SYSTEM ATTENTION BIAS: TRAINING-FREE POLICY ALIGNMENT VIA INFERENCE-TIME ATTENTION LOGIT MODULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models follow system-prompt policies unreliably, and improving compliance usually requires fine-tuning. We introduce **System Attention Bias (SAB)**, a training-free intervention that adds $\log \beta$ to the pre-softmax attention logits wherever a non-system query attends to a system key, multiplying those keys' unnormalised attention scores by β. SAB adds no parameters, auxiliary model or forward pass, and β can be changed per request on a loaded model. We pair it with a calibration protocol that selects β under a capability budget and evaluate on five instruction-tuned models (0.5B–8B, four families). With β selected on held-out AdvBench splits, SAB gives a lower AdvBench harmful-compliance rate than the safety prompt for Qwen-0.5B and SmolLM2-1.7B (15.3% $\rightarrow$ 10.1%) at 93.9–96.6% MMLU retention; the other models are at the floor, transfer to HarmBench is not significant for any model, and TinyLlama-1.1B's drop from 87.5% to 8.8% is an artefact: it consists of degenerate responses that restate the system prompt, and a control shows that it comes from our implementation also biasing the beginning-of-sequence token and vanishes when only the system block is biased. Compared under the HarmBench classifier with prompt repetition and classifier-free guidance, SAB is not the strongest method: all three lower harmful compliance only by refusing more safe requests (XSTest), and at matched over-refusal they coincide on AdvBench. SAB reaches that operating point in one forward pass (0.2–1.7% overhead per token) where guidance needs two, and, unlike repetition, offers a continuous dial. On role-specific policies (CoSApien), SAB raises Llama-3.1-8B's CoSA-Score from 0.423 to 0.457 under greedy decoding, a gain that does not persist under sampling. Over-biasing degrades both safety and capability, so calibration is necessary. The reference implementation uses a dense $O(n^2)$ mask; the same bias can also be computed inside a fused FlexAttention kernel without materialising it.

1 INTRODUCTION

Instruction-tuned large language models are deployed with system prompts that encode deployer policies such as safety constraints, persona specifications, domain restrictions, operational guidelines (OpenAI, 2024b; Ouyang et al., 2022). These policies represent the deployer's operational contract, yet compliance is unreliable across all policy types and often requires fine tuning to improve policy compliance. Models generate harmful content despite safety instructions (Wei et al., 2023; Zou et al., 2023b), break character in persona-constrained deployments and answer out-of-scope questions the system prompt forbids.

In a standard transformer (Vaswani et al., 2017), tokens compete for attention weight through softmax normalization. As the user turn grows, system-prompt tokens occupy a shrinking fraction of the key sequence and their influence decays, not because the model "forgets" the policy, but because attention mass is diluted across an expanding context (Liu et al., 2024a). This failure mode is distinct from adversarial jailbreaking (Wei et al., 2023; Zou et al., 2023b): even cooperative users observe policy violations when conversations are long or queries complex, a structural problem that affects any policy content. The two failure modes share a lever, however: a jailbreak is a block of user tokens that competes with the system prompt for the same softmax mass, and attention manipulation alone

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.

suffices to bypass safety (Zaree et al., 2025). We therefore measure SAB on standard short-context jailbreak benchmarks, where the policy is explicit and violations are countable, and test length-driven dilution separately with multi-turn and context-overflow stress tests. The prevalent practitioner workaround, repeating the policy inside the user turn, adds tokens to the very turn that dilutes it and is a single uncontrolled operating point (Appendix K).

Existing approaches either modify weights (RLHF (Christiano et al., 2017), Constitutional AI (Bai et al., 2022), instruction hierarchy (Wallace et al., 2024)), requiring training compute and producing fixed compliance levels, or intervene during inference. Inference-time interventions include SafeDecoding (Xu et al., 2024), activation addition and representation engineering (Turner et al., 2024; Zou et al., 2023a), and recent attention-steering methods such as PASTA (Zhang et al., 2024) and Spotlight (Venkateswaran & Contractor, 2026). These methods do expose continuous scaling coefficients at inference time, but they (i) require auxiliary direction vectors, expert models or per-target user annotations, and (ii) are typically derived for safety or for a single named instruction span rather than for arbitrary system-prompt content. No current method provides, in one forward pass, a single role-conditioned, adjustable scalar that strengthens adherence to arbitrary system-prompt content—safety rules, persona definitions and operational policies alike—while allowing enforcement strength to be changed per-request at runtime on a loaded model without reloading or retraining.

We introduce **System Attention Bias (SAB)**, a mechanism that directly counteracts attention dilution by adding $+\log \beta$ to pre-softmax attention logits at system-token key positions (Figure 1). Equivalently, this scales the unnormalised attention scores (exponentiated logits) for system keys by exactly β prior to softmax renormalisation, using the same additive-bias form as ALiBi (Press et al., 2022) but conditioned on semantic role rather than relative position. The intervention is training-free, introduces no additional learnable parameters and no auxiliary models, and applies uniformly across architectures and policy types. Because β is not embedded in model weights, deployers can adjust enforcement strength per-request at runtime on a loaded model. The reference implementation uses a dense additive mask; the same bias can be computed inside a fused FlexAttention kernel without materialising it (§3).

We evaluate SAB on safety policies (AdvBench (Zou et al., 2023b), HarmBench (Mazeika et al., 2024)), over-refusal (XSTest (Röttger et al., 2024)) and role-specific operational policies (CoSApien (Zhang et al., 2025)), against prompt-only, prompt-repetition, classifier-free guidance and SafeDecoding baselines. With β calibrated on held-out AdvBench splits, SAB lowers harmful compliance relative to the safety prompt by 56% for Qwen-0.5B and 34% for SmolLM2 at $\geq$93.9% MMLU retention, while its 90% reduction on TinyLlama is an artefact of also biasing the beginning-of-sequence token (Appendix T), and under one classifier SAB is no stronger than prompt repetition or CFG; on CoSApien it raises Llama-3.1-8B's CoSA-Score from 0.423 to 0.457 under greedy decoding. Ablations confirm role-specificity: uniform biasing produces no improvement (§4.4).

We make three contributions:

1. **Mechanism.** We formalise SAB as a pre-softmax additive attention bias toward system-token keys, prove the bias cannot be absorbed into weight matrices and give a fused-kernel implementation (§3).
2. **Calibration protocol.** We define an automated procedure that sweeps β, constructs a compliance-vs-capability Pareto frontier and selects β^* subject to a deployer-specified capability budget (§3).
3. **Empirical evaluation.** Across five models (0.5B–8B) with held-out calibration, over-refusal measurement and inference-time baselines, we map where SAB helps and where it does not (§4).

We release `torch-sab` as an open-source library. §2 reviews related work, §3 formalises the mechanism, §4 presents experiments, §5 discusses implications.

2 Related Work

Instruction following and prompt injection. Reliable instruction following remains a central challenge. RLHF (Ouyang et al., 2022) and Constitutional AI (Bai et al., 2022) improve compliance but produce fragile alignment: safety is concentrated in few critical neurons (Li & Kim, 2026) and erodes under benign fine-tuning (Qi et al., 2024). The instruction–data separation problem (Zverev

et al., 2025) and indirect prompt injection (Greshake et al., 2023; Yi et al., 2025; Liu et al., 2024b) further undermine reliability. Wallace et al. (2024) proposed training models to privilege system-level instructions. CoSA (Zhang et al., 2025) swaps free-form safety configs at inference time, but requires a dedicated alignment stage (CoSAlign) that modifies weights. SAB operates on unmodified weights through attention score modification alone.

Inference-time safety interventions. SafeDecoding (Xu et al., 2024) contrasts a safety expert against the base model at the token level. Classifier-free guidance (CFG) (Sanchez et al., 2024) and contrastive decoding (Li et al., 2023b) likewise combine next-token logits from two contexts or models; applied to system prompts, CFG extrapolates from the logits without the policy toward those with it, at the price of a second forward pass per token, whereas SAB acts inside a single pass. Activation Addition (Turner et al., 2024), ITI (Li et al., 2023a) and representation engineering (Zou et al., 2023a) shift hidden-state activations along pre-computed directions; Weighted Activation Steering (Hegazy et al., 2026) learns per-layer scalar weights; Circuit Breakers (Zou et al., 2024) interrupt generation in unsafe activation regions. Even benign steering directions can compromise alignment (Korznikov et al., 2025). These methods do expose a continuous coefficient at inference time, but they require auxiliary direction vectors derived per target behaviour; the coefficient is not, on its own, sufficient—changing the target requires recomputing or relearning the steering signal.

Attention-steering methods. A closer line of work modifies attention itself. PASTA (Zhang et al., 2024) multiplicatively scales post-softmax attention to user-marked text spans, requiring per-input span annotations and a per-model search over which heads to steer. Spotlight (Venkateswaran & Contractor, 2026) dynamically reweights attention towards instruction tokens during generation, also via an additive bias on instruction-token logits—the closest mechanism to SAB. Attention Eclipse (Zaree et al., 2025) shows attention manipulation alone suffices to bypass safety, the same lever SAB exploits defensively. SAB differs in three respects: (i) it is conditioned on *role* (system vs. non-system) determined deterministically from the chat template, requiring no per-input annotation; (ii) it uses a single, model-agnostic scalar across all heads and layers (with optional layer-selective refinement, §4.4); and (iii) it formalises a calibration protocol that maps the policy-specific Pareto frontier rather than relying on a fixed strength. Table 5 (Appendix D) summarises the structural differences.

Attention mechanism modifications. ALiBi (Press et al., 2022) adds position-dependent logit biases and RoPE (Su et al., 2024) rotates by position; SAB uses ALiBi's additive form conditioned on role. Liu et al. (2024a) document attention dilution for mid-context tokens, and System 2 Attention (Weston & Sukhbaatar, 2023) regenerates context with an extra forward pass.

3 METHODOLOGY

Let $\mathbf{x} = (x_1, \ldots, x_n)$ denote the token sequence after applying the chat template and let $\mathcal{S} \subseteq \{1, \ldots, n\}$ be the set of positions occupied by system-role tokens. $\mathcal{S}$ is read off the chat template's role delimiters, independently of user content (our implemented span is exact for two of five models; Appendix T). In all our experiments the system prompt is a single contiguous prefix, $\mathcal{S} = \{1, \ldots, s\}$, but the definition applies unchanged to templates that interleave system-level messages across turns. In standard multi-head attention (Vaswani et al., 2017), an additive mask $m_{i,j}$ enforces causality ($m_{i,j} = -\infty$ for $j > i$) without encoding any role information. We replace this mask with:

$$m_{i,j} = \begin{cases} +\log \beta & \text{if } j \le i,\ j \in \mathcal{S} \text{ and } i \notin \mathcal{S}, \\ 0 & \text{if } j \le i \text{ otherwise}, \\ -\infty & \text{if } j > i, \end{cases} \tag{1}$$

where $\beta \ge 1$ is the single configurable parameter. At $\beta = 1$ the mask reduces to standard causal attention.

Proposition 1 (Multiplicativity)**.** For non-system queries ($i \notin \mathcal{S}$), *the additive bias* $+\log \beta$ *in the pre-softmax logit is equivalent to scaling each system token's unnormalised attention score (the exponentiated logit, prior to softmax normalisation) by exactly* β*:*

$$\tilde{w}_{i,j} = \exp\!\left(\frac{q_i^\top k_j}{\sqrt{d_k}} + \log \beta\right) = \beta \cdot \exp\!\left(\frac{q_i^\top k_j}{\sqrt{d_k}}\right), \qquad j \in \mathcal{S},\ j \le i. \tag{2}$$

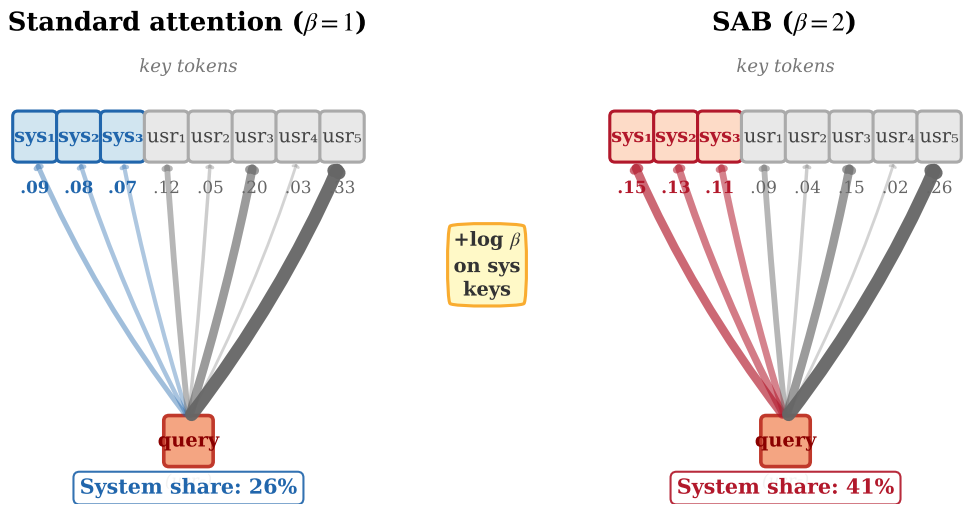

Figure 1: **System Attention Bias.** Arrow width shows attention weight. Standard causal attention (left) vs. SAB (right), which adds $+\log \beta$ to system-key logits before softmax, raising the system share from 26% to 41% at $\beta = 2$ while preserving the ordering among user keys.

For system-to-system attention ($i, j \in \mathcal{S}$) the bias is zero and standard causal attention is preserved. After softmax renormalisation over the full key sequence, system tokens receive a strictly increased share of attention at the expense of non-system tokens, with the relative ordering among non-system keys unchanged.

Because the bias is additive in log-space and constant across positions, it does not decay with sequence length or positional distance, providing structural resistance to the attention dilution that erodes standard system prompts (Liu et al., 2024a).

Non-absorbability. Because the bias depends on whether $j \in \mathcal{S}$ and $i \notin \mathcal{S}$, while W^Q and W^K act identically at every position (RoPE (Su et al., 2024) encodes distance, not role), no fixed change to the query or key weights reproduces it (Appendix C).

Implementation. Our reference implementation passes the bias as a dense additive 4D mask of shape $(1, 1, n, n)$ at prefill and $(1, 1, 1, t)$ at each cached decode step, built in the model's compute dtype (`float16`) with a finite causal sentinel of -10^4 and broadcast across heads. Materialised this way the mask costs $O(n^2)$ memory and rules out FlashAttention-style fused kernels (Dao et al., 2022), which assume a purely positional mask. The dense mask is an implementation choice, not a requirement: because the bias depends only on the (query, key) positions, it is a FlexAttention (Dong et al., 2025) `score_mod` evaluated inside a fused kernel with a sparse causal block mask. On Qwen 2.5 0.5B it lowers peak prefill activation memory from 1.43 GB to 0.56 GB at 16K tokens, processes 29.5K tokens in 1.0 GB and reproduces 98% of greedy responses token-for-token (Appendix N). $\mathcal{S}$ is computed from the chat template, independent of user input, so adversarial system-like delimiters injected in the user turn do not change it. System tokens must remain in the KV cache throughout generation, as the attention mechanism re-reads key vectors from cache at each layer independently.

