Majority Bit-Aware Watermarking for Large Language Models

Jiahao Xu[1,2*] **Rui Hu**[1] **Zikai Zhang**[1] **Olivera Kotevska**[2]
[1]University of Nevada, Reno
[2]Oak Ridge National Laboratory
{jiahaox, ruihu, zikaiz}@unr.edu kotevskao@ornl.gov

Abstract

The growing deployment of Large Language Models (LLMs) has raised concerns about their misuse in generating harmful or deceptive content. To address this issue, watermarking methods have been proposed to embed identifiable multi-bit messages into generated text for misuse tracing. However, existing methods often suffer from a fundamental trade-off between text quality and decoding accuracy. In particular, they have to restrict the size of the preferred token set (i.e., green list) during encoding to maintain a detectable watermark signal for decoding, which inevitably degrades generation quality. To improve this trade-off, we propose a novel message encoding paradigm called *majority bit-aware encoding*, which relaxes the watermark signal strength from the green list size. This strategy allows for a strong watermark signal to be preserved in generated texts even when using a large green list. We introduce two instantiations of this paradigm: MajorMark and MajorMark$^+$, where the latter is specifically optimized for long messages. Extensive experiments on state-of-the-art LLMs demonstrate that our methods achieve higher decoding accuracy and superior text quality compared to prior baselines. The code is at `https://github.com/JiiahaoXU/MajorMark`.

1 Introduction

The widespread deployment of Large Language Models (LLMs) has fundamentally reshaped numerous domains, from education and customer support to sophisticated content generation, owing to their capacity for highly fluent and contextually relevant text. However, this powerful capability simultaneously introduces considerable security and ethical challenges. Malicious users can leverage LLMs to produce harmful content, such as fake news, phishing emails, and fraudulent reviews [31, 45]. To mitigate these risks, developing watermarking techniques is essential. By embedding an imperceptible yet verifiable signal within generated text, watermarking enables effective origin tracing and attribution of misuse [23, 39, 28]. A typical text watermarking method comprises an *encoder* that embeds a specific message into the generated content and a corresponding *decoder* designed to extract this message. This work focuses on a common watermarking scenario [43, 31, 14], where a cloud-based LLM-as-a-Service provider adopts ***multi-bit watermarking*** to uniquely mark each output for a user prompt. The decoder can then recover the embedded message from unverified text.

The first LLM watermarking scheme, KGW [16], was designed to encode a *zero-bit message*–a yes/no signal indicating that an LLM generated the text. During encoding, at each token generation step, KGW pseudo-randomly selects a *green list* $\mathcal{G} \subseteq \mathcal{V}$ from the vocabulary and strategically boosts the logits of the tokens in $\mathcal{G}$ by a watermarking bias δ. This increases their probability of being sampled as the next token. A larger δ produces a stronger watermark but often results in noticeable degradation in the quality of the generated text. During decoding, KGW reconstructs the green list for

*This work was done during Jiahao's internship at the Oak Ridge National Laboratory.

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

each token within the target text and computes the proportion of tokens that fall into their respective green lists. If this proportion is statistically significant (measured via a z-test), the target text is verified as watermarked and, consequently, generated by an LLM.

However, *zero-bit watermarking* methods are inherently limited in our setting, since service providers may need to embed more complex information, such as user identifiers, model versions, or other metadata. To address this limitation, recent studies have explored ***multi-bit watermarking***, which is designed to embed a binary message $\mathbf{m}$ of length $b \geq 2$. Existing methods can be broadly divided into two categories: perturbation-based methods [43, 20, 37, 9, 31] and distortion-free methods [19, 14, 8]. In this work, we primarily focus on perturbation-based methods, as they typically achieve higher decoding accuracy. However, such improvements often come at the cost of reduced quality in the generated watermarked text. Specifically, two key watermarking parameters affect the quality of watermarked text in perturbation-based methods: the green list ratio, defined as $\gamma = |\mathcal{G}|/|\mathcal{V}|$, and the watermarking bias δ used to boost the logits of green-list tokens. In this work, we study the impact of the green list ratio γ while fixing the watermarking bias. Intuitively, a smaller γ can degrade the quality of watermarked text, as a narrower green list provides fewer contextually suitable token choices during generation. In contrast, under a fixed watermarking bias, a smaller γ can improve decoding accuracy, since the decoding process in existing methods relies on *counting the frequency of tokens* that appear in the corresponding green lists. Therefore, existing methods exhibit an empirical trade-off between *watermarked text quality and message decoding accuracy*, as shown in Section 3.

To improve this trade-off, we propose a novel ***majority bit-aware encoding strategy*** that relaxes the watermark signal strength from the size of the green list, allowing a large green list to still carry a strong watermark signal. Specifically, we partition the vocabulary into b disjoint shards, where each shard corresponds to a specific bit in the message. During encoding, the green list is constructed by uniting all shards that correspond to the message's majority bit. This strategy inherently guarantees a green list ratio of $\gamma \geq 0.5$, which is critical for preserving text quality. Crucially, the message can now be represented via the specific selection of green shards. Consequently, the decoding objective shifts from analyzing token frequency to inferring which specific shards were boosted during encoding.

Based on the proposed majority bit-aware encoding strategy, we develop MajorMark, a *Major*ity bit-aware water*Mark*ing method. MajorMark directly leverages majority bit-aware encoding, yielding an expected green list ratio $\mathbb{E}_{\mathbf{m}}[\gamma] \approx 0.5 + 1/\sqrt{2\pi b}$. It is equipped with a clustering-based decoding strategy that operates on shard-level occurrence statistics, enabling fine-grained message decoding. Furthermore, we propose MajorMark$^+$ to better preserve text quality for long messages. MajorMark$^+$ partitions the message into r blocks and applies the majority bit-aware encoding to each block independently, yielding a larger expected green list ratio, $\mathbb{E}_{\mathbf{m}}[\gamma] \approx 0.5 + 1/\sqrt{2\pi(b/r)}$. During decoding, MajorMark$^+$ deterministically reconstructs each message block by inferring the majority bit and the number of green shards. Our key contributions are summarized as follows:

- We propose a novel *majority bit-aware encoding strategy* for multi-bit watermarking of LLM-generated content. By encoding the message into the selection of green shards, this paradigm effectively relaxes watermark detectability from the green list size, thereby improving the trade-off between text quality and decoding accuracy.
- We propose two multi-bit watermarking methods, termed MajorMark and MajorMark$^+$, both of which adopt the majority bit-aware encoding strategy to preserve high text quality. We further design corresponding decoders for each method to ensure accurate message decoding.
- We conduct extensive experiments with state-of-the-art open-source LLMs on different downstream tasks. The results show that our methods consistently achieve higher decoding accuracy and text quality than existing multi-bit watermarking approaches.

