MULTI-BIT WATERMARKING FOR DISCRETE DIFFUSION LANGUAGE MODELS VIA BITWISE EVIDENCE FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete diffusion language models (dLLMs) offer a new paradigm for language generation, but reliable multi-bit watermark embedding and recovery remain challenging. Iterative denoising involves partially masked, non-causal contexts and uneven evidence accumulation across message bits. This work focuses on embedding multi-bit messages during dLLM generation and recovering them from the resulting text. We propose BEFMark, a watermarking framework that uses bitwise evidence feedback to adapt embedding throughout denoising. BEFMark extracts bitwise evidence from partially masked sequences through context-aware multi-layer voting and estimates candidate tokens' expected evidence contributions under incomplete contexts. This feedback guides a KL-regularized update of the generation distribution, strengthening weakly supported bits while penalizing deviation from the base model's predictions. Experiments on LLaDA-8B-Instruct and Dream-v0-Instruct-7B across multiple datasets demonstrate improved bit recovery accuracy, with additional exact-message recovery gains on LLaDA-8B-Instruct. BEFMark also improves recovery robustness under text editing while maintaining competitive log-perplexity and downstream task performance.

1 INTRODUCTION

Discrete Diffusion Language Models (dLLMs) have recently emerged as an increasingly competitive text-generation paradigm alongside autoregressive language models (Sahoo et al., 2024; Nie et al., 2025; Zhou et al., 2026). Unlike autoregressive language models, which decode tokens one by one in a fixed order, dLLMs generate text through iterative denoising, repeatedly predicting and refining masked positions using bidirectional context and allowing multiple tokens to be updated in parallel. As their generative capabilities and applications continue to expand (You et al., 2026; Li et al., 2025), reliable mechanisms for identifying dLLM-generated content and tracing its provenance are becoming increasingly important.

Text watermarking provides a lightweight and verifiable approach to provenance tracing. Most methods target autoregressive models by deriving keyed signals from generated prefixes and modifying next-token distributions to accumulate detectable evidence (Kirchenbauer et al., 2023; Dathathri et al., 2024; Chen et al., 2025). Multi-bit watermarks extend presence detection to embedded-message recovery, enabling richer provenance information (Feng et al., 2025; Yoo et al., 2024; Xu et al., 2026). Recent studies have adapted watermarking to dLLMs by accounting for partially masked contexts and non-causal denoising, with some also supporting multi-bit payloads (Gloaguen et al., 2026; Hong & No, 2026; Yang et al., 2026). Nevertheless, reliably recovering a complete multi-bit message from dLLM-generated sequence remains challenging.

Reliable multi-bit recovery is complicated by both the evolving denoising state and the requirement that every message bit should be decoded correctly. During dLLM denoising, either a token or its watermark-relevant context may be masked, making it challenging to directly infer the watermark evidence associated with that token from the current partially masked sequence. Moreover, the accumulated evidence can be distributed unevenly across message bits. The complete message can be recovered only when every bit is decoded correctly, therefore a small number of weakly supported bits can become the recovery bottleneck. Without considering the current bitwise evidence, the wa-

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.

termarking rule may over-reinforce bits that already have strong support, while failing to utilize the remaining generation opportunities to improve the reliability of weakly supported bits. Effective multi-bit watermarking therefore requires jointly estimating the future watermark evidence of incomplete positions, adaptively guiding subsequent denoising based on the current bitwise evidence, and maintaining fidelity to the base generation distribution.

To address the aforementioned challenges, we propose Bitwise Evidence Feedback Watermarking (BEFMark), a multi-bit watermarking framework for diffusion large language models that dynamically updates the watermark evidence during the generation process and uses feedback signals to adaptively optimize subsequent generations. BEFMark first applies a context-aware multi-layer voting mechanism to the current partially denoised sequence to accumulate evidence for individual message bits, allowing a single token to contribute to multiple bits. For the remaining masked tokens, we reuse the voting mechanism to estimate the expected evidence increments resulting from their possible realizations under the current dLLM predictions. Building on this and further considering that multi-bit watermarking requires the reliable recovery of different message bits simultaneously, we dynamically adjust the importance of each bit according to its current evidence and combine the resulting weights with the expected future evidence increments to construct a feedback signal. This signal guides a KL-regularized update of the dLLM predictive distributions, prioritizing weakly supported bits while limiting deviation from the base model predictions. Repeating this feedback process throughout denoising improves complete-message recovery within a limited generation length. Experiments on multiple dLLMs and datasets demonstrate the effectiveness and superiority of our method in multi-bit watermark recovery, robustness, and text quality preservation.

2 RELATED WORK

Watermarking for Autoregressive Language Models. Autoregressive language model watermarking typically derives a keyed watermark signal from the generated prefix and modifies next-token selection so that detectable evidence accumulates over the sequence. A representative approach is KGW (Kirchenbauer et al., 2023), which uses a context-dependent hash to partition the vocabulary into green and red lists and increases the logits of green-list tokens. Subsequent studies improve the detectability–quality trade-off by adapting watermark strength to token entropy or the generation distribution (Lee et al., 2024; Wang et al., 2025), while EWD (Lu et al., 2024) instead uses entropy to weight token evidence during detection. Another line develops unbiased watermarks through randomized probability reweighting, which perturbs the next-token distribution under each key but preserves the original distribution in expectation over watermark randomness (Hu et al., 2024; Chen et al., 2025; Dathathri et al., 2024; Wu et al., 2026). Despite their different embedding mechanisms, most existing methods are designed for left-to-right generation, where the prefix used to construct each watermark signal is fully observed and remains fixed.

Multi-bit Watermarking for Language Models. Multi-bit watermarking extends watermark detection to the recovery of an embedded binary message, enabling more detailed content attribution and metadata tracking. MPAC (Yoo et al., 2024) uses position allocation to pseudo-randomly assign each generated token to a message bit and encode its value through vocabulary partitioning. Subsequent methods improve recovery under limited text lengths through different designs. XMark (Xu et al., 2026) introduces a tailored encoder and decoder to improve the quality–recovery trade-off, BiMark (Feng et al., 2025) combines multi-bit encoding with unbiased multi-layer reweighting, and MirrorMark (Jiang et al., 2026) uses measure-preserving mirrored sampling together with a context-based scheduler that balances token assignments. In position-allocation schemes such as MPAC, however, each generated token is assigned to a single message bit, which prevents it from simultaneously contributing evidence to multiple bits.

Watermarking for Diffusion Language Models. Recent studies adapt language model watermarking to the non-sequential generation process of diffusion language models. Gloaguen et al. (2026) models dLLM watermarking as an optimization problem under contextual uncertainty, optimizing the watermark contribution of candidate tokens by modeling the expected distribution of context hashes. DMark (Wu et al., 2025) handles unavailable contexts through predictive and bidirectional watermarking strategies. Rather than reweighting token probabilities, dgMARK (Hong & No, 2026) embeds a watermark by steering the unmasking order toward masked sequence positions whose candidate tokens satisfy a keyed parity constraint. Subsequently, UMR (Yang et al., 2026) extends the

research to multi-bit watermark recovery, enhancing message recovery capabilities in diffusion language models through context-aware bit allocation and remapping mechanisms.

Overall, although these methods account for context uncertainty, decoding order, or remasking dynamics in their watermarking designs, most are restricted to zero-bit watermark detection. To the best of our knowledge, UMR (Yang et al., 2026) is the only method that supports multi-bit recovery for dLLMs, but its message-dependent prefix–vocabulary mapping must either be stored as a precomputed matrix or evaluated through hashing during inference, resulting in a space–time trade-off. In contrast, our method does not require message-specific precomputation. It models the accumulated evidence for individual message bits, estimates the expected gain in an overall watermark score, and uses this gain to optimize the token distribution.

3 METHOD

Discrete diffusion language models generate a length-L sequence through iterative denoising. Specifically, at step t, the current texts are $x_t = (x_{t,1}, \ldots, x_{t,L})$, where masked tokens are represented by [MASK], and model predicts a categorical distribution $p_{t,i}$ for each item i to be recovered. Our goal is to embed a binary message $y = (y_1, \ldots, y_B)$ into the generated sequence, where $y_b \in \{0, 1\}$ and B denotes the length of message. Unlike autoregressive (AR) watermarking methods that derive each token's watermark signal from previously generated tokens, such approaches cannot be directly applied to dLLMs because these preceding tokens may still be masked or unrefined during diffusion denoising. Moreover, different message bits often exhibit varying levels of recovery reliability during generation, and weakly supported bits can thus become a bottleneck in reconstructing the complete message.

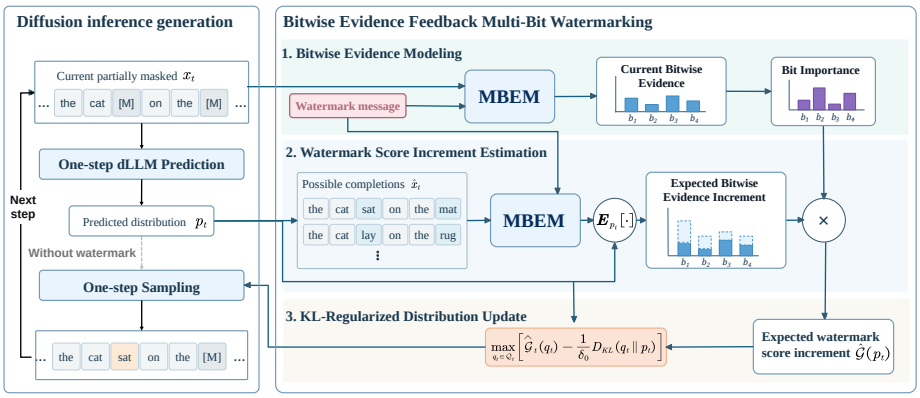


Figure 1: At each stage of denoising-based generation of dLLM, multi-layer Bitwise evidence modeling (MBEM) accumulates bitwise watermark evidence, with which the KL-regularized watermarked probability optimization adapts the generation prob logits for generating the next tokens.

To address these challenges, we propose a multi-bit watermarking via bitwise evidence Feedback (BEFMark) for dLLMs that uses bitwise watermark evidence extracted from the partially masked sequence as feedback for subsequent denoising. As illustrated in Fig. 1, our method mainly comprises the following components. The Multi-layer Bitwise Evidence Modeling (MBEM) converts recovered tokens into bitwise evidences and aggregates them into a unified watermark score, where the sensitivity of score to each bit determines its importance weight. Next, for remaining unrecovered positions, we probabilistically model candidate tokens along with the uncertain states of their contexts to estimate the expected future evidence gain. Finally, this estimated gain guides a KL-regularized update of the predictive distribution. Repeating this feedback process throughout denoising progressively enhances multi-bit recovery. In the following subsections, we formalize the components of our evidence-guided multi-bit watermarking framework at each denoising step t. First, Sec.3.1 describes how to quantify the accumulated watermark score from partially masked sequence. Building upon this score, Sec.3.2 details the estimation of the expected watermark score increment for upcoming generation tokens. Finally, Sec. 3.3 presents an evidence-guided KL-regularized update of the token predictive distribution to maximize this increment.

3.1 BITWISE WATERMARK EVIDENCE AND SCORE MODELING

Recovering a multi-bit message requires every bit to be decoded correctly, while the partially masked sequence may provide various amounts of support for different bits. For each message bit, we track both the net voting margin in favor of its target value and the number of votes it has received to capture the uncertainty. The tracked per-bit statistical information is defined as *bitwise watermark evidence* and the procedure of constructing and accumulating the evidence for layers is called *Multi-layer Bitwise Evidence Modeling (MBEM)*. We subsequently normalize the evidence and aggregate it into a scalar score that places greater emphasis on weakly supported bits.

To compute bitwise watermark evidence from the partially masked sequence x_t, we first define the local context set $\mathcal{C}_{t,i} = \{x_{t,i+s} \mid s \in \mathcal{S}\}$ of each position i, where $\mathcal{S} = \{s_j \mid s_j \in \mathbb{Z} \setminus \{0\}, |s_j| \leq L, 1 \leq j \leq M\}$ denotes the set of offset indices of position i. Bitwise evidence at position i can be computed only when $x_{t,i}$ and all tokens in its local context $\mathcal{C}_{t,i}$ have been recovered. We denote the set of such valid positions by Ω_t. MBEM identifies Ω_t, computes bitwise evidence at each $i \in \Omega_t$, and aggregates the evidence across these positions, as illustrated in Fig. 2(a).

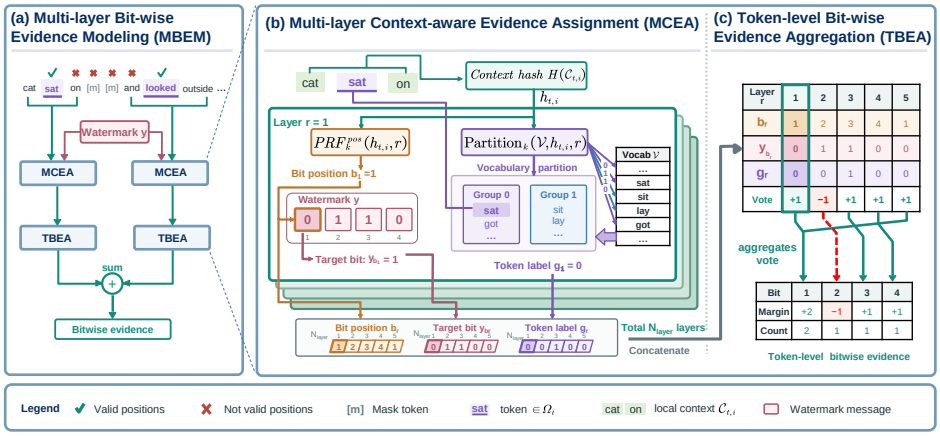


Figure 2: Overview of bitwise watermark evidence modeling. (a) MBEM accumulates evidence from valid positions, where both the token and its local context have been recovered. (b) To allow one token to vote for multiple message bits, MCEA constructs N_{layer} layer-wise votes at each valid position. In each layer, it selects a message-bit position, retrieves the corresponding target bit, and assigns the token a binary label. (c) TBEA compares each layer's token label with its target bit and aggregates the signed votes by message bit to obtain token-level evidence margins and vote coun ts.

Multi-layer Context-aware Evidence Assignment. To allow the token at each $i \in \Omega_t$ to contribute evidence to multiple message bits rather than being restricted to a single bit, we propose *Multi-layer Context-aware Evidence Assignment (MCEA)* to construct N_{layer} voting layers. At each layer, MCEA selects a message-bit position, retrieves the corresponding target bit, and assigns the token at position i a binary label, as shown in Fig. 2(b). To construct these layer-wise votes, MCEA first maps the local context $\mathcal{C}_{t,i}$ to a hash value using a deterministic context-hash function $h_{t,i} = H(\mathcal{C}_{t,i}) = \left(\sum_{j=1}^{M} x_{t,i+s_j}\right) \bmod K$, which maps a local context to $\{0, \ldots, K-1\}$, and K is taken as the size of the token vocabulary. H maps the local context to a discrete hash value, referred to as the context hash. The context hash is shared across layers, while the layer index distinguishes their individual assignments. Given $h_{t,i}$, MCEA uses a secret key k to instantiate two pseudorandom mappings that determine the message-bit position and token label at each layer (Kirchenbauer et al., 2023). Reusing k with the same inputs reproduces both mappings during message recovery.

