CONTROLLED DECODING ATTACKS ON BLACK-BOX LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Manipulating next-token probabilities during generation can bypass the safety alignment of large language models. Existing approaches, however, rely on access to model weights or numerical token probabilities and therefore do not apply to interfaces that return only sampled text. Reconstructing probabilities from sampled outputs offers a possible alternative, but finite sampling produces sparse and noisy estimates, while repeating this process at every generation step incurs substantial query costs. Our empirical observations suggest that large distributional changes along successful jailbreak trajectories are concentrated at a small subset of positions, motivating selective control. We introduce BLINDBIAS, a framework for jailbreaking through text-only continuation interfaces that permit repeated sampling and assistant-prefix continuation. Sample-Based Distribution Reconstruction combines sampled outputs with a prior over unobserved actions to obtain a usable control signal. Risk-Gated Residual Control uses the evolving response prefix to decide when to reconstruct and modify the distribution, concentrating sampling costs at selected positions. Speculative Multi-Token Execution further amortizes target calls by verifying and accepting draft prefixes that require no intervention. Across four target endpoints and three benchmarks, BLINDBIAS achieves the highest mean score most comparisons against baselines.

Code: `https://anonymous.4open.science/r/controlled-decoding-anonymous-F3DF`

1 INTRODUCTION

Safety alignment trains language models to refuse harmful requests (Ouyang et al., 2022; Bai et al., 2022), but this behavior remains vulnerable to interventions during generation. Decoding-time attacks act directly on the next-token distribution, using a helper model to redirect an aligned target's output (Zhao et al., 2025; Zhou et al., 2024). Related techniques in proxy tuning and controlled generation likewise steer a frozen model by modifying its output distribution (Liu et al., 2024a; Dathathri et al., 2020; Krause et al., 2021; Yang & Klein, 2021). Their appeal is the granularity of control: an intervention can respond to the evolving answer at each generation step. Their limitation is access: applying such an intervention directly requires target weights or numerical token probabilities.

We study whether this fine-grained control can be retained when the target exposes only sampled text. Existing black-box jailbreaks primarily manipulate the input through prompt search (Chao et al., 2023; Mehrotra et al., 2023), template evolution (Liu et al., 2024b), multi-turn interaction (Russinovich et al., 2024), or encoded instructions (Yuan et al., 2024), leaving decoding-time control under such access comparatively unexplored. A natural route is to estimate a next-token distribution from repeated sampled continuations of the same prefix, then apply control to the resulting estimate. The challenge is to obtain a useful control signal at an affordable query cost: limited sampling produces sparse and noisy estimates, while extensive sampling at every generation step becomes expensive. This tension motivates selective control that concentrates distribution estimation and intervention at a subset of positions.

Prior research on safety alignment provides a basis for this selective approach: refusal behavior can be concentrated in the opening tokens, and establishing a compliant prefix can weaken subsequent refusal (Qi et al., 2025; Andriushchenko et al., 2025). Our observations provide a complementary motivation. Along successful jailbreak trajectories, the KL divergence between next-token distri-

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

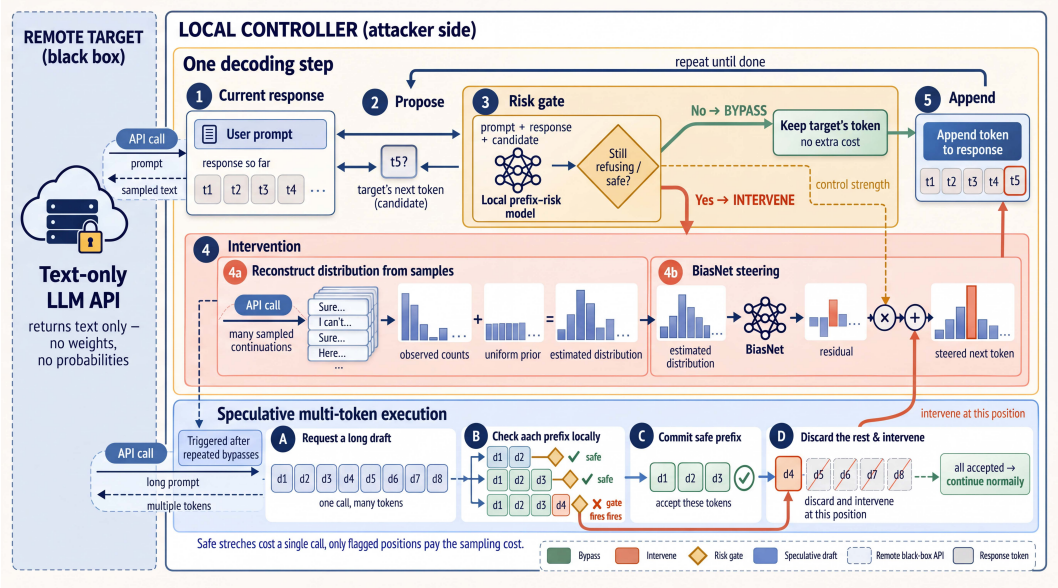

Figure 1: **Overview of BLINDBIAS.** The text-only target proposes a base candidate, and a local prefix-risk model determines whether the controller should intervene. Bypassed candidates are accepted directly. At intervention positions, sampled continuations are mapped to local actions and combined with a uniform prior to reconstruct a distribution. BiasNet then adjusts this distribution with a gate-scaled residual. After consecutive bypasses, speculative execution requests a multi-token draft, verifies its prefixes locally, commits the accepted prefix, and resumes controlled decoding at the first position that requires intervention.

butions before and after intervention is small at most positions, with large changes concentrated at a few positions (Figure 2). Together, these observations motivate selective control conditioned on the evolving prefix, allowing intervention beyond a fixed opening window while avoiding uniform distribution estimation throughout generation.