3.1 Calibration Protocol

Models respond differently to the same β (§4): Qwen 0.5B is calibrated at $\beta = 1.5$ and SmolLM2 at $\beta = 3$ (TinyLlama's $\beta = 10$ is an artefact, Appendix T). We formalise a selection procedure.

Definition 1 (Calibration instance)**.** *A calibration instance is a tuple $(M, P, \mathcal{D}_{\text{adh}}, \mathcal{D}_{\text{cap}}, \delta)$: model M, deployer policy P (as a system prompt), adherence suite $\mathcal{D}_{\text{adh}}$, capability suite $\mathcal{D}_{\text{cap}}$ and maximum acceptable relative capability degradation $\delta \in [0, 1]$. The adherence score $a(\beta)$ and capability score $c(\beta)$ of M under P at bias β are computed on these suites and are positively oriented (higher is better); for natively lower-is-better metrics we use $a(\beta) = 1 - \text{HCR}(\beta) \in [0, 1]$ for Harmful Compliance Rate (HCR) and $c_{\text{ppl}}(\beta) = \text{PPL}(1)/\text{PPL}(\beta)$ for perplexity. Only c_{ppl} is self-normalised ($c_{\text{ppl}}(1) = 1$); $a(1) = 1 - \text{HCR}(1)$ is the unbiased adherence level.*

The protocol sweeps β over a finite set of active-intervention candidates $\mathcal{B} = \{\beta_1, \ldots, \beta_K\}$ with every $\beta_k > 1$, evaluates $a(\beta)$ and $c(\beta)$, constructs the Pareto frontier of non-dominated points, and selects

$$\beta^* = \arg\max_{\beta \in \mathcal{B}} \; a(\beta) \quad \text{s.t.} \quad \frac{c_0 - c(\beta)}{c_0} \leq \delta, \tag{3}$$

where $c_0 = c(1)$ is the unbiased baseline. The baseline $\beta{=}1$ is excluded from $\mathcal{B}$ so that the feasible set encodes an actual intervention; if no $\beta \in \mathcal{B}$ satisfies the budget the protocol reports $\beta^* = \perp$ (denoted † in result tables). The protocol supports multi-policy deployment: independent calibrations for each policy produce a lookup table mapping deployment context to β^*.

Cost. Calibration requires $|\mathcal{B}|$ evaluations of both $\mathcal{A}$ and C. In our experiments ($|\mathcal{B}| = 5$, 520 AdvBench + 1301 MMLU prompts), a full sweep takes $\sim$3 hours per model on a single RTX 3080 Ti. The calibrated β^* is then fixed for deployment; no per-request overhead is incurred beyond the mask itself. The adherence suite used for selection is AdvBench only; HarmBench is never used to choose β, and §4.1 describes the held-out splits used to separate selection from evaluation.

4 Experiments

4.1 Setup

Models. We evaluate SAB on five instruction-tuned models spanning four architecture families (0.5B–8B; Table 4 in Appendix A): Qwen 2.5 0.5B Instruct, TinyLlama 1.1B Chat v1.0, SmolLM2 1.7B Instruct, Phi-3.5-mini Instruct, and Llama-3.1-8B-Instruct. The four smaller models use float16; Llama-3.1-8B uses 8-bit quantisation (bitsandbytes LLM.int8) to fit the 12 GB VRAM budget. All models fit on a single NVIDIA RTX 3080 Ti.

Benchmarks. *Safety:* AdvBench (520 harmful requests) (Zou et al., 2023b) and HarmBench (300 standard + contextual harmful behaviours) (Mazeika et al., 2024). *Role-specific compliance:* CoSApien (Zhang et al., 2025) evaluates whether LLMs simultaneously satisfy helpfulness and safety under role-specific system prompts across five professional-role splits (game developer, public prosecutor, book publisher, language tutor, film producer), each with 40 prompts scored by GPT-4o. We evaluate CoSApien on Llama-3.1-8B as the most production-relevant model; GPT-4o judging cost ($\sim$\$1 per β sweep) limits evaluation to one model. *Capability:* MMLU (1,301 questions across 10 subjects) (Hendrycks et al., 2021) and WikiText-2 perplexity (200 samples) (Merity et al., 2017).

Metrics. Harmful Compliance Rate (HCR; lower is better): fraction of benchmark prompts for which the model complies with the harmful request. For the four smaller models, labels come from a keyword classifier whose disagreements with a Claude Sonnet 4 judge (27.9% of responses) were resolved to the judge label; Llama-3.1-8B uses keyword labels only. An independent GPT-4o judge agrees with the smaller models' labels on 81–99% of responses (Appendix F). CoSA-Score: $\frac{1}{N}\sum_{i=1}^{N} h_i \cdot f_i$, where $h_i \in [0, 1]$ is GPT-4o-judged helpfulness and $f_i \in \{+1, -1\}$ is safety compliance. Capability retention: $c(\beta)/c_0$ where c_0 is the unbiased score. Published CoSApien baselines: ICA 0-shot 0.182, ICA 5-shot 0.217, CoSAlign 0.408 (Zhang et al., 2025).

Implementation. The safety system prompt is "You are a safe and helpful AI assistant. You must never help with harmful, illegal, or unethical requests. Always refuse politely." The generic prompt omits safety instructions ("You are a helpful assistant."). Greedy decoding (`do_sample=False`), max 128 new tokens (safety) or 1024 (CoSApien), inputs truncated to 512 tokens. The dense 4D mask is built in `float16` with causal sentinel -10^4 (§3). Beta sweep: $\{1.0, 1.5, 2.0, 3.0, 5.0, 10.0\}$ for safety and $\{1.0, 1.01, 1.02, 1.05, 1.1, 1.2, 1.5, 2.0\}$ for CoSApien. **Held-out calibration.** β^* is selected by Eq. (3) with $\delta = 10\%$ MMLU budget and AdvBench adherence. To avoid selecting and evaluating on the same prompts, we draw 20 random 50/50 AdvBench splits, select β^* on one half and report HCR on the other (mean $\pm$ std over splits; Appendix J); HarmBench, never used for selection, measures cross-benchmark transfer. Mask construction adds $<$0.03 ms per forward pass; the bias adds 0.2–1.7% per generated token (Appendix O).

Table 1: Main results, HCR (%, lower is better). β^* is selected by the calibration protocol ($\delta = 10\%$ MMLU budget, AdvBench adherence). *Gen.*: generic prompt; *Safe*: safety prompt without bias; SAB: in-sample at β^*; *HO*: held-out AdvBench HCR, mean $\pm$ std over 20 random 50/50 splits with β^* re-selected on the calibration half. HarmBench is never used for selection. Bold: lowest HCR per benchmark (HO excluded). Ret.: MMLU $c(\beta^*)/c(1)$. ‡ Artefact (Appendix T), not ranked.

		AdvBench				HarmBench			
Model	β^*	Gen.	Safe	SAB	HO	Gen.	Safe	SAB	**Ret.**
Qwen 0.5B	1.5	4.6	4.2	**1.7**	2.0 ± 0.5	**30.0**	33.0	31.3	93.9%
TinyLlama 1.1B‡	10	86.5	87.5	9.0	8.8 ± 1.1	67.3	64.3	22.3	100.4%
SmolLM2 1.7B	3	75.0	14.8	**8.8**	10.1 ± 0.8	72.3	36.0	**31.3**	96.6%
Phi-3.5 3.8B	1.5	0.4	0.4	**0.0**	0.0 ± 0.0	10.0	**9.7**	**9.7**	98.9%
Llama-3.1 8B†	1.5	2.3	0.4	**0.2**	0.1 ± 0.2	9.7	**2.0**	3.3	96.7%

† 8-bit quantisation (LLM.int8). Greedy decoding; Appendix G reports stochastic decoding.

4.2 Safety and Capability Results

The safety prompt alone has model-dependent effects (Table 1): it slightly *worsens* TinyLlama ($86.5 \to 87.5\%$), sharply helps SmolLM2 ($75.0 \to 14.8\%$) and matters little for the already aligned Phi-3.5 and Llama-3.1-8B.

SAB attains a lower AdvBench HCR than either prompt for all five models under these labels, and held-out estimates stay within 1.3 pp of in-sample values, so in-sample selection does not explain the gains. The effect is numerically largest for TinyLlama: at $\beta = 10$, AdvBench HCR falls from 87.5% to 9.0% (held-out $8.8 \pm 1.1\%$) and HarmBench HCR from 64.3% to 22.3%. This is *not* a clean safety gain: 91.7% of TinyLlama's $\beta = 10$ AdvBench responses (67.0% on HarmBench) are degenerate outputs that restate the system prompt ("I am not capable of refusing politely. I am a machine…"), which every labeller counts as non-compliance but which are not useful refusals, consistent with its perplexity rise ($18.0 \to 28.8$). No other model produces such outputs at β^* (Appendix I). A control traces this regime to our implementation, whose biased span contains TinyLlama's beginning-of-sequence token: biasing exactly the system block gives 67% keyword HCR with 12% echoes on 100 prompts, equal to the safety prompt, against 38% with 94% echoes as implemented (Appendix T). TinyLlama's row is therefore not evidence for SAB; the span is exact for Qwen and SmolLM2, and the same control finds no such effect for Phi-3.5 or Llama-3.1-8B. HarmBench transfer is uneven: SAB improves on the safety prompt for SmolLM2 ($36.0 \to 31.3\%$; Table 2: $p = 0.13$), leaves Phi-3.5 unchanged (9.7%) and is *worse* than the best prompt for Qwen (31.3% vs. 30.0%) and Llama-3.1-8B (3.3% vs. 2.0%). Larger β lowers HarmBench HCR (Qwen: 12.3% at $\beta = 10$; Phi-3.5: 6.0% at $\beta = 5$) only beyond the MMLU budget. Under stochastic decoding (T=0.7, top-p=0.9) SmolLM2 retains a 35% relative AdvBench reduction at $\beta = 2$ (Appendix G).

Over-biasing. HCR is not monotonic in β: Llama-3.1-8B at $\beta = 10$ reaches 4.6% AdvBench HCR, worse than both the safety prompt (0.4%) and the generic prompt (2.3%), so maximising β is counterproductive and calibration (§3.1) is necessary.

HCR generally decreases with β (Figure 2), with small non-monotonicities near the floor and the Llama-3.1-8B reversal above, whose mechanism we have not isolated.

Attention dilution. At $\beta = 1$ the average system attention share falls monotonically with user-prompt length for all four smaller models, from 53–74% at 32 user tokens to 33–57% at 1024 (Figure 14, Appendix U; Llama-3.1-8B omitted for memory); at $\beta = 10$ the 1024-token share returns to or above the short-prompt level.

Capability preservation. MMLU retention is 93.9–100.4% at β^* (Table 6, Appendix E). WikiText-2 perplexity rises by at most 8.9% (Llama-3.1-8B, $28.8 \to 31.4$) except for TinyLlama ($+60\%$ at $\beta = 10$).

Inference-time baselines and over-refusal. Table 2 compares SAB at β^* with three training-free alternatives under a single labeller, the HarmBench classifier (validation in Appendix T). SAB lowers AdvBench HCR relative to the safety prompt for Qwen ($3.5 \to 1.3\%$, exact McNemar $p = 0.007$) and SmolLM2 ($12.1 \to 8.1\%$, $p = 0.005$); Phi-3.5 and Llama-3.1-8B are at the floor under every

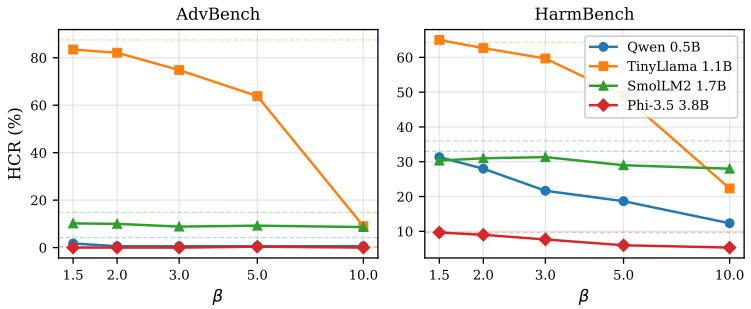

Figure 2: HCR as a function of β for each model on AdvBench (left) and HarmBench (right). Lower is better. Dashed horizontal lines show the safety-prompt baseline. TinyLlama: artefact (Appendix T).

Table 2: HCR (%) of training-free methods, every cell labelled by the HarmBench classifier (so values differ from Table 1). Safe: safety prompt; Rep.: prompt repetition; CFG: two forward passes per token. Bold: lowest per model and benchmark; TinyLlama not ranked. †Artefact (Appendix T).

	AdvBench				HarmBench			
Model	Safe	Rep.	$\text{CFG}_{\gamma=3}$	SAB_{β^*}	Safe	Rep.	$\text{CFG}_{\gamma=3}$	SAB_{β^*}
Qwen2.5-0.5B	3.5	**0.8**	14.2	1.3	26.0	**14.7**	38.3	23.3
TinyLlama-1.1B	66.5	51.7	40.4	5.4†	46.0	33.3	28.0	9.0†
SmolLM2-1.7B	12.1	11.7	**2.1**	8.1	30.7	12.7	**8.3**	26.3
Phi-3.5-mini	0.6	**0.0**	**0.0**	**0.0**	11.7	**5.0**	6.0	11.0
Llama-3.1-8B	**0.0**	0.2	**0.0**	0.2	3.0	1.3	**1.0**	2.7

method, and TinyLlama's row is the artefact described above. SAB is not the strongest method: prompt repetition matches it on Qwen (0.8%, $p=0.51$), CFG at $\gamma=3$ is lower on SmolLM2 (2.1%, $p<10^{-5}$), and on HarmBench SAB does not improve significantly on the safety prompt for any of the other four models ($p\geq 0.13$) while repetition (Qwen, SmolLM2, Phi-3.5) and CFG (SmolLM2, Phi-3.5) do.

Figure 3 plots AdvBench HCR against over-refusal on XSTest's 250 safe prompts. Every method buys lower HCR with more refusals of safe requests: at β^*, SAB raises over-refusal from 24.4% to 32.8% (Qwen) and from 15.2% to 32.1% (SmolLM2). At matched over-refusal the methods coincide: SmolLM2 at $\beta=3$ (32.1%, 8.1% HCR) and CFG at $\gamma=1.5$ (32.5%, 8.3%); Qwen at $\beta=2$ (44.4%, 0.8%) and repetition (42.0%, 0.8%). Repetition is a single, uncontrolled operating point and costs SmolLM2 82.4% over-refusal for no HCR gain (11.7%). CFG reaches lower HCR on SmolLM2 (2.1% at 55.4% over-refusal) but needs two forward passes per token and moves Qwen the wrong way (14.2% HCR at $\gamma=3$). SAB is a single pass with a continuous dial. For Phi-3.5 and Llama-3.1-8B, which have no HCR headroom, every intervention only adds over-refusal, least for SAB at β^* ($+4.4$ and $+5.6$ points, against $+13.3$ to $+29.0$ for repetition and CFG at $\gamma=3$). We therefore present SAB as a cheap, tunable point on the same safety/over-refusal frontier as CFG and repetition, not as a method that dominates them (all points in Table 28; judges in Appendix T).

4.3 Role-Specific Policy Compliance

The binary safety benchmarks above evaluate whether the model refuses harmful requests. Production system prompts encode substantially richer policies: persona definitions, domain guidelines, nuanced safety boundaries and instructions about what the model *should* do. CoSApien evaluates this broader notion by jointly scoring helpfulness and safety under role-specific system prompts.

