$\mathcal{D}^2$-MONITOR: $\mathcal{D}$YNAMIC SAFETY MONITORING FOR $\mathcal{D}$IFFUSION LLMS VIA HESITATION SIGNALS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the emergence of diffusion large language models (D-LLMs) as an alternative to autoregressive large language models (AR-LLMs), safety monitoring for D-LLMs remains largely unexplored. Unlike AR-LLMs, D-LLMs generate text through a *multi-step* denoising process, exposing intermediate hidden representations that may contain safety-relevant information unavailable in standard *single-step* monitoring setups. Motivated by the suitability of lightweight probes for always-on monitoring, we analyze which trajectory-level signals best indicate when such probes are likely to struggle. We find that the most informative signal is *safety hesitation*: intermediate hidden states repeatedly falling within a small margin of the probe's decision boundary. The number of such hesitation steps in D-LLM's trajectory predicts probe failure effectively, providing a proxy for sample difficulty. Building on this analysis, we propose **$\mathcal{D}^2$-Monitor**, in which a lightweight probe runs on every sample, producing a base prediction and a hesitation estimate that decides whether to escalate the sample to a more expressive probe. That probe is trained on trajectories with at least one hesitation step, and reads only the minimal span of those steps. Evaluated on 3 datasets (WildGuardMix, ToxicChat, OpenAI-Moderation) across 4 D-LLMs, $\mathcal{D}^2$-Monitor achieves state-of-the-art performance with a compact parameter footprint ($\leq$ 0.93M parameters), and exhibits the best trade-off between effectiveness and efficiency relative to 8 baselines.

1 INTRODUCTION

Building on causal attention (Vaswani et al., 2017) and the next-token prediction paradigm, autoregressive large language models (AR-LLMs) (Achiam et al., 2023; Grattafiori et al., 2024; Yang et al., 2025) have achieved remarkable performance across diverse tasks, including code generation (Chen et al., 2021; Li et al., 2022) and mathematical reasoning (Cobbe et al., 2021; Wei et al., 2022). Despite their success, this autoregressive paradigm introduces inherent limitations: the sequential decoding constrains generation efficiency and prevents models from revising earlier outputs in light of future context. Diffusion large language models (D-LLMs) (Nie et al., 2025b; Zhu et al., 2025; Bie et al., 2025; Inception Labs et al., 2025) have recently emerged as a promising alternative. Rather than generating tokens sequentially, D-LLMs iteratively refine the entire sequence through a denoising process with bidirectional attention (Sahoo et al., 2024; Shi et al., 2024), enabling faster and more flexible generation. The commercial D-LLM `Mercury 2` (Inception Labs, 2026) generates 1009 tokens per second, significantly outperforming AR-LLMs such as `Claude Haiku 4.5` (89 tokens/sec) and `GPT-5-mini` (71 tokens/sec). On the open-source side, `LLaDA 2.0` (Bie et al., 2025) scales D-LLMs to 100B parameters and rivals leading AR-LLMs (Yang et al., 2025).

Despite these advances, safety monitoring for D-LLMs remains underexplored. Effective monitoring is critical: frontier large language models already significantly lower the barrier for malicious actors to execute harmful tasks (Anthropic, 2025). Initial work on D-LLM safety has focused primarily on alignment techniques (Jeung et al., 2026; Li et al., 2026b) that improve safety awareness within the model. However, alignment alone is insufficient, as such techniques remain vulnerable to adversarial attacks (Nasr et al., 2025). We therefore focus on *external* safety monitors in this paper, which are deployment-time systems that detect harmful user inputs (Jiao et al., 2026; Oldfield et al., 2026) or problematic model behaviors (Goldowsky-Dill et al., 2025; McKenzie et al., 2025).

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

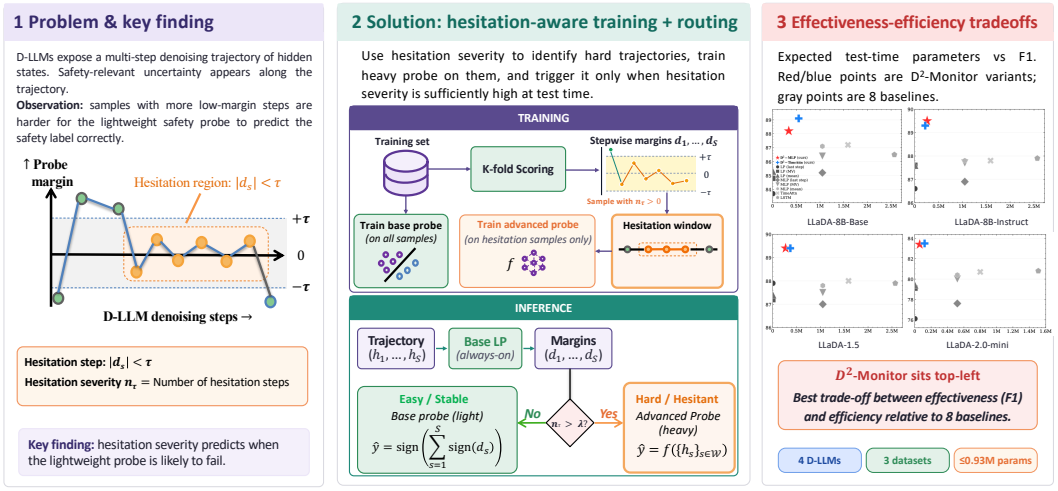


Figure 1: **Left**: Hesitation severity n_τ counts the denoising steps whose probe margin falls within τ, and predicts when the lightweight probe fails. **Middle**: $\mathcal{D}^2$-Monitor uses n_τ to curate the advanced probe's training data and to route at test time. **Right**: Expected test-time parameters versus F1 on *WildGuardMix*; $\mathcal{D}^2$-Monitor attains the best F1 with fewer parameters than most baselines.

Existing safety monitoring literature has focused on AR-LLMs and falls into two broad categories. *LLMs-as-monitors* (Inan et al., 2023; Zeng et al., 2025) employ additional LLMs to classify the safety of user prompts or model responses. *Probe-based monitors* operate on internal model representations, which have been shown to encode rich semantic information (Alain & Bengio, 2016; Meng et al., 2022). Their lightweight architectures suit always-on, low-cost deployment, and are increasingly adopted in production systems such as Google's `Gemini` (Kramár et al., 2026).

In this paper, we first argue that D-LLMs' *multi-step* trajectory provides a richer and more useful signal for safety monitoring than *single-step* representations (Section 3.2). Inspired by recent findings that intermediate D-LLM outputs can oscillate between correct and incorrect answers during mathematical reasoning (Wang et al., 2026b; Li et al., 2026a), we show that analogous instability occurs in the safety probe space. Specifically, we identify *hesitation steps*, i.e., intermediate denoising steps whose representations lie close to the probe decision boundary (Section 3.3). We further demonstrate that trajectories with more hesitation steps are harder for probes to classify correctly. This establishes hesitation as an effective proxy for sample difficulty, and naturally motivates a bi-level monitor design that routes hard samples to a high-complexity probe while processing easy samples with a lightweight one, dynamically allocating computational resources at test time.

Proposed Work We introduce **$\mathcal{D}^2$-Monitor**, a dynamic safety monitor for D-LLMs that harnesses intrinsic safety hesitation in the multi-step denoising trajectory. Following prior work on cascaded monitors (Section 2), $\mathcal{D}^2$-Monitor adopts a bi-level design, in which we use the hesitation severity n_τ as the routing signal and the minimal span of hesitation steps to train the more expressive probe. The base probe serves as an *always-on monitor*, jointly estimating hesitation and performing base-level safety classification. When the hesitation level exceeds a threshold, the router activates the high-complexity advanced probe for second-stage classification. This dynamic routing mechanism allocates monitoring resources efficiently: easy samples incur only lightweight compute cost, while harder samples (such as adversarially crafted inputs) trigger additional safeguards, achieving a practical balance between effectiveness and efficiency.

We evaluate $\mathcal{D}^2$-Monitor on 3 safety datasets (WildGuardMix, ToxicChat, OpenAI-Moderation) across 4 D-LLMs under both *intra-dataset* and *cross-dataset* settings. $\mathcal{D}^2$-Monitor achieves state-of-the-art performance with an extremely compact parameter footprint (fewer than 0.93M parameters, or 0.01% of an 8B model), and exhibits the best trade-off between efficiency and effectiveness relative to 8 baselines. Additional analysis confirms robustness across generation configurations, remasking strategies, and hyperparameter settings. Our main contributions are threefold:

- We characterize *safety hesitation* in the multi-step hidden states of D-LLMs using probe margins, and show that hesitation severity strongly correlates with linear probe performance.

- We introduce **$\mathcal{D}^2$-Monitor**, a dynamic safety monitor for D-LLMs that uses trajectory-level hesitation signals both for curating advanced probe training data and for test-time routing.
- Tested on 3 safety datasets across 4 D-LLMs, $\mathcal{D}^2$-Monitor achieves state-of-the-art performance under both *intra-dataset* and *cross-dataset* settings, with the best trade-off between effectiveness and efficiency against 8 baselines.

2 Related Work

Diffusion Large Language Models Diffusion large language models (D-LLMs) (Nie et al., 2025b; Zhu et al., 2025; Bie et al., 2025), built upon masked diffusion models (Austin et al., 2021; Hoogeboom et al., 2021; Shi et al., 2024; Nie et al., 2025a), extend diffusion-based generative modeling from continuous domains such as images (Yang et al., 2023) to discrete text. Recent work has identified intrinsic safety-relevant properties of diffusion-based generation relative to autoregressive generation (He et al., 2026). Early efforts to safeguard D-LLMs have mainly explored finetuning-based defenses (Jeung et al., 2026) and decoding intervention (Li et al., 2026b). However, finetuning approaches incur substantial computational overhead and may affect model utility, while decoding intervention requires regeneration, which affects efficiency. We instead adopt a probe-based monitoring approach: a lightweight auxiliary module that can be deployed alongside any D-LLM without modifying the underlying model, offering a practical and non-intrusive defense mechanism.

LLM Monitors Despite extensive safety training, LLMs remain vulnerable to adversarial attacks (Liu et al., 2024; Zeng et al., 2024; Chen et al., 2026), making external safety guardrails necessary (Kramár et al., 2026). These guardrails fall into two broad categories. **(1) LLMs-as-monitors.** One approach deploys an additional LLM trained as a safety classifier to filter inputs and outputs (Markov et al., 2022; Han et al., 2024; Padhi et al., 2024; Liu et al., 2025). Representative models such as Llama-Guard (Inan et al., 2023) are fine-tuned on safety tasks to improve detection of adversarially crafted prompts. While capable, LLM-based monitors add substantial compute, which is prohibitive in resource-constrained settings such as edge deployment. **(2) Probe-based monitors.** A more efficient alternative trains lightweight probes on the model's internal representations, which encode rich semantic information (Park et al., 2024; Jiao et al., 2026). Linear probes (Alain & Bengio, 2016) are the canonical example, effective on hallucination detection (Han et al., 2025), toxicity detection (Hu et al., 2024), and deception detection (Goldowsky-Dill et al., 2025; MacDiarmid et al., 2024). More expressive probes range from bilinear and MLP variants (Hewitt & Liang, 2019; Pimentel et al., 2020) to architectures that read the entire sequence of states produced during generation, such as temporal attention (Kramár et al., 2026) and recurrent encoders (Damirchi et al., 2026). For AR-LLMs, each generated token contributes one state, whereas a D-LLM re-encodes the whole sequence at every denoising step, producing a trajectory of intermediate states (Section 3.1).

Cascaded Design Routing easy inputs to a cheap model and escalating hard ones is a long-standing idea, from classical detection cascades (Viola & Jones, 2004) to early-exit networks (Teerapittayanon et al., 2016). Recent work applies this to LLM monitoring. McKenzie et al. (2025) and Cunningham et al. (2026) use an activation probe as a first stage and defer uncertain inputs to an external LLM-as-monitor, reporting substantial savings over running that monitor on every input. The cost is that the second stage remains an LLM: it must be finetuned or prompted in advance, and it adds a model call at inference time. Our work differs in three key respects: (1) we introduce a D-LLM specific multi-step routing signal, (2) we use a probe as second-stage classifier instead of an additional LLM, and (3) we score training samples to curate hesitation trajectories for training the second-stage probe. We do not claim the bi-level design itself as a novelty.

3 Exploring Safety Monitoring in D-LLMs

3.1 Preliminary

Diffusion Large Language Models Diffusion large language models define a discrete diffusion process over token sequences. Let $\mathbf{x}^{(1)} \in \mathcal{V}^L$ denote a clean text sequence, where $\mathcal{V}$ is the vocabulary and L is the sequence length. The forward noising process gradually corrupts $\mathbf{x}^{(1)}$ into noisy states $\mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(S)}$, where $\mathbf{x}^{(S)}$ is a fully masked sequence. This process is specified by a fixed

corruption distribution $q(\mathbf{x}^{(2:S)} \mid \mathbf{x}^{(1)}) = \prod_{s=2}^{S} q(\mathbf{x}^{(s)} \mid \mathbf{x}^{(s-1)})$, where $q(\mathbf{x}^{(s)} \mid \mathbf{x}^{(s-1)})$ masks tokens according to a predefined noise schedule.

The reverse process is parameterized as $p_\theta(\mathbf{x}^{(1:S)}) = p(\mathbf{x}^{(S)}) \prod_{s=2}^{S} p_\theta(\mathbf{x}^{(s-1)} \mid \mathbf{x}^{(s)})$. At each reverse step, the model samples predictions for the whole sequence from $p_\theta(\mathbf{x}^{(1)} \mid \mathbf{x}^{(s)})$, but only replaces the currently masked positions with the predicted tokens. The next state $\mathbf{x}^{(s-1)}$ is then constructed by remasking a fraction ρ_s of the newly predicted positions according to a remasking strategy, such as random or low-confidence remasking (Nie et al., 2025b). In practice, given a prompt, the reverse denoising process starts from a partially masked state $\tilde{\mathbf{x}}^{(S)}$, obtained by placing the prompt as a fixed unmasked prefix in $\mathbf{x}^{(S)}$ while keeping the remaining positions masked.

Problem Setup Given a dataset of I prompts with safety labels $y^{(i)} \in \{0, 1\}$ indicating whether the i-th prompt is safe (0) or unsafe (1), the D-LLM produces a hidden representation $\mathbf{H}^{(i)}_{\text{raw}} \in \mathbb{R}^{D \times L \times S}$ for the i-th prompt at a particular layer, where D, L, and S denote the hidden dimension, sequence length, and number of denoising steps respectively. Since D-LLMs adopt bidirectional attention, safety-relevant information is distributed across tokens. We therefore aggregate over the sequence dimension via mean pooling, yielding a step-wise representation matrix $\mathbf{H}^{(i)} = [\mathbf{h}^{(i)}_1, \ldots, \mathbf{h}^{(i)}_S] \in \mathbb{R}^{D \times S}$, where $\mathbf{h}^{(i)}_s \in \mathbb{R}^D$ denotes the aggregated hidden state at step s. The dataset of I representation matrices and their labels is denoted by $\mathcal{D} = \{\mathbf{H}^{(i)}, y^{(i)}\}_{i=1}^{I}$. A safety probe f is learned by minimizing the empirical cross-entropy loss:

$$\min_f \frac{1}{I} \sum_{i=1}^{I} \mathcal{L}\Big(y^{(i)}, f(\mathbf{H}^{(i)})\Big). \tag{1}$$

We instantiate f as a linear probe, as its lightweight design makes it suitable for always-on monitoring while maintaining strong interpretability (Hewitt & Liang, 2019).