2 Preliminaries and Existing Works

Large Language Models. An LLM f is an autoregressive model that performs the next-token prediction task. Specifically, given an input prompt $\mathbf{x}_p$, the model generates a sequence of tokens over a predefined vocabulary $\mathcal{V}$, which contains all permissible tokens for constructing natural language responses. At the t-th ($t = 1, 2, \ldots, T$) generation step, the model takes as input the prompt $\mathbf{x}_p$ and the previously generated tokens $\mathbf{x}_{:t-1} = \{x_1, x_2, \ldots, x_{t-1}\}$, and outputs a logits vector $\ell^t = (\ldots, f(v \mid \mathbf{x}_p, \mathbf{x}_{:t-1}), \ldots)$, where $f(v \mid \mathbf{x}_p, \mathbf{x}_{:t-1})$ represents the logit for a specific token

$v \in \mathcal{V}$. These logits are then passed through a $\texttt{softmax}(\cdot)$ to produce a probability distribution over $\mathcal{V}$, from which the current token x_t is sampled. This process is repeated autoregressively until a termination condition (e.g., maximum length) is met, yielding an output sequence $\mathbf{x}_g$ of length T.

Zero-Bit Watermarking in LLMs. The first zero-bit watermarking method for LLMs, KGW [16], embeds a detectable signal into generated text without encoding an explicit message. At each generation step t, the LLM produces a logit vector, after which a pseudo-random seed is derived by hashing a secret key k with the previous token x_{t-1}. This seed deterministically permutes and partitions the vocabulary $\mathcal{V}$ into a green list $\mathcal{G}$ and a red list $\mathcal{R}$. A positive watermarking bias δ is then added to the logits of tokens in $\mathcal{G}$, increasing their sampling probability after the softmax operation. While a larger δ strengthens the watermark signal, it also more severely perturbs the token distribution, which can degrade text quality. This procedure is applied at every token generation step, producing a watermarked sequence. To detect a watermark in an unverified text, the decoder reconstructs the green list for each token and performs a z-test on the frequency of tokens falling into their corresponding green lists. A statistically significant excess over the expected frequency indicates the presence of a watermark. Subsequent works have extended this framework to improve decoding accuracy and text quality [17, 19, 38, 2, 10, 22, 30, 3, 27], as well as to enhance robustness against post-generation text editing [21, 22, 13, 4].

Multi-Bit Watermarking in LLMs. Multi-bit watermarking aims to encode and decode a b-bit binary message $\mathbf{m} \in \{0,1\}^b$ (we assume that b is even in this work) into and from the generated text. Multi-bit messages can carry richer information, such as the user's identity, the model version, or a timestamp. To embed $\mathbf{m}$ into LLM-generated text, CTWL [37] and CycleShift [9] leverage $\mathbf{m}$ to calculate the pseudo-random seed during encoding. However, their decoding processes require brute-force enumeration over all 2^b possible message values to find the best candidate, rendering them computationally infeasible for large b. A more efficient method, MPAC [43], divides the message $\mathbf{m}$ into r blocks $\{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_r\}$, where each block is d bits long. During generation, one message block is encoded per token generation step. Specifically, for each block $\mathbf{m}_i$, the encoder partitions the vocabulary into 2^d disjoint shards and selects the $[\mathbf{m}_i]_{10}$-th shard as the green list for that step, where $[\cdot]_{10}$ denotes the decimal (base-10) representation of the binary block. For decoding, MPAC identifies the shard with the highest token count at each step, interprets its index as the base-10 value of the corresponding message block, and converts it back to a d-bit binary string. The full message is then reconstructed by concatenating all recovered blocks. However, MPAC fixes $\gamma = 1/2^d$ (e.g., $\gamma = 0.25$ for $d = 2$ in their default setting), which significantly degrades the utility of the generated text. To address this, RSBH [31] improves MPAC by computing hash seeds using $[\mathbf{m}_i]_{10}$, choosing a larger green list size of $\gamma = 0.5$. This enhancement improves utility but increases decoding complexity by a factor of 2^d. We summarize the decoding complexity of representative multi-bit methods in Table 1.

Table 1: Comparison of representative multi-bit LLM watermarking methods. $\mathbb{E}_{\mathbf{m}}[\gamma]$ denotes the expected green list ratio over all possible messages. $\times|\mathbf{x}|$ indicates that the decoding process requires how many enumerations over all tokens in the input text $\mathbf{x}$. "Eff.?" reflects whether the decoding procedure is computationally efficient.

Method	Encoding $\mathbb{E}_{\mathbf{m}}[\gamma]$	Decoding $\times\lvert\mathbf{x}\rvert$	Eff.?
CTWL [37]†	–	2^b	✗
CycleShift [9]	$= 0.25$	2^b	✗
DepthW [20]	$= 0.50$	2^b	✗
RSBH [31]‡	$= 0.50$	2^d	✓
MPAC [43]	$= 0.25$	1	✓
MajorMark	$\approx 0.50 + 1/\sqrt{2\pi b}$	2	✓
MajorMark$^{+\sharp}$	$\approx 0.50 + 1/\sqrt{2\pi(b/r)}$	$b/r - 1$	✓

† CTWL determines γ dynamically by a proxy LLM.
‡ In RSBH, d denotes the number of bits in a message block.
♯ In MajorMark$^+$, r with $r < b$ denotes the number of message blocks.

3 Problem Formulation and Motivation

Problem Formulation. Given an LLM f, a user prompt $\mathbf{x}_p$, and a b-bit binary message $\mathbf{m} \in \{0,1\}^b$, the goal of multi-bit watermarking is to embed $\mathbf{m}$ into the generated text to produce a watermarked output $\mathbf{x}'_g$. This process is performed by an encoder function $\texttt{Enc}(\cdot)$: $\mathbf{x}'_g = \texttt{Enc}(f, \mathbf{x}_p, \mathbf{m})$, where $\mathbf{x}'_g$ should remain fluent and semantically consistent with the unwatermarked output $\mathbf{x}_g$. The embedded message can be extracted using a decoder $\texttt{Dec}(\cdot)$ to obtain $\mathbf{m}' = \texttt{Dec}(\mathbf{x}'_g)$. Formally, the objective of multi-bit watermarking can be formulated as the following constrained optimization problem:

$$\begin{aligned} \max \quad & \Pr\Big[\mathbf{m}' = \mathbf{m} \;\Big|\; \mathbf{m}' = \texttt{Dec}(\texttt{Enc}(f, \mathbf{x}_p, \mathbf{m}))\Big] \\ \text{s.t.} \quad & \texttt{Quality}(\mathbf{x}'_g \mid \mathbf{x}_p) \geq \tau, \end{aligned}$$

where $\texttt{Quality}(\cdot)$ quantifies the quality of generated text, and τ is a minimum quality threshold (often chosen to be close to the quality of $\mathbf{x}_g$). In practice, conditional perplexity $\texttt{PPL}(\cdot)$ is widely used as a quality metric. This formulation captures the key goal of watermarking: to maximize the probability of exact message decoding while ensuring negligible degradation in text quality.

Motivation. Existing perturbation-based methods typically necessitate careful selection of the green list ratio γ to balance the trade-off between the quality of watermarked text and the accuracy of decoding. Specifically, due to the shift invariance of the $\texttt{softmax}(\cdot)$ function, adding the bias to a larger portion of logits (i.e., using a larger γ) better preserves the original token probability distribution, thereby maintaining the quality of the generated sequence. However, since these methods rely on counting the frequency of generated tokens that fall within the recovered green list $\mathcal{G}$, a larger γ weakens the watermark signal, reducing the distinctiveness of the perturbation and making decoding less accurate. Conversely, using a smaller γ introduces stronger distortions to the output distribution, thereby improving the ability to decode the embedded message. Yet this comes at the cost of generation quality: high-probability (i.e., desirable) tokens may fall outside $\mathcal{G}$ and be suppressed due to the bias, while low-probability tokens within $\mathcal{G}$ may be unintentionally amplified and incorrectly sampled. This imbalance can significantly degrade the quality of $\mathbf{x}'_g$.

