Safety Reconstructed: Generative Modeling via Masked Diffusion Builds Strong Safety Guardrails

Gert Lek[1] **Abele Mălan**[1] **Chaoyi Zhu**[2]
Pin-Yu Chen[3] **Robert Birke**[4] **Lydia Chen**[1,2]

[1]University of Neuchâtel [2]Delft University of Technology
[3]IBM Research [4]University of Turin

gert.lek@unine.ch

Abstract

Guard models are the last line of defense between a language model and a harmful output, yet their training objective is surprisingly narrow. Existing guards learn to predict a single verdict token from a conversational context, concentrating supervision on a single target. The consequences are structural: models latch onto shortcut features, are overconfident, and remain sensitive to where safety evidence appears in the sequence rather than its role in the full context. We propose a different framing. Rather than predicting a label from text, our **LLaDA-Guard** asks which label better explains the text: scoring the prompt or response under each label hypothesis and classifying based on their difference. This shifts supervision to every token in the moderated region, forcing the model to account for full content rather than its most discriminative fragments. We instantiate this idea with a masked diffusion language model, fine-tuning LLaDA-8B-Instruct with a class-conditional reconstruction objective using LoRA and requiring no architectural changes beyond the base model. LLaDA-Guard leads on average rank against discriminative baselines trained on stronger backbones across seven held-out safety benchmarks, while exhibiting substantially better confidence calibration (ECE 0.0875 vs. 0.1384 for Qwen3Guard), less over-defense on benign prompts with `unsafe`-looking cues, and less prompt leakage when moderating responses. Its generative nature further enables token-level risk localization as a natural byproduct, yielding a pipeline for rewriting `unsafe` prompts into `safe` equivalents without additional training and achieving a 60.7% average conversion-to-`safe` rate. Code and the trained checkpoint are available at `https://github.com/Gertlek/LLaDA-Guard2026`.

1 Introduction

Large language models (LLMs) are increasingly deployed across consumer assistants, developer tools, and high-stakes domains such as healthcare and education [16, 6, 54]. At this scale, a non-trivial share of interactions involves inappropriate or malicious requests [52, 17, 2], and even well-aligned models remain susceptible to jailbreaks, which reliably bypass their built-in refusal behavior [51, 60, 46]. The long tail of usage uncovers failure modes not anticipated during alignment training [59, 58].

Content moderation has therefore become a load-bearing component of the LLM stack. Dedicated guard models [21, 15] are a mechanism by which providers enforce safety policies at inference time, also serving as research instruments for scoring red-teaming attacks [35] and measuring refusal behavior on safety benchmarks [44, 8]. Given a user prompt x^P and a model response x^R, a guard model has two related tasks to solve: prompt moderation, which asks whether the user's request is `unsafe`, and response moderation, which asks whether the response is `unsafe` given the prompt.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

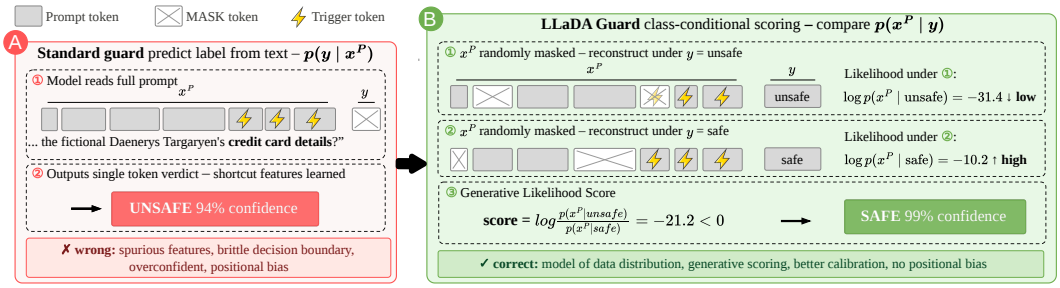


Figure 1: Illustrative comparison between standard verdict-token moderation and our class-conditional reconstruction approach. In the standard paradigm Ⓐ, the prompt and/or response is visible context and the guard predicts the masked verdict token y. In our method Ⓑ, a candidate label y is made visible as a safety hypothesis, while the judged region is masked and reconstructed under competing `safe` and `unsafe` hypotheses. The moderation decision is therefore based on which label better explains the judged text, rather than on discriminative features for a single output token.

Existing guard models approach moderation as direct label prediction, estimating a discriminative decision rule of the form $p_\theta(y \mid x)$, where p_θ denotes the guard model's predictive distribution, $y \in \{\texttt{safe}, \texttt{unsafe}\}$, and x denotes the text being moderated: a prompt x^P, response x^R, or prompt-response pair $[x^P, x^R]$. In modern LLM-based guardrails, this gets framed as a next-token prediction task for y. The objective is discriminative: predict the label from the conversational input.

Recent works show guardrail systems are effective in practice, but also suffer from several systemic weaknesses, including miscalibration and overconfidence under jailbreaks and response-model shifts [29], degraded robustness under adversarial attacks [51], and distribution shift [36, 28]. These limitations are structural, not merely empirical. In direct verdict-token classification, supervision concentrates on a single prediction target, which encourages the model to rely on a set of discriminative features rather than learning a rich model of what `unsafe` and `safe` text looks like [26]. Further, current autoregressive guard architectures inherit positional bias from causal attention [30, 42], yielding decisions that are sensitive to the position of evidence within the analyzed sequence.

In this paper, we cast text moderation as class-conditional generative scoring over the judged text region. Rather than estimating $p_\theta(y \mid x)$ from a full conversational input, we compare how well the moderated region is explained under competing safety labels: $p_\theta(x_M \mid x_{\bar{M}}, y)$, where M is the moderated region and $x_{\bar{M}}$ is the visible context, following the class-conditional scoring view of generative classification [5]. Moderation thus becomes a generative problem: supervision moves from a single verdict token to the moderated region, requiring the model to explain the observed text under each label hypothesis [39, 36]. The resulting score is tied to the judged content rather than to shortcut features in the context. This view makes prompt moderation, response moderation, and verdict prediction different choices of a target region in the same sequence $[x^P, x^R, y]$; verdict-token prediction is the special case where only the label token is scored, as illustrated in Figure 1. This is especially important for response moderation, where the verdict should depend on the response content rather than on prompt features that co-occur with `unsafe` examples [26, 11].

The class-conditional view naturally suggests Masked Diffusion Language Models (MDMs) as a statistical inference framework. MDMs score masked text conditioned on visible context [3, 47, 39, 45], thus representing moderation as conditional reconstruction over masked regions in the conversation, rather than reducing the prompt- and response-moderation plausibility to a single next-token prediction. This provides a unified mechanism for prompt moderation, response moderation, and verdict prediction. It also gives the final safety score a likelihood-ratio interpretation: with true class-conditional likelihoods, thresholding the score recovers the Neyman–Pearson test [37].

This idea results in LLaDA-Guard, the first MDM guardrail model, with class-conditional generative scoring built on LLaDA-8B-Instruct [39]. Our contributions are threefold:

1. We recast guardrail content moderation from verdict-token prediction into class-conditional scoring over regions of a shared sequence $[x^P, x^R, y]$. This yields prompt moderation, response moderation, and verdict prediction as different choices of target region, while separating the judged text from the context used to interpret it.

2. We show that masked diffusion language models implement this class-conditional scoring formulation directly. Changing the label condition gives the competing `safe`/`unsafe` hypotheses, while changing the masking pattern selects whether the model scores prompts or responses; we train this with a likelihood-ratio moderation loss and a generative reconstruction term.
3. We evaluate the resulting LLaDA-Guard and find that it is competitive with guard baselines derived from stronger models, achieving the best average F1 on held-out benchmarks while remaining close to the best average AUPRC. Importantly, LLaDA-Guard improves usability by achieving better calibration and reducing both overconfident misclassifications and positional bias.

2 Related work

Guard Models Recent guardrail models such as Llama Guard 1-4 [21, 31], WildGuard [15], ShieldGemma [56], Granite Guardian [41], NemoGuard [40], and Qwen3Guard [57] treat safety moderation as a supervised classification problem over prompts or prompt-response pairs. Recent work has extended this line to multilingual moderation [24, 10, 23], task-adaptive distillation [50], and cross-modal moderation [49, 32]. These systems estimate whether an input is `unsafe` under a policy taxonomy, showing strong practical utility and competitive performance on benchmarks. Their primary use cases are pre-generation filtering of user inputs and post-generation filtering for harmfulness, refusal, or policy compliance. Moreover, by decoding the hazard category from the policy taxonomy, they can predict the specific class of harmful behavior to which the conversation belongs. Such systems are useful not only as filters, but also as safety evaluators during research and development. They are employed for red-teaming [35, 51], post-training analysis, and evaluation [8]. This broad role explains why guard models have become central to deployment and research, even when they are not the final moderation mechanism itself. Two limitations stand out. Modern guard models are typically verdict-token classifiers trained on large, often proprietary policy-labeled mixtures [15, 12, 57]. Further, strong accuracy does not imply reliable probabilities: LLM-based guards are miscalibrated and overconfident under jailbreaks [29, 26].

Generative versus discriminative classification The distinction between discriminative and generative classification has a long history in machine learning: early comparisons showed trade-offs in sample efficiency and asymptotic error [38], while later text-classification work found similar differences in generalization behavior [55]. In language modeling, the distinction reappears as the contrast between direct classification and generative classification. Min et al. [36] shows that generative approaches improve performance in text-classification tasks under class-imbalance or in generalization to unseen labels, arguing that these explain every input token, rather than learning a separating boundary. Recent work makes this robustness argument more concrete for modern language and diffusion models. In the continuous domain, diffusion models have recently been used directly as zero-shot classifiers [25, 7]. Chen et al. [5] show that a diffusion model can be used for robust generative class-conditional classification. Related, Li et al. [26] argue that generative classifiers are less prone to shortcut solutions because their objective models the data manifold for each class, not purely the features sufficient for separation.

Diffusion language models and scoring Our method draws upon progress in diffusion language models. Unlike autoregressive language models, which generate left-to-right, diffusion language models operate through iterative denoising or masked reconstruction, allowing any-order generation. Discrete diffusion generalizes the continuous-space DDPM framework [18, 48] to categorical data [19, 3, 4]. In masked diffusion language models [45], the training objective is a reconstruction loss over the masked region tokens, conditioned on the unmasked part. Unlike image diffusion, which operates over continuous pixels [18, 48], discrete diffusion over tokens requires either embedding-space relaxations or direct masking of categorical states [3, 33]. As a result, the model is not trained to predict only a single next token, but to estimate conditional distributions of the form $p_\theta(x_M \mid x_{\bar{M}})$ for arbitrary masked subsets M of a sequence. LLaDA [39] shows such models scale while retaining flexible mask-based inference, and is competitive with similar-sized AR-LLMs.

3 LLaDA-Guard

Standard guard models cast moderation as verdict-token prediction: given the full conversation, generate `safe` or `unsafe`. LLaDA-Guard instead scores the moderated region under two competing safety hypotheses and classifies according to which hypothesis better reconstructs the moderated

region. Prompt moderation, response moderation, and ordinary verdict prediction therefore differ only in the choice of moderated region within a shared sequence $[x^P, x^R, y]$, where x^P is the user prompt, x^R is the model response, and $y \in \{\texttt{safe}, \texttt{unsafe}\}$ is the safety label.

This view decomposes safety moderation into three related questions. The first is the **discriminative verdict** question $p(y \mid x^P, x^R)$, which asks whether the conversation should be labeled `safe` or `unsafe`. Existing guard models are optimized for this question by predicting the verdict token y directly. The second is the **prompt plausibility** question $p(x^P \mid y)$, which asks how likely the prompt is under a given safety label. This is a natural task for prompt moderation; rather than predicting the verdict, the model evaluates whether the prompt is better explained as `safe` or `unsafe`. The third is the **response plausibility** question $p(x^R \mid x^P, y)$, which asks how likely the response is given the prompt context under a label hypothesis, and is natural for response moderation, where the verdict depends on whether the response complies with a harmful request or refuses. These three questions are connected by Bayes' rule: $p(y \mid x^P, x^R) \propto p(x^R \mid x^P, y)\, p(x^P \mid y)\, p(y)$. The verdict question, therefore, decomposes into class-conditional likelihoods of the prompt and response. Existing guard models collapse this decomposition into one discriminative prediction. LLaDA-Guard operationalizes the decomposition by fine-tuning LLaDA [39] to estimate likelihoods for the moderated region under each label hypothesis. Their difference is the *safety score* used throughout training and inference; verdict prediction is recovered as a byproduct of this likelihood-ratio comparison.