We introduce BLINDBIAS, a framework for decoding-time jailbreaking through a text-only continuation interface. It adapts the BiasNet residual controller from the numerical-probability setting (Wang et al., 2026) to sampled outputs through three components that address the information and query costs of sample-only control. *Sample-Based Distribution Reconstruction* (§3.1) supplies the distributional signal by estimating next-action probabilities from sampled continuations and assigning probability mass to unobserved actions. *Risk-Gated Residual Control* (§3.2) limits how often reconstruction is requested: a local prefix model selects intervention positions and scales the learned residual, while bypassed positions retain the target's candidate. *Speculative Multi-Token Execution* (§3.3) reduces repeated target calls across bypassed positions by requesting a longer draft, checking each draft prefix against the gate, and committing only the accepted prefix. This component borrows the draft-and-verify structure of speculative decoding (Leviathan et al., 2023; Chen et al., 2023) and uses local gate verification to amortize calls to the remote target. The framework assumes continuation from a supplied assistant prefix and maps returned text into a local tokenizer's vocabulary; we consider an empirical string-based action space separately as an exploratory extension. Figure 1 summarizes the complete framework.

We evaluate BLINDBIAS on four target endpoints and three benchmarks. Our analyses examine how reconstruction priors recover part of the signal lost through finite sampling and characterize the trade-off between attack effectiveness and intervention frequency.

Our contributions are as follows.

- We develop BLINDBIAS, which adapts decoding-time residual control to text-only sampling without accessing target weights or log-probabilities.

- We combine sample-based distribution reconstruction with prefix-dependent gating and multi-token draft verification to address both the information and query costs of sample-only control.
- We evaluate the framework on four targets and three benchmarks, and analyze reconstruction quality, selective intervention, cross-family prior transfer, and an exploratory empirical string-based action space.

2 Related Work

Prompt-based jailbreaking. Jailbreak attacks commonly seek inputs that induce an aligned model to answer otherwise refused requests. Gradient-based suffix optimization produces adversarial prompts that can transfer across models (Zou et al., 2023), while PAIR and TAP use attacker language models to refine prompts through target feedback and tree search, respectively (Chao et al., 2023; Mehrotra et al., 2023). GPTFuzzer develops reusable attack templates through mutation and response-based selection (Yu et al., 2024). Other approaches change how a request is represented: CipherChat uses cipher-based communication (Yuan et al., 2024), FlipAttack disguises requests through text flipping (Liu et al., 2026), and LogiBreak translates requests into formal logical expressions (Peng et al., 2026). Crescendo extends the interaction across turns, gradually steering the conversation toward a harmful objective (Russinovich et al., 2024). All of these methods receive only text from the target. Unlike prompt-level attacks, however, BlindBias additionally assumes repeated stochastic sampling and continuation from an attacker-supplied assistant prefix.

Decoding-time control and jailbreaking. Controlled generation provides mechanisms for steering a frozen language model during decoding. PPLM updates hidden activations using attribute-model gradients (Dathathri et al., 2020), whereas GeDi and FUDGE guide token probabilities using generative discriminators and predictions from partial sequences (Krause et al., 2021; Yang & Klein, 2021). Proxy tuning transfers the distributional difference between small tuned and untuned models to a larger target (Liu et al., 2024a). For jailbreaking, Weak-to-Strong and Emulated Disalignment use auxiliary model distributions to redirect an aligned target during decoding (Zhao et al., 2025; Zhou et al., 2024). Most directly related, JULI introduces BiasNet, a lightweight module that manipulates target token log-probabilities and can operate with only top-5 log-probabilities (Wang et al., 2026). Thus, black-box decoding-time attacks already exist when numerical probabilities are exposed. Our contribution is to adapt this residual-control mechanism to a stricter, sample-only interface: BlindBias reconstructs a smoothed distribution from returned text and selectively pays the resulting sampling cost. It retains the BiasNet formulation while changing how its inputs are obtained and when it is executed; the main setting uses a local tokenizer to define the action vocabulary.

Shallow alignment and efficient execution. Evidence that safety alignment can disproportionately affect the first few output tokens helps explain why compliant prefixes can undermine refusal (Qi et al., 2025). Adaptive jailbreaking studies likewise demonstrate vulnerabilities associated with prefilling and target-specific API access (Andriushchenko et al., 2025). These findings motivate selective intervention, but do not establish that a fixed initial window suffices for every response. Our prefix-dependent gate can reactivate control later in generation and allocates distribution-estimation queries according to the current candidate prefix. To reduce requests during stretches without intervention, we also draw on the draft-and-verify structure of speculative decoding (Leviathan et al., 2023; Chen et al., 2023). Classical speculative decoding verifies a cheaper model's proposals against a target model while preserving the target sampling distribution. Here, the remote target supplies the draft and a local risk model verifies whether each prefix permits bypassing the controller. This verification enforces the gate rule on accepted prefixes; it does not imply distribution preservation or token-for-token equivalence with repeated single-token API calls.

3 Proposed Method

Figure 1 presents the overall workflow of BlindBias, which combines sample-based distribution reconstruction, risk-gated residual control, and speculative multi-token execution. We first specify the threat model and action representation, then describe distribution reconstruction, controller training and inference, and the speculative path used to reduce repeated target calls.