For message-bit position assignment, we let $\text{PRF}_k^{\text{pos}}$ denote a keyed pseudorandom function that maps $h_{t,i}$ to one of the B message-bit positions. It selects the starting position for the first layer, $b_{t,i,1} = \text{PRF}_k^{\text{pos}}(h_{t,i}) \in \{1, \ldots, B\}$. To distribute the layers evenly across the message, the subsequent layers advance cyclically through the bit positions:

$$b_{t,i,r} = 1 + \big((b_{t,i,1} + r - 2) \bmod B\big), \qquad r \in \{2, \ldots, N_{\text{layer}}\}. \tag{1}$$

Consequently, the numbers of layers assigned to any two message bits differ by at most one. Once $b_{i,r}$ is determined, the corresponding message bit $y_{b_{i,r}}$ serves as the target value for layer r. Let $\mathcal{V}$ denote the vocabulary of the dLLM. At layer r, the context hash $h_{t,i}$ and layer index jointly determine a keyed binary partition, $(\mathcal{V}^0_{t,i,r}, \mathcal{V}^1_{t,i,r}) = \mathrm{Partition}_k(\mathcal{V}, h_{t,i}, r)$, where $\mathrm{Partition}_k$ divides $\mathcal{V}$ into two disjoint, equal-sized subsets associated with labels 0 and 1. The membership of $x_{t,i}$ determines its token label: $g_{t,i,r} = c$ if and only if $x_{t,i} \in \mathcal{V}^c_{t,i,r}$, for $c \in \{0, 1\}$. Thus, the r-th layer yields $(b_{t,i,r}, y_{b_{t,i,r}}, g_{t,i,r})$: the selected message-bit position, its target value, and the observed token label.

Token-level Bitwise Evidence Aggregation. Given the layer-wise outputs constructed above, *Token-level Bitwise Evidence Aggregation (TBEA)* converts the outputs of x_i into evidence for individual message bits, as shown in Fig. 2(c). For layer r, TBEA compares the token label $g_{t,i,r}$ with the corresponding target value $y_{b_{t,i,r}}$ and defines the signed vote $s_{t,i,r} = 2 \cdot \mathbf{1}_{\{g_{t,i,r}=y_{b_{t,i,r}}\}} - 1$. A matching label contributes $+1$, whereas a mismatching label contributes -1. TBEA then aggregates the signed votes assigned to each message bit. For bit b, the evidence margin and vote count are

$$m_{t,i,b} = \sum\nolimits_{r=1}^{N_{\text{layer}}} s_{t,i,r} \cdot \mathbf{1}_{\{b_{t,i,r}=b\}}, \qquad n_{t,i,b} = \sum\nolimits_{r=1}^{N_{\text{layer}}} |s_{t,i,r}| \cdot \mathbf{1}_{\{b_{t,i,r}=b\}}. \tag{2}$$

For message bit b, $m_{t,i,b}$ is the evidence margin contributed by $x_{t,i}$, while $n_{t,i,b}$ is the corresponding vote count. Together, $(m_{t,i,b}, n_{t,i,b})$ constitutes the token-level bitwise evidence. MBEM accumulates these quantities over all positions in Ω_t to obtain bitwise evidence $m_{t,b} = \sum_{i\in\Omega_t} m_{t,i,b}$ and $n_{t,b} = \sum_{i\in\Omega_t} n_{t,i,b}$ for each message bit b.

Watermark Evidence Score. The aggregation above yields the evidence vector $\mathbf{m}_t = (m_{t,1}, \ldots, m_{t,B})$ and vote-count vector $\mathbf{n}_t = (n_{t,1}, \ldots, n_{t,B})$ at denoising step t. Because different bits may receive different numbers of votes, their raw evidence margins are not directly comparable. We therefore define the normalized evidence for bit b as $z_{t,b} = m_{t,b}/\sqrt{n_{t,b} + \epsilon}$, where $\epsilon > 0$ is a numerical stabilizer, and let $\mathbf{z}_t = (z_{t,1}, \ldots, z_{t,B})$. A larger $z_{t,b}$ indicates stronger support for the target value of bit b after accounting for its vote count. We next require a scalar score that summarizes the recovery state of the entire message. A simple average can be dominated by well-supported bits, even though the message is recovered correctly only when every bit is decoded correctly. We therefore aggregate the normalized evidence using a soft minimum

$$R(\mathbf{m}_t, \mathbf{n}_t) = -\tau \log \sum\nolimits_{b=1}^{B} \exp(-z_{t,b}/\tau). \tag{3}$$

where the temperature $\tau > 0$ controls the smoothness of the aggregation. The score increases monotonically with the evidence for every bit and approaches $\min_b z_{t,b}$ as $\tau \to 0$. Moreover, its sensitivity is greater for bits with lower normalized evidence. Thus, maximizing R preferentially strengthens the currently weak bits while retaining a differentiable objective. The next subsection estimates how the unresolved positions may increase this score.

3.2 Watermark Score Increment Estimation

Section 3.1 uses the valid positions Ω_t to quantify the bitwise evidence currently available in the partially masked sequence and summarizes this evidence using $R(\mathbf{m}_t, \mathbf{n}_t)$. As denoising proceeds, additional token–context pairs become valid and contribute new evidence. To use these potential contributions as feedback, we first characterize how a possible completion by denoising changes the current score and then average this change under the current dLLM predictions.

For a possible completion $\tilde{x}$, all masked tokens in the partially masked sequence x_t are instantiated. Let Ω_{full} denote the valid-position set of $\tilde{x}$ under the context construction in Sec. 3.1. The set $\Omega_{\text{full}} \setminus \Omega_t$ therefore contains the positions that become newly valid under this completion, with both their tokens and local contexts fully specified. Applying MBEM to these newly valid positions yields, for each bit b, a total additional evidence margin $\Delta m_{t,b}(\tilde{x})$ and vote count $\Delta n_{t,b}(\tilde{x})$. Collecting these MBEM outputs across all bits gives $\Delta\mathbf{m}_t(\tilde{x}) = (\Delta m_{t,1}(\tilde{x}), \ldots, \Delta m_{t,B}(\tilde{x}))$ and $\Delta\mathbf{n}_t(\tilde{x}) = (\Delta n_{t,1}(\tilde{x}), \ldots, \Delta n_{t,B}(\tilde{x}))$. Together, these increments change the watermark score by $R(\mathbf{m}_t + \Delta\mathbf{m}_t(\tilde{x}), \mathbf{n}_t + \Delta\mathbf{n}_t(\tilde{x})) - R(\mathbf{m}_t, \mathbf{n}_t)$.

Because the completion is probabilistic at each step t, the score increase should be computed as the expectation over all completions generated from the current dLLM predictive distribution. Though already recovered tokens remain fixed, masked tokens follow the distribution $p_{t,i}$. For the i-th token

in t-th step, the probability of a token instantiated as $v \in \mathcal{V}$, either masked or not, can be taken as

$$\pi_{t,i}(v) = \begin{cases} p_{t,i}(v), & x_{t,i} = [\text{MASK}], \\ \mathbf{1}_{\{v=x_{t,i}\}}, & x_{t,i} \neq [\text{MASK}], \end{cases} \qquad \pi_t(\tilde{x}) = \prod_{i=1}^{L} \pi_{t,i}(\tilde{x}_i). \tag{4}$$

Thus, π_t extends the current predictions to the entire sequence by representing each recovered position as a point mass at its observed token. Averaging the completion-specific score increase under π_t gives the expected *watermark score increment*:

$$\mathcal{G}_t = \mathbb{E}_{\tilde{x}\sim\pi_t}\left[R(\mathbf{m}_t + \Delta\mathbf{m}_t(\tilde{x}), \mathbf{n}_t + \Delta\mathbf{n}_t(\tilde{x})) - R(\mathbf{m}_t, \mathbf{n}_t)\right]. \tag{5}$$

The expected watermark score increment $\mathcal{G}_t$ in Eq. (5) serves as the desired feedback quantity. Its direct evaluation, however, requires averaging the soft-min score R in Eq. (3) over all possible completions. We approximate this expectation using a first-order Taylor expansion of R around the current evidence $(\mathbf{m}_t, \mathbf{n}_t)$, reducing it to terms involving only the expected evidence margin and vote-count increments. The resulting approximation is stated in the following proposition.

Proposition 3.1. *Under a first-order expansion and the frozen-count approximation, the expected watermark score increment in Eq. (5) can be replaced, for optimization, by the following surrogate:*

$$\widehat{\mathcal{G}}_t = \sum\nolimits_{b=1}^{B} w_{t,b} \frac{\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})]}{\sqrt{n_{t,b} + \epsilon}}, \qquad w_{t,b} = \mathrm{softmax}(-\mathbf{z}_t/\tau)_b. \tag{6}$$

A detailed derivation is provided in Appendix A.3. Proposition 3.1 expresses the score-increment surrogate as a weighted sum of the expected evidence margin increments across message bits. The weight $w_{t,b}$ increases as the current normalized evidence $z_{t,b}$ decreases, so the evidence constructed by MBEM in Sec. 3.1 determines which bits require greater support. Meanwhile, $\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})]$ quantifies the expected evidence margin increment for bit b under the completion distribution. It therefore remains to compute these expectations for the newly valid positions in $\Omega_{\text{full}} \setminus \Omega_t$.

Expected Evidence Margin Increment. For a completion $\tilde{x}$, let $m_{t,i,b}(\tilde{x})$ denote the evidence margin contributed by position i to bit b. The total evidence margin increment is then $\Delta m_{t,b}(\tilde{x}) = \sum_{i\in\Omega_{\text{full}}\setminus\Omega_t} m_{t,i,b}(\tilde{x})$. By linearity of expectation,

$$\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})] = \sum_{i\in\Omega_{\text{full}}\setminus\Omega_t} \mathbb{E}_{\tilde{x}\sim\pi_t}[m_{t,i,b}(\tilde{x})]. \tag{7}$$

For $i \in \Omega_{\text{full}} \setminus \Omega_t$, MCEA and TBEA determine $m_{t,i,b}(\tilde{x})$ from the token $\tilde{x}_i$ and its context hash $h_{t,i}(\tilde{x})$. When these quantities take the values v and h, respectively, we write the resulting contribution as $m_{t,i,b}(v,h)$. The expected contribution can therefore be computed by averaging $m_{t,i,b}(v,h)$ over the joint distribution of the token and its context hash under π_t. Since the local context excludes position i, the token $\tilde{x}_i$ and its context hash $h_{t,i}(\tilde{x})$ are independent under the product distribution π_t. Their joint distribution thus factorizes into the token marginal $\pi_{t,i}(v)$, given in Eq. 4, and the context-hash marginal $p_h(h_{t,i} \mid \pi_t)$. It therefore remains to compute the latter from the context-token marginals. To enable this computation through convolution, we use the hash function H defined in Sec. 3.1. under a completion $\tilde{x}$, it computes as $h_{t,i}(\tilde{x}) = H\left(\{\tilde{x}_{i+s_j}\}_{j=1}^{M}\right)$. The probability of a hash value $h_{t,i}$ is obtained by summing the probabilities of all context-token configurations that map to $h_{t,i}$. Under the product distribution π_t, these probabilities factorize across positions, and the resulting sum can be expressed using discrete convolution

$$p_h(h_{t,i} \mid \pi_t) = \sum_{\substack{u_1,\ldots,u_M\in\mathcal{V} \\ (\sum_{j=1}^{M} u_j) \bmod K = h_{t,i}}} \prod_{j=1}^{M} \pi_{t,i+s_j}(u_j) = \sum_{\substack{0\le a\le M|\mathcal{V}| \\ a \bmod K = h_{t,i}}} \left(\pi_{t,i+s_1} * \cdots * \pi_{t,i+s_M}\right)(a), \tag{8}$$

where $*$ denotes discrete convolution and a is the sum of the context-token indices before reduction modulo K. Using the independence of the evaluated token and its context hash, the expected total increment is

$$\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})] = \sum_{i\in\Omega_{\text{full}}\setminus\Omega_t} \sum_{v\in\mathcal{V}} \pi_{t,i}(v) \sum\nolimits_{h_{t,i}=0}^{K-1} p_h(h_{t,i} \mid \pi_t) m_{t,i,b}(v, h_{t,i}). \tag{9}$$

The completion distribution π_t is induced by the current predictions p_t over the masked positions. Substituting Eq. (9) into Eq. (6) therefore yields the computable surrogate $\widehat{\mathcal{G}}_t(p_t)$. Replacing the masked-position distributions $p_{t,i}$ with candidate watermarked distributions $q_{t,i}$ yields $\widehat{\mathcal{G}}_t(q_t)$, the optimization objective in the next subsection.

3.3 EVIDENCE-GUIDED KL-REGULARIZED DISTRIBUTION UPDATE

Directly maximizing $\widehat{\mathcal{G}}_t(q_t)$ may cause substantial deviation from the base model distribution. Let $\mathcal{M}_t$ denote the masked positions being optimized, and let $\mathcal{Q}_t = \prod_{i\in\mathcal{M}_t} \Delta(\mathcal{V})$, where $\Delta(\mathcal{V})$ denotes the space of simplex over vocabulary $\mathcal{V}$. To balance evidence improvement against distributional distortion, we consider the KL-regularized objective

$$\max_{q_t\in\mathcal{Q}_t} \quad F_t(q_t), \qquad F_t(q_t) = \widehat{\mathcal{G}}_t(q_t) - \frac{1}{\delta_0}\sum_{i\in\mathcal{M}_t} D_{\mathrm{KL}}(q_{t,i}\|p_{t,i}), \tag{10}$$

where $\delta_0 > 0$ is shared across positions and denoising steps. A larger δ_0 places less weight on distributional fidelity relative to evidence improvement. The shared coefficient imposes a common KL penalty while allowing the resulting distortion to vary across positions. At a fixed denoising step, the observed evidence and its associated bit weights are held fixed. The concavity of the objective in Eq. 10 is not guaranteed, we use entropic mirror ascent on F_t with step size δ_0. At an interior reference distribution $\bar{q}_t$, define the evidence-gradient vectors as $g_{t,i} = \nabla_{q_{t,i}}\widehat{\mathcal{G}}_t(q_t)\Big|_{q_t=\bar{q}_t}$. Each mirror ascent step is equivalent to solving the following KL-regularized linearized subproblem:

$$\max_{q_t\in\mathcal{Q}_t}\sum_{i\in\mathcal{M}_t}\left[\langle g_{t,i}, q_{t,i}\rangle - \frac{1}{\delta_0}D_{\mathrm{KL}}(q_{t,i}\|p_{t,i})\right]. \tag{11}$$

The following proposition gives the unique global solution of the subproblem.

Proposition 3.2. *Suppose $p_{t,i}$ has full support on the finite vocabulary $\mathcal{V}$, and $\widehat{\mathcal{G}}_t$ is differentiable at the interior reference distribution $\bar{q}_t$. For any $\delta_0 > 0$, Eq. 11 has the unique maximizer*

$$q^*_{t,i} = \mathrm{softmax}\left(\log p_{t,i} + \delta_0 g_{t,i}\right), \tag{12}$$

which coincides with an entropic mirror ascent step on F_t from $\bar{q}_t$ with step size δ_0.

The proof is provided in Appendix A.4. Based on the linearized subproblem with its closed-form solution, we iteratively update the distribution q_t as follows. We initialize $q_t^{(0)} = p_t$ and apply

$$q_{t,i}^{(j+1)} = \mathrm{softmax}\left(\log p_{t,i} + \delta_0\, \nabla_{q_{t,i}}\widehat{\mathcal{G}}_t(q_t)\Big|_{q_t=q_t^{(j)}}\right), \quad j = 0,\dots \tag{13}$$

Appendix A.4 establishes linear convergence to the unique global maximizer when $\nabla\widehat{\mathcal{G}}_t$ is L_t-Lipschitz and $\delta_0 L_t < 2$. We cannot guarantee convergence outside this regime, but sensitivity studies in Appendices B.5.2 and B.5.3 provide empirical support for the approach across different values of δ_0 and iteration counts J. The watermarking algorithm is presented in Appendix A.1.