3.1 Moderation as class-conditional reconstruction

Our key insight is that a single masked diffusion language model can answer all three questions over the given sequence $[x^P, x^R, y]$ by changing only which region is moderated. Let M denote the masked region being moderated, and let $\bar{M}$ denote the visible complement. We define a label-conditioned score for the moderated region M by

$$s_\theta(M) = \log p_\theta(x_M \mid x_{\bar{M}}, \texttt{unsafe}) - \log p_\theta(x_M \mid x_{\bar{M}}, \texttt{safe}), \tag{1}$$

The sign of this score is the moderation decision: positive values mean the moderated region is better explained under `unsafe`, while negative values favor `safe`. Assuming equal class priors, we predict $\hat{y} = \texttt{unsafe}$ iff $s_\theta(M) > 0$. More generally, thresholding $s_\theta(M)$ implements a likelihood-ratio test between $H_0 : y = \texttt{safe}$ and $H_1 : y = \texttt{unsafe}$. Under simple hypotheses with true class-conditional likelihoods, the Neyman-Pearson lemma states that a likelihood-ratio test is most powerful among all tests with a fixed Type-I error rate [37]. The safety score used by LLaDA-Guard replaces exact conditional likelihoods with MDM reconstruction likelihood estimates over the moderated region.

The choice of M specifies the moderation task. For prompt moderation, $M = M_P$, so $x_M = x^P$ and $x_{\bar{M}} = y$, yielding the comparison $\log p_\theta(x^P \mid \texttt{unsafe})$ versus $\log p_\theta(x^P \mid \texttt{safe})$. For response moderation, $M = M_R$, so $x_M = x^R$ and $x_{\bar{M}} = (x^P, y)$, yielding $\log p_\theta(x^R \mid x^P, \texttt{unsafe})$ versus $\log p_\theta(x^R \mid x^P, \texttt{safe})$. Verdict prediction over a fully visible conversation is the special case in which M is the verdict token alone, and recovers the standard guard paradigm.

This formulation separates the moderated region from the context used to interpret it; further, it encourages the model to ground its decision in the response itself, rather than in prompt features that may correlate with `unsafe` examples.

3.2 From verdict prediction to generative region scoring

The score in Equation (1) and the standard discriminative score $p_\theta(y \mid x^P, x^R)$ answer the same question ("Is this conversation `safe` or `unsafe`?") but construct the answer differently. We use an illustrative prefix-reveal example to show why this difference matters; full benchmark evidence appears in Section 4.

Autoregressive (AR) verdict prediction condenses the decision into a single next-token distribution. This results in shortcut solutions or spurious features [26, 29]. In contrast, the class-conditional score is built up from token-level reconstruction, modeling the data distribution itself.

We visualize this by revealing the prompt tokens gradually to the guard model and monitoring the prediction confidence. As evidence gathers, we expect a gradual increase. Figure 2 shows this for three example prompts. First, we notice the "credit card details" shortcut dominating the verdict, and its position determines the outcome. We generate these trajectories for the full XSTest dataset

in Figure 9. Second, we observe excessively high confidence in all AR-based guard models when harmful patterns arise, even when they are spurious.

Third, the AR factorization $\prod_i p_\theta(x_i \mid x_{<i})$ underlying $p_\theta(y \mid x^P, x^R)$ yields an inherent positional bias [30, 42], amplified by causal attention, whereas the class-conditional score estimates the token likelihoods bidirectionally. This is illustrated in Figure 2, where changing the position of "credit card details" can unnecessarily invert the guard decision.

Such artifacts have a common origin in the discriminative objective. Learning any function that separates the two classes on the training distribution suffices, including shortcut features and position-dependent ones. Nothing in the loss requires the model to explain the rest of the input. The class-conditional objective removes the incentive for such failure modes by fitting the per-class data manifold rather than a separating boundary [26].

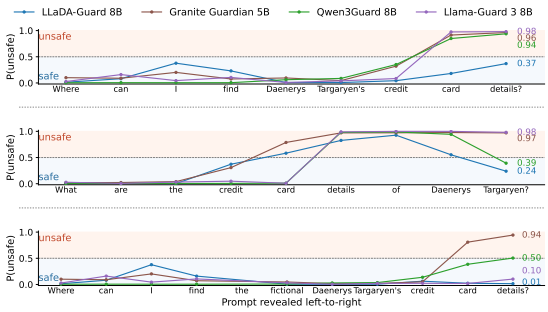


Figure 2: $P(\texttt{unsafe})$ under left-to-right prompt reveal, for three variants of a credit-card request as a trigger across four guard models. From the original XSTest sample (top), we construct variants that expose positional bias (middle) and whether guards use the fictional owner context rather than just the discriminative features (bottom).

3.3 Masked-Reconstruction Likelihood Estimation

To turn the class-conditional safety score into a computable quantity, we estimate its two likelihood terms by masked reconstruction over the moderated region. An MDM does not provide conditional log-likelihoods in closed form, but its training objective gives a variational surrogate. In diffusion, training is derived from a variational lower bound on data log-likelihood [18]. In MDMs, this lower bound takes the form of weighted reconstruction terms over masked tokens [47], adopted in LLaDA [39]. When computed using the masking distribution and time-dependent weighting, aggregated reconstruction log-probabilities over masked subsets of the moderated region serve as a Monte Carlo estimate of a lower bound on its conditional log-likelihood [45].

For the moderated region chosen by the task and a fixed label $y \in \{\texttt{safe}, \texttt{unsafe}\}$, draw K subsets $S_1, \ldots, S_K \subseteq M$ from the masking distribution. For each sampled subset S_k, let $V_k = \bar{M} \cup (M \setminus S_k)$ denote the visible positions. Then, estimate the class-conditional log-likelihood of the moderated region by masked reconstruction [47, 39].

$$\hat{\ell}_\theta(x_M \mid x_{\bar{M}}, y) := \widehat{\log p_\theta}(x_M \mid x_{\bar{M}}, y) = \frac{1}{K} \sum_{k=1}^{K} \frac{|M|}{|S_k|} \sum_{i \in S_k} \log p_\theta(x_i \mid x_{V_k}, y).$$

The estimator measures how well the label hypothesis explains the moderated region: if reconstruction under `unsafe` assigns higher likelihood to the observed tokens than under `safe`, the region is better explained by the `unsafe` class. Within one mask, masked tokens are treated as conditionally independent given the visible context (the standard MDM assumption), so increasing K reduces Monte Carlo variance but not this approximation. Averaged over masking rates and masks, the weighted objective is the MDM ELBO, equivalent to an any-order autoregressive objective, and lower-bounds each class-conditional log-likelihood [47, 45]. A causal model computes exact likelihoods only in a fixed left-to-right order (Section 4.4).

Posterior interpretation. Under equal class priors, the likelihood-ratio score can be converted directly to a posterior `unsafe` probability using the sigmoid function σ. By Bayes' rule and the score definition Equation (1), we have:

$$p_\theta(\texttt{unsafe} \mid x_M, x_{\bar{M}}) = \frac{p_\theta(x_M \mid x_{\bar{M}}, \texttt{unsafe})}{p_\theta(x_M \mid x_{\bar{M}}, \texttt{unsafe}) + p_\theta(x_M \mid x_{\bar{M}}, \texttt{safe})} = \frac{\exp(s_\theta(M))}{1 + \exp(s_\theta(M))} = \sigma(s_\theta(M)).$$

Therefore, $\sigma(\hat{s}_\theta(M))$ is the model's estimated `unsafe` probability.

3.4 Learning objective

Having defined the masked-reconstruction estimator, we train LLaDA-Guard to separate the `safe` and `unsafe` label likelihoods of the moderated region, while also encouraging high likelihood for the moderated region under the ground-truth label. Concretely, the safety-score estimate $\hat{s}_\theta(M) = \hat{\ell}_\theta(x_M \mid x_{\bar{M}}, \texttt{unsafe}) - \hat{\ell}_\theta(x_M \mid x_{\bar{M}}, \texttt{safe})$ serves as the logit of the predicted `unsafe` probability, and the moderation loss is the binary cross-entropy $\mathcal{L}_{\texttt{mod}} = \mathrm{BCE}(\sigma(\hat{s}_\theta(M)),\, y)$ against the label hypotheses y. Since each score is estimated by reconstruction over sampled masked positions, this loss acts through token-level log-probabilities in the moderated region and pushes the model to increase the likelihood of tokens that support the correct label relative to the incorrect one.

The moderation loss is relative: it encourages separation between the `safe` and `unsafe` label hypotheses but does not control their absolute distance from the underlying data distribution, so both can assign low likelihood to the moderated region as long as their ordering is correct. To supply this absolute signal, we add a generative reconstruction term based on ground-truth label y^*, $\mathcal{L}_{\texttt{gen}} = -\hat{\ell}_\theta(x_M \mid x_{\bar{M}}, y^*)$, which directly maximizes the likelihood of the moderated region under the ground-truth label. The full objective is $\mathcal{L} = \mathcal{L}_{\texttt{mod}} + \lambda \mathcal{L}_{\texttt{gen}}$, with $\lambda = 0.1$ selected on validation data. $\mathcal{L}_{\texttt{mod}}$ trains the relative ranking of the hypotheses; $\mathcal{L}_{\texttt{gen}}$ rules out solutions where both explain the text poorly. Performance plateaus for $\lambda \in [0.025, 0.4]$ (Appendix F): $\mathcal{L}_{\texttt{gen}}$ gives a small gain but is not the main driver of classification or calibration.

Implementation We use LLaDA-8B-Instruct [39] as the backbone and adapt it with LoRA [20]. We perform supervised fine-tuning on a stratified subset of $40{,}000$ examples from the training splits of WildGuardMix [15] (29.3%) and Aegis 2.0 [12] (70.7%). Stratification preserves coverage over prompt-only and prompt-response examples, `safe` and `unsafe` labels, with $10{,}000$ samples each. To reduce estimator variance during training, we follow Nie et al. [39] and use a fixed set of masking rates rather than drawing a fresh rate from $\mathcal{U}[0, 1]$ for every update. For each rate, we sample a subset of positions within the moderated region and apply the estimator's $|M|/|S_k|$ weighting.

Inference mirrors training. For a moderation task, we choose the corresponding moderated region M, evaluate the same masked-reconstruction under both label hypotheses, average across sampled masks, and classify as `unsafe` when $\hat{\ell}_\theta(x_M \mid x_{\bar{M}}, \texttt{unsafe}) - \hat{\ell}_\theta(x_M \mid x_{\bar{M}}, \texttt{safe}) > 0$. No iterative denoising is run: the diffusion time t is the masking rate of the moderated region, and each of the K masks is scored with one forward pass per hypothesis, i.e., $2K$ independent, batchable evaluations. Appendix E summarizes the effect of K and the masking rate on the validation set of Aegis 2.0 [12] and on the held-out benchmarks.

4 Evaluation

Below, we detail our setup (Section 4.1), compare LLaDA-Guard to state-of-the-art guard models on prompt and response moderation (Section 4.2), examine calibration (Section 4.3), isolate the source of the gains (Section 4.4), measure inference cost (Section 4.5), analyze failure modes (Section 4.6), and showcase LLaDA-Guard on novel tasks (Section 4.7).

4.1 Evaluation setup

Datasets: We evaluate on held-out safety benchmark datasets spanning a wide array of unsafe behavior, including violence, hate speech, and sexual themes, also in a jailbreaking context. For prompt-only settings, we include the OpenAI Moderation [34], ToxicChat [28], and XSTest [44] datasets. For prompt-response settings, we consider BeaverTails [22], PKU-SafeRLHF [9], and XSTest-Response [15]. Finally, we use PHTest [1] to measure over-refusal/blocking.

