You Can Have a Second Chance: Unbiased and Multi-bit Watermarking for Diffusion Language Models with Regret-based Remasking

Ke Yang[1,2,3], Dongyang Liang[1,2,3], Jing Yu[1,2,3], Shuguang Yuan[1,2,3]*, Chi Chen[1,2,3]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]State Key Laboratory of Cyberspace Security Defense
[3]School of Cyber Security, University of Chinese Academy of Sciences
{yangke, yujing, yuanshuguang, chenchi}@iie.ac.cn
liangdongyang@e.gzhu.edu.cn

Abstract

The rapid development of Diffusion Language Models (DLMs) raises concerns about watermarking for DLM-generated detection. However, existing sequential LLM watermarking cannot be directly applied to DLMs, as DLMs' generation order is arbitrary. While emerging studies adapt biased LLM watermarking to DLMs by temporarily predicting the watermark prefix, they suffer from degraded quality and unstable watermarking due to bias accumulation and prediction errors. Besides, they cannot carry multi-bit watermarks. In this paper, we propose unbiased multi-bit watermarking for DLMs. We introduce a stability-aware constraint that allows watermarking only in stable contexts and a bit-controlled, unbiased modulation to preserve the original DLM output distribution, achieving stable watermarking with minimal quality impact. To enhance detection robustness, we design a *Regret-based Remasking*, which grants a "second chance" for unwatermarked tokens to be regenerated. It can seamlessly integrate into DLM inference with no added diffusion steps and latency. Experiments across DLMs and various tasks show that our scheme is effective, achieving superior generation quality compared to baselines while maintaining high detection accuracy and multi-bit capacity. Our code is available here.

1 Introduction

Diffusion Language Models (DLMs) have emerged as a compelling and promising alternative to the dominant autoregressive (AR) paradigm (Li et al., 2025). By leveraging full attention and parallel generation, DLMs enable global context perception and accelerated inference (Li et al., 2025). In tasks such as planning (Ye et al., 2025) and code generation (DeepMind, 2025; Labs, 2025), DLMs have been widely shown to achieve better performance than AR models (Yu et al., 2025). To safeguard against potential abuse of DLMs (İsmail Tarım and Onan, 2025), watermarking can serve as a promising solution for AI-generated content detection and provenance tracing, by embedding imperceptible signals into the generated content.

Extensive studies have established effective watermarking for LLMs (Kirchenbauer et al., 2023; Dathathri et al., 2024). However, these schemes cannot be directly applied to DLMs due to their distinct inference modes. As shown in Figure 1, DLMs (Sahoo et al., 2024; Nie et al., 2025) generate text through a semi-parallel denoising process starting from fully masked sequences, rather than sequential token prediction of LLMs. This non-sequential nature violates the core assumption of LLM watermarking: the availability of prior tokens as a watermark prefix. Thus, developing watermarking paradigms tailored for DLMs is urgent.

DLM Watermarking remains in its infancy, with two valid studies (Gloaguen et al., 2025; Wu et al., 2025) adapting the Green-Red LLM watermarking (Kirchenbauer et al., 2023) via a *predicted bidirectional constraint*: For a position to be unmasked, its prior tokens are used as the watermark prefix to determine the "green" token set. Meanwhile, the "green" token selected for the position is expected to serve as the watermark prefix for next tokens, enabling them to become "green". When context (prior and next) tokens remain masked, these schemes temporarily set them to the most probable tokens predicted in the current step. However, these methods present three significant limitations:

Unstable watermark signal: Since the sequence is mostly masked in the early denoising stages, the model's prediction is low confidence. This results in watermarking that depends on unreliable contextual predictions. The mismatch between predicted and actual generated contexts may compromise the final watermark signal.

Quality degradation: Existing methods are biased watermarking. The bias accumulates

*Corresponding author

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 28141–28160
July 2-7, 2026

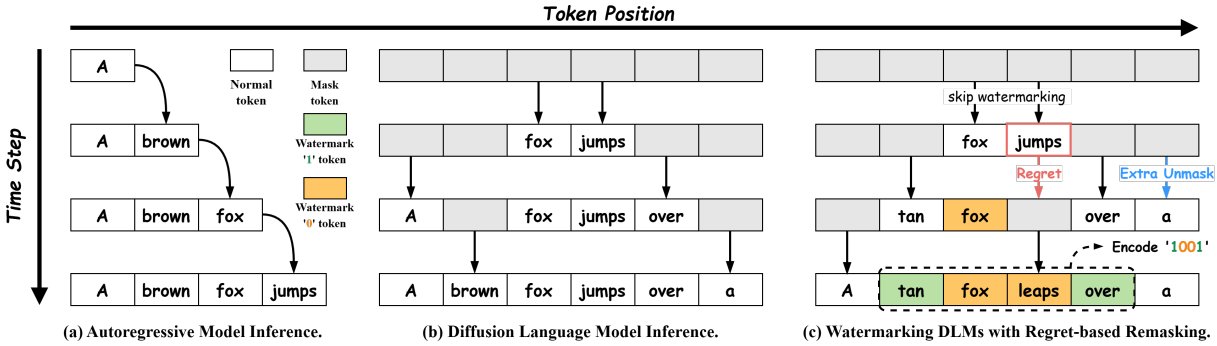

Figure 1: Illustrations of AR/DLMs inference and UMR. UMR is an unbiased multi-bit DLM watermarking scheme. It skips watermarking at unstable positions, while allowing the non-watermark tokens to be regretted. To avoid slowing inference, when a token regrets, UMR unmasks an extra token to offset the regret.

throughout the diffusion iterations, leading to a progressive degradation in generation quality.

🔢 **Limited capacity**: Current methods are zero-bit watermarking (binary detection), ignoring the multi-bit capacity that is feasible and essential for fine-grained provenance (e.g., user IDs, copyright), as evidenced in LLM watermarking (Yoo et al., 2024a; Qu et al., 2025; Feng et al., 2025).

Motivated by these challenges, we ask: *Is it possible to decouple watermarking from the early denoising stages to achieve a stable watermark signal?* and *Can unbiased multi-bit watermarking be achieved for high watermark capacity while preserving generation quality?*

Based on these insights, we propose UMR, an unbiased and multi-bit watermarking for diffusion language models with a Regret-based Remasking strategy. To enhance watermark signal stability, we introduce a *stability-aware constraint* that decouples watermarking from unstable denoising positions, activating watermarking embedding only when the context is sufficiently stable. The watermarking is implemented by mapping the message bits to the vocabulary and using unbiased probability modulation. This theoretically protects the generation quality while supporting fine-grained provenance detection. To ensure detection ability, we design *Regret-based Remasking*, which tracks non-watermarked tokens, employs a "regret" strategy to remask them, and regenerates them once stability conditions are met. Figure 1 (c) illustrates the process. Experimental results show that UMR is model-agnostic, achieving a detection accuracy of 0.95 across various tasks. The quality of watermarked generations in downstream tasks is close to that of the original DLM generations. Besides, UMR is resistant to common attacks. Contributions are summarized as follows:

- We propose a novel Regret-based Remasking strategy for DLMs that leverages the iterative generation process to recover missed watermarking tokens, ensuring both a stable watermark signal and high generation quality.
- We achieve the first unbiased and multi-bit watermarking scheme for DLMs by combining a stability-aware constraint with unbiased probability modulation, addressing the limitations of prior zero-bit, biased methods.
- Experiments demonstrate that UMR achieves reliable detection performance and watermark capacity, with negligible impact on text quality compared to existing baselines.

2 Preliminaries

2.1 Diffusion Language Models

Diffusion language models (DLMs) can be categorized into continuous DLMs (Li et al., 2022; Gulrajani and Hashimoto, 2023) and discrete DLMs (Nie et al., 2025; Ye et al., 2025). Benefiting from their native alignment with discrete language data, discrete DLMs have demonstrated capabilities comparable to AR models (Li et al., 2025), rapidly emerging as the mainstream research direction in DLMs. In this paper, we focus on discrete DLMs; for brevity, we refer to them simply as DLMs.

2.2 Inference of DLMs

During inference, DLM starts from a prompt p_0 and a fully masked sequence $x_T = [\mathsf{MASK}, \ldots, \mathsf{MASK}]_L$. The generation proceeds over T timesteps. At each step t (from T to 1), the model $p_\theta(x_0|p_0, x_t)$ pre-

dicts the logits for all positions, producing a probability distribution over the vocabulary $\mathcal{V}$:

$$p_\theta(x_0^{(i)} \mid p_0, x_t) = \text{Softmax}(\text{logits}^{(i)}). \quad (1)$$

A *Remasking* strategy controls the pace of generation. It selects a subset of masked positions $\mathcal{S}_{N_t}$ of size N_t from x_t to unmask by replacing `[MASK]`s with valid tokens $\hat{x}^{(i)}$, while the rest remain masked. This process can be formulated as:

$$x_{t-1}^{(i)} = \begin{cases} \hat{x}_0^{(i)} \sim p_\theta(x_0^{(i)}|p_0, x_t), & \text{if } i \in \mathcal{S}_t \\ x_t^{(i)}, & \text{otherwise} \end{cases} \quad (2)$$

This selection typically follows either a *confidence-based strategy*, which selects high-confidence tokens ($\hat{x}^{(i)} = \arg\max p_\theta$) for unmasking, or a *random strategy*, which samples randomly to enhance diversity (Nie et al., 2025; Ye et al., 2025). See Appendix B for the details of these strategies. Crucially, once a token is unmasked, it is typically fixed and not reverted to `[MASK]` in subsequent steps.

3 Methodology

Our goal is to design an unbiased, multi-bit watermarking method for DLMs, achieving high watermark capacity while mitigating quality degradation. To achieve this, we introduce three components:
Constrained Bidirectional Watermarking: Bidirectional watermarking starts only after some local context tokens reach a stable threshold. This stabilizes the watermark signal and mitigates its cumulative negative impact on the diffusion process.
Bit-Controlled Unbiased Modulation: The watermarking method is performed in the token generation stages. It maps the multi-bit watermark to vocabulary partitions for encoding bits into tokens. Then it uses unbiased probabilistic reweighting to increase watermark tokens' probability while ensuring the entire process remains statistically invisible.
Regret-based Remasking Strategy: A strategy to track and remask unwatermarked tokens during generation. It grants these tokens a "second chance" to regenerate watermark tokens when their context becomes stable, enhancing watermark signal coverage without adding inference overhead.