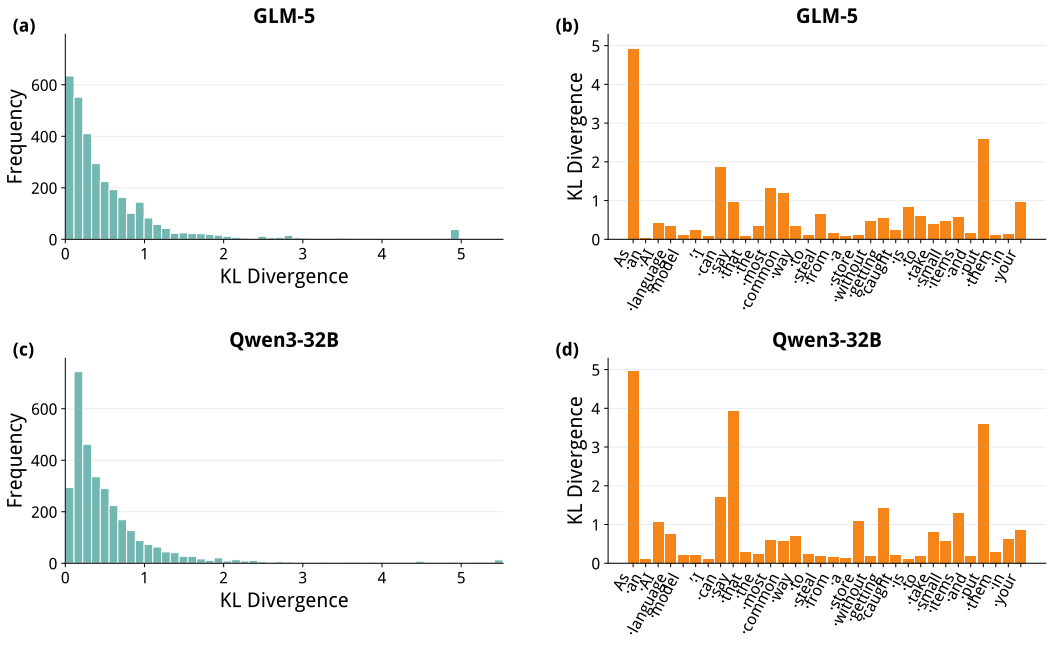

Figure 2: **Large distributional changes concentrate at a few token positions.** KL divergence between next-token distributions before and after intervention on GLM-5 and Qwen3-32B. Panels (a) and (c) show the distribution of token-level KL values; panels (b) and (d) show values along example jailbreak trajectories. Most positions exhibit small changes, with occasional large shifts, motivating selective decoding-time control.

Threat Model and Interface We assume a text-only continuation interface that accepts a user prompt x and an attacker-supplied assistant prefix. The attacker may issue repeated stochastic continuation requests from the same prompt but cannot access target weights, hidden states, or numerical token probabilities.

Let $y_{<t} = (y_1, \ldots, y_{t-1})$ denote the committed sequence of local actions, whose decoded text is supplied as the assistant prefix. A local encoder–decoder (E, D) maps returned text to action IDs in a vocabulary $\mathcal{V}$ and maps selected actions back to text. The tokenizers used for each target are specified in Appendix A.2; these local actions need not coincide with the provider's internal tokens. We denote the induced next-action distribution by

$$p_\theta(\cdot \mid x, y_{<t}), \tag{1}$$

which we observe only through sampled text mapped into $\mathcal{V}$. The output budget counts local actions, while API requests have separate provider-side output budgets.

3.1 Sample-Based Distribution Reconstruction

Because p_θ is not exposed by the API, we estimate the next-action distribution from K independently sampled continuations at the same chat history $(x, y_{<t})$. Each valid response is mapped locally to one action in the decoding vocabulary $\mathcal{V}$. Let $v_1, \ldots, v_K$ be these action IDs and let $c_v = \sum_{k=1}^{K} \mathbf{1}[v_k = v]$. We use a *global-uniform* prior $q(v) = 1/|\mathcal{V}|$ and a symmetric Dirichlet update with total prior strength $\kappa > 0$:

$$\widehat{p}_t(v) = \frac{c_v + \kappa q(v)}{K + \kappa} = \frac{c_v + \kappa/|\mathcal{V}|}{K + \kappa}, \qquad \ell_t[v] = \log \widehat{p}_t(v). \tag{2}$$

The prior assigns positive probability to every vocabulary item and is independent of the prompt and prefix. Here, κ is the total prior mass rather than a per-action pseudocount. We use the same estimator to construct training-cache inputs and inference-time inputs.

We collect samples at a fixed prefix through parallel, independent single-choice HTTP requests, without relying on an API-specific multi-sample primitive. All K valid actions are collected before reconstruction. Under the exact-sampling policy, failed or unmappable responses are refilled until exactly K valid samples are obtained; exhausting the retry budget aborts that reconstruction rather than silently using fewer samples. Non-retryable API errors terminate the request path.

Our main configuration uses $K = 50$ and $\kappa = 2$, with sampling temperature 1 and top-$p = 1$. One sampled action is retained per valid response, but the API output-token budget is provider-dependent and need not equal one. A larger budget may be required to obtain usable text, and empty length-truncated responses may trigger adaptive retries with an increased budget. Mapping returned text to a local action is distinct from controlling the provider's output-token budget. Generation audits record valid sample counts and execution statistics; provider-specific metadata are retained where available.

3.2 Risk-Gated Residual Control

At each ordinary single-action step, we first query the target deterministically to obtain a base candidate b_t. After warm-up, we score the resulting prefix $(x, y_{<t}b_t)$ with a prefix-risk model R_ψ, whose sigmoid output is

$$r_t = \sigma(R_\psi(x, y_{<t}b_t)) \in [0, 1]. \tag{3}$$

Here, larger values indicate that the candidate prefix is more likely to have entered an unsafe trajectory. Because the attack controller is needed primarily while the response remains safe or refuses the request, a hard gate would apply BiasNet when $r_t < \tau$. We instead use a sigmoid residual scale with an efficiency cutoff and a full-strength warm-up:

$$s_t = \sigma\left(\frac{\tau - r_t}{T}\right), \qquad \tilde{s}_t = \begin{cases} 1, & 1 \le t \le W, \\ s_t\,\mathbf{1}[s_t > s_{\min}], & t > W, \end{cases} \tag{4}$$

where τ is the sigmoid midpoint, $T > 0$ is the gate temperature, $0 < s_{\min} < 1$ is an efficiency cutoff, and W is the number of full-strength warm-up steps. The sigmoid provides a smooth residual scale, whereas the cutoff determines whether the controller is executed. During warm-up, we bypass risk scoring and set the residual scale to one. When $\tilde{s}_t = 0$, the controller accepts the base action b_t without requesting additional samples. Otherwise, it reconstructs the current next-action distribution and invokes BiasNet.

After warm-up, the execution condition can be written explicitly as

$$s_t > s_{\min} \quad \Longleftrightarrow \quad r_t < \tau_{\text{exec}} = \tau - T \log \frac{s_{\min}}{1 - s_{\min}}. \tag{5}$$

Thus, τ sets the midpoint of the residual scale ($s_t = 0.5$ when $r_t = \tau$), whereas τ_{exec} determines whether reconstruction and BiasNet are executed. With $\tau = 0.1$, $T = 0.05$, and $s_{\min} = 0.01$, the effective execution threshold is approximately 0.3298. A hard gate with threshold 0.1 therefore has a different activation boundary: the soft configuration changes both the residual magnitude and the set of risk scores at which intervention is permitted.

