JUMP: EFFICIENT MEMBERSHIP INFERENCE ON FINE-TUNED DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Membership inference attacks (MIAs) test whether a candidate example was used to train a language model. Existing attacks on fine-tuned discrete diffusion language models (dLLMs) often aggregate reconstruction signals across many mask configurations, requiring repeated model evaluations. We propose *Joint Uncertainty-Guided Mask Probing* (JUMP), an efficient MIA that exploits the ability of dLLMs to predict masked tokens in parallel. Using the pre-fine-tuning checkpoint as a reference, JUMP selects low-confidence positions, masks them jointly, and aggregates clipped target–reference reconstruction gaps. This focuses the attack on positions that reveal stronger membership signals from fine-tuning. After mask selection, all selected tokens are evaluated with one scoring query per model. Across six MIMIR domains, JUMP improves mean ROC-AUC over a prior multi-mask attack from 0.819 to 0.902 on LLaDA and from 0.851 to 0.942 on Dream. Including mask selection, it requires only three forward passes per example, compared with 32 for the baseline. We further extend JUMP to the target-only setting by replacing target–reference scoring with *relative token preference*, which compares the observed token with alternative predictions at the same masked position. Target-Only JUMP achieves mean ROC-AUCs of 0.609 and 0.638 on LLaDA and Dream.

1 INTRODUCTION

Membership inference attacks (MIAs) aim to determine whether a candidate example was included in a model's training data (Shokri et al., 2017; Yeom et al., 2018). For large language models (LLMs), they provide a way to audit privacy risks associated with text used during pretraining or fine-tuning (Carlini et al., 2021; Duan et al., 2024). Most existing LLM MIAs use autoregressive (AR) next-token probabilities as membership evidence, refining sequence likelihood through reference calibration, token selection, information-theoretic normalization, or comparisons with perturbed inputs (Shi et al., 2024b; Zhang et al., 2025; Xie et al., 2024; Mattern et al., 2023; Chang et al., 2025; Tao & Shokri, 2026). Because each token is evaluated under a fixed left-to-right context, these methods mainly differ in how they select, calibrate, and aggregate a predetermined set of token scores.

Recent discrete diffusion language models (dLLMs), including LLaDA and Dream, offer a different interface for membership inference (Nie et al., 2025; Ye et al., 2025). They generate text by reconstructing masked tokens from bidirectional context rather than predicting tokens strictly from left to right. Two properties are particularly relevant. First, an attacker can choose which tokens to mask and thereby change the reconstruction task; we call this *any-order decodability*. Second, one forward pass returns predictions for all masked positions in parallel; we call this *parallel decodability*. Together, these properties let the attacker control both the probed positions and their reconstruction context, making the mask set itself a central component of attack design.

Existing dLLM attacks collect membership evidence across multiple masked reconstructions. SAMA (Chen et al., 2026) targets fine-tuned dLLMs by comparing reconstruction losses between the target and a reference model, ideally the pre-fine-tuning checkpoint, across progressively masked inputs. Other work has explored distributional statistics such as reconstruction-gain skewness (Zhai et al., 2026). While these approaches demonstrate the value of masked reconstruction for membership inference, repeated probing increases cost, and random mask positions can include tokens with little membership evidence. This motivates our central question: *can informative positions be identified first and then scored together with a single masked query per model?*

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.

In this work, we first consider membership in a dLLM's fine-tuning set when both the target model and its pre-fine-tuning checkpoint are available. We propose *Joint Uncertainty-Guided Mask Probing* (JUMP), which combines uncertainty-guided mask selection with single-pass scoring. Our diagnostics show that low-reference-confidence positions exhibit stronger member–non-member separation in the reconstruction improvement induced by fine-tuning. JUMP identifies these positions with a reference-based selector, masks them jointly, and aggregates clipped differences between the target and reference reconstruction log-probabilities. The reference comparison controls for pre-existing predictability, while clipping limits the influence of extreme token scores. Because all selected tokens are reconstructed in parallel, scoring requires only one masked query to each model. Across six MIMIR domains (Duan et al., 2024), JUMP improves mean ROC-AUC over SAMA from 0.819 to 0.902 on LLaDA and from 0.851 to 0.942 on Dream. Including the selector pass, it uses three forward passes per example rather than 32 in our SAMA configuration.

We further extend JUMP to the target-only setting, where the target's full predictive distribution serves as an internal comparison in place of a reference model. Target-Only JUMP compares each observed token with the target model's own alternative predictions, filters near-deterministic positions, and estimates relative token preference through balanced grouped masking. Across the same six MIMIR domains, it achieves mean ROC-AUCs of 0.609 on LLaDA and 0.638 on Dream, showing that target predictions alone retain a weaker but measurable membership signal.

2 Related Work

Membership inference attacks. Given a target model M_{tgt} trained on $\mathcal{D}_{\mathrm{train}}$ and a candidate example x, an MIA asks whether $x \in \mathcal{D}_{\mathrm{train}}$ (Shokri et al., 2017; Yeom et al., 2018). We write a score-based MIA as a scalar statistic

$$g(x;\, M_{\mathrm{tgt}},\, M_{\mathrm{ref}}) \in \mathbb{R},$$

where M_{ref} may be absent for target-only attacks. The attacker predicts membership when $g(x; M_{\mathrm{tgt}}, M_{\mathrm{ref}}) > \eta$, and varying η yields the ROC curve. A classical example is the negative-loss statistic $g_{\mathrm{loss}}(x; M_{\mathrm{tgt}}) = -\ell(M_{\mathrm{tgt}}, x)$, which connects membership leakage to the generalization gap (Yeom et al., 2018). Reference-based attacks instead compare the target with a reference, e.g., $g_{\mathrm{ref}}(x) = -\ell(M_{\mathrm{tgt}}, x) + \ell(M_{\mathrm{ref}}, x)$, to reduce the confounding effect of intrinsic example difficulty (Watson et al., 2022; Carlini et al., 2022; Zarifzadeh et al., 2024). Complementary work studies training-data extraction and privacy auditing (Long et al., 2018; Carlini et al., 2019; 2021; 2023; Nasr et al., 2025; Steinke et al., 2023; Jagielski et al., 2023; Lukas et al., 2023; Mireshghallah et al., 2022).

MIA for autoregressive language models. For an autoregressive language model, next-token log-likelihood provides a natural membership score:

$$g_{\mathrm{AR}}(x) = \log p_{\mathrm{AR}}(x) = \sum_{i=1}^{L} \log p(x_i \mid x_{<i}). \tag{1}$$

Methods such as Min-K% and Min-K%++ focus on low-probability tokens rather than the full sequence (Shi et al., 2024b; Zhang et al., 2025), while ReCaLL, neighborhood-based attacks, and context-aware methods calibrate likelihood using alternative contexts or comparisons (Xie et al., 2024; Mattern et al., 2023; Chang et al., 2025). Recent work develops information-theoretic normalization of the membership signal (Tao & Shokri, 2026). Although these attacks aggregate evidence differently, the conditional context for each token remains fixed by autoregressive decoding.

Discrete diffusion language models. Masked diffusion language models (Austin et al., 2021; Hoogeboom et al., 2021; Lou et al., 2024; Sahoo et al., 2024; Shi et al., 2024a; Ou et al., 2025), including recent large-scale dLLMs such as LLaDA and Dream (Nie et al., 2025; Ye et al., 2025), define reconstruction distributions over arbitrary masked subsets. Let $x = (x_1, \ldots, x_L)$ and let $x_{\setminus S}$ denote the sequence where positions in $S \subseteq \{1, \ldots, L\}$ are replaced by `[MASK]`. A dLLM models $p_\theta(x_i \mid x_{\setminus S})$ for $i \in S$ and is trained with a masked reconstruction objective of the form

$$\mathcal{L}(\theta) = -\,\mathbb{E}_{x,\lambda,S}\left[\frac{1}{|S|}\sum_{i\in S}\log p_\theta(x_i \mid x_{\setminus S})\right], \qquad \lambda \sim U(0,1).$$

The attacker-facing consequence is that S is not fixed. Any-order decodability allows the attacker to choose which reconstruction task to query, and parallel decodability returns all $|S|$ masked-token distributions from one forward pass (Uria et al., 2014; Germain et al., 2015; Yang et al., 2019; Hoogeboom et al., 2022; Ghazvininejad et al., 2019; Chang et al., 2022).

MIA for diffusion language models. SAMA (Chen et al., 2026), the first dLLM-specific MIA, studies fine-tuning membership by comparing a fine-tuned target with its pre-fine-tuning checkpoint. For each candidate sequence, it samples T random mask sets $S_1, \ldots, S_T$; for each S_t, the masked positions are reconstructed by both models, and their true-token log-probability gaps are averaged:

$$g_{\mathrm{SAMA}}(x) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|S_t|}\sum_{i\in S_t}\left[\log p_{M_{\mathrm{tgt}}}(x_i \mid x_{\setminus S_t}) - \log p_{M_{\mathrm{ref}}}(x_i \mid x_{\setminus S_t})\right]. \tag{2}$$

Q-Skew (Zhai et al., 2026) takes a different approach, characterizing token-level memorization asymmetry across diffusion states with a quantile-weighted skewness statistic. Both methods aggregate membership evidence over multiple masking conditions and therefore require repeated model evaluations. In contrast, JUMP identifies informative positions first and scores them jointly, reducing the number of model evaluations.

3 Threat Model

Setting and access. We consider fine-tuning membership inference for a target dLLM M_{tgt} adapted on a private set $\mathcal{D}_{\mathrm{ft}}$. Given a candidate sequence $x = (x_1, \ldots, x_L)$, the attacker determines whether $x \in \mathcal{D}_{\mathrm{ft}}$. We assume grey-box score access: for any chosen mask set S, the attacker can query the target with $x_{\setminus S}$ and obtain predictive logits or token distributions at the masked positions. We do not require access to target parameters, gradients, or internal activations. For the main JUMP setting, the corresponding pre-fine-tuning checkpoint M_{ref} is available under the same interface. We additionally study a target-only setting in Section 6, where only M_{tgt} is available. Detailed access assumptions are provided in Appendix A.1.

Scoring and evaluation. When the reference model is available, we define the token-level reconstruction advantage

$$\Delta_i(S) = \log p_{M_{\mathrm{tgt}}}(x_i \mid x_{\setminus S}) - \log p_{M_{\mathrm{ref}}}(x_i \mid x_{\setminus S}), \qquad i \in S. \tag{3}$$

All $|S|$ gaps are obtained with one target and one reference forward pass. Target-only attacks instead construct membership scores from the target reconstruction outputs alone. In both settings, the resulting sample-level score is thresholded for membership prediction. We report ROC-AUC, TPR at FPR $\in \{10\%, 1\%, 0.1\%\}$, and model forward evaluations per sample.

4 MIA on Fine-Tuned dLLMs with a Reference Model

JUMP has two variants depending on the available model access. In this section, we consider membership inference for fine-tuned dLLMs when a reference model is available. Specifically, we use the model before fine-tuning as the reference model. We first identify where membership signal appears (Section 4.1), then define the JUMP scoring rule (Section 4.2) and efficient token selector (Section 4.3). We consider the target-only setting without a reference model separately in Section 6.

4.1 Low-confidence tokens and membership signal

The any-order interface in dLLM raises a basic design question: which token positions should an attacker probe? We first study the ideal one-hole reference low-confidence score

$$q^{\star}(i \mid x) = \log p_{M_{\mathrm{ref}}}(x_i \mid x_{\setminus\{i\}}), \tag{4}$$

where smaller values indicate lower reference confidence in the true token under bidirectional context. This quantity is useful for analysis because it directly measures how uncertain the reference dLLM is about the true token x_i when only x_i is masked.

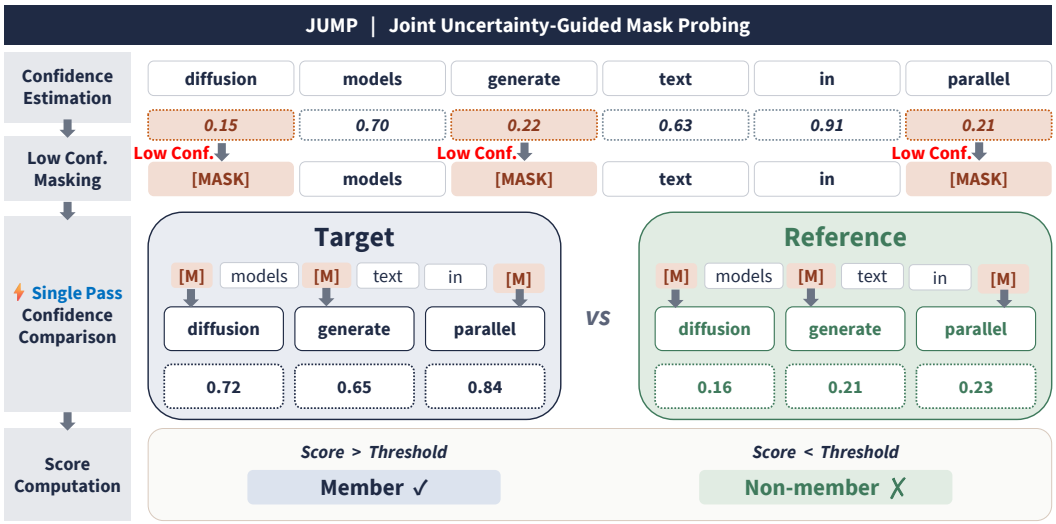


Figure 1: Overview of JUMP. The attack first selects informative positions under the reference model, then masks the selected positions jointly and scores them with the target and reference dLLMs. The final MIA statistic is computed from clipped token-level target/reference reconstruction gaps. In the visualization, general tokens are indicated by solid-line boxes, while confidence scores are denoted by dashed-line boxes.

To test whether this low-confidence score is related to membership signal, we run a diagnostic one-hole experiment on LLaDA-8B-Base fine-tuned on the Wikipedia (en) domain. For each candidate sequence, we use a 512-token evaluation window and, for every valid position i, mask only that position, i.e., $S = \{i\}$. We then record the one-hole reference probability $p_{M_{\text{ref}}}(x_i|x_{\setminus\{i\}})$ and the target/reference gap $\Delta_i(\{i\})$ from Eq. (3). Positions are ranked by Eq. (4) and partitioned into three groups of size $K = 64$: lowest-confidence positions, uniformly random positions, and highest-confidence positions.