3.2 Multi-step as Useful Signal: Beyond Single-Step Safety Probing

The first question to consider in optimizing a probe to monitor D-LLMs is the choice of $\mathbf{H}$ due to the richer multi-step hidden representations compared to AR-LLMs. Specifically, should a probe rely on a single-step representation $\mathbf{h}_s$, or does the full trajectory $\mathbf{H}$ carry additional safety-relevant signal?

To answer this, we compare two monitoring settings that differ in how $\mathbf{H}$ is used. **(1) Single-step probing:** The probe operates on a single denoising step. We use the final-step representation $\mathbf{h}_1$, since it is the most refined hidden state before generation terminates, and train and test the probe on $\mathbf{h}_1$. **(2) Multi-step probing:** The probe operates on the full denoising trajectory $\mathbf{H}$. To keep training cost comparable to the single-step setting, we train on the temporal-mean representation $\bar{\mathbf{h}} = \frac{1}{S}\sum_{s=1}^{S} \mathbf{h}_s$ rather than treating each denoising step as a separate training example. At test time, we consider two trajectory-level readouts: *(a) Mean*, which evaluates the probe directly on $\bar{\mathbf{h}}$, i.e., $f(\bar{\mathbf{h}})$; and *(b) Majority Vote (MV)*, which applies the same probe to each individual step and aggregates via majority voting:

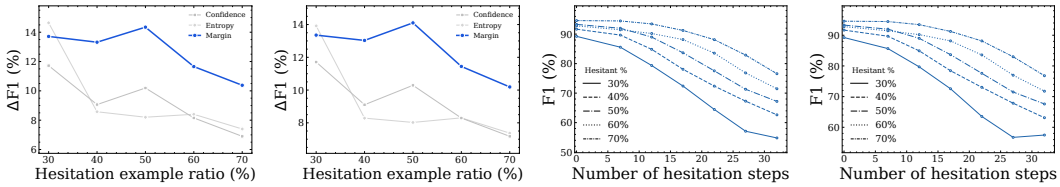
$$\hat{y}^{(i)} = \text{Majority}\left(f(\mathbf{h}^{(i)}_1), \ldots, f(\mathbf{h}^{(i)}_S)\right). \tag{2}$$

Both multi-step readouts use the same probe setting and number of training samples as the single-step setting, enabling a controlled comparison of trajectory utilization.

We denote the three settings by **LP (Last Step)**, **LP (Mean)**, and **LP (MV)** (Appendix C.2). As shown in Tables 1 and 2, both multi-step readouts achieve higher Acc and F1 than the single-step baseline on most models, indicating that intermediate denoising steps carry safety-relevant information that the final step alone misses. We therefore use the full trajectory $\mathbf{H}$ for all subsequent analysis.

3.3 Hesitation Steps as Difficulty Signal: Separating Easy and Hard Samples

Despite linear probes' low cost and interpretable form, they have limited expressivity and may fail to capture non-linear structure in representations (Belinkov, 2022; White et al., 2021), leading to misclassification on "harder" samples. We therefore seek signals that reflect when a linear probe



(a) ΔF1 - LP (MV) (b) ΔF1 - LP (Mean) (c) F1 - LP (MV) (d) F1 - LP (Mean)

Figure 2: **(a)(b):** F1 differences across probing methods under varying ratios of hesitation examples. **(c)(d):** F1 score as a function of the number of hesitation steps under different threshold values τ.

is likely to struggle. Inspired by recent findings that intermediate D-LLM responses can fluctuate between correct and incorrect answers during mathematical reasoning (Wang et al., 2026b; Li et al., 2026a), we hypothesize that analogous instability may occur in the safety context: the model may exhibit uncertainty in its safety decisions across the denoising trajectory. Accordingly, trajectories may be characterized as *stable*, where the model remains consistent across steps, and *hesitant*, where high uncertainty arises at intermediate steps.

Hesitation Characterization To verify this hypothesis, we explore two types of signals that may inform on such hesitation. **(1) Probe-extrinsic signals** quantify uncertainty from the model's predicted token distribution, independently of the probe. Let $\mathcal{R}$ denote the set of sequence positions and $p_s^{(r,v)}$ the predicted probability of token v at position r and denoising step s. We define the step-wise entropy score E_s and confidence score C_s as

$$E_s = -|\mathcal{R}|^{-1} \sum_{r\in\mathcal{R}} \sum_{v} p_s^{(r,v)} \log p_s^{(r,v)}, \quad C_s = |\mathcal{R}|^{-1} \sum_{r\in\mathcal{R}} \max_{v} p_s^{(r,v)}. \tag{3}$$

A step is flagged as hesitant if $E_s \geq \tau_E$ or $C_s \leq \tau_C$ for thresholds τ_E and τ_C. A trajectory is considered hesitant if it contains at least one hesitation step. **(2) Probe-intrinsic signals** measure uncertainty with respect to the probe's decision boundary. Applying the linear probe from Section 3.2 to each $\mathbf{h}_s$ yields a step-wise logit. Let d_s denote the signed margin to the decision boundary. A step is flagged as hesitant if $|d_s| < \tau$ for a margin threshold τ. A trajectory is considered hesitant if at least one of its steps is hesitant. We then compare probe performance on the stable and hesitant subsets characterized by these two kinds of signals across a range of thresholds. For a fair comparison, the thresholds are chosen to produce comparable split ratios between the two subsets. As shown in Figs. 2a and 2b, probe performance differs substantially: *hesitation trajectories yield markedly lower F1 scores than stable ones, confirming that trajectory hesitation is predictive of classification difficulty*. Among the signals evaluated, *the probe margin produces the largest performance gap, indicating that probe-intrinsic signals most effectively identify hard trajectories.* In Appendix E.1, we further conduct a dynamical analysis to understand the underlying mechanism.

Hesitation Severity However, this τ-induced criterion only captures whether a trajectory exhibits hesitation, not its extent. We therefore define hesitation severity $n_\tau = \sum_{s=1}^{S} \mathbb{I}[|d_s| < \tau]$, under which the original criterion is simply $\mathbb{I}[n_\tau \geq 1]$. Comparing probe F1 under the two partitions (Figs. 2a to 2d) yields two observations. **(1) n_τ gives finer levels and a wider F1 gap.** The binary partition into stable ($n_\tau = 0$) and hesitant ($n_\tau \geq 1$) trajectories yields an F1 gap of 10–14 points under the margin signal (Figs. 2a and 2b). When we stratify by n_τ instead, probe F1 decreases monotonically with n_τ, and the gap between the extreme buckets reaches ∼30 points at the 30% hesitant example ratio (Figs. 2c and 2d). **(2) τ over-flags easy trajectories.** Probe F1 on small-n_τ trajectories remains close to that on the stable subset, yet the binary criterion lumps them together with those genuinely hard for the probe (Figs. 2c and 2d). Defining n_{entropy} and $n_{\text{confidence}}$ analogously from probe-extrinsic signals, we find n_τ the most predictive of difficulty (Appendix E.2).

4 $\mathcal{D}^2$-MONITOR

4.1 OVERVIEW

Inspired by prior work on hierarchical monitoring in AR-LLMs (McKenzie et al., 2025; Cunningham et al., 2026; Oldfield et al., 2026) and by our finding that n_τ predicts base-probe failure (Section 3.3),

we propose $\mathcal{D}^2$-Monitor, a monitoring framework in which a low-complexity *base probe* scores every sample and a *router* escalates the hard ones to a high-complexity *advanced probe* (Fig. 1). We make two main design choices. First, whereas existing cascaded monitors escalate on a single uncertainty score from the first-stage probe (McKenzie et al., 2025; Cunningham et al., 2026), we escalate on n_τ, which accumulates evidence across denoising steps. Since n_τ is read off the base probe's per-step margins, the rule reduces to a single threshold test $n_\tau > \lambda$, where λ exposes a compute budget: larger λ escalates fewer samples, and performance remains strong even at low escalation rates (Appendix E.4). Second, hesitation steps tend to occur in a contiguous run rather than scattered across the trajectory (Appendix E.1), so we feed the advanced probe only the minimal span containing them, rather than all S steps; we ablate this choice in Appendix D.3.

4.2 Implementation

Given its lightweight architecture, strong performance on low-hesitation samples (Figs. 2c and 2d), and effectiveness at identifying estimation difficulty (Section 3.3), we adopt the linear probe as the low-complexity base probe. Our framework is flexible with respect to the choice of the high-complexity probe. In this work, we consider two variants with comparable parameter counts: (1) an MLP probe and (2) a temporal attention probe (TimeAttn) that aggregates hidden states within the hesitation window. Additional architectural details are provided in Appendix C.2.

Stage 1: Out-of-Fold Scoring and Hesitation Trajectories Collection We first identify which training trajectories hesitate. To obtain unbiased estimates, we score the training set out-of-fold (OOF): it is split into k folds, and each fold is scored by a linear probe trained on the remaining $k-1$, yielding margins $\{d_s^{(i)}\}_{s=1}^{S}$ for every training example i. Counting the steps with $|d_s^{(i)}| < \tau$ gives $n_\tau^{(i)}$, and we select trajectories with $n_\tau^{(i)} > 0$ for training the advanced probe.

Stage 2: Base and Advanced Probes Training We train the base linear probe on the full training set, which is used at test time to compute step-wise margins and derive the hesitation severity n_τ. For the high-complexity probe, we define the hesitation window $\mathcal{W}^{(i)}$ as the minimal contiguous span containing the hesitation steps of trajectory i. It is empty when the trajectory does not hesitate. We train the advanced probe f on the non-empty windows:

$$\min_{f} \sum_{i:\, n_\tau^{(i)} > 0} \mathcal{L}\Big(y^{(i)},\ f(\{\mathbf{h}_s^{(i)}\}_{s \in \mathcal{W}^{(i)}})\Big), \tag{4}$$

Stage 3: Cascade Detection At test time, the base probe computes the signed margin d_s at every denoising step, and n_τ counts the steps with $|d_s| < \tau$. If $n_\tau \le \lambda$, the base probe classifies the example by majority voting: $\hat{y} = \text{sign}\big(\sum_{s=1}^{S} \text{sign}(d_s)\big)$. Otherwise, the hesitation window $\mathcal{W}$ is extracted and passed to the advanced probe, which produces the final prediction. Details on the hyperparameters τ and λ are provided in Appendix C.1.

5 Experiment

5.1 Experiment Setup

Datasets We evaluate on three safety datasets. **WildGuardMix** (Han et al., 2024) consists of 86.8k training prompts and 1.7k test prompts, each labeled as harmful or unharmful. The dataset includes many adversarially designed inputs, posing a challenging benchmark for safety evaluation. **ToxicChat** (Lin et al., 2023) contains 5.08k training and 5.08k test prompts collected from real user-AI interactions, each annotated with a binary toxicity label (1 for toxic, 0 otherwise). **OpenAI-Moderation** (Markov et al., 2022) consists of 1.68k prompts annotated across eight moderation categories such as hate speech, violence, and self-harm; a prompt is labeled as unsafe if any category is flagged, and safe otherwise. For *intra-dataset* evaluation, we train and test on WildGuardMix and ToxicChat separately. For *cross-dataset* evaluation, we train on WildGuardMix and test on both ToxicChat and OpenAI-Moderation.

Table 1: **Intra-dataset performance on the test set.** Monitors are trained and tested on *WildGuardMix*. Mean±std over seeds 0, 1, 2. Best results are in **bold**, and second-best are underlined.

Method	E[P]	LLaDA-8B-Base Acc	LLaDA-8B-Base F1	LLaDA-8B-Instruct Acc	LLaDA-8B-Instruct F1	LLaDA-1.5 Acc	LLaDA-1.5 F1	E[P]	LLaDA-2.0-mini Acc	LLaDA-2.0-mini F1
Single-step methods										
LP (Last Step)	4×10^{-3}M	84.2±0.2	83.7±0.2	87.0±0.3	86.6±0.3	88.2±0.1	87.9±0.1	2×10^{-3}M	77.4±0.2	76.1±0.2
MLP (Last Step)	1.05M	85.6±0.3	85.2±0.4	87.2±0.5	86.9±0.5	87.3±0.0	87.0±0.0	0.52M	78.5±0.1	77.6±0.1
Full-trajectory methods										
LP (MV)	4×10^{-3}M	85.2±0.7	84.8±0.7	87.8±0.5	87.6±0.5	87.4±0.1	87.2±0.1	2×10^{-3}M	80.1±0.3	79.1±0.2
LP (Mean)	4×10^{-3}M	85.7±1.0	85.3±1.0	87.9±0.4	87.6±0.4	87.7±0.1	87.4±0.1	2×10^{-3}M	80.3±0.2	79.4±0.2
MLP (MV)	1.05M	86.8±0.4	86.4±0.4	88.0±0.4	87.7±0.4	87.8±0.1	87.5±0.0	0.52M	80.9±0.1	80.0±0.0
MLP (Mean)	1.05M	87.5±0.3	87.1±0.4	88.1±0.4	87.8±0.4	88.1±0.0	87.8±0.0	0.52M	81.2±0.1	80.4±0.0
TimeAttn (Kramár et al., 2026)	1.59M	87.6±0.5	87.2±0.6	88.1±0.2	87.8±0.2	88.2±0.4	88.0±0.5	0.80M	81.7±0.4	80.7±0.2
LSTM (Damirchi et al., 2026)	2.57M	87.0±0.4	86.5±0.4	88.2±0.4	87.9±0.4	88.1±0.4	87.9±0.4	1.51M	81.7±0.6	80.8±0.5
$\mathcal{D}^2$-MLP (Ours)	≤0.34M	88.5±0.2	88.2±0.3	**89.8±0.1**	**89.5±0.1**	89.7±0.1	89.4±0.1	0.06M	84.0±0.1	83.4±0.1
$\mathcal{D}^2$-TimeAttn (Ours)	≤0.55M	**89.4±0.3**	**89.1±0.4**	89.6±0.1	89.3±0.1	**89.7±0.2**	**89.4±0.2**	0.12M	**84.2±0.3**	**83.5±0.3**

Table 2: **Intra-dataset performance on the test set.** Monitors are trained and tested on *ToxicChat*. Mean±std over seeds 0, 1, 2. Best results are in **bold**, and second-best are underlined.