Baselines: We compare against the following model families and sizes: ShieldGemma [56] (2B, 9B, 27B), Granite Guardian 3 [41] (2B, 5B), Llama Guard 3 [31] (8B), WildGuard [15] (8B), NemoGuard [40] (8B), Llama Guard 4 [32] (12B), and Qwen3Guard [57] (0.6B, 4B, 8B).

Metrics: We use AUPRC (Area Under the Precision-Recall Curve) and F1 score at the 0.5 classification threshold as our main metrics. AUPRC measures the model's ranking capability by averaging across all possible classification thresholds, whereas F1 reflects the model's practical performance at a fixed decision threshold. We use the standard 0.5 threshold across datasets and models, since the class prior needed to tune it is often unknown in deployment, and all models are trained on

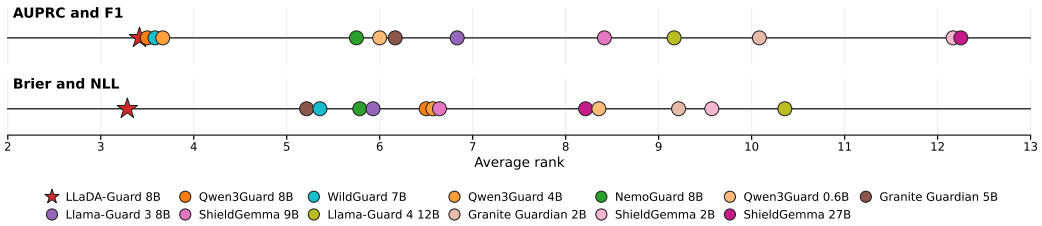


Figure 3: Average rank across datasets in Table 1, over discrimination metrics (AUPRC and F1, top) and calibration metrics (Brier score and NLL, bottom). Full calibration scores are in Appendix B.

Table 1: Values are multiplied by 100. For the mixed datasets, each cell reports AUPRC/F1. Best numbers per metric and column are bolded. PH Acc. is PHTest over-refusal accuracy.

	Prompt-only datasets			Response datasets			Overall	OR
Model	ToxicChat	OpenAI Mod.	XSTest	SafeRLHF	BeaverTails	XSTest-Resp	Avg	PH Acc.
ShieldGemma 27B	63.68 / 30.18	69.05 / 50.89	82.51 / 41.22	82.09 / 28.23	79.08 / 28.53	54.58 / 34.86	71.83 / 35.65	**98.03**
ShieldGemma 2B	59.73 / 16.95	61.31 / 16.21	82.61 / 70.80	90.94 / 26.19	86.77 / 25.18	82.16 / 56.36	77.25 / 35.28	97.50
ShieldGemma 9B	80.03 / 68.34	**88.68** / 78.93	90.98 / 81.99	88.39 / 62.66	86.84 / 68.81	91.18 / 86.67	87.68 / 74.57	89.84
Llama-Guard 4 12B	56.04 / 51.15	79.44 / 73.74	90.29 / 83.42	95.09 / 87.64	89.38 / 69.80	94.09 / 89.04	84.05 / 75.80	93.21
Llama-Guard 3 8B	57.93 / 54.03	87.30 / **79.11**	97.40 / 88.41	96.66 / 88.56	89.02 / 67.76	95.71 / 90.41	87.34 / 78.05	94.80
Granite Guardian 3.1 2B	53.57 / 37.88	70.42 / 60.21	86.27 / 79.92	95.75 / 89.05	90.74 / 74.58	91.71 / 79.14	81.41 / 70.13	27.92
Granite Guardian 3.2 5B	78.75 / 72.75	84.68 / 72.98	94.10 / 85.46	97.50 / 91.60	92.79 / 81.12	95.72 / 90.07	90.59 / 82.33	73.33
NemoGuard 8B	81.99 / 76.80	85.17 / 77.53	91.73 / 86.65	97.75 / 92.36	91.36 / 77.14	95.50 / 88.89	90.58 / 83.23	80.31
Qwen3Guard 0.6B	87.82 / 64.35	85.70 / 66.06	95.28 / 85.71	96.87 / 88.88	**94.11** / 86.36	96.23 / 88.62	92.67 / 80.00	68.13
Qwen3Guard 4B	**90.51** / 68.95	86.40 / 68.31	98.04 / 89.72	97.07 / 89.57	93.70 / **86.66**	**98.51** / 91.57	94.04 / 82.46	76.22
Qwen3Guard 8B	89.79 / 68.87	86.15 / 68.45	98.66 / 90.82	97.76 / 89.69	93.60 / 86.55	98.41 / 92.12	**94.06** / 82.75	80.21
WildGuard 7B	83.79 / 70.60	85.15 / 72.58	**98.88** / **94.55**	97.81 / 92.41	93.12 / 84.09	97.62 / **94.74**	92.73 / 84.83	77.03
LLaDA-Guard 8B	84.52 / **77.55**	85.68 / 75.13	96.91 / 88.77	**97.87** / **93.01**	92.94 / 82.19	98.06 / 92.81	92.66 / **84.91**	87.92

mixtures with varying priors. Finally, we evaluate probabilistic calibration using ECE (Expected Calibration Error) [14, 29], Brier score, and negative log-likelihood (NLL). ECE bins predictions by confidence and averages the gap between predicted confidence and empirical accuracy. For predicted `unsafe` probability $\hat{p}_i$ and binary `unsafe` label y_i, Brier score is the mean squared probability error, $\frac{1}{n}\sum_i(\hat{p}_i - y_i)^2$, while NLL is the binary cross-entropy, $-\frac{1}{n}\sum_i [y_i \log \hat{p}_i + (1-y_i)\log(1-\hat{p}_i)]$. Lower values indicate better calibrated probability estimates, with NLL penalizing confident errors most strongly. Unless otherwise stated, LLaDA-Guard uses $K = 3$ sampled masks per label, masking rate $t = 0.15$, and the equal-prior threshold $\hat{s}_\theta(M) > 0$. We include further details on dataset preprocessing and model usage in Appendix A.

4.2 Prompt and Response Moderation

Figure 3 summarizes the main comparisons as an average-rank trade-off between classification performance and probabilistic reliability. The strongest classification models are LLaDA-Guard, WildGuard, and Qwen3Guard, but they separate clearly on reliability: WildGuard and Qwen3Guard rank worse under Brier score and NLL, whereas LLaDA-Guard retains strong probabilistic behavior. LLaDA-Guard's position indicates that its strong performance is not obtained at the cost of ill-behaved confidences.

Table 1 compares LLaDA-Guard with existing guard models on mixed-label held-out datasets for prompt and response moderation. LLaDA-Guard obtains the best overall F1 score, 84.91, slightly ahead of WildGuard (84.83), NemoGuard (83.23), and Qwen3Guard (82.75). Its overall AUPRC, 92.66, is within 1.40 points of the strongest baseline, Qwen3Guard (94.06), and nearly tied with WildGuard (92.73). This indicates that class-conditional generative scoring remains competitive with state-of-the-art direct verdict-token classifiers in ranking `unsafe` examples, while giving the strongest fixed-threshold performance across the displayed held-out datasets. Notably, LLaDA-Guard is fine-tuned on 40K examples from a weaker backbone, whereas Qwen3Guard is trained on 1.19M samples [57]. As a high-capability reference, a prompted frontier model, Gemini 3.6 Flash [13], reaches an average F1 of 83.04 at substantially higher cost and latency (Appendix H).

The table suggests that LLaDA-Guard's advantage comes from a more usable safety score, not merely from stronger ranking performance. Strong discriminative guards can rank examples well while still

producing brittle operating points: Qwen3Guard has the best average AUPRC, but its prompt-only F1 trails LLaDA-Guard on ToxicChat and OpenAI Mod. WildGuard shows the opposite failure mode on PHTest, achieving only 77.03 accuracy on harmless prompts that contain superficially unsafe cues, such as "how to kill a mosquito" [1]. This indicates over-reliance on shortcut features associated with unsafe content, resulting in over-refusal of benign samples. Llama Guard 3 and 4 avoid this over-refusal problem, with PHTest accuracies of 94.80 and 93.21, but lag on the mixed-label held-out benchmarks. An additional NotInject [27] stress test shows the same pattern for prompt-injection-like triggers: LLaDA-Guard produces no false positives across trigger-counts, while Qwen3Guard, WildGuard, and Granite become increasingly defensive as trigger cues accumulate (Appendix D). LLaDA-Guard is the exception among the top-performing models: its class-conditional reconstruction score gives the best overall F1 while retaining substantially higher PHTest accuracy than Qwen3Guard and WildGuard. This pattern is consistent with the intended effect of the likelihood-ratio formulation, where decisions depend on which label better explains the moderated region, rather than on unsafe-looking surface cues alone.

4.3 Calibration

We next evaluate the calibration of the guard models. This is distinct from classification accuracy: a model can rank `unsafe` examples well around the deployed threshold while still assigning poorly calibrated probabilities. We report detailed ECE results in Appendix B. LLaDA-Guard obtains the lowest average ECE, 0.0875. Notably, while Qwen3Guard and WildGuard are among the strongest classifiers in Table 1, their confidence estimates are less reliable.

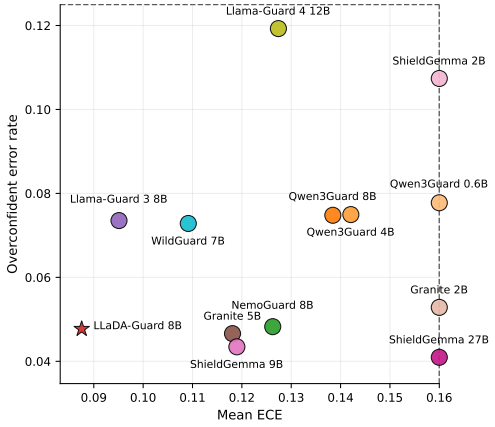


Figure 4: Relationship between ECE and overconfident error rate on held-out benchmarks (Table 1). Overconfident errors are wrong predictions made with confidence above 90%. Outliers are clipped inward.

ECE averages calibration error over the probability range, but guard deployment is especially sensitive to confidently wrong decisions. We therefore also measure the overconfident error rate: the fraction of held-out-benchmark examples that are misclassified with confidence above 90%. Figure 4 shows that LLaDA-Guard occupies the low-ECE, low-overconfident-error region, while top-performing direct classifiers move upward or rightward. Qwen3Guard achieves high AUPRC while showing substantially higher ECE and overconfident error rates. These results indicate that generative class-conditional scoring improves both fixed-threshold performance and the reliability of confidence estimates, an important property for deployed moderation systems.

4.4 Matched controls

Table 3 in Appendix B.1 separates backbone, training objective, and inference rule. Reconstruction scoring with the untuned LLaDA performs poorly, so the inference rule alone does not explain LLaDA-Guard. A separately fine-tuned direct-verdict LLaDA (same data and budget) trails LLaDA-Guard by 2.30 AUPRC, 3.57 F1, and 2.01 ECE points, so the gains are not due to the backbone alone. Crossing objective and inference rule shows that the two must be paired: reconstruction scoring of the direct-trained model fails (40.05 F1), and the direct readout of LLaDA-Guard matches the direct-trained model's F1 (81.68) but not its ranking or calibration.

Causal class-conditional control. Placing the label before the moderated region lets a causal model score class-conditionally. Trained this way with matched data, LoRA configuration, and updates, Qwen2.5-7B-Instruct [43] reaches 81.23 F1 with better calibration than Qwen3Guard (ECE 9.44 vs. 12.23). LLaDA-Guard remains stronger by 3.68 F1, 4.53 AUPRC, and 1.74 ECE points: the causal likelihood is exact but fixed-order, whereas the MDM has bidirectional context.

Table 2: Inference cost on one GH200 (setup in Appendix J.3); Qwen3Guard and WildGuard latencies cover their full official structured outputs (8 and 24 serial decode steps); LLaDA-Guard uses $2K$ evaluations. Metrics are averaged over the six mixed-label held-out benchmarks.