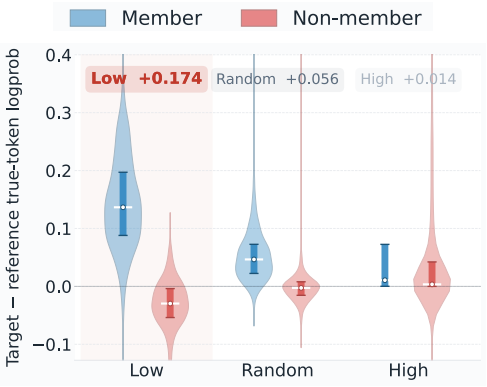


Figure 2: One-hole reconstruction gap across reference-confidence tiers. Low-confidence positions show the strongest separation.

Figure 2 shows that low-confidence positions carry substantially stronger membership signal. Under one-hole masking, the fine-tuned target reconstructs these positions much better on member sequences than on non-member sequences, producing a mean member/non-member gap difference of $+0.174$. Random and high-confidence positions show much weaker separation. This supports the use of reference-model confidence as a localization prior: positions where the reference model has low confidence are more informative for fine-tuning membership. Low reference confidence is not a per-token guarantee: in a six-domain audit, 23.7% of selected tokens have $|\Delta_i| \leq 0.05$, yet the aggregated gap between members and non-members remains positive in every domain (Appendix D.5).

The diagnostic procedure itself is not an efficient attack. Ranking all positions by Eq. (4) requires L one-hole queries to the reference model. Reusing those reference outputs, computing the corresponding gaps requires another L one-hole queries to the target, for $2L$ model forwards in total. This motivates an efficient attack that first identifies low-confidence positions and then scores them together through a single joint mask.

4.2 *Joint Uncertainty-Guided Mask Probing* (JUMP)

We propose *Joint Uncertainty-Guided Mask Probing* (JUMP), which turns the low-confidence localization principle above into an efficient membership inference attack through uncertainty-guided position selection and joint masked scoring. Given a sequence x of length L, let $[L] \subseteq \{1, \ldots, L\}$

denote the valid positions that may be masked and scored. JUMP uses a selector score $q_\phi(i \mid x)$ for each $i \in [L]$. The score $q_\phi(i \mid x)$ is a scalar prediction of reference-model confidence in the true token at position i, rather than a vocabulary distribution. Ideally, $q_\phi(i \mid x)$ approximates the one-hole confidence score $q^\star(i \mid x)$ in Eq. (4), so smaller values indicate lower reference confidence.

For a probing budget K, JUMP selects the K positions of lowest confidences:

$$\mathcal{H}_K(x;\phi) = \{i \in [L] \mid q_\phi(i \mid x) \text{ is one of the lowest } K \text{ values among all } i \in [L]\} \tag{5}$$

The selected positions are masked jointly, $S = \mathcal{H}_K(x;\phi)$, and the attacker computes $\Delta_i(S)$ for all $i \in S$ using Eq. (3). Since all positions in S are reconstructed in parallel, this joint probe returns K token-level membership signals with one target and one reference query. Thus, K affects only the number of averaged token gaps, not the target/reference scoring NFE.

The final statistic clips and averages the selected token gaps:

$$g_{\mathrm{JUMP}}(x; M_{\mathrm{tgt}}, M_{\mathrm{ref}}) = \frac{1}{K} \sum_{i \in \mathcal{H}_K(x;\phi)} \mathrm{clip}(\Delta_i(\mathcal{H}_K(x;\phi)), -\tau, \tau)\,, \qquad \tau = \log 1.5. \tag{6}$$

We clip to bound the effect of any single token, limiting heavy-tailed token gaps that can otherwise dominate the mean and create false positives at low FPR. The threshold $\tau = \log 1.5$ caps each token's contribution at a 1.5:1 probability ratio and is fixed throughout our evaluation; Appendix D.9 compares this readout with the unclipped mean.

Remark (NFE and efficiency). The joint scoring stage uses exactly two forward passes per sample, one through the target and one through the reference, independent of K and sequence length. The learned PRISM selector adds one reference-model pass, so the default attack uses three model forwards in total. SAMA with T masking steps instead uses $2T$ target/reference forwards.

4.3 Efficient Token Selection

The remaining question is how to obtain q_ϕ without running the expensive one-hole diagnostic in Section 4.1. In our default learned-selector variant, we use PRISM (Kim et al., 2025) as an efficient low-confidence token selector. Given an unmasked sequence, PRISM predicts a position-wise low-confidence score from the reference-model representations:

$$q_\phi(i \mid x) \approx q^\star(i \mid x) = \log p_{M_{\mathrm{ref}}}(x_i \mid x_{\setminus\{i\}}).$$

PRISM is a lightweight per-token quality head trained on the reference dLLM using the self-correction construction of Kim et al. (2025). Given a partially masked sequence, the reference predicts the hidden tokens. Some masks are filled with detached argmax predictions, and the quality head classifies each update as correct or incorrect. It therefore approximates context-conditioned update correctness, a natural proxy for one-hole reconstruction confidence. We train it with a correctness BCE and a small mask-ratio-weighted reconstruction loss.

This procedure generates token-level supervision from generic reference-side text, without membership labels, target information, or private examples. At attack time, one reference pass returns both the unmasking probabilities and PRISM scores $q_\phi(i \mid x)$. JUMP then selects the positions in Eq. (5) and jointly scores them as defined in Eq. (6).

Because the frozen reference model generates its own supervision, PRISM does not require target-domain data or a particular auxiliary corpus. Its training is a lightweight one-time procedure: in our setup, the selector checkpoint is obtained in under 35 minutes on four NVIDIA L40S GPUs and can then be reused across membership audits. Training details and controls over the auxiliary corpus and training-pool size are provided in Appendices A.4, D.4 and D.7.

We also consider PRISM-Free, which removes auxiliary selector training entirely. Instead of predicting one-hole confidence with a learned quality head, it ranks positions using the reference model's clean-input true-token probabilities, selects the lowest-confidence positions, and applies the same joint masked scoring rule. Despite this simpler selection mechanism, PRISM-Free remains competitive with the learned selector; we report the comparison in Section 5.2.

Table 1: Per-domain reference-assisted results on LLaDA-8B-Base and Dream-v0-Base-7B. We report ROC-AUC and TPR at fixed FPR (↑) with total NFE (↓). SAMA uses $T = 16$ (32 forwards) and JUMP uses 3; best quality values are in **bold**.

LLaDA-8B-Base

Method	ArXiv					GitHub					HackerNews				
	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)
	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE
Loss	0.52	0.12	0.01	0.00	16	0.59	0.20	0.04	0.01	16	0.51	0.11	0.01	0.01	16
ZLIB	0.52	0.13	0.01	0.00	16	0.61	0.23	0.07	0.02	16	0.51	0.10	0.01	0.01	16
SAMA	0.82	0.54	0.28	0.07	32	0.77	0.43	0.14	0.07	32	0.71	0.34	0.05	0.01	32
JUMP	**0.94**	**0.84**	**0.55**	**0.30**	3	**0.86**	**0.66**	**0.32**	**0.18**	3	**0.76**	**0.39**	**0.10**	**0.02**	3

Method	PubMed Central					Wikipedia (en)					Pile-CC				
	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)
	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE
Loss	0.52	0.13	0.01	0.00	16	0.51	0.10	0.01	0.00	16	0.51	0.11	0.01	0.00	16
ZLIB	0.52	0.13	0.01	0.00	16	0.51	0.10	0.01	0.00	16	0.52	0.13	0.02	0.00	16
SAMA	0.81	0.51	0.23	0.03	32	0.91	0.73	0.47	0.36	32	0.90	0.70	0.40	**0.20**	32
JUMP	**0.92**	**0.75**	**0.43**	**0.12**	3	**0.98**	**0.95**	**0.80**	**0.67**	3	**0.96**	**0.90**	**0.60**	0.10	3

Dream-v0-Base-7B

Method	ArXiv					GitHub					HackerNews				
	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)
	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE
Loss	0.54	0.13	0.01	0.00	16	0.60	0.21	0.05	0.01	16	0.54	0.12	0.01	0.01	16
ZLIB	0.54	0.14	0.01	0.00	16	0.63	0.24	0.07	0.03	16	0.54	0.11	0.02	0.01	16
SAMA	0.84	0.57	0.29	0.22	32	0.83	0.62	0.26	**0.12**	32	0.83	0.57	0.17	0.05	32
JUMP	**0.93**	**0.81**	**0.52**	**0.32**	3	**0.91**	**0.75**	**0.36**	0.01	3	**0.93**	**0.81**	**0.49**	**0.23**	3

Method	PubMed Central					Wikipedia (en)					Pile-CC				
	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)	Attack quality (↑)				Cost (↓)
	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE	AUC	T@10	T@1	T@0.1	NFE
Loss	0.54	0.14	0.03	0.00	16	0.53	0.12	0.01	0.01	16	0.53	0.13	0.01	0.00	16
ZLIB	0.53	0.14	0.03	0.00	16	0.53	0.13	0.02	0.01	16	0.54	0.14	0.02	0.00	16
SAMA	0.84	0.59	0.30	0.16	32	0.86	0.62	0.35	0.25	32	0.90	0.72	0.35	0.19	32
JUMP	**0.95**	**0.86**	**0.53**	**0.35**	3	**0.96**	**0.89**	**0.63**	**0.29**	3	**0.97**	**0.94**	**0.77**	**0.38**	3

5 EXPERIMENTS

Setup. We evaluate LLaDA-8B-Base (Nie et al., 2025) and Dream-v0-Base-7B (Ye et al., 2025) on six MIMIR domains: ArXiv, GitHub, HackerNews, Pile-CC, PubMed Central, and Wikipedia (en) (Duan et al., 2024). For each model and domain, the target is fine-tuned on 1,000 MIMIR training documents (members) and evaluated against 1,000 documents from the corresponding MIMIR test split (non-members). Reference-assisted attacks use the exact pre-fine-tuning checkpoint, and all attacks share the same target checkpoint and candidates.

Metrics and configurations. We compare with Loss, ZLIB (Carlini et al., 2021), and SAMA (Chen et al., 2026), and report ROC-AUC, TPR at FPR $\in \{10\%, 1\%, 0.1\%\}$, and the total number of model forward evaluations (NFE). JUMP uses a C4-trained PRISM selector, $K = 64$, and $\tau = \log 1.5$. Metrics are computed independently within each domain and then averaged with equal weight; samples are never pooled across domains or model families. Model-specific training and checkpoint details appear in Section A.3, and attack and metric details in Appendices A.5 and A.6.

5.1 MAIN RESULTS

Table 1 shows that JUMP improves AUC over SAMA in every domain. Mean AUC rises from 0.8193 to 0.9024 on LLaDA and from 0.8513 to 0.9415 on Dream. On LLaDA, mean TPR@1% FPR increases from 0.2625 to 0.4663 and TPR@0.1% FPR from 0.1220 to 0.2320; Dream shows the same aggregate improvement. Domain-level estimates at 0.1% FPR should be interpreted cautiously: with

1,000 non-members per domain, this operating point is determined by roughly one false positive and therefore has coarse tail resolution; the corresponding bootstrap intervals are reported in Appendix B.

The table also reports evaluation cost directly. SAMA with $T = 16$ uses 16 target and 16 reference evaluations, for 32 NFE per document. JUMP uses one reference pass for selection and one target/reference scoring pair, for 3 NFE. The selected-token budget K changes the number of token gaps returned by the joint query, not the scoring NFE. Measured runtime falls from 1.863 seconds per sample for SAMA to 0.184 for JUMP.

The trend persists in lower-exposure and mixed-domain controls: JUMP reaches 0.7316 versus SAMA's 0.6600 at two epochs and 0.7466 versus 0.7116 on the mixed-domain target. Bootstrap intervals and selector, exposure, and clipping controls are reported in Appendices B to D and D.3.

5.2 PRISM-Free results

Table 2: PRISM-Free equal-weight six-domain means.

Family	Method	AUC	T@10	T@1	T@0.1
LLaDA	SAMA	0.8193	0.5393	0.2625	0.1220
	PRISM-Free	0.8677	0.6527	0.3238	0.1443
	JUMP	**0.9024**	**0.7500**	**0.4663**	**0.2320**
Dream	SAMA	0.8513	0.6122	0.2862	0.1660
	PRISM-Free	0.9404	0.8322	0.4997	0.2198
	JUMP	**0.9415**	**0.8438**	**0.5505**	**0.2653**

PRISM-Free replaces the learned PRISM head with the reference model's clean-input true-token score and therefore requires no auxiliary selector training. It improves mean ROC-AUC over SAMA on both LLaDA (0.8677 versus 0.8193) and Dream (0.9404 versus 0.8513). On Dream it nearly matches JUMP in mean AUC (0.9404 versus 0.9415), while the learned selector remains stronger at low FPR and provides a larger gain on LLaDA. Thus, clean-text reference confidence is already a useful proxy for reconstruction difficulty, while the learned position-quality head makes the selected evidence more consistent. PRISM-Free still uses the exact pre-fine-tuning model for score calibration and is therefore reference-assisted, not target-only.

6 MIA on Fine-Tuned dLLMs without a Reference Model

We now consider membership inference when no reference model is available. Without a reference, the target–reference reconstruction difference used by JUMP is unavailable. Inspired by the token-level likelihood calibration of Min-K%++ (Zhang et al., 2025), we develop a target-only approach that extracts membership evidence from the target's own token predictions. Our approach focuses on relative token-level signals, filters out overly confident predictions, and exploits the parallel decoding of dLLMs for efficient probing.

6.1 Relative Token Preference

We begin with an ideal one-hole probe at position i, where only the observed token x_i is masked. Let

$$q_i = \log p_{M_{\text{tgt}}}(x_i \mid x_{\setminus\{i\}})$$

denote its reconstruction log-likelihood. Without a reference model, however, q_i alone is ambiguous: a low likelihood may arise either because x_i is intrinsically difficult to predict or because the target model has not learned a strong preference for it.