3.4 WATERMARK MESSAGE RECOVERY

Let x_0 denote the watermarked sequence generated by the dLLM. Given x_0 and the watermark key k introduced with MCEA in Sec.3.1, message recovery repeats the message-bit assignment and token-labeling steps of MCEA on the token–context pairs in x_0. For each valid position $i \in \Omega_0$ and layer r, these steps reproduce the message-bit position $b_{0,i,r}$ and token label $g_{0,i,r}$. Each layer contributes one vote of value $g_{0,i,r}$ to message bit $b_{0,i,r}$. Because watermark embedding favors $g_{0,i,r} = y_{b_{0,i,r}}$, the majority of votes assigned to message bit b is expected to agree with its embedded value y_b. For message bit b and candidate value $l \in \{0,1\}$, let $C_{b,l}$ denote the number of votes:

$$C_{b,l} = \sum\nolimits_{i\in\Omega_0}\sum\nolimits_{r=1}^{N_{\mathrm{layer}}} \mathbf{1}_{\{b_{0,i,r}=b\}}\mathbf{1}_{\{g_{0,i,r}=l\}}. \quad l\in\{0,1\}, \tag{14}$$

The bit is recovered by majority voting $\hat{y}_b = \arg\max_{l\in\{0,1\}} C_{b,l}$. In the rare event that $C_{b,0} = C_{b,1}$, $\hat{y}_b$ is set to 0 or 1 with equal probability. The recovered watermarking message is $\hat{y} = (\hat{y}_1,\dots,\hat{y}_B)$. The complete message-recovery algorithm is provided in Appendix A.2.

4 EXPERIMENTS

We conduct watermarking experiments across different lengths of bits and generated texts. We also assess robustness under text-editing attacks, evaluate the impact of watermarking on downstream summarization and translation tasks, and present ablation studies and parameter analyses.

Experimental Setup. We evaluate BEFMark on two discrete diffusion language models, LLaDA-8B-Instruct (Nie et al., 2025) and Dream-v0-Instruct-7B (Ye et al., 2025), across three datasets: ELI5 (Fan et al., 2019), FiQA (Maia et al., 2018), and C4 (Raffel et al., 2020). We use the bit recovery accuracy (Acc) between the embedded and recovered messages to evaluate watermark recovery. Generation quality is evaluated using log-PPL ($\log_e(\mathrm{PPL})$), where PPL denotes perplexity computed with Qwen3-30B-A3B-Instruct-2507 (Yang et al., 2025). Reported values are averaged over test samples, higher Acc and lower log-PPL are better. Detailed generation, sampling, and watermark settings are provided in Appendix B.1.

Compared Baselines. We compare BEFMark with UMR (Yang et al., 2026), a multi-bit watermark designed for dLLMs. We further adapt two representative multi-bit watermarking methods of AR models, MPAC (Yoo et al., 2024) and BiMark (Feng et al., 2025), to dLLM generation. Both methods use preceding tokens to compute context hashes, but these tokens may remain masked during denoising. We therefore apply DMark's context-prediction strategy (Wu et al., 2025) to estimate the unavailable preceding context while preserving the original embedding rules of each method. We refer to the adapted methods as PredMPAC and PredBiMark.

Table 1: Bit recovery accuracy (Acc, ↑) and log-PPL (↓) for 8-, 16-, and 32-bit messages at different generation lengths. Results are averaged over ELI5, FiQA, and C4. Best values are bolded.

Bits	Method	LLaDA-8B-Instruct						Dream-v0-Instruct-7B					
		300		500		800		300		500		800	
		Acc	log-PPL	Acc	log-PPL	Acc	log-PPL	Acc	log-PPL	Acc	log-PPL	Acc	log-PPL
8	PredMPAC	82.87	2.951	87.41	2.749	88.85	2.471	77.48	1.928	77.63	1.621	77.63	1.500
	PredBiMark	92.45	2.769	95.16	2.510	95.72	**2.003**	89.52	2.147	89.02	1.690	88.08	**1.261**
	UMR	91.51	**2.482**	93.60	2.234	94.67	2.051	90.05	2.063	91.01	1.810	91.48	1.668
	BEFMark	**97.65**	2.534	**98.06**	**2.227**	**99.19**	2.033	**96.79**	**1.806**	**96.23**	**1.581**	**97.13**	1.442
16	PredMPAC	75.48	2.926	79.11	2.754	82.08	2.466	71.30	1.972	71.64	**1.605**	69.21	1.481
	PredBiMark	85.79	2.761	88.13	2.506	90.21	2.020	81.57	2.119	82.07	1.665	81.68	**1.288**
	UMR	87.02	2.492	91.29	2.261	93.61	2.069	80.29	2.092	80.56	1.831	85.51	1.713
	BEFMark	**93.51**	**2.488**	**95.57**	**2.240**	**96.46**	**2.009**	**93.94**	**1.874**	**94.57**	1.639	**93.89**	1.491
32	PredMPAC	68.52	2.928	71.78	2.751	74.44	2.458	64.93	2.007	65.29	**1.695**	65.75	1.503
	PredBiMark	76.23	2.773	80.35	2.505	82.29	**2.012**	72.44	2.159	72.65	1.808	72.24	**1.251**
	UMR	78.76	2.487	83.51	2.255	87.30	2.043	73.11	2.072	74.73	1.821	76.98	1.662
	BEFMark	**87.21**	**2.462**	**90.08**	**2.203**	**92.30**	2.031	**89.21**	**1.993**	**89.85**	1.757	**89.67**	1.491

Recovery Accuracy and Quality. We evaluate 8-, 16-, and 32-bit messages at generation lengths of 300, 500, and 800 tokens. For each configuration, Acc is averaged over samples from ELI5, FiQA, and C4. Table 1 reports Acc and log-PPL for both dLLMs. BEFMark achieves the highest Acc in every reported configuration, outperforming the strongest baseline by 2.85–16.10 percentage points. It also achieves the best or second-best log-PPL in nearly all configurations, demonstrating strong recovery while maintaining competitive generation quality. Recovery–quality trade-off across all configurations is provided in Appendix B.2. Qualitative examples are provided in Appendix C.

Complete-Message Reliability. Average Acc cannot directly reveal whether the whole embedded watermark message is recovered. We further report the exact-message recovery rate (EMR), defined as the proportion of test samples for which the recovered multi-bit message exactly matches the embedded message. Fig. 3 reports EMR for 500-token generations on LLaDA-8B-Instruct. At 8 and 16 bits, BEFMark achieves EMRs of 92.67% and 57.17%, outperforming the strongest baseline by 23.84 and 26.17 percentage points. At 32 bits, all baselines nearly fail at exact-message recovery, with EMRs of at most 1.17%, whereas BEFMark still exactly recovers 14.33% of the messages. EMR results across all three generation lengths and the corresponding acc distributions of samples are provided in Appendix B.3. Overall, BEFMark's bit-level recovery gains enforce stronger ability to recover complete multi-bits.

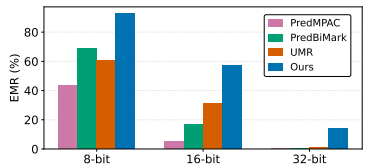


Figure 3: EMR on LLaDA-8B-Instruct for 500-token generations. Results are averaged over ELI5, FiQA, and C4.

Robustness to Text Editing. To verify the model's robustness against text omission and substitution, we randomly select a certain proportion of words from each text. The word proportion is indicated by a subscript at the bottom right. We then conduct three types of experiments: WD: word deletion, SS:

synonym substitution using WordNet (Miller, 1995), CASS: context-aware synonym substitution using (Devlin et al., 2019). Table 2 reports post-attack bit recovery accuracy for 8-, 16-, and 32-bit messages embedded in 500-token texts generated by LLaDA-8B-Instruct. BEFMark performs best across all attack settings and message sizes, exceeding the strongest baseline by 0.53–5.29, 2.35–3.98, and 2.91–5.14 percentage points at 8, 16, and 32 bits, demonstrating the robustness.

Table 2: Robustness under text-editing attacks for 8-, 16-, and 32-bit messages.

Method	8-bit			16-bit			32-bit		
	$WD_{0.1}$	$WD_{0.2}$	$WD_{0.3}$	$WD_{0.1}$	$WD_{0.2}$	$WD_{0.3}$	$WD_{0.1}$	$WD_{0.2}$	$WD_{0.3}$
PredMPAC	87.79	85.75	81.22	76.62	74.52	71.24	69.93	68.38	65.94
PredBiMark	96.00	94.93	91.79	87.29	85.54	81.26	78.33	76.44	72.94
UMR	94.30	93.41	90.15	89.23	87.61	83.22	80.63	78.78	74.23
BEFMark	**97.18**	**95.96**	**92.56**	**92.44**	**90.26**	**85.67**	**85.77**	**82.86**	**77.50**
	$SS_{0.1}$	$SS_{0.2}$	$CASS_{0.1}$	$SS_{0.1}$	$SS_{0.2}$	$CASS_{0.1}$	$SS_{0.1}$	$SS_{0.2}$	$CASS_{0.1}$
PredMPAC	86.20	82.07	86.05	76.31	72.11	72.07	68.97	65.92	65.96
PredBiMark	94.79	91.43	91.62	85.86	81.78	81.74	73.25	69.18	70.07
UMR	93.71	90.12	89.98	88.16	83.87	83.22	79.05	74.88	74.99
BEFMark	**96.46**	**91.96**	**96.91**	**90.71**	**86.22**	**87.20**	**83.69**	**77.79**	**79.29**

Downstream Task Performance. To assess whether watermarking preserves downstream performance, we evaluate a 4-bit message on LLaDA-8B-Instruct for CNN/DailyMail summarization (See et al., 2017) and WMT19 German-to-English translation (Ng et al., 2019). We use a smaller message because downstream outputs are shorter than those in main experiments. We randomly sample 200 examples per dataset whose reference targets exceed 150 tokens. We report BERTScore (Zhang* et al., 2020), ROUGE-1, and ROUGE-L (Lin, 2004) for summarization, and BERTScore and BLEU (Papineni et al., 2002) for translation. All methods use the same model and decoding settings. Table 3 shows that BEFMark remains close to unwatermarked generation on both tasks while achieving recovery Acc of 87.25% and 81.14% for summarization and translation, respectively. Among watermarked methods, BEFMark obtains the best summarization scores despite a small Acc gap to PredMPAC, whose task quality degrades substantially. On translation, BEFMark achieves the highest Acc and BERTScore, with BLEU comparable to the best result. These results show that BEFMark preserves downstream task quality while retaining reliable watermark recovery.

Table 3: Performance on text summarization and machine translation.

Methods	Text Summarization				Machine Translation		
	BERTScore ↑	ROUGE-1 ↑	ROUGE-L ↑	Acc. ↑	BERTScore ↑	BLEU ↑	Acc. ↑
No watermark	22.28	45.64	25.17	–/–	50.92	31.23	–/–
PredMPAC	12.26	39.04	20.13	**90.20**	38.23	20.79	79.50
PredBiMark	<u>20.98</u>	<u>44.17</u>	<u>24.52</u>	54.12	49.22	29.89	52.03
UMR	20.44	44.12	24.52	78.81	<u>49.44</u>	**30.40**	<u>80.02</u>
BEFMark	**21.32**	**45.34**	**24.69**	<u>87.25</u>	**49.65**	<u>30.28</u>	**81.14**

Additional Analyses. We further visualize the evolution of bitwise evidence during feedback-guided denoising in Appendix B.4. We also conduct ablation and sensitivity studies in Appendix B.5, including adaptive bit weighting, the update scale δ_0, the number of optimization iterations, the score temperature τ, the number of voting layers, layer-wise bit assignment, context selection, and the diffusion sampling strategy. We also present runtime and peak GPU memory in Appendix B.6.

5 Conclusion

We present BEFMark, an exploration for multi-bit watermark embedding and recovery in discrete diffusion language models. Using evolving bitwise evidence to guide embedding, BEFMark strengthens weakly supported bits under partially masked, non-causal contexts. Experiments show improved bit recovery accuracy and robustness to evaluated token-level edits, with exact-message recovery gains on LLaDA-8B-Instruct, while maintaining competitive text quality and downstream performance. As an initial study, reliable recovery at higher payloads and robustness to broader text transformations remain open challenges. Future work will address these challenges, refine the theoretical analysis of evidence dynamics, and extend BEFMark to multimodal diffusion models.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing, including improving the clarity, grammar, and presentation of author-written text. We also used generative AI tools to assist with reviewing mathematical proofs and visual presentation of figures, such as refining color schemes and improving visual appearance. The methodologies, experimental results and quantitative data reported in this paper were conducted by the authors, and AI-assisted modifications to figures were limited to their visual presentation without changing the underlying data or results. We reviewed all AI-assisted materials for accuracy and consistency with the intended content. We take responsibility for the final content of this work, including all text, claims, figures, and other artifacts produced with the assistance of generative AI.

REFERENCES

Ruibo Chen, Yihan Wu, Junfeng Guo, and Heng Huang. Improved unbiased watermark for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 20587–20601, 2025.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel, Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, Ilia Shumailov, Ciprian Baetu, Sven Gowal, Demis Hassabis, and Pushmeet Kohli. Scalable watermarking for identifying large language model outputs. *Nature*, 634 (8035):818–823, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, 2019.

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

Xiaoyan Feng, He Zhang, Yanjun Zhang, Leo Yu Zhang, and Shirui Pan. BiMark: Unbiased multilayer watermarking for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Thibaud Gloaguen, Robin Staab, Nikola Jovanović, and Martin Vechev. Watermarking diffusion language models. In *International Conference on Learning Representations*, pp. 100104–100149, 2026.

Pyo Min Hong and Albert No. dgMARK: Decoding-guided watermarking for diffusion language models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. In *International Conference on Learning Representations*, pp. 45408–45436, 2024.

Ya Jiang, Massieh Kordi Boroujeny, Surender Suresh Kumar, and Kai Zeng. MirrorMark: Generalizable mirrored sampling for multi-bit LLM watermarking. *arXiv preprint arXiv:2601.22246*, 2026.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 17061–17084, 2023.

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. Who wrote this code? Watermarking for code generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 4890–4911, 2024.

Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. LaViDa: A large diffusion language model for multimodal understanding. In *Advances in Neural Information Processing Systems*, pp. 105101–105134, 2025.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, 2004.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. An entropy-based text watermarking detection method. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 11724–11735, 2024.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. Www'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, pp. 1941–1942, 2018.

George A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11): 39–41, 1995.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation*, pp. 314–319, 2019.

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 *Advances in Neural Information Processing Systems*, pp. 50608–50646, 2025.

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

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. *Journal of Machine Learning Research*, 21(140):1–67, 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 *Advances in Neural Information Processing Systems*, pp. 130136–130184, 2024.

Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1073–1083, 2017.

Zongqi Wang, Tianle Gu, Baoyuan Wu, and Yujiu Yang. MorphMark: Flexible adaptive watermarking for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 4842–4860, 2025.

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

Yihan Wu, Ruibo Chen, Georgios Milis, and Heng Huang. An ensemble framework for unbiased language model watermarking. In *International Conference on Learning Representations*, pp. 17899–17917, 2026.

Jiahao Xu, Rui Hu, Olivera Kotevska, and Zikai Zhang. XMark: Reliable multi-bit watermarking for LLM-generated texts. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 14747–14763, 2026.

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

Ke Yang, Dongyang Liang, Jing Yu, Shuguang Yuan, and Chi Chen. You can have a second chance: Unbiased and multi-bit watermarking for diffusion language models with regret-based remasking. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 28141–28160, 2026.

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.

KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. 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*, pp. 4031–4055, 2024.

Zebin You, Shen Nie, Xiaolu Zhang, JUN ZHOU, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. LLaDA-v: Large language diffusion models with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10093–10105, 2026.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020.

Zhanhui Zhou, Lingjie Chen, Hanghang Tong, and Dawn Song. dLLM: Simple diffusion language modeling. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 78–88, 2026.

A ALGORITHMS

This section provides the complete watermark embedding and message-recovery algorithms for BEFMark. Algorithm 1 implements the default shared-scale KL-regularized embedding procedure by modeling accumulated watermark evidence, estimating the expected watermark score increment, and updating token predictive distributions during dLLM denoising. Algorithm 2 recomputes the keyed layer-wise assignments and recovers each message bit by majority voting.