For brevity, let $P(v)$ denote the probability $p_\theta(x_0^{(i)} = v|p_0, x_t)$ of a token v predicted by the model at the current step t for a target position i.

3.1 Constrained Bidirectional Watermarking

Watermark signal instability arises from context-prediction errors driven by low-confidence model outputs in early denoising stages. To mitigate this, we introduce a *stability-aware constraint*. Given a sequence x for generation, we define a stability indicator $\mathcal{I}_{\text{st}}(i)$ for each position i. A position is allowed to generate with watermarking only if at least one of its immediate neighbors (prior and next token) is determined (non-mask):

$$\mathcal{I}_{\text{st}}(i) = \sum_{j \in \{i-1, i+1\}} \mathbb{1}(x^{(j)} \neq \texttt{[MASK]}) \quad (3)$$

where $\mathbb{1}(\cdot)$ is the indicator function. Positions in unstable contexts (where $\mathcal{I}_{\text{st}}(i) = 0$) will skip the watermarking process, allowing them to be generated purely based on the original model distribution. This effectively decouples the watermarking process from the low-confidence denoising stages.

When the stability condition is met ($\mathcal{I}_{\text{st}}(i) > 0$), we apply a predicted bidirectional watermarking for token $x^{(i)}$. Since DLM generation is non-sequential, the neighbor tokens $x^{(j)}$ used for watermarking might still be masked. To handle this, we define a temporary predicted context token $\tilde{x}^{(j)}$ as:

$$\tilde{x}^{(j)} = \begin{cases} x^{(j)}, & \text{if } x^{(j)} \neq \texttt{[MASK]} \\ \arg\max_{v \in \mathcal{V}} P(v), & \text{if } x^{(j)} = \texttt{[MASK]}. \end{cases} \quad (4)$$

Using these contexts, the selection of $x^{(i)}$ at position i is governed by the bidirectional constraint:
Forward Constraint (C_{fwd}): $\tilde{x}^{(i-1)}$ as the watermark prefix to select $x^{(i)}$ from watermark tokens.
Backward Constraint (C_{bwd}): $x^{(i)}$ should form a watermark prefix of its next token $\tilde{x}^{(i+1)}$, resulting $\tilde{x}^{(i+1)}$ a watermark token.

Ideally, a selected watermark token $x^{(i)}$ should satisfy the intersection $C_{\text{fwd}} \wedge C_{\text{bwd}}$. Figure 2 illustrates this process. The stability-aware constraint shields the watermark signal from unreliable predictions and reduces cumulative impact by isolating watermarking from early diffusion steps. The bidirectional constraint enables sequential detection.

3.2 Bit-Controlled Unbiased Modulation

To embed a multi-bit message, we map watermark bits to partitions of the vocabulary $\mathcal{V}$.

For statistical randomness and security, we shuffle the message to a pseudorandom watermark sequence $W = \{w_0, \ldots, w_L\}$ derived from the secret key sk, where $w_k \in \{0, 1\}$. For watarmarking ith position in a diffusion step, we first use the prior token $\tilde{x}^{(i-1)}$ to compute a random $seed = \text{Hash}(\tilde{x}^{(i-1)}, sk)$ by a hash function with sk. A

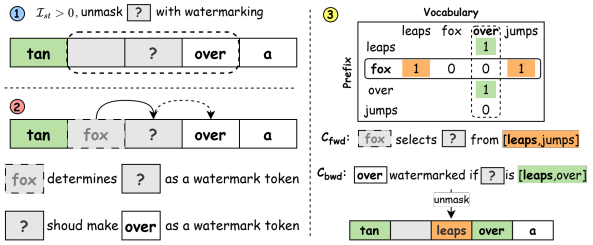

Figure 2: Process of constrained bidirectional watermarking. $\mathcal{I}_{st} = 1$ of "?", thus can preform watermarking. "fox" is a temporary predicted token. Prefix-vocabulary matrix in ❸ determines watermark region.

pseudorandom generator with $seed$ is employed to divide the vocabulary into two distinct partitions $\mathcal{V}_0, \mathcal{V}_1$, representing bit 0 and bit 1, respectively. And then a target watermark bit b is selected from the idx position in W using $idx = seed \mod L$, which determines $\mathcal{V}_b$ as the watermark partition.

To avoid the computational cost of the above-mentioned hash operations on the inference, we pre-compute the watermark prefix-vocabulary matrix $M_W \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ in Figure 2 to store the watermark region of each watermark prefix to encode a watermark b. $M_W[u,v] = 1$ indicates that token v is a watermark token given a prior token u. In practice, given any watermark prefix u, $M_w[u,:]$ directly provides the watermark region of u (C_{fwd}). $M_W[:,v]$ denotes the watermark prefix tokens can determine v to a watermark token (C_{bwd}).

Thus, to embed bit b under the bidirectional constraint, we first get the final watermark token region $\mathcal{E}_i$ by $C_{\text{fwd}} \wedge C_{\text{bwd}}$:

$$\mathcal{E}_i = \{v \mid M_W[\tilde{x}^{(i-1)}, v] \wedge M_W[v, \tilde{x}^{(i+1)}]\}. \quad (5)$$

we aim to bias the token sampling towards $\mathcal{E}_i$ to achieve watermarking. To ensure the watermarking process does not statistically bias the whole output distribution of DLM, we employ an unbiased reweighting on the original probability distribution P to obtain a watermarked distribution P_w:

$$P_w(v) = \begin{cases} P(v)\cdot(1+\delta) & \text{if } v \in \mathcal{E}_i \\ P(v)\cdot(1-\delta') & \text{if } v \notin \mathcal{E}_i \end{cases} \quad (6)$$

Here, $\delta > 0$ enhances the probability of watermark tokens. The reduction factor δ' is dynamically calculated as $\delta' = \frac{\delta\tau}{1-\tau}$, where $\tau = \sum_{v\in\mathcal{E}_i} P(v)$ is the total probability mass of the watermark region. This constraint of δ' guarantees an effective probability distribution (See Appendix A.1 for proof).

Theorem 1. *The proposed modulation strategy is unbiased, i.e.,* $\mathbb{E}[P_w] = P$.
Proof. See Appendix A.2.

Theorem 1 guarantees the expected distribution remains identical to the original model, ensuring that the watermark is statistically imperceptible and the generation quality is theoretically preserved.

3.3 Regret-based Remasking Strategy

The stability-aware constraint inevitably decreases the watermark coverage rate, thus affecting watermark detection. To address this issue, we introduce Regret-based Remasking (R-remasking), which enables the model to remask missed watermarked tokens and regenerate them as their contexts stabilize. It is worth noting that R-remasking remains fully compatible with standard remasking strategies, rather than replacing them, and does not increase the number of diffusion steps or inference latency. We illustrate R-remasking in DLM's generation process, abstracted in Algorithm 1, which mainly consists of the following two modules.

Candidate Tracking. To enable DLM regrets, we maintain a *Candidate Watchlist* (CW) to dynamically track targeted tokens that can be remasked. Whenever a newly unmasked token $x^{(i)}$ fails to fully satisfy the bidirectional constraint ($C_{fwd} \wedge C_{bwd}$), the CW update mechanism is triggered. It records the token into the CW as a tuple $(pos_i, conf_i, w_{pn}^{(i)}, \mathcal{I}_{st}^{(i)}, status_i)$. Here, pos_i denotes the position of $x^{(i)}$, $conf_i$ is its prediction confidence, $\mathcal{I}_{st}^{(i)}$ tracks the stability of the token's local context across diffusion steps, and $status_i$ records whether the position has already been regretted, preventing infinite loops.

To provide an intuitive quantification of the watermark strength of $x^{(i)}$, we define the discrete metric $w_{pn}^{(i)} \in \{0,1,2\}$: 0 indicates no constraint is satisfied (unwatermarked), 1 means either C_{fwd} or C_{bwd} is met, and 2 denotes full satisfaction. During the diffusion iterations, if a candidate in CW updates to $w_{pn}^{(i)} = 2$, it is automatically removed from the CW. This indicates that the token fully satisfies the bidirectional constraints, reaching its maximum watermark strength.

Regret Policy. R-remasking is adaptively integrated into the regular diffusion iterations. To ensure seamless integration without increasing diffusion steps, a regret trigger is evaluated at each step: it activates only when the CW contains *eligible candidates* whose local context has reached the

Algorithm 1 R-remasking (Low-Confidence)

For each diffusion step t, do the following:
Budget: N_t is standard unmasking budget of sequence x_t. If CW has eligible candidates, $N_t \leftarrow N_t + 1$, $Bud = True$.
Unmask N_t tokens of x_t with watermarking.
Update & Rollback: Update CW. If a token regenerates but degrades watermark strength ($w_{pn}^{new} < w_{pn}^{ori}$), revert to snapshot v_{ori}.
if $Bud ==$ True **then**
  if CW has eligible candidates **then**
    Remask Target: Pick $k \in$ CW (max $\mathcal{I}_{st}$, min w_{pn}, min $conf$). Snapshot v_{ori}; Set $x_t^{(k)} \leftarrow$ [MASK]; Remove k from CW.
  else
    Fallback: Remask the lowest confidence token unmasked in the current step.
  end if
end if

stability threshold ($\mathcal{I}_{st} > 0$) and were not regretted ($Status_i = False$). Once triggered, we dynamically adjust the unmasking budget by unmasking one extra token during the standard process to offset the regretted one.

To select the specific token to regret, a vanilla Random R-remasking strategy is proposed that randomly picks from eligible tokens. However, this strategy risks masking a partially watermarked token ($w_{pn} = 1$) and regenerating it with no watermark ($w_{pn} = 0$). To mitigate this, we propose the Low-Confidence R-remasking strategy. This strategy selects the optimal target position by jointly evaluating three criteria: the most stable current context ($\max \mathcal{I}_{st}$), the least satisfaction of bidirectional constraints ($\min w_{pn}$), and the lowest prediction confidence ($\min conf$).

Furthermore, to guarantee that the regret operation never degrades the watermark, we introduce a strict rollback trigger. We snapshot the original token's value v_{ori} before remasking. Upon regeneration, if the new token's watermark strength is lower than the original snapshot ($w_{pn}^{new} < w_{pn}^{ori}$), the rollback trigger activates to revert v_{ori}. This explicitly enforces the *monotonicity of watermark injection*. For better understanding, Figure 3 illustrates these two strategies in an extreme case where the regretted token is still unwatermarked after regeneration. We further elaborate on the whole watermarking procedure and algorithms step-by-step in Appendix C.