To empirically illustrate the role of γ in shaping this trade-off, we conduct experiments using RSBH [31] with LLaMA-2-7B [36], varying the green list ratio γ from 0.3 to 1.0 under message length $b = 64$. Note that $\gamma = 1.0$ corresponds to the extreme case where watermarking is ineffective. We report both the bit accuracy (i.e., the proportion of bits correctly decoded from the watermarked text) and the perplexity of the watermarked text in Figure 1, under different bias settings. As expected, increasing γ consistently improves text quality, as reflected by lower perplexity across all bias levels, but at the cost of reduced bit accuracy. These results empirically demonstrate the trade-off between text utility and decoding accuracy influenced by γ. We summarize the green list ratios γ used in prior methods in Table 1. All of them set $\gamma \leq 0.5$, by following previous works [37, 9, 20, 31] or empirical tuning [43].

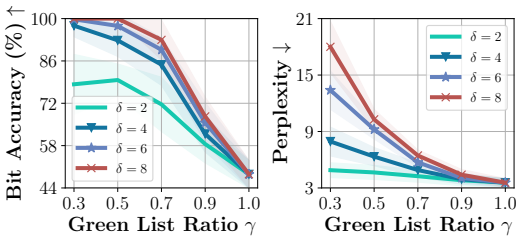


Figure 1: Impact of green list ratio γ on the utility of watermarked text (right) and the decoding accuracy (left).

4 Proposed Methods

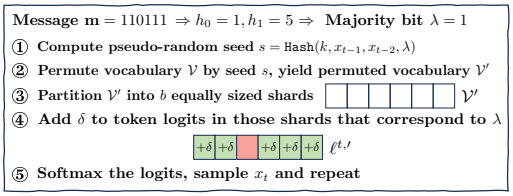
4.1 Majority Bit-Aware Encoding

We first formally define the majority bit λ of a multi-bit message in Definition 1.

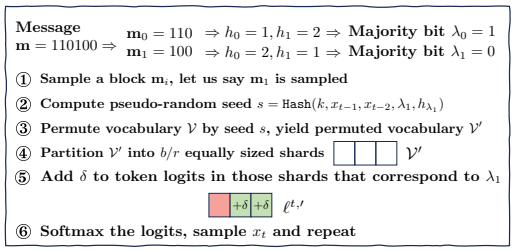
Definition 1 (Majority Bit)**.** *Given a binary message $\mathbf{m} \in \{0,1\}^b$ with $b \geq 2$, let h_0 and h_1 denote the number of occurrences of bit 0 and 1 in $\mathbf{m}$, respectively. The majority bit $\lambda \in \{0,1\}$ of $\mathbf{m}$ is defined as $\lambda = 1$ if $h_1 \geq h_0$, and $\lambda = 0$ otherwise.*

For example, given a message $\mathbf{m} = 110111 \in \{0,1\}^6$, the majority bit is $\lambda = 1$ because $h_1 = 5$ and $h_0 = 1$. In cases where the occurrences are equal (i.e., $h_0 = h_1 = b/2$), we define the majority bit as $\lambda = 1$ by default. Crucially, the majority bit for any message appears at least $\lceil b/2 \rceil$ times, which provides a stable heuristic for message encoding.

We present the overview of the majority bit-aware encoding process at the t-th token generation step of LLM in Figure 2a. Specifically, prior to the generation process of the LLM f, it first identifies the majority bit λ of the b-bit binary message $\mathbf{m}$ to be embedded. During the generation of the t-th token, after obtaining the logits ℓ^t from f, it performs the following steps: ① Computes a pseudo-random seed s based on a hash function (the design of this hash function is detailed later); ② Applies the seed s to permute the vocabulary $\mathcal{V}$, yielding a permuted vocabulary $\mathcal{V}'$; ③ Evenly partitions $\mathcal{V}'$ into b disjoint shards $[\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_b]$, where each shard $\mathcal{V}_i$ corresponds to the i-th bit of the message $\mathbf{m}$. The green list $\mathcal{G}$ is constructed by taking the union of the shards associated with the majority bit value λ, such that $\mathcal{G} = \cup_{i:m_i=\lambda} \mathcal{V}_i$. ④ Uses the same seed s to permute ℓ^t, yielding permuted logits $\ell^{t,\prime}$ and adds a bias δ to the logits of tokens in $\mathcal{G}$; ⑤ Applies the $\texttt{softmax}(\cdot)$ function to the biased logits, yielding a new probability distribution from which the current token x_t is sampled. Our majority



(a) Majority bit-aware encoding process at generation step t.



(b) Block-wise majority bit-aware encoding process at generation step t.

Figure 2: An overview of the majority bit-aware encoding process of MajorMark and MajorMark$^+$.

bit-aware encoding strategy ensures a minimum green list ratio of $\gamma \geq 0.5$, thereby preserving the utility of watermarked text.

4.2 MajorMark

We propose MajorMark, which directly utilizes majority bit-aware encoding as the encoder and is equipped with a clustering decoder. We first derive the theoretical guarantee for γ of MajorMark.

Theorem 1 (γ of MajorMark)**.** *For any message $\mathbf{m} \in \{0,1\}^{b \geq 2}$, the green list $\mathcal{G}$ constructed via MajorMark satisfies $|\mathcal{G}| \geq 0.5|\mathcal{V}|$, i.e., $\gamma \geq 0.5$. Moreover, for random b-bit binary messages, we have $\mathbb{E}_{\mathbf{m}}[\gamma] \approx 0.5 + 1/\sqrt{2\pi b}$.*

Proof. The detailed proof is in Appendix R. □

Remark 1. *MajorMark achieves a strictly larger expected γ ($\mathbb{E}_{\mathbf{m}}[\gamma] > 0.5$) than existing methods for any finite b, indicating improved text utility. As $b \to \infty$, the expectation $\mathbb{E}_{\mathbf{m}}[\gamma]$ gradually converges to 0.5, making the improvements less significant. Note that we derive the expected γ by assuming each bit is an independent random variable following a $\text{Bernoulli}(0.5)$ distribution. However, there are two extreme cases $\mathbf{m} = \mathbf{0}^b$ and $\mathbf{m} = \mathbf{1}^b$ that will result in $\gamma = 1.0$, thereby rendering the watermarking ineffective. In practice, these two special cases can be explicitly disallowed during encoding, and this has a negligible impact on the expected value of γ when b is large.*

Design of the Hash Function. We revisit the design of the hash function in majority bit-aware encoding. Prior works [16, 43] typically compute the seed s as $\texttt{Hash}(k, x_{t-1})$, where k is a secret key and x_{t-1} is the previous token. However, frequent words (e.g., *the*, *is*, *to*) appear disproportionately in generated text, causing certain seeds, and thus token-to-shard mappings, to repeat [31]. This leads to repetitive green lists $\mathcal{G}$, over-sampling of specific tokens, and degraded text quality. To address this, we follow [20] and extend the hash input to include the second-to-last token x_{t-2}, using $\texttt{Hash}(k, x_{t-1}, x_{t-2})$. This yields more diverse token-to-shard mappings. We further incorporate the majority bit λ, forming $\texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda)$, which increases seed diversity and facilitates recovery of λ during decoding, as detailed below.