A.1 WATERMARK EMBEDDING ALGORITHM

Algorithm 1 Watermark embedding via bitwise evidence feedback during dLLM denoising

Require: dLLM p_θ, message y, watermark key k, denoising steps T, successive linearization iterations J, shared scale δ_0

Initialize x_T with masked generation positions
for $t = T, \dots, 1$ **do**
  Obtain the base predictions $p_t \leftarrow p_\theta(\cdot \mid x_t, t)$
  Identify Ω_t and compute $(\mathbf{m}_t, \mathbf{n}_t)$ using MBEM
  Construct $\widehat{\mathcal{G}}_t(\cdot)$ as in Sec. 3.2
  Initialize $q_t^{(0)} \leftarrow p_t$
  for $j = 0, \dots, J-1$ **do**
    $\mathbf{g}_t^{(j)} \leftarrow \nabla_{q_t} \widehat{\mathcal{G}}_t(q_t)\Big|_{q_t = q_t^{(j)}}$
    For each optimized position i, set
      $q_{t,i}^{(j+1)} \leftarrow \mathrm{softmax}\Big(\log p_{t,i} + \delta_0 g_{t,i}^{(j)}\Big)$
  end for
  Denoise x_t using $q_t^{(J)}$ to obtain x_{t-1}
end for
return Watermarked sequence x_0

A.2 WATERMARK MESSAGE RECOVERY ALGORITHM

Algorithm 2 Watermark message recovery

Require: Watermarked sequence x_0, watermark key k, message length B, layer count N_{layer}

Identify the valid positions Ω_0 using the context construction in Sec.3.1
Initialize $C_{b,l} \leftarrow 0$ for every $b \in \{1, \dots, B\}$ and $l \in \{0, 1\}$
for $i \in \Omega_0$ **do**
  for $r = 1, \dots, N_{\text{layer}}$ **do**
    Recompute $b_{0,i,r} \in \{1, \dots, B\}$ and $g_{0,i,r} \in \{0, 1\}$ using the keyed MCEA mappings
    $C_{b_{0,i,r}, g_{0,i,r}} \leftarrow C_{b_{0,i,r}, g_{0,i,r}} + 1$
  end for
end for
for $b = 1, \dots, B$ **do**
  $\hat{y}_b \leftarrow \arg\max_{l \in \{0,1\}} C_{b,l}$
end for
return Recovered message $\hat{y} = (\hat{y}_1, \dots, \hat{y}_B)$

A.3 DERIVATION OF THE WATERMARK SCORE INCREMENT SURROGATE

Proposition 3.1. *Under a first-order expansion and the frozen-count approximation, the expected watermark score increment in Eq. (5) can be replaced, for optimization, by the following surrogate:*

$$\widehat{\mathcal{G}}_t = \sum\nolimits_{b=1}^{B} w_{t,b} \frac{\mathbb{E}_{\tilde{x} \sim \pi_t}[\Delta m_{t,b}(\tilde{x})]}{\sqrt{n_{t,b} + \epsilon}}, \qquad w_{t,b} = \mathrm{softmax}(-\mathbf{z}_t/\tau)_b. \tag{6}$$

Proof. The current evidence vectors are $\mathbf{m}_t = (m_{t,1}, \ldots, m_{t,B})$ and $\mathbf{n}_t = (n_{t,1}, \ldots, n_{t,B})$. For a completion $\tilde{x}$, a first-order expansion of the score around the current evidence $(\mathbf{m}_t, \mathbf{n}_t)$ gives

$$\begin{aligned} &R(\mathbf{m}_t + \Delta\mathbf{m}_t(\tilde{x}), \mathbf{n}_t + \Delta\mathbf{n}_t(\tilde{x})) - R(\mathbf{m}_t, \mathbf{n}_t) \\ &\quad\approx \sum_{b=1}^{B} \frac{\partial R}{\partial m_{t,b}} \Delta m_{t,b}(\tilde{x}) + \sum_{b=1}^{B} \frac{\partial R}{\partial n_{t,b}} \Delta n_{t,b}(\tilde{x}). \end{aligned} \tag{15}$$

Since R depends on $m_{t,b}$ and $n_{t,b}$ through the normalized evidence $z_{t,b}$, the chain rule gives

$$\frac{\partial R}{\partial m_{t,b}} = \frac{\partial R}{\partial z_{t,b}} \frac{\partial z_{t,b}}{\partial m_{t,b}}, \qquad \frac{\partial R}{\partial n_{t,b}} = \frac{\partial R}{\partial z_{t,b}} \frac{\partial z_{t,b}}{\partial n_{t,b}}. \tag{16}$$

From Eq. (3), the derivative of the soft minimum with respect to the normalized evidence is

$$\frac{\partial R}{\partial z_{t,b}} = \frac{\exp(-z_{t,b}/\tau)}{\sum_{b'=1}^{B} \exp(-z_{t,b'}/\tau)} = w_{t,b}. \tag{17}$$

The definition $z_{t,b} = m_{t,b}/\sqrt{n_{t,b} + \epsilon}$ further gives

$$\frac{\partial z_{t,b}}{\partial m_{t,b}} = \frac{1}{\sqrt{n_{t,b} + \epsilon}}, \qquad \frac{\partial z_{t,b}}{\partial n_{t,b}} = -\frac{m_{t,b}}{2(n_{t,b} + \epsilon)^{3/2}}. \tag{18}$$

Substituting Eqs. (17) and (18) into the chain-rule identities above yields

$$\frac{\partial R}{\partial m_{t,b}} = \frac{w_{t,b}}{\sqrt{n_{t,b} + \epsilon}}, \qquad \frac{\partial R}{\partial n_{t,b}} = -\frac{w_{t,b} m_{t,b}}{2(n_{t,b} + \epsilon)^{3/2}}. \tag{19}$$

Substituting Eq. (19) into the first-order expansion in Eq. (15) and using the definition of $\mathcal{G}_t$ in Eq. (5) give

$$\begin{aligned} \mathcal{G}_t &= \mathbb{E}_{\tilde{x}\sim\pi_t} \left[R(\mathbf{m}_t + \Delta\mathbf{m}_t(\tilde{x}), \mathbf{n}_t + \Delta\mathbf{n}_t(\tilde{x})) - R(\mathbf{m}_t, \mathbf{n}_t) \right] \\ &\approx \mathbb{E}_{\tilde{x}\sim\pi_t} \left[\sum_{b=1}^{B} \frac{w_{t,b} \Delta m_{t,b}(\tilde{x})}{\sqrt{n_{t,b} + \epsilon}} - \sum_{b=1}^{B} \frac{w_{t,b} m_{t,b} \Delta n_{t,b}(\tilde{x})}{2(n_{t,b} + \epsilon)^{3/2}} \right] \\ &= \sum_{b=1}^{B} w_{t,b} \frac{\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})]}{\sqrt{n_{t,b} + \epsilon}} - \sum_{b=1}^{B} \frac{w_{t,b} m_{t,b}}{2(n_{t,b} + \epsilon)^{3/2}} \mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta n_{t,b}(\tilde{x})]. \end{aligned} \tag{20}$$

The last equality follows from the linearity of expectation, since $w_{t,b}$, $m_{t,b}$, and $n_{t,b}$ are determined by the current evidence and are fixed with respect to $\tilde{x}$. The second sum in Eq. 20 reflects only the change in vote-count normalization, whereas the first sum captures the signed evidence that directly supports the target message bits. Under cyclic assignment, each newly valid position distributes its N_{layer} votes broadly across the message: every bit receives the same number of votes when N_{layer} is a multiple of B, and otherwise the per-position counts differ by at most one. The relatively balanced vote-count trajectories in Fig. 14 further show that the accumulated counts remain close in practice. Explicitly retaining the second sum would require estimating and differentiating the expected count increments in addition to the expected evidence margins, despite their limited variation across bits. We therefore adopt a frozen-count approximation that treats these count increments as fixed when comparing candidate distributions. This removes the additional count-estimation term and keeps the feedback objective simple and efficient.

Under this approximation, the second sum becomes a candidate-independent term, which we denote by C_t. The first-order approximation can then be written as

$$\mathcal{G}_t \approx C_t + \sum_{b=1}^{B} w_{t,b} \frac{\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})]}{\sqrt{n_{t,b} + \epsilon}}. \tag{21}$$

Because C_t does not change the ranking of candidate distributions or the optimizer, we omit it and define the optimization surrogate as

$$\widehat{\mathcal{G}}_t = \sum_{b=1}^{B} w_{t,b} \frac{\mathbb{E}_{\tilde{x}\sim\pi_t}[\Delta m_{t,b}(\tilde{x})]}{\sqrt{n_{t,b} + \epsilon}}, \tag{22}$$

Thus, $\widehat{\mathcal{G}}_t$ has the form stated in Eq. (6), completing the proof. □

A.4 Theoretical Analysis of KL-Regularized Distribution Update

Proposition 3.2. *Suppose $p_{t,i}$ has full support on the finite vocabulary $\mathcal{V}$, and $\widehat{\mathcal{G}}_t$ is differentiable at the interior reference distribution $\bar{q}_t$. For any $\delta_0 > 0$, Eq. 11 has the unique maximizer*

$$q^*_{t,i} = \mathrm{softmax}\left(\log p_{t,i} + \delta_0 g_{t,i}\right), \tag{12}$$

which coincides with an entropic mirror ascent step on F_t from $\bar{q}_t$ with step size δ_0.

Proof. Fix the gradient vectors $g_{t,i}$. The objective in Eq. (11) separates across positions. Because $p_{t,i}$ has full support, $D_{\mathrm{KL}}(q_{t,i}\|p_{t,i})$ is strictly convex in $q_{t,i}$ on the probability simplex. Hence, the subproblem objective is strictly concave.

For each optimized position i, let $\alpha_{t,i}$ be the multiplier for $\sum_{v\in\mathcal{V}} q_{t,i}(v) = 1$. The Lagrangian is

$$\mathcal{L}_{t,i}(q_{t,i}, \alpha_{t,i}) = \langle g_{t,i}, q_{t,i}\rangle - \frac{1}{\delta_0}\sum_{v\in\mathcal{V}} q_{t,i}(v)\log\frac{q_{t,i}(v)}{p_{t,i}(v)} + \alpha_{t,i}\left(\sum_{v\in\mathcal{V}} q_{t,i}(v) - 1\right).$$

Differentiating with respect to $q_{t,i}(v)$ gives

$$[g_{t,i}]_v - \frac{1}{\delta_0}\left(\log\frac{q_{t,i}(v)}{p_{t,i}(v)} + 1\right) + \alpha_{t,i} = 0.$$

Therefore,

$$q_{t,i}(v) = p_{t,i}(v)\exp(\delta_0[g_{t,i}]_v)\exp(\delta_0\alpha_{t,i} - 1).$$

Enforcing $\sum_{v\in\mathcal{V}} q_{t,i}(v) = 1$ yields the unique stationary point

$$q^*_{t,i}(v) = \frac{p_{t,i}(v)\exp(\delta_0[g_{t,i}]_v)}{\sum_{u\in\mathcal{V}} p_{t,i}(u)\exp(\delta_0[g_{t,i}]_u)} = \mathrm{softmax}(\log p_{t,i} + \delta_0 g_{t,i})_v.$$

Full support of $p_{t,i}$ ensures that $q^*_{t,i}$ is feasible and interior. Since the objective is strictly concave, this stationary point is the unique global maximizer. Thus, Eq. (12) is the unique global solution of Eq. (11). □

Theorem A.1. *Fix a denoising step t, with the base predictions, observed evidence, and associated bit weights held fixed. Suppose that $p_{t,i}$ has full support for every $i \in \mathcal{M}_t$, and that $\widehat{\mathcal{G}}_t$ is continuously differentiable on a neighborhood of $\mathcal{Q}_t$, with an L_t-Lipschitz gradient on $\mathcal{Q}_t$. If $\delta_0 > 0$ and $\delta_0 L_t < 2$, then the iterates*

$$q^{(j+1)}_{t,i} = \mathrm{softmax}\Big(\log p_{t,i} + \delta_0 \nabla_{q_i}\widehat{\mathcal{G}}_t(q^{(j)}_t)\Big)$$

converge from any $q^{(0)}_t \in \mathcal{Q}_t$ to the unique global maximizer $q^\star_t$ of F_t, with

$$\|q^{(j)}_t - q^\star_t\|_2 \le \left(\frac{\delta_0 L_t}{2}\right)^j \|q^{(0)}_t - q^\star_t\|_2, \qquad j \ge 1.$$

Proof. Fix t and suppress its subscript. Write $G = \widehat{\mathcal{G}}_t$, $\delta = \delta_0$, $L = L_t$, and

$$D(q\|r) = \sum_{i\in\mathcal{M}_t} D_{\mathrm{KL}}(q_i\|r_i), \qquad F(q) = G(q) - \frac{1}{\delta}D(q\|p).$$

All logarithms are natural, and the Euclidean norm is taken over the concatenated probability vectors. For $s = \mathrm{softmax}(z)$, its Jacobian is

$$J(z) = \mathrm{diag}(s) - ss^\top.$$

Since $J(z)$ is symmetric and

$$\sum_\ell |J_{k\ell}(z)| = 2s_k(1 - s_k) \le \frac{1}{2},$$

we have

$$\|J(z)\|_2 \le \sqrt{\|J(z)\|_1 \|J(z)\|_\infty} \le \frac{1}{2}.$$

Integrating along the line segment between any a, b yields

$$\|\operatorname{softmax}(a) - \operatorname{softmax}(b)\|_2 \le \frac{1}{2}\|a - b\|_2.$$

Define

$$[T(q)]_i = \operatorname{softmax}(\log p_i + \delta \nabla_{q_i} G(q)).$$

Applying the preceding bound to each position and using the Lipschitz continuity of ∇G, we obtain

$$\begin{aligned}\|T(q) - T(r)\|_2^2 &\le \frac{\delta^2}{4} \sum_i \|\nabla_{q_i} G(q) - \nabla_{q_i} G(r)\|_2^2 \\ &\le \frac{\delta^2 L^2}{4}\|q - r\|_2^2.\end{aligned}$$

Thus T is a contraction with factor $\rho = \delta L/2 < 1$. Since $\mathcal{Q}$ is closed in a finite-dimensional Euclidean space and $T(\mathcal{Q}) \subseteq \mathcal{Q}$, the Banach fixed-point theorem gives a unique fixed point $q^\star$ and

$$\|q^{(j)} - q^\star\|_2 \le \rho^j \|q^{(0)} - q^\star\|_2.$$

Full support of p ensures that every component of $q^\star = T(q^\star)$ is strictly positive.

We further prove the Global optimality of the fixed point. Let $g^\star = \nabla G(q^\star)$. The fixed-point relation implies

$$\log \frac{q_i^\star(v)}{p_i(v)} = \delta [g_i^\star]_v - \log Z_i,$$

where

$$Z_i = \sum_{u \in \mathcal{V}} p_i(u) \exp(\delta [g_i^\star]_u).$$

For any $q \in \mathcal{Q}$, we therefore obtain

$$\begin{aligned}&D(q\|p) - D(q^\star\|p) - D(q\|q^\star) \\ &= \sum_{i,v} \big(q_i(v) - q_i^\star(v)\big) \log \frac{q_i^\star(v)}{p_i(v)} \\ &= \delta \langle g^\star, q - q^\star \rangle,\end{aligned}$$

where the normalization terms vanish because $\sum_v (q_i(v) - q_i^\star(v)) = 0$. Consequently,

$$\begin{aligned}F(q) - F(q^\star) &= G(q) - G(q^\star) - \langle g^\star, q - q^\star \rangle - \frac{1}{\delta} D(q\|q^\star) \\ &\le \frac{L}{2}\|q - q^\star\|_2^2 - \frac{1}{\delta} D(q\|q^\star).\end{aligned}$$

The inequality follows from the Lipschitz continuity of ∇G.