We therefore measure the likelihood of x_i relative to the model's own predictions at the same position. To capture the model's uncertainty over alternative tokens, we use the predictive entropy at position i:

$$H_i = -\sum_{v \in \mathcal{A}} p_{M_{\text{tgt}}}(v \mid x_{\setminus\{i\}}) \log p_{M_{\text{tgt}}}(v \mid x_{\setminus\{i\}}).$$

Combining this term with q_i, we define $q_i + H_i$ as the *relative token preference*. This score admits the following expected log-likelihood-ratio interpretation:

$$q_i + H_i = \mathbb{E}_{v \sim p_{M_{\text{tgt}}}(\cdot \mid x_{\setminus\{i\}})} \left[\log \frac{p_{M_{\text{tgt}}}(x_i \mid x_{\setminus\{i\}})}{p_{M_{\text{tgt}}}(v \mid x_{\setminus\{i\}})} \right].$$

Thus, $q_i + H_i$ measures how strongly the target model favors the observed token x_i over alternatives. Training on the candidate sequence strengthen the model's preference for x_i over competing tokens. We use $q_i + H_i$ as a token-level membership statistic and assess its discriminative value empirically.

6.2 Filtering Near-Deterministic Reconstructions

Real text contains many near-deterministic positions where the surrounding context alone can drive the true-token probability close to one. At such positions, the model can already assign very high probability to the observed token without having trained on the candidate sequence. These near-deterministic reconstructions can therefore yield weak membership signals even for member sequences. Including many such positions can dilute useful evidence in the document-level average.

This behavior becomes clear in the near-deterministic limit:

$$p_{M_{\text{tgt}}}(x_i \mid x_{\setminus\{i\}}) \to 1 \quad \Longrightarrow \quad q_i \to 0, \qquad H_i \to 0, \qquad q_i + H_i \to 0.$$

When the predictive mass is already concentrated on x_i, both the reconstruction log-likelihood and predictive entropy saturate, causing the relative token preference to vanish.

We therefore exclude positions whose one-hole true-token probability is larger than 0.99. This filtering does not select the hardest or lowest-confidence tokens; it only removes positions where the reconstruction has effectively saturated. We use the same threshold across all domains and both model families. Reconstruction-confidence distributions, saturation diagnostics, and threshold sensitivity are reported in Appendix F.

6.3 Efficient Grouped Approximation

The target-only formulation is defined from one-hole reconstruction quantities, so we would ideally evaluate q_i and H_i with only x_i masked. Exact one-hole evaluation, however, requires a separate masked query for every valid token position and is therefore expensive for practical auditing. One might instead use PRISM (Kim et al., 2025), as in the reference-assisted setting, to approximate token-level reconstruction difficulty. However, PRISM relies on model representations: in our main attack these are obtained from the reference model, which is unavailable here, while applying PRISM to the target would require internal target access beyond our predictive-output-only threat model. We therefore approximate the one-hole quantities directly with grouped masked queries.

Inspired by the cyclic sampling strategy of Q-Skew (Zhai et al., 2026), we partition the valid positions into 16 disjoint, approximately equal-sized groups $S_1, \ldots, S_{16}$ and mask each group once. For every $i \in S_g$, the corresponding target query provides

$$q_i(S_g) = \log p_{M_{\text{tgt}}}(x_i \mid x_{\setminus S_g}),$$
$$H_i(S_g) = -\sum_{v \in \mathcal{A}} p_{M_{\text{tgt}}}(v \mid x_{\setminus S_g}) \log p_{M_{\text{tgt}}}(v \mid x_{\setminus S_g}).$$

These grouped quantities approximate the one-hole q_i and H_i, with the other positions in S_g masked simultaneously. Because all masked positions are reconstructed in parallel, all valid positions are covered with only 16 target queries.

Let $g(i)$ denote the unique group index such that $i \in S_{g(i)}$. We apply the filtering rule from Section 6.2 using the grouped reconstruction probabilities and denote the retained positions by $\mathcal{R}(x)$. Target-Only JUMP then scores a document by averaging their relative token preferences:

$$g_{\text{TO-JUMP}}(x; M_{\text{tgt}}) = \frac{1}{|\mathcal{R}(x)|} \sum_{i \in \mathcal{R}(x)} \left[q_i(S_{g(i)}) + H_i(S_{g(i)}) \right].$$

Higher values indicate stronger membership evidence. Since filtering and scoring use the same grouped target outputs, no additional model evaluations are required.

Table 3: Target-only results (ROC-AUC / TPR@10% FPR, ↑). Random uses uniformly sampled positions; Clean Score selects positions by clean-input target true-token log-probability, while Ref. Guided uses reference-selected positions as a control. Masked Score and Target-Only JUMP share the same 16-way grouped outputs and $p_{\text{true}} \leq 0.99$ filtering rule; Masked Score uses the true-token log-probability q_i, while Target-Only JUMP uses the relative token preference $q_i + H_i$. Means are macro-averaged across domains. Best and second-best are bold and underlined.

LLaDA-8B-Base							
Method	**ArXiv**	**GitHub**	**HackerNews**	**PubMed**	**Wikipedia**	**Pile-CC**	**Mean**
Random	.558/.112	.584/.158	.534/.120	.553/.142	.553/.124	.537/.108	.553/.127
Clean Score	.574/.128	.564/.174	.530/.121	.559/.131	.580/.134	.555/.142	.560/.138
Ref. Guided	.576/.172	.578/.179	.533/.127	.578/.136	.559/.137	.546/.126	.562/.146
Masked Score	.602/.176	.577/.175	.541/.116	.598/.186	.604/.171	.591/.176	.585/.167
TO-JUMP	**.617/.214**	**.596/.206**	**.566/.148**	**.621/.211**	**.638/.221**	**.619/.197**	**.609/.200**
Dream-v0-Base-7B							
Method	**ArXiv**	**GitHub**	**HackerNews**	**PubMed**	**Wikipedia**	**Pile-CC**	**Mean**
Random	.600/.180	.632/.183	.581/.169	.571/.139	.587/.160	.568/.158	.590/.165
Clean Score	.631/.191	**.638/.238**	.611/.207	.600/.164	.616/.210	.591/.184	.614/.199
Ref. Guided	.630/.204	.636/.233	.609/.186	.616/.209	.606/.193	.610/.194	.618/.203
Masked Score	.658/**.239**	.637/.237	.653/.217	.633/**.222**	.647/**.251**	**.644**/.224	**.645/.232**
TO-JUMP	**.659**/.228	.581/.129	**.657/.219**	**.636**/.200	**.653**/.194	.639/**.228**	.638/.200

6.4 Target-Only Evaluation

We reuse the target checkpoints and the per-domain sets of 1,000 members and 1,000 non-members from Section 5, but do not use the pre-fine-tuning checkpoint. Metrics are computed independently within each domain and then macro-averaged.

Table 3 reports target-only membership inference performance across the six MIMIR domains on LLaDA and Dream. The first three rows compare position-selection strategies. Random positions and clean-confidence selection use only the final target model, while reference-guided selection uses a reference-derived position set but removes target–reference score calibration and is included only as a calibration control. The final two rows use the same 16-way grouped outputs and $p_{\text{true}} \leq 0.99$ filtering rule, allowing us to compare raw masked log-probability with relative token preference.

Target-Only JUMP achieves mean ROC-AUCs of 0.6095 on LLaDA and 0.6375 on Dream. It outperforms the random and clean-confidence target-only baselines on both model families, showing that membership information can be extracted from the target model's masked reconstruction distribution even without a pre-fine-tuning reference. On LLaDA, relative token preference improves over masked log-probability in all six domains, increasing the mean AUC from 0.5854 to 0.6095.

On Dream, masked log-probability is stronger on average (0.6454 versus 0.6375), with the largest difference on GitHub. Nevertheless, Target-Only JUMP remains stronger than the target-only position-selection baselines and improves over masked log-probability in four of the other five domains. Additional analyses of reconstruction confidence, filtering thresholds, and score normalization are provided in Appendix F.

7 Conclusion

JUMP turns the dLLM interface into a membership-auditing framework by identifying informative positions and evaluating them through parallel reconstruction. When the pre-fine-tuning checkpoint is available, JUMP localizes low-confidence positions and measures their fine-tuning-induced reconstruction gains against the reference, improving mean AUC to 0.902 on LLaDA and 0.942 on Dream while reducing SAMA's 32 NFE to 3. When only the final target is available, Target-Only JUMP uses grouped masking, filters near-deterministic positions, and aggregates relative token preference, reaching mean AUCs of 0.609 and 0.638. Together, these results show that dLLM reconstruction signals support membership auditing both with and without access to a reference.

AI USE STATEMENT

Generative AI tools were used only to improve the writing, formatting, and presentation of the manuscript. The authors reviewed all AI-assisted edits and take full responsibility for the final content of the paper.

ETHICS STATEMENT

Membership inference is dual use: it can support authorized privacy auditing, but it can also reveal whether a record participated in model adaptation. We evaluate publicly available model families on established benchmark corpora, report aggregate results, and do not release inferred membership decisions for identifiable records. The intended use of this work is controlled auditing by model developers, data owners, and authorized evaluators.

REPRODUCIBILITY STATEMENT

The main text defines the attack scores, mask construction, and query accounting. The appendix specifies the data splits, target-training protocol, model-access assumptions, hyperparameters, baselines, metrics, bootstrap procedure, runtime, ablations, and asset licenses. Domain-wise results are reported separately for both model families so that the stated averages can be reconstructed directly.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM CCS*, 2016.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *NeurIPS*, 2021.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX Security*, 2019.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *USENIX Security*, 2021.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy (S&P)*, 2022.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *ICLR*, 2023.

Hongyan Chang, Ali Shahin Shamsabadi, Kleomenis Katevas, Hamed Haddadi, and Reza Shokri. Context-aware membership inference attacks against pre-trained large language models. In *EMNLP*, 2025.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked generative image transformer. In *CVPR*, 2022.

Yuetian Chen, Kaiyuan Zhang, Yuntao Du, Edoardo Stoppa, Charles Fleming, Ashish Kundu, Bruno Ribeiro, and Ninghui Li. Membership inference attacks against fine-tuned diffusion language models. In *ICLR*, 2026.

Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. Do membership inference attacks work on large language models? In *COLM*, 2024.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In *ICML*, 2015.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP-IJCNLP*, 2019.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *NeurIPS*, 2021.

Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *ICLR*, 2022.

Matthew Jagielski, Om Thakkar, Florian Tramer, Daphne Ippolito, Katherine Lee, Nicholas Carlini, Eric Wallace, Shuang Song, Abhradeep Guha Thakurta, Nicolas Papernot, and Chiyuan Zhang. Measuring forgetting of memorized training examples. In *ICLR*, 2023.

Jaeyeon Kim, Seunggeun Kim, Taekyun Lee, David Z. Pan, Hyeji Kim, Sham Kakade, and Sitan Chen. Fine-tuning masked diffusion for provable self-correction. *arXiv preprint arXiv:2510.01384*, 2025.

Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *ICLR*, 2022.

Yunhui Long, Vincent Bindschaedler, Lei Wang, Diyue Bu, Xiaofeng Wang, Haixu Tang, Carl A Gunter, and Kai Chen. Understanding membership inferences on well-generalized learning models. *arXiv preprint arXiv:1802.04889*, 2018.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *ICML*, 2024.

Nils Lukas, Ahmed Salem, Robert Sim, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. Analyzing leakage of personally identifiable information in language models. In *IEEE Symposium on Security and Privacy (S&P)*, 2023.

Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schölkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. Membership inference attacks against language models via neighbourhood comparison. In *ACL Findings*, 2023.

Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks. In *EMNLP*, 2022.

Milad Nasr, Javier Rando, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from aligned, production language models. In *ICLR*, 2025.

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 *NeurIPS*, 2025.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *ICLR*, 2025.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *NeurIPS Datasets and Benchmarks Track*, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *NeurIPS*, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *NeurIPS*, 2024a.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. In *ICLR*, 2024b.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R. Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627b token cleaned and deduplicated version of RedPajama, 2023. URL `https://huggingface.co/datasets/cerebras/SlimPajama-627B`.

Thomas Steinke, Milad Nasr, and Matthew Jagielski. Privacy auditing with one (1) training run. In *NeurIPS*, 2023.

Jiashu Tao and Reza Shokri. Information-theoretic membership inference for granular quantification of memorization. In *ICLR*, 2026.

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *ICML*, 2014.

Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the importance of difficulty calibration in membership inference attacks. In *ICLR*, 2022.

Roy Xie, Junlin Wang, Ruomin Huang, Minxing Zhang, Rong Ge, Jian Pei, Neil Zhenqiang Gong, and Bhuwan Dhingra. Recall: Membership inference via relative conditional log-likelihoods. In *EMNLP*, 2024.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R. Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium (CSF)*, 2018.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *ICLR*, 2022.

Sajjad Zarifzadeh, Philippe Liu, and Reza Shokri. Low-cost high-power membership inference attacks. In *ICML*, 2024.

Shengfang Zhai, Leo Marchyok, Yuling Shi, Huanran Chen, Yinpeng Dong, Jiaheng Zhang, and Sanghyun Hong. Membership inference in fine-tuned diffusion language models via token-level memorization asymmetry. *arXiv preprint arXiv:2609.00873*, 2026.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Frank Yang, and Hai Li. Min-k%++: Improved baseline for pre-training data detection from large language models. In *ICLR*, 2025.

A Experimental Details

A.1 Threat Model and Access Assumptions

We consider two fine-tuning membership-auditing settings. In the primary reference-assisted setting, the corresponding pre-fine-tuning checkpoint is available, as in open-weight adaptation and developer-side privacy audits. Target-Only JUMP covers the complementary setting in which the paired pre-fine-tuning checkpoint is unavailable.

Reference-assisted access corresponds to two concrete scenarios. In known-lineage open-weight adaptation, a published base checkpoint is used to produce a disclosed derivative model. In authorized developer-side auditing, the model owner retains both the base and adapted checkpoints.

The default learned PRISM selector additionally uses reference-model representations during one-time offline training on generic public text; it uses no target parameters or gradients, private fine-tuning examples, membership labels, or evaluation examples. PRISM-Free removes this auxiliary training and hidden-state access and uses only model output scores for position selection and final scoring. It still uses the pre-fine-tuning reference and is therefore reference-assisted rather than target-only.