Clustering-Based Decoding. The message decoding process takes as input a generated text $\mathbf{x}$ of length T and produces the decoded message $\mathbf{m}'$. Specifically, for each token $x_t \in \mathbf{x}$ with $t = 3, 4, \ldots, T$, MajorMark reconstructs the token-to-shard mapping for x_t used in encoding. We discard the first two tokens x_1 and x_2 from decoding, as computing their token-to-shard mappings requires access to the prompt tokens in $\mathbf{x}_p$, which are unavailable during decoding. To reconstruct the token-to-shard mapping for x_t, we first compute the seed s using $\texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda)$. We then determine the shard to which x_t belongs and increment its corresponding count. After processing all tokens, we obtain a shard-wise token occurrence count that reflects the frequency of tokens belonging to each shard. Based on this occurrence count, we apply a clustering algorithm (e.g., `KMeans` [25] with $K = 2$) to partition the shards into two clusters. The cluster with the higher average token count, denoted by $\mathcal{C}$, is interpreted as corresponding to the majority bit λ, as the logits of tokens in these shards were positively perturbed by δ during encoding. The message bits are then recovered by setting $m_i = \lambda$ for each shard $i \in \mathcal{C}$ and $m_i = 1 - \lambda$ for those not in the cluster (i.e., $i \notin \mathcal{C}$). We provide the theoretical justification of this decoding and analyze decoding errors in Appendix L.

One remaining challenge in MajorMark's decoding process is determining the correct value of λ. Recall that we explicitly include λ as the input of the hash function $\texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda)$. This design allows the recovery of λ. The key intuition is that only the correct seed s can reproduce the token-to-shard mappings used during encoding. Therefore, in the decoding phase, we compute the shard-wise token occurrence counts for each possible value of λ (i.e., 0 or 1). The correct value of λ is determined as the one that produces a more dispersed occurrence distribution (e.g., exhibiting a larger standard deviation), whereas the incorrect value yields a nearly uniform distribution, as illustrated in Figure 3. This approach enables reliable recovery of the embedded message with only a one-time additional enumeration over $\mathbf{x}$. The detailed encoding and decoding algorithms of MajorMark and the implementation of the hash function are presented in Appendix J.

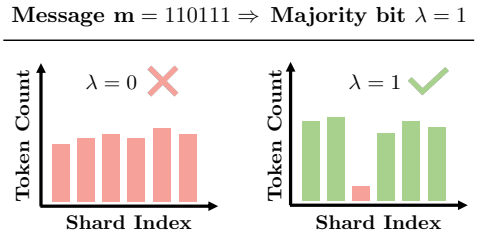


Figure 3: Occurrence distributions for candidate λ (0 or 1). Distribution produced by the correct λ exhibits large dispersion.

4.3 MajorMark$^+$: An Improved Method

We now introduce MajorMark$^+$, an enhanced version of MajorMark that further improves both the quality of watermarked text and decoding accuracy. Specifically, compared with MajorMark, MajorMark$^+$ (1) guarantees a larger expected γ through a *block-wise majority bit-aware encoding* strategy; and (2) enhances decoding accuracy by replacing the boundary-sensitive clustering-based decoding with a *deterministic decoding* method.

Block-Wise Majority Bit-Aware Encoding. As we discussed in Remark 1, when b becomes large, the improvement of γ achieved by MajorMark becomes less significant as γ converges to 0.5. Inspired by previous methods that divide the full message into several blocks [43, 31, 14] to perform watermark encoding, MajorMark$^+$ is equipped with a *block-wise majority bit-aware encoding* method to slow the convergence of γ given a finite b, especially when b is large. We present an overview of MajorMark$^+$'s block-wise majority bit-aware encoding process at generation step t in Figure 2b. Specifically, given any message $\mathbf{m} \in \{0,1\}^b$ with $b \geq 2$, MajorMark$^+$ divides it into r equal-sized blocks $\{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_r\}$, where r divides b. During the t-th generation step of the LLM f, MajorMark$^+$ pseudo-randomly assigns the i-th block $\mathbf{m}_i$ to the current token x_t, where $i = (x_{t-2} + x_{t-1}) \mod r$. As a result, each token contributes to encoding only one message block.

To encode a given block $\mathbf{m}_i$ where $i \in [r]$, MajorMark$^+$ applies the same *majority bit-aware encoding* strategy used in MajorMark: the green list is constructed based on the block-specific majority bit λ_i, and the logits are perturbed accordingly. Importantly, the hash function of MajorMark$^+$ is in the form of $\texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda_i, h_{\lambda_i})$. Specifically, we further incorporate the number of occurrences of the majority bit h_{λ_i} into the calculation of the hash seed s, which enables deterministic decoding in MajorMark$^+$, as discussed later. We now present the formal guarantee on MajorMark$^+$'s expected γ as follows in Theorem 2.

Theorem 2 (γ of MajorMark$^+$)**.** *MajorMark$^+$ preserves the lower bound of the green list ratio as in MajorMark, i.e., $|\mathcal{G}| \geq 0.5|\mathcal{V}|$ and $\gamma \geq 0.5$. Moreover, for random b-bit binary messages, we have $\mathbb{E}_{\mathbf{m}}[\gamma] \approx 0.5 + 1/\sqrt{2\pi(b/r)}$.*

Proof. The detailed proof is in Appendix S. □

Remark 2. *Given any finite b, MajorMark$^+$ always guarantees a larger expected γ than MajorMark for any $r > 1$. This property improves the text quality of MajorMark$^+$. Recall from Remark 1 that MajorMark discards extreme cases where the message is entirely composed of zeros or ones, i.e., $\mathbf{m} = \mathbf{0}^b$ or $\mathbf{m} = \mathbf{1}^b$. MajorMark$^+$ similarly excludes such extreme messages at the block level. These infeasible cases are extremely rare and do not meaningfully affect the practical effectiveness of MajorMark$^+$ when b is large. Specifically, the number of infeasible codes is given by $2^b - (2^{b/r} - 2)^r$. For instance, given a large $b = 32$ and $r = 2$, the total codes are 2^{32}, while the number of infeasible codes is only 262,140, accounting for approximately 0.006% of the code space.*

Deterministic Decoding. When the message length b is large and the watermarking bias δ is not large enough, the resulting shard-wise token occurrence count becomes less dispersed. This weak

signal may degrade the effectiveness of MajorMark's decoding process in specific cases, as clustering algorithms are *unsupervised* and inherently sensitive to data points near *decision boundaries*, failing to accurately separate the shards. To address this limitation, MajorMark$^+$ introduces a deterministic decoding method designed to recover the majority bit λ_i and its occurrences h_{λ_i} in the message for each block with high reliability. Specifically, given the hash function $\texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda_i, h_{\lambda_i})$, MajorMark$^+$ exhaustively enumerates all possible combinations of λ_i and h_{λ_i}. For each combination, it reconstructs the corresponding shard-wise token occurrence count. The correct configuration yields a distinctly dispersed occurrence counts, while incorrect configurations result in near-uniform counts. By identifying the configuration that induces the largest dispersion, MajorMark$^+$ reliably recovers the correct values of λ_i and h_{λ_i}. The message bits corresponding to those shards with top-h_{λ_i} token occurrences are then assigned the recovered bit λ_i, while the remaining bits are assigned $1 - \lambda_i$. The full message is subsequently reconstructed by assembling all blocks. This deterministic approach eliminates the reliance on clustering, thereby avoiding sensitivity to ambiguous token counts and improving decoding accuracy. The detailed encoding and decoding algorithms of MajorMark$^+$ and the implementation of the hash function are presented in Appendix K.