Method	E[P]	LLaDA-8B-Base Acc	LLaDA-8B-Base F1	LLaDA-8B-Instruct Acc	LLaDA-8B-Instruct F1	LLaDA-1.5 Acc	LLaDA-1.5 F1	E[P]	LLaDA-2.0-mini Acc	LLaDA-2.0-mini F1
Single-step methods										
LP (Last Step)	4×10^{-3}M	89.7±0.8	74.6±1.3	90.3±0.2	76.0±0.4	89.5±1.2	75.0±1.5	2×10^{-3}M	93.6±0.3	79.8±0.5
MLP (Last Step)	1.05M	95.6±0.1	79.9±0.4	96.6±0.0	86.9±0.0	96.3±0.2	85.3±0.8	0.52M	96.1±0.1	82.1±0.0
Full-trajectory methods										
LP (MV)	4×10^{-3}M	89.7±1.5	75.1±1.8	90.6±1.3	76.6±1.6	90.0±1.4	75.9±1.9	2×10^{-3}M	93.4±0.3	80.0±0.4
LP (Mean)	4×10^{-3}M	90.5±1.3	76.3±1.7	90.9±1.4	77.2±1.8	90.1±1.4	76.1±2.0	2×10^{-3}M	93.6±0.2	80.8±0.3
MLP (MV)	1.05M	96.1±0.1	83.8±0.2	96.7±0.1	87.1±0.5	96.8±0.1	87.7±0.0	0.52M	95.9±0.4	81.5±0.1
MLP (Mean)	1.05M	96.1±0.0	84.3±0.1	96.7±0.0	87.1±0.2	96.8±0.2	87.8±0.1	0.52M	96.3±0.3	83.9±0.3
TimeAttn (Kramár et al., 2026)	1.59M	96.6±0.0	86.0±0.0	96.7±0.3	87.3±0.9	96.6±0.1	86.8±1.0	0.80M	95.8±0.2	80.3±1.3
LSTM (Damirchi et al., 2026)	2.57M	96.4±0.2	86.3±0.1	97.0±0.1	87.6±0.2	96.9±0.1	87.8±0.2	1.51M	95.9±0.2	81.6±0.2
$\mathcal{D}^2$-MLP (Ours)	≤0.18M	96.7±0.2	87.9±0.6	**97.4±0.0**	**90.3±0.2**	97.3±0.0	89.9±0.1	0.06M	96.9±0.1	87.2±0.2
$\mathcal{D}^2$-TimeAttn (Ours)	≤0.37M	**96.9±0.0**	**88.3±0.3**	97.4±0.1	90.1±0.2	**97.5±0.0**	**90.4±0.1**	0.08M	**96.9±0.1**	**87.4±0.3**

Models We use four open-source D-LLMs for our experiments: `LLaDA-8B-Base` (Nie et al., 2025b), `LLaDA-8B-Instruct` (Nie et al., 2025b), `LLaDA-1.5-8B` (Zhu et al., 2025), and `LLaDA-2.0-mini-16B` (Bie et al., 2025).

Baselines We compare our method against eight baselines, organized into two categories. **(1) Single-step methods** use only the hidden state from the last denoising step $\mathbf{h}_1$ for both training and prediction. **(2) Multi-step methods** leverage all S denoising steps. For mean-based approaches, LP and MLP are trained on the temporal mean $\bar{\mathbf{h}} = \frac{1}{S}\sum_s \mathbf{h}_s$. At test time, two prediction strategies are considered: the *Mean* variant predicts directly from $\bar{\mathbf{h}}$, while the *MV* variant applies the trained probe to each step independently and takes a majority vote. This yields four baselines: LP (Mean), LP (MV), MLP (Mean), and MLP (MV). For sequence-based approaches, TimeAttn (Kramár et al., 2026) and LSTM (Damirchi et al., 2026) operate directly on the full ordered sequence $(\mathbf{h}_1, \ldots, \mathbf{h}_S)$. TimeAttn uses a temporal attention mechanism to aggregate hidden states, while LSTM encodes the sequence with a recurrent model. Experiment details are provided in Appendix C.

Evaluation Metrics We report three metrics. **Accuracy (Acc)** measures the fraction of correctly classified prompts. **Macro-F1** is the unweighted mean of the F1 scores for the safe and unsafe classes, each computed as the harmonic mean of that class's precision and recall. **E[P]** measures the expected number of parameters used per example at test time, capturing the effective monitor size under cascade routing, which is a proxy for runtime memory cost. For our method, $\mathrm{E}[P] = |\theta_{\mathrm{LP}}| + \rho \cdot |\theta_g|$, where $|\theta_{\mathrm{LP}}|$ and $|\theta_g|$ are the parameter counts of the linear probe and the advanced probe respectively, and ρ is the fraction of examples routed to the advanced probe. Additional evaluation metrics, including F2-score, false rejection rate (FRR), inference time, and FLOPs, are reported in Appendix D.1.

5.2 Main Results

Tables 1 to 3 summarize the intra-dataset and cross-dataset performance. $\mathcal{D}^2$-Monitor outperforms all baselines in both accuracy and F1 score in every model–dataset setting. Relative to the strongest baseline, the margin is $+1.4$ to $+3.5$ F1 intra-dataset and $+2.7$ to $+4.3$ F1 under cross-dataset transfer. More importantly, $\mathcal{D}^2$-Monitor achieves state-of-the-art performance while incurring substantially lower computational cost than non-linear baselines. As shown in Fig. 1, our method provides the best

Table 3: **Cross-dataset generalization.** Monitors are trained on *WildGuardMix* and tested on *ToxicChat* and *OpenAI-Moderation*. Mean±std over seeds 0, 1, 2. Best results are in **bold**, and second-best are underlined.

		LLaDA-8B-Base				LLaDA-8B-Instruct			
		ToxicChat		OpenAI-Moderation		ToxicChat		OpenAI-Moderation	
Method	**E[P]**	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Single-step methods									
LP (Last Step)	4×10^{-3}M	80.1±1.0	62.5±0.5	61.6±0.5	61.4±0.5	84.2±2.0	66.5±2.2	66.9±0.8	66.4±0.7
MLP (Last Step)	1.05M	82.8±1.3	65.1±1.3	61.3±1.5	61.2±1.4	85.1±2.6	67.8±2.6	65.3±2.0	65.0±1.9
Full-trajectory methods									
LP (MV)	4×10^{-3}M	83.5±0.7	65.8±1.3	59.6±3.6	59.5±3.5	84.6±0.7	67.1±0.7	63.8±0.8	63.5±0.8
LP (Mean)	4×10^{-3}M	84.2±1.0	66.5±1.4	60.0±3.6	59.9±3.5	85.3±0.4	67.8±0.5	64.3±1.0	63.9±1.0
MLP (MV)	1.05M	79.4±1.7	62.3±1.3	60.9±2.2	60.8±2.1	86.9±1.1	70.5±1.3	62.4±0.7	62.2±0.6
MLP (Mean)	1.05M	82.5±1.5	65.4±1.3	62.2±2.1	62.1±2.0	87.3±1.0	71.1±1.1	63.0±0.8	62.7±0.7
TimeAttn (Kramár et al., 2026)	1.59M	84.3±1.9	67.7±1.5	66.0±2.1	65.4±1.9	88.2±0.8	72.2±1.1	67.5±1.9	67.1±1.8
LSTM (Damirchi et al., 2026)	2.57M	83.2±2.0	66.1±3.0	66.1±2.3	65.5±2.0	88.1±1.8	72.2±2.1	68.6±2.8	67.8±2.2
$\mathcal{D}^2$-MLP (Ours)	≤0.68M	**88.4±0.7**	**72.0±0.2**	67.2±0.8	66.8±0.7	89.2±0.2	73.2±0.1	69.6±0.7	68.9±0.6
$\mathcal{D}^2$-TimeAttn (Ours)	≤0.93M	87.7±0.4	71.6±0.8	**69.4±1.1**	**68.6±0.8**	**90.6±0.6**	**75.3±0.7**	**71.3±0.7**	**70.5±0.7**

effectiveness-efficiency trade-off overall. In contrast, sequence-based baselines such as LSTM and TimeAttn incur significantly higher computational cost without delivering better performance. We attribute the advantage of $\mathcal{D}^2$-Monitor to its hesitation-aware routing mechanism, which accurately directs "hard" samples to the advanced probe for further processing (Section 5.3). Moreover, hesitation reflects a form of *model-intrinsic* uncertainty rather than dataset-specific patterns. Consequently, the performance gains can generalize across datasets.

5.3 Analysis

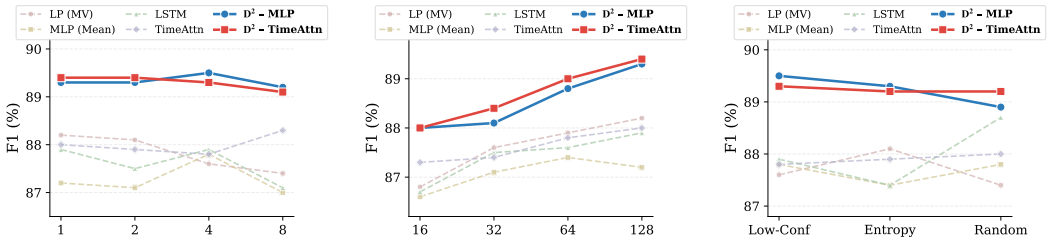
Efficiency-Effectiveness Tradeoff Model providers can run safety probes alongside the D-LLM with negligible overhead, but user-side deployment operates under a tighter budget. Raising λ lowers the expected parameter count $\mathrm{E}[P]$, which reflects runtime memory cost, at the risk of missing difficult cases. In practice the trade-off is mild: sweeping λ at test time without retraining changes F1 by less than 0.8 points on ToxicChat (Fig. 6). As shown in Fig. 1, $\mathcal{D}^2$-Monitor achieves the best F1 while using fewer parameters than most baselines.

Comparison with LLM-Based Monitors To situate $\mathcal{D}^2$-Monitor against the LLMs-as-monitors paradigm, we fine-tune Llama-Guard-3-1B (Llama Team, 2024) separately on WildGuardMix and ToxicChat and evaluate it on each. It attains 89.9 and 90.4 F1 respectively (Table 6). $\mathcal{D}^2$-Monitor achieves comparable performance on most models using fewer than 0.05% parameters. Under transfer the comparison favors the LLM monitor: aligned on WildGuardMix, it reaches 80.4 F1 on ToxicChat and 74.5 on OpenAI-Moderation, against 71.6–75.3 and 66.8–70.5 for $\mathcal{D}^2$-Monitor (Table 7). Note that every probe-based method in Table 3 degrades sharply under distribution shift, so this is not specific to our design. We regard this as an open problem for representation-based monitors (Wang et al., 2026a; Orgad et al., 2025).

Robustness to Generation Length and Step Length In practice, D-LLMs can be deployed with varying generation lengths and step lengths. Training a separate monitor for each configuration is costly and impractical. A desirable property is therefore to train once under a single configuration and generalize to others. To evaluate this, we train all methods with generation length 128 and step length 4, and test under varying settings without retraining. We vary the step length $L_S \in \{1, 2, 4, 8\}$ with generation length fixed at 128 (Fig. 3a), and the generation length $L \in \{16, 32, 64, 128\}$ with step length fixed at 1 (Fig. 3b). $\mathcal{D}^2$-Monitor consistently outperforms all baselines across both axes of variation, confirming that our method transfers reliably across decoding configurations.

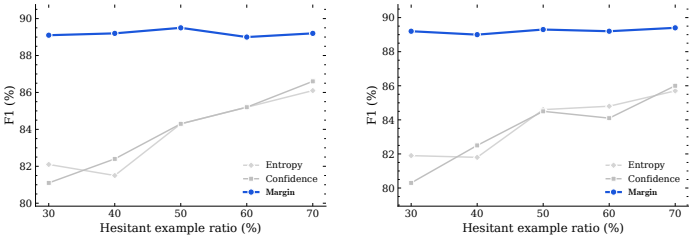
Robustness to Remasking Strategy We evaluate the impact of different remasking strategies on detection performance. All methods are trained under low-confidence remasking and tested under three strategies: low-confidence, entropy, and random. As shown in Fig. 3c, $\mathcal{D}^2$-Monitor maintains consistent superiority across all strategies.

Ablation on Routing Signal We replace the margin-based criterion with entropy- and confidence-based ones on ToxicChat, retraining the advanced probe on each signal's hesitation subset and holding



(a) (b) (c)

Figure 3: **(a)** Performance with different step lengths with generation length fixed at 128. **(b)** Performance with different generation lengths with step length fixed at 1. **(c)** Performance under different remasking strategies. All results are reported as F1 using LLaDA-8B-Instruct on WildGuardMix.



(a) D^2–MLP (b) D^2–TimeAttn

Figure 4: Ablation on routing signals under different hesitant ratios. Margin-based routing consistently outperforms entropy and confidence, and shows greater robustness to threshold selection.

the rest of the pipeline fixed. Across both $\mathcal{D}^2$-MLP and $\mathcal{D}^2$-TimeAttn at hesitation ratios from 30% to 70%, the margin signal gives the highest and most stable F1 (Fig. 4), indicating that probe-intrinsic uncertainty routes better than probe-extrinsic uncertainty. At a 50% hesitation ratio, the margin signal produces the largest accuracy gap between the base and advanced probes on the routed subset (13.5% / 12.4%), well above entropy (7.2% / 7.4%) and confidence (7.1% / 7.3%), so it isolates the samples that gain most from escalation. That subset is also disproportionately adversarial: 64–88% on *WildGuardMix* against a dataset-wide baseline of 47% (Section E.3), so margin-based routing concentrates the advanced probe on likely attacks rather than on generically hard inputs.

Effect of Curation and Routing We ablate the two ingredients separately on LLaDA-8B-Instruct (Table 4): curation, which trains the advanced probe on hesitation windows, and routing, which applies it only to escalated samples. Both help on their own, and combining them brings a further improvement. Averaged over the two datasets, routing alone adds $+0.9/+1.0$ F1 for MLP/TimeAttn, curation $+1.0/+0.5$, and the combination $+2.5/+2.2$. Routing also cuts $\mathrm{E}[P]$ by 4–7×. In every case, curation on top of routing is worth $+1.0$ to $+2.0$ F1 at unchanged $\mathrm{E}[P]$, and routing on top of curation is 4–7× cheaper and more accurate.

6 Conclusion

We showed that the denoising trajectory of a D-LLM carries a usable difficulty signal: on some inputs the intermediate states repeatedly fall within a small margin of a linear probe's decision boundary. The number of these hesitation steps n_τ predicts when the probe will fail. Acting on this signal, $\mathcal{D}^2$-Monitor escalates samples whose n_τ exceeds a threshold to a more expressive probe that reads only the hesitation window. On three safety datasets across four D-LLMs, it achieves state-of-the-art performance both in-domain and cross-dataset, with the best effectiveness–efficiency trade-off among eight baselines and at most 0.93M expected parameters.

Limitations Although we expect these results to extend to larger models (Bie et al., 2025; 2026), our experiments are currently limited to D-LLMs of up to 16B parameters due to computational constraints. Our method is stable across generation configurations without retraining (Fig. 3), but calibration under shifts in checkpoint or safety policy remains an open problem (Ovadia et al., 2019). Beyond these empirical gaps, a theoretical account of why hesitation arises during denoising would also be interesting. Appendix A expands on these limitations.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experiments, to assist in implementing methods, to assist with translation, and to help interpret results. We have not used generative AI tools to generate synthetic data sets, to develop the theoretical models or conceptual framework of this work, to propose or refine hypotheses, to clean or reformat datasets, or to support qualitative and thematic data analysis; formulating mathematical claims, providing critical ingredients for proving mathematical claims, and assisting in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools to edit this paper to improve readability, to suggest a structure for the paper, and to create or edit software code. All AI-assisted work was reviewed by the authors: AI-assisted code was read and tested for correctness, and AI-assisted text was revised before inclusion. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work involves no human subjects and releases no new data or models; all experiments use publicly available safety benchmarks under their respective licenses (Appendix B). Two risks are worth noting. First, like any safety classifier, our monitor can falsely block benign prompts; we report false rejection rates alongside detection performance (Appendix D.1), where $\mathcal{D}^2$-Monitor attains the lowest rate among all methods on the four D-LLMs we evaluate. Second, because routing depends on hesitation severity, an adversary who crafts inputs that keep hesitation low could remain on the lightweight path and evade the advanced probe.