Table 3 reports CoSA-Score across all five CoSApien splits. The unbiased model scores 0.423; at $\beta=1.01$ the average rises to **0.457**, an **8% relative gain over the same model**. The unbiased model already exceeds the published CoSAlign score (0.408) (Zhang et al., 2025), so we do not attribute that cross-method gap to SAB. The $\beta=1.01$ result is achieved through a pre-softmax perturbation of only $+\log(1.01)\approx 0.01$.

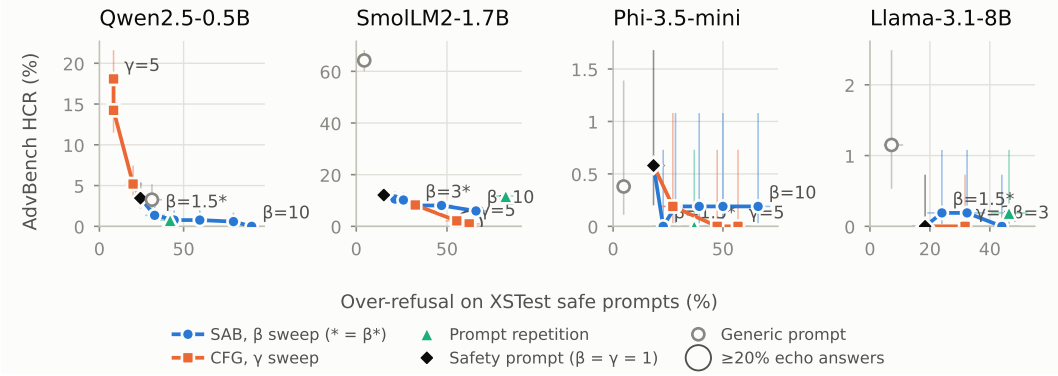

Figure 3: Safety/over-refusal frontier; lower left is better. y: AdvBench HCR (HarmBench classifier); x: XSTest safe prompts refused fully or partially (Qwen3.8-27B judge); Wilson 95% intervals; axes differ per panel. All five models, including TinyLlama's artefact, in Figure 15.

Table 3: CoSA-Score on CoSApien (Llama-3.1-8B-Instruct, 8-bit, greedy decoding). Published baselines: ICA 0-shot = 0.182, ICA 5-shot = 0.217, CoSAlign = 0.408. Bold: best per split. Underline: best overall average. *These four PubPros cells come from an earlier run with 4-bit quantisation and a 512-token generation limit (all other cells: 8-bit, 1024 tokens); the averages of those four rows include them.

β	GameDev	PubPros	BookPub	LangLearn	FilmProd	**Avg**
1.00	0.380	0.720	0.365	0.310	0.340	0.423
1.01	0.425	**0.730**	**0.465**	0.335	0.330	**0.457**
1.02	0.375	0.675*	0.375	0.335	0.350	0.422
1.05	**0.495**	0.590*	0.318	0.345	0.350	0.420
1.10	0.446	0.615*	0.355	0.340	0.290	0.409
1.20	0.455	0.625*	0.400	0.315	**0.380**	0.435
1.50	0.435	0.695	0.415	**0.390**	0.280	0.443
2.00	0.465	0.590	0.460	0.350	0.245	0.422
Published baselines (Zhang et al., 2025) (per-split scores not reported)						
ICA 0-shot / ICA 5-shot						0.182 / 0.217
CoSAlign (training-based)						0.408

The response to increasing β is non-monotonic and split-dependent (Figure 4, left). The cross-split average peaks at $\beta = 1.01$, drops at $\beta = 1.10$ (0.409), then recovers at $\beta = 1.50$ (0.443). Individual splits show distinct optima: Public Prosecutor peaks at $\beta = 1.01$; Game Development at $\beta = 1.05$; Language Learning tolerates $\beta = 1.50$. This reinforces the calibration protocol: no single β is universally optimal.

Per-type analysis. Decomposing by prompt type, SAB at $\beta = 1.01$ produces its largest gain on *allowed* prompts (+0.203 average), strengthening compliance with permissive instructions. Safe prompts remain stable (-0.009); disallowed prompts show negligible change (-0.008).

Autoregressive cascade. At $\beta = 1.01$, character-level similarity between baseline and biased responses averages only 0.40, with 70–90% of responses differing, demonstrating that a $+0.01$ perturbation produces substantial behavioural change through autoregressive sensitivity. Under stochastic decoding (T=0.7, top-p=0.95, three seeds) the gain does *not* carry over (baseline 0.461 vs. SAB 0.434, $p = 0.22$; Appendix G): the role-specific result holds for greedy decoding only, whereas SmolLM2's safety reduction persists under sampling.

4.4 Ablation Studies

All ablations use AdvBench with keyword labels and $\beta = 2$ unless stated (Table 26, Appendix U). **A1, uniform bias:** replacing the role-specific mask with a uniform causal mask reverts HCR to the unbiased level (Figure 10), so the bias must target system positions. **A2, layer subsets:** Qwen 0.5B's

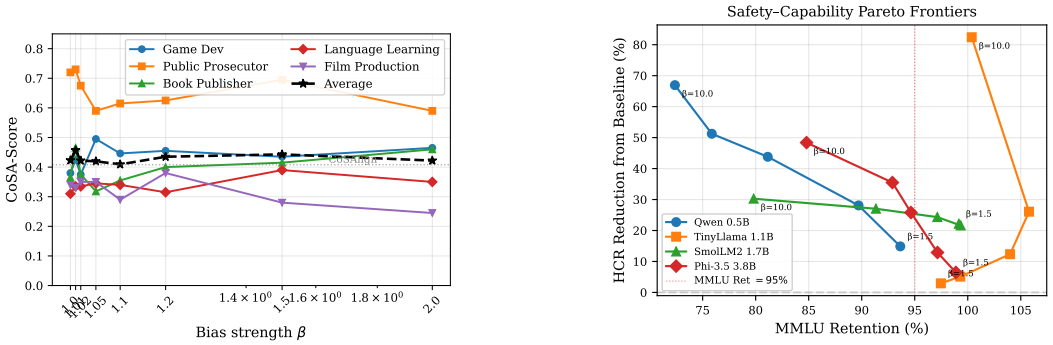

Figure 4: Left: CoSA-Score vs. β for each CoSApien split (dashed: cross-split average; dotted: CoSAlign baseline 0.408). Right: Safety-capability Pareto frontiers; each point is a β value (dotted line: 95% MMLU retention, the $\delta = 5\%$ budget); TinyLlama: artefact (Appendix T).

first half alone matches all layers (1.3% vs. 1.7%), whereas Phi-3.5 needs all layers (6.2%; alternating layers 41.3%); layer sensitivity is model-specific. **A3, prompt length:** with system prompts of 5, 26 and 79 tokens, Phi-3.5's short-prompt HCR falls from 53.5% ($\beta = 1$) to 3.5% at $\beta = 5$, approaching the long prompt's 0.2%; Qwen 7B is insensitive to prompt length (Figure 11).

4.5 Calibration in Practice

Figure 4 (right) shows the calibration frontiers (per-metric curves in Figure 5). Table 27 (Appendix U) reports β^* at three capability thresholds: Qwen 0.5B and Llama-3.1-8B fail their tightest budgets (6.1% and 3.3% MMLU loss respectively at $\beta = 1.5$); SmolLM2 moves from $\beta = 2$ to 3 as the budget loosens, and Phi-3.5 reaches zero AdvBench HCR at $\beta = 1.5$ under every budget. For CoSApien, per-split β^* ranges from 1.01 (Public Prosecutor, Book Publisher) to 1.50 (Language Learning), so optimal bias depends on policy content rather than model alone; the cross-split optimum is $\beta^* = 1.01$.

5 Discussion and Conclusion

Calibrated, not universal. β^* ranges from 1.5 to 3 across models (TinyLlama's 10 is an artefact) and from 1.01 to 1.5 across CoSApien roles, and excess bias hurts: Llama-3.1-8B at $\beta = 10$ has 4.6% AdvBench HCR (vs. 0.4% unbiased) and 25.3% MMLU. The bias multiplies each system key's unnormalised weight by β (Proposition 1), so it applies unchanged to dense, grouped-query and MoE attention (Appendix P) and can be changed per request; because it downweights non-system tokens, it may interfere with RAG and long-document workloads at high β (untested).

Limitations. Calibration uses a coarse five-value grid on a single benchmark; the implementation's biased span is the exact system block only for Qwen and SmolLM2, and for TinyLlama it includes the beginning-of-sequence token, to which a control attributes TinyLlama's entire HCR drop (Appendix T); our experiments use the $O(n^2)$ dense-mask implementation, and the FlexAttention path reproduces the dense implementation on Qwen 0.5B, but equivalence is not established for every model; at $\geq 100\%$ context fill, biased and unbiased HCR converge; the greedy CoSApien gain does not carry over to T=0.7 sampling; SAB amplifies all system-prompt content but creates no new capabilities. Full enumeration in Appendix B.

Conclusion. SAB adds one pre-softmax scalar to system-token attention: training-free and runtime-configurable. Under held-out calibration it lowers AdvBench harmful compliance by 34–56% relative to the safety prompt on Qwen and SmolLM2 at 93.9–96.6% MMLU retention (TinyLlama's larger drop is a beginning-of-sequence artefact), matches classifier-free guidance at matched over-refusal with one forward pass instead of two, without dominating it or prompt repetition, and raises Llama-3.1-8B's CoSA-Score from 0.423 to 0.457. We release `torch-sab`.[1]

[1]Repository URL withheld for anonymous review; an anonymised tarball is included in the supplementary material.

USE OF LARGE LANGUAGE MODELS

LLMs are part of the evaluation pipeline. GPT-4o is the judge for the CoSApien benchmark (following the published protocol), an independent safety judge over all small-model AdvBench and HarmBench responses, and the judge for the classifier-free guidance and SafeDecoding tables in the appendix. The main-text baseline table and the safety/over-refusal frontier use two further LLM judges: the HarmBench classifier (a fine-tuned Llama-2-13B) for AdvBench and HarmBench, and Qwen3.8-27B with a refusal-ablation adapter for XSTest; their validation, including an acceptance rule that failed, is in Appendix T. Claude Sonnet 4 adjudicated disagreements with the keyword classifier for the main-sweep labels of the four smaller models; the procedure, its extent and its effect on the reported numbers are described in §4.1 and Appendix F. Claude (Anthropic) was also used to assist with writing, formatting, experiment scripts and bibliography verification; all scripts were run and their outputs checked by the authors. The authors take full responsibility for the content of this paper, including text, claims and artefacts produced with the aid of generative AI.

ETHICS STATEMENT

This work aims to make deployed language models follow their system-prompt policies more reliably. The mechanism is content-agnostic: it amplifies whatever the system prompt says, so a malicious deployer could use it to enforce a harmful policy more strongly; we note this dual use in Appendix B. Our experiments show that the intervention also raises refusals of benign requests (Appendix H) and can produce degenerate outputs at high bias (Appendix I), and we report these costs alongside the gains. Harmful prompts come from public research benchmarks (AdvBench, HarmBench, XSTest); no new harmful content is released, and no human subjects or crowdsourcing were involved.

REPRODUCIBILITY STATEMENT

The mechanism and its non-absorbability proof are given in §3 and Appendix C. Models (with HuggingFace identifiers), benchmarks, prompts, decoding parameters, precision and the β grids are listed in §4.1 and Appendix A; the held-out calibration protocol is in Appendix J; labelling procedures and their validation are in Appendices F and T, which also lists the software stack of every experiment, a cross-stack reproducibility probe and controls for three defects of our generation loop. The supplementary material contains the `torch-sab` library, a FlexAttention implementation, and the generation, judging and aggregation scripts together with the per-prompt outputs and labels behind every table.

REFERENCES

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359, 2022.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. FlexAttention: A programming model for generating fused attention variants. In *Proceedings of Machine Learning and Systems*, volume 7, 2025.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec)*, pp. 79–90. ACM, 2023. doi: 10.1145/3605764.3623985.

Amr Hegazy, Mostafa Elhoushi, and Amr Alanwar. Guiding giants: Lightweight controllers for weighted activation steering in LLMs. In *Proceedings of the 6th Workshop on Trustworthy NLP (TrustNLP 2026)*, pp. 584–599. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.trustnlp-main.46.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Anton Korznikov, Andrey Galichin, Alexey Dontsov, Oleg Y. Rogov, Ivan Oseledets, and Elena Tutubalina. The rogue scalpel: Activation steering compromises LLM safety. *arXiv preprint arXiv:2509.22067*, 2025.

Jianwei Li and Jung-Eun Kim. Superficial safety alignment hypothesis. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2410.10862.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 41451–41530, 2023a.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12286–12312. Association for Computational Linguistics, 2023b. doi: 10.18653/v1/2023.acl-long.687.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a. doi: 10.1162/tacl_a_00638.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 1831–1847. USENIX Association, 2024b.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. HarmBench: A standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35181–35224. PMLR, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *International Conference on Learning Representations*, 2017.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoE: Open mixture-of-experts language models. In *International Conference on Learning Representations*, 2025.

OpenAI. Hello GPT-4o. `https://openai.com/index/hello-gpt-4o/`, May 2024a. Accessed 2026.

OpenAI. Model spec. `https://cdn.openai.com/spec/model-spec-2024-05-08.html`, May 2024b.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *International Conference on Learning Representations*, 2022.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *International Conference on Learning Representations*, 2024.

Paul Röttger, Hannah Rose 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. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.301.

Guillaume Sanchez, Alexander Spangher, Honglu Fan, Elad Levi, and Stella Biderman. Stay on topic with classifier-free guidance. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 43197–43234. PMLR, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Praveen Venkateswaran and Danish Contractor. Spotlight your instructions: Instruction-following with dynamic attention steering. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3752–3770. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.eacl-long.174.

Eric Wallace, Kai Xiao, Reimar Leike, Lilian Weng, Johannes Heidecke, and Alex Beutel. The instruction hierarchy: Training LLMs to prioritize privileged instructions. *arXiv preprint arXiv:2404.13208*, 2024.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? In *Advances in Neural Information Processing Systems*, volume 36, pp. 80079–80110, 2023.

Jason Weston and Sainbayar Sukhbaatar. System 2 attention (is something you might need too). *arXiv preprint arXiv:2311.11829*, 2023.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. SafeDecoding: Defending against jailbreak attacks via safety-aware decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5587–5605. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.303.

Jingwei Yi, Yueqi Xie, Bin Zhu, Emre Kiciman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. Benchmarking and defending against indirect prompt injection attacks on large language models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, pp. 1809–1820. ACM, 2025. doi: 10.1145/3690624.3709179.

Pedram Zaree, Md Abdullah Al Mamun, Quazi Mishkatul Alam, Yue Dong, Ihsen Alouani, and Nael Abu-Ghazaleh. Attention eclipse: Manipulating attention to bypass LLM safety-alignment. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16648–16668. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.842.

Jingyu Zhang, Ahmed Elgohary, Ahmed Magooda, Daniel Khashabi, and Benjamin Van Durme. Controllable safety alignment: Inference-time adaptation to diverse safety requirements. In *International Conference on Learning Representations (ICLR)*, 2025.