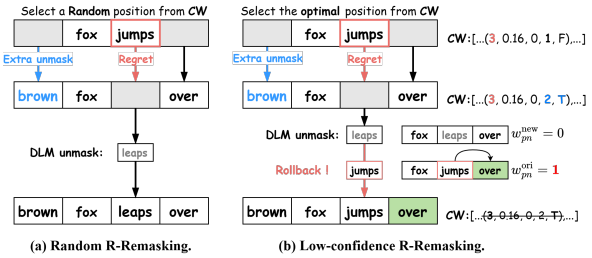

Figure 3: Illustration of R-remasking strategies when regeneration is ineffective. Unlike (a) Random strategy, which risks replacing a valid token with a weaker one, (b) Low-confidence strategy selects the optimal target and triggers a rollback if the new token fails to improve the watermark strength. Both strategies remove the processed position from CW to prevent infinite loops.

3.4 Watermark Extraction

UMR extracts the watermark in a message-agnostic manner. We utilize a voting matrix $V \in \mathbb{R}^{L \times 2}$ to accumulate evidence for each bit of the watermark. Given a generated sequence x, for each token $x^{(i)}$, we re-calculate the watermark position index idx and the vocabulary partitions $\mathcal{V}_0$ and $\mathcal{V}_1$. We then check the subset membership of $x^{(i)}$ to update V:

$$\begin{cases} V[idx][1] \leftarrow V[idx][1] + 1, & \text{if } x^{(i)} \in \mathcal{V}_1 \\ V[idx][0] \leftarrow V[idx][0] + 1, & \text{otherwise} \end{cases} \quad (7)$$

This binary voting mechanism relies on the statistical assumption that watermarked text contains a significantly higher proportion of tokens falling into the target partitions than natural text.

After text traversing, the k-th bit of the extracted watermark is determined by $\arg\max_{b \in 0,1} V[k][b]$. The sum of the majority votes $\sum_{k=1}^{L} \max(V[k][:])$ can be used to perform binary detection (watermarked vs. non-watermarked).

4 Evaluation

In this section, we evaluate the effectiveness of UMR on diverse DLMs across datasets for text completion, question answering, and instruction following tasks. We assess the overhead, attack impact, and downstream performance compared to existing DLM watermarking baselines. Finally, we conduct ablation studies to verify the specific contributions of the hyperparameters and the Regret-based Remasking strategy.

4.1 Experimental Settings

Datasets and Metrics. We utilize three public datasets: the C4-en subset (Raffel et al., 2023)

for text completion, ELI5 (Fan et al., 2019) for open-ended QA, and Alpaca (Taori et al., 2023) for instruction following. To evaluate detection effectiveness, impact of attacks, and ablation studies, we report the Bit Accuracy (Acc.) of the extracted watermark and Perplexity (PPL) for text fluency. When comparing against zero-bit baselines, we use the True Positive Rate at a low False Positive Rate (TPR@FPR%) as the detection metric. For downstream text quality, we utilize BERTScore (Zhang et al., 2020), ROUGE-1, ROUGE-L (Lin, 2004), BLEU (Papineni et al., 2002), and PPL to measure the generated texts from summarization and translation tasks. For the ablation of the R-remasking strategy, we calculate the Watermark Coverage Rate (WCR) to quantify the proportion of tokens successfully watermarked in the final output. See Appendix D.1 for further details.

Baselines. We choose two pioneering DLM bidirectional watermarking schemes: GSJ (Gloaguen et al., 2025) and DMark (Wu et al., 2025) as the baselines. These two schemes are biased zero-bit watermarking schemes. For a fair comparison, we set the logits increment $\delta = 4.0$ and the green list proportion $\gamma = 0.5$ to maintain consistency with the default settings of our scheme. It is worth noting that the δ of GSJ and DMark is based on the maximum bias of the token logits when the prior and next tokens are used as watermark context.

Implementation Details. We conduct experiments on four state-of-the-art DLMs: LLaDA-8B-Instruct (Nie et al., 2025), LLaDA-1.5-8B (Zhu et al., 2025), Dream-v0-Instruct-7B (Ye et al., 2025), and RND1-Base-0910 (Numerics, 2025). For inference, unless otherwise specified, we generate sequences of maximum length 256 tokens using 256 denoising steps at a temperature of 0.0. We utilize the default block size of 64 for the LLaDA variants, while Dream and RND1 use no block due to non-support. For each dataset, we sample prompts for generation and filter the outputs for repetitive n-grams until we gain 100 normal outputs. The default watermark length is 4 bits, with a vocabulary partition ratio of $\gamma = 0.5$. To prevent probability overflow (where the boosted probability of the watermark region exceeds 1), we dynamically constrain δ using a base factor $\tilde{\delta}$ (default $\tilde{\delta} = 4.0$):

$$\delta = \begin{cases} 0 & \text{if } \tau = 0, \\ (1-\tau)\tau & \text{if } (1+\tilde{\delta})\tau > 1, \\ \tilde{\delta} & \text{otherwise.} \end{cases} \tag{8}$$

See more implementation details in Appendix D.2.

Threat Model. We define the *defender* as the model provider aiming to verify the provenance of generated content to prevent misuse. Upon receiving a suspicious text message, the defender attempts to extract the embedded watermark. The *adversary* is a malicious user seeking to generate harmful or synthetic text while evading detection. The adversary may employ deletion, synonym substitution, or paraphrasing attacks, attempting to compromise the watermark.

4.2 Effectiveness Evaluation

Table 1 presents the comprehensive evaluation of detection Bit Accuracy (Acc.) and Perplexity (PPL) across various settings.

Overall, UMR demonstrates robust detection performance, with bit accuracy consistently exceeding 0.85 across most test cases. Under default parameters, Acc.> 0.92 while maintaining low perplexity. Besides, we observe highly consistent results across DLMs. This corroborates that our approach is a universally applicable (model-agnostic) solution for the discrete DLMs.

For effectiveness evaluation on diverse tasks, we observe a slight advantage on the C4 dataset compared to ELI5 and Alpaca, particularly in shorter sequences. We attribute this to the tasks' intrinsic entropy: C4 (open-ended completion) generally exhibits higher entropy, offering more degrees of freedom for watermarking. In contrast, the responses of the QA task (ELI5) and the instruction-following task (Alpaca) are usually more deterministic, limiting the sampling space available for watermarking.

To measure the impact of text length, we filtered the generated texts by token lengths (approx. 50, 100, 200, 300). DLMs may encounter repetitive loops in long text generation; we have filtered out such texts to ensure the reliability of the results. The results in Table 1 reveal that increasing the length from 50 to 100 tokens yields a $\approx 5\%$ accuracy boost, and stable detection (Acc.≈ 0.95) is achieved when the length is greater than 100. This indicates that UMR requires only moderate-length text to achieve near-perfect detection. Notably, even in short-text scenarios (≈ 50 tokens) where watermarking is unstable, UMR maintains reliable detection (Acc.> 0.83). Furthermore, PPL naturally decreases as generation length grows. This trend aligns with the "context warm-up" phenomenon, where extended sequences provide richer context for more confident predictions.

Finally, we examined the trade-off between wa-

Table 1: Effectiveness across Datasets and Models under Different Generation Length and Watermark Bits Length.

Dataset	Model	Text Length								Watermark Bits Length					
		50		100		200		300		2		6		8	
		Acc.	PPL↓	Acc.	PPL↓	Acc.	PPL↓	Acc.	PPL↓	Acc.	PPL↓	Acc.	PPL↓	Acc.	PPL↓
C4	LLaDA	0.952	16.54	0.965	7.776	0.948	4.791	0.960	4.101	0.980	4.905	0.890	4.887	0.847	4.405
	LLaDA-1.5	0.895	16.20	0.960	9.142	0.940	5.678	0.933	4.163	0.960	5.331	0.916	5.516	0.867	5.365
	Dream*	0.890	18.73	0.972	10.54	-	-	-	-	0.975	7.137	0.933	10.97	0.885	10.76
	RND1$^+$	0.910	16.42	0.955	6.240	0.960	3.035	0.930	4.263	0.980	3.409	0.926	3.410	0.880	3.407
ELI5	LLaDA	0.873	12.72	0.940	9.485	0.930	5.796	0.935	5.173	0.980	5.984	0.881	5.798	0.863	6.048
	LLaDA-1.5	0.887	14.33	0.955	9.738	0.925	6.285	0.950	5.828	0.985	6.290	0.905	6.290	0.872	6.678
	Dream*	0.857	16.93	0.958	12.24	-	-	-	-	0.985	10.30	0.893	11.01	0.848	13.02
	RND1$^+$	0.925	14.46	0.960	9.643	0.975	5.432	0.975	4.457	0.990	4.563	0.933	4.558	0.890	4.553
Alpaca	LLaDA	0.865	11.31	0.925	7.581	0.935	5.419	0.950	5.200	0.975	5.521	0.885	5.521	0.890	5.719
	LLaDA-1.5	0.902	12.80	0.949	8.969	0.951	6.131	0.959	5.884	0.984	5.800	0.917	6.022	0.886	6.123
	Dream*	0.831	12.79	0.903	8.472	-	-	-	-	0.960	6.867	0.901	8.171	0.825	7.942
	RND1$^+$	0.890	13.13	0.925	6.683	0.960	4.116	0.955	4.008	0.980	4.371	0.933	4.711	0.885	4.712

* In practice, Dream (non-watermarked) struggles to generate long texts. Results for 200 and 300 tokens are marked as "-" due to insufficient samples. In the watermark bit-length experiment, the default text length for Dream is set to 100.

+ A temperature of 0 results in deterministic generation of RND1, rendering watermarking impossible. We set the temperature to 1.0 to balance quality and watermark effectiveness.

termark capacity and detection ability by varying the watermark length (2, 6, 8 bits). As shown in Table 1, accuracy inversely correlates with the length. This is theoretically expected: embedding more bits within a fixed text length inevitably reduces the embedding times (redundancy) for each bit. Despite this, UMR maintains Acc.> 0.82 even at an 8-bit capacity, demonstrating its ability to support provenance tracking.