REPRODUCIBILITY STATEMENT

All reported numbers are means over three random seeds (0, 1, and 2), reported with standard deviations. The datasets, splits, models, and evaluation protocol are specified in Appendix C; probe architectures and their exact dimensions in Appendix C.2; and the procedure for selecting the test-time hyperparameters τ and λ in Appendix C.1.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Anthropic. Disrupting the first reported ai-orchestrated cyber espionage campaign. https://www.anthropic.com/news/disrupting-AI-espionage, 2025.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 04 2022. ISSN 0891-2017. doi: 10.1162/coli_a_00422. URL https://doi.org/10.1162/coli_a_00422.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, et al. Llada2.0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, et al. Llada2.1: Speeding up text diffusion via token editing. *arXiv preprint arXiv:2602.08676*, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yupeng Chen, Junchi Yu, Aoxi Liu, Philip Torr, and Adel Bibi. The alignment curse: Cross-modality jailbreak transfer in omni-models. *arXiv preprint arXiv:2602.02557*, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Hoagy Cunningham, Jerry Wei, Zihan Wang, Andrew Persic, Alwin Peng, Jordan Abderrachid, Raj Agarwal, Bobby Chen, Austin Cohen, Andy Dau, et al. Constitutional classifiers++: Efficient production-grade defenses against universal jailbreaks. *arXiv preprint arXiv:2601.04603*, 2026.

Hamed Damirchi, Ignacio Meza De la Jara, Ehsan Abbasnejad, Afshar Shamsi, Zhen Zhang, and Javen Qinfeng Shi. Truth as a trajectory: What internal representations reveal about large language model reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44774–44790, 2026.

Raghavv Goel, Risheek Garrepalli, Sudhanshu Agrawal, Christopher Lott, Mingu Lee, and Fatih Porikli. Skip to the good part: Representation structure & inference-time layer skipping in diffusion vs autoregressive LLM. In *ICLR 2026 2nd Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2026. URL https://openreview.net/forum?id=PoKFgcta4N.

Nicholas Goldowsky-Dill, Bilal Chughtai, Stefan Heimersheim, and Marius Hobbhahn. Detecting strategic deception with linear probes. In *International Conference on Machine Learning*, pp. 19755–19786. PMLR, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jiatong Han, Neil Band, Muhammed Razzak, Jannik Kossen, Tim GJ Rudner, and Yarin Gal. Simple factuality probes detect hallucinations in long-form natural language generation. *Findings of the Association for Computational Linguistics: EMNLP*, pp. 16209–16226, 2025.

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

Zeyuan He, Yupeng Chen, Lang Lin, Yihan Wang, Shenxu Chang, Eric Sommerlade, Philip Torr, Junchi Yu, Adel Bibi, and Jialin Yu. Safer by diffusion, broken by context: Diffusion llm's safety blessing and its failure mode. *arXiv preprint arXiv:2602.00388*, 2026.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)*, pp. 2733–2743, 2019.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Zhanhao Hu, Julien Piet, Geng Zhao, Jiantao Jiao, and David Wagner. Toxicity detection for free. *Advances in Neural Information Processing Systems*, 37:17518–17540, 2024.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, et al. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*, 2023.

Inception Labs. Introducing mercury 2. https://www.inceptionlabs.ai/blog/introducing-mercury-2, 2026.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Wonje Jeung, Sangyeon Yoon, Yoonjun Cho, Dongjae Jeon, Sangwoo Shin, Hyesoo Hong, and Albert No. A2d: Any-order, any-step safety alignment for diffusion language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=URTnuyQJI1.

Difan Jiao, Yilun Liu, Ye Yuan, Zhenwei Tang, Linfeng Du, Haolun Wu, and Ashton Anderson. Llm safety from within: Detecting harmful content with internal representations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on machine learning*, pp. 5637–5664. PMLR, 2021.

János Kramár, Joshua Engels, Zheng Wang, Bilal Chughtai, Rohin Shah, Neel Nanda, and Arthur Conmy. Building production-ready probes for gemini. *arXiv preprint arXiv:2601.11516*, 2026.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Yi Liang, Soroush Vosoughi, and Shiwei Liu. Diffusion language model knows the answer before it decodes. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=g88nt4ieTG.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

Zherui Li, Zheng Nie, Zhenhong Zhou, Yue Liu, Yitong Zhang, Yu Cheng, Qingsong Wen, Kun Wang, Yufei Guo, and Jiaheng Zhang. Diffuguard: How intrinsic safety is lost and found in diffusion large language models. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=zBPzxhso8M.

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, 2023.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7Jwpw4qKkb.

Yue Liu, Hongcheng Gao, Shengfang Zhai, Yufei He, Jun Xia, Zhengyu Hu, Yulin Chen, Xihong Yang, Jiaheng Zhang, Stan Z Li, et al. Guardreasoner: Towards reasoning-based llm safeguards. *arXiv preprint arXiv:2501.18492*, 2025.

AI @ Meta Llama Team. The llama 3 family of models. https://github.com/meta-llama/PurpleLlama/blob/main/Llama-Guard3/1B/MODEL_CARD.md, 2024.

Monte MacDiarmid, Timothy Maxwell, Nicholas Schiefer, Jesse Mu, Jared Kaplan, David Duvenaud, Sam Bowman, Alex Tamkin, Ethan Perez, Mrinank Sharma, et al. Simple probes can catch sleeper agents. *Anthropic Research Updates*, 2024.

Todor Markov, Chong Zhang, Sandhini Agarwal, Tyna Eloundou, Teddy Lee, Steven Adler, Angela Jiang, and Lilian Weng. A holistic approach to undesired content detection. *arXiv preprint arXiv:2208.03274*, 2022.

Alex McKenzie, Urja Pawar, Phil Blandfort, William Bankes, David Krueger, Ekdeep Singh Lubana, and Dmitrii Krasheninnikov. Detecting high-stakes interactions with activation probes. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=8YniJnJQ0P.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Milad Nasr, Nicholas Carlini, Chawin Sitawarin, Sander V Schulhoff, Jamie Hayes, Michael Ilie, Juliette Pluto, Shuang Song, Harsh Chaudhari, Ilia Shumailov, et al. The attacker moves second: Stronger adaptive attacks bypass defenses against llm jailbreaks and prompt injections. *arXiv preprint arXiv:2510.09023*, 2025.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=WNvvwK0tut.

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. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=KnqiC0znVF.

James Oldfield, Philip Torr, Ioannis Patras, Adel Bibi, and Fazl Barez. Beyond linear probes: Dynamic safety monitoring for language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=AGWa8whf92.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=KRnsX5Em3W.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

Inkit Padhi, Manish Nagireddy, Giandomenico Cornacchia, Subhajit Chaudhury, Tejaswini Pedapati, Pierre Dognin, Keerthiram Murugesan, Erik Miehling, Martín Santillán Cooper, Kieran Fraser, et al. Granite guardian. *arXiv preprint arXiv:2412.07724*, 2024.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *International Conference on Machine Learning*, pp. 39643–39666. PMLR, 2024.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. Pareto probing: Trading off accuracy for complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3138–3153, 2020.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

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

Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

Cheng Wang, Zeming Wei, Qin Liu, Wenxuan Zhou, and Muhao Chen. False sense of security: Why probing-based malicious input detection fails to generalize. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 26100–26113, 2026a.

Wen Wang, Bozhen Fang, Chenchen Jing, Yongliang Shen, Yangyi Shen, Qiuyu Wang, Hao Ouyang, Hao Chen, and Chunhua Shen. Time is a feature: Exploiting temporal dynamics in diffusion language models. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=HsB6CtagP7.

Xu Wang, Bingqing Jiang, Yu Wan, Baosong Yang, Lingpeng Kong, and Difan Zou. Dlm-scope: Mechanistic interpretability of diffusion language models via sparse autoencoders. *arXiv preprint arXiv:2602.05859*, 2026c.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Jennifer C White, Tiago Pimentel, Naomi Saphra, and Ryan Cotterell. A non-linear structural probe. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 132–138, 2021.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM computing surveys*, 56(4):1–39, 2023.

Wenjun Zeng, Dana Kurniawan, Ryan Mullins, Yuchi Liu, Tamoghna Saha, Dirichi Ike-Njoku, Jindong Gu, Yiwen Song, Cai Xu, Jingjing Zhou, et al. Shieldgemma 2: Robust and tractable image content moderation. *arXiv preprint arXiv:2504.01081*, 2025.

Yi Zeng, Hongpeng Lin, Jingwen Zhang, Diyi Yang, Ruoxi Jia, and Weiyan Shi. How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14322–14350, 2024.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, et al. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025.

A LIMITATION

Model Scale Our experiments cover four D-LLMs, from 8B to 16B parameters. Within this range the advantage of $\mathcal{D}^2$-Monitor does not shrink with model size. On LLaDA-2.0-mini, the largest model we evaluate, it improves F1 over the strongest baseline by 2.7 points on *WildGuardMix* and 3.5 points on *ToxicChat*, compared with 1.4 to 1.9 and 2.0 to 2.7 points on the 8B models. We therefore expect the approach to carry over to larger D-LLMs (Bie et al., 2025; 2026). Absolute F1 is lowest on this model for every method, which may be due to its narrower hidden states.

Calibration under Distribution Shift Calibration under distribution shift is a long-standing open problem (Ovadia et al., 2019; Koh et al., 2021), and we do not claim to solve it. A deployed D-LLM may run with a different generation length, step length, or remasking strategy, and its checkpoint and safety policy may change over time. A probe-based monitor, in contrast, is fitted once on representations from a single setting. However, two properties of $\mathcal{D}^2$-Monitor partially mitigate this. A probe trained under one generation configuration transfers to others without retraining (Figure 3), and performance is flat over a wide range of λ (Appendix E.4), so the routing threshold does not have to be re-tuned precisely.

Why Hesitation Arises We characterize hesitation empirically and show that it predicts when the base probe fails, but we do not explain why the intermediate states of some inputs repeatedly fall close to the decision boundary. Work on the internals of D-LLMs is still early. Recent studies describe how representations are organized across layers and denoising steps (Goel et al., 2026) and decompose them with sparse autoencoders (Wang et al., 2026c), but neither accounts for the dynamics of the probe margin along a trajectory. Why some trajectories hesitate, and whether this can be predicted before decoding starts, remains open. Answering either would put the routing rule on firmer ground, and we see it as an interesting direction for future work.

B BROADER IMPACTS, SAFEGUARDS, AND LICENSES

Broader Impacts Our work has clear positive societal impact: $\mathcal{D}^2$-Monitor provides a lightweight, always-on safety mechanism for D-LLMs that can detect harmful or adversarial inputs at low computational cost, which is particularly valuable for resource-constrained or edge deployment where running heavyweight LLM-as-monitor solutions is infeasible. By improving the practicality of safety monitoring, our work helps reduce the risk of D-LLMs being misused for harmful content generation. We also acknowledge potential negative impacts. (1) Like any safety classifier, the monitor may incur false rejections on benign inputs, potentially over-blocking legitimate queries; we report FRR explicitly in Appendix D.1 to make this trade-off transparent. (2) As discussed in Appendix A, hesitation-based routing could in principle be targeted by adaptive adversaries who craft prompts that suppress hesitation steps to evade the second-stage probe. We recommend that practitioners deploying $\mathcal{D}^2$-Monitor pair it with complementary defenses and continue to monitor for adversarial drift over time.

Safeguards The paper does not release any high-risk pretrained models, generative models, or scraped datasets. The artifacts produced by our work are lightweight safety probes ($\leq$ 0.93M parameters) trained on publicly available safety benchmarks; these probes are themselves a defensive mechanism intended to mitigate misuse rather than enable it. The base D-LLMs (LLaDA family) and safety datasets we build on are released by their original authors under their respective licenses, which we do not modify or redistribute.

Licenses for Existing Assets All existing assets used in this work are properly cited and used in accordance with their licenses.

Datasets WildGuardMix (Han et al., 2024) is released under the ODC-BY license; ToxicChat (Lin et al., 2023) is released under CC-BY-NC-4.0 and used for non-commercial research only; OpenAI-Moderation (Markov et al., 2022) is released under the MIT license.

Models LLaDA-8B-Base, LLaDA-8B-Instruct (Nie et al., 2025b), and LLaDA-1.5 (Zhu et al., 2025) are released under the MIT license; LLaDA-2.0-mini (Bie et al., 2025) is released under the Apache-2.0 license.

C Experiment Details

C.1 Hyperparameter Tuning

We tune all hyperparameters on a validation set derived from the training data. Specifically, we split the original training set into training and validation subsets with a ratio of 4:1. To ensure fair comparison across different probe architectures, we control model capacity by fixing key architectural dimensions (e.g., hidden size or hidden dimension) for each probe, rather than tuning them extensively. For each probe, we define the hyperparameter search space as follows:

LinearProbe:

- **Learning rate:** $\{3\text{e}{-3}, 1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}, 3\text{e}{-5}, 1\text{e}{-5}\}$
- **Weight decay:** $\{0, 1\text{e}{-6}, 1\text{e}{-5}, 1\text{e}{-4}, 1\text{e}{-3}\}$

MLP:

- **Learning rate:** $\{1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}, 3\text{e}{-5}, 1\text{e}{-5}\}$
- **Weight decay:** $\{0, 1\text{e}{-6}, 1\text{e}{-5}, 1\text{e}{-4}\}$
- **Dropout:** $\{0, 0.1, 0.2, 0.3\}$
- **Hidden dimension (K):** 256

TimeAttn:

- **Learning rate:** $\{1\text{e}{-3}, 3\text{e}{-4}, 1\text{e}{-4}, 3\text{e}{-5}, 1\text{e}{-5}\}$
- **Weight decay:** $\{0, 1\text{e}{-6}, 1\text{e}{-5}, 1\text{e}{-4}\}$
- **Dropout:** $\{0, 0.1, 0.2, 0.3\}$
- **MLP hidden dimension (K):** 256
- **Attention dimension (d_a):** 128

LSTM:

- **Projection dimension (d_p):** 512
- **Hidden size (d_h):** 128
- **Learning rate:** $\{1\text{e}{-5}, 1\text{e}{-4}, 1\text{e}{-3}\}$
- **Weight decay:** $\{0, 1\text{e}{-6}, 1\text{e}{-5}, 1\text{e}{-4}\}$
- **Dropout:** $\{0, 0.1, 0.2, 0.3\}$

Baseline Methods For all baseline methods, hyperparameters are selected via grid search on the validation set using the search ranges above. The best configuration is chosen based on validation performance, and the model is retrained on the full training set before evaluation on the test set.

Our Method $\mathcal{D}^2$-Monitor has two trained components whose hyperparameters cannot be selected independently. The linear probe used for out-of-fold (OOF) scoring on the training set determines which steps are labelled as hesitation, and therefore which examples the advanced probe is trained on; the advanced probe in turn determines final performance. We treat the two stages as a joint search: for each candidate configuration of the OOF probe we train the corresponding advanced probe (MLP or TimeAttn), and select the pair by validation performance.

Fairness and Evaluation Protocol All hyperparameter tuning is performed strictly on the validation set, and no test data is used at any stage of model selection.

Test-Time Hyperparameters Our method involves two test-time hyperparameters: the hesitation threshold τ and the routing parameter λ. Neither requires retraining, so both can be chosen by evaluating the trained monitor on the validation set.