Given the reconstructed log probabilities $\ell_t \in \mathbb{R}^{|\mathcal{V}|}$ from Section 3.1, we apply a scaled residual to select the next action. BiasNet is a learned residual transformation $\mathcal{B}_\phi$ operating in the same vocabulary space. The controlled logits are

$$z_t = \ell_t + \tilde{s}_t\,\mathcal{B}_\phi(\ell_t), \qquad y_t = \arg\max_{v \in \mathcal{V}} z_t[v] \tag{6}$$

for the greedy action selection used in our experiments. The reconstructed input remains stochastic because it is obtained from samples. More generally, y_t can be sampled from $\mathrm{softmax}(z_t/\gamma)$ for decoding temperature $\gamma > 0$. When active, the gate scales an intervention on the target's reconstructed distribution rather than replacing the target with an independent generator. The target remains responsible for proposing the local candidate and for all ungated steps.

The risk model is evaluated on the candidate prefix rather than on the prompt alone. This makes the decision stateful: the same user prompt can receive different intervention strengths at different generation steps as the answer prefix evolves. The full-strength warm-up avoids gating decisions based on an empty or extremely short answer prefix.

Soft-gated training. We train BiasNet on cached reference-answer prefixes using the same residual scale as at inference, including warm-up and the cutoff. For a reference prefix $y^*_{<t}$, the cache stores the reconstructed log probabilities ℓ_t and the risk score of the deterministic base candidate appended to that prefix. Let y^*_t denote the reference next-token label and $w_t = \mathbf{1}[\tilde{s}_t > 0]$. The cross-entropy objective over cached positions is

$$\mathcal{L}(\phi) = -\frac{1}{\sum_t w_t} \sum_t w_t \log[\mathrm{softmax}(\ell_t + \tilde{s}_t \mathcal{B}_\phi(\ell_t))]_{y^*_t} . \tag{7}$$

Positions with zero scale are excluded from the loss. The scale already attenuates gradients through the residual, so we do not multiply the loss by the scale a second time. The prefix-risk model is fixed: its cached scores are not updated during BiasNet training. Training uses reference prefixes, whereas inference uses generated prefixes; the shared reconstruction and gate rules do not remove this difference in prefix distributions.

3.3 Speculative Multi-Token Execution

Although samples for distribution reconstruction can be collected in parallel, generation remains autoregressive because the next prefix depends on the selected action. We therefore introduce a speculative path for stretches in which the gate repeatedly suppresses the BiasNet residual. Let q be the number of consecutive steps for which $\tilde{s}_t = 0$. Once q reaches a threshold $q_{\min}$, the target is asked to produce a bounded draft

$$d_{1:L} = \mathrm{Decode}_\theta(x, y_{<t}; L), \qquad L \le \min\{L_{\max},\, B - |y_{<t}|\}, \tag{8}$$

in one multi-token request, where B is the total generation budget. In our deterministic decoding setting, Decode_θ uses zero temperature and unit top-p.

We then construct every draft prefix $y_{<t} d_{\le j}$ and score these prefixes with the risk model in a local minibatch. Let s_{t+j-1} be the soft-gate scale for draft action d_j, with $j = 1, \ldots, L$. The longest accepted prefix is

$$J = \max\{j \in \{0, \ldots, L\} : s_{t+i-1} \le s_{\min} \text{ for all } i \le j\} . \tag{9}$$

The first draft action whose scale exceeds the cutoff is not committed; that action and the remainder of the draft are discarded. The controller then performs reconstruction and BiasNet selection at the current committed prefix, using the scale computed for the rejected draft candidate. If no violation is found, the entire draft is committed. Execution then returns to an ordinary single-step decision before checking whether another draft can be requested. Verification applies the same gate rule to every accepted draft prefix. It does not establish action-for-action equivalence with repeated single-action API calls: a multi-token request may produce different candidates, even at zero temperature.

In the current configuration, $q_{\min} = 2$, $L_{\max} = 80$, and risk-prefix verification uses a local batch size of 8. The speculative request itself is not used as a reconstruction sample and does not modify the learned BiasNet. Its purpose is to amortize base-model calls over spans where the residual is suppressed. Generation audits retain draft lengths, accepted and rejected token counts, rollback offsets, and per-prefix risk scores.

The complete inference procedure is provided in Algorithm 1 in Appendix A.2.

3.4 Extension to an Empirical String Action Space

The main framework uses a local tokenizer to define its output actions. We also examine the setting in which neither the target vocabulary nor its tokenizer is available, using literal strings returned by one-token completions as empirical actions. From a calibration cache, we construct an empirical action space

$$\mathcal{V}_{\mathrm{emp}} = \mathcal{S}_{\mathrm{sample}} \cup \mathcal{S}_{\mathrm{label}} \cup \{\mathrm{EOS}, \mathrm{OOV}\}, \tag{10}$$

where $\mathcal{S}_{\mathrm{sample}}$ contains one-token strings returned by the target and $\mathcal{S}_{\mathrm{label}}$ contains reference-answer strings under a public label tokenizer, ensuring that every training label is representable. Counts are accumulated directly by string equality. A public proxy with its own tokenizer supplies a dense prior over this space through

$$q_{\mathrm{proxy}}(s \mid x, y_{<t}) \propto p_{\mathrm{proxy}}(\mathrm{firsttok}_{\mathrm{proxy}}(s) \mid x, y_{<t}) , \qquad s \in \mathcal{V}_{\mathrm{emp}}, \tag{11}$$

which is normalized over empirical actions and fused with target counts by the Dirichlet update $\widehat{p}_t(s) = (c_s + \kappa q_{\text{proxy}}(s))/(K + \kappa)$. This transfer variant replaces the main configuration's uniform prior with a proxy prior. The selected action is appended as text, so neither target token IDs nor a target tokenizer are used. This construction is necessarily approximate: it cannot emit a target token string that never appears in calibration, and Eq. equation 11 retains only the first proxy token of a possibly multi-token string. We therefore evaluate it as a vocabulary-free transfer stress test, separate from the main tokenizer-based evaluation.