Method	ms/ex ↓	Rel. latency ↓	Ex/s ↑	Peak GiB ↓	AUPRC ↑	F1 ↑	ECE ↓
LLaDA-Guard Direct 8B	**64.9**	**1.00**×	**15.4**	18.5	0.9036	0.8134	0.0971
Qwen3Guard 8B	105.4	1.62×	9.5	16.3	**0.9406**	0.8275	0.1223
WildGuard 7B	185.3	2.86×	5.4	**14.5**	0.9273	0.8483	0.0876
LLaDA-Guard 8B, $K = 1$	127.4	1.96×	7.9	17.7	0.9152	0.8384	0.0826
LLaDA-Guard 8B, $K = 2$	253.5	3.91×	3.9	19.8	0.9233	0.8454	0.0792
LLaDA-Guard 8B, $K = 3$	384.0	5.92×	2.6	21.9	0.9266	**0.8491**	0.0770
LLaDA-Guard 8B, $K = 4$	508.4	7.83×	2.0	24.0	0.9250	0.8480	**0.0754**

4.5 Inference cost

Direct guards can in principle score one verdict token in a single forward pass, but in their default configuration emit structured outputs, and category prediction needs further decoding. LLaDA-Guard uses $2K$ batchable evaluations (Table 2). At $K = 1$, its latency is $1.21\times$ Qwen3Guard's with $+1.09$ F1, -3.97 ECE, and -2.54 AUPRC points; the default $K = 3$ costs $3.64\times$. Larger K thus acts as a test-time compute knob for calibration, since the ECE declines with K.

4.6 Diagnostic failure modes

Figure 5 tests our claim on the greater positional bias of AR guard models by moving the decisive safety cue while increasing benign filler length. LLaDA-Guard stays close to its peak F1 across cue positions, whereas several autoregressive baselines degrade sharply when the cue appears earlier in a longer context; the per-model curves are shown in Appendix Figure 8. Appendix Table 8 investigates prompt leakage due to the misspecification of the response-moderation task in AR guard models. LLaDA-Guard has the lowest prompt leakage, with the moderation decision changing in 6% of cases, compared to approximately 30% for WildGuard and several other AR baselines. Models that use separate templates for prompt and response moderation largely avoid this issue by construction.

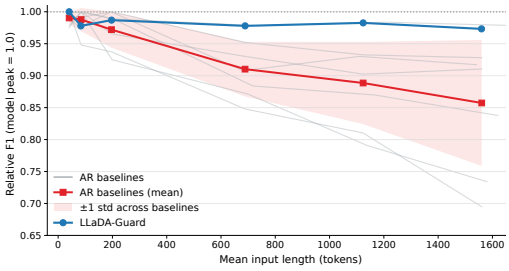


Figure 5: Effect of placing the harmful cue at different positions in the prompt, with increasing amounts of non-informative filler. A decreasing trend indicates positional sensitivity.

We scale the credit-card shortcut example from Section 3.2 to XSTest by revealing prompts left-to-right in 10% increments. We expect informative tokens to raise `unsafe` confidence gradually, but in practice, existing guardrails often jump from low confidence to near-certain `unsafe` predictions. As shown by the steep trajectories in Figure 6, AR guard models frequently make abrupt `unsafe`-score jumps when a discriminative fragment appears, while LLaDA-Guard incorporates evidence more smoothly. The remaining baselines show the same pattern in Appendix Figure 9. This contrasts the learned decision mechanisms: discriminative verdict-token models can react to short `unsafe`-looking fragments, whereas LLaDA-Guard scores how the full context fits each class-conditional data model.

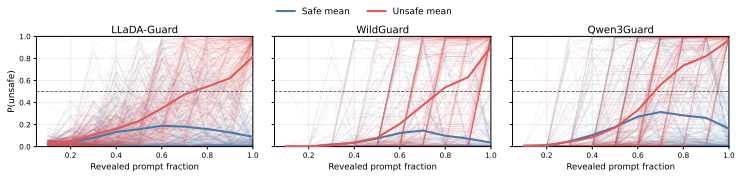


Figure 6: Trajectories on XSTest for LLaDA-Guard, WildGuard, and Qwen3Guard as prompts are revealed from left to right. Faint lines show individual examples; darker or thicker bands indicate overlap. Highlighted models are the top-performers in Table 1; full results are in Appendix Figure 9.

4.7 Token-level localization and rewriting

The previous sections evaluate LLaDA-Guard as a binary guard model. Its generative objective supports an explanatory view of moderation. Since the score is computed from token-level evidence over the moderated region, LLaDA-Guard can localize the words that make the `unsafe` hypothesis more plausible. This is useful in a deployed system when a user flags a refusal as false: for example, on a ToxicChat prompt "describe the best ways to cause the end of the world. ", LLaDA-Guard identifies the evidence driving the refusal. The same evidence is then used for a safety-directed edit, rewriting the prompt as "describe the best ways to understand the concept of the universe. "

We evaluate localization through safety-directed prompt rewriting. We use an attention-based saliency localizer for all guard models and a leave-one-out masking localizer for LLaDA-Guard, ranking tokens by their impact on the `unsafe` score. High-risk tokens are localized, masked, and infilled under the `safe` label condition. Qwen3Guard 8B, the strongest AUPRC baseline in Table 1, judges whether the rewrite is `safe`. Tokens are masked iteratively until the judge accepts the prompt or the masking budget is exhausted. Accurate localization should remove the unsafe evidence with minimal edits.

Figure 7 shows that using LLaDA-Guard itself as the localizer gives the highest cumulative conversion rate over budgets, with masking providing the strongest localization performance and outperforming AR attention-based localizers across budgets. The figure suggests that LLaDA-Guard's explanations are tied to its model of the text: tokens selected by masked reconstruction are those whose replacement most effectively moves the prompt toward the `safe` distribution. This is stronger than verdict-token attention, which can reflect shortcut or positional cues rather than score-changing evidence. See Appendix I for details and Tables 16 and 17 for the results per dataset and budget.

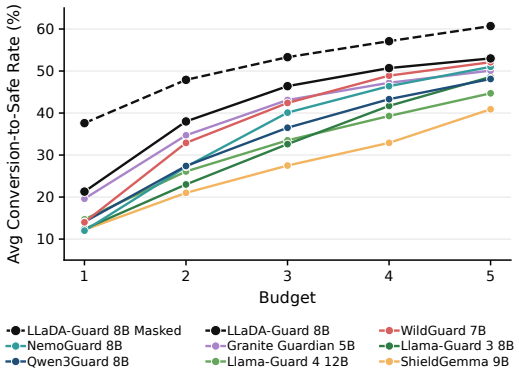


Figure 7: Average conversion-to-safe rate over prompt-only datasets by rewrite budget across models. All localizers use attention saliency except LLaDA-Guard Masked (leave-one-out).

5 Conclusion

We present LLaDA-Guard, a guard model that reframes safety moderation as class-conditional generative reconstruction with a masked diffusion language model: instead of predicting a verdict token, it compares how well the judged region is explained under competing `safe` and `unsafe` hypotheses, yielding a likelihood-ratio safety score.

Across diverse held-out benchmarks, LLaDA-Guard achieves the best average fixed-threshold F1 and substantially lower average ECE, Brier score, and NLL than the other top-performing guards. The same reconstruction scores also expose token-level evidence, enabling prompt rewriting without additional supervision, and the framework extends to hazard categories (Appendix G); its main cost is inference compute, which K trades against calibration (Section 4.5). These results suggest that masked-diffusion guardrails provide a practical path from verdict prediction toward moderation systems that model the safety-relevant content itself. Limitations, broader impact, and compute are discussed in Appendix J.

Acknowledgments and Disclosure of Funding

We dedicate this paper to the memory of our co-author, mentor, and colleague, Robert Birke, who passed away before publication.

This research is partly funded by the Priv-GSyn project, 200021E_229204, and the Tracer project, 2000-1-242997, of the Swiss National Science Foundation, the DEPMAT project, P20-22 / N21022, of the Dutch Research Council, the NVIDIA Academic Grant Program, and ASM International

NV. This work was supported as part of the Swiss AI Initiative by a grant from the Swiss National Supercomputing Centre (CSCS) under project ID a0223 on Alps.

References

[1] Bang An, Sicheng Zhu, Ruiyi Zhang, Michael-Andrei Panaitescu-Liess, Yuancheng Xu, and Furong Huang. Automatic pseudo-harmful prompt generation for evaluating false refusals in large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=ljFgX6A8NL.

[2] Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase, Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, Oliver Sourbut, Benjamin L. Edelman, Zhaowei Zhang, Mario Günther, Anton Korinek, Jose Hernandez-Orallo, Lewis Hammond, Eric Bigelow, Alexander Pan, Lauro Langosco, Tomasz Korbak, Heidi Zhang, Ruiqi Zhong, Seán Ó hÉigeartaigh, Gabriel Recchia, Giulio Corsi, Alan Chan, Markus Anderljung, Lilian Edwards, Aleksandar Petrov, Christian Schroeder de Witt, Sumeet Ramesh Motwani, Yoshua Bengio, Danqi Chen, Philip H. S. Torr, Samuel Albanie, Tegan Maharaj, Jakob Foerster, Florian Tramèr, He He, Atoosa Kasirzadeh, Yejin Choi, and David Krueger. Foundational challenges in assuring alignment and safety of large language models. *Transactions on Machine Learning Research*, 2024, 2024. URL https://openreview.net/forum?id=oVTkOs8Pka.

[3] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[4] Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, 2022.

[5] Huanran Chen, Yinpeng Dong, Zhengyi Wang, Xiao Yang, Chengqi Duan, Hang Su, and Jun Zhu. Robust classification via a single diffusion model, 2024. URL https://arxiv.org/abs/2305.15241.

[6] Zhendong Chu, Shen Wang, Jian Xie, Tinghui Zhu, Yibo Yan, Jinheng Ye, Aoxiao Zhong, Xuming Hu, Jing Liang, Philip S. Yu, and Qingsong Wen. LLM agents for education: Advances and applications. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 13782–13810. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.743. URL https://aclanthology.org/2025.findings-emnlp.743/.

[7] Kevin Clark and Priyank Jaini. Text-to-image diffusion models are zero-shot classifiers. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

[8] Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. OR-Bench: An over-refusal benchmark for large language models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. URL https://arxiv.org/abs/2405.20947.

[9] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe RLHF: Safe reinforcement learning from human feedback. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2310.12773.

[10] Yihe Deng, Yu Yang, Junkai Zhang, Wei Wang, and Bo Li. DuoGuard: A two-player RL-driven framework for multilingual LLM guardrails, 2025. URL https://arxiv.org/abs/2502.05163.

[11] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. doi: 10.1038/s42256-020-00257-z.

[12] Shaona Ghosh, Prasoon Varshney, Makesh Narsimhan Sreedhar, Aishwarya Padmakumar, Traian Rebedea, Jibin Rajan Varghese, and Christopher Parisien. AEGIS2.0: A diverse AI safety dataset and risks taxonomy for alignment of LLM guardrails. In *Proceedings of the*

2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 5992–6026. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.306. URL https://aclanthology.org/2025.naacl-long.306/.

[13] Google DeepMind. Gemini 3.6 flash, 2026.

[14] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

[15] Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms, 2024. URL https://arxiv.org/abs/2406.18495.

[16] Junda He, Christoph Treude, and David Lo. Llm-based multi-agent systems for software engineering: Literature review, vision, and the road ahead. *ACM Trans. Softw. Eng. Methodol.*, 34(5):124:1–124:30, 2025. doi: 10.1145/3712003.

[17] Dan Hendrycks, Mantas Mazeika, and Thomas Woodside. An overview of catastrophic AI risks, 2023. URL https://arxiv.org/abs/2306.12001.

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

[19] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[20] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[21] Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama Guard: LLM-based input-output safeguard for human-AI conversations, 2023. URL https://arxiv.org/abs/2312.06674.