Choice of τ The threshold τ controls a trade-off between coverage and purity of hesitation examples. A larger τ includes more training samples but may introduce stable (non-hesitant) steps, while a smaller τ yields cleaner hesitation signals at the cost of fewer training examples. To balance this trade-off, we select τ such that approximately 50% of the samples are identified as hesitation examples. As shown in Fig. 10, varying this proportion from 30% to 70% leads to only minor performance differences, indicating that our method is not sensitive to the exact choice of τ. For intra-dataset evaluation we fix τ at the value selected on the training data and reuse it at test time, since the data distributions are similar.

Selecting the operating point Given τ, we select λ in two steps. We first locate the configuration with the highest validation F1; we then restrict attention to all configurations whose validation F1 fall slightly below that maximum, and among these take the one with the smallest expected parameter count $\mathrm{E}[P]$. This selects the cheapest point on the validation plateau rather than the nominally best one, trading no measurable detection performance for lower test-time cost. The resulting operating point is then applied unchanged at test time.

Cross-dataset evaluation For cross-dataset transfer (trained on WildGuardMix and evaluated on ToxicChat and OpenAI-Moderation), the distribution shift means τ can no longer be carried over from training. We therefore run a grid search over (τ, λ) jointly on a held-out validation split of the target dataset. The operating point is then chosen from this grid by the same rule as before: among all configurations whose validation F1 fall slightly below that maximum, we take the one with the smallest $\mathrm{E}[P]$. As shown in Fig. 7, F1 varies little across the entire grid, so the reported results do not hinge on which configuration the search happens to return.

C.2 Probe Architectures

Here we provide specific details about the architectures of all probe baselines considered. Given the step-level trajectory $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_S] \in \mathbb{R}^{D \times S}$, each probe instantiates a particular $f \in \mathcal{F}$ that maps $\mathbf{H}$ to a scalar logit $s \in \mathbb{R}$ for binary classification. When presenting architectures below, we omit bias terms in the hidden layer(s) and output for brevity. All probes are trained with Adam with a batch size of 256 for 50 epochs on an NVIDIA A40 GPU with 48GB VRAM. Each individual probe performs its own grid search over the hyperparameters listed in Appendix C.1.

Normalization Before training, we normalize the activations using statistics computed from the training split. We use two normalization modes depending on the probe type. For *full-trajectory methods*, we apply *per-feature* normalization: the mean and standard deviation are computed over both the sample and step axes, yielding statistics of shape $[D]$. For *single-step methods*, we apply *per-step* normalization: statistics are computed per denoising step, yielding shape $[S, D]$.

Pooling Strategies For probes that require a fixed-length input (Linear Probe and MLP), we reduce the step dimension via one of three pooling strategies applied after normalization:

- **Mean pooling:** $\bar{\mathbf{h}} = \frac{1}{S}\sum_{s=1}^{S} \mathbf{h}_s \in \mathbb{R}^D$, used at both train and test time.
- **Last-step:** $\bar{\mathbf{h}} = \mathbf{h}_1 \in \mathbb{R}^D$, using only the final step's activation at both train and test time. Normalization statistics are computed from the last step only.
- **Majority vote (MV):** The probe is trained on mean-pooled features. At test time, each step $\mathbf{h}_s$ is classified independently, and the final prediction is determined by majority vote across S steps: $\hat{y} = \mathbf{1}\left[\sum_{s=1}^{S} \mathbf{1}[\hat{y}_s = 1] \geq S/2\right]$.

Linear Probe (LP) After pooling, the linear probe computes:

$$s = \mathbf{w}^\top \bar{\mathbf{h}}\,, \tag{5}$$

with $\mathbf{w} \in \mathbb{R}^D$. Combined with the three pooling strategies, this yields three variants: LP (Mean), LP (Last Step), and LP (MV).

MLP The two-layer MLP probe computes:

$$s = \mathbf{W}_{\text{out}} \,\text{Dropout}\big(\text{ReLU}\big(\mathbf{W}_{\text{in}}\bar{\mathbf{h}}\big)\big)\,, \tag{6}$$

with $\mathbf{W}_{\text{in}} \in \mathbb{R}^{K\times D}$ and $\mathbf{W}_{\text{out}} \in \mathbb{R}^{1\times K}$, where $K = 256$ is the hidden dimension. Analogously to the linear probe, this yields three variants: MLP (Mean), MLP (Last Step), and MLP (MV).

TimeAttn This probe operates directly on the full trajectory $\mathbf{H} \in \mathbb{R}^{D\times S}$ without step-level pooling. It first applies layer normalization, then computes attention weights over denoising steps via an additive attention mechanism:

$$\alpha_s = \frac{\exp\big(\mathbf{v}^\top \tanh(\mathbf{W}_a \mathbf{h}_s)\big)}{\sum_{s'=1}^{S} \exp(\mathbf{v}^\top \tanh(\mathbf{W}_a \mathbf{h}_{s'}))}\,, \tag{7}$$

where $\mathbf{W}_a \in \mathbb{R}^{d_a\times D}$ and $\mathbf{v} \in \mathbb{R}^{d_a}$ with attention dimension $d_a = 128$. The attended representation $\mathbf{c} = \sum_{s=1}^{S} \alpha_s \mathbf{h}_s \in \mathbb{R}^D$ is then classified via a two-layer MLP with layer normalization:

$$s = \mathbf{W}_2 \,\text{Dropout}(\text{ReLU}(\mathbf{W}_1 \,\text{LN}(\mathbf{c})))\,, \tag{8}$$

with $\mathbf{W}_1 \in \mathbb{R}^{K\times D}$, $\mathbf{W}_2 \in \mathbb{R}^{1\times K}$, and $K = 256$.

LSTM The LSTM probe first projects each step representation through a feedforward layer with layer normalization:

$$\mathbf{h}'_s = \text{GELU}(\mathbf{W}_{\text{proj}} \,\text{LN}(\mathbf{h}_s))\,, \tag{9}$$

with $\mathbf{W}_{\text{proj}} \in \mathbb{R}^{d_p\times D}$ and projection dimension $d_p = 512$. The projected sequence is then processed by a 2-layer unidirectional LSTM:

$$\tilde{\mathbf{h}}_1, \ldots, \tilde{\mathbf{h}}_S = \text{LSTM}(\mathbf{h}'_1, \ldots, \mathbf{h}'_S)\,, \tag{10}$$

with hidden size $d_h = 128$. The final hidden state is classified via a layer-normalized linear head:

$$s = \mathbf{w}_{\text{out}}^\top \text{LN}(\tilde{\mathbf{h}}_S)\,, \tag{11}$$

with $\mathbf{w}_{\text{out}} \in \mathbb{R}^{d_h}$.

D Additional Results

D.1 More Evaluation Metrics

Additional Metrics Beyond accuracy and macro-F1 reported in the main text, we evaluate four additional metrics. The **F2-score** is an instance of the F_β-score with $\beta = 2$, which weights recall β^2 times as much as precision, reflecting a preference in safety monitoring for catching harmful content at the cost of slightly more false alarms. **Precision** measures the fraction of inputs flagged as unsafe that are truly unsafe; high precision means the monitor rarely triggers on benign content, reducing unnecessary refusals. **Recall** measures the fraction of truly unsafe inputs correctly identified; high recall ensures that harmful prompts are not missed in safety-sensitive deployments. Finally, the **False Rejection Rate (FRR)** is the fraction of benign inputs incorrectly flagged as unsafe, i.e., $\text{FRR} = \text{FP}/(\text{FP} + \text{TN})$; unlike $1 - \text{Precision}$, which conditions on positive predictions, FRR conditions on the actual benign population and thus directly measures how often legitimate users are unnecessarily blocked (lower is better).

Tables 12–15 report all six metrics for each model. The results are consistent with the main findings: $\mathcal{D}^2$-MLP and $\mathcal{D}^2$-TimeAttn achieve the best or second-best performance across nearly all metrics and models, confirming that the gains observed in accuracy and F1 are not obtained at the expense of precision, recall balance, or false refusal rate.

Inference Time We report the inference time of different methods on WildGuardMix with 1,725 prompts. To isolate the overhead of the safety monitor, timing starts after the base model has completed generation and all hidden states have been extracted. For LP (Last Step), LP (MV), MLP (Last Step), and MLP (MV), we measure only the time required to process the last step. This is

because, in typical deployment, intermediate steps are processed online during generation (i.e., one step is processed as it is generated), and thus do not introduce additional latency after generation. For LP (Mean) and MLP (Mean), the reported time includes both the cost of computing the mean over all hidden states and the subsequent probe evaluation. In contrast, TimeAttn and LSTM operate on the full trajectory after generation, and their inference time includes processing all steps. For our method, the total inference time consists of two components: (i) the linear probe applied to the last step, and (ii) the additional computation of the advanced probe (e.g., MLP or TimeAttn) over the selected window when routing is triggered.

As shown in Table 11, D^2-Monitor substantially reduces inference cost relative to probes that score and aggregate every denoising step. Compared to MLP (Mean), D^2-MLP is 2.4×–6.6× faster across the four LLaDA models, and D^2-TimeAttn is 4.1×–5.1× faster than TimeAttn. D^2-MLP also stays within the same order of magnitude as single-step probes such as LP (Last Step) and MLP (Last Step), indicating that the routing logic itself adds negligible overhead.

FLOPs Computation We analytically compute the expected FLOPs per sample for each probe based on the architectures detailed in Appendix C.2. We adopt the standard convention that a multiply-add operation counts as 2 FLOPs. Throughout this section, we denote the number of denoising steps as S, the hidden dimension of the diffusion LLM as D, and the average length of the hesitation window for our cascade as S_{win}. The MLP hidden width K, attention dimension d_a, LSTM projection dimension d_p, and LSTM hidden size d_h follow the values specified in Appendix C.2. For brevity, we omit lower-order terms such as activation functions (ReLU, tanh, GELU), softmax, layer normalization, and bias additions, which contribute at most $O(D)$ or $O(S_{\text{win}}D)$ FLOPs and are negligible relative to the dominant matrix-vector products.

Linear Probe (LP) A single linear probe forward maps a D-dimensional vector to a scalar logit, requiring $2D$ FLOPs. The three pooling variants differ in how the trajectory $\mathbf{H} \in \mathbb{R}^{D\times S}$ is reduced to this input. **LP (Last Step)** uses only $\mathbf{h}_1$, with total cost $2D$. **LP (Mean)** sums S vectors of dimension D (SD FLOPs) and then applies one LP forward, with total cost $SD + 2D$. **LP (MV)** applies the LP independently at every denoising step and aggregates by majority vote, with total cost $S \cdot 2D = 2SD$.

MLP A single MLP forward consists of $\mathbf{W}_{\text{in}} \in \mathbb{R}^{K\times D}$ and $\mathbf{W}_{\text{out}} \in \mathbb{R}^{1\times K}$, costing $2DK + 2K \approx 2DK$ FLOPs. **MLP (Last Step)** applies one MLP forward on $\mathbf{h}_S$, with total cost $2DK$. **MLP (Mean)** performs mean pooling followed by one MLP forward, with total cost $SD + 2DK$. **MLP (MV)** applies an MLP forward at every denoising step, with total cost $S \cdot 2DK = 2SDK$.

TimeAttn TimeAttn operates on the full trajectory and consists of two main components: per-step attention scoring and a classifier head over the attended representation. The attention scoring computes $\mathbf{W}_a\mathbf{h}_s$ for each step ($\mathbf{W}_a \in \mathbb{R}^{d_a\times D}$), costing $S \cdot 2Dd_a$ FLOPs in total. The weighted sum $\sum_s \alpha_s\mathbf{h}_s$ contributes a further $2SD$ FLOPs. The classifier head is a 2-layer MLP applied to the resulting D-dimensional context vector, costing $2DK$ FLOPs. The total cost is therefore $2SDd_a + 2SD + 2DK \approx 2SDd_a + 2DK$, where the first term dominates for typical S and d_a.

LSTM The LSTM probe first projects each $\mathbf{h}_s$ from D to d_p via $\mathbf{W}_{\text{proj}} \in \mathbb{R}^{d_p\times D}$, costing $S \cdot 2Dd_p$ FLOPs across all steps. The projected sequence is then processed by a 2-layer LSTM. Each LSTM cell contains four gates, each with a matrix-vector product of size $d_h \times (d_{\text{in}} + d_h)$, where $d_{\text{in}} = d_p$ for layer 1 and $d_{\text{in}} = d_h$ for layer 2. The total per-step LSTM cost is approximately $8d_h(d_p + d_h) + 8d_h \cdot 2d_h$, applied at all S steps. The final linear head contributes $2d_h$ FLOPs and is negligible. The total cost is dominated by the input projection, $S \cdot 2Dd_p$, since $D \gg d_h$.

$\mathcal{D}^2$-Monitor (Cascade) Our cascade has two components. The base linear probe is applied at every denoising step to compute the per-step margin $|d_s|$ used for hesitation detection, regardless of whether the sample is escalated; this incurs $2SD$ FLOPs per sample. The expert (MLP or TimeAttn) is invoked only on the fraction p_{esc} of samples flagged as hesitant, and operates on the minimal window covering all hesitation steps with average length $S_{\text{win}} \le S$, rather than the full trajectory. The expected per-sample cost is

$$\mathbb{E}[\text{FLOPs}] = \underbrace{2SD}_{\text{base, always}} + \underbrace{p_{\text{esc}} \cdot F_{\text{expert}}(S_{\text{win}})}_{\text{expert, conditional}}, \tag{12}$$

Table 4: 2×2 ablation over training data (all examples vs. hesitation-curated) and application (all samples vs. routed samples), on LLaDA-8B-Instruct, mean $\pm$ std over 3 seeds. C2 is evaluated at the same λ that C4 selects, so C2 and C4 route an identical sample set and carry an identical $\mathrm{E}[P]$: C4 vs. C2 isolates the training data at matched test-time cost. C4 is $\mathcal{D}^2$-Monitor.

	Advanced probe		F1 (%)		$\mathrm{E}[P]$	
	Training data	Applied to	MLP	TimeAttn	MLP	TimeAttn
WildGuardMix						
C1	all data	all samples	87.82±0.42	87.81±0.21	1.05M	1.59M
C2	all data	routed	88.46±0.46	88.32±0.30	0.26M	0.22M
C3	curated	all samples	89.28±0.13	88.80±0.30	1.05M	1.59M
C4	curated	routed	**89.54±0.10**	**89.34±0.14**	0.26M	0.22M
ToxicChat						
C1	all data	all samples	87.08±0.18	87.28±0.92	1.05M	1.59M
C2	all data	routed	88.33±0.10	88.86±0.29	0.16M	0.37M
C3	curated	all samples	87.68±0.93	87.30±1.05	1.05M	1.59M
C4	curated	routed	**90.33±0.17**	**90.13±0.20**	0.16M	0.37M

where the expert FLOPs are

$$F_{\text{MLP-expert}}(S_{\text{win}}) = S_{\text{win}}D + 2DK \tag{13}$$

$$F_{\text{TimeAttn-expert}}(S_{\text{win}}) = 2S_{\text{win}}Dd_a + 2S_{\text{win}}D + 2DK. \tag{14}$$

The values of p_{esc} and S_{win} are measured empirically on *WildGuardMix* for each model and reported alongside the FLOPs in Table 10.