Decoding Complexity Analysis. MajorMark$^+$ needs to identify both the majority bit λ_i and its occurrence h_{λ_i} for each block. When block-level all-zero and all-one codes are excluded, each block has $b/r - 1$ feasible configurations. Since the block assignment of each token is independent of the hypothesized $(\lambda_i, h_{\lambda_i})$, the decoder can evaluate the corresponding configurations for all blocks within the same pass over the unverified token sequence $\mathbf{x}$. Therefore, MajorMark$^+$ requires only $(b/r - 1) \times |\mathbf{x}|$ decoding operations, which is much lower than enumeration-based methods such as CycleShift [9] and DepthW [20], which require 2^b enumerations.

5 Experiments

5.1 Experimental Settings

General Settings. By default, we consider a total of 20 users, corresponding to 20 randomly generated messages. Each user submits two prompts, and each prompt is used to generate a text of $T/2 = 250$ tokens, resulting in $T = 500$ tokens per user for decoding. We compare our proposed methods with four state-of-the-art perturbation-based methods, including DepthW [20], CycleShift [9], MPAC [43], and RSBH [31]. In the larger-scale evaluation, we additionally compare against the distortion-free methods StealthInk [14] and BiMark [8]. For DepthW and CycleShift, we only evaluate the case of $b = 8$, as their decoding time grows exponentially and becomes impractical for larger message lengths. For MajorMark, we use `KMeans` [25] clustering with $K = 2$ for decoding, and all other hyperparameters of `KMeans` are kept at their default settings. For MajorMark$^+$, the default number of blocks r is set to 2. All experiments were carried out on a self-managed Linux-based computing cluster running Ubuntu 20.04.6 LTS. The cluster is equipped with eight NVIDIA RTX A6000 GPUs (each with 48 GB of memory) and AMD EPYC 7763 CPUs featuring 64 cores. In total, the experiments accumulated roughly two weeks of GPU compute time.

Datasets and Models. Following prior work [16, 43, 31], we primarily evaluate our method on the text completion task using the C4 news dataset [32]. Unless otherwise stated, we use LLaMA-2-7B [36]. We additionally evaluate recent public LLMs in Section 5.2, with further results on additional models, datasets, and benchmark tasks provided in Appendix B.

Metrics. We evaluate the quality of generated text using perplexity (PPL) computed by a larger LLaMA-2-13B model. In addition, we report BERTScore and ROUGE in Appendix E, where the results show that our method maintains higher semantic and lexical quality than the other methods. We also report the Top-5 hit rate, which measures the proportion of generated tokens that appear among the top-5 logits of the original model output, indicating how often desirable tokens are still sampled. For decoding accuracy, we adopt bit accuracy (BA), following prior works [46, 24, 42, 43], defined as the proportion of correctly decoded bits relative to the ground-truth message.

5.2 Empirical Results

Main Results. We conduct a comprehensive evaluation of representative perturbation-based methods under varying message lengths ($b \in \{8, 32, 64\}$) and watermarking biases ($\delta \in \{2, 4, 6\}$). Table 2 reports BA, PPL, and Top-5 hit ratio for each setting. When $b = 8$, all methods achieve high BA

Table 2: Comparison of methods across various message lengths and watermarking biases. The PPL of non-watermarked text is 3.81, serving as a lower bound for the PPL of all watermarked texts.

Method	$\delta = 2$			$\delta = 4$			$\delta = 6$			Avg.	Avg.
	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓
Message length $b = 8$											
CycleShift	**100.00**	5.09	90.72	**100.00**	8.64	81.43	**100.00**	15.44	75.65	**100.00**	9.72
DepthW	95.62	4.53	91.65	**100.00**	8.24	79.53	**100.00**	25.49	61.13	98.54	12.75
MPAC	99.38	5.03	90.41	**100.00**	8.74	81.44	**100.00**	16.06	74.52	99.79	9.94
RSBH	**100.00**	4.80	91.19	**100.00**	6.36	89.30	**100.00**	8.66	86.87	**100.00**	6.61
MajorMark	**100.00**	4.50	92.19	**100.00**	5.87	**90.65**	**100.00**	7.81	89.23	**100.00**	6.06
MajorMark$^+$	**100.00**	**4.43**	**92.43**	**100.00**	**5.75**	90.36	**100.00**	**6.97**	**90.06**	**100.00**	**5.72**
Message length $b = 32$											
MPAC	89.22	5.03	90.09	98.75	9.37	80.75	**100.00**	15.32	75.35	95.99	9.91
RSBH	89.38	4.68	92.15	97.66	6.43	88.48	97.50	8.47	87.13	94.85	6.53
MajorMark	96.74	**4.43**	91.90	99.06	6.32	89.28	**100.00**	8.36	88.09	98.60	6.37
MajorMark$^+$	**97.81**	4.49	**92.42**	**100.00**	**6.05**	**90.17**	**100.00**	**7.90**	**88.49**	**99.27**	**6.15**
Message length $b = 64$											
MPAC	81.48	4.99	90.08	93.59	8.64	82.01	96.48	15.95	75.14	90.52	9.86
RSBH	81.25	4.93	91.80	90.39	6.41	88.91	97.58	9.22	87.06	89.74	6.85
MajorMark	83.59	4.78	91.46	93.83	**6.11**	**89.34**	98.67	8.82	87.27	92.03	6.57
MajorMark$^+$	**86.95**	**4.74**	**91.83**	**97.81**	6.21	89.30	**99.69**	**7.93**	**88.43**	**94.82**	**6.29**

across all watermarking bias settings. In this low-bit regime, MajorMark and MajorMark$^+$ exhibit the best text quality, reflected by their average PPLs of 6.06 and 5.72, which are the second-lowest and lowest among all methods. As b increases, all methods generally exhibit lower BA and higher PPL due to the increased difficulty of encoding and decoding longer messages. Nonetheless, MajorMark and MajorMark$^+$ consistently maintain strong performance, outperforming MPAC and RSBH in both PPL and BA. For example, when $b = 64$, MajorMark and MajorMark$^+$ achieve PPLs of 6.57 and 6.29, which represent improvements of $+0.28$ and $+0.56$ over RSBH's PPL of 6.85. These improvements stem from the majority bit-aware encoding strategy, which enables a larger green list during encoding, thereby better preserving the text quality. Furthermore, our methods achieve higher Top-5 hit rates across all settings, further demonstrating their ability to maintain text quality. In terms of BA, they achieve 92.03% and 94.82%, outperforming MPAC's 90.52% by $+1.51\%$ and $+4.30\%$. We report the standard deviations and 95% CIs for the results with $b = \{32, 64\}$ in Appendix B.