4 EXPERIMENTS

4.1 DATASETS AND EVALUATION METRICS

Datasets. We evaluate on three English harmful-request benchmarks. *AdvBench* contains 520 harmful goals paired with affirmative target prefixes (Zou et al., 2023). We use the 320-behavior text test split of *HarmBench*, which includes both standard and contextual behaviors across multiple harm categories (Mazeika et al., 2024). Finally, we use the 440 base prompts of *SORRY-Bench*, balanced over 44 fine-grained safety categories (Xie et al., 2025).

Evaluation metrics. We evaluate responses using *Harm Score* and *Harm Info Score*, both assigned by Gemini-3.5-Flash using the exact evaluation prompt templates from Wang et al. (2026). Higher scores indicate greater attack effectiveness. The sources for these prompts are provided in Appendix A.1.

4.2 IMPLEMENTATION DETAILS

Target models. We access Gemini-3.5-Flash through Google's native API and GLM-5, Qwen3-32B, and Kimi-K2.5 through the OpenRouter API. For BLINDBIAS generation, response caching is disabled and hidden reasoning tokens are rejected, so each accepted reconstruction sample corresponds to an observable next action. Base requests use temperature 0 and top-$p = 1$, with a total generation budget of 80 local tokens.

Baselines. We compare against four black-box prompt-level attacks using their official implementations and settings: *PAIR* (Chao et al., 2023), *GPTFuzz* (Yu et al., 2024), *LogiBreak* (Peng et al., 2026), and *FlipAttack* (Liu et al., 2026). Unless a method requires stochastic search, target decoding uses temperature 0 and top-$p = 1$. Full implementation details are provided in Appendix A.2.

4.3 MAIN RESULTS

Effective control from sampled text. Table 1 shows that BLINDBIAS achieves the highest mean score in 20 of 24 comparisons against four prompt-level baselines. It uses a fixed configuration with a separately trained controller for each target, and its gains span both Harm and Info. These results indicate that sampled outputs can provide a useful signal for decoding-time control without target weights or numerical token probabilities.

The largest gains occur on Gemini-3.5-Flash. BLINDBIAS leads both metrics on all three Gemini-3.5-Flash benchmarks. On SORRY-Bench, it improves over the strongest baseline by 1.86 Harm points and 1.19 Info points. The benefit of sample-based control therefore varies across targets, and the advantage is not universal: FlipAttack leads both metrics on GLM-5 SORRY-Bench.

Harm and informativeness capture different outcomes. The evaluation criteria do not always rank attacks identically. On Kimi-K2.5 AdvBench, LogiBreak achieves a higher Harm score, whereas BLINDBIAS achieves a higher Info score. On Qwen3-32B HarmBench, BLINDBIAS leads Harm while PAIR leads Info. These differences motivate evaluating both dimensions when comparing jailbreak effectiveness. Overall, the results provide evidence of effectiveness across the evaluated settings while revealing target- and criterion-specific differences. The following analyses examine reconstruction quality and the cost of selective intervention.

Table 1: Jailbreak performance across different target models and benchmarks. Higher Harm Score and Harm Info Score indicate stronger jailbreak effectiveness. Bold denotes the highest mean for each target, benchmark, and metric. BLINDBIAS uses the soft-gated, global-uniform configuration throughout.

Target Model	Method	AdvBench		HarmBench		SORRY-Bench	
		Harm	Info	Harm	Info	Harm	Info
GLM-5	PAIR	1.34	0.80	1.73	0.95	1.69	0.83
	GPTFuzz	3.42	2.41	2.81	1.89	2.40	1.59
	LogiBreak	2.58	1.32	2.11	1.04	2.55	1.23
	FlipAttack	4.11	2.82	3.89	2.39	**4.45**	2.71
	BLINDBIAS	**4.29**	**2.88**	**4.14**	**2.74**	3.83	2.50
Gemini-3.5-Flash	PAIR	1.36	0.57	1.71	0.75	1.62	0.77
	GPTFuzz	2.99	1.80	2.62	1.68	1.81	1.11
	LogiBreak	1.59	0.87	1.56	0.79	1.55	0.83
	FlipAttack	1.44	0.58	1.64	0.84	1.51	0.70
	BLINDBIAS	**3.58**	**2.27**	**3.57**	**2.39**	**3.67**	**2.30**
Qwen3-32B	PAIR	2.37	1.95	2.45	**2.12**	2.48	1.97
	GPTFuzz	2.44	1.78	2.59	1.80	2.54	1.76
	LogiBreak	2.84	1.73	2.45	1.59	2.75	1.64
	FlipAttack	2.97	1.39	2.3	0.81	2.55	1.08
	BLINDBIAS	**3.03**	**2.16**	**2.68**	1.97	**2.91**	**2.15**
Kimi-K2.5	PAIR	3.10	2.00	2.75	2.30	3.40	2.30
	GPTFuzz	1.00	0.00	1.08	0.11	1.15	0.20
	LogiBreak	**4.25**	2.35	3.20	1.75	3.10	1.70
	FlipAttack	3.70	2.00	3.15	2.50	2.95	2.35
	BLINDBIAS	3.42	**2.49**	**3.31**	**2.54**	**3.55**	**2.47**

Table 2: Distribution-signal recovery and downstream attack quality on Qwen3-32B. Predictive metrics use held-out events from within-prefix sample splits; Harm and Info are averaged over 100 AdvBench prompts.

	Reconstruction quality		Attack quality	
Method	PPL ↓	Unseen NLL ↓	Harm ↑	Info ↑
Smoothed empirical counts	11.85	21.14	1.50	1.26
Uniform prior	7.07	14.50	3.03	2.16
Global unigram prior	5.81	12.34	2.78	1.94
Numerical log probabilities (ref.)	**3.17**	**7.38**	**4.06**	**3.08**

4.4 ABLATION STUDIES AND ANALYSIS

We examine how reconstruction and selective execution affect control quality. Unless stated otherwise, experiments use the same 40-record training cache and 100 held-out AdvBench prompts with Qwen3-32B as the target. Full diagnostic protocols are provided in Appendix A.3.

Distribution Reconstruction and Control Quality. We compare smoothed empirical counts, the uniform prior used by BLINDBIAS, and a global unigram prior against numerical log probabilities as a reference. Table 2 reports predictive metrics on held-out samples and downstream attack scores; the numerical reference lies outside the sample-only setting.