Across all four LLaDA models, D^2-Monitor achieves a substantially better efficiency–effectiveness trade-off than full-trajectory baselines. D^2-MLP requires under 1 MFLOP per sample, which is 1.5×–3.3× cheaper than MLP (Mean), 37×–78× cheaper than TimeAttn and over 150× cheaper than LSTM, while remaining competitive in F1 (Table 1). D^2-TimeAttn is more expensive than D^2-MLP due to the heavier expert, but still costs 3.5×–7× less than running TimeAttn on the full trajectory. The savings come from two sources. First, the cascade only invokes the expert on the fraction p_{esc} of samples flagged as hesitant, so most samples incur only the lightweight $2SD$ FLOPs of the base linear probe. Second, even on escalated samples, the expert operates on a localized hesitation window of average length $S_{\text{win}} \leq S$ rather than the full trajectory. As a result, D^2-MLP costs less than even a single-step MLP probe (MLP (Last Step), ∼2 MFLOPs) while attaining performance close to full-trajectory models, demonstrating that conditional computation effectively decouples cost from accuracy.

D.2 Disentangling Routing from Hesitation-Curated Training

$\mathcal{D}^2$-Monitor changes two things at once relative to a standalone advanced probe: the probe is *trained* on hesitation windows rather than on full trajectories, and it is *applied* only to the samples the router escalates. We separate the two factors with a 2×2 control over training data and application on LLaDA-8B-Instruct (Table 4). Two details make the comparison well posed. First, C2 is evaluated at the same λ that C4 selects, so C2 and C4 escalate an identical set of samples and carry an identical $\mathrm{E}[P]$; C4 versus C2 therefore isolates the training data at matched test-time cost. Second, in C3 the hesitation-trained probe is applied to every test sample: hesitant samples are scored on their hesitation window, while samples with no hesitation step fall back to the full trajectory.

Routing contributes $+0.64/+0.51$ F1 (MLP/TimeAttn) on WildGuardMix and $+1.25/+1.58$ on ToxicChat with an all-data probe, and $+0.26/+0.54$ and $+2.65/+2.83$ respectively with a curated probe. Curated training contributes $+1.46/+0.99$ and $+0.60/+0.02$ when the probe is applied to every sample, and $+1.08/+1.02$ and $+2.00/+1.27$ at matched cost. All sixteen contrasts are non-negative, but the relative size of the two factors reverses between the two datasets, as does the sign of the interaction term ($-0.38/+0.03$ on WildGuardMix, $+1.40/+1.25$ on ToxicChat). We therefore report the decomposition as dataset-dependent rather than claiming a single ordering. What is stable across both datasets is curated training at matched cost, which is at least $+1.0$ F1 in all four cases, and routing, which never lowers F1 while reducing $\mathrm{E}[P]$ by 4.0–7.2×. These deltas are differences of 3-seed means and carry no error bars.

Window construction	$\lvert w\rvert$	$\mathcal{D}^2$-MLP $\lambda^\star$	$\mathcal{D}^2$-MLP F1 @ $\lambda^\star$	$\mathcal{D}^2$-MLP F1 @ λ_{ref}	$\mathcal{D}^2$-TimeAttn $\lambda^\star$	$\mathcal{D}^2$-TimeAttn F1 @ $\lambda^\star$	$\mathcal{D}^2$-TimeAttn F1 @ λ_{ref}
Threshold-based							
Hesitation span (default)	16.9	24	**90.51**	**90.51**	4	**90.35**	**90.35**
Hesitant steps only	12.9	29	90.05	90.00	4	90.22	90.22
Fixed budget							
Top-k, $k = 4$	4.0	29	89.74	89.76	29	89.72	89.14
Top-k, $k = 8$	8.0	29	89.91	89.76	21	89.81	89.66
Top-k, $k = 16$	16.0	22	90.28	90.16	21	89.75	89.68
Top-k, $k = 24$	24.0	17	90.42	90.20	17	89.99	89.71
Soft weighting							
Soft, $c = 0.25$	13.0	29	90.05	89.88	21	89.85	89.91
Soft, $c = 0.5$	21.8	29	90.17	90.12	19	89.41	89.01
Soft, $c = 1$	28.2	13	90.08	89.86	18	89.74	89.46
Soft, $c = 2$	30.9	21	90.20	90.13	18	90.21	90.21
Full trajectory (uniform, $c \to \infty$)	32.0	15	90.32	90.13	17	90.22	89.66

Table 5: **Ablation over hesitation-window construction** on ToxicChat with LLaDA-8B-Instruct (F1 in %, seed 0). Only the window rule changes; everything else is held fixed. $|w|$ is the mean effective window length in steps (out of $S = 32$), identical for both probes since both the window rule and the margins are shared. We report each variant under two routing thresholds. $\lambda^\star$ is the threshold selected for that variant by the same rule as the main results, and therefore differs across rows. λ_{ref} is the default window's $\lambda^\star$ (24 for $\mathcal{D}^2$-MLP, 4 for $\mathcal{D}^2$-TimeAttn), held fixed across rows. Because routing depends on the number of hesitant steps rather than on the window rule, every row at λ_{ref} escalates the same samples at the same $\mathrm{E}[P]$ (140.5K and 629.7K respectively), which makes that column a cost-matched comparison. Best per column in **bold**.

D.3 Constructing the Hesitation Window

We compare different ways of building the hesitation window (Table 5). The default takes the contiguous span from the first to the last hesitant step, which also contains the non-hesitant steps in between; *hesitant steps only* drops those, keeping the same steps but not the contiguity. *Top-k* ignores τ and keeps the k steps with the smallest $|d_t|$, a fixed budget rather than a data-dependent one ($k \in \{4, 8, 16, 24\}$). *Soft* replaces hard selection with a weight $\exp(-|d_t|/c\tau)$ on every step, so c controls how sharply the probe concentrates on low-margin steps ($c \in \{0.25, 0.5, 1, 2\}$); as $c \to \infty$ this becomes a uniform average over the whole trajectory, which serves as the no-window control.

What the span buys is consistency, being the best or tied-best variant in every column, which is why we keep it as the default. How much it buys depends on the second-stage architecture. $\mathcal{D}^2$-MLP mean-pools its input, so extra steps are averaged away and most variants land within noise of the span, a uniform average over the entire trajectory included (90.13 against 90.51). $\mathcal{D}^2$-TimeAttn attends over the sequence and registers what is in it: the uniform average costs 0.69 points, and taking the same number of steps non-contiguously by margin (top-k, $k = 16$) costs 0.67.

D.4 Comparison with LLM-based Monitors

Setup We finetune Llama-Guard-3-1B (Llama Team, 2024) is against its own prompt template, which we apply unmodified through `tokenizer.apply_chat_template`. The template asks the model to check the conversation against the MLCommons hazard taxonomy (S1–S13: violent crimes, non-violent crimes, sex-related crimes, child sexual exploitation, defamation, specialized advice, privacy, intellectual property, indiscriminate weapons, hate, suicide and self-harm, sexual content, and elections) and to answer `safe` or `unsafe` on the first line, followed on a second line by the violated category keys. We keep the default category set and treat `unsafe` as a positive detection.

We evaluate two configurations. *Zero-shot* applies the released checkpoint directly. *Aligned* adapts it with LoRA on the training split of each dataset, using the same splits as the probe-based monitors so that both sides see identical training data: rank 16, $\alpha = 32$, dropout 0.05, applied to all attention and MLP projection matrices, learning rate 2×10^{-4} with a linear schedule, and effective batch size 16.

Table 6: Zero-shot vs. LoRA-aligned Llama-Guard-3-1B on binary safe/unsafe prompt classification (%). Unsafe is the positive class; FRR is the false-flag rate on truly safe inputs. † marks in-domain evaluation (eval split matches the LoRA's training set); all other LoRA rows are out-of-distribution. Note that on OpenAI-Moderation, alignment leaves accuracy unchanged and raises FRR.

Eval data	Model	Acc.	F1	Prec.	Rec.	FRR ↓
WildGuardMix	Zero-shot	79.5	78.8	82.4	68.4	11.6
	LoRA (WildGuardMix)†	90.1	89.9	92.1	85.0	5.8
OpenAI-Mod	Zero-shot	75.3	73.6	57.4	79.9	26.8
	LoRA (WildGuardMix)	75.2	74.5	56.1	92.9	32.7
ToxicChat	Zero-shot	82.7	60.7	21.8	55.5	15.2
	LoRA (WildGuardMix)	93.2	80.4	51.4	86.7	6.3
	LoRA (ToxicChat)†	97.5	90.4	83.2	80.9	1.2

	Llama-Guard-3-1B		$\mathcal{D}^2$-Monitor	
	zero-shot	LoRA	$\mathcal{D}^2$-MLP	$\mathcal{D}^2$-TimeAttn
Adapted on WildGuardMix				
Monitor params	1.12B	1.12B	≤0.34M	≤0.55M
WildGuardMix†	78.8	89.9	89.5	89.3
ToxicChat	60.7	80.4	73.2	75.3
OpenAI-Mod	73.6	74.5	68.9	70.5
Adapted on ToxicChat				
Monitor params	1.12B	1.12B	≤0.18M	≤0.37M
ToxicChat†	60.7	90.4	90.3	90.1

Table 7: $\mathcal{D}^2$-Monitor versus an LLM-based monitor at matched adaptation data. Entries are F1 (%) except the parameter rows. † marks in-domain evaluation; all other entries are transfer. The zero-shot column is the released Llama-Guard checkpoint and does not depend on the adaptation set, so it is repeated in both blocks as a reference point. $\mathcal{D}^2$-Monitor rows are LLaDA-8B-Instruct; the parameter count is $\mathrm{E}[P]$ for the in-domain setting (marked †); values for the transfer settings are in Table 3. We did not evaluate transfer from ToxicChat, so that block reports in-domain results only.

What the two monitors have access to The comparison is not symmetric, and the asymmetry cuts both ways. Llama-Guard classifies the input text directly, so it needs no access to the model being monitored and a single row applies to all four D-LLMs, whereas $\mathcal{D}^2$-Monitor reads internal states and is trained per model. In exchange, $\mathcal{D}^2$-Monitor rides on a forward pass that is already happening, while Llama-Guard adds an independent model call to every request. Which side this favors depends on the deployment: a provider serving many different models may prefer one text-level monitor, while a setting with a fixed model and a tight per-request budget favors the probe.

Results Table 6 reports the full metric breakdown and Table 7 places the two monitors side by side at matched training data. Three points stand out. First, without task-specific adaptation Llama-Guard is substantially weaker, reaching 60.7 F1 on ToxicChat at 21.8 precision. Second, once aligned it matches $\mathcal{D}^2$-Monitor in-domain, but does so with over 2000× as many monitor parameters at inference. Third, under transfer it retains more of its performance than any probe-based method in Table 3, though not uniformly: on OpenAI-Moderation, alignment adds only 0.9 F1 over zero-shot while raising the false-flag rate on safe inputs from 26.8% to 32.7%.

E Additional Analysis

E.1 Analysis of Hesitation Dynamics

Cross-Boundary Probability We first analyze the local instability of model predictions by measuring how likely a step is to cross the decision boundary in the next step. Given the signed margin d_s at step s, we compute the probability that the sign of the margin changes at the next step, i.e., $\mathrm{sign}(d_{s+1}) \neq \mathrm{sign}(d_s)$, as a function of the margin $|d_s|$. Figure 9 shows the crossing probability

binned over $|d_s|$. Across all four LLaDA models, we observe a consistent pattern: steps close to the decision boundary exhibit a significantly higher probability of crossing it in the next step, while steps far from the boundary are highly stable. The crossing probability decreases sharply as $|d_s|$ increases, quickly approaching zero beyond a small margin threshold. This indicates that hesitation steps, characterized by small margin magnitudes, are intrinsically unstable and prone to prediction flips, suggesting that the decision boundary region concentrates most of the local uncertainty in the trajectory.

Margin Persistence We further analyze the temporal structure of hesitation by measuring how long a model remains in a hesitant state. Given a threshold τ, we define a step as hesitant if $|d_s| < \tau$. For each such step, we compute the probability that the model remains hesitant after k steps, i.e.,

$$P(|d_{s+k}| < \tau \mid |d_s| < \tau),$$

for $k = 1, 2, \ldots, K$. Fig. 9 shows the persistence curves for different models. We also report the unconditional probability of being in a hesitant state as a baseline. Across models, we observe that hesitation exhibits clear temporal persistence: once the model enters a low-margin region, it tends to remain in that region for multiple subsequent steps, with the persistence probability decaying gradually as k increases. These results suggest that hesitation is not only a local phenomenon but also forms contiguous segments along the generation trajectory. This temporal coherence further motivates our design of operating on localized hesitation windows, as they capture the regions where uncertainty is both concentrated and temporally structured.

E.2 Margin Outperforms Entropy and Confidence as Step-Count Signal

In Section 3.2, we noted that the step-count construction underlying n_τ is general and can be applied to any per-step hesitation signal. We instantiate this for the two probe-extrinsic signals introduced in Section 3.3, defining

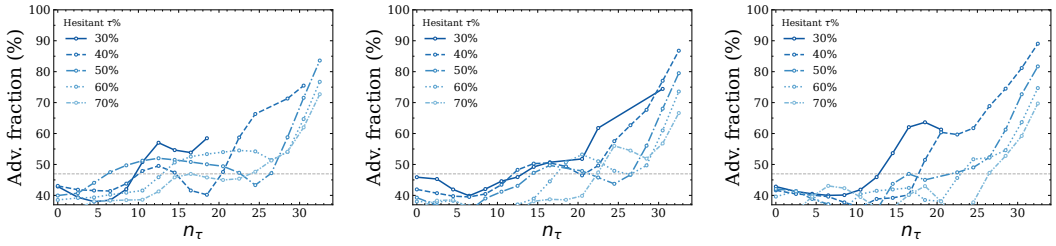
$$n_{\text{entropy}} = \sum_{s=1}^{S} \mathbb{I}[E_s \geq \tau_E], \tag{15}$$

$$n_{\text{confidence}} = \sum_{s=1}^{S} \mathbb{I}[C_s \leq \tau_C], \tag{16}$$

where E_s and C_s are the step-wise entropy and confidence scores, and τ_E, τ_C are their respective thresholds. We compare the three step-count signals on LLaDA-8B-Instruct in Fig. 8, where for each signal we evaluate probe F1 across n_τ buckets (more generally, n_{signal} buckets) under five threshold settings parameterized by the resulting hesitant ratio. Two observations stand out. First, all three signals are predictive of difficulty in the qualitative sense: F1 decreases monotonically as the count increases, regardless of which signal is used. This confirms that the step-count construction itself is a general approach to extracting trajectory-level difficulty information from per-step signals. Second, n_τ produces the steepest and most extended F1 decline across all hesitant ratios. The margin-based curves (blue) drop from $\sim 90\%$ at $n_\tau = 0$ to 55–77% at the largest buckets, whereas entropy-based and confidence-based curves (orange and green) plateau at $\sim$ 75–80% and cover a much narrower range of n_{signal} values. The larger F1 spread under n_τ indicates that the probe margin produces a finer and more discriminative stratification of difficulty than entropy or confidence: trajectories deemed "highly hesitant" under n_τ are substantially harder for the probe than those deemed "highly hesitant" under entropy or confidence. This justifies our choice of n_τ as the routing signal in $\mathcal{D}^2$-Monitor. A natural explanation is that the probe margin is *probe-aware*: it directly measures distance to the probe's decision boundary, which is the quantity governing whether the probe will misclassify a sample. In contrast, entropy and confidence are computed from the D-LLM's predicted token distribution and are agnostic to the specific probe being used. They capture model-level uncertainty but miss the probe-specific decision dynamics that matter for routing.