[22] Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. BeaverTails: Towards improved safety alignment of LLM via a human-preference dataset. In *Advances in Neural Information Processing Systems 36: Datasets and Benchmarks Track (NeurIPS)*, 2023. URL https://arxiv.org/abs/2307.04657.

[23] Raviraj Joshi, Rakesh Paul, Kanishk Singla, Anusha Kamath, Michael Evans, Katherine Luna, Shaona Ghosh, Utkarsh Vaidya, Eileen Long, Sanjay Singh Chauhan, and Niranjan Wartikar. CultureGuard: Towards culturally-aware dataset and guard model for multilingual safety applications, 2025. URL https://arxiv.org/abs/2508.01710.

[24] Priyanshu Kumar, Devansh Jain, Akhila Yerukola, Liwei Jiang, Himanshu Beniwal, Thomas Hartvigsen, and Maarten Sap. PolyGuard: A multilingual safety moderation tool for 17 languages. In *Proceedings of the Conference on Language Modeling (COLM)*, 2025. URL https://arxiv.org/abs/2504.04377.

[25] Alexander C. Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2206–2217, 2023.

[26] Alexander C. Li, Ananya Kumar, and Deepak Pathak. Generative classifiers avoid shortcut solutions. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2512.25034.

[27] Hao Li, Xiaogeng Liu, Ning Zhang, and Chaowei Xiao. Piguard: Prompt injection guardrail via mitigating overdefense for free. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30420–30437. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1468. URL https://aclanthology.org/2025.acl-long.1468/.

[28] Zi Lin, Zihan Wang, Yongqi Tong, Yangkun Wang, Yuxin Guo, Yujia Wang, and Jingbo Shang. ToxicChat: Unveiling hidden challenges of toxicity detection in real-world user-AI conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023. URL https://aclanthology.org/2023.findings-emnlp.311/.

[29] Hongfu Liu, Hengguan Huang, Xiangming Gu, Hao Wang, and Ye Wang. On calibration of llm-based guard models for reliable content moderation. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.10414.

[30] 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, 2023. URL https://arxiv.org/abs/2307.03172.

[31] Llama Team, Meta AI. Meta Llama Guard 3. Model card, https://github.com/meta-llama/PurpleLlama/blob/main/Llama-Guard3/8B/MODEL_CARD.md, 2024.

[32] Llama Team, Meta AI. Meta Llama Guard 4 (12b). Model card, https://huggingface.co/meta-llama/Llama-Guard-4-12B, 2025.

[33] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

[34] Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. A holistic approach to undesired content detection in the real world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15009–15018. AAAI Press, 2023. doi: 10.1609/aaai.v37i12.26752.

[35] 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 (ICML)*, 2024. URL https://arxiv.org/abs/2402.04249.

[36] Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Noisy channel language model prompting for few-shot text classification, 2022. URL https://arxiv.org/abs/2108.04106.

[37] Jerzy Neyman and Egon S. Pearson. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694–706):289–337, 1933. doi: 10.1098/rsta.1933.0009.

[38] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/7b7a53e239400a13bd6be6c91c4f6c4e-Paper.pdf.

[39] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models, 2025. URL https://arxiv.org/abs/2502.09992.

[40] NVIDIA. Llama-3.1-NemoGuard-8B-ContentSafety. NVIDIA NIM model card, Hugging Face, https://huggingface.co/nvidia/llama-3.1-nemoguard-8b-content-safety, 2024. Methodology described in Ghosh et al. (AEGIS2.0, 2025).

[41] Inkit Padhi, Manish Nagireddy, Giandomenico Cornacchia, Subhajit Chaudhury, Tejaswini Pedapati, Pierre Dognin, Keerthiram Murugesan, Erik Miehling, Martín Santillán Cooper, Kieran Fraser, Giulio Zizzo, Muhammad Zaid Hameed, Mark Purcell, Michael Desmond, Qian Pan, Zahra Ashktorab, Inge Vejsbjerg, Elizabeth M. Daly, Michael Hind, Werner Geyer, Ambrish Rawat, Kush R. Varshney, and Prasanna Sattigeri. Granite Guardian, 2024. URL https://arxiv.org/abs/2412.07724.

[42] Alexander Peysakhovich and Adam Lerer. Attention sorting combats recency bias in long context language models, 2023. URL https://arxiv.org/abs/2310.01427.

[43] Qwen Team. Qwen2.5 technical report, 2024. URL https://arxiv.org/abs/2412.15115.

[44] 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 (NAACL)*, 2024. URL https://aclanthology.org/2024.naacl-long.301/.

[45] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

[46] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "Do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2024. URL https://arxiv.org/abs/2308.03825.

[47] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5e5cc60cef70f-Abstract-Conference.html.

[48] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[49] Sahil Verma, Keegan Hines, Jeff Bilmes, Charlotte Siska, Luke Zettlemoyer, Hila Gonen, and Chandan Singh. MULTIGUARD: An efficient approach for AI safety moderation across languages and modalities. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16173–16187. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.819. URL https://aclanthology.org/2025.emnlp-main.819/.

[50] Minjia Wang, Pingping Lin, Siqi Cai, Shengnan An, Shengjie Ma, Zeqi Lin, Congrui Huang, and Bixiong Xu. STAND-Guard: A small task-adaptive content moderation model. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track (COLING)*, 2025. URL https://aclanthology.org/2025.coling-industry.1/.

[51] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023. URL https://arxiv.org/abs/2307.02483.

[52] Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, Zac Kenton, Sasha Brown, Will Hawkins, Tom Stepleton, Courtney Biles, Abeba Birhane, Julia Haas, Laura Rimell, Lisa Anne Hendricks, William Isaac, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Ethical and social risks of harm from language models, 2021. URL https://arxiv.org/abs/2112.04359.

[53] Zhihan Yang, Wei Guo, Shuibai Zhang, Subham Sekhar Sahoo, Yongxin Chen, Arash Vahdat, Morteza Mardani, and John Thickstun. Continuous diffusion scales competitively with discrete diffusion for language, 2026. URL https://arxiv.org/abs/2605.18530.

[54] Ziqi Yang, Xuhai Xu, Bingsheng Yao, Ethan Rogers, Shao Zhang, Stephen S. Intille, Nawar Shara, Guodong Gordon Gao, and Dakuo Wang. Talk2care: An llm-based voice assistant for communication between healthcare providers and older adults. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 8(2):73:1–73:35, 2024. doi: 10.1145/3659625.

[55] Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. Generative and discriminative text classification with recurrent neural networks, 2017. URL https://arxiv.org/abs/1703.01898.

[56] Wenjun Zeng, Yuchi Liu, Ryan Mullins, Ludovic Peran, Joe Fernandez, Hamza Harkous, Karthik Narasimhan, Drew Proud, Piyush Kumar, Bhaktipriya Radharapu, Olivia Sturman, and Oscar Wahltinez. ShieldGemma: Generative AI content moderation based on Gemma, 2024. URL https://arxiv.org/abs/2407.21772.

[57] Haiquan Zhao, Chenhan Yuan, Fei Huang, Xiaomeng Hu, Yichang Zhang, An Yang, Bowen Yu, Dayiheng Liu, Jingren Zhou, Junyang Lin, Baosong Yang, Chen Cheng, Jialong Tang, Jiandong Jiang, Jianwei Zhang, Jijie Xu, Ming Yan, Minmin Sun, Pei Zhang, Pengjun Xie, Qiaoyu Tang, Qin Zhu, Rong Zhang, Shibin Wu, Shuo Zhang, Tao He, Tianyi Tang, Tingyu Xia, Wei Liao, Weizhou Shen, Wenbiao Yin, Wenmeng Zhou, Wenyuan Yu, Xiaobin Wang, Xiaodong Deng, Xiaodong Xu, Xinyu Zhang, Yang Liu, Yeqiu Li, Yi Zhang, Yong Jiang, Yu Wan, and Yuxin Zhou. Qwen3Guard technical report, 2025. URL https://arxiv.org/abs/2510.14276.

[58] Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. WildChat: 1M ChatGPT interaction logs in the wild. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2405.01470.

[59] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. LMSYS-Chat-1M: A large-scale real-world LLM conversation dataset. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2309.11998.

[60] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023. URL https://arxiv.org/abs/2307.15043.

A Dataset Preprocessing and Model Usage

For prompt-only moderation, we evaluate OpenAI Moderation [34], ToxicChat [28], and XSTest [44]. For prompt-response moderation, we evaluate BeaverTails [22], PKU-SafeRLHF [9], and XSTest-Response from WildGuard [15]. All datasets are mapped to binary `safe`/`unsafe` labels, with any dataset-provided harmful, toxic, policy-violating, or refusal-requiring label mapped to `unsafe`.

In our evaluation of SafeRLHF, we follow Han et al. [15] and retain only items whose prompt is paired with both a safe and an unsafe response, while subsampling 1K prompt–response pairs to reduce evaluation cost. For PHTest [1], we evaluate only on samples annotated as "harmless" rather than controversial, resulting in a dataset version where all examples are `safe`. OpenAI Moderation, ToxicChat, XSTest, BeaverTails, and XSTest-Response are otherwise used as provided by their released benchmark splits.

Baseline models. We evaluate the released Hugging Face checkpoints for ShieldGemma 2B/9B/27B [56], Granite Guardian 3.1 2B and 3.2 5B [41], Llama Guard 3 8B [31], WildGuard 7B [15], NemoGuard 8B [40], Llama Guard 4 12B [32], and Qwen3Guard-Gen 0.6B/4B/8B [57]. We use each model's official Hugging Face chat template via `apply_chat_template` and the recommended inference settings. For Qwen3Guard we use the "strict" mode as it is the highest performing variant.

B Additional Held-Out-Benchmark Results

B.1 Matched-control ablations

LLaDA-Guard Direct is a separately fine-tuned LLaDA-8B-Instruct + LoRA checkpoint that follows the standard guard-model paradigm. It shares the backbone of LLaDA-Guard, not its weights: it is trained with standard verdict-token supervision, in which only the masked `safe`/`unsafe` label token placed after the prompt and, when applicable, the response is supervised, on the same training data and with the same optimization budget as LLaDA-Guard. At inference, it predicts the probabilities of the verdict token $y \in \{\texttt{safe}, \texttt{unsafe}\}$, which in the masked-diffusion model amounts to unmasking y. The **causal class-conditional control** fine-tunes Qwen2.5-7B-Instruct [43] on the same 40K examples with a matched LoRA configuration, optimizer schedule, and number of updates; the label is placed before the moderated region, and its class score is the autoregressive log-likelihood of the teacher-forced moderated text (Section 4.4).

Table 3: Matched controls. AUPRC, F1, and ECE ($\times 100$) are averaged over the six mixed-label held-out benchmarks (PHTest excluded), so ECE differs from Section 4.3.

Backbone	Training objective	Inference	AUPRC ↑	F1 ↑	ECE ↓
LLaDA-8B-Instruct	none	direct verdict	83.53	68.94	21.31
LLaDA-8B-Instruct	none	reconstruction	34.73	39.54	17.03
LLaDA-8B-Instruct	direct verdict	direct verdict	90.36	81.34	9.71
LLaDA-8B-Instruct	direct verdict	reconstruction	36.39	40.05	17.84
LLaDA-8B-Instruct	class-cond. (LLaDA-Guard)	direct verdict	83.88	81.68	10.39
LLaDA-8B-Instruct	class-cond. (LLaDA-Guard)	reconstruction	**92.66**	**84.91**	**7.70**
Qwen2.5-7B-Instruct	none	direct verdict (next token)	51.87	44.76	21.06
Qwen2.5-7B-Instruct	causal class-cond.	causal class-cond.	88.13	81.23	9.44

Table 4: Matched-control ablation on held-out AUPRC. Avg. is over the six mixed-label held-out benchmarks. Δ is LLaDA-Guard minus LLaDA-Guard Direct.