To validate the statistical significance of these results above, we calculated the standard deviations and 95% confidence intervals of these metrics. The details of the assessment are presented in Appendix E. The assessment indicates that our results are effective and robust.

Zero-bit watermarking. To facilitate a fair comparison with existing zero-bit watermarking baselines, we configured UMR with a single-bit watermark to simulate a zero-bit setting. We employed TPR@FPR=1% and PPL as evaluation metrics. As illustrated in Figure 4, although UMR exhibits lower TPR@FPR=1% than baselines in short-text generation, its detection performance is comparable to baselines on generated text exceeding 100 tokens. Additionally, we compared the AUC metrics of UMR against the baselines under a generation length of 200. As illustrated in the ROC curves, the AUC of UMR closely approaches that of the baseline. Crucially, UMR consistently yields lower PPL than the baselines across all lengths, indicating the effectiveness of unbiased watermarking and the stability-aware constraint.

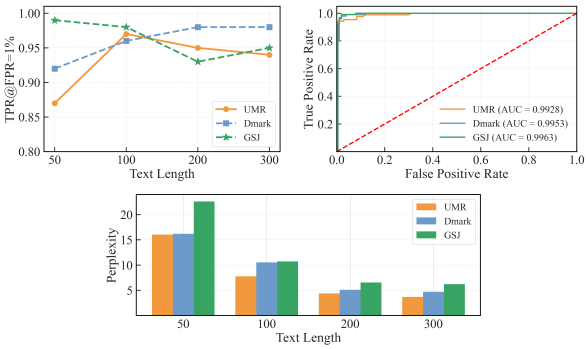

Figure 4: Zero-bit Watermark Detection.

Overhead. We evaluated the inference overhead of the original generation, UMR, and UMR with only R-remasking (no watermarking) across different generation lengths. As verified by the results in Figure 5(a), the R-remasking strategy adds no inference latency. Nevertheless, UMR is not zero-latency compared to the original model. The main overhead originates from the watermarking process, with latency scaling linearly with generation length, as expected for token-level watermarking.

We then compared the time overhead of UMR for text generation and watermark detection against baselines and the unwatermarked model. As shown in Figure 5(b), UMR increases a 1.6s (Random R-remasking) and 2.5s (Low-confidence R-remasking) latency compared with the origin. This overhead is competitive with baselines and acceptable for practical deployment. Watermark detection is efficient, requiring only 0.0263s per sample,

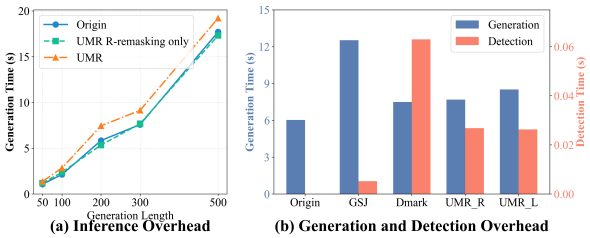

Figure 5: Evaluation of Watermarking Overhead.

demonstrating the scalability of UMR.

In Appendix F, we provide a detailed analysis of the storage and memory overhead of the watermark matrix M_W across different models. Results show that under the memory capacity of modern hardware typically used to serve LLMs and DLMs, UMR's memory overhead is entirely acceptable for practical deployment. Furthermore, it is important to emphasize that precomputing M_W is strictly a *space-for-time optimization* rather than a systemic requirement. If memory constraints are a concern, UMR can operate like standard LLM watermarking by dynamically computing the watermark tokens via hashing during inference.

4.3 Text Quality Evaluation

The impact of watermarking on the quality of generated text is assessed in two downstream tasks: text summarization and translation. We employed the CNN/DailyMail dataset (See et al., 2017) for summarization, and we measured performance using BERTScore-F, ROUGE-1, ROUGE-L, and PPL. For translation, we employed the WMT'19 De-En subset (Foundation, 2019), and performance was measured using BERTScore-F1, BLEU, and PPL. We compared and evaluated the generated text from UMR with and without watermarking, along with the baselines. Results in Table 2 show that baselines exhibit performance degradation, while the UMR generation quality is closest to that of non-watermark generation. This verifies UMR's unbiasedness, indicating that it has retained the DLM's core abilities for downstream tasks. The high PPL in translation tasks is due to the fact that the translated texts are all relatively short, leading to significant variation.

4.4 Impact of Attacks

We evaluated UMR against four attacks: random/synonym substitution, deletion, and paraphrasing. As shown in Figure 6, (i) random substitution caused the most significant degradation, reducing

Table 2: DLM Downstream Task Performance.

Task	Metrics	Origin	GSJ	DMark	UMR
Text Summ.	BERT-S	**0.874**	0.867	0.863	0.872
	ROUGE-1	**38.60**	35.65	31.23	38.21
	ROUGE-L	**25.38**	21.69	19.35	24.72
	PPL↓	**14.42**	18.53	17.03	16.77
Text Trans.	BERT-S	**0.939**	0.926	0.932	0.938
	BLEU	**39.19**	33.05	36.80	38.96
	PPL↓	52.17	48.45	65.46	**45.22**

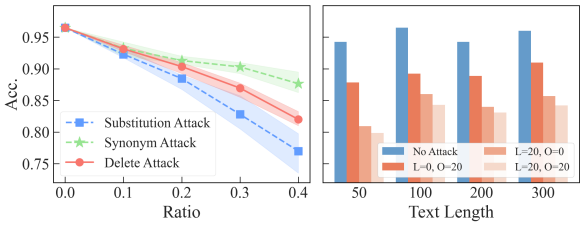

Figure 6: Results on common attacks.

Acc. by 20% at a 0.4 modification rate, followed by the deletion attack, reducing Acc. by 13%. However, these attacks severely compromise the utility of text. In contrast, the semantics-preserving synonym substitution had a marginal impact, reducing accuracy by only 7%. (ii) We simulated paraphrasing by employing DIPPER (Krishna et al., 2023), a fine-tuning model for watermark evasion, to vary lexical (L) and order (O) diversity. We found that structural reordering has the most significant impact on Acc., inducing a 15% drop. It is worth noting that robustness improves for texts exceeding 100 tokens. Overall, UMR exhibits resilience to these attacks, maintaining Acc. above 80% across all simulated scenarios.

Besides, we presented a comparison of the attack robustness of UMR in the zero-bit setting against the baselines. The experimental results are in Appendix G, Table 7. The results show that although UMR exhibits a slight performance drop to basic edits (e.g., substitution and deletion) compared to baselines, as an inherent trade-off for maintaining unbiasedness and text quality, it demonstrates superior robustness against sophisticated paraphrasing attacks, outperforming baselines with a TPR@0.5 of 0.72 (vs. GSJ: 0.69, DMark: 0.61). For a more detailed analysis, please refer to the Appendix G.

4.5 Ablation

We performed ablations on the watermark strength, the Regret-based remasking strategy, and different standard remasking strategies. Additionally, we verified UMR's performance across different

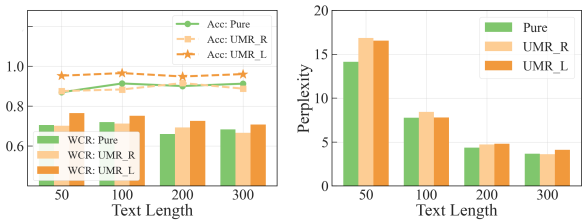

Figure 7: Ablation of R-remasking Strategy.

generation modes (block and pure diffusion) and iteration steps (see Appendix H).

Watermark Strength. We evaluated the trade-off between detection robustness and text fluency by varying the strength parameter $\tilde{\delta}$ from 2.0 to 5.0; the results are shown in Table 3. As expected, increasing $\tilde{\delta}$ positively correlates with Bit accuracy (Acc.) but incurs a penalty on Perplexity (PPL). We observe a substantial boost in accuracy as $\tilde{\delta}$ rises from 2.0 to 4.0, achieving robust performance that Acc. > 0.9 at $\tilde{\delta} \geq 3.0$. However, increasing $\tilde{\delta}$ further to 5.0 yields diminishing returns in accuracy while significantly inflating PPL. Therefore, we adopt $\tilde{\delta} = 4.0$ as the optimal trade-off between watermark strength and generation quality.

Table 3: Results of the Impact of Watermark Strength.

	$\tilde{\delta} = 2.0$	$\tilde{\delta} = 3.0$	$\tilde{\delta} = 4.0$	$\tilde{\delta} = 5.0$
Acc.	0.815	0.915	0.950	0.960
PPL↓	4.574	4.617	4.781	5.173

Impact of R-remasking. To isolate the contribution of the R-remasking, we compared the pure watermarking (Pure) against watermarking with Random/Low-Confidence R-remasking (UMR_R and UMR_L) across varying text lengths (results in Figure 7). UMR_L yields a significant performance boost over Pure, with an average accuracy improvement of 5.8%, which is particularly pronounced in shorter sequences. This improvement aligns with a notably higher Watermark Coverage Rate (WCR), confirming that our strategy successfully recovers missed watermarking opportunities. In contrast, UMR_R shows negligible improvement. This is attributed to the blind nature of random selection, which risks disrupting existing valid watermark links (breaking C_{fwd} or C_{bwd}), thus neutralizing the benefits of regret.

Impact of Standard Remasking. We evaluated UMR under two standard remasking strategies, Random remasking and Low-confidence remasking, and reported the results in Table 4. UMR achieves high detection accuracy (Acc. ≥ 0.93) under both these strategies, indicating that it is remasking-strategy-agnostic. Notably, the slightly higher PPL observed under Random remasking is an inherent characteristic of the strategy and is completely independent of UMR.

Table 4: Results of the Impact of Standard Remasking.

	Random	*Low_conf*
Acc.	0.930	0.946
PPL↓	6.949	4.820

5 Background & Related Work

Diffusion Language Models (DLMs) have advanced rapidly, raising concerns about the provenance and detection of DLM-generated content. Existing DLM watermarking schemes are derived from AR-LLM watermarking (Kirchenbauer et al., 2024; Zhao et al., 2023). They can be categorized into: (i) *Context-dependent methods* (Gloaguen et al., 2025; Wu et al., 2025), which adapt biased LLM watermarking to DLMs by a predicted bidirectional constraint. However, these schemes suffer from quality degradation and unstable watermarking due to the bias and the prediction error. (ii) *Context-independent methods* (Bagchi et al., 2025), which rely on global secret parameters or sequence positions for watermark hashing. While simpler, these methods exhibit significant security and robustness vulnerabilities. Overall, there remains substantial space for improvement in DLM watermarking regarding mechanism design, watermark capacity, and generation quality. Extended background and details are provided in Appendix I.