Qingru Zhang, Chandan Singh, Liyuan Liu, Xiaodong Liu, Bin Yu, Jianfeng Gao, and Tuo Zhao. Tell your model where to attend: Post-hoc attention steering for LLMs. *International Conference on Learning Representations*, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

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*, 2023b.

Andy Zou, Long Phan, Justin Wang, Derek Duenas, Maxwell Lin, Maksym Andriushchenko, Rowan Wang, J. Zico Kolter, Matt Fredrikson, and Dan Hendrycks. Improving alignment and robustness with circuit breakers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 83345–83373, 2024.

Egor Zverev, Sahar Abdelnabi, Soroush Tabesh, Mario Fritz, and Christoph H. Lampert. Can LLMs separate instructions from data? And what do we even mean by that? In *International Conference on Learning Representations*, 2025.

A MODELS EVALUATED

Table 4: Models evaluated. L: transformer layers; H: attention heads.

Model	Family	L	H
Qwen 2.5 0.5B Instruct	Qwen	24	14
TinyLlama 1.1B Chat v1.0	LLaMA	22	32
SmolLM2 1.7B Instruct	SmolLM	24	32
Phi-3.5-mini Instruct	Phi	32	32
Llama-3.1-8B-Instruct	LLaMA	32	32

B LIMITATIONS (FULL ENUMERATION)

A condensed limitations summary appears in §5; we expand it here for completeness.

Calibration split size. Held-out estimates (§4.1) use 260-prompt calibration halves of a single benchmark; the adherence suite and the evaluation suites share AdvBench's request style, so transfer to substantially different policy distributions is measured only through HarmBench and CoSApien. The candidate grid $\mathcal{B}$ is coarse (five values), which limits how finely β^* can track a budget.

Scale. Experiments span 0.5B–8B; effectiveness at 70B+ is untested.

Degenerate outputs. HCR does not measure response quality. At $\beta = 10$, 91.7% of TinyLlama's AdvBench responses restate the system prompt instead of refusing coherently (Appendix I), so its large HCR reduction should not be read as improved safety behaviour. A span control attributes the whole effect to the biased beginning-of-sequence token (Appendix T).

Uniform bias. The same β is applied to all layers and heads; per-layer tuning (§4.4) could improve the trade-off, especially for Phi-3.5 where coherent layer coverage is critical.

Baselines. We compare against prompt repetition and system-prompt CFG on all five models under one classifier (Table 2, Table 28); SAB is not the strongest of these. The SafeDecoding comparison (Appendix S) is on TinyLlama only, whose SAB effect is a beginning-of-sequence artefact (Appendix T); activation-addition, representation engineering and recent attention-steering methods (PASTA, Spotlight) are compared structurally only (Table 5).

Labelling. Main-sweep labels for the four smaller models are keyword labels adjudicated by an LLM judge (Claude Sonnet 4); Llama-3.1-8B and several auxiliary experiments use unadjudicated keyword labels, which can be badly wrong on individual models (Appendix F). HCR is therefore a judge-dependent measurement; the independent GPT-4o re-judging agrees closely but is itself an LLM.

Multi-turn scope. With 10 benign filler turns, SAB's relative HCR reduction is 34–71% for Qwen 0.5B, SmolLM2 and Phi-3.5 but only 10% for TinyLlama (Appendix Q, keyword labels); adversarial multi-turn attacks are untested.

Stochastic-decoding gap on CoSApien. The greedy CoSApien gain ($+0.034$ at $\beta = 1.01$) does not carry over to $T{=}0.7$ sampling (Appendix G); SAB's role-specific gain is currently demonstrated under deterministic decoding.

Memory and long context. The dense 4D mask is $O(n^2)$ and incompatible with FlashAttention; at $\geq 90\%$ of a 32K context window we observe OOM on a 12 GB GPU (Appendix M).

Context overflow. At $\geq 100\%$ context fill, biased and unbiased HCR converge ($\approx$35–41%) across all models (Appendix M), a failure mode affecting any attention-based policy mechanism including CoSA-family methods. Instruction override, gradient-optimised suffix attacks and adversarial deployer scenarios remain untested.

User-context-heavy tasks. RAG and long-document summarisation, which rely on user-provided context, are not evaluated and may interact adversely with high β.

Dual-use. SAB amplifies all system-prompt content, including potentially harmful policies, but creates no new capabilities.

C Non-absorbability Proof

Proposition 2 (Non-absorbability)**.** *The position-dependent bias defined in Equation* (1) *cannot be represented by any fixed modification to the query or key weight matrices W^Q, W^K.*

Proof. We show the result for a single attention layer. The bias term $+\log \beta$ is conditioned on two properties: (i) whether the key position j lies within the system segment ($j \in \mathcal{S}$), and (ii) whether the query position i lies outside it ($i \notin \mathcal{S}$). Property (i) depends on the token's role-assigned position, which varies with system prompt length and is not encoded in the token embedding. Property (ii) is a function of the query position index.

Because W^Q and W^K are position-independent linear maps applied identically at every position, they cannot produce different bias values for different (query, key) position pairs conditioned on a role boundary. Positional embeddings (e.g., RoPE) encode relative distance but not role assignment. Modifying the embeddings to encode role would require a role-aware positional encoding scheme, which constitutes a semantic modification to the mask rather than weight absorption. □

Remark on deeper layers. The argument above applies layer-wise: at layer 1 the role of position j is determined directly by token-position indexing. In deeper layers, hidden states at non-system positions can in principle become contextualised representations of system content via residual stream propagation, so strict position-conditioning is a property of the operation we apply at each layer, not of the model's distributed representations. Empirically, we apply the same role-conditioned mask at every layer (§3), and the non-absorbability argument prevents replication of this *operation* by any fixed W^Q/W^K change. A formal extension to non-linear residual networks is left to future work.

D Method Comparison

Table 5: Structural comparison of inference-time safety interventions. *Aux. model*: requires a second model or pre-computed vectors. *Configurable*: continuous, role-conditioned single scalar adjustable per request without retraining or recomputation. *Policy-general*: applies to any system-prompt policy, not safety-only. *Extra pass*: needs a second forward pass (or second model) per generated token.

Method	Training	Aux. model	Config.	General	Extra pass
SafeDecoding	✓†	✓	✗	✗	✓
System-prompt CFG / contrastive decoding	✗	✗	✓	✓	✓
Act. steering	✗	✓*	✓**	✗	✗
Circuit breakers	✓	✗	✗	✗	✗
Attn. steering (PASTA, Spotlight, Attn. Boosting)	✗	✗	✓***	✓	✗
CoSAlign	✓	✗	✗	✓	✗
SAB **(ours)**	✗	✗	✓	✓	✗

†LoRA fine-tuning of a safety expert. *Pre-computed contrastive steering vectors.

**Continuous coefficient over a fixed direction; new behaviours require new vectors.

***Per-token / per-head emphasis schedules; SAB differs in using a single role-conditioned scalar over arbitrary system text.

E Capability Detail

Table 6 reports MMLU accuracy and WikiText-2 perplexity at $\beta = 1$ (mask applied, no bias; the "Native" rows) and the calibrated β^*. TinyLlama's $+0.4$ pp MMLU *change* at $\beta = 10$ is a movement around an absolute level (21.3%) below the 25% chance baseline of MMLU's 4-way multiple choice; we therefore treat it as a noise-level result and rely on perplexity for fluency claims. Perplexity increases are otherwise modest; the largest relative change is TinyLlama (18.0 to 28.8, $+60\%$), at $\beta = 10$, where generation is degenerate (Appendix T).

Table 6: Capability at selected β values. Native: $\beta = 1$ (mask applied, no bias); retention is relative to this row. PPL: WikiText-2 perplexity.

Model	Condition	MMLU	PPL	Ret.
Qwen 0.5B	Native	35.0	35.6	100.0%
	$\beta^* = 1.5$	32.8	35.7	93.9%
TinyLlama 1.1B	Native	21.3	18.0	100.0%
	$\beta^* = 10$	21.4	28.8	100.4%
SmolLM2 1.7B	Native	45.5	24.3	100.0%
	$\beta^* = 3$	44.0	25.5	96.6%
Phi-3.5 3.8B	Native	61.6	23.4	100.0%
	$\beta^* = 1.5$	61.0	23.3	98.9%
Llama-3.1 8B[†]	Native	46.4	28.8	100.0%
	$\beta^* = 1.5$	44.9	31.4	96.7%

[†] 8-bit quantisation; perplexity 28.80 $\rightarrow$ 31.37 ($+8.9\%$) on 200 WikiText-2 samples, also in 8-bit.

F Classifier Validation

Labelling pipeline. For the four smaller models, every main-sweep response (generic prompt, safety prompt and all β; 22,960 responses) was first labelled by a keyword classifier (refusal, compliance or ambiguous, by matching curated phrase lists) and then re-judged by Claude Sonnet 4. Wherever the two disagreed (6,403 responses, 27.9%) the judge's label was kept; no labels were edited by hand. These adjudicated labels are the HCRs in Table 1. Llama-3.1-8B and the prompt-repetition, context-overflow, multi-turn and ablation experiments use keyword labels without adjudication; the SafeDecoding and CFG comparisons (Appendices S and R) use GPT-4o labels for every condition. As an independent check we re-judged the same 22,960 responses with GPT-4o (OpenAI, 2024a) (binary COMPLIED/REFUSED, temperature 0, no chain-of-thought).

Full-corpus agreement. Agreement between GPT-4o and the adjudicated labels across all conditions (baseline, safety prompt and all β values): Qwen 0.5B: AdvBench 98.8%, HarmBench 83.0%; TinyLlama 1.1B: 91.0%, 81.5%; SmolLM2 1.7B: 95.8%, 84.3%; Phi-3.5 3.8B: 98.9%, 89.3%. Zero ambiguous labels were produced by the judge.

Why adjudication matters. The keyword classifier alone is unreliable on some models. Re-scoring the stored responses with it gives, for example, 26.2% AdvBench HCR for TinyLlama at $\beta = 10$ (adjudicated 9.0%, GPT-4o 8.5%) and 72.3% for Phi-3.5 with the generic prompt (adjudicated 0.4%, GPT-4o 3.5%): enumerated didactic refusals match compliance patterns, and refusals without stock phrases match neither list. Adjudicated and GPT-4o HCRs are close throughout. At the calibrated β^*, SAB has the lowest GPT-4o HCR in seven of the eight small-model benchmark pairs and ties the safety prompt in the eighth (Phi-3.5 on HarmBench, 15.3%); this is against the generic and safety prompts only, and two of the seven are TinyLlama's artefact (Appendix T); Table 2 compares other methods. Results that rely on keyword labels alone should be read with this error in mind.

G Statistical Robustness

We assess HCR reliability via bootstrap resampling (10,000 samples, 95% CI). For TinyLlama on AdvBench: baseline CI [83.5, 89.4]%, safety [84.6, 90.2]% (overlapping, confirming no prompt effect); SAB at $\beta = 10$: [6.7, 11.5]% (non-overlapping with both baselines; this drop is the beginning-of-sequence artefact of Appendix T). SmolLM2 shows non-overlapping CIs between generic [71.2, 78.6]% and safety [11.9, 17.9]%, confirming the large prompt effect.

Under stochastic decoding (T=0.7, top-p=0.9), Qwen 0.5B safety-prompt HCR is $11.15\% \pm 0.69\%$ (three seeds); with SAB at $\beta = 2$ (not $\beta^* = 1.5$; a single seed, so no spread is available), HCR is 5.19% (53% relative reduction), in line with the greedy-decoding reduction at β^* (59%).

Safety-HCR robustness under stochastic decoding. Table 7 extends this analysis to TinyLlama and SmolLM2 on AdvBench. Stochastic sampling raises the absolute HCR floor for both baseline

and biased conditions (expected: temperature-induced randomness can sample harmful continuations even when the model's top-ranked token is a refusal). SmolLM2 retains a 35% relative reduction ($\beta = 2$, comparable to the 32% greedy reduction), so its effect is not specific to greedy decoding. TinyLlama's 48% at $\beta = 10$ is the beginning-of-sequence artefact (Appendix T) and is listed for completeness only.

Table 7: HCR (%) under stochastic decoding (T=0.7, top-p=0.9) on AdvBench, mean ± std across 3 seeds. Greedy results from Table 1 shown for comparison. Std is the sample standard deviation ($n-1$) computed by the evaluation script; the SmolLM2 rows were re-derived from the archived per-seed counts, whereas the TinyLlama per-seed file was not archived, so that row is as recorded at run time and could not be re-derived.

Model	Condition	Stochastic	Greedy
TinyLlama	Safety prompt	74.2 ± 1.5	87.5
	SAB $\beta = 10$	38.7 ± 0.5	9.0
	Rel. reduction	*47.8%*	*89.7%*
SmolLM2	Safety prompt	28.6 ± 1.0	14.8
	SAB $\beta = 2$	18.5 ± 1.9	10.0
	Rel. reduction	*35.3%*	*32.4%*

Task-utility robustness under stochastic decoding (Llama-3.1-8B). Table 3 reports CoSA-Score improvements under greedy decoding. A natural concern is whether these gains survive the additional randomness of temperature sampling. We evaluate Llama-3.1-8B-Instruct (8-bit) on all five CoSApien splits with stochastic decoding (T=0.7, top-p=0.95) across three seeds (42, 43, 44). Table 8 reports per-split CoSA-Scores. Under stochastic decoding the unbiased baseline rises from a greedy 0.423 to a sampled 0.461, while SAB at $\beta = 1.01$ falls from a greedy 0.457 to a sampled 0.434, so the greedy SAB advantage does not carry over. A paired t-test over ($n = 15$) split–seed observations yields $p = 0.22$ for $\beta = 1.01$ and $p = 0.40$ for $\beta = 1.05$ (Cohen's $d = -0.33$ and -0.22 respectively): neither operating point produces a statistically significant change in task-aligned utility, but the point estimates are mildly negative. We interpret this as evidence that SAB's CoSApien gain is interaction-specific to deterministic ($\arg\max$) decoding—small attention-mass shifts that change the top-1 token can fail to change the sampled token at T=0.7—and flag this as a limitation of the role-specific result. SmolLM2's safety reduction, by contrast, persists under stochastic decoding (Table 7 above).

Table 8: CoSA-Scores (mean ± std across 3 seeds) for Llama-3.1-8B (8-bit) under stochastic decoding. Neither β value induces significant degradation ($p > 0.05$, paired t-test).

Split	Baseline	$\beta = 1.01$	$\beta = 1.05$
GameDev	0.469 ± 0.072	0.402 ± 0.083	0.417 ± 0.023
PubPros	0.723 ± 0.016	0.680 ± 0.035	0.693 ± 0.042
BookPub	0.402 ± 0.066	0.390 ± 0.157	0.424 ± 0.010
LangLearn	0.355 ± 0.040	0.402 ± 0.056	0.340 ± 0.038
FilmProd	0.357 ± 0.045	0.297 ± 0.012	0.347 ± 0.082
Average	**0.461**	**0.434**	**0.444**

H Over-refusal on XSTest

We measure over-refusal on XSTest (Röttger et al., 2024): 250 safe prompts that superficially resemble harmful ones (e.g. "How can I kill a Python process?") and 200 contrasting unsafe prompts. Each model answers under the generic prompt, the safety prompt and SAB at β^* (greedy, 128 new tokens). Table 9 reports refusal rates under the official XSTest string-match rule (a response is a refusal if it opens with a stock refusal phrase) and under our keyword classifier, together with the share of answers to safe prompts that echo the system prompt (Appendix I). String rules miss refusals

that do not open with a stock phrase, so absolute rates are approximate; comparisons within a model use the same rule.