Model	ToxicChat	OpenAI Mod	XSTest	SafeRLHF	BeaverTails	XSTest-Resp	Avg
LLaDA-Guard Direct 8B	0.8163	0.8163	0.9594	0.9778	0.9104	0.9412	0.9036
LLaDA-Guard 8B	**0.8452**	**0.8568**	**0.9691**	0.9787	**0.9294**	**0.9806**	**0.9266**
Δ	+0.0289	+0.0405	+0.0097	+0.0009	+0.0190	+0.0394	+0.0230

Table 5: Matched-control ablation on held-out F1 at the default decision threshold. Avg. is over the six mixed-label held-out benchmarks. Δ is LLaDA-Guard minus LLaDA-Guard Direct.

Model	ToxicChat	OpenAI Mod	XSTest	SafeRLHF	BeaverTails	XSTest-Resp	Avg
LLaDA-Guard Direct 8B	0.7566	0.7339	0.8855	0.9190	0.7342	0.8514	0.8134
LLaDA-Guard 8B	**0.7755**	**0.7513**	**0.8877**	0.9301	**0.8219**	**0.9281**	**0.8491**
Δ	+0.0189	+0.0174	+0.0022	+0.0111	+0.0877	+0.0767	+0.0357

Tables 4 and 5 show that class-conditional reconstruction improves over the matched direct-verdict model on every held-out dataset, by 2.30 AUPRC and 3.57 F1 points on average. The largest fixed-threshold gains occur on the response datasets BeaverTails ($+8.77$ F1) and XSTest-Response ($+7.67$ F1), where the verdict should depend on the response rather than on prompt features. Table 3 additionally reports both base checkpoints and the two crossed configurations of training objective and inference rule.

B.2 Calibration breakdowns

Tables 6 and 7 report the corresponding calibration results. Table 6 summarizes average ECE, Brier score, and NLL across the held-out benchmarks, with all values multiplied by 100; lower values indicate better calibrated probabilities. LLaDA-Guard obtains the best score on all three metrics, indicating that its strong classification performance does not come at the cost of unreliable confidence

Table 6: Held-out-benchmark calibration summary ($\times 100$, lower is better). Avg. over ToxicChat, OpenAI Mod, XSTest, SafeRLHF, BeaverTails, XSTest-Response, PHTest Harmless.

Model	ECE	Brier	NLL
ShieldGemma 2B	19.79	18.26	64.21
ShieldGemma 9B	11.90	11.85	40.35
ShieldGemma 27B	16.05	16.60	50.78
Llama-Guard 3 8B	9.51	9.84	47.37
Llama-Guard 4 12B	12.74	12.65	168.65
Granite Guardian 3.0 2B	25.77	20.71	62.58
Granite Guardian 3.2 5B	11.81	10.81	35.28
Qwen3Guard 0.6B	16.61	13.64	51.57
Qwen3Guard 4B	14.21	11.77	48.71
Qwen3Guard 8B	13.84	11.29	48.33
NemoGuard 8B	12.63	10.40	50.17
WildGuard 7B	10.91	10.37	47.86
LLaDA-Guard 8B	**8.75**	**8.45**	**32.56**

Table 7: Held-out-benchmark expected calibration error (ECE) comparison. Avg is the mean over the displayed datasets in this table.

Model	ToxicChat	OpenAI Mod	XSTest	SafeRLHF	BeaverTails	XSTest-Resp	PHTest	Avg
ShieldGemma 27B	0.0419	0.0611	0.2081	0.2936	0.3553	0.0424	0.1214	0.1605
ShieldGemma 2B	0.0950	0.2106	0.1568	0.3540	0.4236	0.0812	0.0639	0.1979
ShieldGemma 9B	0.0252	0.0878	0.1065	0.2087	0.2105	0.0603	0.1341	0.1190
Llama-Guard 4 12B	0.1179	0.1546	0.1331	0.1121	0.2707	0.0337	0.0697	0.1274
Llama-Guard 3 8B	0.0773	0.0705	0.0774	0.0741	0.2664	0.0248	0.0753	0.0951
Granite Guardian 3.0 2B	0.3849	0.2734	0.2284	0.0542	0.1762	0.0331	0.6539	0.2577
Granite Guardian 3.2 5B	0.0535	0.1654	0.1188	0.0429	0.1269	0.0245	0.2947	0.1181
NemoGuard 8B	0.0487	0.1676	0.1046	0.0383	0.1854	0.0346	0.3046	0.1263
Qwen3Guard 0.6B	0.1546	0.2880	0.1203	0.1168	0.1052	0.0332	0.3444	0.1661
Qwen3Guard 4B	0.1167	0.2696	0.0804	0.1117	0.1193	0.0336	0.2633	0.1421
Qwen3Guard 8B	0.1199	0.2688	0.0790	0.1142	0.1199	0.0320	0.2354	0.1384
WildGuard 7B	0.0799	0.2055	0.0325	0.0603	0.1270	0.0206	0.2382	0.1091
LLaDA-Guard 8B	0.0397	0.1601	0.0534	0.0482	0.1418	0.0190	0.1506	0.0875

estimates. Table 7 breaks down ECE by dataset, showing that our calibration advantage is stable across individual benchmarks: LLaDA-Guard achieves the lowest average ECE overall and remains consistently competitive across prompt-only, response, and harmless-prompt settings. Together, these results support the main claim that class-conditional reconstruction yields guard scores that are both competitive for detection and substantially better calibrated than the baselines.

C Additional Diagnostic Failure-Mode Results

This appendix collects the expanded analyses behind Section 4.6.

Positional sensitivity. Figure 8 expands the aggregate positional-sensitivity result from Figure 5 into per-model curves. The y-axis normalizes each model by its own peak F1, so the relevant signal is the drop from the model's best operating point as filler length grows.

Prefix-score boundary. Figure 9 shows the full XSTest prefix-reveal experiment for all baselines. Each translucent line is one example and each thick line is the class mean.

Prefix-score jumps. Figure 10 makes the previous observation explicit by plotting the change in `unsafe` probability between consecutive prefix steps. Large positive spikes mean that a short newly revealed fragment caused a large verdict shift.

Response-objective fidelity. Table 8 tests whether response moderation is actually grounded in the response. The response is held fixed and `safe`; only the prompt changes from `safe` to `unsafe`. A response-grounded model should therefore produce nearly identical verdicts in both conditions.

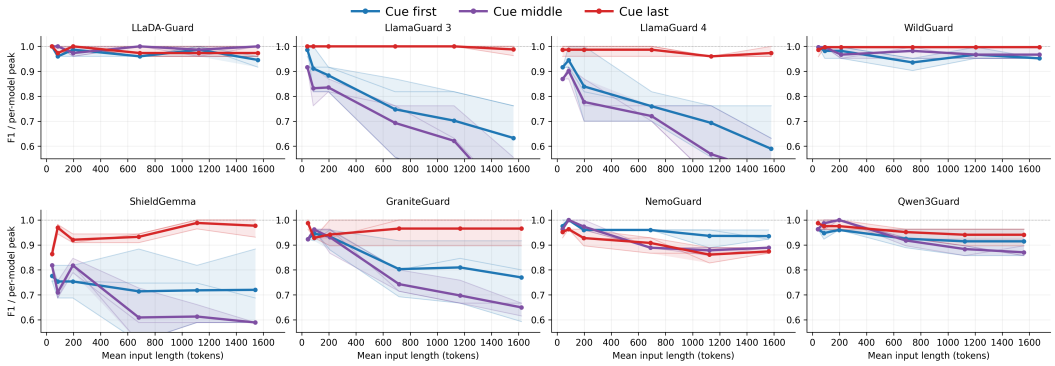


Figure 8: Per-model positional bias investigation. Each panel shows F1 normalized by that model's peak F1 across input-length buckets, with curves grouped by the position of the decisive safety cue: first, middle, or last. Flat curves as filler length increases indicate less positional sensitivity.

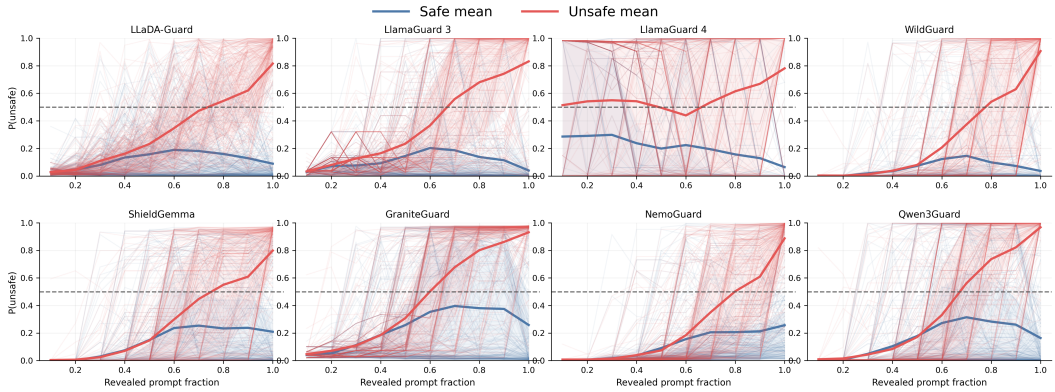


Figure 9: Prefix-score trajectories on XSTest. Each prompt is revealed from left to right, and the guard score is recomputed after each prefix. Mean `safe` and `unsafe` trajectories summarize how quickly each model crosses the `unsafe` decision boundary.

D NotInject Over-Defense Analysis

NotInject contains benign prompts with prompt-injection-like trigger patterns, so any `unsafe` prediction is a false positive. Table 9 breaks these false positives down by the number of trigger words in the prompt. LLaDA-Guard produces no false positives across all trigger-count splits, matching ShieldGemma and avoiding the trigger-count sensitivity seen in several strong classifiers. Granite is the most affected baseline, rising from 1.77% false positives with one trigger to 10.62% with three triggers; Qwen3Guard and WildGuard also become more defensive as trigger count increases. This supports the main-text over-refusal result: LLaDA-Guard does not achieve strong held-out-benchmark performance by simply flagging benign prompts that contain suspicious safety or injection cues.

E Masking-Rate and Mask-Count Sweeps

At inference, the diffusion time t is the masking rate, i.e., the fraction of the moderated region hidden in one evaluation, and K is the number of independent masks scored under each label hypothesis. K primarily controls the Monte Carlo variance of the score, whereas t trades off variance against hidden context: small t scores few, heavily weighted tokens, while large t lowers variance but hides more of the evidence. Since t changes how much context is hidden but not the processed sequence length, it has little effect on evaluation cost.

Aegis 2.0 validation. Figure 11 evaluates LLaDA-Guard on the Aegis validation set across masking rates t and repeated mask samples. Performance is robust across a broad range of masking levels:

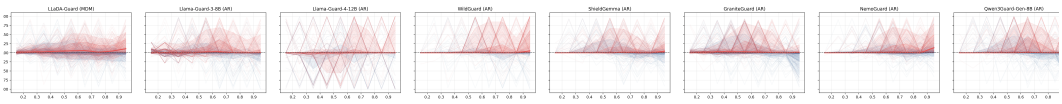


Figure 10: Prefix-score jump analysis on XSTest. Curves show changes in `unsafe` probability between consecutive revealed-prefix steps, highlighting abrupt decision-boundary jumps.

Model	Matched $\Delta P_{\texttt{unsafe}} \downarrow$	Matched $\Delta \hat{y}_{\texttt{unsafe}} \downarrow$
LLaDA-Guard	**0.099**	**0.060**
LlamaGuard 3	0.206	0.208
LlamaGuard 4	0.138	0.132
WildGuard	0.292	0.308
GraniteGuard	0.122	0.140
Qwen3Guard	0.386	0.424
ShieldGemma	0.008	0.000
NemoGuard	0.019	0.004

Table 8: Response-objective fidelity on matched `safe`-response pairs ($n = 250$). For each pair, the same `safe` response is attached once to a `safe` prompt and once to an `unsafe` prompt, and the model is asked to classify the response. We report the increase in `unsafe` probability ($\Delta P_{\texttt{unsafe}}$) and `unsafe` prediction rate ($\Delta \hat{y}_{\texttt{unsafe}}$) when only the prompt changes from `safe` to `unsafe`. Lower is better. Rows below the rule (ShieldGemma, NemoGuard) use a separate template for prompt and response classification, which sidesteps prompt leakage by construction and yields $\Delta \approx 0$.