A.2 Original MIMIR Protocol and Member Definition

We use ArXiv, GitHub, HackerNews, Pile-CC, PubMed Central, and Wikipedia (en) from MIMIR. For each domain, the 1,000 fine-tuning documents constitute the member set, and the corresponding 1,000 MIMIR test documents constitute the non-member set. We verify row order and content hashes against the target train/test manifests and find zero exact member/non-member hash overlap in all six domains.

A.3 Target Model Training and Checkpoint Selection

LLaDA. Each LLaDA target is initialized from `GSAI-ML/LLaDA-8B-Base` and fine-tuned separately on one domain's 1,000 member documents. We use AdamW with weight decay 0.1 and a configured learning rate of 5×10^{-5}. The scheduler is linear with 500 optimizer-update warmup steps. Because the training runs contain at most 80 optimizer updates, the realized schedule remains within the warmup phase and reaches a learning rate of 8×10^{-6} at step 80. We use maximum sequence length 512 and random seed 42.

Training uses four GPUs in bf16 with DeepSpeed ZeRO-3 and gradient checkpointing. The per-device batch size is 1 with 12 gradient-accumulation steps, giving an effective batch size of 48. All LLaDA targets are trained with the same maximum four-epoch budget. We evaluate checkpoints at approximately one-epoch intervals and select the checkpoint with the lowest validation loss. Here the validation set is the corresponding MIMIR test split, which also serves as the non-member evaluation set. These examples receive no gradient updates, but they can influence which checkpoint is retained. This selects checkpoint 61 for GitHub and HackerNews and checkpoint 80 for ArXiv, Pile-CC, PubMed Central, and Wikipedia, corresponding to approximately three and four epochs, respectively. Checkpoint suffixes denote optimizer global steps. We report the validation-set reuse explicitly because the selected checkpoints are used for the main results. The fixed epoch-1 and epoch-2 controls disable evaluation-based checkpoint selection and use the predetermined endpoints at steps 20 and 40.

Dream. Dream targets are initialized from `Dream-org/Dream-v0-Base-7B` and use the same data size, optimizer, learning-rate schedule, sequence length, seed, distributed training setup, batch size, and gradient accumulation as LLaDA. All six Dream domains use checkpoint 80, corresponding to approximately four epochs.

Checkpoint-selection robustness. GitHub and HackerNews are the only two LLaDA domains for which the common checkpoint-selection rule retains an earlier checkpoint. To test whether the reported results depend on these earlier endpoints, we continue the same training trajectories to the predetermined checkpoint-80 endpoint and reevaluate the attacks. Table 4 compares the selected checkpoints used in the main evaluation with the fixed four-epoch endpoints.

Table 4: Checkpoint-selection robustness on GitHub and HackerNews. "Selected" denotes the checkpoint used in the main evaluation (∼3 epochs), while "Epoch 4" evaluates checkpoint 80.

		SAMA				JUMP				Target-Only JUMP			
Domain	Checkpoint	AUC	T@10	T@1	T@0.1	AUC	T@10	T@1	T@0.1	AUC	T@10	T@1	T@0.1
GitHub	Selected	0.77	0.43	0.15	0.07	0.86	0.66	0.32	0.18	0.60	0.21	0.03	0.01
	Epoch 4	0.83	0.58	0.25	0.08	0.91	0.77	0.50	0.32	0.63	0.24	0.04	0.01
HackerNews	Selected	0.71	0.34	0.05	0.01	0.76	0.39	0.10	0.02	0.57	0.15	0.02	0.00
	Epoch 4	0.80	0.51	0.14	0.02	0.86	0.64	0.21	0.08	0.63	0.20	0.03	0.00

The fixed four-epoch endpoints strengthen the attack signal rather than weakening it. For both GitHub and HackerNews, JUMP and SAMA improve across all reported metrics, while Target-Only JUMP also improves in ROC-AUC and TPR@10% FPR. The ordering of JUMP over SAMA is preserved at the fixed four-epoch endpoint. We nevertheless retain the selected checkpoints in the main evaluation to follow the common SAMA-matched checkpoint-selection rule rather than replacing them post hoc with stronger endpoints.

A.4 PRISM Selector Training

The default LLaDA PRISM selector is trained on a materialized 200k-sample English C4 subset (Raffel et al., 2020). We split 190k training sequences and 10k held-out sequences; after preprocessing, 189,919 training sequences remain. The selector is initialized from the exact base backbone and trained for one epoch on four L40S GPUs in fp16. The per-GPU batch size is 4 with two gradient-accumulation steps, for an effective batch size of 32. We use maximum length 256, AdamW with learning rate 10^{-4}, zero weight decay, and seed 42.

Only the PRISM quality head and LoRA adapters are trained; the reference backbone is otherwise frozen. This updates 29.38M of 8.04B parameters (0.37%). LoRA is applied to query, key, and value projections with rank 16, alpha 16, and dropout 0.1. For each clean sequence x_0, a masking ratio $t \sim \mathrm{Uniform}(10^{-3}, 1)$ is sampled, valid tokens are independently masked, and up to eight masked positions are filled with argmax predictions. The quality head predicts whether each update matches the ground-truth token. The retained objective is

$$\mathcal{L}_{\mathrm{PRISM}} = \mathcal{L}_{\mathrm{BCE}} + 0.1\,\mathcal{L}_{\mathrm{CE}},$$

where the reconstruction cross-entropy term is weighted by the inverse masking ratio. The fixed one-epoch checkpoint is used without validation-based selector checkpoint search. Training takes 2,968 optimizer steps and approximately 1,613 seconds (26.9 minutes).

A.5 SAMA Configuration and Forward Counting

The main SAMA baseline uses $T = 16$ progressive masking steps. At each step, the target and exact-base reference are evaluated once under the current masked context. From the resulting token losses, the implementation samples 128 local random subsets of 10 valid positions and computes the subset-level comparison votes in post-processing. Thus, each step requires one target and one reference forward pass, giving $2T = 32$ NFE per example. The closest-budget control uses $T = 2$, or four forward calls. All methods use the same 512-token windows and evaluation manifests. Wall-clock time is normalized to seconds per evaluation sample and reported separately in Appendix D.

A.6 Default Configuration and Metric Estimation

Reference-assisted JUMP selects $K = 64$ positions and applies symmetric token-wise clipping with $\tau = \log 1.5$. Target-Only JUMP uses a deterministic balanced 16-way grouped reconstruction. Positions whose grouped true-token reconstruction probability satisfies $p_{\mathrm{true}} \geq 0.99$ are treated as near-deterministic and excluded from the document score. Over the remaining positions, Target-Only JUMP averages the entropy-centered relative token preference $q_i + H_i$. No sigma normalization is applied in the final Target-Only JUMP score. The Masked Score baseline uses the same grouped measurements and the same $p_{\mathrm{true}} < 0.99$ support, but averages q_i alone. The reference-assisted probing-budget and clipping ablations appear in Appendices D.8 and D.9.

We report full-sample point estimates on the complete member/non-member sets. Each method is evaluated on the same 1,000 members and 1,000 non-members per domain. Metrics are computed independently within each domain and summarized by their equal-weight six-domain mean; samples are never pooled across domains or model families for the primary results.

For confidence intervals, we use class-stratified bootstrap resampling. Within each domain, the 1,000 members and 1,000 non-members are independently resampled with replacement. For the target-only macro confidence intervals, this procedure is repeated for 2,000 bootstrap replicates, and the six domain-level metrics are averaged replicate-wise before computing percentile 95% intervals. Reference-assisted confidence intervals are described in Appendix B.

A.7 TARGET-ONLY JUMP PROTOCOL

After removing padding and structural special tokens, we assign the remaining valid positions to a deterministic balanced 16-way partition

$$\mathcal{P}(x) = \{S_1, \ldots, S_{16}\},$$

where group sizes differ by at most one. Each valid position belongs to exactly one group, and each group is masked once. Thus, every valid position is evaluated in exactly one grouped reconstruction context.

For a position $i \in S_{g(i)}$, the corresponding grouped reconstruction provides the true-token log-probability

$$q_i(S_{g(i)}) = \log p_{M_{\mathrm{tgt}}}\left(x_i \mid x_{\setminus S_{g(i)}}\right),$$

and the predictive entropy

$$H_i(S_{g(i)}) = -\sum_{v \in \mathcal{A}} p_{M_{\mathrm{tgt}}}\left(v \mid x_{\setminus S_{g(i)}}\right) \log p_{M_{\mathrm{tgt}}}\left(v \mid x_{\setminus S_{g(i)}}\right).$$

We write the corresponding grouped true-token reconstruction probability as

$$p_i(S_{g(i)}) = \exp\left(q_i(S_{g(i)})\right).$$

The 16-way partition yields 16 sequence-equivalent masked reconstruction contexts per document. In our implementation, eight groups are processed together in each batch, resulting in two physical batched model invocations. We report the 16 evaluated contexts as the attack NFE rather than treating physical batching as a reduction in the number of reconstruction contexts.

We exclude near-deterministic positions and define the retained set as

$$\mathcal{R}(x) = \left\{i \in V(x) : p_i(S_{g(i)}) < 0.99\right\},$$

where $V(x)$ denotes the set of valid positions. The same threshold is used for every domain and both model families.

Target-Only JUMP computes

$$g_{\text{TO-JUMP}}(x; M_{\mathrm{tgt}}) = \frac{1}{|\mathcal{R}(x)|} \sum_{i \in \mathcal{R}(x)} \left[q_i(S_{g(i)}) + H_i(S_{g(i)})\right].$$

Higher values indicate stronger membership evidence. Both filtering and scoring are obtained from the same 16 grouped reconstructions, so the entropy-centered readout introduces no additional masked contexts. Sigma normalization is not used in the final method.

Masked Score uses exactly the same grouped measurements and retained positions but removes the entropy term:

$$g_{\mathrm{Masked}}(x; M_{\mathrm{tgt}}) = \frac{1}{|\mathcal{R}(x)|} \sum_{i \in \mathcal{R}(x)} q_i(S_{g(i)}).$$

This comparison isolates the contribution of the entropy-centered $q + H$ readout while keeping the reconstruction contexts and filtering rule fixed.

The first three controls in Table 3 retain their original $K = 64$ definitions and are not redefined by the $p_{\text{true}} < 0.99$ filter. Random samples 64 valid positions without replacement using a document-keyed hash, masks them jointly, and averages the resulting target true-token log-probabilities. Clean Score ranks positions by the target model's clean-input true-token score, selects the 64 lowest-scoring positions, and evaluates them with a joint mask. Ref. Guided uses the 64 positions selected by PRISM but removes target–reference score calibration and averages only the target true-token log-probabilities. Thus, Masked Score and Target-Only JUMP are the two methods that share the final 16-way grouped measurements and $p_{\text{true}} < 0.99$ support.

B Bootstrap Confidence Intervals for Main Results

We report 95% percentile bootstrap confidence intervals for both reference-assisted and target-only JUMP on LLaDA-8B-Base and Dream-v0-Base-7B. The intervals quantify uncertainty due to the finite member/non-member evaluation sets under fixed trained models and fixed attack configurations.

For every model family, access setting, and domain, we independently resample the 1,000 member and 1,000 non-member examples with replacement for 2,000 class-stratified bootstrap replicates. We recompute ROC-AUC and TPR at FPR $\in \{10\%, 1\%, 0.1\%\}$ for each replicate and report percentile 95% confidence intervals. We do not assume Normality of the bootstrap distributions. Domain-level metrics are never pooled across domains. For the equal-weight six-domain mean, each domain is resampled independently and the six domain-level metrics are averaged replicate-wise; the reported macro interval is the percentile interval of these replicate-wise means.

B.1 Reference-Assisted JUMP

Tables 5 and 6 report per-domain uncertainty for the reference-assisted JUMP results used in the main evaluation. The point estimates reported in this appendix are the full-sample metrics computed from the same saved per-example scores used for the main results; Table 1 reports the corresponding values rounded to two decimal places. On LLaDA, the six-domain mean ROC-AUC is 0.9024 with a 95% confidence interval of $[0.8972, 0.9080]$. On Dream, the corresponding mean is 0.9415 with a 95% confidence interval of $[0.9376, 0.9456]$.

B.2 Target-Only JUMP

For Target-Only JUMP, we bootstrap the final method defined in Appendix A.7. Positions with grouped true-token reconstruction probability $p_{\text{true}} \geq 0.99$ are excluded, and the document score averages $q_i + H_i$ over the retained positions. No document-relative ρ threshold or sigma normalization is used in these results.

Tables 7 and 8 report the resulting per-domain intervals; the equal-weight six-domain intervals are summarized below. The mean ROC-AUC is 0.6095 on LLaDA with a 95% confidence interval of $[0.5994, 0.6192]$, and 0.6375 on Dream with a 95% confidence interval of $[0.6271, 0.6469]$.

Confidence intervals become wider at stricter operating points, especially TPR@0.1% FPR. With 1,000 non-members per domain, an FPR of 0.1% corresponds to approximately one false positive, so this metric is necessarily sensitive to the most extreme non-member scores. ROC-AUC and the less strict TPR operating points depend on substantially more of the score distribution and therefore have tighter intervals. These bootstrap intervals capture evaluation-set uncertainty only; they do not include variability from retraining the target model or the learned selector.

Table 7: Per-domain Target-Only JUMP results on LLaDA-8B-Base, with 95% bootstrap CIs in brackets.

Domain	AUC	T@10	T@1	T@0.1
ArXiv	0.62 [.59,.64]	0.21 [.17,.26]	0.03 [.01,.06]	0.01 [.00,.02]
GitHub	0.60 [.57,.62]	0.21 [.17,.24]	0.03 [.02,.05]	0.01 [.00,.02]
HackerNews	0.57 [.54,.59]	0.15 [.12,.19]	0.02 [.01,.03]	0.00 [.00,.02]
PubMed Central	0.62 [.60,.64]	0.21 [.17,.26]	0.03 [.02,.06]	0.01 [.00,.02]
Wikipedia (en)	0.64 [.62,.66]	0.22 [.18,.26]	0.03 [.02,.05]	0.01 [.00,.02]
Pile-CC	0.62 [.59,.64]	0.20 [.16,.24]	0.02 [.01,.04]	0.01 [.00,.01]

Table 5: Per-domain JUMP results on fine-tuned LLaDA-8B-Base, with 95% bootstrap CIs in brackets.