Results on Available Tokens T for Decoding. Figure 4 reports BA under different numbers of observed tokens $T \in \{400, 450, 500, 550, 600\}$, with $b = 64$ and $\delta = 2$ (left) and $\delta = 4$ (right). As expected, BA increases for all methods as T grows. Under $\delta = 2$, MajorMark and MajorMark$^+$ consistently outperform MPAC and RSBH across all token budgets, indicating higher decoding efficiency when only limited tokens are available. When δ increases to 4, the advantage of MajorMark$^+$ becomes more pronounced. For example, at $T = 400$, MPAC and RSBH achieve BA of 91.48% and 89.22%, respectively, while MajorMark$^+$ reaches 95.78%, improving BA by $+6.56\%$ over MPAC and $+4.30\%$ over RSBH.

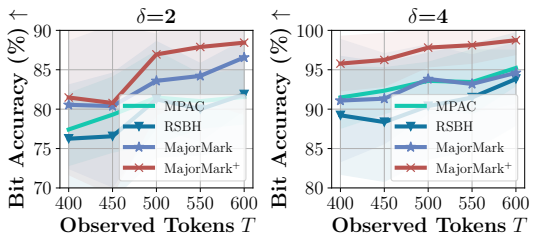


Figure 4: BA of different methods under varying numbers of observed tokens T with $b = 64$.

Additional Results on Message Length. We further investigate how the message length ($b \in \{32, 40, 48, 56, 64\}$) affects the performance of different watermarking methods on decoding accuracy (left) and text quality (right) under fixed decoding token budget $T = 500$, as shown in Figure 5. A larger b increases the complexity of the decoding process, which generally leads to reduced BA. As expected, all methods exhibit a decline in BA as b increases. MajorMark maintains a comparable BA to MPAC but achieves significantly better

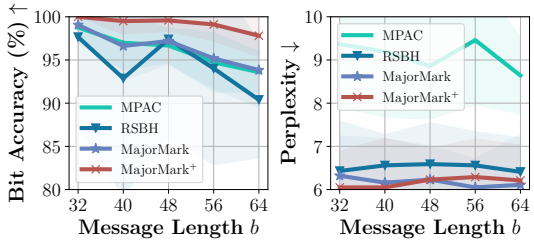


Figure 5: BA and PPL of methods with increasing message length $b \in \{32, 40, 48, 56, 64\}$ with $\delta = 4$.

PPL. Notably, our proposed method, MajorMark$^+$, consistently yields the highest BA while maintaining a low PPL across all message lengths, demonstrating its robustness even when b is as large as 64. These results demonstrate that under a fixed decoding token budget, our methods MajorMark and MajorMark$^+$ not only maintain high decoding accuracy but also ensuring high quality of the watermarked texts.

Larger-Scale Experiments. To further evaluate the generality of our method, we conduct additional experiments with 100 users on two recent public LLMs, including LLaMA-3.1-8B and Qwen3-8B [40], and two distortion-free watermarking methods StealthInk [14] and BiMark [8]. The results with $b = 32$ and $\delta = 3$ for MPAC and our methods are summarized in Table 3. Note that for StealthInk and BiMark we adopt their default settings. As shown in Table 3, StealthInk maintains relatively low PPL, but its decoding accuracy is less reliable, with BA dropping to 49.31% on Qwen3-8B, which is close to random guessing. In contrast, MPAC and BiMark achieve higher BA, but they introduce larger degradation in text quality, especially on LLaMA-3.1-8B, where their PPL values exceed 8.3. Compared with these baselines, MajorMark$^+$ achieves a better balance between text quality and decoding accuracy across both models. Specifically, it achieves the highest BA on both LLaMA-3.1-8B and Qwen3-8B, reaching 99.81% and 98.50%, respectively, while maintaining substantially lower PPL than the high-BA baselines MPAC and BiMark.

Table 3: Evaluation on recent public LLMs with 100 users.

Method	LLaMA-3.1-8B		Qwen3-8B	
	PPL↓	BA↑	PPL↓	BA↑
MPAC	8.34	97.72	5.65	92.66
StealthInk	**4.83**	83.84	**3.50**	49.31
BiMark	8.39	99.47	5.05	96.66
MajorMark	6.79	98.41	4.91	96.91
MajorMark$^+$	6.51	**99.81**	4.75	**98.50**

Robustness Against Attacks. We evaluate the robustness of our methods against two widely studied post-generation text editing attacks: *Copy-Paste* and *Paraphrase* attacks [44]. Following previous works [43, 31], we replace 10% of the watermarked text with non-watermarked text while keeping the total length unchanged. For the Paraphrase attack, we use the DIPPER proposed in [18] as the paraphraser with $L = 20$ and $O = 20$. Figure 6 reports BA results under both attacks, evaluated across watermarking biases $\delta \in \{2, 4, 6\}$ with a fixed message length $b = 32$. For the Copy-Paste attack, MajorMark and MajorMark$^+$ generally outperform MPAC and RSBH across δ values, demonstrating their robustness even when some tokens are unwatermarked in the modified text. Paraphrase attacks remain a significant open challenge in the literature: all methods experience BA degradation under this attack. While MPAC benefits from its small γ and achieves higher BA at larger δ, our methods maintain comparable performance to MPAC and significantly outperform RSBH. Notably, MajorMark achieves slightly higher BA than MajorMark$^+$, likely due to its simpler decoding process. More specifically, MajorMark$^+$ requires enumerating all combinations of majority bits and their occurrences for each message block. Under the Paraphrase attack, token occurrences become more diverse and less reliable, leading to a slight degradation in BA of MajorMark$^+$. Additionally, we theoretically analyze the robustness of our methods under text editing attacks in Appendix M.

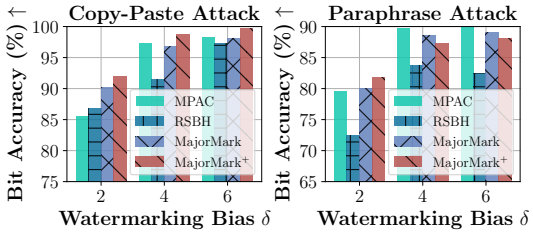


Figure 6: BA of different methods under Copy-Paste and Paraphrase attacks.

Sentence Shuffling and Character-Level Perturbation Attacks. We further evaluate the robustness of different methods under two types of text perturbations: sentence shuffling (SenS) and character-level perturbation (CharP) [45]. For SenS, we randomly shuffle the sentences in the generated text. For CharP, following [45], we randomly perturb 10% of the characters. The results are summarized in Table 4. Since all methods rely on token-level decoding, SenS causes only limited BA degradation. In contrast, CharP substantially reduces the decoding accuracy of all methods because character-level perturbations can change tokenization and disturb the recovered green-list statistics. Despite this stronger perturbation, MajorMark and MajorMark$^+$ still achieve competitive decoding accuracy

Table 4: Robustness evaluation under character-level perturbation (CharP) and sentence shuffling (SenS).