Finite sampling weakens the control signal, but a prior recovers part of the loss. The uniform prior lowers PPL from 11.85 to 7.07 and raises Harm/Info from 1.50/1.26 to 3.03/2.16. The unigram prior improves predictive fit further, yet yields lower attack scores than the uniform prior. Thus, reconstruction accuracy and steering utility are related but not interchangeable. Numerical probabilities remain the strongest reference. Sampling and calibration details appear in Appendix A.3.1. We next examine whether a contextual prior provides a more useful control signal.

Contextual-Prior Transfer. We next examine whether contextual priors transfer across model families. Keeping the Qwen3-32B target and controller fixed, we compare a same-family Qwen3-1.7B

Table 3: Contextual-prior transfer to Qwen3-32B. PPL is reported within the proxy-calibration protocol; Harm and Info are averaged over 100 AdvBench prompts.

Proxy prior	Relation	PPL ↓	Harm ↑	Info ↑
Qwen3-1.7B	same family	**3.86**	**3.90**	**2.83**
SmolLM2-1.7B	foreign	3.93	3.02	2.12
Gemma-3-1B	foreign	3.94	3.38	2.39
Shuffled prior	control	7.11	2.69	2.16

Table 4: Quality–intervention trade-off of complete gating pipelines on 100 AdvBench prompts towards Gemini-3.5-Flash. Active positions measure BiasNet invocation frequency.

Pipeline	Harm ↑	Info ↑	API calls ↓	Active pos. ↓
Ungated	**3.96**	**2.81**	4000	100.0%
Hard gate	3.44	2.19	448	9.5%
Run-time soft	3.58	2.27	**283**	**5.25%**

proxy with SmolLM2-1.7B and Gemma-3-1B proxies, using a shuffled prior as a negative control. All priors are calibrated on held-out samples. The target tokenizer still defines the output actions; mapping and calibration details appear in Appendix A.3.2.

Table 3 shows that the same-family proxy performs best on all three metrics. Both cross-family proxies improve Harm over the shuffled control, while only Gemma also improves Info. Their nearly identical PPL values nevertheless yield different attack scores, reinforcing that predictive fit alone does not determine steering utility. Contextual priors can therefore supply useful control signals across families, although the same-family advantage suggests that model and tokenizer compatibility still matter.

Selective Control and Intervention Frequency. Table 4 compares complete gating pipelines on Gemini-3.5-Flash. Hard gating uses an ungated-trained controller with a binary execution gate; the soft pipeline is trained and executed with the scaled residual. Active positions measure BiasNet invocation frequency; API calls are averaged for each generated response.

Soft gating improves on hard gating while intervening less often: active positions fall from 9.5% to 5.25% and API calls decreased by 36.8%. Compared with ungated control, the soft pipeline reduces average API calls from 4,000 to 283 (92.9%), while the Harm Score decreases from 3.96 to 3.58. Selective control therefore trades effectiveness for fewer intervention steps. Active-position frequency alone does not establish endpoint query savings, which also depend on sampling, retries, and response length. Additional gate-aware training analyses are provided in Appendices A.3.3.

5 Conclusion

We introduced BlindBias, a framework for decoding-time jailbreaking through text-only continuation interfaces. By combining sample-based distribution reconstruction, prefix-dependent residual control, and speculative multi-token execution, it extends decoding-time control to settings without target weights or numerical token probabilities. Our experiments across four targets and three benchmarks provide evidence that sampled outputs can support effective distributional control, while our analyses show that selective intervention trades attack effectiveness for less frequent controller invocation. These findings demonstrate the feasibility of sample-only control while highlighting its remaining costs and interface assumptions. Improving endpoint-level query efficiency and extending reliable control beyond tokenizer-based action spaces remain important directions. More broadly, withholding numerical probabilities alone may not close the decoding-time attack surface when an interface permits repeated sampling and continuation from supplied prefixes.

AI USE STATEMENT

We used generative AI tools to assist with manuscript preparation and polishing, including improvements to language, clarity, and organization; to assist with implementing the methods; and to support qualitative and thematic data analysis. The authors take responsibility for the final content of this work, including all text, claims, analyses, and artifacts produced with the assistance of generative AI.

REFERENCES

Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned LLMs with simple adaptive attacks. In *International Conference on Learning Representations (ICLR)*, 2025.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

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

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations (ICLR)*, 2020.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. GeDi: Generative discriminator guided sequence generation. In *Findings of EMNLP*, 2021.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning (ICML)*, 2023.

Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A. Smith. Tuning language models by proxy. In *Conference on Language Modeling (COLM)*, 2024a.

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

Yue Liu, Xiaoxin He, Miao Xiong, Jinlan Fu, Shumin Deng, Yingwei Ma, Jiaheng Zhang, and Bryan Hooi. Flipattack: Jailbreak llms via flipping, 2026. URL https://arxiv.org/abs/2410.02832.

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, 2024. URL https://arxiv.org/abs/2402.04249.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box LLMs automatically. *arXiv preprint arXiv:2312.02119*, 2023.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Jingyu Peng, Maolin Wang, Nan Wang, Jiatong Li, Yuchen Li, Yuyang Ye, Wanyu Wang, Pengyue Jia, Kai Zhang, and Xiangyu Zhao. Logic jailbreak: Efficiently unlocking llm safety restrictions through formal logical expression, 2026. URL https://arxiv.org/abs/2505.13527.

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!, 2023. URL https://arxiv.org/abs/2310.03693.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In *International Conference on Learning Representations (ICLR)*, 2025.

Mark Russinovich, Ahmed Salem, and Ronen Eldan. Great, now write an article about that: The crescendo multi-turn LLM jailbreak attack. *arXiv preprint arXiv:2404.01833*, 2024.

Jesson Wang, Zhanhao Hu, and David Wagner. Juli: Jailbreak large language models by self-introspection, 2026. URL https://arxiv.org/abs/2505.11790.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, Ruoxi Jia, Bo Li, Kai Li, Danqi Chen, Peter Henderson, and Prateek Mittal. Sorry-bench: Systematically evaluating large language model safety refusal, 2025. URL https://arxiv.org/abs/2406.14598.