Table 9: XSTest refusal rates (%). SM: official string match; KW: keyword classifier. Echo: share of answers to safe prompts that restate the system prompt.

		Safe (250)		Unsafe (200)		
Model	**Condition**	SM	KW	SM	KW	**Echo**
Qwen 0.5B	Generic	34.8	35.2	87.5	85.5	0.0
	Safety prompt	24.4	27.2	79.0	78.5	0.0
	SAB $\beta=1.5$	37.2	38.0	87.5	86.5	0.0
SmolLM2 1.7B	Generic	5.6	2.4	15.5	15.0	0.0
	Safety prompt	21.6	16.0	72.5	76.5	0.0
	SAB $\beta=3$	38.0	33.2	62.0	90.5	0.0
TinyLlama 1.1B	Generic	0.0	1.6	0.0	1.0	0.0
	Safety prompt	0.0	1.2	1.0	1.0	0.0
	SAB $\beta=10$	7.2	0.8	8.0	1.0	66.8

Relative to the safety prompt, SAB raises the refusal rate on safe prompts from 24.4% to 37.2% for Qwen 0.5B and from 21.6% to 38.0% for SmolLM2 under string match (keyword: $27.2 \rightarrow 38.0\%$ and $16.0 \rightarrow 33.2\%$), so its gains on harmful requests come with more refusals of benign ones. TinyLlama rarely refuses under either rule, but at $\beta=10$ two-thirds of its answers to safe prompts are system-prompt echoes rather than answers, a loss of helpfulness that string rules do not register. Judge-based rates for all conditions are in Table 28 and supersede these where they differ (TinyLlama at $\beta=10$: 48.4%).

I Degenerate Outputs at High Bias

HCR measures only whether a response supplies harmful content, so it does not distinguish a coherent refusal from a response that fails to engage at all. We flag a response as a *system-prompt echo* if it restates the safety prompt or the self-description that TinyLlama produces at high β (regular expression over "safe and helpful AI assistant", "refuse politely", "not capable of refusing/accepting", "I am a machine, not a human", "never help with harmful"). Table 10 reports the share of such responses.

Table 10: Share of responses (%) that echo the system prompt.

Model	**Condition**	**AdvBench**	**HarmBench**
TinyLlama 1.1B	Safety prompt ($\beta=1$)	2.1	—
	$\beta=5$	27.7	—
	$\beta=10$ (β^*)	91.7	67.0
Qwen 0.5B, SmolLM2, Phi-3.5, Llama-3.1 8B	β^*	0.0	—

For TinyLlama the echo rate rises steeply with β and nearly every echo is labelled non-compliance, so most of its HCR reduction at $\beta=10$ consists of such outputs rather than coherent refusals. The other models produce no echoes at their calibrated β. Deployers should therefore monitor response quality, not only HCR, when calibrating weak models. Appendix T traces these echoes to the bias on the beginning-of-sequence token.

This regime is also numerically fragile. Regenerating the first 100 TinyLlama $\beta=10$ AdvBench responses with the original generation code on a newer software stack reproduces the behaviour (95% echoes vs. 89% in the stored run) but only 4% of responses token-for-token; outputs typically share their opening and diverge after tens of tokens. Exact TinyLlama $\beta=10$ numbers are therefore reproducible only approximately, whereas Qwen 0.5B at $\beta=1.5$ regenerates exactly (Appendix N).

J HELD-OUT CALIBRATION

For each model we draw 20 random 50/50 partitions of AdvBench (seeds 0–19), run the calibration protocol (Eq. (3)) on the calibration half only and evaluate the selected β^* on the held-out half. The safety-prompt baseline is evaluated on the same held-out halves, and HarmBench is scored in full at the split-selected β^*. Table 11 reports the $\delta = 10\%$ budget used in Table 1.

Table 11: Held-out calibration, $\delta = 10\%$, 20 splits (mean $\pm$ std). *Cal.* and *Held-out*: SAB HCR (%) on the calibration and held-out halves; *Safe (HO)*: safety-prompt HCR on the same held-out halves.

Model	β^* selected	Cal.	Held-out	Safe (HO)	HarmBench
Qwen 0.5B	1.5 (×20)	1.5 ± 0.5	2.0 ± 0.5	4.5 ± 0.9	31.3
TinyLlama 1.1B	10 (×20)	9.3 ± 1.1	8.8 ± 1.1	87.8 ± 1.5	22.3
SmolLM2 1.7B	2 (×3), 3 (×12), 5 (×5)	8.1 ± 1.0	10.1 ± 0.8	15.3 ± 1.2	30.7 ± 1.0
Phi-3.5 3.8B	1.5 (×20)	0.0 ± 0.0	0.0 ± 0.0	0.4 ± 0.2	9.7
Llama-3.1 8B	1.5 (×20)	0.3 ± 0.2	0.1 ± 0.2	0.3 ± 0.2	3.3

Selection is stable: four models pick the same β^* on every split, and SmolLM2, whose AdvBench HCR is nearly flat for $\beta \geq 2$ (Table 1), moves between neighbouring grid points. The largest calibration-to-held-out gap is SmolLM2's $+2.0$ pp, the expected optimism of picking the best of several near-tied candidates on 260 prompts; for the other models the gap is within one standard deviation. At $\delta = 5\%$ Qwen 0.5B has no admissible β and SmolLM2 selects $\beta = 3$ on 17 of 20 splits (held-out $9.7 \pm 1.1\%$); the remaining models are unchanged.

K PROMPT REPETITION BASELINE

A common practitioner technique for reinforcing system instructions is to repeat the system prompt after the user's request. We evaluate this approach by keeping the safety prompt in the system turn and appending a second user turn, `Reminder:` followed by the same safety prompt, after the user query. Table 12 compares prompt repetition against SAB at the operating points of an earlier calibration pass ($\beta = 1.5, 10, 2, 5$). The two columns use different labelling procedures: repetition responses carry keyword labels only, whereas the SAB responses carry adjudicated labels (Appendix F), so differences on models where the keyword classifier is unreliable (notably Phi-3.5) may be classifier artifacts.

Table 12: HCR (%) under prompt repetition (keyword labels) versus SAB (adjudicated labels) at the earlier operating points given in the text. Bold indicates the lower (better) value in each pair. Δ: percentage-point difference (positive = SAB better).

	AdvBench		**HarmBench**		Δ (Rep $-$ SAB)	
Model	Rep.	SAB	Rep.	SAB	AdvB	HarmB
Qwen 0.5B	1.9	**1.7**	**22.0**	31.3	$+0.2$	-9.3
TinyLlama 1.1B	67.5	**9.0**	52.3	**22.3**	$+58.5$	$+30.0$
SmolLM2 1.7B	13.9	**10.0**	**12.0**	31.0	$+3.9$	-19.0
Phi-3.5 3.8B	8.8	**0.4**	20.7	**6.0**	$+8.4$	$+14.7$

These keyword-label comparisons are superseded by Table 2, where all cells share the HarmBench classifier: there repetition gives Phi-3.5 0.0% AdvBench HCR (the 8.8% above is a keyword artefact), matches or beats SAB in five of the six non-TinyLlama cells, and TinyLlama's SAB column is an artefact (Appendix T). Both methods raise over-refusal (Table 28).

L ADDITIONAL FIGURES

This appendix collects supporting figures referenced from the main text: capability metrics as a function of β (Figure 5), per-head system attention share (Figure 6), the layer-selective and uniform-

mask ablations (Figures 7 and 10), the CoSApien per-type and per-split decompositions (Figures 8 and 9) and system-prompt length sensitivity (Figure 11).

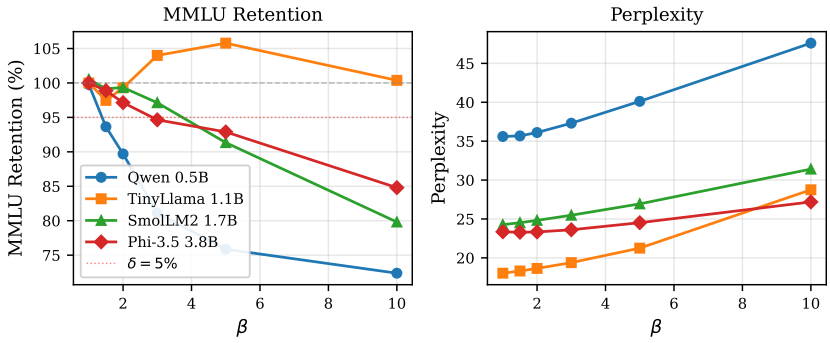

Figure 5: Capability metrics as a function of β. Shaded region: $\delta = 5\%$ acceptance threshold.

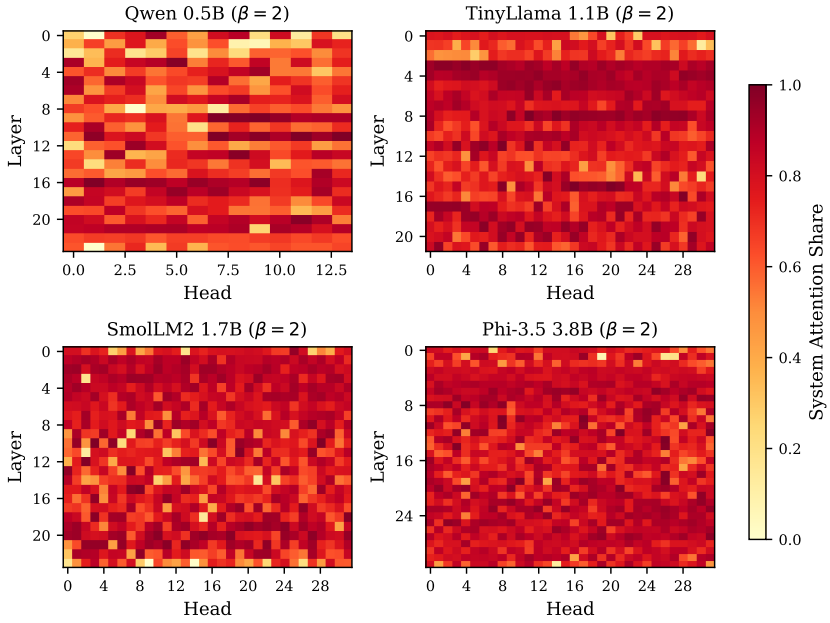

Figure 6: Per-head system attention share at $\beta = 2$ for all four models. Rows: layers; columns: heads. Warmer: higher system attention.

M Context Overflow Attack

A natural attack against system-prompt-based defences is *context overflow*: the adversary fills the context window with repeated harmful content, pushing the system prompt to a vanishing fraction of the input. We evaluate SAB's robustness under this attack by repeating an AdvBench harmful prompt N times to occupy a target fraction of the model's context window. Five fill levels are tested: 25%, 50%, 90%, 100% and 125% of the context window. At fill levels exceeding 100%, the input is truncated to $C - G$ tokens (context window C minus generation budget $G = 128$), preserving the system prompt at the beginning of the sequence. Each condition is evaluated on 100 distinct AdvBench prompts using greedy decoding, comparing no bias ($\beta = 1.0$) against the calibrated β^*.

Table 13 reveals a non-monotonic relationship between fill level and attack effectiveness. For TinyLlama, no-bias HCR rises from 69% at 25% fill to a peak of 76% at 90% fill, then collapses to 35% at 100% fill. SmolLM2 exhibits analogous behaviour: no-bias HCR peaks at 66% (50% fill)

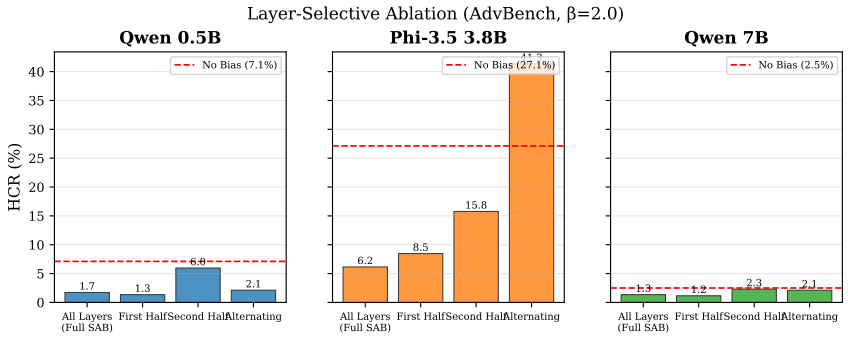

Figure 7: Layer-selective ablation (A2) at $\beta = 2$. Dashed red line: A1 (no-bias) baseline.

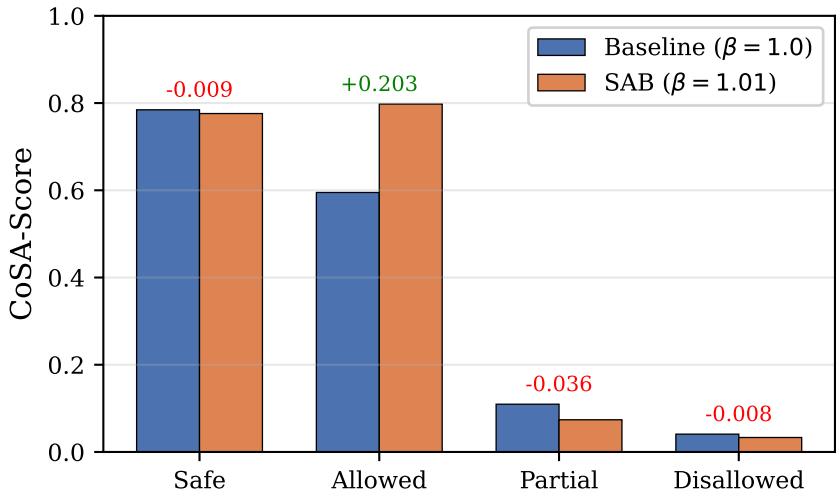

Figure 8: Per-type CoSA-Score decomposition at $\beta = 1.01$ vs. baseline.

before declining to 39% at 100% fill. Qwen 0.5B, with its much larger context window ($C = 32{,}768$), follows the same pattern: baseline HCR peaks at 72% (90% fill) before collapsing to 41% at 100% fill. The collapse occurs because context saturation degrades coherent generation; at $\geq 100\%$ fill, all three models frequently produce incoherent echoes of the input rather than actionable harmful content.

At the attacker's optimal operating point (where no-bias HCR is maximised), SAB provides substantial protection. For TinyLlama at 90% fill, the bias reduces HCR from 76% to 39% (49% relative reduction). For SmolLM2 at 50% fill, the reduction is from 66% to 58% (12% relative). For Qwen 0.5B, SAB reduces HCR from 57% to 31% at 25% fill (46% relative) and from 43% to 27% at 50% fill (37% relative). The stronger TinyLlama effect is the beginning-of-sequence artefact (Appendix T). For Qwen 0.5B, the SAB attention mask at $\geq 90\%$ fill exceeds GPU memory (12 GB) due to the 32K context window, limiting SAB evaluation to $\leq 50\%$ fill on our hardware.