E.3 Hesitation Severity Captures Adversarial Inputs

We further examine *which* samples accumulate large n_τ, beyond their being difficult for the linear probe. Recall that *WildGuardMix* contains a mixture of natural and adversarially designed prompts,



(a) LLaDA-8B-Base (b) LLaDA-8B-Instruct (c) LLaDA-1.5

Figure 5: **Adversarial fraction vs. hesitation severity n_τ.** For each n_τ bucket, we report the fraction of samples drawn from the adversarial split of *WildGuardMix*. Each curve corresponds to a different τ setting (parameterized by the resulting hesitant ratio). The gray dashed line marks the dataset-wide adversarial fraction ($\sim 47\%$). Adversarial fraction increases with n_τ on all three LLaDA variants, starting below the dataset baseline and ending well above it.

where the latter are constructed to evade safety classifiers and are widely regarded as the harder portion of the benchmark (Han et al., 2024). We compute, for each n_τ bucket, the fraction of samples drawn from the adversarial split (*Adv. fraction*) and report the result on three LLaDA variants in Fig. 5. The relationship is consistent across all three models. At $n_\tau = 0$, the adversarial fraction is 38–46%, noticeably below the dataset-wide baseline of $\sim 47\%$ (gray dashed line); as n_τ increases, the fraction rises monotonically and reaches 67–89% at the largest buckets across the five τ settings. The overall trend, including both the sub-baseline behavior at $n_\tau = 0$ and the monotonic rise toward $n_\tau \to S$, holds for LLaDA-8B-Base, LLaDA-8B-Instruct, and LLaDA-1.5, indicating that the association between hesitation severity and adversarial inputs is a model-agnostic property rather than an artifact of any specific model or threshold choice. Conceptually, this is consistent with the design intent of adversarial prompts: they are crafted to push the model into a borderline decision, which manifests as repeated proximity to the probe's decision boundary across denoising steps. Hesitation severity n_τ thus naturally captures this signature of adversarial conditioning, providing a concrete semantic interpretation beyond abstract probe uncertainty. The above analysis is conducted at the level of n_τ buckets and characterizes the data property of hesitation. We now turn to the operational consequence of this property in our cascade: *which samples are actually routed to the second-stage probe?* Since routing in $\mathcal{D}^2$-Monitor is triggered by n_τ exceeding a threshold (selected on a held-out validation set), the routed subset should, by construction, inherit the adv-rich property documented above. We empirically verify this by computing the adversarial fraction within the routed subset for each LLaDA variant and report the result in Table 9. The routed subset is consistently and substantially enriched in adversarial inputs relative to the dataset-wide baseline of $\sim 47\%$: the adversarial fraction reaches 68.2% on LLaDA-8B-Base, 80.8% on LLaDA-8B-Instruct, 80.5% on LLaDA-1.5, and 75.5% on LLaDA-2.0-mini for $\mathcal{D}^2$-MLP, with $\mathcal{D}^2$-TimeAttn exhibiting comparable enrichment (64.4%, 87.5%, 80.5% ,75.5%, respectively). In other words, the cascade routes predominantly adversarial samples to the expert. This confirms that $\mathcal{D}^2$-Monitor does not merely allocate extra capacity to "hard" samples in a generic sense but specifically channels it toward adversarial inputs that pose the highest risk to safety classification. Viewed in this light, the bi-level design becomes more than a cost-saving heuristic: it is a targeted defense mechanism, with hesitation severity acting as an implicit detector of adversarial conditioning, and the second-stage expert serving as the specialized classifier for these flagged cases.

E.4 Sensitivity to Test-time Hyperparameters

Both τ and λ are selected at test time and require no retraining, so a practitioner can move along the resulting operating curve at no additional training cost. We therefore sweep them directly on the trained monitors. For in-distribution evaluation we keep τ at the value selected on the training data and sweep only λ, since the test distribution matches the training one. For cross-dataset evaluation, we sweep τ and λ jointly.

Fig. 6 reports the in-distribution sweep. F1 varies by less than 0.8 points across $\lambda \in [4, 27]$ for both probe architectures, so λ can be traded against $\mathrm{E}[P]$ over a wide range at little cost in detection quality.

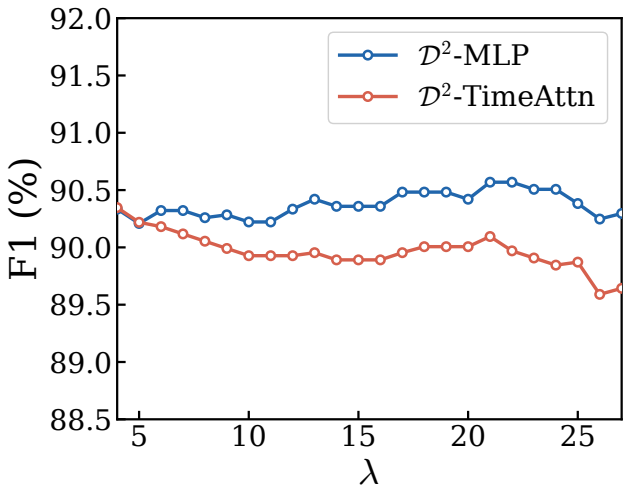


Figure 6: In-distribution sensitivity of F1 to λ for $\mathcal{D}^2$-Monitor on ToxicChat (LLaDA-8B-Instruct), with τ fixed at its training value. F1 varies by less than 0.8 points across $\lambda \in [4, 27]$ for both probe architectures.

Fig. 7 reports the cross-dataset sweep. Over the entire (τ, λ) grid, F1 spans 3.5 points for $\mathcal{D}^2$-MLP and 3.3 points for $\mathcal{D}^2$-TimeAttn, indicating that the cross-dataset results in Table 3 do not depend on a narrow choice of hyperparameters.

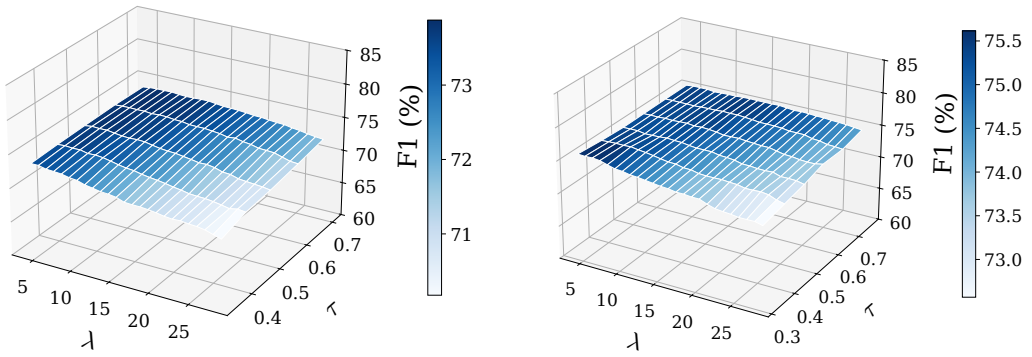
E.5 Sensitivity to the Probing Layer

The hidden states we probe are taken from a single layer of the D-LLM. We check how much this choice matters by repeating the evaluation with probes trained on layers 20, 24, 28 and 32 of LLaDA-8B-Instruct, keeping the rest of the pipeline fixed. Unlike the multi-seed results elsewhere in this appendix, these numbers come from a single run (seed 0).

Across the four mid-to-late layers, F1 varies by about 0.5 points for both variants (89.2–89.7 for $\mathcal{D}^2$-MLP and 89.1–89.4 for $\mathcal{D}^2$-TimeAttn), so the method does not depend on the exact layer. This is consistent with recent evidence that diffusion LLMs carry substantially redundant representations across layers, with many layers encoding similar information (Goel et al., 2026).

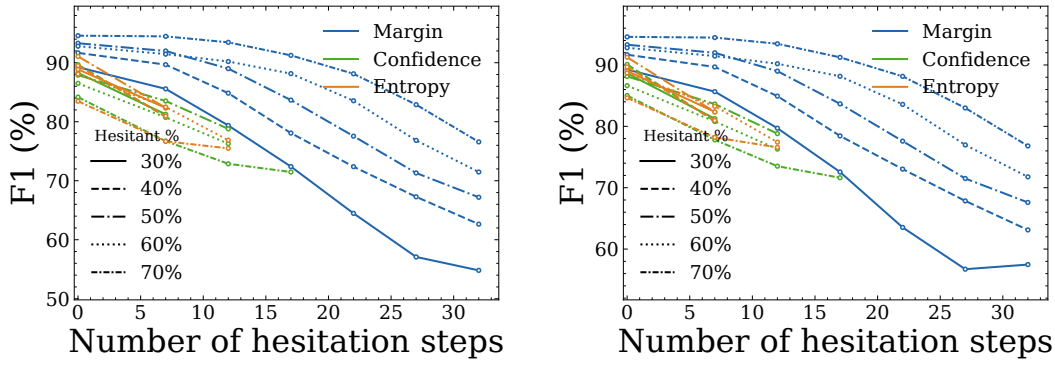
Table 8: **Sensitivity to the probing layer.** F1 on *WildGuardMix* with LLaDA-8B-Instruct, probes taken from different layers. Single run (seed 0); all other settings follow Appendix C.

Probing layer	$\mathcal{D}^2$-MLP	$\mathcal{D}^2$-TimeAttn
20	89.2	89.1
24	89.3	89.1
28	89.2	89.2
32	89.7	89.4



(a) $\mathcal{D}^2$-MLP (b) $\mathcal{D}^2$-TimeAttn

Figure 7: Cross-dataset sensitivity of F1 to (τ, λ) for $\mathcal{D}^2$-Monitor trained on WildGuardMix and evaluated on ToxicChat (LLaDA-8B-Instruct). Both τ and λ are varied at test time without retraining.



(a) LP (MV) (b) LP (Mean)

Figure 8: **Comparison of step-count signals on LLaDA-8B-Instruct.** For each step-count signal (n_τ from probe margin, n_{entropy} from step-wise entropy, $n_{\text{confidence}}$ from step-wise confidence), we report probe F1 across buckets of increasing hesitation count, under two base classifier variants: **(a)** LP (MV) and **(b)** LP (Mean). Each color corresponds to one signal, and each line style corresponds to a different threshold setting parameterized by the resulting hesitant ratio. Across both base classifiers, the margin-based signal n_τ produces the steepest and most extended F1 decline, indicating it stratifies difficulty more discriminatively than the probe-extrinsic signals.

Table 9: **Adversarial fraction in the routed subset.** For each LLaDA variant, we report the percentage of routed samples that are drawn from the adversarial split of *WildGuardMix*, against a dataset-wide baseline of $\sim 47\%$. Routing thresholds are selected on a held-out validation set per method.

Method	LLaDA-8B-Base	LLaDA-8B-Instruct	LLaDA-1.5	LLaDA-2.0-mini
$\mathcal{D}^2$-MLP	68.2%	80.8%	80.5%	75.5%
$\mathcal{D}^2$-TimeAttn	64.4%	87.5%	80.5%	75.5%

Table 10: **Expected FLOPs (MFLOPs) per sample.** Computed analytically using the architectures in Appendix C.2, with denoising steps S, hidden dimension D, and (for our cascade) average escalation rate p_{esc} and average hesitation window S_{win} measured on *WildGuardMix*.

Method	LLaDA-8B-Base FLOPs	LLaDA-8B-Instruct FLOPs	LLaDA-1.5 FLOPs	LLaDA-2.0-mini FLOPs
Single-step methods				
LP (Last Step)	0.008	0.008	0.008	0.004
MLP (Last Step)	2.10	2.10	2.10	1.05
Full-trajectory methods				
LP (MV)	0.26	0.26	0.26	0.52
LP (Mean)	0.14	0.14	0.14	0.27
MLP (MV)	67.1	67.1	67.1	134
MLP (Mean)	2.23	2.23	2.23	1.31
TimeAttn	35.7	35.7	35.7	68.2
LSTM	163	163	163	386
$\mathcal{D}^2$-MLP (Ours)	0.92	0.68	0.97	0.87
$\mathcal{D}^2$-TimeAttn (Ours)	8.16	5.10	8.48	19.4

Table 11: **Post-generation inference time (ms).** Results are measured on WildGuardMix (1,725 prompts) after generation on different LLaDA models.

	LLaDA-8B-Base	**LLaDA-8B-Instruct**	**LLaDA-1.5**	**LLaDA-2.0-mini**
Method	Time	Time	Time	Time
Single-step methods				
LP (Last Step)	0.73	0.31	0.42	0.37
MLP (Last Step)	0.79	0.52	0.66	0.34
Full-trajectory methods				
LP (MV)	0.61	0.42	0.36	0.44
LP (Mean)	2.46	1.78	1.10	3.33
MLP (MV)	0.63	0.68	0.71	0.43
MLP (Mean)	3.04	1.94	1.59	3.65
TimeAttn	30.96	33.34	31.61	26.31
LSTM	346.44	317.42	307.88	348.56
$\mathcal{D}^2$-MLP (Ours)	0.57	0.56	0.66	0.55
$\mathcal{D}^2$-TimeAttn (Ours)	7.36	7.69	7.63	5.11

Table 12: **Intra-dataset performance on LLaDA-8B-Base.** Monitors are trained and tested on *WildGuardMix*. Results are mean±std over 3 seeds. Best results are in **bold**, and second-best are underlined. ↓ indicates lower is better.

Method	**E[P]**	**Acc**	**F1**	**F2**	**Precision**	**Recall**	**FRR ↓**
Single-step methods							
LP (Last Step)	4×10^{-3}M	84.2±0.2	83.7±0.2	79.3±0.6	84.5±0.7	78.1±0.8	11.1±0.6
MLP (Last Step)	1.05M	85.6±0.3	85.2±0.4	80.2±0.5	87.2±0.4	78.6±0.5	9.0±0.2
Full-trajectory methods							
LP (MV)	4×10^{-3}M	85.2±0.7	84.8±0.7	79.8±0.9	86.6±1.0	78.3±0.9	9.4±0.7
LP (Mean)	4×10^{-3}M	85.7±1.0	85.3±1.0	80.7±1.2	87.0±1.2	79.3±1.2	9.2±0.9
MLP (MV)	1.05M	86.8±0.4	86.4±0.4	81.4±0.9	89.0±0.2	79.8±1.1	7.7±0.3
MLP (Mean)	1.05M	87.5±0.3	87.1±0.4	82.1±1.0	89.9±0.4	80.4±1.3	7.0±0.4
TimeAttn	1.59M	87.6±0.5	87.2±0.6	82.8±3.0	89.4±1.6	81.3±3.3	7.5±1.0
LSTM	2.57M	87.0±0.4	86.5±0.4	82.6±0.6	88.0±0.2	81.3±0.7	8.6±0.2
$\mathcal{D}^2$-MLP (Ours)	0.34±0.06M	88.5±0.2	88.2±0.3	83.6±0.5	90.9±0.0	81.9±0.6	6.4±0.1
$\mathcal{D}^2$-TimeAttn (Ours)	0.55±0.03M	**89.4±0.3**	**89.1±0.4**	**85.1±0.4**	**91.3±0.5**	**83.7±0.4**	**6.2±0.4**

Table 13: **Intra-dataset performance on LLaDA-8B-Instruct.** Monitors are trained and tested on *WildGuardMix*. Results are mean±std over 3 seeds. Best results are in **bold**, and second-best are underlined. ↓ indicates lower is better.