For probability vectors a, b, Pinsker's inequality gives

$$D_{\mathrm{KL}}(a\|b) \ge \frac{1}{2}\|a - b\|_1^2.$$

Moreover, $d = a - b$ has zero sum. If its positive and negative parts each have total mass A, then

$$\|d\|_2^2 \le 2A^2 = \frac{1}{2}\|d\|_1^2.$$

Hence

$$D_{\mathrm{KL}}(a\|b) \ge \|a - b\|_2^2,$$

and summing over positions yields

$$D(q\|q^\star) \ge \|q - q^\star\|_2^2.$$

It follows that

$$F(q) - F(q^\star) \le -\left(\frac{1}{\delta} - \frac{L}{2}\right)\|q - q^\star\|_2^2.$$

Because $\delta L < 2$, the right-hand side is strictly negative for every $q \ne q^\star$. Thus $q^\star$ is the unique global maximizer of F. □

B Additional Experiments

We provide additional analyses to complement the main experiments and further examine the behavior of BEFMark. We first provide implementation and experimental details in Sec. B.1. Sec. B.2 studies the trade-off between watermark recovery and generation quality across models, message lengths, and generation lengths. Sec. B.3 examines distributions of bit recovery accuracy across samples and provides extended EMR results. Sec. B.4 analyzes the dynamics of bitwise evidence feedback during denoising at both the sample and dataset levels. Sec. B.5 evaluates the effects of the major design choices in BEFMark, including diffusion sampling, voting-layer configuration, adaptive bit weighting, update scale and KL calibration, fixed-point iterations, and context selection. Sec. B.6 concludes with an evaluation of end-to-end generation time and peak GPU memory usage.

B.1 Implementation and Experimental Details

We evaluate all methods on LLaDA-8B-Instruct and Dream-v0-Instruct-7B using ELI5, FiQA, and C4, with message lengths of 8, 16, and 32 bits and maximum generation lengths of 300, 500, and 800 tokens. Actual outputs may be shorter when generation terminates early. We use ELI5 and FiQA directly as question-answering datasets without additional preprocessing. For C4, we randomly sample long passages and truncate them into prefixes for text continuation. Because UMR requires constructing a large cache matrix for each key, we pre-sample four keys for each message length, evaluate each key on the complete test datasets, and average the resulting metrics. For the remaining methods, which do not require key-specific precomputation, we pre-generate an independent random watermark message for each test sample. For PredMPAC and PredBiMark, we additionally apply the context-prediction strategy of DMark (Wu et al., 2025) while preserving their original embedding rules. Unless otherwise stated, BEFMark uses the shared-scale KL-regularized update with $\delta_0 = 4$, one successive linearization step $J = 1$, score temperature $\tau = 1$, $N_{\text{layer}} = 12$ voting layers, and context offsets $\mathcal{S} = \{-1\}$. Following prior work, all methods use a model sampling temperature of 0.5 on LLaDA-8B-Instruct and 0.2 on Dream-v0-Instruct-7B. All methods use block-wise sampling, denoising tokens block by block with a block length of 25.

B.2 Recovery–Quality Trade-off

A practical watermarking method should improve message recovery without substantially degrading generation quality. Although Table 1 reports Acc and log-PPL, examining the two metrics separately does not directly reveal this trade-off. We therefore jointly visualize Acc and log-PPL across models, message lengths, and generation lengths in Fig. 4, where points closer to the upper-left corner indicate a more favorable balance between recovery accuracy and generation quality.

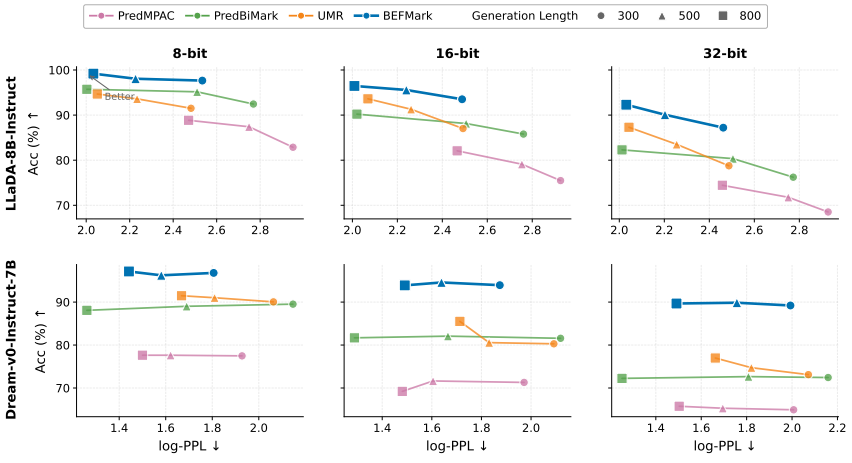


Figure 4: Recovery–quality trade-offs for 8-, 16-, and 32-bit messages on LLaDA-8B-Instruct and Dream-v0-Instruct-7B. Marker shapes denote generation lengths of 300, 500, and 800 tokens. Results are averaged over ELI5, FiQA, and C4. Upper left is better, and lines are visual guides only.

For both LLaDA-8B-Instruct and Dream-v0-Instruct-7B, BEFMark consistently achieves the highest recovery accuracy for 8-, 16-, and 32-bit messages while maintaining competitive log-PPL. Its advantage becomes more pronounced as the message length increases, where the recovery accuracy of competing methods declines more sharply. The trajectories also show that longer generations generally move the operating points toward the upper-left, while BEFMark maintains its recovery advantage across generation lengths. These results show that BEFMark's recovery gains are achieved without a substantial degradation in generation quality.

B.3 Message-Level Recovery Analysis

This section provides a more detailed view of message recovery beyond the average Acc reported in the main experiments. We examine both the distribution of bit recovery accuracy across samples and exact-message recovery rates. The analysis covers 8-, 16-, and 32-bit messages at generation lengths of 300, 500, and 800 tokens using LLaDA-8B-Instruct.

Bit Recovery Accuracy Distributions. To provide a more comprehensive view of message recovery reliability, we analyze the distribution of bit recovery accuracy (Acc) across samples in addition to the average Acc reported in the main experiments. Using LLaDA-8B-Instruct, we evaluate message lengths of 8, 16, and 32 bits at generation lengths of 300, 500, and 800 tokens. For a B-bit message, a sample with k correctly recovered bits has $\mathrm{Acc} = k/B$, where $k \in \{0, \ldots, B\}$. Thus, Acc takes only $B + 1$ discrete values. For each method and each combination of message and generation lengths, we pool 200 samples from each of ELI5, FiQA, and C4 and report the empirical frequency at each Acc level across the resulting 600 samples. The corresponding distributions for the three message lengths are shown in Figs. 5, 6, and 7, respectively.

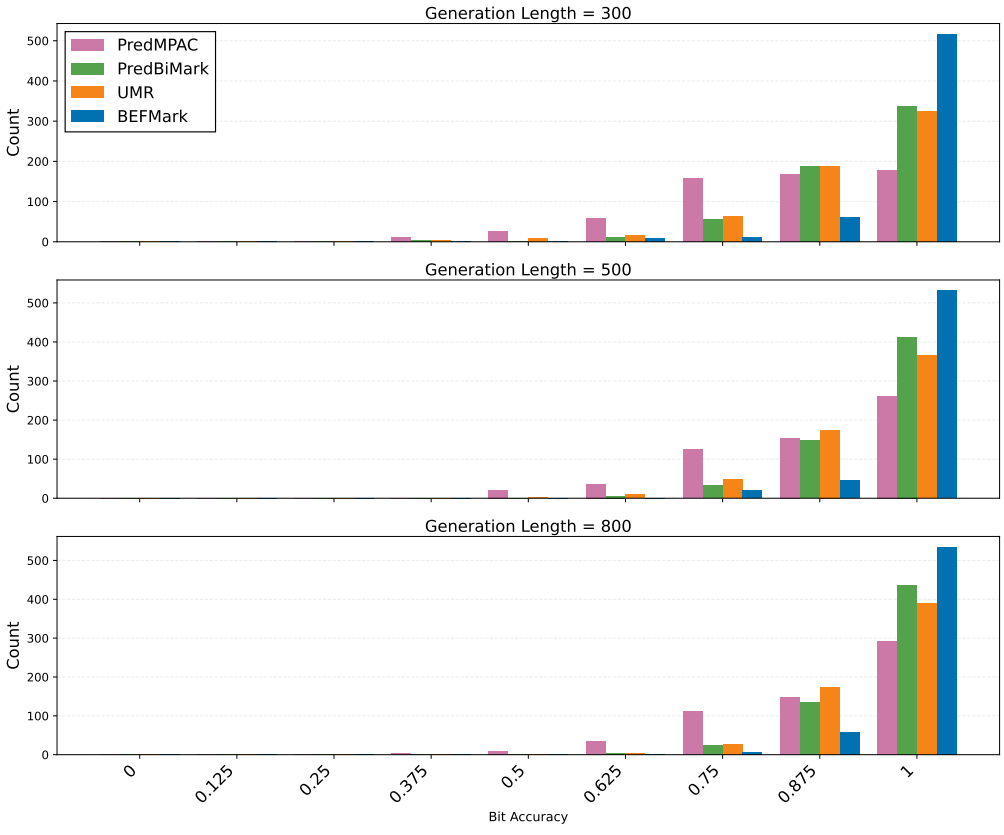


Figure 5: Empirical distributions of bit recovery accuracy (Acc) across samples for 8-bit messages on LLaDA-8B-Instruct at generation lengths of 300, 500, and 800 tokens.

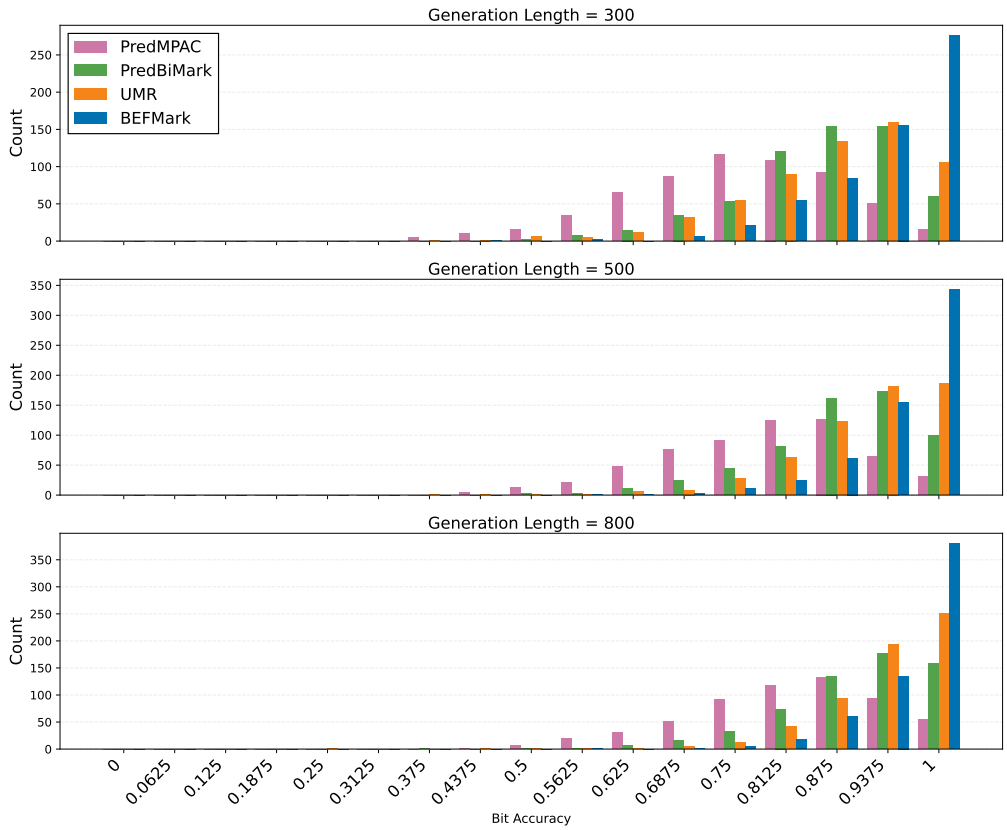


Figure 6: Empirical distributions of bit recovery accuracy (Acc) across samples for 16-bit messages on LLaDA-8B-Instruct at generation lengths of 300, 500, and 800 tokens.

Across all message and generation lengths, our method consistently shifts the empirical Acc distributions toward higher values and yields a substantially larger fraction of samples at high Acc levels than the competing methods. In particular, it yields more samples with $\mathrm{Acc} = 1$, corresponding to exact-message recovery, and fewer samples with partial message recovery. These results further support our design of preferentially strengthening weakly supported bits: by preventing a small number of weak bits from becoming bottlenecks, our method improves not only average Acc but also the reliability of recovering the complete embedded message.

Exact-Message Recovery. The main text reports EMR for 500-token generations (Fig. 3). To assess whether complete-message recovery remains consistent across generation lengths, Table 4 extends the comparison to 300-, 500-, and 800-token generations for all three message lengths. Following the definition in the main text, EMR is computed over samples and counts a recovery as successful only when all bits of the embedded message are decoded correctly.

Table 4: Exact-message recovery (EMR, %) on LLaDA-8B-Instruct for message lengths of 8, 16, and 32 bits at generation lengths of 300, 500, and 800 tokens. Samples are pooled across ELI5, FiQA, and C4. Higher values are better.

	8-bit			16-bit			32-bit		
Method	300	500	800	300	500	800	300	500	800
PredMPAC	29.67	43.67	48.83	2.67	5.17	9.00	0.00	0.00	0.00
PredBiMark	56.17	68.83	72.50	10.00	16.67	26.33	0.00	0.17	0.00
UMR	53.83	60.92	65.17	17.58	31.00	41.83	0.25	1.17	3.75
BEFMark	**86.17**	**92.67**	**94.83**	**46.00**	**57.17**	**63.33**	**7.33**	**14.33**	**20.67**

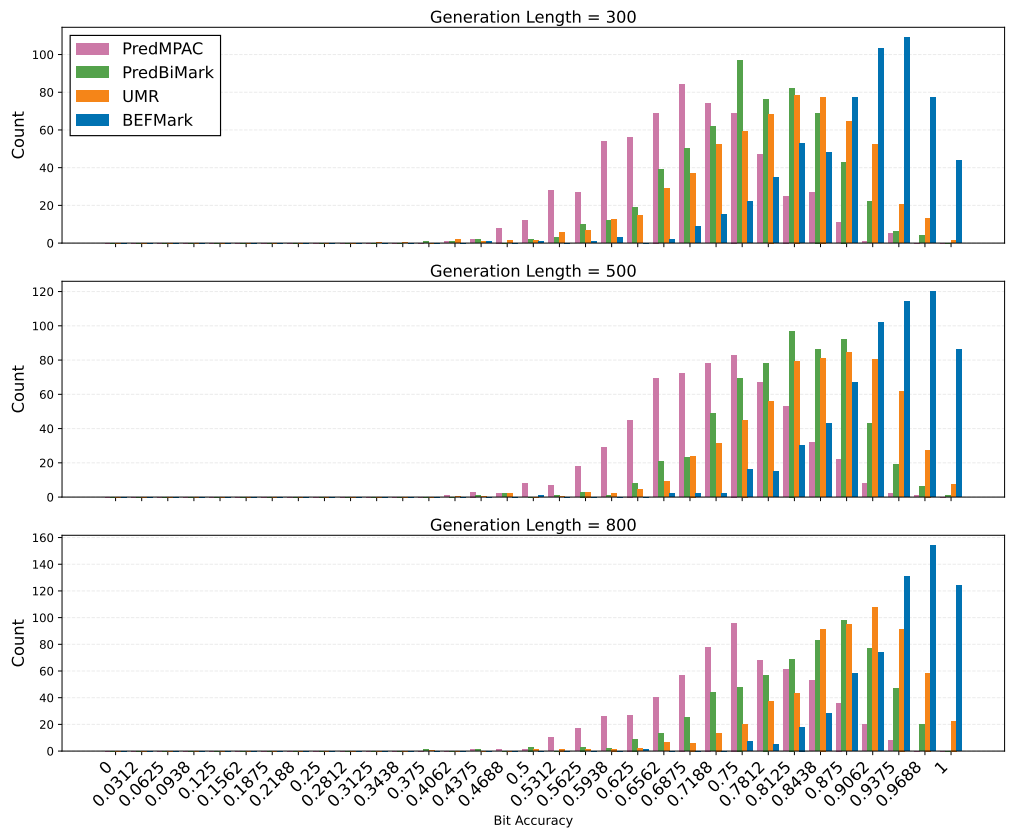


Figure 7: Empirical distributions of bit recovery accuracy (Acc) across samples for 32-bit messages on LLaDA-8B-Instruct at generation lengths of 300, 500, and 800 tokens.