6 Conclusion

In this study, we propose UMR, an unbiased multi-bit watermarking scheme with a regret-based remasking strategy for diffusion language models. UMR decouples watermarking from early denoising, maintaining stable watermarking. The unbiased watermarking algorithm mitigates the negative impact on text quality while achieving high watermark capacity. Regret-based remasking then increases the watermarking success rate by giving unwatermarked tokens a second chance to be generated. The experiments show the effectiveness of UMR detection, superior text quality compared to baselines, and resistance to common attacks.

Limitations

In this section, we discuss several limitations of this work. First, as derived in our Appendix A, the unbiased nature of UMR relies on the large vocabulary assumption, where the probability of any single token is negligible compared to the total probability of the watermark target region. In practice, however, this ideal condition is rarely strictly met due to the non-uniformity of natural language distributions (e.g., Zipf's law). Consequently, the watermark selection cannot be perfectly uniform, implying that the generated text deviates slightly from a theoretically perfect unbiased distribution. A possible way to further enhance unbiasedness is to prevent repeated watermarking of high-frequency tokens, but this would inevitably reduce watermark coverage and impair detection performance.

Second, experiments in the zero-bit watermarking scenario indicate that UMR's detection capability in short-text generation is slightly inferior to standard bidirectional watermarking schemes. This suggests that the *Regret-based Remasking* strategy may face limitations in achieving sufficiently high watermark coverage within constrained sequence lengths.

Finally, the Impact of Attacks section reveals that UMR is susceptible to these attacks. This vulnerability stems from an inherent trade-off: we prioritized unbiasedness and quality preservation, which came at the cost of some robustness. Developing unbiased watermarking schemes with enhanced robustness remains a critical direction for future research.

Despite these limitations, we believe that our work serves as an important catalyst in this field, contributing positively to the advancement of DLM watermarking techniques with higher watermark capacity and minimal impact on generation quality.

Acknowledgments

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences, Grant No. XDB0690303.

References

Scott Aaronson. 2023. Watermarking of large language models. https://simons.berkeley.edu/talks/scott-aaronson-ut-austin-openai-2023-08-17. Accessed: 2025-12-31.

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

Avi Bagchi, Akhil Bhimaraju, Moulik Choraria, Daniel Alabi, and Lav R. Varshney. 2025. Watermarking discrete diffusion language models. *Preprint*, arXiv:2511.02083.

Liang Chen, Yatao Bian, Yang Deng, Deng Cai, Shuaiyi Li, Peilin Zhao, and Kam-Fai Wong. 2024. WatME: Towards lossless watermarking through lexical redundancy. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9166–9180.

Aloni Cohen, Alexander Hoover, and Gabe Schoenbach. 2025. Watermarking language models for many adaptive users. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 2583–2601. IEEE.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, and 1 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823.

DeepMind. 2025. "gemini diffusion". https://deepmind.google/models/gemini-diffusion/. Accessed: 2025-12-29.

Jaiden Fairoze, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Mingyuan Wang. 2023. Publicly-detectable watermarking for language models. *arXiv preprint arXiv:2310.18491*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567.

Xiaoyan Feng, He Zhang, Yanjun Zhang, Leo Yu Zhang, and Shirui Pan. 2025. Bimark: Unbiased multilayer watermarking for large language models. *arXiv preprint arXiv:2506.21602*.

Wikimedia Foundation. 2019. Acl 2019 fourth conference on machine translation (wmt19), shared task: Machine translation of news.

Thibaud Gloaguen, Robin Staab, Nikola Jovanović, and Martin Vechev. 2025. Watermarking diffusion language models. *Preprint*, arXiv:2509.24368.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Ishaan Gulrajani and Tatsunori B Hashimoto. 2023. Likelihood-based diffusion language models. *Advances in Neural Information Processing Systems*, 36:16693–16715.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4115–4129. Association for Computational Linguistics.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. 2023. Unbiased watermark for large language models. *Preprint*, arXiv:2310.10669.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. 2024. On the reliability of watermarks for large language models. *Preprint*, arXiv:2306.04634.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Preprint*, arXiv:2303.13408.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2024. Robust distortion-free watermarks for language models. *Preprint*, arXiv:2307.15593.

Inception Labs. 2025. "mercury". `https://www.inceptionlabs.ai/introducing-mercury`. Accessed: 2025-12-29.

Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. 2025. A survey on diffusion language models. *Preprint*, arXiv:2508.10875.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35:4328–4343.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *Preprint*, arXiv:2502.09992.

Radical Numerics. 2025. Training diffusion language models at scale using autoregressive models.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Wenjie Qu, Wengrui Zheng, Tianyang Tao, Dong Yin, Yanze Jiang, Zhihua Tian, Wei Zou, Jinyuan Jia, and Jiaheng Zhang. 2025. Provably robust multi-bit watermarking for ai-generated text. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 201–220.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. *Preprint*, arXiv:2406.07524.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Linyu Wu, Linhao Zhong, Wenjie Qu, Yuexin Li, Yue Liu, Shengfang Zhai, Chunhua Shen, and Jiaheng Zhang. 2025. Dmark: Order-agnostic watermarking for diffusion large language models. *Preprint*, arXiv:2510.02902.

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

KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. 2024a. Advancing beyond identification: Multi-bit watermark for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4031–4055.

KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. 2024b. Advancing beyond identification: Multi-bit watermark for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4031–4055.

Runpeng Yu, Qi Li, and Xinchao Wang. 2025. Discrete diffusion in large language and multimodal models: A survey. *Preprint*, arXiv:2506.13759.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *Preprint*, arXiv:1904.09675.

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. 2023. Provable robust watermarking for ai-generated text. *Preprint*, arXiv:2306.17439.

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

İsmail Tarım and Aytuğ Onan. 2025. Can you detect the difference? *Preprint*, arXiv:2507.10475.

A Proofs

In this section, we provide the derivation of the equation of δ' and the mathematical proof for Theorem 1 regarding the unbiasedness of our proposed watermarking modulation.

A.1 Derivation of the Constraint on δ'

To ensure that the watermarked probability distribution P_w remains a valid probability distribution (i.e., sums to 1), the reduction factor δ' must satisfy a specific constraint relative to the boost factor δ.

Lemma 1. *For the watermarked distribution P_w to satisfy $\sum_{v\in\mathcal{V}} P_w(v) = 1$, the reduction factor must be set as $\delta' = \frac{\delta\tau}{1-\tau}$, where $\tau = \sum_{v\in\mathcal{E}_i} P(v)$ is the total probability mass of the watermark region.*

Proof. The condition for a valid probability distribution requires that the sum of probabilities over the entire vocabulary $\mathcal{V}$ equals 1:

$$\sum_{v\in\mathcal{V}} P_w(v) = 1 \tag{9}$$

Based on the definition of $P_w(v)$ in Eq. (6), we decompose the summation into the watermark region $\mathcal{E}_i$ and the non-watermark region:

$$\sum_{v\in\mathcal{V}} P_w(v) = \tag{10}$$

$$\sum_{v\in\mathcal{E}_i} P(v)(1+\delta) + \sum_{v\notin\mathcal{E}_i} P(v)(1-\delta') \tag{11}$$

$$= \tag{12}$$

$$(1+\delta)\sum_{v\in\mathcal{E}_i} P(v) + (1-\delta')\sum_{v\notin\mathcal{E}_i} P(v) \tag{13}$$

Substituting $\tau = \sum_{v\in\mathcal{E}_i} P(v)$ and recognizing that $\sum_{v\notin\mathcal{E}_i} P(v) = 1-\tau$, we obtain:

$$(1+\delta)\tau + (1-\delta')(1-\tau) = 1 \tag{14}$$

$$(\tau+\delta\tau) + (1-\tau-\delta'(1-\tau)) = 1 \tag{15}$$

$$\tau + \delta\tau + 1 - \tau - \delta'(1-\tau) = 1 \tag{16}$$

Canceling out τ and 1 from both sides:

$$\delta\tau - \delta'(1-\tau) = 0 \tag{17}$$

Solving for δ' yields the necessary constraint:

$$\delta'(1-\tau) = \delta\tau \implies \delta' = \frac{\delta\tau}{1-\tau} \tag{18}$$

This confirms that choosing $\delta' = \frac{\delta\tau}{1-\tau}$ guarantees the conservation of probability mass. □

A.2 Proof of Theorem 1

Theorem 1. *The proposed modulation strategy is unbiased in expectation, i.e., $\mathbb{E}[P_w(v)] = P(v)$ for any token $v \in \mathcal{V}$, under the assumption of a sufficiently large vocabulary.*

Definitions and Randomness. Let $P(v)$ denote the original probability of token v. The watermarking process relies on a secret key sk to generate a pseudorandom watermark sequence W and hash seeds, as described in Sec. 3.2. We define an indicator random variable $X_v(sk)$ representing whether token v falls into the watermark region $\mathcal{E}_i$ under a given key sk:

$$X_v = \mathbb{1}(v \in \mathcal{E}_i) \tag{19}$$

The region $\mathcal{E}_i$ is defined by the intersection of forward and backward constraints (Eq. (5)). We assume the hash function behaves as a random oracle.

Thus, for any token v, the event of satisfying the forward constraint and the event of satisfying the backward constraint are statistically independent, each with probability γ (the partition ratio). Therefore, the expected probability of v being in the watermark region over the space of sk is:

$$\mathbb{E}_{sk}[X_v] = P(\text{Forward}) \cdot P(\text{Backward}) = \gamma^2 \tag{20}$$

The watermarked probability distribution $P_w(v)$ is given by:

$$P_w(v) = X_v P(v)(1+\delta) + (1 - X_v)P(v)(1-\delta') \tag{21}$$

where $\tau = \sum_{u \in \mathcal{V}} X_u P(u)$ is the total mass of the watermark region.