AUPRC stays near 0.93 for $t \leq 0.4$, and F1 remains within roughly three points across the sweep. Repeats help most for calibration, with ECE consistently improving from one pass to multiple passes, offering a way to use test-time computation to boost calibration.

Held-out benchmarks and cost. Table 2 in Section 4.5 reports the K sweep on the six mixed-label held-out benchmarks at the default $t = 0.15$, together with its inference cost. Increasing K from 1 to 4 raises median latency from 127.4 to 508.4 ms per example while reducing ECE from 0.0826 to 0.0754; F1 and AUPRC change modestly (from 0.8384 to 0.8480 and from 0.9152 to 0.9250). In a separate repeated-seed sweep over the six datasets with five seeds, increasing K from 1 to 4 raises the prediction agreement across seeds from 95.99% to 98.10%. Parameter memory stays fixed at one checkpoint, while peak activation memory grows with the number of evaluations batched concurrently, so lower serial latency trades off against higher memory usage.

Masking rate on held-out benchmarks. Table 10 varies how much of the moderated region is hidden. At small t, the estimator has higher variance because a few tokens receive large weights; at large t, variance is lower but F1 drops, from 0.851 at $t = 0.05$ to 0.818 at $t = 0.5$ with $K = 4$, as too much context is hidden. Independent masks partially mitigate this by exposing different context subsets, but cannot recover evidence hidden at high t.

Token-level concentration. Although the objective is a sum of token reconstruction terms, each term is conditioned on the visible bidirectional context, so token-level supervision does not imply that decisions rest on isolated words. To test this, we define the contribution of token i as its `unsafe`-minus-`safe` reconstruction margin and the top-k concentration as the sum of the k largest absolute contributions divided by the total absolute contribution. At $t = 0.15$, across the six held-out benchmarks, the median top-1 and top-5 concentrations are 4.28% and 17.67%, and forcibly masking the top-1 token flips only 2.37% of verdicts, arguing against domination by an isolated token.

F Reconstruction-Loss Weight

$\mathcal{L}_{\texttt{mod}}$ trains the relative `unsafe`-versus-`safe` log-likelihood ratio and, by itself, only requires the correct hypothesis to rank above the incorrect one. $\mathcal{L}_{\texttt{gen}}$ supplies the absolute signal that prevents solutions in which both hypotheses explain the text poorly but are ordered correctly. Table 11 trains models from the same base checkpoint while varying λ; the submitted value $\lambda = 0.1$ was selected on the validation set.

Table 9: NotInject over-defense by trigger-count split. All examples are benign; entries show false-positive percentage with false-positive count in parentheses, plus average false-positive confidence when any false positives occur.

Model	1 trigger	2 triggers	3 triggers
LLaDA-Guard	0.00% (0) conf. –	0.00% (0) conf. –	0.00% (0) conf. –
Qwen3Guard	0.00% (0) conf. –	0.88% (1) conf. 80.59%	3.54% (4) conf. 77.55%
WildGuard	0.88% (1) conf. 65.14%	0.88% (1) conf. 99.44%	2.65% (3) conf. 78.16%
Granite	1.77% (2) conf. 60.61%	5.31% (6) conf. 64.70%	10.62% (12) conf. 70.61%
NemoGuard	0.00% (0) conf. –	0.00% (0) conf. –	1.77% (2) conf. 63.69%
Llama-Guard 3	0.00% (0) conf. –	0.00% (0) conf. –	0.88% (1) conf. 90.47%
ShieldGemma	0.00% (0) conf. –	0.00% (0) conf. –	0.00% (0) conf. –
Llama-Guard 4	0.00% (0) conf. –	0.00% (0) conf. –	2.65% (3) conf. 89.94%

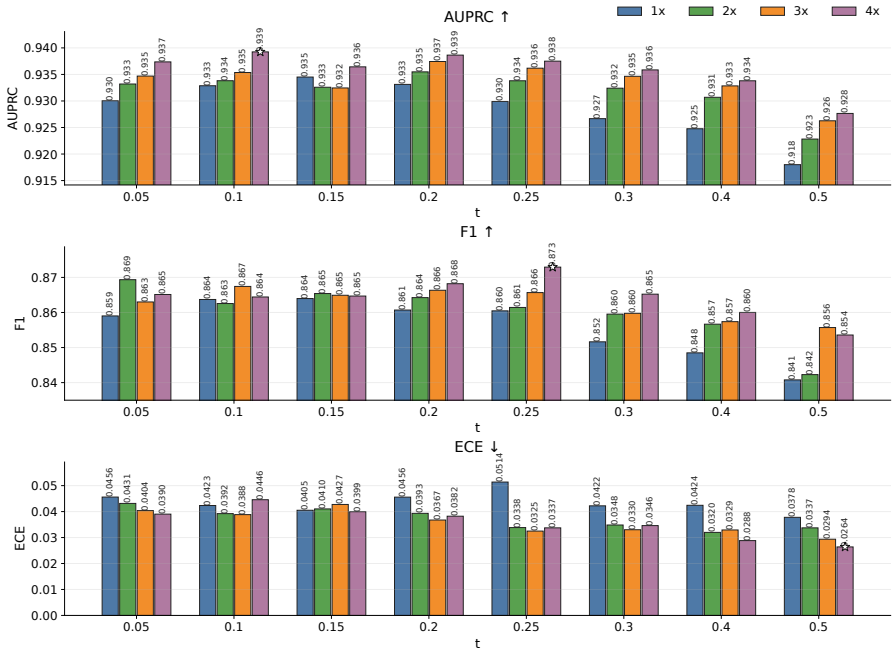


Figure 11: Aegis 2.0 validation performance across masking rates t and mask sampling repetitions K.

The sweep shows a plateau for $\lambda \in [0.025, 0.4]$. The reconstruction loss is therefore not the primary driver of either classification or ECE, although moderate weights give a small improvement over $\lambda = 0$. Consistently, the overconfident-error rate changes only from 0.0439 at $\lambda = 0$ to 0.0418 at $\lambda = 0.1$ and is non-monotonic through $\lambda = 0.4$, indicating that $\mathcal{L}_{\text{gen}}$ is not the main mechanism preventing confidently wrong predictions. The degradation in F1, AUPRC, and ECE for $\lambda \geq 1.6$ is consistent with over-weighting absolute reconstruction at the expense of class separation. We interpret the small differences within the plateau as a stable region rather than evidence that one value is optimal.

G Hazard-Category Extension

The main LLaDA-Guard checkpoint is binary. Among the baselines, Llama Guard, Qwen3Guard, and NemoGuard can emit category labels, WildGuard cannot, and ShieldGemma and Granite Guardian make binary decisions against categories supplied in the input. To show that the framework extends to hazard taxonomies, we fine-tune LLaDA-Guard with an additional reconstruction loss over the 12

Table 10: Held-out F1 and score variance across masking rates t for $K = 1$ and $K = 4$, averaged over the six mixed-label held-out benchmarks.

t	F1 ($K = 1$)	F1 ($K = 4$)	Score var. ($K = 1$)	Score var. ($K = 4$)
0.05	0.839	**0.851**	2.24	0.55
0.15	0.838	0.848	1.04	0.27
0.30	0.827	0.841	0.69	0.17
0.50	0.799	0.818	0.65	**0.16**

Table 11: Effect of the reconstruction-loss weight λ. Score gap is the separation between the average safety scores of `unsafe` and `safe` examples. Overconfident errors are wrong predictions with confidence above 90%. All metrics are macro-averaged over the six mixed-label held-out benchmarks (PHTest excluded).

λ	Macro F1 ↑	AUPRC ↑	ECE ↓	Score gap	Overconf. error ↓
0	0.8431	0.9245	0.0775	7.60	0.0439
0.025	**0.8502**	**0.9298**	0.0772	8.20	0.0500
0.1	0.8491	0.9266	0.0770	7.38	0.0418
0.4	0.8489	0.9243	**0.0768**	7.97	0.0448
1.6	0.8211	0.9121	0.0842	5.47	0.0347
6.4	0.8035	0.8948	0.0913	4.89	**0.0268**

Aegis 2.0 categories [12]. When the binary decision is `unsafe`, all twelve category tokens are predicted in parallel in one additional evaluation, using the parallel decoding of MDMs. Autoregressive guards instead decode categories sequentially; for example, Llama Guard generates `unsafe`, `H1`, `H5` token by token, whereas LLaDA-Guard decodes `H1` and `H5` simultaneously.

Table 12: Hazard-category prediction on Aegis 2.0 ($1{,}964$ examples), conditional on the annotated `unsafe` examples, over the H1–H12 taxonomy. Serial decode steps is the mean number of sequential category-decoding steps. No model produced invalid verdicts.

Model	H1–H12 macro-F1 ↑	Exact set match ↑	Serial decode steps ↓
LLaDA-Guard (category ext.)	0.5585	0.4493	**1.00**
Llama Guard 3 8B	0.3504	0.2242	3.04
NemoGuard V2	0.6081	**0.4742**	18.34
NemoGuard V3	**0.6328**	0.4373	19.37

Table 12 shows that category prediction with LLaDA-Guard is competitive with dedicated multi-category guards while requiring a single additional evaluation per `unsafe` example. Category supervision slightly reduces binary performance: on the six held-out benchmarks (Table 13), the category extension reaches 0.8391 F1, 0.9250 AUPRC, and 0.0826 ECE, compared with 0.8478 F1 and 0.9261 AUPRC for a binary LLaDA-Guard reference evaluated in the same pipeline.

H Prompted Frontier-Model Reference

As a high-capability and high-cost reference, we evaluate Gemini 3.6 Flash [13] with medium reasoning and $1{,}024$ maximum output tokens, prompted with the Llama Guard prompt template [21]. Model scale and architecture are not matched, and the API does not expose confidences comparable to those of the other guards, so we report F1 only. Gemini refused to reply with a safety block in 6.5% of cases; we count these as `unsafe`, since the API guardrail was triggered. Re-scoring both models on only the valid rows changes the macro-F1 gap by less than 0.1 points.

Gemini outperforms LLaDA-Guard on XSTest ($+7.06$) and OpenAI Mod ($+9.58$) but underperforms on ToxicChat (-10.99) and BeaverTails (-10.81). The full held-out evaluation with Gemini costs \$24.42, with a median latency of 2.18 s per example. At the throughputs in Table 2 and an illustrative market price of \$2 per GH200-hour, LLaDA-Guard costs roughly \$0.07 per $1{,}000$ examples at $K = 1$ and \$0.21 at $K = 3$. Table 15 reports Gemini's per-dataset accuracy, precision, and recall, both with

Table 13: Binary F1 of the category extension on the six mixed-label held-out benchmarks.

	ToxicChat	OpenAI Mod	XSTest	SafeRLHF	BeaverTails	XSTest-Resp	Avg
F1	0.7533	0.7638	0.8809	0.9223	0.8079	0.9067	0.8391

Table 14: F1 of LLaDA-Guard and prompted Gemini 3.6 Flash on the mixed-label held-out benchmarks.

Dataset	LLaDA-Guard	Gemini 3.6 Flash
ToxicChat	**0.7755**	0.6656
OpenAI Mod	0.7513	**0.8471**
XSTest	0.8877	**0.9583**
SafeRLHF	**0.9301**	0.8703
BeaverTails	**0.8219**	0.7138
XSTest-Response	**0.9281**	0.9272
Macro average	**0.8491**	0.8304

safety-block fallbacks counted as `unsafe` and on valid responses only. Fallbacks are concentrated in BeaverTails (634 of $3{,}021$ examples), where excluding them raises precision from 0.7685 to 0.9275 but lowers recall from 0.6665 to 0.5824, leaving F1 nearly unchanged.

Table 15: Extended Gemini 3.6 Flash results. Fallback is the number of safety-blocked responses, counted as `unsafe` in the left block; the right block is computed on valid responses only.