B.4 Bitwise Evidence Dynamics during Denoising

To characterize how BEFMark distributes watermark evidence across multiple message bits during denoising, we track the normalized evidence $z_{t,b}$ of each bit, the aggregate watermark score $R(\mathbf{m}_t, \mathbf{n}_t)$, and the intermediate message recovery accuracy (Acc). The analysis uses LLaDA-8B-Instruct with 16-bit messages and a generation length of 300 tokens. As a controlled reference, we use *BEFMark (uniform weighting)*, which replaces the adaptive bit weights with equal weights while leaving the remaining components unchanged. Paired runs use the same prompt, embedded message, and sampling seed. This comparison distinguishes whether the feedback improves the weakly supported bits that limit joint recovery, rather than merely increasing the average evidence across bits. We first use a representative generation to reveal the sample-specific evidence dynamics and then examine whether the same pattern persists on average across 40 samples, with 20 samples each from ELI5, FiQA.

The normalized evidence $z_{t,b}$ has the same sign as the evidence margin $m_{t,b}$: $z_{t,b} > 0$ supports the embedded bit, $z_{t,b} < 0$ supports the opposite value, and $z_{t,b} = 0$ defines the decoding boundary. Once a bit lies above this boundary, increasing its positive margin does not change its decoded value. Thus, a larger mean evidence or a larger value at every bit is not itself the objective; successful multi-bit recovery instead requires sufficient positive evidence across all bits. We therefore inspect the individual bit trajectories to identify weak bits and use the cross-bit mean and variance, watermark score, and Acc to summarize how evidence imbalance affects recovery. Together, these views reveal whether adaptive weighting strengthens the weakly supported bits that limit joint recovery rather than allocating additional evidence primarily to already strong bits.

Sample-Level Dynamics. We first examine a representative paired example from ELI5 to characterize how watermark evidence evolves within a single denoising trajectory. Fig. 8 tracks the

normalized evidence of each of the 16 message bits, while Fig. 9 summarizes the same generation using the cross-bit mean and variance of $z_{t,b}$, the watermark score, and the intermediate Acc.

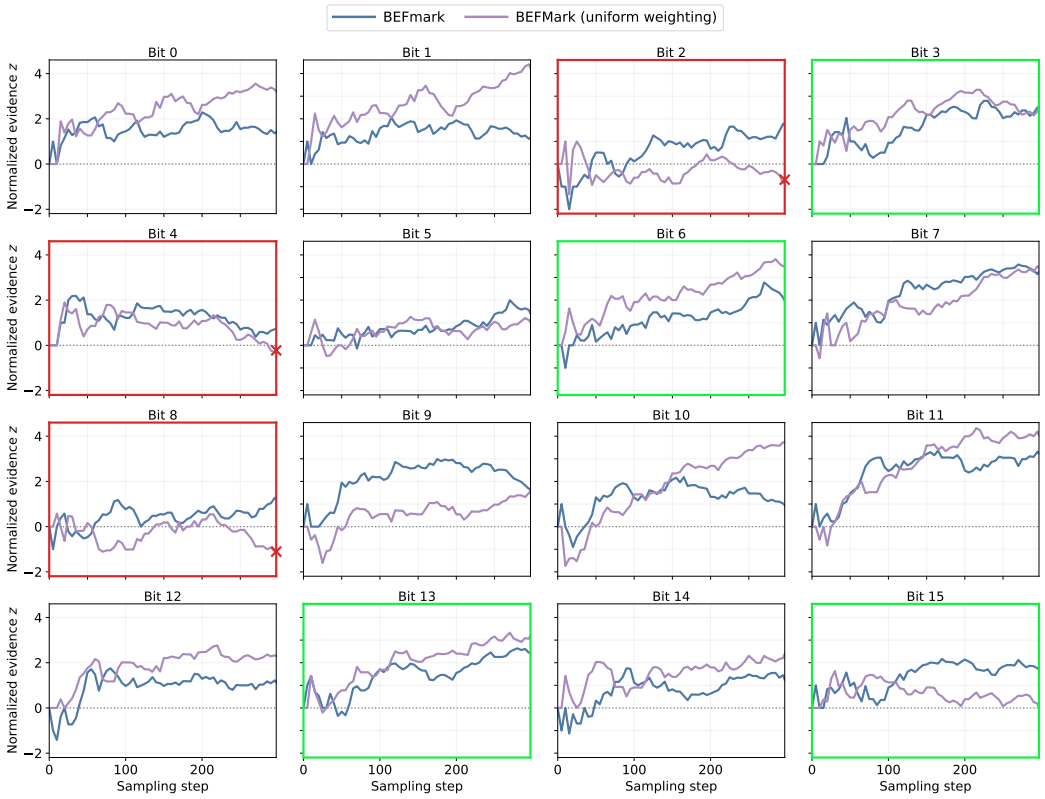


Figure 8: Normalized-evidence trajectories of the 16 message bits for a representative paired ELI5 generation using LLaDA-8B-Instruct with a 16-bit message and a generation length of 300 tokens. The dotted line at $z_{t,b} = 0$ denotes the decoding boundary. Red boxes and crosses mark bits incorrectly decoded at the final step under uniform weighting, while green boxes highlight selected trajectories used to illustrate the effect of adaptive feedback.

At the end of denoising, BEFMark maintains positive normalized evidence for all 16 bits and correctly recovers the complete message. In contrast, uniform weighting leaves Bits 2, 4, and 8 below the decoding boundary, causing three bit errors, as highlighted by the red boxes in Fig. 8. The green-boxed trajectories further illustrate how adaptive feedback acts on weakened evidence during denoising. For Bits 3, 6, 13, and 15, the evidence under BEFMark is initially weak or moves toward the decoding boundary at intermediate steps. As their support weakens, subsequent updates reverse the downward trend or move the evidence farther from the boundary, so that all four bits finish with positive margins. This behavior is consistent with adaptive weighting assigning greater importance to bits with weaker current evidence. For the remaining bits, both configurations generally preserve the correct sign, and uniform weighting sometimes produces larger margins, as for Bits 0 and 1. These larger margins, however, cannot compensate for the final errors of Bits 2, 4, and 8. The example therefore shows that BEFMark does not seek to maximize every bit trajectory; instead, it preserves sufficient evidence for already-correct bits while preventing weak bits from becoming final decoding errors.

Fig. 9 clarifies why the average evidence alone is insufficient to characterize multi-bit recovery. Although uniform weighting attains a comparable and eventually larger mean z, its cross-bit variance grows substantially during the later stages of denoising. BEFMark maintains a much smaller cross-bit variance, indicating a more balanced evidence profile rather than uniformly larger evidence at every bit. Although some correctly decoded bits under BEFMark have lower z values than under uniform weighting, this does not affect their decoded values. By emphasizing the weakest-supported bits, BEFMark achieves a watermark score that rises more steadily and remains higher, while Acc

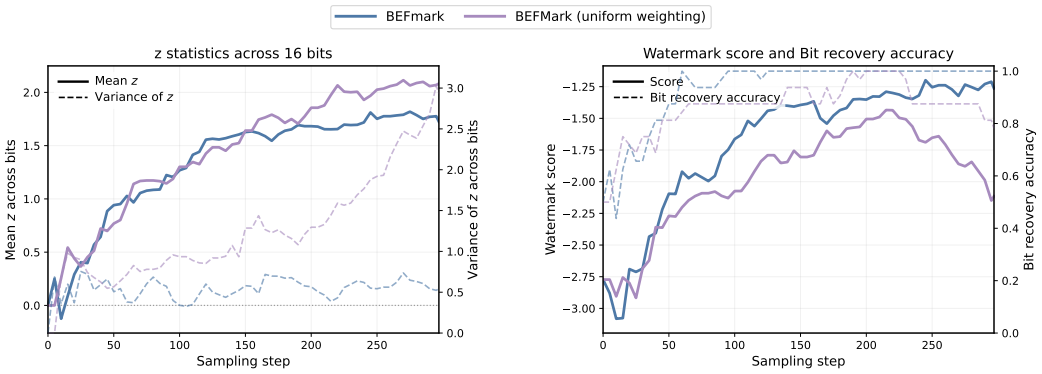


Figure 9: Cross-bit evidence statistics and bit recovery accuracy metrics for a representative paired ELI5 generation using LLaDA-8B-Instruct with a 16-bit message and a generation length of 300 tokens. Blue and purple curves denote BEFMark and BEFMark (uniform weighting), respectively. Left: the cross-bit mean (solid; left axis) and variance (dashed; right axis) of $z_{t,b}$. Right: the watermark score (solid; left axis) and intermediate Acc (dashed; right axis).

reaches 1.0. Under uniform weighting, the score deteriorates late in denoising and Acc falls to $13/16$, corresponding to the three final errors in Fig. 8.

Aggregate Dynamics across Samples. We next average the evidence dynamics over 40 samples to determine whether the behavior observed in the representative example persists across generations. Fig. 10 reports the average trajectory of each bit position, while Fig. 11 summarizes the corresponding cross-bit mean and variance, watermark score, and Acc.

Fig. 10 shows that BEFMark maintains positive average evidence at every bit position by the end of denoising. Uniform weighting produces larger margins at several already strong positions, but its average evidence for Bit 8 becomes increasingly negative, while Bits 2, 4, 13, and 14 also receive weaker support than under BEFMark. These results extend the sample-level observation: adaptive weighting does not maximize every bit trajectory, but shifts feedback toward positions with weaker evidence. Because the trajectories are averaged across samples, a negative value at a particular bit index indicates an aggregate weakness rather than an error shared by every sample.

Fig. 11 provides further evidence for this redistribution. The two configurations achieve similar mean normalized evidence, with uniform weighting ending slightly higher. Their cross-bit variances, however, differ substantially: BEFMark maintains a low variance throughout denoising, whereas the variance under uniform weighting increases steadily. Thus, the improvement does not arise from increasing the overall evidence magnitude, but from distributing evidence more evenly across bits and avoiding a weak lower tail. Consistent with this pattern, BEFMark achieves a steadily increasing watermark score and an average Acc approaching 1.0, while the score under uniform weighting deteriorates in the later stages and its Acc remains lower. Together, the two figures show that adaptive weighting maintains more balanced positive evidence across bit positions, leading to more reliable multi-bit recovery. The individual and averaged results show that adaptive weighting strengthens weakly supported bits while maintaining positive evidence for the remaining bits.

B.5 Ablation Studies

The following ablation and sensitivity studies examine BEFMark's adaptive feedback, optimization settings, evidence construction, and sampling strategy. We first isolate the contribution of adaptive bit weighting (Sec. B.5.1) and examine the update scale δ_0, the number of successive linearization iterations J, and the score temperature τ (Sections B.5.2, B.5.3, and B.5.4). We then study the number of voting layers, layer-wise bit assignment, and context selection (Sections B.5.5, B.5.6, and B.5.7). Finally, we replace block-wise sampling with random-order sampling to assess whether BEFMark depends on the default diffusion sampling strategy (Sec. B.5.8).

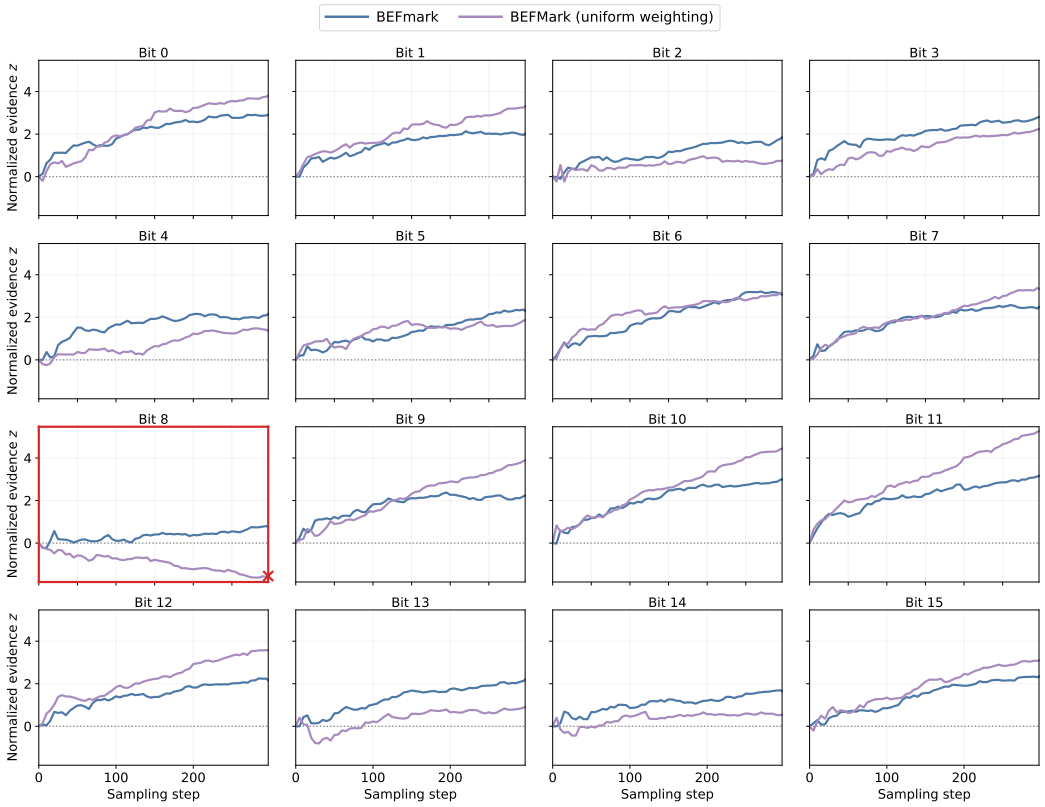


Figure 10: Per-bit normalized-evidence trajectories averaged over 40 paired generations using LLaDA-8B-Instruct with 16-bit messages and a generation length of 300 tokens. Blue and purple curves denote BEFMark and BEFMark (uniform weighting), respectively, and the dotted line at $z_{t,b} = 0$ denotes the decoding boundary. The red box and cross mark Bit 8, whose average final evidence remains negative under uniform weighting.

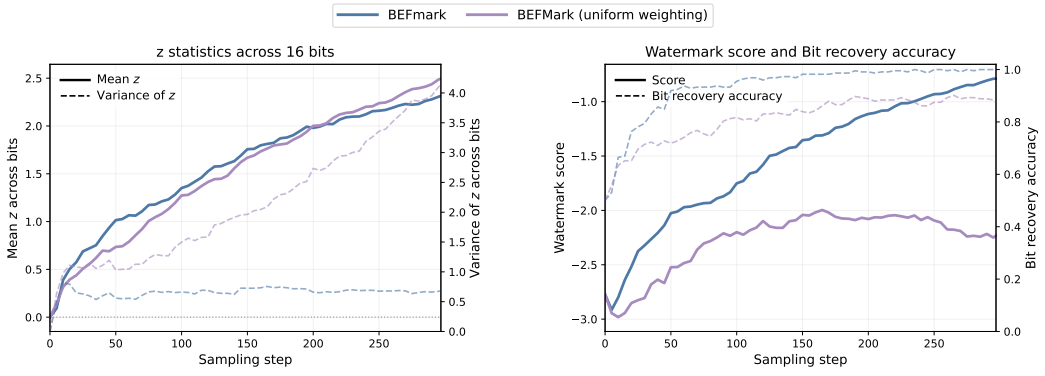


Figure 11: Cross-bit evidence statistics and recovery metrics averaged over 40 paired generations using LLaDA-8B-Instruct with 16-bit messages and a generation length of 300 tokens. Blue and purple curves denote BEFMark and BEFMark (uniform weighting), respectively. Left: the cross-bit mean (solid; left axis) and variance (dashed; right axis) of $z_{t,b}$. Right: the watermark score (solid; left axis) and intermediate Acc (dashed; right axis).