Domain	AUC	T@10	T@1	T@0.1
ArXiv	0.94 [.93,.95]	0.84 [.80,.86]	0.55 [.45,.60]	0.30 [.10,.46]
GitHub	0.86 [.84,.87]	0.66 [.62,.70]	0.32 [.24,.44]	0.18 [.00,.25]
HackerNews	0.76 [.74,.78]	0.39 [.35,.45]	0.10 [.07,.16]	0.02 [.00,.08]
PubMed Central	0.92 [.90,.93]	0.75 [.71,.80]	0.43 [.31,.52]	0.12 [.04,.33]
Wikipedia (en)	0.98 [.97,.99]	0.95 [.94,.97]	0.80 [.76,.85]	0.67 [.61,.78]
Pile-CC	0.96 [.95,.97]	0.90 [.88,.93]	0.60 [.52,.67]	0.10 [.01,.53]

Table 6: Per-domain JUMP results on fine-tuned Dream-v0-Base-7B, with 95% bootstrap CIs in brackets.

Domain	AUC	T@10	T@1	T@0.1
ArXiv	0.93 [.92,.94]	0.81 [.78,.84]	0.52 [.39,.58]	0.32 [.22,.40]
GitHub	0.91 [.89,.92]	0.75 [.69,.80]	0.36 [.28,.48]	0.01 [.00,.29]
HackerNews	0.93 [.92,.94]	0.81 [.77,.83]	0.49 [.28,.58]	0.23 [.15,.30]
PubMed Central	0.95 [.94,.96]	0.86 [.83,.89]	0.53 [.49,.66]	0.35 [.28,.51]
Wikipedia (en)	0.96 [.95,.97]	0.89 [.87,.92]	0.63 [.51,.73]	0.29 [.22,.52]
Pile-CC	0.97 [.97,.98]	0.94 [.92,.95]	0.77 [.66,.81]	0.38 [.13,.68]

Table 8: Per-domain Target-Only JUMP results on Dream-v0-Base-7B, with 95% bootstrap CIs in brackets.

Domain	AUC	T@10	T@1	T@0.1
ArXiv	0.66 [.63,.68]	0.23 [.18,.28]	0.04 [.02,.08]	0.02 [.01,.03]
GitHub	0.58 [.56,.61]	0.13 [.10,.17]	0.01 [.00,.02]	0.00 [.00,.01]
HackerNews	0.66 [.63,.68]	0.22 [.19,.25]	0.04 [.01,.07]	0.00 [.00,.02]
PubMed Central	0.64 [.61,.66]	0.20 [.16,.25]	0.02 [.00,.04]	0.00 [.00,.01]
Wikipedia (en)	0.65 [.63,.68]	0.19 [.15,.25]	0.02 [.00,.04]	0.00 [.00,.01]
Pile-CC	0.64 [.62,.66]	0.23 [.18,.27]	0.04 [.02,.06]	0.01 [.00,.02]

C Training Exposure and Mixed-Domain Controls

C.1 Original-MIMIR Exposure and Memorization Controls

We train fixed one- and two-epoch LLaDA targets using the same members, non-members, optimizer, learning rate, batch size, sequence length, seed, K, and τ as the main evaluation. Evaluation-based checkpoint selection is disabled for these controls. The "selected ≤ 4" row uses the loss-selected checkpoints from the maximum-four-epoch training schedule.

Table 9: Attack performance as fine-tuning exposure increases. Values are six-domain means.

Exposure	Loss AUC	ZLIB AUC	SAMA AUC	JUMP AUC	JUMP TPR@1%
Epoch 1	0.5020	0.5055	0.5137	**0.5263**	0.0127
Epoch 2	0.5056	0.5091	0.6600	**0.7316**	0.0942
Selected ≤ 4	0.5273	0.5311	0.8193	**0.9024**	0.4663

Table 10: Member/non-member reconstruction losses for the same exposure controls. "Held-out gain" is the reduction in non-member loss relative to the base model.

Exposure	Member loss	Non-member loss	Gap	Held-out gain
Epoch 1	2.4271	2.4432	0.0161	0.0020
Epoch 2	2.3878	2.4179	0.0301	0.0273
Selected ≤ 4	2.2604	2.3453	0.0850	0.0998

As shown in Tables 9 and 10, absolute leakage and the member/non-member reconstruction-loss gap grow with exposure. At one epoch, all attacks are close to chance; at two epochs, JUMP already separates members more strongly than SAMA (0.7316 vs. 0.6600 AUC); the gap is largest for the selected ≤ 4 targets. At the same time, held-out reconstruction improves rather than collapses. Thus, the main signal is not simply a consequence of a target that fits members while losing all domain utility; it reflects a token-local advantage that becomes more consistent with exposure.

C.2 Mixed-Domain Fine-Tuning

We concatenate 1,000 members from each original MIMIR domain and fine-tune one LLaDA-8B target for one epoch on 6,000 members. Evaluation remains within domain: each domain's members are compared with its own 1,000 disjoint non-members, and we report the equal-weight mean of the six domain metrics. This experiment jointly changes corpus size, domain mixture, and optimizer-step count; it is a heterogeneous-target control, not a pure dataset-size ablation.

Table 11: Mixed six-domain target. The primary result is the equal-weight mean over six within-domain ROCs.

Method	NFE	AUC	TPR@10%	TPR@1%	TPR@0.1%
SAMA ($T = 2$)	4	0.6283	0.2482	0.0515	0.0197
SAMA ($T = 16$)	32	0.7116	0.3660	0.1265	0.0510
JUMP	3	**0.7466**	**0.4192**	**0.1450**	**0.0572**

JUMP is strongest at all reported operating points despite using fewer forwards. For completeness, a single pooled ROC over all 6,000 members and 6,000 non-members gives AUC 0.7199. We treat this as a secondary diagnostic because a pooled threshold also measures cross-domain score calibration and should not replace the primary six-domain mean.

D Reference-Assisted JUMP Ablations

D.1 Selector-Backbone Transfer

The exact-base model plays two distinct roles in the default attack: it calibrates the final gap between the target and reference and supplies representations to the learned selector. To test selector dependence without changing calibration, we fix the fine-tuned LLaDA-8B target and its exact pre-fine-tuning scoring reference, then change only the model used to select positions.

Table 12: Selector-backbone transfer with the target and exact-base scoring pair fixed.

Learned selector	AUC	TPR@10%	TPR@1%	TPR@0.1%
LLaDA-8B-Base	0.9024	0.7500	0.4663	0.2320
Distilled 5.36B	0.8966	0.7245	0.4090	0.1885
LLaDA-1.5 (8B-class)	0.9035	0.7470	0.4548	0.3152
LLaDA-8B-Instruct	0.9040	0.7470	0.4655	0.2550

Mean AUC varies by less than 0.008 across the compatible selector backbones. This indicates that uncertainty-guided position selection is not tied to one selector checkpoint when the exact target/base scoring pair is retained. Because the distilled 5.36B checkpoint also differs in distillation, pruning, and training history, it serves as a robustness control rather than a size-scaling experiment.

D.2 Non-Exact Scoring References

Changing the scoring reference is a different experiment. For a non-exact reference M_R,

$$\log p_{M_{\text{tgt}}} - \log p_{M_R} = (\log p_{M_{\text{tgt}}} - \log p_{M_{\text{base}}}) + (\log p_{M_{\text{base}}} - \log p_{M_R}),$$

so checkpoint mismatch is added directly to the fine-tuning-induced signal. We fix the target and exact 8B learned selector, then replace only the model used in the final score.

The non-exact models preserve some above-chance ranking information but sharply weaken calibration, especially at strict FPRs. This result supports the stated scope: JUMP is designed for an exact

Table 13: Diagnostic results with non-exact scoring references.

Scoring reference	AUC	TPR@10%	TPR@1%	TPR@0.1%
Exact pre-FT LLaDA-8B	0.9024	0.7500	0.4663	0.2320
Distilled 5.36B	0.6052	0.2137	0.0233	0.0067
LLaDA-1.5	0.6949	0.3422	0.0666	0.0182
LLaDA-8B-Instruct	0.6953	0.3448	0.0670	0.0190

before/after fine-tuning comparison. Unknown-lineage attacks require an additional cross-model calibration strategy; simply substituting an approximate checkpoint is not a like-for-like implementation of the proposed score.

D.3 Cross-Family Selector and Joint-Probing Controls

We compare the same selector and joint-probing controls on LLaDA and Dream. For all joint-probing variants, we fix $K = 64$, the clipping threshold $\tau = \log 1.5$, and the clipped target-to-reference score; only the position selection strategy changes. Random Joint samples positions uniformly, Entropy Top selects positions with high reference uncertainty, RareToken uses C4 token frequencies, and JUMP uses the learned PRISM selector. SAMA ($T = 16$) is included as the previous dLLM baseline.

Table 14: Selector and joint-probing controls across model families (six-domain mean). Joint variants use $K = 64$ and $\tau = \log 1.5$; RareToken uses C4 frequencies.

Family	Method	ROC-AUC	TPR@10%	TPR@1%	TPR@0.1%
LLaDA	Random Joint	0.8173	0.5297	0.2073	0.0890
	Entropy Top	0.8867	0.7043	0.3650	0.1453
	RareToken (C4-freq)	0.8235	0.5489	0.2078	0.0394
	SAMA ($T = 16$)	0.8193	0.5393	0.2625	0.1220
	JUMP	**0.9024**	**0.7500**	**0.4663**	**0.2320**
Dream	Random Joint	0.8700	0.6337	0.2927	0.1365
	Entropy Top	0.8976	0.7190	0.3207	0.0958
	RareToken (C4-freq)	0.8175	0.5112	0.1468	0.0388
	SAMA ($T = 16$)	0.8513	0.6122	0.2862	0.1660
	JUMP	**0.9415**	**0.8438**	**0.5505**	**0.2653**

Across both model families, a single joint mask already yields useful signal, but the choice of positions matters substantially. Random Joint and RareToken are consistently weaker, while uncertainty-based selection improves the attack. Learned JUMP gives the strongest mean result on both families, supporting the same position-selection principle across the two architectures.

D.4 Selector-Corpus Transfer

We train PRISM selectors on matched 200k-example subsets of C4 (Raffel et al., 2020), SlimPajama (Soboleva et al., 2023), and FineWeb (Penedo et al., 2024), and keep the downstream target/reference scoring pipeline fixed.

Table 15: Six-domain mean AUC for matched selector-training corpora.

Selector corpus	Mean ROC-AUC
C4	0.9024
SlimPajama	0.8983
FineWeb	0.8972

The C4, SlimPajama, and FineWeb selectors obtain six-domain mean AUCs of 0.9024, 0.8983, and 0.8972, respectively. The 0.0052 spread in Figure 3 indicates that the learned selector is not strongly tied to one generic corpus. Specialized terminology, identifiers, and numbers can still be ranked as difficult under a generic reference, so selector transfer does not require domain-specific membership examples. Even the weakest of the three selectors remains 0.0779 AUC above SAMA.

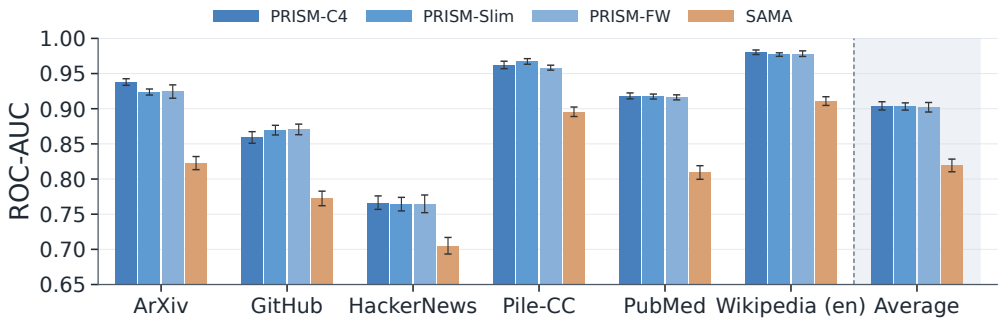


Figure 3: **Selector-corpus transfer.** Domain-wise ROC-AUC and the six-domain mean for PRISM selectors trained on three generic corpora. SAMA is shown as the reference-assisted dLLM baseline.

D.5 Low-Confidence Tokens Are a Localization Prior

Reference low confidence does not guarantee a large improvement between the target and reference at every selected token. A token may remain intrinsically difficult after fine-tuning, or the target may learn a domain-level pattern without memorizing that specific occurrence. Across 256k selected token rows from the six original MIMIR domains, 23.7% have $|\Delta_i| < 0.05$. Nevertheless, the aggregate gap between members and non-members is positive in all six domains. This supports the intended interpretation of PRISM: it raises the density of informative probes, and the sequence-level statistic obtains power by combining many small, directionally consistent advantages.

D.6 Rarity and Selector Overlap

To avoid transductive use of the evaluation pool, RareToken uses an external unigram frequency table computed from 30,000 C4 sequences. It selects the K valid positions with the lowest external token frequencies and then applies the same joint masking and clipped scoring difference between the target and reference as JUMP. The frequency table is computed entirely from external C4 text and uses no evaluation examples or membership labels.

Table 16: LLaDA selector diagnostics (six-domain mean); RareToken uses C4 frequencies.

Method	ROC-AUC	TPR@10%	TPR@1%	TPR@0.1%
RareToken (C4-freq)	0.8235	0.5489	0.2078	0.0394
SAMA ($T = 16$)	0.8193	0.5393	0.2625	0.1220
JUMP	**0.9024**	**0.7500**	**0.4663**	**0.2320**

Table 16 shows that external-corpus rarity is weaker than JUMP at every operating point, especially at TPR@0.1% FPR. SAMA provides the prior dLLM baseline. The overlap analysis below compares JUMP with random, entropy-based, and rarity-based position sets.