Method	PPL↓	BA↑	
		CharP	SenS
MPAC	6.36	81.88	95.31
RSBH	5.47	78.28	93.25
MajorMark	5.31	**82.84**	94.97
MajorMark$^+$	**5.19**	81.56	**96.72**

compared with MPAC and RSBH, while maintaining lower PPL. In particular, MajorMark$^+$ achieves the best BA under SenS, reaching 96.72%, and obtains the lowest PPL among all methods.

Runtime Analysis. We evaluate the computational overhead by measuring the average encoding and decoding time under default settings ($b = 32$), with results shown in Table 5. For encoding, our methods add negligible overhead compared to standard generation (about 36s). For decoding, MajorMark requires two passes over the suspect text and an additional clustering step, leading to a moderate overhead compared to MPAC (2.18s vs. 0.13s). MajorMark$^+$ performs $b/r - 1$ decoding passes and takes 12.70s in total. This is still much faster than exponential-time methods such as DepthW (about 8.89×10^8s), and also faster than RSBH (25.56s). Meanwhile, our methods achieve substantially higher BA than existing approaches, showing that they introduce only modest computational cost while providing strong decoding performance for practical watermark verification.

Table 5: Comparison of runtime (in seconds) and BA. The decoding time for DepthW is estimated.

Method	**Enc.** (s)	**Dec.** (s)	**BA**↑
DepthW	37.87	$\approx 8.89 \times 10^8$ (Est.)	-
MPAC	36.13	0.13	89.22
RSBH	36.06	25.56	89.38
MajorMark	36.58	2.18	96.74
MajorMark$^+$	36.85	12.70	97.81

Ablation Study. We present the ablation study on two key design choices: the clustering method used by MajorMark and the number of message blocks r in MajorMark$^+$. Table 6 summarizes the resulting BA and PPL with message length fixed at $b = 32$. For MajorMark, we evaluate two additional clustering methods beyond the default `KMeans`: Agglomerative Clustering (`AC`) [26] and Gaussian Mixture Models (`GMM`) [33]. With a strong watermarking bias ($\delta = 4$), all three methods achieve comparable BA. When the bias is weaker ($\delta = 2$), `KMeans` slightly outperforms the others. Overall, the performance remains stable across different clustering choices, indicating that MajorMark is robust to the selection of the clustering method. For MajorMark$^+$, we vary the number of blocks $r \in \{1, 2, 4, 8\}$, where $r = 1$ treats the message as a single block. Increasing r generally improves PPL, which matches our analysis of how block granularity affects the expected green list size γ. Among these settings, $r = 2$ provides a good balance between BA and PPL, making it a practical choice for real-world use. Additionally, we plot the Pareto curves for different (r, δ) in Appendix I.

Table 6: Ablation study of MajorMark (first three rows) and MajorMark$^+$ (last four rows), evaluating clustering methods and block sizes, respectively.

Method	$\delta = 2$		$\delta = 4$		**Avg.**	**Avg.**
	BA↑	**PPL**↓	**BA**↑	**PPL**↓	**BA**↑	**PPL**↓
`KMeans`	96.74	4.43	99.06	6.32	97.90	5.38
`AC`	94.06	4.43	99.84	6.32	96.95	5.38
`GMM`	92.81	4.43	99.84	6.32	96.33	5.38
$r = 1$	95.94	4.74	100.00	6.55	97.97	5.65
$r = 2$	97.81	4.49	100.00	6.05	98.91	5.27
$r = 4$	90.00	4.42	98.44	5.61	94.22	5.02
$r = 8$	91.88	4.46	98.59	5.20	95.24	4.83

Additional Results and Discussions. Examples of watermarked texts are provided in Appendix F. We discuss how our methods address practical false positive cases in Appendix G. We further discuss how extremely imbalanced messages impact our methods in Appendix O.

6 Conclusion

We find that existing perturbation-based multi-bit watermarking methods rely on counting how frequently generated tokens fall within the reconstructed green list to decode the embedded message. This paradigm inherently constrains the size of the green list to ensure decoding accuracy. To address this, we propose MajorMark and MajorMark$^+$, both of which employ a novel majority bit-aware encoding strategy. This design enables the construction of larger green lists, as we theoretically prove. Extensive experiments demonstrate that MajorMark and MajorMark$^+$ consistently outperform existing baselines in both text utility and decoding accuracy.

While our majority bit-aware encoding automatically determines the green list size for each message, this design imposes constraints on the flexibility of γ in certain scenarios. In the future, it would be desirable to develop methods that preserve the advantages of MajorMark and MajorMark$^+$, namely, their independence from green list token frequencies, while allowing for a more flexible specification of γ. In addition, MajorMark$^+$ requires additional computation time to decode messages from the generated text. Although it remains more efficient than several existing approaches, further improving its decoding efficiency remains a promising direction for future work. Co-designing the green list ratio γ and the watermarking bias δ is also a promising future direction.

Acknowledgements and Disclosures

This material is based upon work co-supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Contract No. DE-AC05-00OR22725. This manuscript has been co-authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

References

[1] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[2] Yapei Chang, Kalpesh Krishna, Amir Houmansadr, John Frederick Wieting, and Mohit Iyyer. PostMark: A robust blackbox watermark for large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8969–8987, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[3] Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 1125–1139. PMLR, 2024.

[4] Amirhossein Dabiriaghdam and Lele Wang. Simmark: A robust sentence-level similarity-based watermarking algorithm for large language models. *arXiv preprint arXiv:2502.02787*, 2025.

[5] Abdulrahman Diaa, Toluwani Aremu, and Nils Lukas. Optimizing adaptive attacks against watermarks for language models. In *Forty-second International Conference on Machine Learning*, 2025.

[6] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[7] William Feller. *An introduction to probability theory and its applications, Volume 2*, volume 2. John Wiley & Sons, 1991.

[8] Xiaoyan Feng, He Zhang, Yanjun Zhang, Leo Yu Zhang, and Shirui Pan. Bimark: Unbiased multilayer watermarking for large language models. In *Forty-second International Conference on Machine Learning*, 2025.

[9] Pierre Fernandez, Antoine Chaffin, Karim Tit, Vivien Chappelier, and Teddy Furon. Three bricks to consolidate watermarks for large language models. In *2023 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6. IEEE, 2023.

[10] Eva Giboulot and Teddy Furon. WaterMax: breaking the LLM watermark detectability-robustness-quality trade-off. *Advances in Neural Information Processing Systems*, 37:18848–18881, 2024.

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

[12] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

[13] Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. Semstamp: A semantic watermark with paraphrastic robustness for text generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4067–4082, 2024.

[14] Ya Jiang, Chuxiong Wu, Massieh Kordi Boroujeny, Brian Mark, and Kai Zeng. Stealthink: A multi-bit and stealthy watermark for large language models. In *Forty-second International Conference on Machine Learning*, 2025.

[15] Nikola Jovanović, Robin Staab, and Martin Vechev. Watermark stealing in large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 22570–22593. PMLR, 21–27 Jul 2024.