B.5.1 Ablation of Adaptive Bit Weighting

Adaptive bit weighting directs the feedback signal toward bits with weak current evidence, which are most likely to prevent complete-message recovery. To isolate its contribution, we replace the adaptive weights with equal weights while leaving all other components unchanged; we refer to this configuration as BEFMark (uniform weighting). We compare the two configurations on LLaDA-8B-Instruct using 16-bit messages and generation lengths of 300 and 500 tokens, with results averaged over ELI5, FiQA, and C4. Table 5 reports the average Acc and log-PPL. Because average Acc does not indicate how often every bit in a message is decoded correctly, Fig. 12 further presents the distribution of bit recovery accuracy across samples.

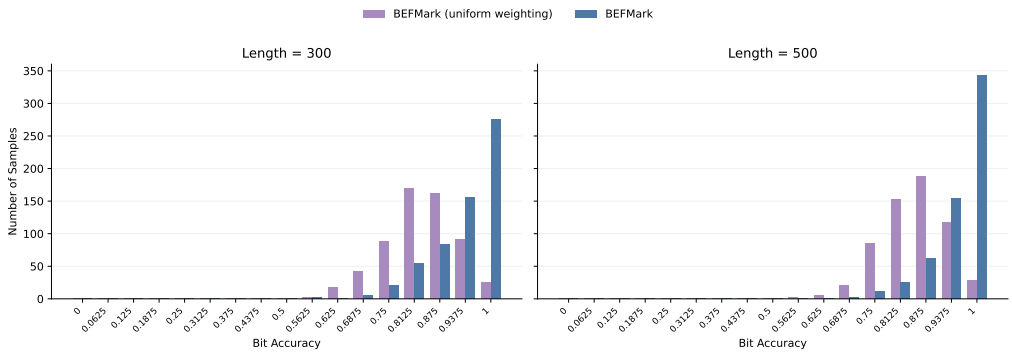


Figure 12: Distributions of per-sample bit recovery accuracy with and without adaptive bit weighting on LLaDA-8B-Instruct using 16-bit messages and generation lengths of 300 and 500 tokens. For each generation length, results are averaged over ELI5, FiQA, and C4.

Table 5 shows that adaptive weighting increases the average Acc from 83.11 to 93.51 at 300 tokens and from 84.93 to 95.57 at 500 tokens, yielding gains of 10.40 and 10.64 percentage points, respectively. The per-sample distributions in Fig. 12 show that these gains also translate into substantially more samples with $\mathrm{Acc} = 1.0$, indicating more frequent recovery of the complete message. Generation quality remains stable: log-PPL changes only from 2.473 to 2.488 at 300 tokens and decreases from 2.257 to 2.240 at 500 tokens. Adaptive bit weighting therefore improves both average bit accuracy and complete-message recovery with negligible change in generation quality.

Table 5: Ablation of adaptive bit weighting on LLaDA-8B-Instruct with 16-bit messages. Results are averaged over ELI5, FiQA, and C4.

	300 tokens		500 tokens	
Variant	Acc ↑	log-PPL ↓	Acc ↑	log-PPL ↓
BEFMark (uniform weighting)	83.11	**2.473**	84.93	2.257
BEFMark	**93.51**	2.488	**95.57**	**2.240**

B.5.2 Analysis of δ_0

Under shared KL regularization, performance depends on the magnitude of δ_0. A small value limits the influence of evidence-guided feedback and may reduce message recovery accuracy, whereas a large value can shift the predictive distribution more aggressively and degrade generation quality. We therefore vary δ_0 around the default value of 4 while holding all other settings constant. Table 6 reports Acc and log-PPL, allowing us to assess whether the default value provides a stable recovery–quality balance rather than relying on a narrowly tuned value.

Table 6 shows a recovery–quality trade-off at both generation lengths. Increasing δ_0 from 2 to 5 monotonically improves Acc but also increases log-PPL. Compared with $\delta_0 = 3$, the default value improves Acc by 5.20 and 4.53 percentage points at 300 and 500 tokens, respectively. Increasing δ_0 further to 5 yields smaller gains of 2.88 and 2.29 points while continuing to degrade log-PPL. Thus, $\delta_0 = 4$ provides a balanced default rather than optimizing recovery or generation quality alone.

Table 6: Sensitivity of Acc and log-PPL to the shared regularization scale δ_0 on LLaDA-8B-Instruct with 16-bit messages. Results are averaged over ELI5, FiQA, and C4.

δ_0	300 tokens		500 tokens	
	Acc ↑	log-PPL ↓	Acc ↑	log-PPL ↓
2.0	80.69	2.102	83.43	1.876
3.0	88.31	2.281	91.04	2.061
4.0 (default)	93.51	2.488	95.57	2.240
5.0	96.39	2.717	97.86	2.466

B.5.3 Ablation of Successive Linearization Iterations

The KL-regularized update in Sec.3.3 applies Eq. (12) repeatedly, relinearizing the coupled evidence objective at the current iterate. The choice of J determines how extensively the watermarked distribution is refined before each denoising transition, while each additional iteration incurs extra computation. We therefore vary J to assess the sensitivity of watermark recovery and generation quality to the number of successive linearized updates.

Table 7: Effect of the number of successive linearization iterations J on Acc, log-PPL, and generation time for 16-bit messages on LLaDA-8B-Instruct. Results are averaged over ELI5, FiQA, and C4; time is reported per sample.

Iterations J	300 tokens			500 tokens		
	Acc ↑	log-PPL ↓	Time (s) ↓	Acc ↑	log-PPL ↓	Time (s) ↓
1 (default)	93.51	2.488	18.18	95.57	2.240	42.50
3	93.78	2.423	20.38	95.89	2.191	46.20
5	94.07	2.429	22.02	96.03	2.184	48.74

As shown in Table 7, increasing the number of successive linearization iterations generally improves both message recovery and generation quality, at the cost of additional runtime. Increasing J from 1 to 5 improves Acc by 0.56 and 0.46 percentage points for 300- and 500-token generations, respectively, while reducing log-PPL by 0.059 and 0.056. Acc increases monotonically at both generation lengths, and log-PPL decreases monotonically at 500 tokens. For 300-token generations, log-PPL reaches its minimum of 2.423 at $J = 3$ and rises only marginally to 2.429 at $J = 5$, still below the value at $J = 1$. Meanwhile, generation time increases by 3.84 and 6.24 seconds, respectively. Overall, additional iterations yield only modest gains in recovery and generation quality while introducing a steadily increasing computational overhead. We therefore use $J = 1$ by default.

B.5.4 Ablation of Sensitivity to the Score Temperature τ

The temperature τ in Eq. (3) controls both the smoothness of the soft-min watermark score and the concentration of the resulting bit importance weights. A smaller τ places greater emphasis on the weakest-supported bits, whereas a larger τ distributes the feedback more evenly across bits. We therefore evaluate five values of τ while holding all other settings fixed. Table 8 reports the resulting Acc and log-PPL at generation lengths of 300 and 500 tokens, allowing us to assess the sensitivity of BEFMark to weak-bit prioritization and the recovery–quality trade-off.

Table 8 shows that BEFMark is relatively insensitive to τ, with Acc varying by less than one percentage point across all evaluated values at both generation lengths. A moderate temperature of $\tau = 0.5$ yields the highest Acc, whereas increasing τ generally improves log-PPL by distributing the feedback more evenly across bits. In contrast, the smallest temperature does not provide additional recovery gains and produces the highest log-PPL, suggesting that overly concentrated bit weights are unnecessary. The default $\tau = 1.0$ therefore provides a stable recovery–quality trade-off rather than optimizing either metric in isolation.

Table 8: Effect of the score temperature τ on Acc and log-PPL for 16-bit messages with LLaDA-8B-Instruct. Results are averaged over ELI5, FiQA, and C4.

Score temperature τ	300 tokens		500 tokens	
	Acc ↑	log-PPL ↓	Acc ↑	log-PPL ↓
0.2	93.69	2.578	96.02	2.378
0.5	94.17	2.523	96.28	2.318
0.8	94.03	2.525	95.66	2.284
1.0 (default)	93.51	2.488	95.57	2.240
1.2	93.23	2.459	96.11	2.231

B.5.5 Ablation of the Number of Voting Layers

The multi-layer voting mechanism introduced in Sec.3.1 allows each valid token to contribute evidence to multiple message bits. The layer count N_{layer} controls how many bit-specific votes are associated with each token. Increasing N_{layer} broadens coverage across the message, but also requires a single predictive distribution to accommodate more watermark objectives. We therefore vary N_{layer} from 1 to 20 to examine the resulting recovery–quality trade-off across message and generation lengths. Experiments use LLaDA-8B-Instruct with 8- and 16-bit messages and generation lengths of 300 and 500 tokens. To limit the computational cost of this sweep, we use ELI5 and FiQA and pool samples from the two datasets. All other settings remain unchanged. Fig. 13 reports Acc and log-PPL as functions of N_{layer}.

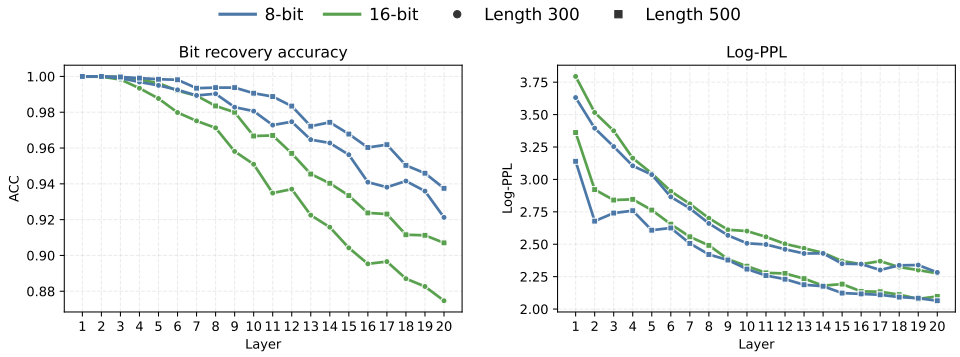


Figure 13: Effect of N_{layer} on Acc and log-PPL for 8- and 16-bit messages at generation lengths of 300 and 500 tokens. Samples from ELI5 and FiQA are pooled.

Fig. 13 shows a consistent effect of N_{layer} across 8- and 16-bit messages and both generation lengths. Increasing N_{layer} generally lowers Acc while improving log-PPL, with the largest quality gains occurring at smaller layer counts. The agreement across message lengths confirms that this recovery–quality trade-off is not specific to a particular message length. Although more layers add votes, the layer-wise votes at a position are determined by the same generated token and therefore do not provide independent evidence. Their induced token preferences may also overlap or conflict, so broader coverage does not necessarily strengthen the effective evidence for each bit. Overall, $N_{\text{layer}} = 12$ provides a balanced trade-off across the evaluated settings; larger values offer diminishing quality gains while Acc continues to decline.

B.5.6 Ablation of Layer-wise Bit Assignment

The message-bit position assignment in MCEA (Sec.3.1) selects the first-layer position $b_{t,i,1}$ from the context hash using $\text{PRF}_k^{\text{pos}}$, after which the subsequent layers advance cyclically through the bit positions. This design is intended to distribute evidence stably and evenly across the embedded message. We ablate it to determine whether structured layer-wise assignment improves recovery and, more specifically, whether its benefit comes from strict cyclic balancing or from allowing different layers to cover different bits.

Using the context hash $h_{t,i}$ and $\mathrm{PRF}_k^{\mathrm{pos}}$ defined in Sec.3.1, we compare the following three definitions of $b_{t,i,r}$ for $r = 1, \ldots, N_{\mathrm{layer}}$:

$$b_{t,i,r} = \begin{cases} 1 + \big((\mathrm{PRF}_k^{\mathrm{pos}}(h_{t,i}) + r - 2) \bmod B\big), & \text{cyclic}, \\ \mathrm{PRF}_k^{\mathrm{pos}}(h_{t,i}, r), & \text{random}, \\ \mathrm{PRF}_k^{\mathrm{pos}}(h_{t,i}), & \text{shared}. \end{cases} \tag{23}$$

The default *cyclic* scheme advances from $b_{t,i,1}$ without repetition until all bit positions have been traversed. The *random* scheme includes the layer index in the keyed mapping and independently assigns each layer to a pseudorandom bit position. The *shared* scheme omits the layer index and assigns all layers associated with the same context hash to $b_{t,i,1}$. All three schemes are deterministic given the context and key and can therefore be reproduced during recovery. With $B = 16$ and $N_{\mathrm{layer}} = 12$, cyclic assignment covers exactly 12 distinct bit positions per token, shared assignment covers one, and the coverage of random assignment is determined by its layer-specific mappings.

We evaluate the three schemes on LLaDA-8B-Instruct using 16-bit messages and generation lengths of 300 and 500 tokens. Table 9 reports Acc and log-PPL averaged over ELI5, FiQA, and C4.

Table 9: Effect of layer-wise bit assignment on LLaDA-8B-Instruct with 16-bit messages and $N_{\mathrm{layer}} = 12$. Results are averaged over ELI5, FiQA, and C4

	300 tokens		500 tokens	
Assignment	Acc ↑	log-PPL ↓	Acc ↑	log-PPL ↓
Shared	86.64	2.120	91.41	1.911
Random	93.30	2.499	95.93	2.258
Cyclic (default)	93.51	2.488	95.57	2.240

The cyclic and random schemes perform similarly: their Acc differs by only 0.21 and 0.36 percentage points at 300 and 500 tokens, respectively, while their log-PPL differs by at most 0.018. This suggests that strict cyclic ordering is not the primary source of the average recovery performance. In contrast, shared assignment reduces Acc by 6.87 and 4.52 percentage points relative to the best-performing scheme, although it yields lower log-PPL. These results reveal a recovery–quality trade-off and show that distributing layer-wise votes across multiple message bits is important for reliable recovery. Cyclic assignment provides this cross-bit coverage deterministically while maintaining performance comparable to random assignment.

Evidence Dynamics. Because aggregate Acc does not reveal how the assignment rule shapes the within-sample distribution of evidence, Fig. 14 presents a matched case study that tracks the per-bit vote counts and normalized evidence throughout diffusion sampling. The three schemes use the same prompt, embedded message, key, and sampling seed, allowing their evidence trajectories to be compared under matched conditions.

Fig. 14 shows that cyclic and random assignment yield positive final evidence for all 16 bits and therefore recover the complete message in this example. Shared assignment instead leaves Bits 1, 2, 4, 5, 8, 9, 13, and 14 below the decoding boundary, as highlighted by the red boxes. Under cyclic assignment, the vote counts increase smoothly and remain relatively close across bits. With $N_{\mathrm{layer}} = 12$, each valid token contributes votes to 12 distinct bit positions, distributing new evidence broadly across the message. Random assignment produces similarly balanced vote-count trajectories, consistent with the small differences between cyclic and random assignment in Table 9.

Shared assignment exhibits both spatial and temporal imbalance. Its vote counts grow in large steps and differ substantially across bits: Bit 13 accumulates over 400 votes, whereas Bits 7, 11, and 15 receive far fewer. Several incorrectly recovered bits also undergo long intervals without receiving new votes. Because all layers at a position target one hash-selected bit, evidence is concentrated on a single bit at each update rather than distributed across the message. The trajectory of Bit 13 further shows that a large vote count alone does not guarantee a positive normalized margin. Consequently, eight bits remain below the decoding boundary at the end of sampling, while additional votes assigned elsewhere cannot compensate for these errors.