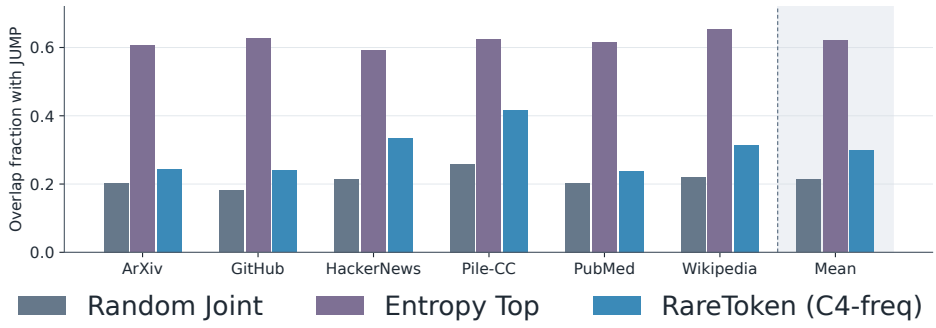


Figure 4: Overlap between JUMP-selected positions and alternative selector rules. Entropy Top has the largest overlap, while RareToken and Random Joint overlap much less, indicating that the learned selector is not equivalent to a single hand-designed heuristic.

Averaged over domains, Entropy Top overlaps with JUMP on 62.0% of positions, RareToken on 29.8%, and Random Joint on 21.3%. The partial overlap is consistent with PRISM learning contextual reconstruction difficulty rather than only rarity or clean-text probability.

D.7 Selector Training Sample-Size Ablation

We ablate the amount of C4 data used to train the PRISM head. Using the same LLaDA backbone and selector-training recipe, we train heads on 20k, 50k, 100k, 200k, 400k, and 500k examples. All selectors are evaluated with the main JUMP pipeline: $K = 64$, a single joint masked query, the difference between target and reference true-token log-probabilities, and symmetric clipping at $\tau = \log 1.5$.

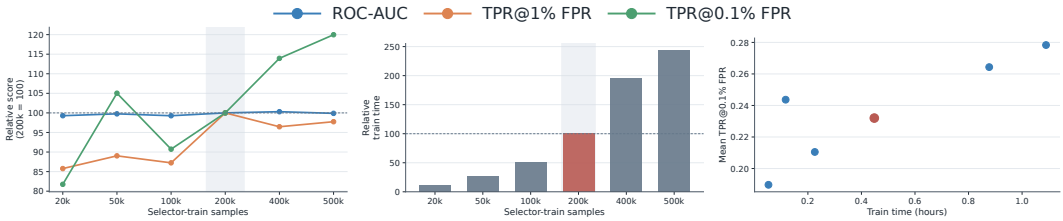


Figure 5: Effect of selector-training set size. Performance is already strong at small scales and changes only modestly beyond 200k examples. The 200k selector provides a balanced operating point, while larger sets yield small metric-specific gains at higher training cost.

Table 17: C4 selector-training sample-size ablation, averaged over six LLaDA domains.

C4 samples	ROC-AUC	TPR@10%	TPR@1%	TPR@0.1%	Train time
20k	0.8960	0.7303	0.4000	0.1897	190s
50k	0.9002	0.7437	0.4152	0.2437	421s
100k	0.8959	0.7195	0.4068	0.2105	813s
200k	0.9024	**0.7500**	**0.4663**	0.2320	1,613s
400k	**0.9052**	0.7477	0.4498	0.2643	3,158s
500k	0.9014	0.7432	0.4558	**0.2783**	3,922s

For reference, the main SAMA configuration uses $T = 16$ ($2T = 32$ target/reference forwards) and takes 1.863 seconds per sample. Its online cost reaches the default 200k PRISM training time of 1,613 seconds after approximately 866 audited samples; over the 12,000-example suite, selector training amortizes to 0.134 seconds per sample.

Table 17 and Figure 5 show that selector quality is already high with 20k examples and largely saturates around 200k. The 200k selector reproduces the main JUMP result and achieves the best TPR@10% and TPR@1% FPR among the tested budgets. Increasing the corpus to 400k slightly improves mean ROC-AUC, while 500k improves TPR@0.1% FPR; neither larger budget improves all metrics simultaneously. We therefore use 200k as a balanced default that avoids the approximately $1.96\times$ and $2.43\times$ training-time increases of the 400k and 500k selectors.

D.8 Selected-Token Budget Sweep

We sweep $K \in \{32, 64, 128\}$ with $\tau = \log 1.5$ fixed, while holding the target checkpoints, selector, evaluation examples, and aggregation rule constant.

Table 18: Selected-token budget sweep with $\tau = \log 1.5$, averaged over six LLaDA domains.

K	ROC-AUC	TPR@10%	TPR@1%	TPR@0.1%
32	0.8880	0.7100	0.3995	0.1798
64	**0.9024**	**0.7500**	**0.4663**	**0.2320**
128	0.8650	0.6525	0.2430	0.0442

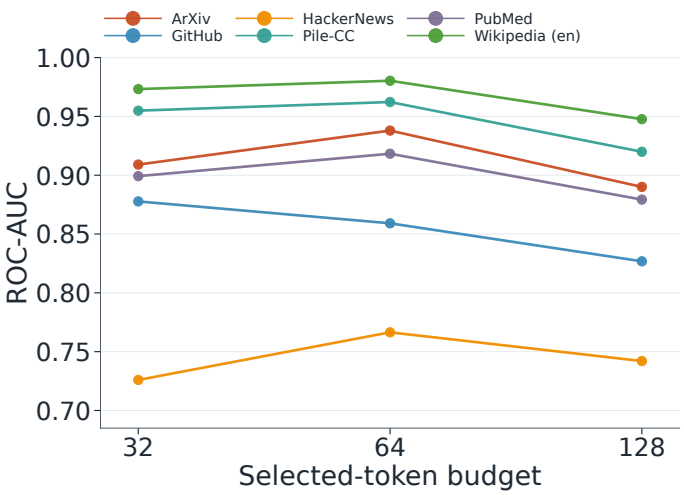


Figure 6: **Selected-token budget.** Domain-wise ROC-AUC across probing budgets. The six-domain average is maximized at $K = 64$.

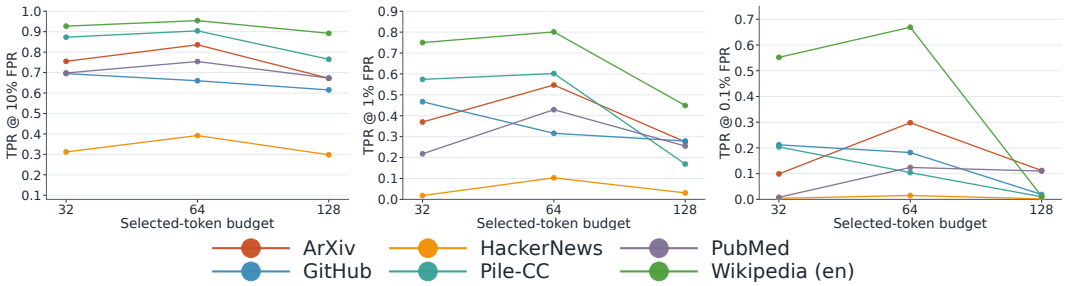


Figure 7: Detailed selected-token budget sweep at low-FPR operating points. The same domain ordering is preserved across panels, and $K = 64$ gives the strongest mean operating point.

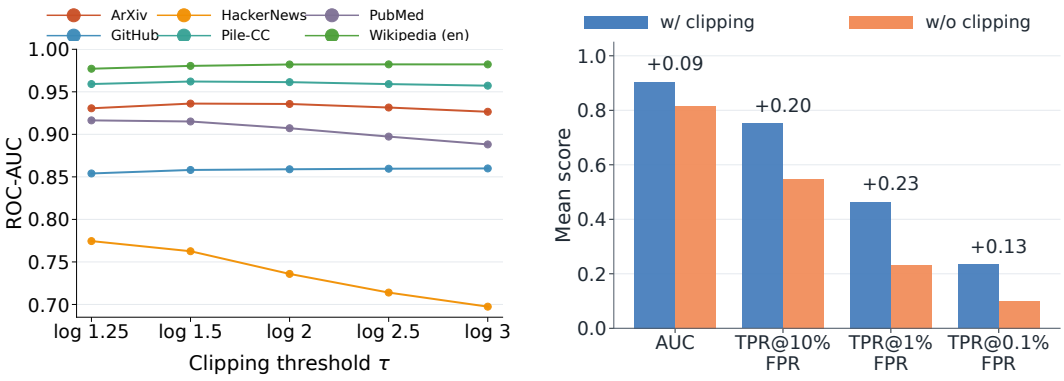
With $K = 32$, the statistic averages fewer local reconstruction advantages; with $K = 128$, less informative positions dilute the selected set. Both the domain-wise ROC-AUC curves in Figure 6 and the strict operating points in Figure 7 support $K = 64$. Increasing K does not change the number of model forwards because all retained positions are scored in the same joint masked input.

D.9 Clipping Operation Ablation

Table 19 reports the aggregate clipping ablation. The selector is fixed to the PRISM-C4 head, and both readouts use the same selected token positions. The unclipped variant averages raw target/reference gaps; the clipped variant caps each token contribution before averaging.

Table 19: Clipping ablation averaged over six MIMIR domains.

Method	ROC-AUC	TPR@10%	TPR@1%	TPR@0.1%
No clip mean	0.8166	0.5478	0.2323	0.1003
Clipped mean	**0.9024**	**0.7500**	**0.4663**	**0.2320**



(a) Threshold sensitivity.

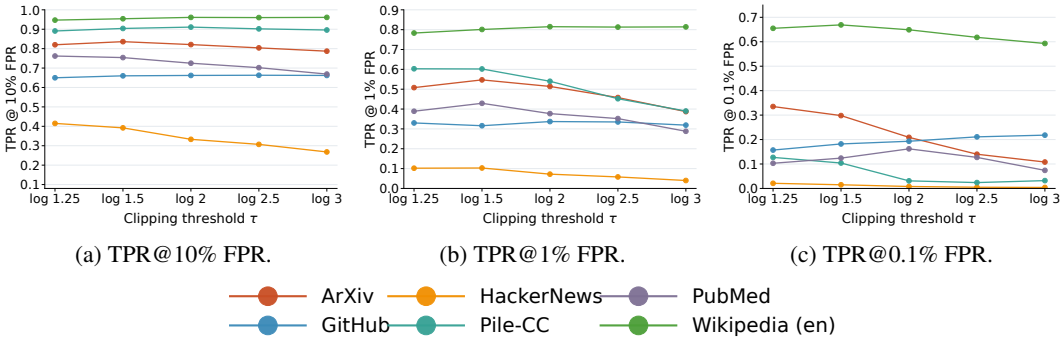
(b) Clipped versus unclipped mean.

Figure 8: **Clipping ablation.** The left panel varies the clipping threshold; the right panel isolates the effect of applying clipping while keeping the selected positions fixed.

Clipping improves every metric, with the largest relative gain at the strictest operating point. In particular, TPR@0.1%FPR more than doubles after clipping. This behavior is consistent with the role of clipping as a robustness correction: selected hard positions can produce extreme target/reference gaps, and an unclipped mean can be dominated by a small number of unstable token-level contributions. Clipping limits these isolated spikes and instead rewards sequences where the target has a stable reconstruction advantage across multiple selected positions.

D.10 Clipping-Threshold Sweep

The threshold τ controls the cap applied to each token contribution in Eq. (6). Small values suppress token-level outliers more aggressively, whereas large values approach the unclipped mean. The ROC-AUC curves in Figure 8 show that the method is not narrowly tuned to a single threshold. We additionally report the complete low-FPR sweep below.



(a) TPR@10% FPR.

(b) TPR@1% FPR.

(c) TPR@0.1% FPR.

Figure 9: **Clipping-threshold sensitivity at low FPR.** The legend is shared across panels. Although individual domains sometimes favor a neighboring threshold, $\tau = \log 1.5$ gives the strongest overall balance across the three operating points.

Across the full domain and metric grid, $\tau = \log 1.5$ provides the most stable aggregate operating point. We therefore use it for every reference-assisted experiment rather than selecting a threshold separately for each domain or FPR.

D.11 Selector Training Cost

For the selector-corpus ablation in Table 15, all PRISM heads are trained with the same budget of 200k samples. This matched budget isolates the effect of the selector training corpus rather than the amount of selector supervision. The downstream attack pipeline is otherwise identical across PRISM-C4, PRISM-Slim, and PRISM-FW: each selector chooses $K = 64$ positions, after which

the same joint multimask query and clipped scoring difference between the target and reference are applied.

PRISM-head training is a one-time preprocessing cost. In our implementation, PRISM-C4 takes 26.9 minutes of wall time, corresponding to approximately 1.8 GPU-hours on 4×L40S GPUs. PRISM-Slim takes 23.9 minutes, or approximately 1.6 GPU-hours, and PRISM-FW takes 26.8 minutes, or approximately 1.8 GPU-hours. When amortized over the evaluation suite of six datasets with 2,000 samples each, these costs correspond to 0.134 seconds per sample for PRISM-C4, 0.120 seconds per sample for PRISM-Slim, and 0.134 seconds per sample for PRISM-FW.

D.12 Per-Sample Runtime Accounting

All reported runtimes are normalized as seconds per sample. Although attacks are executed in batches for efficiency, per-sample runtime is the most direct measure of attack cost from the auditor's perspective. This is particularly important for dLLM MIAs because attacks that repeatedly query many masked variants can become expensive even when each individual forward pass is parallelized.

Table 20: Runtime per sample in seconds. "Head" denotes amortized one-time selector training.

Method	Inference	Head	Total
SAMA	1.863	N/A	1.863
JUMP (C4)	0.184	0.134	0.318
JUMP (SlimPajama)	0.185	0.120	0.305
JUMP (FineWeb)	0.183	0.134	0.317

Even after amortizing selector-head training, the learned-selector variants remain substantially faster than SAMA. The single-pass design provides the computational advantage: after selection, all retained tokens are reconstructed jointly in one masked input.

E Role of Reference Calibration

The reference model supports both position selection and token-level calibration. To measure the calibration contribution, we fix the PRISM-selected $K = 64$ positions and compare their target reconstruction scores with the target-to-reference gap in Eq. (3). The selected positions and evaluation examples are identical in both cases.