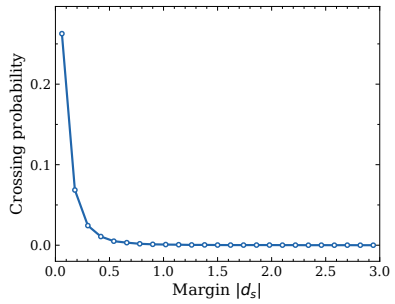
Method	E[P]	Acc	F1	F2	Precision	Recall	FRR ↓
Single-step methods							
LP (Last Step)	4×10^{-3}M	87.0±0.3	86.6±0.3	83.1±0.2	87.3±1.0	82.1±0.5	9.3±0.9
MLP (Last Step)	1.05M	87.2±0.5	86.9±0.5	83.5±0.5	87.4±0.7	82.5±0.5	9.2±0.6
Full-trajectory methods							
LP (MV)	4×10^{-3}M	87.8±0.5	87.6±0.5	84.8±0.7	87.5±0.7	84.2±0.8	9.3±0.5
LP (Mean)	4×10^{-3}M	87.9±0.4	87.6±0.4	84.4±0.7	88.2±0.8	83.5±1.0	8.7±0.7
MLP (MV)	1.05M	88.0±0.4	87.7±0.4	84.1±0.8	88.8±0.1	83.0±1.0	8.1±0.2
MLP (Mean)	1.05M	88.1±0.4	87.8±0.4	84.2±0.7	89.0±0.1	83.1±0.9	8.0±0.1
TimeAttn	1.59M	88.1±0.2	87.8±0.2	84.3±0.7	88.9±1.3	83.2±1.1	8.1±1.1
LSTM	2.57M	88.2±0.4	87.9±0.4	84.6±1.1	88.7±1.5	83.6±1.5	8.3±1.3
$\mathcal{D}^2$-MLP (Ours)	0.26±0.03M	**89.8±0.1**	**89.5±0.1**	**86.1±0.3**	91.2±0.3	**84.9±0.5**	6.4±0.3
$\mathcal{D}^2$-TimeAttn (Ours)	0.22±0.04M	89.6±0.1	89.3±0.1	85.6±0.6	**91.2±0.6**	84.3±0.8	**6.3±0.5**

Table 14: **Intra-dataset performance on LLaDA-1.5.** Monitors are trained and tested on *WildGuardMix*. Results are mean±std over 3 seeds. Best results are in **bold**, and second-best are underlined. ↓ indicates lower is better.

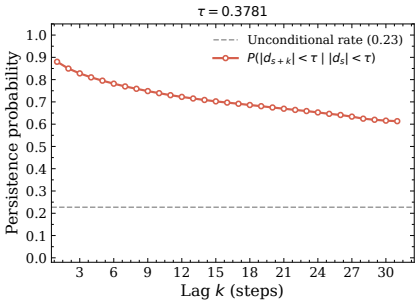
Method	E[P]	Acc	F1	F2	Precision	Recall	FRR ↓
Single-step methods							
LP (Last Step)	4×10^{-3}M	88.2±0.1	87.9±0.1	83.9±0.1	89.6±0.3	82.6±0.2	7.5±0.3
MLP (Last Step)	1.05M	87.3±0.0	87.0±0.0	83.9±0.1	87.3±0.2	83.1±0.2	9.4±0.2
Full-trajectory methods							
LP (MV)	4×10^{-3}M	87.4±0.1	87.2±0.1	84.7±0.2	86.7±0.1	84.2±0.3	10.1±0.1
LP (Mean)	4×10^{-3}M	87.7±0.1	87.4±0.1	84.9±0.1	87.0±0.2	84.4±0.2	9.8±0.2
MLP (MV)	1.05M	87.8±0.1	87.5±0.0	84.1±0.5	88.3±0.7	83.2±0.8	8.5±0.7
MLP (Mean)	1.05M	88.1±0.0	87.8±0.0	84.2±0.5	88.9±0.7	83.1±0.8	8.0±0.6
TimeAttn	1.59M	88.2±0.4	88.0±0.5	84.1±1.0	90.0±0.2	82.8±1.3	7.1±0.2
LSTM	2.57M	88.1±0.4	87.9±0.4	84.5±0.7	88.6±0.3	83.6±0.9	8.3±0.2
$\mathcal{D}^2$-MLP (Ours)	0.27±0.08M	89.7±0.1	89.4±0.1	**86.1±0.3**	90.7±0.2	**85.1±0.4**	6.8±0.2
$\mathcal{D}^2$-TimeAttn (Ours)	0.37±0.07M	**89.7±0.2**	**89.4±0.2**	86.0±0.2	**90.9±0.4**	84.9±0.1	**6.6±0.3**

Table 15: **Intra-dataset performance on LLaDA-2.0-mini.** Monitors are trained and tested on *WildGuardMix*. Results are mean±std over 3 seeds. Best results are in **bold**, and second-best are underlined. ↓ indicates lower is better.

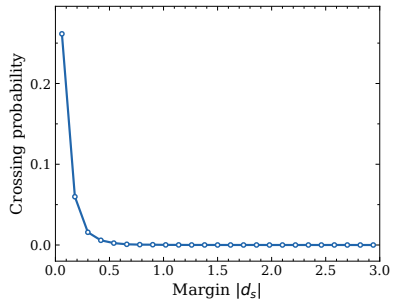
Method	E[P]	Acc	F1	F2	Precision	Recall	FRR ↓
Single-step methods							
LP (Last Step)	2×10^{-3}M	77.4±0.2	76.1±0.2	65.2±0.4	81.8±0.5	62.1±0.4	10.7±0.3
MLP (Last Step)	0.52M	78.5±0.1	77.6±0.1	68.9±0.1	81.1±0.3	66.4±0.2	12.0±0.3
Full-trajectory methods							
LP (MV)	2×10^{-3}M	80.1±0.3	79.1±0.2	69.7±0.5	84.3±1.3	66.8±0.8	9.6±1.1
LP (Mean)	2×10^{-3}M	80.3±0.2	79.4±0.2	70.4±1.5	84.4±1.9	67.6±2.0	9.8±1.7
MLP (MV)	0.52M	80.9±0.1	80.0±0.0	71.3±1.3	84.9±1.8	68.6±1.8	9.5±1.5
MLP (Mean)	0.52M	81.2±0.1	80.4±0.0	71.9±1.0	85.1±1.4	69.2±1.3	9.4±1.2
TimeAttn	0.80M	81.7±0.4	80.7±0.2	70.6±1.2	88.1±2.6	67.2±1.7	7.3±2.0
LSTM	1.51M	81.7±0.6	80.8±0.5	71.4±0.6	87.1±2.2	68.3±1.0	7.9±1.7
$\mathcal{D}^2$-MLP (Ours)	0.06±0.01M	84.0±0.1	83.4±0.1	**76.4±0.3**	87.4±0.4	**74.0±0.4**	8.3±0.4
$\mathcal{D}^2$-TimeAttn (Ours)	0.12±0.04M	**84.2±0.3**	**83.5±0.3**	75.7±0.7	**88.7±0.5**	73.1±0.8	**7.2±0.4**



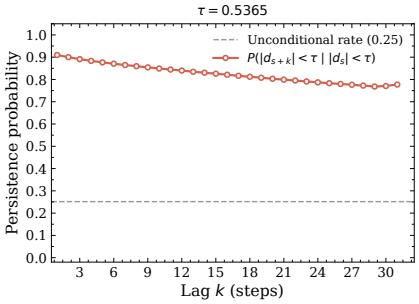
(a) LLaDA-8B-Base: Crossing



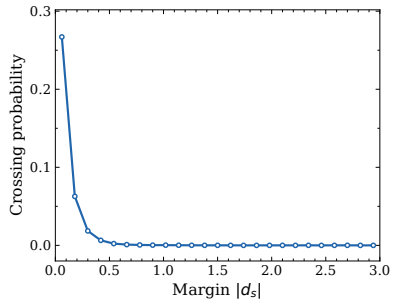
(b) LLaDA-8B-Base: Persistence



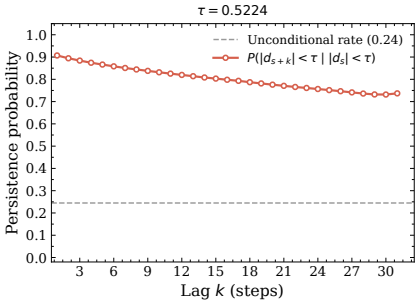
(c) LLaDA-8B-Instruct: Crossing



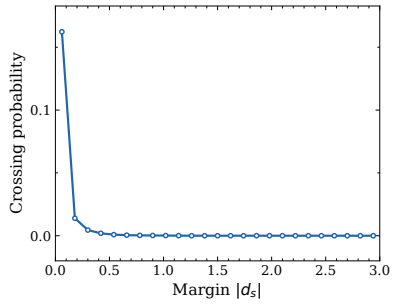
(d) LLaDA-8B-Instruct: Persistence



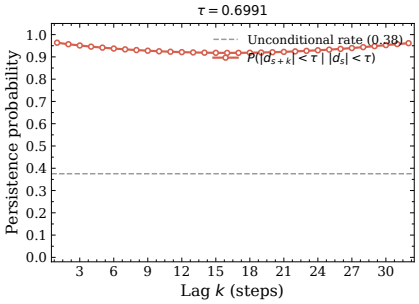
(e) LLaDA-1.5: Crossing



(f) LLaDA-1.5: Persistence

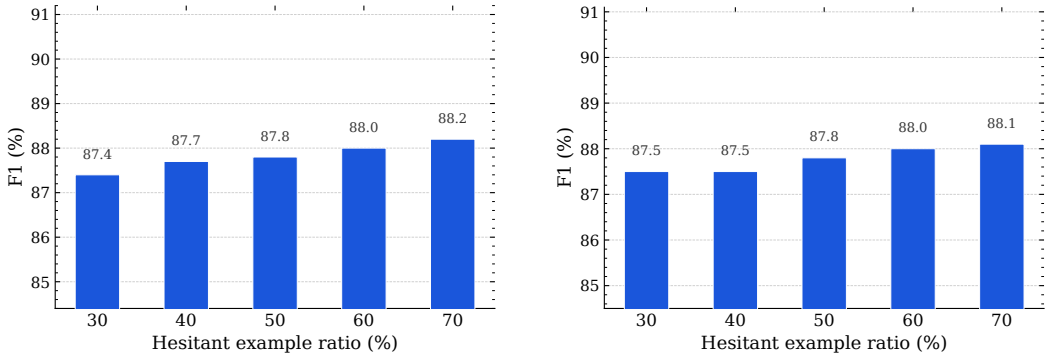


(g) LLaDA-2.0-mini: Crossing



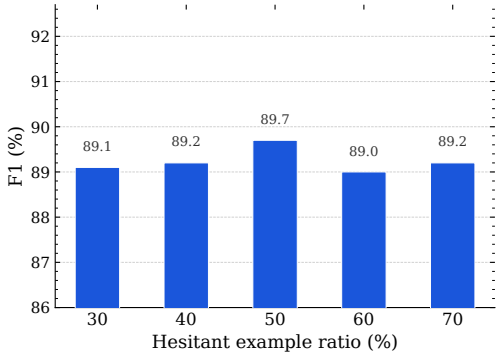
(h) LLaDA-2.0-mini: Persistence

Figure 9: Cross-boundary probability (left) and margin persistence (right) across four LLaDA variants. Left: crossing probability as a function of margin $|d_s|$. Right: persistence probability $P(|d_{s+k}| < \tau \mid |d_s| < \tau)$. Margins are determined by OOF scoring on *WildGuardMix*.

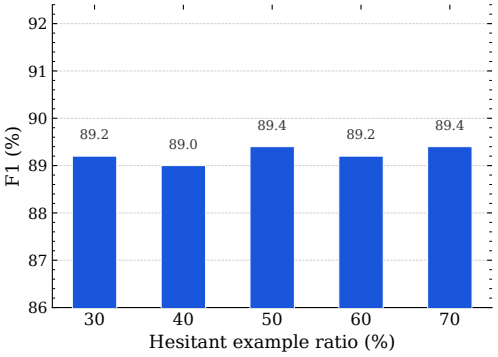


(a) LLaDA-8B-Base: D^2-MLP

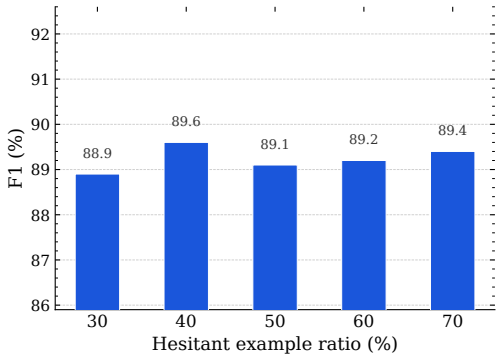
(b) LLaDA-8B-Base: D^2-TimeAttn



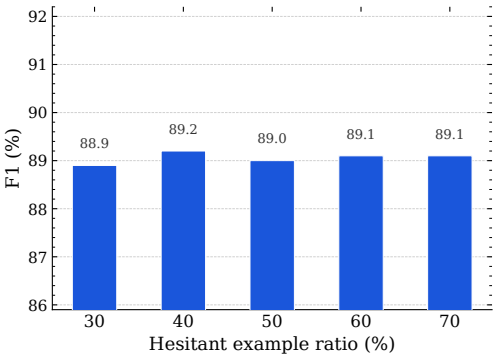
(c) LLaDA-8B-Instruct: D^2-MLP



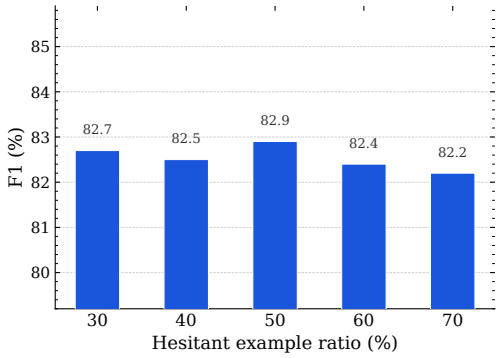
(d) LLaDA-8B-Instruct: D^2-TimeAttn



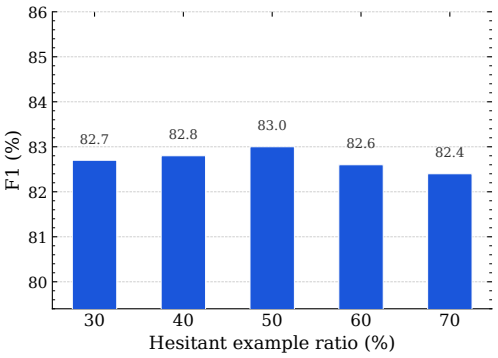
(e) LLaDA-1.5: D^2-MLP



(f) LLaDA-1.5: D^2-TimeAttn



(g) LLaDA-2.0-mini: D^2-MLP



(h) LLaDA-2.0-mini: D^2-TimeAttn

Figure 10: F1 score (%) vs. hesitant example ratio under D^2-MLP (left) and D^2-TimeAttn (right) across four LLaDA variants. All probes are trained and tested on *WildGuardMix*.