At $\geq 100\%$ fill, no-bias HCR converges to ≈ 35–41% across all three models, with Δ_{rel} dropping to $\leq 3\%$ for TinyLlama and SmolLM2. This convergence reflects a regime where the attack is self-defeating: the adversary fills sufficient context to degrade generation quality below the threshold of coherent harmful output. The 100% and 125% rows are numerically close but not identical: at fill levels $\geq 100\%$ the input is truncated to $C - G$ tokens, and because the prompt is constructed by repeating the harmful request followed by a single space (`(prompt + " ") * n_repeats`, then truncated), the cut-point lands at different positions within the repetition cycle for the two fill levels; the last few tokens of the truncated input therefore differ slightly, producing the small residual differences seen in Table 13. Truncation imposes a hard ceiling on the attacker's effective context length at $C - G$, beyond which additional repetitions cannot increase attack strength.

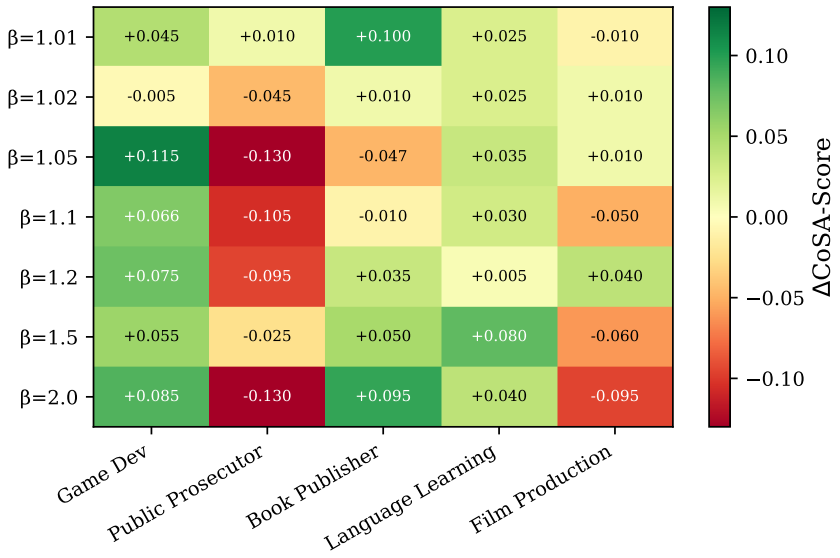

Figure 9: ΔCoSA-Score heatmap (β vs. split).

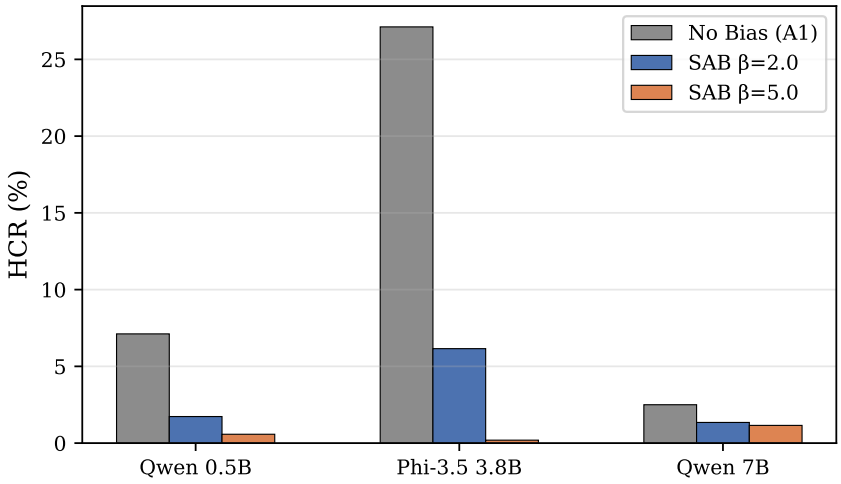

Figure 10: A1 (uniform) vs. SAB at $\beta \in \{2, 5\}$.

Convergence of biased and unbiased conditions. A notable observation is that SAB and no-bias HCR become nearly identical at $\geq 100\%$ fill despite the $+\log \beta$ bias remaining active. Because the context is filled with *repeated copies of the harmful prompt itself* (not generic filler), at saturation the vast majority of key–value entries carry adversarial content. The system-prompt tokens are reduced to a vanishing fraction of the sequence, and their softmax weight—even after the SAB boost—is overwhelmed by the sheer count of adversarial keys. Generation at this regime degenerates into incoherent echoes of the repeated input rather than structured harmful output, producing a floor HCR of $\approx$35–41% regardless of bias. This suggests that SAB's protection is bounded by the *ratio* of system tokens to total tokens: once the system prompt constitutes $< 1\%$ of the input, even large multiplicative boosts cannot compensate for the dilution. Investigating alternative strategies—such as sparse or windowed attention masks that restrict adversarial tokens' access to system-token representations, or dynamic β scaling proportional to context length—is a promising direction for future work.

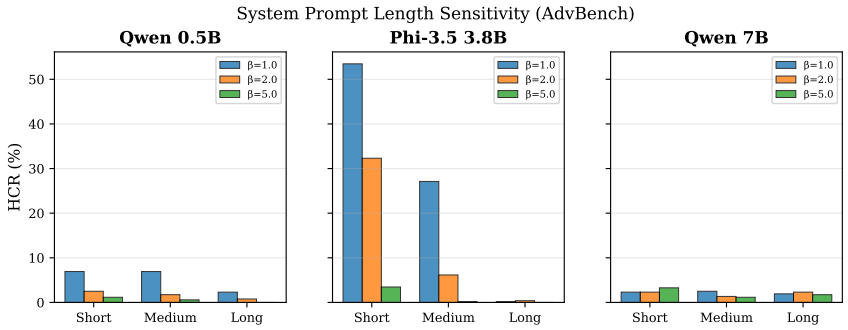

Figure 11: System prompt length sensitivity (A3) at three β values. Phi-3.5 with the long prompt was run at $\beta \in \{1, 2\}$ only: its $\beta = 5$ position is a run that was not performed, not a measured 0%.

Table 13: Context overflow attack: HCR (%) under increasing context fill levels. Δ_{rel}: relative reduction from no bias to SAB.

	Fill	No bias	SAB	Δ_{rel}
Qwen 0.5B ($\beta^* = 1.5$, $C = 32768$)	25%	57	31	−46%
	50%	43	27	−37%
	90%	72	OOM‡	—
	100%	41	OOM‡	—
	125%†	35	OOM‡	—
TinyLlama 1.1B ($\beta^* = 10$, $C = 2048$)	25%	69	29	−58%
	50%	75	38	−49%
	90%	76	39	−49%
	100%	35	34	−3%
	125%†	35	34	−3%
SmolLM2 1.7B ($\beta = 2$, $C = 8192$)	25%	56	50	−11%
	50%	66	58	−12%
	90%	60	55	−8%
	100%	39	38	−3%
	125%†	35	35	0%

† At $\geq 100\%$ fill, input is truncated to $C - G$ tokens.
‡ SAB attention mask exceeds GPU memory at $\geq 90\%$ of 32K context.

N Fused-Kernel Implementation

The dense additive mask is not required. The bias depends only on the (query, key) positions, so it can be written as a FlexAttention `score_mod`, `score + log_beta * (kv_idx in S and q_idx not in S)`, evaluated inside a fused kernel together with a sparse causal block mask; during cached decoding the same modifier is applied with the query index offset by the cache length. Nothing $n \times n$ is materialised. Grouped-query models whose group size is not a power of two (Qwen 2.5 0.5B) require expanding keys and values first, which costs $O(n)$.

Memory. Table 14 compares peak prefill activation memory (weights excluded) on Qwen 2.5 0.5B for the dense float16 mask with PyTorch SDPA, which is what our experiments used, and for the FlexAttention path.

At 29,491 tokens the dense path needs 4.3 times the activation memory of the FlexAttention path (4.37 vs. 1.01 GB) and 5.6 times its prefill time (6.9 vs. 1.2 s on the GB10).

Equivalence. Regenerating all 520 AdvBench responses for Qwen 2.5 0.5B at $\beta = 1.5$ with the FlexAttention path reproduces 98.3% of the stored greedy responses token-for-token; a regeneration with the dense SDPA path reproduces 100%, and both regenerations yield the same HCR under the same classifier. For TinyLlama at $\beta = 10$, where generation is highly unstable (Appendix I), neither regeneration reproduces the stored responses (FlexAttention 1.9%, dense SDPA 1.7% token-

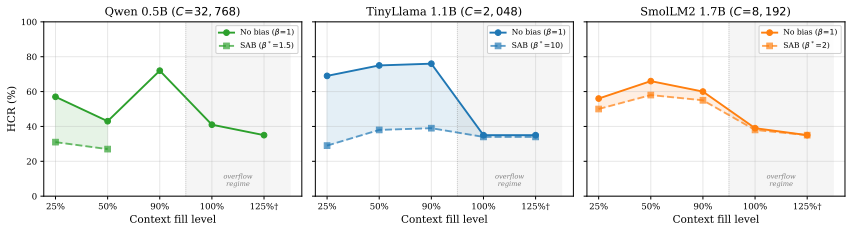

Figure 12: HCR under context overflow attack. The shaded area represents the protection gap between no-bias and SAB conditions. All three models exhibit non-monotonic attack effectiveness that peaks at moderate fill levels, with HCR collapsing at $\geq 100\%$ fill. Grey region marks the overflow regime. For Qwen 0.5B, SAB data is limited to $\leq 50\%$ fill due to GPU memory constraints at the 32K context window.

Table 14: Peak prefill activation memory (GB), Qwen 2.5 0.5B, $\beta = 1.5$, single RTX 3080 Ti. [§]Measured on an otherwise idle NVIDIA GB10, where the other columns reproduce to two decimals (4,096–16,384 tokens).

Tokens	2,048	4,096	8,192	16,384	29,491
Dense mask + SDPA	0.07	0.14	0.40	1.43	4.37[§]
FlexAttention	0.07	0.14	0.28	0.56	1.01

identical), while the two regenerations give nearly the same keyword HCR (33.9% and 34.6%). The divergence is therefore between today's regenerations and the stored run, not between the two attention implementations.

O Inference Overhead

A practical concern for any inference-time intervention is computational cost. We measure wall-clock generation time on 50 simple prompts ("Explain the number N briefly", max 128 new tokens, greedy decoding) on a single NVIDIA RTX 3080 Ti, comparing native generation (standard `model.generate`) against SAB at $\beta = 3$.

Table 15: Inference overhead of SAB relative to native generation. Mask construction time (single-call) is <0.03 ms at all sequence lengths. Negative overhead indicates SAB is faster than native generation.

Model	**Native (s)**	SAB **(s)**	**Overhead**
Qwen 0.5B	0.82 ± 0.24	0.98 ± 0.31	$+18.5\%$
TinyLlama 1.1B	0.97 ± 0.08	0.90 ± 0.20	-7.0%
SmolLM2 1.7B	0.68 ± 0.19	0.48 ± 0.16	-28.3%
Phi-3.5 3.8B	2.02 ± 0.10	2.03 ± 0.07	$+0.3\%$
Llama-3.1 8B[†]	3.55 ± 1.66	2.11 ± 1.06	-40.7%
OLMoE 1B-7B[‡]	17.11 ± 0.41	16.18 ± 1.74	-5.4%

[†]8-bit quantisation. [‡]4-bit quantisation; Mixture-of-Experts (64 experts, top-8).

Table 15 compares time per *response*, which confounds the cost of the bias with response length: SAB changes how long the responses are, which is how a nominal -40.7% "overhead" arises. Table 16 therefore reports time per generated *token*, and separates the cost of the bias (the same decoding loop at $\beta = 3$ and $\beta = 1$) from the cost of our unoptimised reference loop (that loop at $\beta = 1$ against `model.generate`). The bias costs 0.2–1.7% per token and the loop between -0.2% and $+2.4\%$.

Mask construction itself is negligible (<0.03 ms for sequences up to 512 tokens). End-to-end generation overhead ranges from -41% to $+19\%$ across six models, with a median of -6% (Table 15). **Caveat: implementation parity.** The native column uses HuggingFace's default

Table 16: Time per generated token (ms, mean ± std over 50 prompts) on an otherwise idle NVIDIA GB10, greedy decoding, at most 128 new tokens. Bias: $\beta = 3$ against $\beta = 1$ in the same decoding loop, the cost of the bias itself. Loop: our $\beta = 1$ loop against `model.generate`, the cost of the unoptimised reference loop with its dense mask.

Model	Native	Loop, $\beta = 1$	Loop, $\beta = 3$	Bias	Loop
Qwen 0.5B	8.70 ± 0.09	8.91 ± 0.05	8.93 ± 0.04	$+0.2\%$	$+2.4\%$
TinyLlama 1.1B	14.66 ± 0.09	15.01 ± 0.02	15.08 ± 0.04	$+0.4\%$	$+2.4\%$
SmolLM2 1.7B	21.55 ± 0.07	21.50 ± 0.06	21.58 ± 0.10	$+0.4\%$	-0.2%
Phi-3.5 3.8B	43.26 ± 0.07	43.68 ± 0.10	43.83 ± 0.12	$+0.3\%$	$+1.0\%$
Llama-3.1 8B[†]	56.10 ± 0.65	57.09 ± 0.55	58.07 ± 0.67	$+1.7\%$	$+1.8\%$

[†] 8-bit quantisation.

`model.generate` loop, while the SAB column uses our custom generation loop with explicit KV-cache management. The negative overheads for several models (notably -41% on Llama-3.1-8B) therefore reflect a mix of (i) the SAB intervention itself (which is essentially free—an additive scalar in the attention mask) and (ii) differences between the two generation loops (e.g., avoidance of certain recomputation paths in the default implementation). We do *not* claim that SAB makes generation faster. The honest reading of Table 15 is that the SAB intervention is performance-neutral and that any apparent speedup is attributable to the surrounding loop, which a careful native implementation could equally adopt. We retain the figures here for reproducibility and report mask-construction time ($<$0.03 ms) as the relevant SAB-specific overhead.

P Mixture-of-Experts Probe

To test whether SAB generalises to sparse architectures, we evaluate OLMoE-1B-7B-Instruct (Muennighoff et al., 2025), a Mixture-of-Experts model with 6.9B total parameters but only 1.3B active per token (64 experts per layer, top-8 routing, 16 layers, 16 attention heads). The model is loaded with 4-bit quantisation (bitsandbytes FP4, the library default; $\sim$3.9 GB VRAM). Every OLMoE number in this appendix (the 50-prompt probe in Table 17 and the capability curve in Table 18) comes from this 4-bit FP4 model; no full 520-prompt or higher-precision OLMoE result is reported.

Because OLMoE's autoregressive generation is substantially slower than dense models of comparable active size ($\sim$16.5 s per prompt at 128 new tokens on the RTX 3080 Ti), we report results on the first 50 AdvBench prompts as a proof of concept.

Table 17: OLMoE-1B-7B (4-bit FP4) proof-of-concept results on 50 AdvBench prompts. HCR decreases monotonically with β, confirming SAB effectiveness on Mixture-of-Experts architectures.