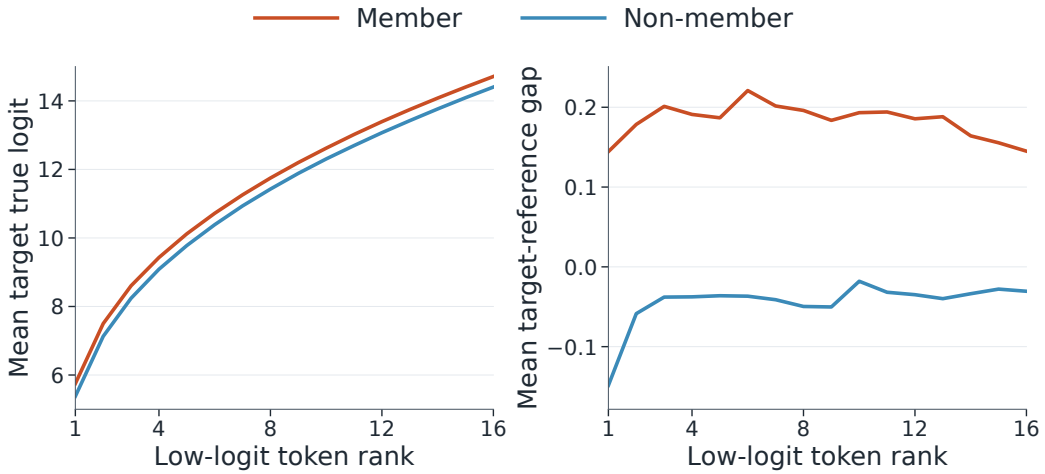


Figure 10: **Effect of reference calibration.** The left panel reports target true-token scores on the PRISM-selected positions. The right panel evaluates the target-to-reference gap on those same positions. Absolute scores retain substantial token-difficulty variation, whereas the paired gap produces a clearer separation between members and non-members.

Figure 10 visualizes the token-level mechanism, and the sequence-level ablation gives the same conclusion. With the PRISM mask set held fixed, the target-only readout reaches mean ROC-AUC

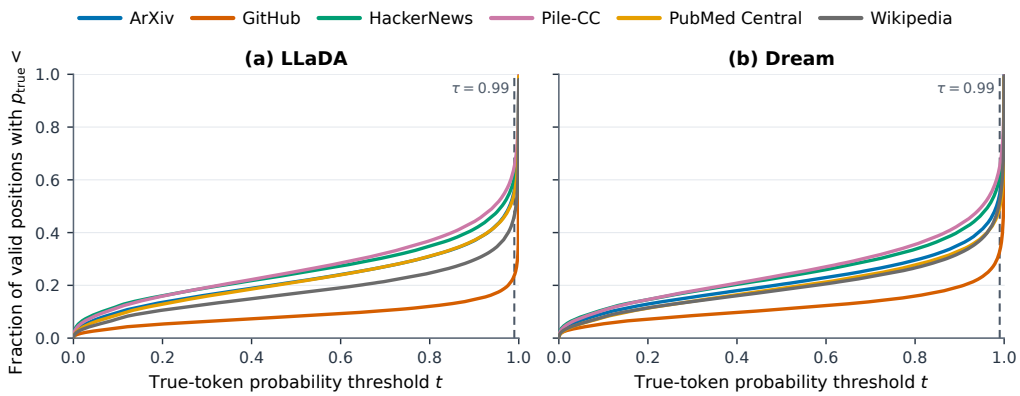


Figure 11: Empirical CDFs of grouped true-token reconstruction probabilities for LLaDA (left) and Dream (right). A substantial fraction of valid positions lies in the near-deterministic region close to $p_{\text{true}} = 1$, while the size of this region varies considerably across domains.

0.5616 and TPR@10% FPR 0.1462 on LLaDA, compared with 0.9024 and 0.7500 for reference-calibrated JUMP. On Dream, the corresponding values are 0.6176 and 0.2032, versus 0.9415 and 0.8438. Thus, even a learned selector does not make an absolute target score directly comparable across documents. A low score may reflect intrinsic ambiguity rather than non-membership, while a high score may reflect generic predictability rather than memorization. The paired gap addresses this ambiguity when the lineage checkpoint is available; the entropy-centered target-only construction in Section 6 is used when it is not.

The broader selector controls in Table 3 reinforce this separation. Random positions, Clean confidence, and Reference-guided reach LLaDA mean AUC 0.5530, 0.5603, and 0.5616, respectively; the corresponding Dream values are 0.5899, 0.6143, and 0.6176. Low-confidence selection therefore adds signal, especially on Dream, but the small change from target-clean to PRISM-selected target scoring confirms that better localization alone is not a substitute for calibrating token difficulty.

F Target-Only JUMP Ablations

We evaluate the components of Target-Only JUMP using the same 1,000 members and 1,000 non-members per MIMIR domain as in the main evaluation. Unless otherwise stated, all analyses reuse the deterministic balanced 16-way grouped reconstructions defined in Appendix A.7. The threshold, readout, and normalization ablations therefore reaggregate the same target-model outputs and introduce no additional masked reconstruction contexts.

We focus on three questions: why near-deterministic positions should be removed, how the entropy-centered readout behaves across domains and model families, and whether the 16-way grouped reconstruction provides a sufficiently accurate approximation to the ideal one-hole measurement.

F.1 Reconstruction Confidence and the Near-Deterministic Regime

Target-Only JUMP removes positions whose grouped true-token reconstruction probability is at least 0.99. This filter is intended to exclude a saturated reconstruction regime, rather than to select a fixed fraction of low-confidence tokens. To characterize this regime directly, we examine the empirical distribution of grouped true-token reconstruction probabilities over all valid positions.

At the final threshold $p_{\text{true}} \leq 0.99$, the six-domain mean fraction of retained valid positions is 51.1% for LLaDA and 52.4% for Dream. The retained fraction varies substantially across domains. For example, only 23.7% of LLaDA GitHub positions and 32.9% of Dream GitHub positions remain below 0.99, whereas more than 60% are retained for HackerNews and Pile-CC.

The filter therefore does not impose a fixed token budget. Instead, it removes the portion of each document for which the reconstruction distribution has already become nearly deterministic. The CDFs in Figure 11 should not be interpreted as showing that 0.99 is mathematically optimal. Rather, they show that the saturated regime is substantial and that its prevalence varies across domains.

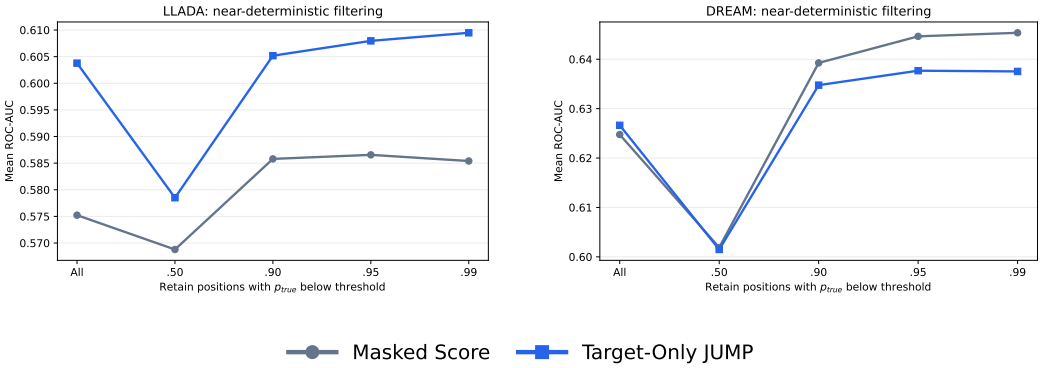


Figure 12: Mean ROC-AUC across six domains as the near-deterministic filtering threshold varies for LLaDA (left) and Dream (right). Aggressive filtering at $p_{\text{true}} \leq 0.50$ removes useful evidence, whereas performance is comparatively stable over the 0.90–0.99 range.

F.2 Saturation-Threshold Ablation

We next vary the probability threshold while holding the grouped measurements and score definitions fixed. We compare using all valid positions with $p_{\text{true}} \leq 0.50$, 0.90, 0.95, and 0.99. For each retained support, Masked Score averages q, whereas Target-Only JUMP averages $q + H$.

Table 21: Mean ROC-AUC across six domains under different near-deterministic filtering thresholds. "All" uses every valid position.

Model	Score	All	$p \leq .50$	$p \leq .90$	$p \leq .95$	$p \leq .99$
LLaDA	Masked Score	.5752	.5688	.5858	.5866	.5854
LLaDA	Target-Only JUMP	.6038	.5785	.6052	.6080	**.6095**
Dream	Masked Score	.6248	.6019	.6393	.6446	**.6454**
Dream	Target-Only JUMP	.6266	.6015	.6348	**.6377**	.6375

Table 22: Mean number of retained positions per document under different probability thresholds, averaged equally across the six domains.

Model	All	$p \leq .50$	$p \leq .90$	$p \leq .95$	$p \leq .99$
LLaDA	315.0	59.6	102.8	118.9	155.3
Dream	305.9	57.4	100.9	117.4	156.3

The threshold sweep shows that Target-Only JUMP is not simply a hard-token selection rule. The aggressive $p_{\text{true}} \leq 0.50$ filter retains only about 60 positions per document on average and reduces attack performance on both model families. Performance recovers once the threshold reaches 0.90 and changes only modestly over the 0.90–0.99 range.

For LLaDA, the mean ROC-AUC of $q + H$ increases from 0.6052 at $p_{\text{true}} \leq 0.90$ to 0.6080 at 0.95 and 0.6095 at 0.99. For Dream, the corresponding values are 0.6348, 0.6377, and 0.6375. Thus, 0.99 is not selected because it uniquely maximizes ROC-AUC. We use it as a common boundary that removes only the near-deterministic tail while retaining the broader set of non-saturated reconstruction events.

F.3 Membership Signal Inside and Outside the Saturated Regime

The threshold sweep shows that excluding near-deterministic positions can improve the document-level statistic, but it does not by itself show whether the removed positions contain membership information. We therefore directly compare the token-level member–non-member difference in $q + H$ inside and outside the $p_{\text{true}} = 0.99$ boundary.

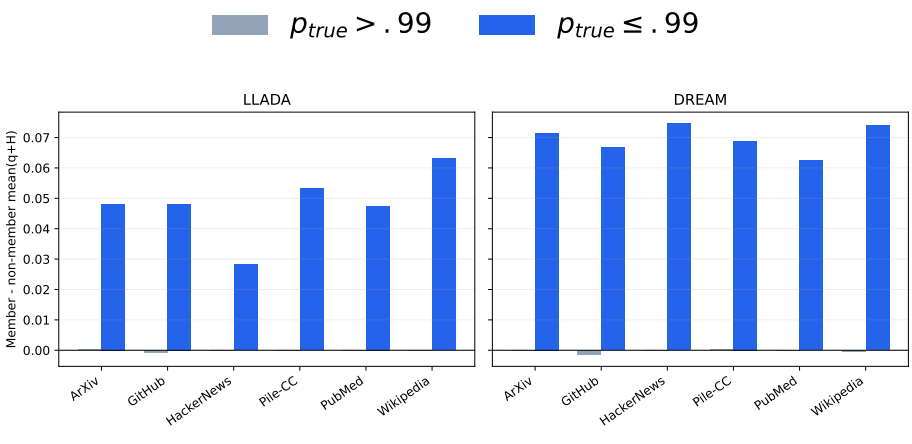


Figure 13: Token-level member–non-member difference in $q + H$ inside and outside the saturated reconstruction regime. Saturated positions contribute essentially no directional membership signal, whereas non-saturated positions retain a consistently positive member–non-member difference.

Table 23: Token-level member–non-member difference in $q + H$ inside and outside the saturated regime. Saturated positions satisfy $p_{\text{true}} > 0.99$.

Model	Domain	Saturated gap	Non-saturated gap	Saturated fraction
LLaDA	ArXiv	0.00018	0.04816	44.8%
LLaDA	GitHub	-0.00090	0.04816	76.6%
LLaDA	HackerNews	-0.00002	0.02818	40.1%
LLaDA	Pile-CC	0.00003	0.05321	35.7%
LLaDA	PubMed Central	-0.00004	0.04752	46.0%
LLaDA	Wikipedia (en)	-0.00021	0.06306	53.9%
LLaDA	Mean	-0.00016	0.04805	49.5%
Dream	ArXiv	-0.00001	0.07149	46.2%
Dream	GitHub	-0.00149	0.06670	67.2%
Dream	HackerNews	-0.00029	0.07456	39.7%
Dream	Pile-CC	0.00018	0.06890	35.7%
Dream	PubMed Central	-0.00014	0.06239	51.6%
Dream	Wikipedia (en)	-0.00037	0.07392	48.5%
Dream	Mean	-0.00035	0.06966	48.2%

The difference is effectively zero in the saturated region for both architectures. Averaged across domains, the member–non-member gap is -0.00016 on LLaDA and -0.00035 on Dream for positions with $p_{\text{true}} > 0.99$. In contrast, the corresponding non-saturated gaps are positive: 0.04805 on LLaDA and 0.06966 on Dream.

This behavior matches the limiting argument in the main text. As $p_{\text{true}} \to 1$, the true-token log-probability q approaches zero and the predictive entropy H also vanishes. Consequently, $q + H$ collapses toward zero regardless of membership. Including many such positions in the document-level average therefore dilutes evidence from reconstruction events for which multiple alternatives remain plausible.

F.4 Entropy-Centered Readout Across Domains

We next isolate the contribution of the entropy-centered readout. Masked Score and Target-Only JUMP use exactly the same $p_{\text{true}} \leq 0.99$ support and the same grouped target outputs. The only difference is whether the document score averages q or $q + H$.

Table 24: Per-domain ROC-AUC using the same $p_{\text{true}} \leq 0.99$ support. Masked Score averages q, while Target-Only JUMP averages $q + H$.