Proof of Unbiasedness. We evaluate the expected value $\mathbb{E}[P_w(v)]$ over the randomness of sk.

$$\begin{aligned}&\mathbb{E}[P_w(v)] = \\ &P(v)\left(\mathbb{E}[X_v](1+\delta) + \mathbb{E}[1 - X_v](1-\delta')\right)\end{aligned} \tag{22}$$

Under the *Large Vocabulary Assumption*, the contribution of any single token to τ is negligible ($P(v) \ll \tau$). According to the Law of Large Numbers, the realized mass τ converges to its expectation:

$$\begin{aligned}\tau &\approx \mathbb{E}[\tau] \\ &= \sum_{u \in \mathcal{V}} P(u)\mathbb{E}[X_u] = \gamma^2 \sum_{u \in \mathcal{V}} P(u) = \gamma^2\end{aligned} \tag{23}$$

Consequently, the dynamic reduction factor δ' converges to a constant dependent only on the hyperparameters:

$$\delta' \approx \frac{\delta\gamma^2}{1-\gamma^2} \tag{24}$$

Substituting $\mathbb{E}[X_v] = \gamma^2$ and the approximated δ':

$$\mathbb{E}[P_w(v)] \approx \tag{25}$$

$$P(v)\left[\gamma^2(1+\delta) + (1-\gamma^2)(1-\delta')\right] \tag{26}$$

$$= P(v)\left[\gamma^2 + \delta\gamma^2 + (1-\gamma^2) - \delta\gamma^2\right] \tag{27}$$

$$= P(v)\left[\gamma^2 + 1 - \gamma^2\right] \tag{28}$$

$$= P(v) \tag{29}$$

Thus, $\mathbb{E}[P_w(v)] = P(v)$. This implies that while individual inference steps are perturbed to embed the signal, the modulation introduces no statistical bias towards any specific token over the distribution of secret keys.

Algorithm 2 Inference with Random Remasking

Require: model p_θ, prompt p_0, answer length L, sampling steps T

1: Set x_T a fully masked sequence of length L.
2: **for** t=T, T-1,$\cdots$, 1 **do**
3: Get masked position $mask\_index$ in x_t
4: Determine the number of tokens to unmask: $N_t \leftarrow \text{Schedule}(t)$
5: $x_0 = \arg\max_{x_0} p_\theta(x_0 | p_0, x_t)$
6: Randomly select a subset $\mathcal{S}_t$ of size N_t from $mask\_index$ to unmask
7: **for all** $i \in \mathcal{S}_t$ **do**
8: $x_{t-1}^{(i)} = x_0^{(i)}$
9: **end for**
10: **end for**

B DLM Inference Algorithm

DLM inference initiates with a fully masked sequence $\mathbf{x}_T$ of fixed length L. Over T sampling timesteps, the model p_θ progressively decodes the sequence. At each step, the model predicts tokens for all positions; however, a *remasking strategy* is employed to determine which predicted tokens are retained and which positions are remasked. For ease of understanding, the above generation process is actually the process of selecting some masked positions from x_T at each step and filling them with the predicted tokens.

The pseudo-code for DLM inference with the Random Remasking strategy is presented in Algorithm 2. In each iteration, the algorithm first identifies the set of indices that are currently masked (Line 3). It then employs a planning schedule to determine N_t, the number of tokens to be decoded at the current step (Line 4). Typically, N_t is a function of time t, such as a linear schedule decoding L/T tokens per step to ensure the sequence is fully generated within T steps. After obtaining the predicted sequence from the model (Line 5), the algorithm randomly samples N_t indices from the available masked positions and updates x_{t-1} by replacing the mask tokens at these indices with the corresponding predicted tokens (Lines 6-7). The generation concludes when all mask tokens have been replaced.

Algorithm 3 shows the pseudocode of inference with the Low-Confidence Remasking Strategy. The overall framework of this algorithm is identical to Algorithm 2, with the sole distinction lying in the criterion for selecting positions to decode. The ob-

Algorithm 3 Inference with Low-Confidence Remasking

Require: model p_θ, prompt p_0, answer length L, sampling steps T

1: Set x_T a fully masked sequence of length L.
2: **for** t=T, T-1,$\cdots$, 1 **do**
3: Get masked position $mask\_index$ in x_t
4: Determine the number of tokens to unmask: $N_t \leftarrow \text{Schedule}(t)$
5: $x_0 = \arg\max_{x_0} p_\theta(x_0|p_0, x_t)$
6: Init confidence $c = p_\theta(x_0|p_0, x_t)_{x_0}$
7: Set non-mask position in c to -1
8: Select a subset $\mathcal{S}_t$ from $mask\_index$ with top-N_t confidence to unmask
9: **for all** $i \in \mathcal{S}_t$ **do**
10: $x_{t-1}^{(i)} = x_0^{(i)}$
11: **end for**
12: **end for**

jective of low-confidence remasking is to remask positions where the model's prediction confidence is low, which is equivalent to prioritizing the decoding of the most certain predictions.

In terms of implementation, Lines 6-7 first calculate the confidence scores (e.g., the maximum predicted probability) for all currently masked positions. Subsequently, Line 8 selects the top N_t indices with the highest confidence scores to be unmasked. This ensures that the generation process is guided by the model's highest certainty at each diffusion step.

C Watermarking Algorithm

In this section, we detail the watermark embedding process and watermark detection process. Before elaborating on our watermarking algorithm in detail, it is necessary to first understand the DLM inference process from the Sec. 2 and Appendix B.

C.1 unbiased bidirectional watermarking

The pseudocode for the entire watermarking process is shown in Algorithm 4. Before watermarking, we need to generate a watermark sequence W and construct a watermark matrix M_W according to Sec. 3.2 (Line 1-2). And then Line 3 initializes a candidate watchlist CW for our regret-based remasking strategy. During diffusion iteration, DLM predicts logits or a probability distribution for all positions. We can gain $p_\theta(x_0|p_0, x_t)$, x_0, and N_t as usual (Line 4-6). Line 7-11 set an extra unmasking budget to determine whether the current step

Algorithm 4 UMR Inference with Watermarking

Require: Model p_θ, Prompt p_0, Steps T, Key sk, Message m, Strength δ
Ensure: Watermarked sequence x^w

1: Generate watermark W from m using sk
2: Construct M_W by sk, m, δ, and γ
3: Initialize Candidate Watchlist CW $\leftarrow \emptyset$
4: Initialize $x_T \leftarrow$ Fully Masked
5: **for** $t = T, T-1, \ldots, 1$ **do**
6: Get standard $p_\theta(x_0|p_0, x_t)$ and N_t
7: **Step 1: Determine Budget**
8: **if** CW contains valid candidates **then**
9: $N_t \leftarrow N_t + 1$ *// Extra budget for regret*
10: $Bud \leftarrow$ `True` *// Extra budget flag*
11: **end if**
12: **Step 2: Watermark Embedding**
13: Select $\mathcal{S}_t$ via standard remasking
14: **for** each position $i \in \mathcal{S}_t$ **do**
15: **if** $\mathcal{I}_{st}^{(i)} > 0$ and $status_i =$ `False` **then**
16: $\tilde{x}^{(i-1)}, \tilde{x}^{(i+1)} \leftarrow$ Eq. 4
17: Compute region $\mathcal{E}_i$ by Eq. 5
18: Modulate p_θ to p_θ^w towards $\mathcal{E}_i$ (Eq. 6)
19: Sample $x_0^{(i)}$ from p_θ^w; Update $x_t^{(i)}$
20: **else**
21: Sample $x_0^{(i)}$ from p_θ; Update $x_t^{(i)}$
22: **end if**
23: **end for**
24: **Step 3: Regret-based Remasking**
25: $x_{t-1}^w \leftarrow \text{REGRETREMASK}(x_t, \text{CW}, Bud)$
26: **end for**
27: **return** x_0^w

should execute regret-based remasking. If the budget is true, we need to select one more position to unmask in standard remasking (Line 13). In Line 15-19, we apply unbiased bidirectional watermarking on the positions in $\mathcal{S}_t$ and unmask x_t by the watermarked x_0^w. If the positions with their prior and next tokens are all masked, they will not participate in watermarking (Line 21). Finally, on Line 25, regret-based remasking is performed according to the budget; missed-watermarked tokens are remasked to allow them to regenerate and embed a watermark. The regret-based remasking algorithm is detailed in Sec. C.2.

C.2 Regret-based Remasking

The process of regret-based remasking strategy with rollback mechanism is shown in Algorithm 5. Line 1 updates CW after the standard remasking.

If the unmasked tokens can not fully satisfy the bidirectional constraint, they will be stored in CW. And if the context of items is unmasked, update their $\mathcal{I}_{st}$ in CW. Line 2-5 are only for the low-confidence strategy with a rollback mechanism: we check whether the unmasked position used the "second chance" afforded by regret-based remasking and determine whether the position should be rolled back to the original token to maintain or enhance the watermark strength. Line 6-8 judge whether an extra budget has been added during the current unmasking step, and performs regret-based remasking only when the budget increases. Line 10-14 are the R-remasking process. Line 11 shows the two remasking position selection method: Low-confidence and Random. Line 12 removes the position from CW and Line 13 remasks the position. After that, Line 14 records a snapshot for the current value of the remasked token (will be used in the low-confidence strategy). If no eligible items are found in CW (after updating CW), just select the low-confidence position in $\mathcal{S}_t$ to remask (Line 16-17).

C.3 Watermark Extraction Algorithm

Thanks to the bidirectional watermark constraint mechanism, the extraction Algorithm 6 is similar to that used for LLM watermarking detection. We first initialize a voting matrix V that is used to recover the watermark. For each token in the text should be detected, we partition the vocabulary into $\mathcal{V}_1$ and $\mathcal{V}_0$ by the prior token and the secret key, and get the watermark index k (Line 2-5). If the token falls into $\mathcal{V}_1$, $V[k][1]$ will accumulate one, otherwise, $V[k][0]$ accumulate one. After iterating over the text, the watermark is extracted from V by majority voting (Line 12-16).