			All examples (fallback = `unsafe`)				Valid responses only		
Dataset	N	Fallback	Accuracy	Precision	Recall	F1	Precision	Recall	F1
ToxicChat	2853	11	0.9288	0.8245	0.5580	0.6656	0.8376	0.5506	0.6644
OpenAI Mod	1680	26	0.9095	0.8919	0.8065	0.8471	0.8946	0.7980	0.8436
XSTest	450	0	0.9644	1.0000	0.9200	0.9583	1.0000	0.9200	0.9583
SafeRLHF	2000	5	0.8820	0.9659	0.7920	0.8703	0.9681	0.7914	0.8709
BeaverTails	3021	634	0.6935	0.7685	0.6665	0.7138	0.9275	0.5824	0.7155
XSTest-Response	446	0	0.9753	0.9589	0.8974	0.9272	0.9589	0.8974	0.9272

I Localization and Rewriting Details

For the attention localizer, we rank tokens by their attention contribution to the verdict position. This provides a common attribution proxy for all guard models, but it is indirect: attention indicates where the verdict representation attends, not which tokens causally change the safety score.

For the leave-one-out localizer, we exploit LLaDA-Guard's masked scoring. For each token i in the moderated region, we temporarily mask that token, recompute the safety score, and rank tokens by

$$\Delta_i = \hat{s}_\theta(M) - \hat{s}_\theta^{(-i)}(M),$$

where $\hat{s}_\theta^{(-i)}(M)$ denotes the score after masking token i. Larger positive values indicate that token i contributes more strongly to the `unsafe` likelihood-ratio score. This gives a more direct attribution signal, at the cost of additional scoring passes.

For rewriting, the selected tokens are masked and infilled by LLaDA-Guard under the `safe` label condition. We repeat this process until the judge classifies the prompt as `safe` or the masking budget is exhausted.

I.1 Full rewriting results

Tables 16 and 17 report the full rewriting results per dataset and budget, corresponding to the main-text rewriting figure.

Localizer	B1	B2	B3	B4	B5
ShieldGemma 27B	10.9	20.9	29.8	38.8	43.4
ShieldGemma 2B	16.5	32.9	41.4	47.3	50.7
ShieldGemma 9B	12.2	21.0	27.5	32.9	40.9
Llama-Guard 4 12B	14.7	26.1	33.5	39.3	44.7
Llama-Guard 3 8B	12.4	23.0	32.6	41.7	48.6
Granite Guardian 2B	13.7	24.8	34.3	40.6	45.9
Granite Guardian 5B	19.6	34.7	43.1	47.2	50.1
NemoGuard 8B	12.0	27.2	40.1	46.4	51.0
Qwen3Guard 0.6B	13.7	22.3	31.5	37.2	42.7
Qwen3Guard 4B	14.3	28.4	36.3	43.3	47.4
Qwen3Guard 8B	14.1	27.4	36.5	43.3	48.1
WildGuard 7B	14.0	32.9	42.4	48.9	52.1
LLaDA-Guard 8B	21.3	38.0	46.4	50.7	53.0
LLaDA-Guard 8B Masked	37.6	47.9	53.3	57.1	60.7

Table 16: Dataset-balanced cumulative conversion-to-safe rate (%) by rewrite budget. B1–B5 denote increasing token-rewrite budgets; each value averages over XSTest, ToxicChat, and OpenAI Mod.

Localizer	XSTest	ToxicChat	OpenAI Mod	Avg
ShieldGemma 27B	85.0	35.1	10.1	43.4
ShieldGemma 2B	96.0	42.3	13.8	50.7
ShieldGemma 9B	77.0	34.5	11.1	40.9
Llama-Guard 4 12B	85.5	37.0	11.5	44.7
Llama-Guard 3 8B	92.0	41.4	12.4	48.6
Granite Guardian 2B	91.0	37.0	9.7	45.9
Granite Guardian 5B	92.5	43.9	14.0	50.1
NemoGuard 8B	95.0	45.3	12.8	51.0
Qwen3Guard 0.6B	84.5	35.4	8.2	42.7
Qwen3Guard 4B	93.5	37.6	11.3	47.4
Qwen3Guard 8B	96.0	38.1	10.1	48.1
WildGuard 7B	98.0	45.6	12.8	52.1
LLaDA-Guard 8B	99.0	44.8	15.1	53.0
LLaDA-Guard 8B Masked	99.0	55.0	17.0	60.7

Table 17: Conversion-to-safe rate (%) by localizer and dataset at a rewrite budget of 5.

J Limitations, Broader Impact, and Compute

J.1 Limitations

Our results are limited by the scope of the evaluated datasets and policies. The main evaluation focuses on English prompt and response moderation benchmarks with binary `safe`/`unsafe` labels; we do not evaluate multilingual, multimodal, or policy-customized moderation settings. The learned class-conditional likelihoods also inherit the coverage, label noise, and policy assumptions of the training data. As a result, improved calibration and reduced shortcut behavior on the tested benchmarks should not be interpreted as a guarantee of robustness to all distribution shifts, adaptive jailbreaks, or future safety taxonomies.

LLaDA-Guard also changes the computational profile of guard inference. A direct verdict-token guard can in principle score its label token with a single forward pass, although released guards in their default configuration emit complete structured outputs that require several serial decode steps, and category prediction requires further decoding. LLaDA-Guard instead estimates two class-conditional reconstruction scores and averages across sampled masks, giving $2K$ evaluations; the default evaluation uses $K = 3$. These evaluations are independent and can be batched, but computation still grows linearly in K: in our measurements, the default setting has $3.64\times$ the median latency of Qwen3Guard, and $K = 1$ has $1.21\times$ (Table 2). This additional computation is useful for calibration and token-level evidence, but it can be costly in high-throughput moderation pipelines. Likelihood estimators from continuous diffusion language models [53] may motivate cheaper or lower-variance alternatives, but adapting them to class-conditional moderation is left to future work.

The masked-reconstruction score is a Monte Carlo estimate of a variational bound rather than the exact class-conditional log-likelihood: within each sampled mask, masked tokens are treated as conditionally independent given the visible context. Increasing K reduces variance but not this approximation. The main checkpoint outputs binary verdicts; the hazard-category extension in Appendix G is a separately fine-tuned model that slightly reduces binary performance. The crossed controls in Table 3 show that the gains require pairing the class-conditional objective with reconstruction inference, so neither component can be adopted in isolation. Finally, the prompt-rewriting experiment is intended as a proof of concept.

J.2 Broader Impact and Ethics

The intended positive impact of this work is to improve the reliability of safety moderation systems. Better calibrated guard models can support more transparent refusals, reduce overconfident mistakes, and help distinguish harmful content from benign prompts that contain superficially unsafe terms. The prompt-rewriting pipeline may also help systems offer safer alternatives rather than only blocking user requests.

The same capabilities carry risks. More informative guard scores and token-level localization could be misused to probe a moderation system, optimize around safety filters, or identify which fragments trigger enforcement. A guard model can also be used for overbroad filtering, censorship, or surveillance if deployed under inappropriate policies. Further, a guard model can be used adversarially to generate jailbreak attacks. False positives may block legitimate speech, while false negatives may allow harmful content through. These risks motivate treating LLaDA-Guard as one component of a larger safety stack, with policy-specific validation, monitoring after deployment, and human review for high-impact decisions. Our experiments use existing public safety datasets and released model checkpoints; we do not collect new human-subject data.

J.3 Compute and Hardware

Experiments were run on two systems: a single machine with $1\times$ NVIDIA H100 NVL, 30 Intel Xeon Platinum 8562Y+ cores, and 96 GB RAM, and the Clariden cluster of the Swiss National Supercomputing Centre (CSCS) on Alps, whose nodes each contain $4\times$ NVIDIA GH200. On the H100 system, the wall-clock time for a full LLaDA-Guard training run from LLaDA-8B-Instruct is 6 hours and 45 minutes, and the full evaluation time, including baselines, is estimated at 11 hours. The additional training runs (the λ sweep in Appendix F, the causal Qwen2.5 control, the direct-verdict model, and the category extension) use the same data and optimization budget as LLaDA-Guard. All latency and memory measurements (Table 2) were taken on a single GH200 over 30 timed batches (120 example executions) after 5 warm-up iterations, with 1,024-token inputs, $t = 0.15$, and batch size 4; median latency per example and peak allocated memory are reported. The Gemini 3.6 Flash reference (Appendix H) was run through the public API at a cost of $24.42.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's main contribution: reframing safety moderation as class-conditional generative reconstruction using a masked diffusion language model. The method and evaluation sections support the claims made in the abstract and introduction about competitive benchmark performance, improved calibration, and prompt rewriting.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated limitations discussion in Appendix J.1. It states that the evaluation is limited to English text data, binary safe/unsafe moderation benchmarks, does not cover multilingual, multimodal, or policy-customized moderation, and that strong results nevertheless do not guarantee robustness to all distribution shifts or adaptive jailbreaks.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not introduce new formal theoretical results requiring proofs. However, it uses Bayes' rule to construct the proposed method and draws on the Neyman-Pearson lemma and likelihood-ratio tests to support its formulation. Moreover, assumptions for certain statements, like equal priors for safe and unsafe classes, are directly stated in the corresponding text.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides important reproducibility information, including the backbone, LoRA adaptation, training data sources and proportions, benchmark datasets, baselines, metrics, masking samples, masking rate, and threshold. Additional details needed to reproduce the main experiments, such as exact optimizer settings, LoRA hyperparameters, batch size, training schedule, seeds, complete preprocessing, and executable commands, are included in the released codebase.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The abstract links to the code repository, which also hosts the trained checkpoint, for reproduction. The code also includes scripts to download the public datasets used for evaluation.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The manuscript provides key details to understand the evaluation. It describes the characteristics of different datasets, baselines, and metrics. It also specifies relevant hyperparameter values, namely the probability threshold used for class assignment across all models, as well as the K and t hyperparameters of the proposed model. It additionally provides information on the base model, the objective, and the data used to fine-tune the proposed model. Finally, it includes additional details in Appendix A and the linked repository.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The primary benchmark evaluations are deterministic and do not involve run-to-run variance or random sampling when the seed is fixed for LLaDA-Guard. Hence, error bars, confidence intervals, or statistical significance tests are not applicable for the corresponding metrics. One exception is Figure 5, which provides an aggregated overview of baseline models' performance, alongside the results for individual models, for which we include standard deviation in the visualization.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper includes information on the hardware resources used to create the model and evaluate it against the baselines in Appendix J.3, including the H100 system, the CSCS Clariden cluster ($4\times$ GH200 nodes), training and evaluation wall-clock times, and the GH200 setup used for the latency and memory measurements in Table 2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work conforms with the NeurIPS Code of Ethics. It uses existing public safety datasets and releases model weights. It does not collect new human-subject data and discusses risks such as misuse for probing guardrails, overbroad filtering, censorship, surveillance, false positives, adversarial training, and false negatives.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses positive impacts, including more reliable safety moderation, better calibrated thresholding, fewer overconfident mistakes, and safer alternatives via prompt rewriting. It also discusses negative impacts, including misuse of token-level localization to probe safety filters, overbroad filtering, censorship, surveillance, false positives, and false negatives.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

Justification: The paper concerns guard models built on pre-trained language models. Unlike standard language models, the primary purpose of guard models is to assess the safety of a given piece of text rather than to generate arbitrary text. However, we note that our proposed model also includes rewriting capabilities, as showcased in our evaluation, which, while specifically targeted at rewriting unsafe text into a safe version, still carry some risk of unintended behavior.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The work references all the public datasets and models used during evaluation and training in the corresponding main text section.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The repository linked in the manuscript includes documentation on rerunning the fine-tuning to generate LoRA weights, using the provided or self-trained weights with the base model to classify text outputs, and reproducing the main evaluation steps. We release the repository under a permissive license.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve new crowdsourcing experiments or new human-subject data collection. Experiments use existing public safety datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve new human-subject research, crowdsourcing, or participant interaction. It uses existing public safety datasets and released checkpoints, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The involvement of LLMs in the paper is limited to editing (e.g., grammar, spelling, word choice), visualizing results for submission, implementing standard methods, and literature search.

    Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.