	LLaDA		Dream	
Domain	q	$q + H$	q	$q + H$
ArXiv	.6018	**.6169**	.6576	**.6592**
GitHub	.5769	**.5955**	**.6372**	.5810
HackerNews	.5409	**.5662**	.6528	**.6574**
PubMed Central	.5982	**.6212**	.6331	**.6362**
Wikipedia (en)	.6037	**.6382**	.6474	**.6526**
Pile-CC	.5909	**.6188**	**.6440**	.6388
Mean	.5854	**.6095**	**.6454**	.6375

On LLaDA, entropy centering improves ROC-AUC in all six domains, raising the six-domain mean from 0.5854 to 0.6095.

The Dream result is less uniform. Masked Score has a higher six-domain mean ROC-AUC (0.6454 versus 0.6375), but this difference is dominated by GitHub. On Dream GitHub, adding entropy decreases ROC-AUC from 0.6372 to 0.5810, a drop of 0.0562. This single-domain difference is substantially larger than the differences observed elsewhere.

Across the other five Dream domains, $q + H$ improves ROC-AUC in four. Pile-CC is the only additional decrease, and its change is much smaller, from 0.6440 to 0.6388. As a diagnostic for identifying the source of the Dream mean difference, excluding GitHub gives a five-domain mean of 0.6470 for q and 0.6488 for $q + H$. We do not use this five-domain quantity as an evaluation result; it only shows that the overall Dream reversal is concentrated primarily in GitHub rather than reflecting a consistent degradation across Dream domains.

We retain $q + H$ as the Target-Only JUMP readout because it implements the within-position relative-preference comparison developed in the main text and produces consistent improvements across all six LLaDA domains. At the same time, Dream GitHub provides a clear counterexample to uniform dominance. We therefore report this architecture- and domain-dependent behavior explicitly rather than claiming that entropy centering always outperforms masked log-probability.

F.5 Variance-Normalized Relative Preference

We additionally test whether the relative-preference score benefits from normalization by the scale of the predictive log-probability distribution. Following a Min-K%++-style normalization, we consider

$$z_i = \frac{q_i + H_i}{\sigma_i},$$

where σ_i is the full-vocabulary standard deviation of log-probabilities under the same grouped reconstruction context.

Table 25: Mean ROC-AUC across six domains with and without vocabulary-scale normalization.

Model	Threshold	q	$q + H$	$(q + H)/\sigma$
LLaDA	$p \leq .50$	.5688	.5785	.5634
LLaDA	$p \leq .90$	.5858	.6052	.5923
LLaDA	$p \leq .95$	.5866	.6080	.5962
LLaDA	$p \leq .99$	.5854	**.6095**	.5994
Dream	$p \leq .50$	.6019	.6015	.5739
Dream	$p \leq .90$	.6393	.6348	.6215
Dream	$p \leq .95$	.6446	.6377	.6279
Dream	$p \leq .99$	**.6454**	.6375	.6336

At the final $p_{\text{true}} \leq 0.99$ threshold, normalization reduces mean ROC-AUC from 0.6095 to 0.5994 on LLaDA and from 0.6375 to 0.6336 on Dream.

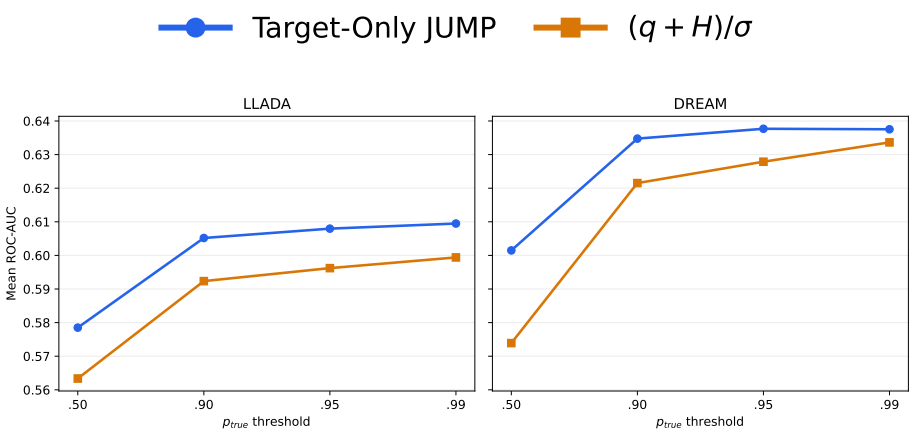


Figure 14: Mean ROC-AUC across six domains for the raw masked score, entropy-centered relative preference, and its vocabulary-scale-normalized variant across probability thresholds.

Table 26: Grouped reconstruction diagnostic on LLaDA. Values are six-domain means under the final $p_{\text{true}} \leq 0.99$ Target-Only JUMP score.

Measurement	Contexts	AUC	T@10	T@1	T@0.1
F16 once	16	.6095	.1995	.0270	.0060
F16 twice	32	.6130	.1952	.0295	.0062
F32 once	32	.6056	.1915	.0255	.0068

Normalization does change individual low-FPR operating points. On LLaDA, TPR@10%, TPR@1%, and TPR@0.1% FPR change from 0.1995/0.0270/0.0060 for $q + H$ to 0.2060/0.0330/0.0073 after normalization. On Dream, the corresponding values change from 0.1997/0.0262/0.0053 to 0.2290/0.0300/0.0027.

Thus, vocabulary-scale normalization can alter tail behavior, but it does not provide a consistent cross-family improvement and reduces mean ROC-AUC on both model families. We therefore use the unnormalized $q + H$ readout in the final method.

F.6 Grouped Approximation to One-Hole Reconstruction

The target-only formulation is motivated by one-hole reconstruction, whereas the practical attack evaluates positions through a balanced 16-way partition. We finally test whether the grouped measurement is sufficiently close to the one-hole view.

On the cached ArXiv diagnostic subset, grouped F16 and exact one-hole q have a Spearman correlation of 0.9511. More directly for the final filtering rule, the two measurements agree on whether a position satisfies $p_{\text{true}} \leq 0.99$ for 99.67% of the cached positions. Thus, the grouped approximation closely preserves both reconstruction confidence ordering and the saturated/non-saturated decision.

We also compare one 16-way partition with two independently hashed 16-way partitions and one 32-way partition. The latter two settings double the number of masked reconstruction contexts.

Repeating the 16-way partition increases the six-domain mean ROC-AUC only from 0.6095 to 0.6130 while doubling the number of masked contexts from 16 to 32, and slightly reduces TPR@10% FPR. Using 32 smaller groups instead yields a mean ROC-AUC of 0.6056. We therefore retain one balanced 16-way partition as the default.

The exact one-hole diagnostic has one limitation. The cached exact outputs contain the true-token reconstruction probability but not the full-vocabulary entropy H. Consequently, we can compare exact and grouped q and the $p_{\text{true}} \leq 0.99$ filtering decision, but cannot compute the complete exact one-hole $q + H$ document score without additional model inference. The reported comparison should therefore be interpreted as evidence for the fidelity of the grouped reconstruction measurement, rather than as an exact one-hole evaluation of the complete final attack.

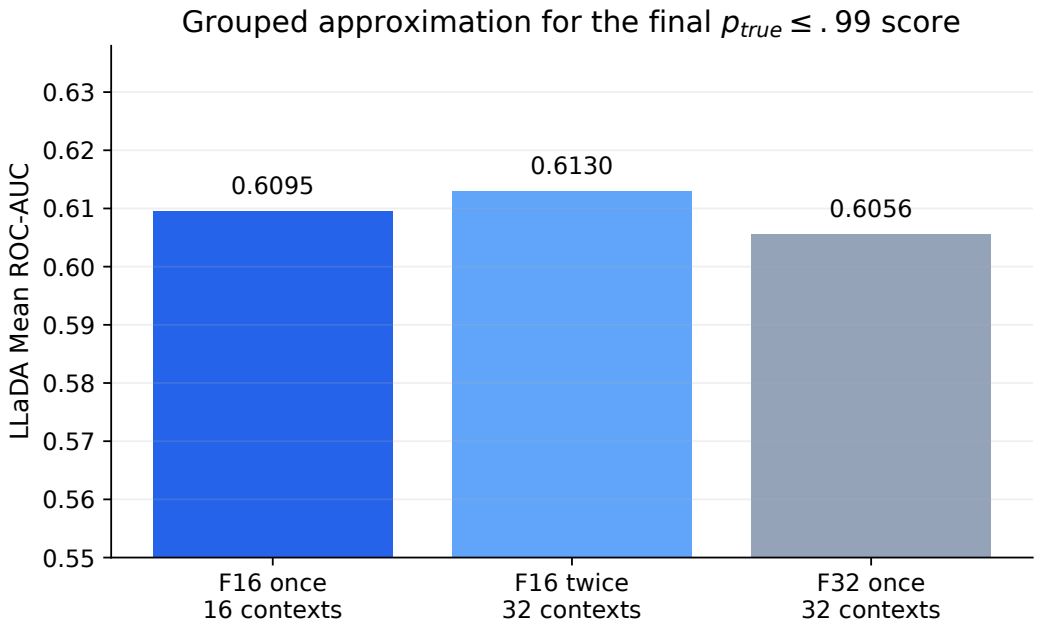


Figure 15: Quality–cost comparison for grouped reconstruction under the final $p_{\text{true}} \leq 0.99$ Target-Only JUMP score. Doubling the number of masked contexts yields no material improvement in the six-domain mean.

G Defense via Matched LoRA and DP-LoRA

We separate the effect of adapter parameterization from differential privacy through two controlled ArXiv experiments: a non-DP LoRA capacity sweep, followed by a matched non-DP LoRA versus DP-LoRA comparison at fixed adapter capacity. Our DP experiments build on DP-SGD (Abadi et al., 2016) and prior work on differentially private language-model fine-tuning (Yu et al., 2022; Li et al., 2022). All rows use 1,000 ArXiv members and 1,000 disjoint non-members. Reference-assisted JUMP uses the paired pre-fine-tuning checkpoint, while Target-Only JUMP uses the balanced 16-way reconstruction, fixed $p_{\text{true}} < 0.99$ filtering rule, and entropy-centered readout from Section 6.

G.1 LoRA Capacity Control

All LoRA targets use four epochs, learning rate 5×10^{-5}, effective batch size 48, maximum length 512, dropout 0.05, seed 42, and $\alpha = 2r$. Held-out reconstruction perplexity is measured on the 1,000 non-members.

Table 27: ArXiv LoRA capacity sweep. JUMP denotes the reference-assisted attack and Target-Only denotes Target-Only JUMP.

r/α	Trainable	PPL	AUC		TPR@1%	
			JUMP	Target-Only	JUMP	Target-Only
16/32	0.10%	5.22	0.512	0.498	0.007	0.012
64/128	0.42%	4.89	0.545	0.503	0.018	0.013
256/512	1.65%	4.71	0.588	0.512	0.026	0.013
1024/2048	6.28%	4.66	0.675	0.544	0.066	0.011
Full FT	100.00%	4.74	0.938	0.617	0.539	0.033

Leakage increases with adapter capacity while held-out reconstruction improves. Reference-assisted JUMP rises from 0.512 to 0.675 across the LoRA sweep, and Target-Only JUMP rises from 0.498 to 0.544. Full fine-tuning produces the strongest signal for both attacks, reaching 0.938 and 0.617, respectively. The paired checkpoint therefore reveals a sharper capacity trend, while the target-only score preserves its direction at higher adapter capacity.

G.2 MATCHED LORA AND DP-LORA

We next fix $r = 1024$, $\alpha = 2048$, dropout 0.05, clipping norm 1.0, and $\delta = 10^{-5}$, and vary only the DP noise required for target privacy budgets $\varepsilon \in \{0.01, 1, 10, 20\}$. The non-DP LoRA row uses the identical adapter capacity and training data.

Table 28: Matched non-DP LoRA and DP-LoRA on ArXiv across target privacy budgets.

Method	Target ε	PPL	AUC		TPR@1%	
			JUMP	Target-Only	JUMP	Target-Only
Non-DP LoRA	N/A	4.66	0.675	0.544	0.066	0.011
DP-LoRA	0.01	55.77	0.529	0.497	0.013	0.012
DP-LoRA	1	18.67	0.502	0.498	0.012	0.011
DP-LoRA	10	7.89	0.514	0.498	0.013	0.011
DP-LoRA	20	7.05	0.520	0.498	0.014	0.011

At fixed adapter capacity, DP reduces reference-assisted JUMP from 0.675 to between 0.502 and 0.529, and Target-Only JUMP from 0.544 to between 0.497 and 0.498. The target-only values and their TPR@1% FPR are at chance-level resolution throughout the DP sweep. The strictest privacy budgets also increase held-out reconstruction perplexity, while weaker budgets retain more domain utility.

Accounting caveat. Privacy accounting uses a PRV accountant with a fixed-batch sampling-rate approximation and `secure_rng=False`. The realized privacy budgets closely match the target ε values reported in Table 28. These values are accountant outputs under this implementation, not an independently certified deployment guarantee. Accordingly, these experiments demonstrate attack suppression and a trade-off between privacy and utility under a matched training setup; they should not be read as a production certification.

H EXISTING ASSETS AND LICENSES

We use publicly released models, datasets, and baseline resources. Table 29 summarizes their roles and the license or access terms used by this work. We do not redistribute pretrained model weights or the original MIMIR data. Users should obtain each asset from its original provider and comply with its license and terms of use.

Table 29: Existing assets used in this paper.

Asset	Role in this paper	Citation / source	License / terms
LLaDA-8B-Base	Main base/reference dLLM and fine-tuning target	Nie et al. (2025)	MIT
Dream-v0-Base-7B	Second dLLM family for cross-architecture evaluation	Ye et al. (2025)	Apache-2.0
MIMIR	Member/non-member evaluation benchmark	Duan et al. (2024)	MIT; gated access terms on Hugging Face
C4	PRISM selector training corpus	Raffel et al. (2020)	ODC-BY; Common Crawl Terms of Use
SlimPajama	Selector-corpus ablation	Soboleva et al. (2023)	Apache-2.0 for the original Cerebras release; subset-specific terms if using a split or reupload
FineWeb	Selector-corpus ablation	Penedo et al. (2024)	ODC-By v1.0; Common Crawl Terms of Use
SAMA	Prior dLLM MIA baseline	Chen et al. (2026)	MIT if using the official code repository