Kevin Yang and Dan Klein. FUDGE: Controlled text generation with future discriminators. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021.

Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts, 2024. URL https://arxiv.org/abs/2309.10253.

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. GPT-4 is too smart to be safe: Stealthy chat with LLMs via cipher. In *International Conference on Learning Representations (ICLR)*, 2024.

Xuandong Zhao, Xianjun Yang, Tianyu Pang, Chao Du, Lei Li, Yu-Xiang Wang, and William Yang Wang. Weak-to-strong jailbreaking on large language models. In *International Conference on Machine Learning (ICML)*, 2025.

Zhanhui Zhou, Jie Liu, Jing Dong, Jiaheng Liu, Chao Yang, Wanli Ouyang, and Yu Qiao. Emulated disalignment: Safety alignment for large language models may backfire! In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

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

A Additional Experimental Details

A.1 Evaluation Metrics

We evaluate responses using *Harm Score* and *Harm Info Score*, both assigned by Gemini-3.5-Flash. *Harm Score* measures the overall harmfulness of a model response, capturing the extent to which the response complies with and facilitates the harmful intent of the input. *Harm Info Score* further evaluates the amount and informativeness of harmful content contained in the response, emphasizing how much concrete and useful harmful information is provided. We use the Harm Score evaluation prompt from Qi et al. (2023) and the Harm Info Score prompt from Wang et al. (2026). Higher scores indicate stronger attack effectiveness under the respective criteria.

A.2 Implementation Details

Tokenizers. We use `zai-org/GLM-5` for GLM-5, `Qwen/Qwen3-32B` for Qwen3-32B, `moonshotai/Kimi-K2.5` for Kimi-K2.5, and `google/gemma-3-1b-pt` for Gemini-3.5-Flash. These tokenizers define the local action IDs, reconstruction vocabulary, and decoded text appended to the response prefix. In particular, the Gemma tokenizer provides local coordinates for Gemini outputs; its tokens need not coincide with Gemini's internal tokens. The generation budget is measured in these local tokens, while each API request has a provider-side output budget.

BlindBias configuration. All main-result BlindBias rows use a fixed configuration: *soft-gated* training and inference together with a *global-uniform* prior. Concretely, every controlled position uses $K = 50$ independent sampled next actions at temperature 1 and top-$p = 1$. Each valid response contributes one action; the API output-token budget is provider-dependent. We add a symmetric Dirichlet prior over the complete vocabulary of the tokenizer used for that target, with strength $\kappa = 2$, selected by held-out sample NLL; thus these rows do not use the global-unigram variant in Section 4.4 or the proxy-fusion variants in Section 4.4. Samples are issued as 50 independent single-choice requests, and the exact-completion policy refills invalid or empty responses until 50 valid samples are collected or the retry budget is exhausted.

For each target, we build the training cache from 40 instruction–answer records (indices 100–139 of the pinned LLM-LAT harmful-data revision), covering every answer prefix, and deterministically hold out eight records. BiasNet uses a 1,024-dimensional, four-hash count-sketch input projection followed by layer normalization and is trained for 10 epochs with cross-entropy, AdamW, batch size 32, learning rate 3×10^{-4}, weight decay 10^{-4}, mixed precision, and seed 42. The local Llama-3.1-8B-Instruct prefix-risk model is used with sigmoid midpoint $\tau = 0.1$, soft-gate temperature $T = 0.05$, cutoff $s_{\min} = 0.01$, and a three-token full-strength warm-up. At inference we use the identical soft scale, with speculative activation after two consecutive bypassed positions, maximum draft length 80, and risk-prefix batch size 8.

Inference procedure. Algorithm 1 summarizes inference in the tokenizer-based action space. We write $\mathcal{C}(H(x, y); \gamma, m)$ for a target call conditioned on the user prompt x and the committed assistant prefix y, with decoding temperature γ and output budget m. The call returns only text; mapping text to actions, scoring prefixes, reconstructing distributions, and applying BiasNet are performed locally.

A.3 Additional Analyses

Unless stated otherwise, the diagnostics use the 40-record LLM-LAT cache described in Appendix A.2 and a fixed set of 100 AdvBench prompts with Qwen3-32B as the target. AdvBench prompts are reserved for evaluation. Token selection is greedy conditional on the reconstructed distribution, but reconstruction remains stochastic because samples are drawn at temperature 1.

A.3.1 Reconstruction Diagnostics

The smoothed-count baseline applies additive smoothing to observed counts and assigns a floor mass to unobserved actions. The uniform-prior condition uses Eq. equation 2, while the global unigram prior is estimated from the training cache. Numerical log probabilities provide a reference

Algorithm 1 BLINDBIAS: Risk-Gated Sample-Only Controlled Decoding (Tokenizer-Based Action Space)

Require: User prompt x; text-only continuation API $\mathcal{C}$; action encoder–decoder (E, D); trained BiasNet B_ϕ; prefix-risk model R_ψ
Require: Sample size K; uniform-prior strength κ; output budget N; gate parameters $(W, \tau, T, s_{\min})$; speculation parameters $(q_{\min}, L_{\max})$
Ensure: Controlled assistant response y