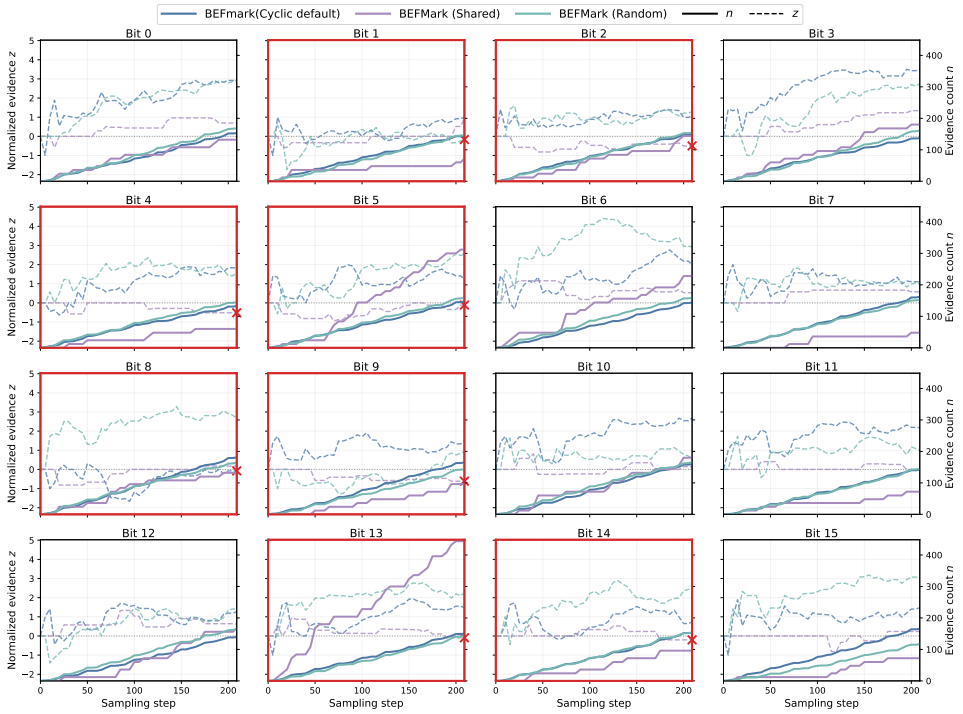


Figure 14: Matched single-sample evolution of normalized evidence $z_{t,b}$ (dashed lines, left axis) and vote count $n_{t,b}$ (solid lines, right axis) during diffusion sampling. Blue, purple, and teal denote cyclic, shared, and random assignment, respectively. Red boxes and crosses mark the bits decoded incorrectly under shared assignment.

Overall, the aggregate and trajectory results identify broad cross-bit coverage, rather than the exact cyclic order, as the key factor for reliable recovery. Cyclic assignment guarantees such coverage, random assignment achieves comparable message recovery accuracy and generation quality, and shared assignment trades substantially lower Acc for lower log-PPL because its evidence is concentrated unevenly across bits and sampling steps.

B.5.7 Ablation of Local Context Selection

The local context set $\mathcal{C}_{t,i}$ in Sec. 3.1 is constructed using the offset set $\mathcal{S}$, which specifies the relative positions included around position i. This choice affects both the information used to construct the context hash and the availability of bitwise evidence during denoising: incorporating more contextual tokens provides a richer basis for context-dependent assignment, but the evidence at position i becomes available only after all selected context tokens have been recovered. We compare $\mathcal{S} = \{-1\}$, which uses the immediately preceding token, with $\mathcal{S} = \{-1, 1\}$, which uses the adjacent tokens on both sides. The experiments use LLaDA-8B-Instruct with 16-bit messages and generation lengths of 300 and 500 tokens. Table 10 reports Acc, log-PPL, and generation time to evaluate the effects on watermark recovery, generation quality, and computational cost.

Table 10 shows that the two-sided context improves log-PPL but substantially reduces Acc and increases generation time. With $\mathcal{S} = \{-1, 1\}$, a position enters Ω_t only after both adjacent tokens have been recovered, delaying the availability of its observed evidence. Before then, its expected future evidence must be computed over the uncertainty of two context tokens, which distributes the estimate across more possible context-hash outcomes and weakens the resulting bit-specific feedback. This delayed and less concentrated feedback explains the lower recovery accuracy. Modeling the additional context-token distribution also increases the cost of evidence estimation throughout denoising, accounting for the longer generation time. The two-sided context therefore gains generation quality at the cost of both watermark recovery and computational efficiency, supporting $\mathcal{S} = \{-1\}$ as the default context selection.

Table 10: Effect of context selection on watermark recovery, generation quality, and generation time on LLaDA-8B-Instruct using 16-bit messages and generation lengths of 300 and 500 tokens. Results at each generation length are averaged over ELI5, FiQA, and C4. Time denotes the average generation time per sample in seconds. Higher Acc, lower log-PPL, and lower time are better.

	300 tokens			500 tokens		
Context offsets	Acc ↑	log-PPL ↓	Time (s) ↓	Acc ↑	log-PPL ↓	Time (s) ↓
$\mathcal{S} = \{1\}$	90.91	2.413	18.54	93.37	2.207	42.94
$\mathcal{S} = \{-1, 1\}$	80.26	**2.099**	18.60	85.23	**1.875**	43.06
$\mathcal{S} = \{-1\}$ (default)	**93.51**	2.488	**18.18**	**95.57**	2.240	**42.50**

B.5.8 Ablation of Diffusion Sampling Strategy

The main experiments use block-wise sampling, which restricts denoising to one block at a time. To evaluate whether BEFMark relies on this sampling strategy, we replace it with full-sequence random-order sampling, where positions are unmasked in a random order across the entire response rather than within predefined blocks. All other experimental settings remain unchanged. Table 11 reports Acc and log-PPL for 16-bit messages at generation lengths of 300 and 500 tokens.

Table 11: Watermark recovery and generation quality under full-sequence random-order sampling on LLaDA-8B-Instruct using 16-bit messages. Results at each generation length are averaged over ELI5, FiQA, and C4. Higher Acc and lower log-PPL are better.

Method	300 tokens		500 tokens	
	Acc	log-PPL	Acc	log-PPL
PredMPAC	74.17	2.961	76.41	2.701
PredBiMark	79.53	2.748	81.67	**2.286**
UMR	71.45	2.886	73.72	2.737
BEFMark	**86.38**	**2.737**	**86.82**	2.363

Compared with the block-wise results in Table 1, full-sequence random-order sampling reduces Acc across all methods. BEFMark nevertheless achieves the highest Acc at both generation lengths, outperforming the strongest baseline by 6.85 and 5.15 percentage points at 300 and 500 tokens, respectively. It also obtains the best log-PPL at 300 tokens and the second-best at 500 tokens. These results show that BEFMark maintains its recovery advantage and competitive generation quality when denoising positions are selected across the entire sequence rather than within blocks.

B.6 Computational Efficiency

We compare the average end-to-end generation time and peak GPU memory of BEFMark with those of unwatermarked generation and existing multi-bit watermarking methods. Experiments are conducted on LLaDA-8B-Instruct using 16-bit messages and 300-token outputs, with a batch size of 1 and 300 denoising steps on a single NVIDIA GPU with 48 GB of memory. Generation time includes denoising and watermark computation but excludes model loading. Table 12 shows that BEFMark has a runtime comparable to PredMPAC and lower than PredBiMark and UMR. It uses only 0.43 GB more memory than unwatermarked generation and 14.83 GB less than UMR, indicating modest computational overhead.

Table 12: Computational efficiency on LLaDA-8B-Instruct using 16-bit messages and 300-token outputs. w/o WM: without watermark. Lower values are better.

Metric	w/o WM	PredMPAC	PredBiMark	UMR	BEFMark
Time (s) ↓	16.32	18.14	19.35	19.18	18.18
Peak memory (GB) ↓	17.03	17.25	17.22	32.29	17.46

C Examples of Watermarked Generation and Message Recovery

We present one qualitative example from each of FiQA, ELI5, and C4, comparing unwatermarked and BEFMark-generated text and reporting the corresponding message-recovery results.

Example 1 *FIQA | 16-bit | 300 tokens*

Prompt: You are a helpful assistant, please answer the following question with financial knowledge within 300 words: Renting from self during out of area remodel project - deductible?

No Watermark: Renting a home from yourself during an out-of-area remodel project can be tax-deductible in most cases, subject to certain conditions. 1. **Calculate the rental cost**: The rent you pay needs to be the cost you would have paid if you were renting elsewhere at the same time. This includes any mortgage interest, insurance, and property taxes. 2. **Verify the purpose**: Make sure the rental is specifically for a project related to your business, such as a home renovation, construction, or home improvement project. 3. **Document the rental expenses**: Keep track of all rental expenses, including rent, mortgage interest, property taxes, and any other property-related expenses.4. **Time limits**: Typically, the home needs to be occupied for more than 180 days and no more than 365 days to be deductible for a home renovation or construction. 5. **Interest in the business**: Ensure the interest in the business related to the home renovation is significant, and the rental is related to the business. 6. **Tax limits**: Be aware of the annual limitations for deductions, which is 10,000 per year.Remember, tax laws can vary by state and individual circumstances, so it's always a good idea to consult with a tax professional or review your state's tax laws to ensure you're eligible to deduct your rental expenses during your out-of-area remodel project.

BEFMark: Renting your own home for a remodel or renovation project can be a gray area but taxable. Here are some guidelines: 1. **Personal use**: If you're renting for personal use, like a remodel or renovation, the rent is typically tax-deductible. 2. **Temporary use property**: If you're temporarily out of your home for a remodel or renovation, it may be classified as a temporary use property, allowing you to deduct the rental expenses. 3. **Home office**: If you're temporarily using your home as a home office for work, the project is considered a home renovation, and the rent may be eligible for deduction as a business expense. 4. **Business expenses**: If you're renting for a business purpose, such as a remodel or renovation, the rent may be tax-deductible as a business expense, but you may need to consider the temporality of your income and expenses. 5. **Temporary use and expenses**: If you're using your own home for work out of area, such as a temporary project or renovation, the rent may be tax-deductible as a business expense. In summary, the tax deductibility of renting your own home for a remodel or renovation project depends on various factors, such as the purpose of the rental, the type of use, and the temporality of your income and expenses. It's essential to understand the guidelines and consult with a tax professional to match your situation to tax regulations.

Message Recovery

Embedded message:	`0110000001100010`
Extracted message:	`0110000001100010`
Message recovery accuracy (Acc):	100%
No Watermark Log-PPL:	1.788
BEFMark Log-PPL:	1.607

Vote counts (0/1):

[76, 72]	[62, 87]	[61, 75]	[67, 55]	[63, 58]	[65, 55]	[70, 60]	[72, 64]
[73, 63]	[66, 73]	[65, 81]	[75, 71]	[78, 72]	[80, 67]	[65, 78]	[87, 64]

Example 2 *ELI5 | 16-bit | 300 tokens*

Prompt: You are a helpful assistant, please answer the following question within 300 words: The US ' beef with its cops I just want a little light on what 's happening with u guys like, why are your cops brutalizing and sometimes even killing the citizens they were sworn to protect? Please explain like I'm five.assistant

No Watermark: Alright, imagine you have a job where you're supposed to take care of your toys. But some of your toys are getting lost or messed up, and you're not sure how to fix it. Laws are like your toys, but for people, and police officers are like grown-ups whose job is to help protect them and make sure everyone is safe. But sometimes, these grown-ups might still make mistakes or accidentally do things they shouldn't. When that happens, it can cause problems, which is why it's called a "beef" with the police. It also happens in the United States, just like any other place in the world. There have been some cases where police officers have made mistakes and hurt people they were supposed to protect. These mistakes can be very sad and cause people to feel very scared, hurt, or even angry. The people in the United States have been trying to fix these problems and make sure that police officers act fairly and justly. They have created programs for police officers to learn the laws and the rules, and how to help people when they need it. It's important to remember that it's okay sometimes to make mistakes, and when everyone works together to make things right, we can make a better and safer world for everyone. So, the "beef" with police in the United States is a reminder that sometimes we all need to learn more and work together to be safer and more fair for everyone.

BEFMark: So, what's happening with the US about police brutality and killings are complicated. Simply put, police brutality and police shootings in the US have increased in recent years, and it's mostly due to a combination of issues like racism, racial inequality, and a flawed police training. The US is a racist country, which means that people of color have been treated unfairly and unjustly for a long time. This racism has been reflected in many police officer's behavior, especially when it comes to black people. Research has started to show that black people are more likely to be unfairly targeted and killed by police officers. Furthermore, the US police system is thought to be the only way to protect the community. Over the years, many departments have failed to teach their officers about police reform, empathy, and community engagement. This has resulted in a lack of trust between the police force and the communities they serve. Let's remember that police brutality and killings are not just affecting black people, but they affect every community. Also, many citizens are afraid to seek police help because they can't trust the police officer or they're scared about what will happen. In conclusion, the US police system is flawed due to a history of systemic racism, racial inequality, and inadequate police training, which has resulted in police brutality and killings. It is crucial to recognize and acknowledge this issue, and to work towards police reform to build trust between the police force and the communities they serve.

Message Recovery

Embedded message:	`0000001110001001`
Extracted message:	`0000001110001001`
Message recovery accuracy (Acc):	100%
No Watermark PPL:	1.969
BEFMark PPL:	2.134

Vote counts (0/1):

[102, 82]	[96, 78]	[95, 82]	[92, 76]	[97, 77]	[106, 89]	[85, 108]	[91, 100]
[90, 101]	[104, 83]	[89, 89]	[97, 79]	[70, 107]	[95, 75]	[102, 76]	[80, 111]

Example 3 *C4 | 16-bit | 300 tokens*

Prompt: You are a helpful assistant, please continue writing the following content within 300 words: Gene Blevins/ReutersA tornado in Kansas last week. A new poll suggests the public feels that global warming is real. Sunday will be the 42nd annual Earth Day celebration. Here are 10 ideas for commemorating the day, observed each year on April 22, or exploring issues related to the environment in the classroom or home on any day of the year. When was the first Earth Day (PDF), and what major themes did it address? Are these themes still relevant today? How has Earth Day changed since 1970? In the past five or 10

No Watermark: years, what events have contributed to the growth of the Earth Day movement?

1. On Earth Day, bring your reusable cups, coffee cups, plates, shopping bags, and utensils to local stores and restaurants. 2. Have a garden party in your backyard, or help to plant a tree in a local park. 3. Go on a nature walk to a local park, forest, or beach. Look for plants, animals, birds, and other wildlife. 4. Participate in a local clean-up event. 5. Build a birdhouse, create a park, or plant trees in a local park. 6. Research and explore wind farms, solar energy, and other green energy sources in your community. 7. Help to conserve energy and reduce your carbon footprint. 8. Learn more about the air, land, and water that are important for our health. 9. Start a fundraiser to support an environmental organization in your community. 10. Find out more about Earth Day and its environmental history

BEFMark: years, what were some of the most memorable Earth Day events? The first Earth Day, April 22, 1970, addressed issues such as reducing pollution, saving energy through the use of public transport, and the preservation of natural areas. These themes are still relevant today as environmental issues remain a significant part of societal problems. However, Earth Day has evolved over the years to include a wider range of environmental concerns. For example, in 1974, Earth Day highlighted the impact of pesticides on the environment, and by 1990, the focus had shifted to include issues such as pollution and climate change. In the early 2000s, Earth Day also started to address issues of sustainability and environmental justice.

One of the most memorable Earth Day events of the past few years was the 2011 Global Run, where thousands participated in a run to raise funds for environmental causes. Another event was the 2014 Climate March, which saw people around the world march in protest to demand climate action. Additionally, the 2019 Earth Day Festival in the US became a massive tourist draw, attracting millions of people from around the world.

Overall, Earth Day has remained an important platform for raising awareness about environmental issues and promoting action on environmental problems. As the planet faces increasing environmental challenges, it remains crucial for us to take action to protect the environment for generations to come.

Message Recovery

Embedded message:	`0010010001101111`
Extracted message:	`0010010001101111`
Message recovery accuracy (Acc):	100%
No Watermark Log-PPL:	1.755
BEFMark Log-PPL:	1.513

Vote counts (0/1):

[92, 84]	[91, 86]	[74, 103]	[93, 81]	[100, 82]	[83, 113]	[110, 92]	[100, 99]
[116, 90]	[99, 107]	[90, 111]	[104, 81]	[74, 109]	[81, 87]	[80, 87]	[86, 103]