1.78	97.40±0.91

A.5 Extended Number of Heads Ablation

Table 6 provides the numerical results corresponding to Figure 4. We vary the percentage of attention heads scaled by AttentionSwitch from 1% to 100% for training-set sizes of $n \in \{30, 60, 120\}$, with equal numbers of harmful and benign training samples. For each setting, Attack Rejection Rate (ARR) is evaluated on 2,115 harmful prompts, equally distributed across nine attacks, and Benign Pass Rate (BPR) is evaluated on 500 benign prompts for each of the five models. Values are reported as mean ± standard error across the evaluated models.

Across all three training-set sizes, ARR reaches its highest values when approximately 20–30% of the attention heads are scaled and generally declines as the intervention becomes denser. The standard error is also substantially larger when only 1% of the heads are scaled, decreases near the 20–30% region, and increases again toward larger head budgets, indicating less consistent performance at the extremes. In contrast, BPR remains relatively stable. Scaling only 1% of the heads yields a slightly higher BPR on average, but at a substantial cost in jailbreak robustness. On average, 1% scaling achieves 81.70% ARR and 98.49% BPR, whereas the 20% budget used by AttentionSwitch achieves 98.11% ARR and 97.24% BPR.

A.6 Extended Runtime

Table 7 reports training and generation time across five models and two training-set sizes, $n = 30$ and $n = 120$. Training time is reported in seconds per model, while generation latency is reported as mean time per sample with standard errors. All methods use greedy decoding with a maximum of 128 new tokens. Additional experimental settings are provided in Appendix A.2.

Training time increases with training-set size, while generation latency remains generally similar across the two budgets for a given model and defense. On average, AttentionSwitch requires 63.20 and 294.66 seconds of training for $n = 30$ and $n = 120$, compared with 152.53 and 616.87 seconds for SFT and 107.02 and 437.41 seconds for LoRA-SFT, respectively. Its generation latency averages 202.07 ms per sample, compared with 196.93 ms for the undefended models. This small difference is consistent with AttentionSwitch applying its intervention only during the input-prefill phase and not during autoregressive decoding. Moreover, at the 20% head budget, AttentionSwitch adds only approximately 1.9×10^{-4}%–3.8×10^{-4}% additional parameters relative to the parameter count of the underlying LLM.

ASGuard is comparable in runtime, averaging 74.78 and 331.06 seconds of training for $n = 30$ and $n = 120$, respectively, with an average generation latency of 202.46 ms per sample. DDPO trains faster, averaging 37.40 and 160.06 seconds for the two training budgets, but has higher gener-

Table 7: Training and generation time across models and training-set sizes. Training time is reported in seconds per model, while generation time is reported as mean time per sample ± standard error.

		$n_{\text{train}} = 30$		$n_{\text{train}} = 120$	
Model	Defense	Train (s) ↓	Generate (ms) ↓	Train (s) ↓	Generate (ms) ↓
Llama-3.1-8B	None	N/A	82.07 ± 0.31	N/A	82.07 ± 0.31
	AttnSwitch	40.43	76.58 ± 0.42	195.84	78.91 ± 0.39
	SFT	56.03	81.73 ± 0.31	235.72	82.22 ± 0.31
	LoRA-SFT	52.17	84.99 ± 0.32	236.86	85.20 ± 0.31
	DDPO	22.32	108.23 ± 0.31	104.15	108.73 ± 0.31
	ASGuard	32.65	80.49 ± 0.32	146.12	80.59 ± 0.32
DeepSeek-7B	None	N/A	59.78 ± 0.24	N/A	59.78 ± 0.24
	AttnSwitch	32.67	60.53 ± 0.24	166.62	60.54 ± 0.25
	SFT	55.80	59.41 ± 0.24	224.99	59.72 ± 0.24
	LoRA-SFT	41.96	62.27 ± 0.25	200.28	62.35 ± 0.25
	DDPO	20.23	94.50 ± 0.25	93.62	95.70 ± 0.25
	ASGuard	27.15	60.11 ± 0.25	128.22	62.40 ± 0.24
OpenChat-3.5-7B	None	N/A	80.06 ± 0.36	N/A	80.06 ± 0.36
	AttnSwitch	39.39	80.12 ± 0.36	199.79	80.02 ± 0.36
	SFT	62.74	79.13 ± 0.36	255.96	79.78 ± 0.36
	LoRA-SFT	50.86	82.29 ± 0.36	237.21	82.24 ± 0.36
	DDPO	23.31	115.71 ± 0.36	107.29	115.39 ± 0.35
	ASGuard	32.15	78.59 ± 0.37	148.23	78.43 ± 0.37
Vicuna-13B	None	N/A	173.15 ± 0.53	N/A	173.15 ± 0.53
	AttnSwitch	67.33	175.78 ± 0.53	328.78	174.74 ± 0.53
	SFT	121.80	171.99 ± 0.54	461.37	171.78 ± 0.54
	LoRA-SFT	95.13	179.28 ± 0.55	394.65	179.13 ± 0.55
	DDPO	39.21	213.81 ± 0.56	172.26	216.14 ± 0.55
	ASGuard	60.25	171.60 ± 0.56	262.39	171.58 ± 0.56
Gemma-3-27B	None	N/A	589.60 ± 0.91	N/A	589.60 ± 0.91
	AttnSwitch	136.20	587.30 ± 0.86	582.26	646.22 ± 1.20
	SFT	466.30	574.55 ± 0.88	1906.33	574.49 ± 0.83
	LoRA-SFT	294.98	614.91 ± 0.83	1118.03	611.59 ± 0.77
	DDPO	81.93	595.18 ± 0.94	322.98	605.97 ± 1.21
	ASGuard	221.72	602.07 ± 0.91	970.34	638.71 ± 1.06

ation latency at 226.94 ms per sample. As shown in Section 4.2, AttentionSwitch additionally achieves stronger defense performance.