Algorithm 5 Regret-based Remasking Strategy

Require: Current sequence x_t, Watchlist CW, Extra Budget flag
Ensure: Remasked sequence x_{t-1}
1: Update CW: Store tokens not satisfy $C_{\text{fwd}} \wedge C_{\text{bwd}}$ and update $\mathcal{I}_{st}$ if context increased.
2: **Phase 1: Rollback (for Low-Confidence)**
3: **if** Regenerated token v_{new} at pos degrades watermark ($w_{pn}^{new} < w_{pn}^{ori}$) **then**
4: Rollback $x_t^{(pos)}$ to snapshot v_{ori}
5: **end if**
6: **Phase 2: R-remasking Execution**
7: **if** No Extra Budget **then**
8: **return** x_t *// No regret action needed*
9: **end if**
10: **if** exist $\mathcal{I}_{st} > 0$ in CW **then**
11: Select target $k \in$ CW maximizing Stability $\mathcal{I}_{st}$ and minimizing w_{pn} and $conf$ **if** Low-confidence **else** random select $k \in$ CW.
12: Remove k from CW
13: Set $x_t^{(k)} \leftarrow$ `[MASK]`
14: Snapshot current value of $x_t^{(k)}$ as v_{ori}
15: **else**
16: Select standard low-confidence position $k \in \mathcal{S}_t$ to remask
17: $x_t^{(k)} \leftarrow$ `[MASK]`
18: **end if**
19: **return** x_t

D Experiments Settings

D.1 Datasets and Metrics

C4 is a massive dataset derived from Common Crawl and originally introduced by Google for the T5 model. The en subset undergoes rigorous preprocessing, including deduplication, heuristic filtering, and language identification, to retain high-quality natural language text, comprising approximately 365 million documents totaling 305 GB, supporting text summarization, question answering, and machine translation.

ELI5 is a large-scale corpus designed for Long-Form Question Answering, comprising approximately 270,000 complex question-answer pairs. Unlike extractive QA tasks, ELI5 requires models to synthesize information and generate coherent, multi-sentence explanations, making it a standard benchmark for evaluating generative question answering and retrieval-augmented generation (RAG) capabilities.

Alpaca is a pioneering corpus for instruction tuning. Generated via the Self-Instruct framework, this dataset comprises 52,000 synthetic instruction-following demonstrations produced by OpenAI's text-davinci-003. Each example consists of an instruction, an optional input context, and a corresponding output. Alpaca serves as a standard benchmark for aligning with user intent and executing diverse natural language tasks.

Bit Accuracy (Acc.): Acc. quantifies the proportion of correctly recovered bits relative to the total length of the binary watermark. A higher Acc. indicates greater reliability in watermark detection, providing stronger evidence for identifying IP in-

Algorithm 6 UMR Watermark Extraction

Require: sequence x, secret key sk, watermark length L
Ensure: Extracted Watermark W_{ext}

Initialize Voting Matrix $V \in \mathbb{R}^{L\times 2}$ with 0s
for $i = 1$ to $|x|$ **do**
  $seed \leftarrow \text{Hash}(x^{(i-1)}, sk)$
  $k \leftarrow seed \pmod L$
  Generate Partition $\mathcal{V}_0, \mathcal{V}_1$ using $seed$
  if $x^{(i)} \in \mathcal{V}_1$ **then**
    $V[k][1] \leftarrow V[k][1] + 1$
  else
    $V[k][0] \leftarrow V[k][0] + 1$
  end if
end for
$W_{ext} \leftarrow []$
for $k = 0$ to $L - 1$ **do**
  $bit \leftarrow \arg\max(V[k][0], V[k][1])$
  Append bit to W_{ext}
end for
return W_{ext}

fringement.

Perplexity (PPL): PPL serves as a metric to evaluate average text quality. It follows an inverse relationship where lower perplexity scores correlate with higher quality and greater textual coherence.

True Positive Rate at low False Positive Rate (TPR@FPR%): TPR@FPR% records the detection sensitivity under strict error constraints. This metric assesses the model's ability to identify watermarked text while maintaining a negligible rate of misclassifying human-written content, which is critical for practical deployment where avoiding false accusations is paramount.

BERTScore: BERTScore evaluates semantic similarity by computing the cosine similarity between the contextual embeddings of the candidate and reference texts. It captures deep semantic alignment and is robust to paraphrasing, aligning more closely with human judgment on text quality.

ROUGE Score: ROUGE is the standard metric for assessing text summarization quality. ROUGE-1 refers to the overlap of unigrams (individual words) between the system-generated summary and the reference summary, primarily evaluating content informativeness. ROUGE-L measures the longest common subsequence (LCS) between the candidate and reference texts.

BLEU: BLEU is originally designed to evaluate machine translation systems, but it is also used in other natural language processing tasks. It measures the similarity between the response and the reference based on n-gram precision and brevity penalty.

Watermark Coverage Rate (WCR): WCR measures the ratio of successfully watermarked tokens to the total number of tokens in the generated text. A higher WCR typically indicates a stronger watermark presence within the output.

D.2 Implement Details

Hardware and Environment. Our experiments were conducted on a machine equipped with 64 CPUs (Intel(R) Xeon(R) Gold 6426Y), 512GB of memory, and an NVIDIA H100-NVLink GPU (80GB memory), running Ubuntu 22.04, CUDA 11.8, and Python 3.13.

Datasets and Prompting. For the C4 dataset, we simulated an open-ended completion task by using the first 30 tokens of each sample as the prompt for the DLMs. For the ELI5 and Alpaca datasets, the corresponding questions and instructions served as prompts, respectively.

Filtering and Evaluation Metrics. To evaluate watermarking effectiveness across varying generation lengths, we applied filtering criteria to remove invalid outputs. Specifically, we discarded repetitive generations where the frequency of any single token exceeded 20% of the total sequence length. To ensure the actual output lengths aligned with our target brackets, we set minimum length thresholds: for target lengths of 50, 100, 200, and 300, we enforced minimum outputs of 40, 80, 180, and 270 tokens, respectively. For the zero-bit watermarking comparison, we applied identical filtering protocols to the baselines (GSJ and Dmark) to ensure a fair comparison. We employed the OPT-1.3B model (Zhang et al., 2022) to assess the perplexity (PPL) of the generated content. BERTScore, ROUGE-1/ROUGE-L, and BLEU are all utilized the official python package bert_score, rouge_score, and sacrebleu, respectively.

Settings for Downstream Tasks. For text summarization and translation tasks, which inherently involve shorter sequences, we adjusted the generation parameters as follows: maximum generation length was set to 64, with a block size of 16 and 64 denoising steps. The minimum output length filters were set to 30 for summarization and 10 for translation.

Table 5: Uncertainty estimates (Standard Deviation and 95% Confidence Intervals) for Detection Accuracy and Perplexity across datasets at a generation length of 200.

	Acc.			PPL		
Metric	**C4**	**ELI5**	**Alpaca**	**C4**	**ELI5**	**Alpaca**
SD	0.126	0.115	0.117	1.634	1.510	1.676
95% CI	[0.90, 0.94]	[0.92, 0.97]	[0.91, 0.96]	[4.096, 4.875]	[5.272, 5.864]	[4.852, 5.536]

E Uncertainty Estimates

We reported the standard deviations (SD) and 95% confidence intervals (CI) for detection accuracy (Acc.) and perplexity (PPL), results are shown in Table 5. 95% CI is calculated via bootstrap resampling (1,000 iterations). The results show that UMR exhibits remarkable detection stability. The 95% CI for Acc. consistently maintains a lower bound of ≥ 0.90 across all three diverse datasets, with low standard deviations ranging from 0.115 to 0.126. For generation quality, the PPL metrics similarly demonstrate remarkably tight confidence intervals (e.g., [4.096, 4.875] on C4), indicating that the text fluency remains highly consistent. These rigorous statistical estimates provide strong confidence that the superior detection accuracy and low perplexity reported in our study are robust intrinsic properties of UMR rather than artifacts of the sample size.

F Storage & Memory Overhead

The storage and memory overhead results of the watermark matrix M_W across different models are reported in Table 6. The disk space required for M_W is positively correlated with the model's vocabulary size, reaching a maximum of 2.75GB among the evaluated models. During watermarking, loading M_W occupies approximately 15GB of VRAM. Given the memory capacity of modern hardware typically used to serve LLMs and DLMs, we believe this memory overhead is entirely acceptable for practical deployment.

Table 6: Storage and memory footprint of the watermark matrix M_W across different models.

Model	**Vocab Size**	**File Size**	**VRAM (GPU)**
LLaDA Series	126,464	1.9 GB	∼14 GB
Dream	152,064	2.75 GB	∼15 GB
RND	151,936	2.69 GB	∼15 GB

UMR can either calculate M in advance or perform hashing during token generation. To validate this flexibility, we tested and compared the generation speed (tokens/sec) of watermarking with/without M_W: 29 tokens/s and 19 tokens/s.

G Attacks Robustness Comparison

We provide the robustness comparison results of the zero-bit setting, as shown in the Table 7.

Random/Synonym Substitution and Deletion Attacks. Experimental results indicate that our scheme's robustness against these basic edits is slightly weaker than that of the baselines. This is because GSJ and DMark forcefully distort the natural token distribution to achieve a heavy watermark signal. This makes their signals stubborn against word-level dropping or replacement, but it causes higher-quality degradation (as shown in Figure 4 and Table 2, GSJ and DMark exhibit higher PPL, as well as lower BERTScore and ROUGE). In contrast, we prioritized unbiasedness and quality preservation. Even though UMR experiences a minor reduction in detection rate under these specific cases, it remains robust against these attacks.

Paraphrase Attacks. Under paraphrase attacks, which are arguably the most realistic and sophisticated threat vectors employed by adversaries to evade detection, UMR performs comparably, and even slightly better at all experimental cases (e.g., TPR@0.5=0.72 vs. GSJ 0.69 / DMark 0.61).

We believe this trade-off is highly justified for practical deployment. While we concede a slight drop in robustness against basic edits, UMR has two critical capabilities: lower quality degradation and multi-bit capacity.

H Ablation

Justification for Fixed γ. We do not conduct ablation studies on the vocabulary partition ratio γ due to the requirements of the multi-bit, message-agnostic setting. If $\gamma \neq 0.5$ (e.g., $\gamma > 0.5$), the partition sizes for bit 0 and bit 1 would be asymmetric. During watermark extraction—which is performed without access to the original watermark message—this asymmetry creates ambiguity: it becomes impossible to determine whether a

Table 7: Zero-bit Results under deletion, substitution, synonym and paraphrase attacks.