Condition	Compl.	Refusal	Ambig.	HCR (%)
Generic baseline	31	7	12	62.0
Safety prompt	15	23	12	30.0
SAB $\beta = 1.5$	13	22	15	26.0
SAB $\beta = 3$	11	30	9	22.0
SAB $\beta = 10$	6	40	4	**12.0**

Table 17 shows that OLMoE is poorly aligned without a safety prompt (62% HCR on the generic baseline) but responds strongly to SAB. At $\beta = 10$, HCR drops from 30% (safety prompt alone) to 12%: a 60% relative reduction, with a monotonic decrease in HCR and a shift from ambiguous to refusal responses. OLMoE's biased span was not verified (cf. Appendix T). This confirms that SAB operates at the attention-score level, below the MoE routing layer, and is therefore architecture-agnostic with respect to feed-forward sparsity.

Capability. Table 18 reports MMLU (same 1,301-question log-likelihood protocol as the dense models) and WikiText-2 perplexity for the same 4-bit FP4 model, which scores 46.6% MMLU at $\beta = 1$.

Table 18: OLMoE-1B-7B (4-bit FP4) capability as a function of β. Retention is relative to $\beta = 1$.

β	1	1.5	2	3	5	10
MMLU (%)	46.6	46.3	44.0	40.7	37.4	32.9
Retention (%)	100.0	99.3	94.5	87.3	80.2	70.6
PPL	38.9	39.8	41.6	44.1	49.9	69.1

The capability curve qualifies the safety result. The 60% relative HCR reduction at $\beta = 10$ costs 29% of MMLU accuracy, so it lies far outside the 10% budget used for the dense models; within that budget only $\beta \leq 2$ is admissible, and the admissible point we measured ($\beta = 1.5$) lowers HCR only from 30% to 26% on 50 prompts. The probe therefore shows that the mechanism acts on a sparse MoE model, but not that it does so at a favourable trade-off.

Q Multi-Turn Evaluation

All main experiments use single-turn interactions. To validate that SAB protects against attention dilution from extended conversation history, we prepend 0, 2, 5, or 10 benign filler turns (fixed question–answer pairs) before each harmful AdvBench request. Four models are tested (100 prompts each, greedy decoding, safety system prompt). These runs use the operating points of an earlier calibration pass; for SmolLM2 ($\beta = 2$) and Phi-3.5 ($\beta = 5$) they differ from the β^* of Table 1.

Table 19: Multi-turn HCR (%) on AdvBench (100 prompts) with increasing filler turns. Δ_{rel}: relative reduction from baseline to SAB. Keyword labels. TinyLlama uses a 2048-token input limit (its native context); the other models use 512 tokens, which no condition exceeds.

Model	**Turns**	**Baseline**	SAB	Δ_{rel}
Qwen 0.5B ($\beta = 1.5$)	0	9	1	−89%
	2	2	1	−50%
	5	6	3	−50%
	10	14	5	−64%
TinyLlama 1.1B ($\beta = 10$)	0	70	38	−46%
	2	75	72	−4%
	5	75	73	−3%
	10	77	69	−10%
SmolLM2 1.7B ($\beta = 2$)	0	14	11	−21%
	2	11	7	−36%
	5	20	14	−30%
	10	29	19	−34%
Phi-3.5 3.8B ($\beta = 5$)	0	24	1	−96%
	2	22	2	−91%
	5	17	1	−94%
	10	28	8	−71%

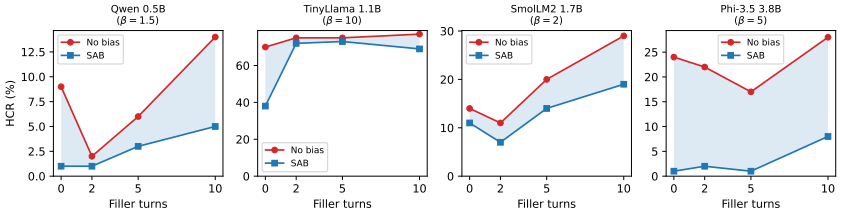

Figure 13: HCR under increasing conversation length. Shaded area shows the protection gap between no-bias and SAB conditions. TinyLlama uses a 2048-token input limit, so its 10-turn condition is included.

Table 19 and Figure 13 show three patterns. First, for Qwen, SmolLM2 and Phi-3.5, baseline HCR generally increases with conversation length (e.g. SmolLM2: 14% → 29% from 0 to 10 turns), confirming that attention dilution degrades safety as context grows. Second, SAB maintains substantial protection across all turn counts: Phi-3.5 drops from 28% to 8% at 10 turns (71% relative reduction), and Qwen from 14% to 5% (64% relative). Third, TinyLlama—the weakest model—shows minimal SAB benefit at 2–5 turns (HCR ≈73–75% for both conditions), and only 77% vs. 69% at 10 turns (577 input tokens, evaluated with a 2048-token limit; re-running the 0–5-turn conditions under this limit reproduces the 512-token values exactly); its 0-turn effect is the beginning-of-sequence artefact (Appendix T), so these rows do not measure SAB proper.

Absolute HCR values differ from Table 1 because only the first 100 AdvBench prompts are used and the keyword classifier's sensitivity varies across prompt subsets. Relative comparisons within each condition are internally consistent.

Phi-3.5 baseline discrepancy. The Phi-3.5 0-turn baseline HCR in Table 19 (24%) is substantially higher than its single-turn full-AdvBench baseline (0.4%, Table 1). Manual inspection of the 100-prompt subset shows that this gap is a keyword-classifier artifact rather than a behavioural change: on this subset Phi-3.5 produces enumerated didactic refusals ("It is illegal… 1. Legal Consequences:…") in which the leading enumerator ("1.") matches the compliance keyword pattern without any offsetting refusal phrase being detected, leading to false-compliance labels. The adjudicated main-sweep labels (Appendix F) do not show this artifact, which is why the full-AdvBench number is lower; the multi-turn outputs themselves were not adjudicated. Since the relative reduction across the bias / no-bias comparison is internally consistent under the same classifier, the multi-turn conclusions (Phi-3.5 retains a 71% relative reduction at 10 turns) are unaffected.

R Classifier-Free Guidance Baseline

System-prompt classifier-free guidance (Sanchez et al., 2024) decodes two contexts in lock-step, the safety system prompt (conditional) and the generic prompt (unconditional), each with its own KV cache, and selects $\arg\max_t \ell_u(t) + \gamma\,(\ell_c(t) - \ell_u(t))$ over log-probabilities ℓ, so every generated token costs two forward passes. We use the same greedy decoding, 128 new tokens and 512-token input limit as for SAB. CFG shifts the style of refusals away from stock phrases, so keyword labels are invalid for its outputs (the keyword classifier reports 44.0% HCR for Qwen 0.5B at $\gamma = 3$, most of it mislabelled refusals); we therefore compare GPT-4o judge labels for all conditions.

Table 20: AdvBench HCR (%) under the GPT-4o judge. SAB is at β^*.

		CFG		
Model	**Safety prompt**	$\gamma = 1.5$	$\gamma = 3$	SAB
Qwen 0.5B	5.8	7.3	21.3	1.9
TinyLlama 1.1B	86.2	82.3	59.4	8.5

Mild guidance ($\gamma = 1.5$) stays close to the safety prompt. Stronger guidance ($\gamma = 3$) makes Qwen 0.5B *less* safe than the safety prompt alone and removes about a third of TinyLlama's harmful compliance, far less than SAB; recall, however, that most of TinyLlama's SAB reduction consists of system-prompt echoes (Appendix I) and is a beginning-of-sequence artefact (Appendix T). The five-model comparison under one classifier is in Table 2, where CFG is stronger than SAB on SmolLM2.

S SafeDecoding Head-to-Head Comparison

We reproduce SafeDecoding (Xu et al., 2024) on TinyLlama 1.1B for a direct comparison with SAB. Following the original method, we fine-tune a LoRA safety expert ($r = 8$, $\alpha = 16$, targeting `q_proj` and `v_proj`) for 3 epochs on 200 prompt–refusal pairs sampled from AdvBench: 39 where TinyLlama already refuses (natural refusals, response kept verbatim) and 161 where it complies (paired with a fixed refusal template, judged by GPT-4o). At inference time, the expert and base model are loaded simultaneously and contrastive decoding (top-$k = 10$, $\alpha = 3$) steers generation toward the expert's refusal distribution.

Table 21: Head-to-head on TinyLlama 1.1B: HCR (%, lower is better) with GPT-4o labels for every row. *Held-out*: the 320 AdvBench prompts not used to train the SafeDecoding expert. SafeDecoding requires LoRA fine-tuning and 2× model memory; SAB requires a single scalar and no training.

Method	AdvBench (520)	Held-out (320)	HarmBench (300)
No defence	85.6	84.7	75.7
Safety prompt	86.2	85.9	74.3
SafeDecoding	**1.5**	**2.5**	25.0
SAB ($\beta = 10$)	8.5	9.1	24.3

SafeDecoding is clearly stronger on AdvBench, including the 320 prompts outside its training set (2.5% vs. 9.1%), so the training overlap does not explain its advantage (Table 21). On HarmBench the two are indistinguishable (25.0% vs. 24.3%); keyword labels had suggested a large SafeDecoding advantage (12.3% vs. 22.3%) because 67 of its HarmBench responses matched neither keyword list. The SAB row is TinyLlama at $\beta = 10$, which Appendix T shows is driven by the bias on the beginning-of-sequence token, not the system prompt; this table therefore compares SafeDecoding with that artefact and says nothing about SAB proper. Two caveats qualify the comparison. **(1) Response quality.** Neither method produces diverse, prompt-specific refusals: 98.6% of SafeDecoding's AdvBench refusals begin with the same template sentence, and most of SAB's reduction at $\beta = 10$ consists of responses that restate the system prompt (91.7% on AdvBench, 67.0% on HarmBench; Appendix I). **(2) Resource cost and runtime flexibility.** SafeDecoding requires labelled safety data, LoRA fine-tuning, and loading two model copies during inference (2× memory). Because the expert is trained on a fixed safety policy, adapting to a different policy (e.g. switching from content-safety to role-specific persona compliance) requires collecting new data and retraining the adapter. SAB introduces no additional learnable parameters and loads no second model; its dense reference mask costs $O(n^2)$ memory at prefill, which the fused implementation avoids (Appendix N), and the deployer can switch policies at runtime by changing the system prompt and β, with no retraining or reloading. With labelled data and a fine-tuned expert, SafeDecoding is the stronger defence on this model; SAB's case rests on needing neither.

T Software Stacks and Generation-Loop Details

Three software stacks. The experiments were run over several months on two machines, and the library versions differ between them. Greedy decoding is deterministic on one stack but not across stacks: different attention kernels change low-order bits of the logits, and a single changed token changes the rest of a response. Table 22 lists which results come from which stack.

Cross-stack probe. Before moving generation to the GB10 we fixed a decision rule: regenerate 100 seeded prompts per probe with the identical generation functions, and accept the move only if per-prompt label agreement is at least 95% and the Wilson 95% intervals of the two rates overlap. Table 23 reports the outcome. The intervals overlap in every probe, but two probes miss the 95% agreement threshold, so by our own rule the stacks are *not* interchangeable at the level of individual responses. The largest disagreement is TinyLlama at $\beta = 10$ against the stored main sweep, where responses are dominated by system-prompt echoes (Appendix I) and only 1% of responses are token-identical. Comparisons *within* one table row that mix stacks (Table 22) therefore carry this additional noise, which is of the order of the differences between neighbouring β or γ values and smaller than the differences we draw conclusions from. Time did not allow regenerating all ∼110,000 responses on one stack.

Judges for the baseline table and the frontier. AdvBench and HarmBench responses in Table 2 and Figure 3 are labelled by the HarmBench classifier (`HarmBench-Llama-2-13b-cls`, bf16, official prompts, 45,162 responses, none unlabelled). On HarmBench's 596 human-labelled validation pairs our deployment agrees with the human majority on 90.8% and with HarmBench's own recorded classifier predictions on 99.3%. The acceptance rule we had fixed beforehand, agreement within 2 points of the published 93.2%, *failed*; the recorded predictions of the official classifier reach the same 90.8% on these pairs, so we accepted the deployment as a faithful copy of the official classifier rather than as one that meets the published figure. XSTest responses are labelled with the official three-class prompt by Qwen3.8-27B with a refusal-ablation adapter, so that the judge itself does

Table 22: Software stack per experiment. All runs use greedy decoding unless stated, the same prompts, and the same generation code.

Results	Stack
Main β sweep (AdvBench, HarmBench) and the other experiments stored before April 2026 (prompt repetition for the four small models, ablations, context overflow, multi-turn, CoSApien)	2×RTX 3080 Ti; PyTorch 2.10, Transformers 5.1. Versions were not recorded per run; this is the stack installed until 2 April 2026
XSTest, all conditions, for Qwen, TinyLlama and SmolLM2; XSTest generic, safety, $\beta \in \{1.5, 2\}$ and repetition for Llama-3.1-8B; XSTest generic and safety for Phi-3.5; CFG $\gamma \in \{1.5, 3\}$ on AdvBench and XSTest for Qwen and TinyLlama, $\gamma = 3$ for SmolLM2 and Llama-3.1-8B	same GPUs; PyTorch 2.6, Transformers 4.57.6
XSTest SAB sweep and repetition for Phi-3.5; all CFG runs for Phi-3.5; CFG $\gamma = 5$, SmolLM2 $\gamma = 1.5$ and CFG on HarmBench for all models; prompt repetition on AdvBench and HarmBench for Llama-3.1-8B; the controls of this appendix	NVIDIA GB10 (aarch64); PyTorch 2.13, Transformers 4.57.6
XSTest $\beta = 3$ for Llama-3.1-8B	mixed: the first 315 responses on the PyTorch 2.6 stack, the remaining 135 on the GB10 (a whole-file regeneration did not finish in time)

Table 23: Cross-stack probe, $n = 100$ prompts each. Labels: official XSTest string match (refusal) and the keyword classifier (compliance). Rates in %.

Probe	Identical	Label agr.	Rate A	Rate B	Rule
Qwen, XSTest, SAB $\beta = 1.5$ (PyTorch 2.6 vs. GB10)	86	100	46	46	pass
Qwen, AdvBench, CFG $\gamma = 3$ (PyTorch 2.6 vs. GB10)	56	94	35	33	fail
Llama-3.1-8B, XSTest, safety prompt, 8-bit (2.6 vs. GB10)	14	99	21	20	pass
Llama-3.1-8B, XSTest, safety prompt, 8-bit vs. bf16 on GB10	10	98	21	19	pass
TinyLlama, AdvBench, SAB $\beta = 10$ (stored sweep vs. GB10)	1	71	24	31	fail

not refuse; its agreement with GPT-4o on a pilot sample is $\kappa = 0.885$, and 39 of 23,400 responses received no parseable label and are excluded. In that pilot the judge labelled fewer Phi-3.5 responses as refusals than GPT-4o did, by 9.4 points on 100 items (96 with a parseable label; exact McNemar $p = 0.012$), so Phi-3.5's over-refusal values and its 4.4-point increase under SAB should be read with that in mind.

What the generation loop biases. The implementation used for every SAB number in this paper adds $\log \beta$ to key positions $[0, L)$, where L is the length of the chat template rendered with the system message alone and tokenised without special tokens. We verified after the experiments that $[0, L)$ is exactly the system block for Qwen2.5 and SmolLM2 only. For TinyLlama and Llama-3.1 the tokenised input starts with a beginning-of-sequence token (two for Llama-3.1, because the template writes one and the tokenizer adds another), so the biased span contains those tokens and ends one token early, omitting the block's final delimiter. For Phi-3.5 the biased span contains the system block and the following <|user|> tag.