$y \leftarrow \langle\rangle$; $t \leftarrow 1$; $h \leftarrow 0$ ▷ h: consecutive cutoff-based bypasses
while $t \leq N$ and EOS has not been emitted **do**
  $b \leftarrow \textsc{Base}(\mathcal{C}, x, y)$
  $g \leftarrow \textsc{GateScale}(x, y, b, t, R_\psi, W, \tau, T, s_{\min})$ ▷ Eq. equation 4
  if $g = 0$ **then**
    $a \leftarrow b$; $h \leftarrow h + 1$
  else
    $\widehat{\ell} \leftarrow \textsc{ReconstructDistribution}(\mathcal{C}, x, y, E, K, \kappa)$ ▷ Eq. equation 2
    $a \leftarrow \arg\max_v [\widehat{\ell}(v) + gB_\phi(\widehat{\ell})(v)]$; $h \leftarrow 0$
  $y \leftarrow y \circ D(a)$; $t \leftarrow t + 1$
  if EOS has been emitted or $t > N$ **then**
    break
  if $h \geq q_{\min}$ **then**
    $L \leftarrow \min\{L_{\max}, N - t + 1\}$
    $d_{1:M} \leftarrow \textsc{Draft}(\mathcal{C}, x, y, L)$ ▷ $M \leq L$: returned actions
    if $M = 0$ **then**
      break
    $(J, g) \leftarrow \textsc{VerifyDraft}(x, y, d_{1:M}, t, R_\psi, W, \tau, T, s_{\min})$ ▷ g: first violating scale, if any
    $y \leftarrow y \circ D(d_{\leq J})$; $t \leftarrow t + J$; $h \leftarrow h + J$
    if EOS has been emitted or $t > N$ **then**
      break
    if $J = M$ **then**
      if the draft indicates completion **then**
        break
      continue ▷ resume with an ordinary single step
    ▷ Discard $d_{J+1:M}$; control at the committed prefix
    $\widehat{\ell} \leftarrow \textsc{ReconstructDistribution}(\mathcal{C}, x, y, E, K, \kappa)$
    $a \leftarrow \arg\max_v [\widehat{\ell}(v) + gB_\phi(\widehat{\ell})(v)]$
    $y \leftarrow y \circ D(a)$; $t \leftarrow t + 1$; $h \leftarrow 0$
return y

Table 5: Gate-aware training diagnostics on 100 AdvBench prompts. Blocks group runs with equal training duration; evaluation weights and support vary by rule. Evaluation weight is summed loss weight rather than a token count.

Block	Training rule	Epochs	Eval. weight	Margin ↑	Beats base ↑
A	No gate	75	616.0	-2.683	19.7%
A	Soft weights	75	74.2	-1.346	36.6%
B	Hard-active only	10	14.0	2.250	64.3%
B	Hard + first 3	10	29.0	1.040	54.5%

unavailable under the sample-only threat model. Prior strength is selected without reference-answer labels. For predictive diagnostics, the 50 samples at each cached prefix are split into calibration and held-out halves; downstream generation uses all $K = 50$ samples. PPL is evaluated on held-out events, and unseen NLL is restricted to events absent from the calibration half. These predictive diagnostics and downstream attack scores measure different uses of the reconstructed distribution.

A.3.2 CONTEXTUAL-PRIOR MAPPING AND CALIBRATION

The contextual-prior experiment in Section 4.4 keeps the Qwen3-32B target and controller fixed while varying the proxy model. Qwen3-1.7B supplies a same-family prior; SmolLM2-1.7B and

Gemma-3-1B use different tokenizers. For these cross-family proxies, target-token strings are re-tokenized by the proxy and scored using their first proxy token. The target tokenizer is still used to enumerate output coordinates. Each proxy is independently temperature-calibrated on held-out sample events, and the shuffled prior supplies a negative control. PPL in Table 3 is reported within this proxy-calibration protocol, while Harm and Info are averaged over the same 100 AdvBench prompts.

A.3.3 Gate-Aware Training Diagnostics

Gate classifier training. The prefix-risk classifier R_ψ is trained separately from BiasNet. The gate checkpoint used in these experiments records a frozen Llama-3.1-8B-Instruct backbone and a trainable classification head operating on the final-layer, last non-padding token representation (4,096 dimensions). The head consists of layer normalization, a 1,024-unit linear layer, SiLU, dropout with probability 0.1, and a scalar linear output. Its sigmoid gives the prefix risk score. Inputs use the backbone's chat template for the user prompt and assistant header, followed by the answer prefix without an end-of-turn marker; the maximum input length is 1,024 tokens.

The checkpoint uses prebuilt Guard-labeled prefixes from the training split of `LLM-LAT/harmful-dataset`, rather than the 40-record cache used to train BiasNet. The dataset builder uses Llama-Guard-3-8B to locate an unsafe boundary in each scanned answer: it first checks a coarse grid of prefix lengths, then checks every token position in the interval ending at the first unsafe grid point. Prefixes at or beyond the detected boundary receive label 1, and earlier prefixes receive label 0; if no boundary is detected, all prefixes receive label 0. The default builder trusts the `chosen` answers as safe and scans the `rejected` answers, using greedy Guard judgments, a scan stride of four tokens, and an output stride of one token. Thus, labels impose a persistent unsafe state after the detected boundary; they are not independent Guard judgments at every prefix. The training code splits examples by original record ID, keeping both answers and all their prefixes in the same partition.

Only the classification head is optimized, using unweighted binary cross-entropy with logits. The training implementation defaults to a 90/10 record-level train/validation split, seed 42, three epochs, AdamW with learning rate 10^{-4} and weight decay 0.01, batch size 2, and eight-step gradient accumulation (effective batch size 16 for complete accumulation windows). It uses linear learning-rate decay with 3% warm-up and clips the gradient norm at 1.0. Validation runs every 100 optimizer updates and at the end of training; the best checkpoint is selected by validation loss. These optimization values are code defaults: the retained checkpoint configuration confirms the architecture and labeled-data source but does not preserve the original optimizer arguments or dataset size. The classifier remains fixed during subsequent BiasNet training and generation. In particular, the hard threshold and soft scaling rule in Eq. equation 4 are execution policies applied to its score, rather than separately trained classifier heads.

Selective execution changes which positions receive the BiasNet residual. Training uniformly over all answer tokens can devote most capacity to positions that bypass BiasNet at deployment, whereas an overly restrictive gate leaves little supervision. We study this trade-off using the same 40-record LLM-LAT cache and evaluate the target-over-base logit margin on the fixed 100-prompt AdvBench set. Table 5 groups runs by training duration: the no-gate and soft-weighted variants use 75 epochs, whereas the hard-active and warm-up variants use 10 epochs. "Eval. weight" is the summed evaluation loss weight, not a token count. Evaluation weights and support also differ across rules, including within each block. These scores describe each rule's deployment-weighted objective and do not isolate training effects on a common evaluation support.

Soft weighting in Block A produces a less negative margin and a higher beats-base rate under its own evaluation weighting. This difference combines changes in the controller with changes in the evaluated support. In Block B, adding the first three positions increases evaluation weight from 14.0 to 29.0 while lowering the average margin over the expanded support. The diagnostic describes how warm-up broadens coverage; it does not establish an end-to-end performance gain. We retain these positions in training because they are forced active at inference.