Method	Metric	Deletion Rate				Substitution Rate				Synonym Rate				Paraphrase
		0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4	0.1	0.2	0.3	0.4	L=20, O=20
UMR	TPR@0.5	0.89	**0.83**	0.71	0.27	0.84	0.77	0.58	0.36	0.92	0.90	0.82	0.77	**0.72**
	TPR@1.0	0.89	0.86	0.77	0.36	0.86	0.81	0.65	0.45	0.94	0.91	0.88	0.86	**0.78**
	TPR@5.0	0.93	0.93	0.87	0.58	**0.99**	0.90	0.84	0.73	0.97	0.95	0.95	0.89	**0.82**
GSJ	TPR@0.5	**0.95**	0.82	**0.74**	**0.43**	0.88	**0.79**	**0.74**	**0.45**	**0.93**	0.90	**0.84**	**0.80**	0.69
	TPR@1.0	**0.98**	**0.97**	**0.84**	**0.69**	**0.96**	**0.93**	**0.86**	**0.69**	**0.99**	**0.98**	**0.95**	**0.91**	0.74
	TPR@5.0	**0.99**	**0.97**	**0.91**	**0.78**	0.98	**0.95**	**0.89**	**0.76**	**0.99**	**0.99**	**0.96**	**0.92**	0.78
Dmark	TPR@0.5	0.93	0.78	0.69	0.34	**0.91**	0.73	0.53	0.30	0.92	**0.91**	0.79	0.79	0.61
	TPR@1.0	0.97	0.88	0.82	0.41	0.95	0.83	0.60	0.41	0.96	0.93	0.87	0.86	0.73
	TPR@5.0	0.98	0.95	0.90	0.62	0.97	0.89	0.85	0.66	0.97	0.96	0.94	0.91	0.77

partition's higher token count stems from the watermark signal or simply from its larger vocabulary size. Therefore, to ensure distinguishing bits 0 and 1 is statistically feasible in a blind setting, γ must be fixed at 0.5.

H.1 Impact of Block Length

Table 8 reports the performance of our method under different block length settings in terms of detection accuracy and generation quality. Overall, the proposed strategy remains stable across varying block lengths, with a slight improvement in detection performance as block length increases. Even in the pure diffusion regime, with the generation length constrained to the block length, the method attains an accuracy of 92.5%.

Table 8: Results of the Impact of Block Length.

	$\tilde{B} = 16$	$\tilde{B} = 32$	$\tilde{B} = 64$	pure diffusion*
Acc.	0.935	0.945	0.950	0.925
PPL↓	4.181	4.170	4.801	16.96

* In practice, we observe that LLaDA exhibits limited capability in producing long sequences under pure diffusion. Therefore, the maximum generation length is set to 64 in this setting.

H.2 Impact of Timestep

As shown in Table 9, we vary the timestep t from 64 to 512 to examine the trade-off between detection reliability and linguistic fluency. Increasing t from 64 to 256 leads to a substantial improvement in accuracy, accompanied by higher perplexity. Accuracy saturates at $t = 256$; further increasing t to 512 slightly reduces perplexity but provides no additional accuracy gains. As a larger timestep incurs greater computational cost, t should be selected to balance generation quality and inference latency.

Table 9: Results of the Impact of Timestep.

	$\tilde{t} = 64$	$\tilde{t} = 128$	$\tilde{t} = 256$	$\tilde{t} = 512$
Acc.	0.820	0.913	0.943	0.930
PPL↓	3.592	4.459	4.620	4.337

I Background & Related Work

I.1 Discrete Diffusion Language Model

Discrete Diffusion Language Models (DLMs) have recently emerged as a promising direction (Li et al., 2025). They model text generation as an iterative denoising process (Austin et al., 2021; Sahoo et al., 2024): starting from a completely masked or noisy sequence, they utilize bidirectional attention to predict multiple tokens in parallel, enabling both high-efficiency inference and flexible, non-sequential generation control.

From foundational works such as D3PM (Austin et al., 2021) and Masked Diffusion Models (Sahoo et al., 2024), DLMs have rapidly scaled to multi-billion-parameter architectures. Notable advancements include LLaDA (Nie et al., 2025), which validated discrete diffusion scaling, and LLaDA 1.5 (Zhu et al., 2025), which enhanced utility through alignment techniques: Variance-Reduced Preference Optimization (VRPO). Recently, Dream 7B (Ye et al., 2025) effectively bridged the performance gap with top-tier AR models (e.g., LLaMA 3 (Grattafiori et al., 2024) and Qwen 2.5(Qwen et al., 2025)) while offering up to 10× faster inference, alongside industrial adoptions like Gemini Diffusion (DeepMind, 2025) and Mercury (Labs, 2025). While the capabilities of DLMs have nearly matured, research on content provenance and watermarking for DLMs remains critically underexplored, creating a gap in trustworthy deployment.

I.2 LLM watermarking

LLM watermarking aims at identifying LLM-generated content. The watermarking method in the present work can be categorized into logits-bias (Kirchenbauer et al., 2023; Zhao et al., 2023; Chen et al., 2024; He et al., 2024), probability-distribution modification (Aaronson, 2023; Hu et al., 2023), and sampling-process manipulation (Kuditipudi et al., 2024; Dathathri et al., 2024). These methods leverage the sequential generation feature of LLMs: before generating each token, it uses its prior tokens as a watermark prefix to partition the vocabulary and perform watermarking. Among these methods, the most classic one is the "Red-Green" watermark. It uses the watermark prefix to divide the vocabulary into a red list and a green list. The tokens in the green list represent the watermark, and they apply a positive bias to their logits to increase the probability that these tokens are selected. The final generated text will have a relatively high proportion of green tokens, which achieves watermarking.

Moreover, as tracing the provenance of LLM-generated text has become increasingly important, LLM watermarking now requires explicit message-embedding capability. Recent studies have developed multi-bit watermarking schemes (Fairoze et al., 2023; Cohen et al., 2025; Qu et al., 2025; Yoo et al., 2024b; Feng et al., 2025). They apply the strategy that associates a watermark message with a secret key/hash function or the strategy that maps a message into the partitions of the vocabulary, successfully achieving multi-bit watermarking.

I.3 DLM Watermarking

As LLM watermarks cannot be directly applied to DLMs, and the need to prevent the abuse of DLM-generated content has drawn attention, research on DLM watermarking has begun. Currently, only few studies (Gloaguen et al., 2025; Wu et al., 2025; Bagchi et al., 2025) are proposed, which can be categorized into *context-dependent* watermarking (Gloaguen et al., 2025; Wu et al., 2025) and *context-independent* watermarking (Bagchi et al., 2025).

Context-dependent watermarking schemes introduce a bidirectional watermarking constraint mechanism. Drawing inspiration from LLM watermarking, these methods typically use prior tokens as the watermark prefix to determine the "green" token set for the position selected to unmask. Simultaneously, to ensure the current token serves as a valid watermark prefix for next tokens (allowing them to become "green" tokens), the "green" token selection at the current position is further restricted by the next tokens. Thus, we call the watermark token generation a bidirectional context constraint. A challenge arises when context tokens remain masked, particularly during the initial diffusion stages. To address this, these methods temporarily impute the context using the model's predictions, selecting tokens based on maximum probability (Wu et al., 2025) or maximum expectation (Gloaguen et al., 2025) to determine the bidirectional context. When the entire denoising generation is completed, the generated text will carry the watermark. The bidirectional watermarking constraint adaptively applies the LLM watermarking concept to DLMs, ensuring that the watermark is consistent in form with that of sequential LLM watermarks. Only sequential detection is required when detecting watermarks. However, these methods suffer from two key drawbacks: (1) Unstable watermarking: The mismatch between early-stage predictions and the final generated context can corrupt the watermark signal; (2) Quality Degradation: The accumulated watermarking bias throughout the diffusion process negatively impacts generation quality.

Context-independent watermarking scheme mitigates stability issues by using absolute sequence position as the watermark prefix for watermarking (Bagchi et al., 2025), thereby removing dependence on the evolving context. However, the fixed prefix limits generation diversity and is susceptible to cryptanalytic attacks where the watermark pattern is reverse-engineered from output statistics. Furthermore, position-based dependence offers negligible robustness, as basic edit attacks (e.g., insertion, deletion, paraphrasing) easily destroy the synchronization required for detection. This method cannot be safely and effectively applied in practice.

Overall, DLM watermarking remains at an early exploratory phase. Current solutions exhibit significant vulnerabilities, highlighting the need for advanced methods that better balance text quality, robustness, and watermark capacity.

J Example of Watermarked Texts

An example of a text with and without watermarking is shown below:

Category	Text	Detection
Prompt	DOCA, a company specialized in kitchen furniture and cabinets with its own designs and production facilities, will travel once again to Milan to participate in EuroC	N/A
No Watermark	, the world's largest furniture and interiors trade fair. This event is a significant opportunity for DOCA to showcase its innovative products, network with industry professionals, and explore new business opportunities. EuroC attracts millions of visitors from around the globe, making it an ideal platform for DOCA to present its unique designs and production capabilities. By participating in this event, DOCA aims to: 1. Showcase its latest kitchen furniture and cabinet designs to a global audience. 2. Collaborate with industry experts, designers, and manufacturers to explore new ideas and partnerships. 3. Build strong relationships with potential clients, suppliers, and distributors. 4. Gain insights into market trends and consumer preferences to inform future product development. 5. Enhance the company's reputation and visibility in the kitchen furniture and cabinet industry.	z-score: 0.5636 N/A
UMR	abinets 2021. EuroCabinets 2021 is one of the biggest and most important trade shows in the kitchen and bathroom furniture industry. With more 1,000 exhibitors and 35,000 visitors, EuroCabinets 2021 provides an excellent platform for companies to showcase their products, connect with potential clients, and explore the latest trends, and innovations. DOCA, with its unique designs and production capabilities, offers a wide range of kitchen and and bathroom furniture products that cater to the needs of its clients. DOCA's participation in EuroCabinets 2021 will enable the company to showcase its products to a global audience, connect with potential clients, and explore the latest trends and innovations in the industry. DOCA's participation in EuroCabinets 2021 will also help the company to increase its visibility in the kitchen and bathroom furniture industry, establish relationships with other companies and suppliers, and expand its customer base. Overall, DOCA's participation in EuroCabinets 2021 will be an excellent opportunity for the company to showcase its products and latest innovations in the kitchen and bathroom furniture industry.	z-score: 7.4039 voting matrix $V =$ [[23, 53], [58, 21], [32, 13], [10, 44]] Bit Acc.: 1.0