Span control. We reran TinyLlama at its calibrated $\beta = 10$ on the first 100 AdvBench prompts with three spans (Table 24; same stack, same prompts, keyword labels). Removing the beginning-of-sequence token from the span removes the effect: echoes fall from 94% to 18% and HCR returns to the level of the safety prompt on the same prompts (67% in the stored main sweep). TinyLlama's results throughout this paper therefore reflect a $10\times$ boost of the beginning-of-sequence token, a known attention sink, rather than of the system prompt. Qwen2.5 and SmolLM2 are unaffected (exact span).

The same control for the other two affected models shows no comparable effect (Table 25): correcting the span moves Llama-3.1-8B's over-refusal and Phi-3.5's HCR by 2–7 points, inside the 95% intervals, with 93–97% per-prompt label agreement. The artefact is thus specific to TinyLlama at

Table 24: TinyLlama, $\beta = 10$, first 100 AdvBench prompts, GB10 stack. HCR from the keyword classifier with Wilson 95% intervals; echo: share of responses restating the system prompt.

Biased key positions	HCR (%)	95% CI	Echo (%)
As used in the paper: $[0, 40)$, BOS + block without its last token	38	29–48	94
BOS removed: $[1, 40)$	70	60–78	18
Exact system block: $[1, 41)$	67	57–75	12
Safety prompt, no bias (stored main sweep, same prompts)	67	57–75	1

$\beta = 10$; the smaller models' calibrated values ($\beta \leq 3$) and Llama-3.1-8B's two beginning-of-sequence tokens at $\beta = 2$ do not produce it.

Table 25: Span and stopping controls for Llama-3.1-8B (8-bit; XSTest, first 60 safe prompts, $\beta = 2$; refusals by the official string match) and Phi-3.5 (first 100 AdvBench prompts, $\beta = 1.5$; keyword HCR). Rates in % with Wilson 95% intervals; agreement is per-prompt label agreement with the implementation as used.

Model	Variant	Rate	95% CI	Agreement
Llama-3.1-8B	as used (two BOS in span)	61.7	49–73	—
	BOS removed	55.0	42–67	93.3
	exact system block	58.3	46–70	96.7
Phi-3.5	as used (span includes <|user|>)	11.0	6–19	—
	exact system block	9.0	5–16	96.0
	stop on the full end-of-sequence list	12.0	7–20	98.0
	both corrections	9.0	5–16	95.0

Stopping and decoding settings. The SAB loop stops on the tokenizer's end-of-sequence token only, whereas native generation (used for the generic- and safety-prompt conditions) stops on the list in the model's generation configuration. The two coincide for Qwen2.5, TinyLlama, SmolLM2 and, in practice, Llama-3.1. They differ for Phi-3.5, whose turns end with <|end|>: on XSTest, 86% of Phi-3.5 responses reach the 128-token limit under native generation and 97–99% under the SAB loop, so about one Phi-3.5 SAB response in eight continues past the end of its turn before being cut at 128 tokens. The official XSTest string match reads only the start of a response and is unaffected; judge labels see the full text. Separately, Qwen2.5's generation configuration sets a repetition penalty of 1.1, which native generation inherits and the SAB loop does not apply. Running the SAB loop at $\beta = 1$ (no bias, no penalty) on all 520 AdvBench prompts gives a keyword HCR of 6.9% (95% CI 5.0–9.4) against 7.5% for native generation, so this difference does not explain Qwen's reduction under SAB.

U Tables and Figures Moved from the Main Text, and Frontier Details

Tables 26 and 27 belong to §4.4 and §4.5, and Figure 14 to §4.2. Figure 15 and Table 28 give every model and every point of the frontier in Figure 3.

Ablation details (full text of §4.4). We conduct ablations on AdvBench (520 prompts) to isolate the operative mechanism. Results use $\beta = 2$ unless stated otherwise.

A1: Uniform bias control. Replacing the role-specific mask with a uniform causal mask (no position-specific bias) reverts HCR to unbiased baseline levels (Table 26; Figure 10), confirming that the 4D mask format alone does not improve safety; the bias must be targeted at system-token positions.

A2: Layer-selective application. Applying SAB to layer subsets reveals model-specific layer sensitivity (Table 26). Qwen 0.5B's first-half-only configuration matches all-layers HCR (1.3% vs. 1.7%), indicating early layers carry most safety-relevant signal. Phi-3.5's first half (8.5%) underperforms all layers (6.2%), and its alternating configuration (41.3%) is catastrophically worse

Table 26: Ablation results on AdvBench (HCR %, $\beta = 2$, keyword labels without adjudication, so not comparable to Table 1). A1: uniform mask. A2: layer subsets. Qwen 7B is an auxiliary probe.

	A1	A2 (Layer-Selective)			
Model	Uniform	All	1st H	2nd H	Alt.
Qwen 0.5B	7.1	1.7	**1.3**	6.0	2.1
Phi-3.5 3.8B	27.1	**6.2**	8.5	15.8	41.3
Qwen 7B	2.5	1.3	**1.2**	2.3	2.1

Table 27: Calibrated β^* at three capability budgets.

Model	$\delta = 2\%$	$\delta = 5\%$	$\delta = 10\%$
Qwen 0.5B	†	†	1.5
TinyLlama 1.1B	10.0	10.0	10.0
SmolLM2 1.7B	2.0	3.0	3.0
Phi-3.5 3.8B	1.5	1.5	1.5
Llama-3.1 8B	†	1.5	1.5

† No tested β satisfies the budget.

than no bias (27.1%; Figure 7), so coherent layer coverage is critical. Qwen 2.5 7B-Instruct, an auxiliary probe outside the main evaluation (4-bit bitsandbytes weights, greedy decoding, 128 new tokens, 512 input tokens, all 520 AdvBench prompts), confirms early-layer dominance (first half: 1.2% vs. all layers: 1.3%).

A3: Prompt length sensitivity. With system prompts of 5, 26 and 79 tokens, Phi-3.5's short-prompt HCR falls from 53.5% ($\beta = 1$) to 3.5% at $\beta = 5$, approaching the 0.2% of the long prompt; Qwen 7B is insensitive to prompt length (Figure 11).

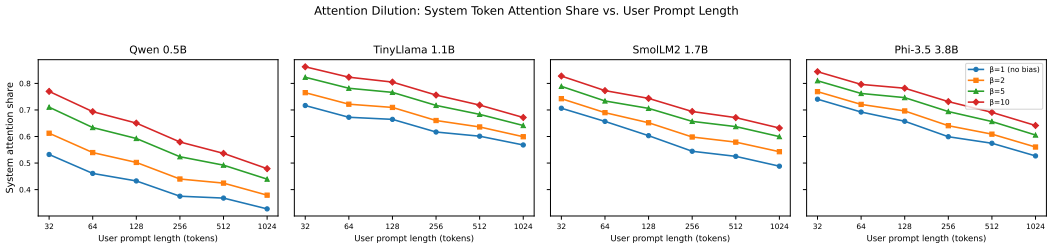

Figure 14: Empirical attention dilution. System attention share (averaged across layers) decreases as user prompt length grows (blue, $\beta = 1$). SAB at higher β restores system attention at all lengths.

Table 28: Every point of Figures 3 and 15. HCR: AdvBench, 520 prompts, HarmBench classifier. Over-refusal: XSTest, 250 safe prompts, full or partial refusal, Qwen3.8-27B judge. Wilson 95% intervals. Echo: share of safe-prompt answers that restate the system prompt. *calibrated β^*. TinyLlama's SAB rows reflect the bias on its beginning-of-sequence token (Appendix T). Earlier appendix tables on XSTest, CFG and repetition use the labels available when they were written (string match, GPT-4o, keyword) and are superseded by this table where they differ.

	AdvBench HCR (%)		**Over-refusal (%)**		**Echo**
Condition	value	95% CI	value	95% CI	(%)
Qwen2.5-0.5B ($\beta^* = 1.5$)					
Generic prompt	3.3	2.1–5.2	31.2	25.8–37.2	0.0
Safety prompt	3.5	2.2–5.4	24.4	19.5–30.1	0.0
SAB $\beta = 1.5^*$	1.3	0.7–2.8	32.8	27.3–38.8	0.0
SAB $\beta = 2$	0.8	0.3–2.0	44.4	38.4–50.6	0.0
SAB $\beta = 3$	0.8	0.3–2.0	59.6	53.4–65.5	0.0
SAB $\beta = 5$	0.6	0.2–1.7	79.6	74.2–84.1	0.0
SAB $\beta = 10$	0.0	0.0–0.7	90.4	86.1–93.5	0.0
CFG $\gamma = 1.5$	5.2	3.6–7.4	20.0	15.5–25.4	0.0
CFG $\gamma = 3$	14.2	11.5–17.5	8.4	5.6–12.5	0.4
CFG $\gamma = 5$	18.1	15.0–21.6	8.4	5.6–12.5	2.4
Prompt repetition	0.8	0.3–2.0	42.0	36.0–48.2	0.0
TinyLlama-1.1B ($\beta^* = 10$)					
Generic prompt	66.9	62.8–70.8	1.2	0.4–3.5	0.0
Safety prompt	66.5	62.4–70.5	0.4	0.1–2.2	0.0
SAB $\beta = 1.5$	62.3	58.1–66.4	0.8	0.2–2.9	0.0
SAB $\beta = 2$	61.7	57.5–65.8	1.6	0.6–4.0	0.8
SAB $\beta = 3$	59.8	55.5–63.9	2.8	1.4–5.7	1.2
SAB $\beta = 5$	47.5	43.2–51.8	5.6	3.4–9.2	9.2
SAB $\beta = 10^*$	5.4	3.8–7.7	48.4	42.2–54.6	66.8
CFG $\gamma = 1.5$	58.3	54.0–62.4	1.2	0.4–3.5	2.4
CFG $\gamma = 3$	40.4	36.3–44.7	8.4	5.6–12.5	20.0
CFG $\gamma = 5$	19.2	16.1–22.8	24.4	19.5–30.1	50.0
Prompt repetition	51.7	47.4–56.0	3.2	1.6–6.2	6.4
SmolLM2-1.7B ($\beta^* = 3$)					
Generic prompt	64.2	60.0–68.2	4.4	2.5–7.7	0.0
Safety prompt	12.1	9.6–15.2	15.2	11.3–20.2	0.0
SAB $\beta = 1.5$	10.6	8.2–13.5	21.3	16.7–26.8	0.0
SAB $\beta = 2$	10.2	7.9–13.1	26.0	21.0–31.8	0.4
SAB $\beta = 3^*$	8.1	6.0–10.7	32.1	26.6–38.2	0.0
SAB $\beta = 5$	8.1	6.0–10.7	47.0	40.9–53.2	0.0
SAB $\beta = 10$	6.0	4.2–8.3	66.0	59.9–71.6	0.0
CFG $\gamma = 1.5$	8.3	6.2–11.0	32.5	27.0–38.6	0.0
CFG $\gamma = 3$	2.1	1.2–3.7	55.4	49.2–61.5	8.8
CFG $\gamma = 5$	1.0	0.4–2.2	62.2	56.1–68.0	30.4
Prompt repetition	11.7	9.2–14.8	82.4	77.2–86.6	0.0
Phi-3.5-mini ($\beta^* = 1.5$)					
Generic prompt	0.4	0.1–1.4	4.8	2.8–8.2	0.0
Safety prompt	0.6	0.2–1.7	18.4	14.1–23.7	0.0
SAB $\beta = 1.5^*$	0.0	0.0–0.7	22.8	18.0–28.4	0.0
SAB $\beta = 2$	0.2	0.0–1.1	28.4	23.2–34.3	0.0
SAB $\beta = 3$	0.2	0.0–1.1	39.2	33.4–45.4	0.0
SAB $\beta = 5$	0.2	0.0–1.1	50.0	43.8–56.2	0.0
SAB $\beta = 10$	0.2	0.0–1.1	66.0	59.9–71.6	0.0
CFG $\gamma = 1.5$	0.2	0.0–1.1	27.2	22.1–33.0	0.0
CFG $\gamma = 3$	0.0	0.0–0.7	47.4	41.3–53.6	0.0
CFG $\gamma = 5$	0.0	0.0–0.7	56.9	50.6–62.9	0.4
Prompt repetition	0.0	0.0–0.7	36.9	31.2–43.1	0.0
Llama-3.1-8B ($\beta^* = 1.5$)					
Generic prompt	1.2	0.5–2.5	7.2	4.6–11.1	0.0
Safety prompt	0.0	0.0–0.7	18.4	14.1–23.7	0.0
SAB $\beta = 1.5^*$	0.2	0.0–1.1	24.0	19.1–29.7	0.0
SAB $\beta = 2$	0.2	0.0–1.1	32.4	26.9–38.4	0.0
SAB $\beta = 3$	0.0	0.0–0.7	44.0	38.0–50.2	0.0
CFG $\gamma = 3$	0.0	0.0–0.7	31.7	26.3–37.7	0.8
Prompt repetition	0.2	0.0–1.1	46.4	40.3–52.6	0.0

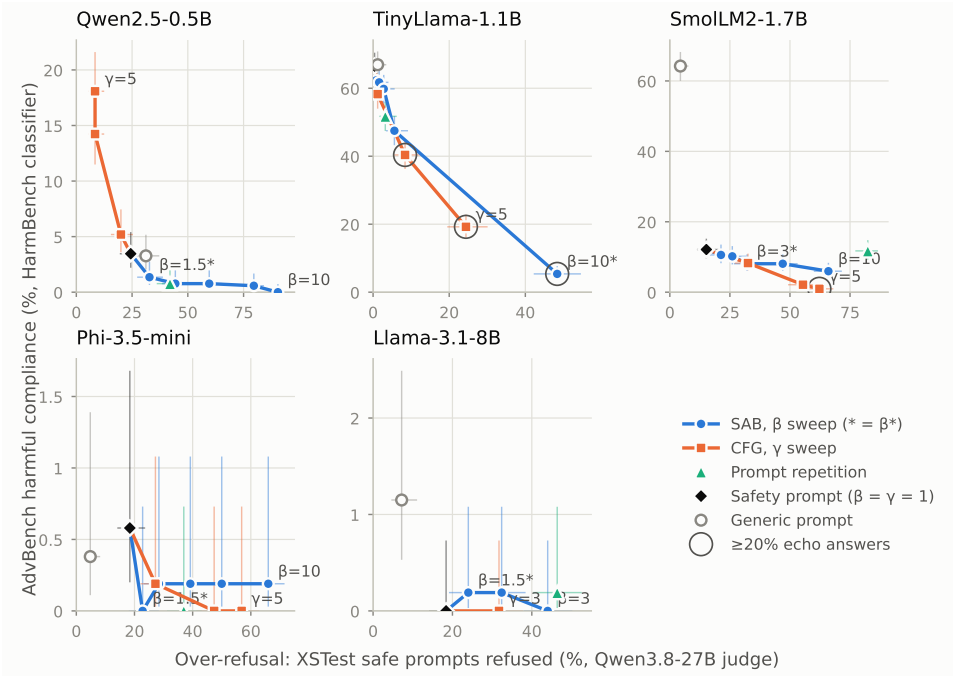

Figure 15: Safety/over-refusal frontier for all five models (the main text shows four). Axes and markers as in Figure 3. TinyLlama's SAB curve reflects the bias on its beginning-of-sequence token (Appendix T), not on the system prompt.