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

[17] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. On the reliability of watermarks for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[18] Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36:27469–27500, 2023.

[19] Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *Transactions on Machine Learning Research*, 2024.

[20] Liying Li, Yihan Bai, and Minhao Cheng. Where am I from? identifying origin of LLM-generated content. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12218–12229, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[21] Aiwei Liu, Leyi Pan, Xuming Hu, Shuang Li, Lijie Wen, Irwin King, and Philip S. Yu. An unforgeable publicly verifiable watermark for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[22] Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[23] Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. A survey of text watermarking in the era of large language models. *ACM Computing Surveys*, 57(2):1–36, 2024.

[24] Xiyang Luo, Ruohan Zhan, Huiwen Chang, Feng Yang, and Peyman Milanfar. Distortion agnostic deep watermarking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13548–13557, 2020.

[25] James B McQueen. Some methods of classification and analysis of multivariate observations. In *Proc. of 5th Berkeley Symposium on Math. Stat. and Prob.*, pages 281–297, 1967.

[26] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

[27] Travis Munyer, Abdullah All Tanvir, Arjon Das, and Xin Zhong. Deeptextmark: a deep learning-driven text watermarking approach for identifying large language model generated text. *IEEE Access*, 12:40508–40520, 2024.

[28] Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, et al. MarkLLM: An open-source toolkit for LLM watermarking. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 61–71, 2024.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[30] Julien Piet, Chawin Sitawarin, Vivian Fang, Norman Mu, and David Wagner. Markmywords: Analyzing and evaluating language model watermarks. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 68–91. IEEE, 2025.

[31] Wenjie Qu, Wengrui Zheng, Tianyang Tao, Dong Yin, Yanze Jiang, Zhihua Tian, Wei Zou, Jinyuan Jia, and Jiaheng Zhang. Provably robust multi-bit watermarking for ai-generated text. *arXiv preprint arXiv:2401.16820*, 2024.

[32] 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.

[33] Douglas Reynolds. Gaussian mixture models. In *Encyclopedia of biometrics*, pages 827–832. Springer, 2015.

[34] Christoph Schuhmann. Essays with instructions dataset. `https://huggingface.co/datasets/ChristophSchuhmann/essays-with-instructions`, 2023. Accessed: 2025-07-14.

[35] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[36] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[37] Lean Wang, Wenkai Yang, Deli Chen, Hao Zhou, Yankai Lin, Fandong Meng, Jie Zhou, and Xu Sun. Towards codable watermarking for injecting multi-bits information to LLMs. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[38] Zongqi Wang, Tianle Gu, Baoyuan Wu, and Yujiu Yang. Morphmark: Flexible adaptive watermarking for large language models. *arXiv preprint arXiv:2505.11541*, 2025.

[39] Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. A survey on LLM-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, 51(1):275–338, 2025.

[40] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

[41] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[42] Xi Yang, Jie Zhang, Kejiang Chen, Weiming Zhang, Zehua Ma, Feng Wang, and Nenghai Yu. Tracing text provenance via context-aware lexical substitution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11613–11621, 2022.

[43] KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. Advancing beyond identification: Multi-bit watermark for large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4031–4055, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[44] Ruisi Zhang, Shehzeen Samarah Hussain, Paarth Neekhara, and Farinaz Koushanfar. REMARK-LLM: A robust and efficient watermarking framework for generative large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1813–1830, 2024.

[45] Zhaoxi Zhang, Xiaomei Zhang, Yanjun Zhang, He Zhang, Shirui Pan, Bo Liu, Asif Qumer Gill, and Leo Yu Zhang. Character-level perturbations disrupt LLM watermarks. *arXiv preprint arXiv:2509.09112*, 2025.

[46] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 657–672, 2018.

Table 7: Notation table.

Symbol	Description
γ	The green list ratio
$\mathbf{m}$	The binary multi-bit message
m_i	The i-th bit of the message $\mathbf{m}$
b	The length of message $\mathbf{m}$
f	The large language model
$\mathbf{x}_p$	The input prompt
$\mathbf{x}$	The unverified text
x_t	The t-th token generated by f
$\mathcal{V}$	The vocabulary of f
T	The total generation step of f and the length of $\mathbf{x}_g$
$\mathbf{x}_g$	The generated text of f without watermark
$\mathbf{x}'_g$	The generated text of f with watermark
$\mathcal{G}$	The green list
$\mathcal{R}$	The red list
δ	The watermarking bias
$\texttt{Enc}(\cdot)$	The encoding function for a watermarking method
$\texttt{Dec}(\cdot)$	The decoding function for a watermarking method
$\mathbf{m}'$	The decoded message via $\texttt{Dec}(\cdot)$
$\texttt{Quality}(\cdot)$	The quality function
$\texttt{PPL}(\cdot)$	The conditional perplexity
τ	The tolerance margin on text quality for $\texttt{Enc}(\cdot)$
λ	The majority bit of a message $\mathbf{m}$
k	The secret hash key
$\texttt{Hash}(\cdot)$	The hash function
s	The pseudo-random seed generated by $\texttt{Hash}(\cdot)$
$\mathcal{C}$	The cluster with larger mean produced by MajorMark
r	The number of blocks in MajorMark$^+$
h_0 or h_1	The occurrences of bit 0 or 1 in a message
d	The number of bits in a message block

A Notation Table

We present the detailed notation table in Table 7.

B Additional Results

Results with Uncertainty Estimates. We report the standard deviation and 95% confidence interval (CI) for the results with message lengths ($b = 32$) and ($b = 64$) in Table 8. These statistics provide additional information about the variability of BA and PPL across different messages.

Results on Additional LLMs We additionally report the performance of our methods on three other popular language models: Qwen2.5-7B [41], Gemma-2B [35], and LLaMA-3.1-8B [11]. Specifically, we present the BA, PPL, and Top-5 hit rate results under message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{2, 4\}$ in Table 9. Note that for evaluating PPL, we use the larger models Qwen2.5-32B, Gemma-7B, and LLaMA-3.1-70B. As shown, both MajorMark and MajorMark$^+$ perform well across the board. These results demonstrate the strong generalization ability of MajorMark and MajorMark$^+$ across different LLM architectures and capacities.

Additional Results on OpenGen and Essays Datasets We conduct a comprehensive evaluation of MPAC, RSBH, and our two proposed methods, MajorMark and MajorMark$^+$, on the OpenGen [18] and Essays [34] datasets. Results for message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{2, 4, 6\}$ are reported in Table 10 and Table 11, respectively.

Consistent with the trends observed on the C4 dataset, our methods consistently outperform the state-of-the-art baselines. In particular, MajorMark$^+$ achieves the highest average BA and Top-5 hit

Table 8: Part of results from Table 2 with standard deviations and 95% confidence intervals. The first line in each cell reports mean$_{\pm \mathrm{STD}}$, and the second line reports the 95% confidence interval.

Method	$\delta = 2$		$\delta = 4$		$\delta = 6$	
	BA↑	**PPL↓**	**BA↑**	**PPL↓**	**BA↑**	**PPL↓**
Message length $b = 32$						
MPAC	$89.22_{\pm 6.66}$ [86.10, 92.34]	$5.03_{\pm 0.70}$ [4.70, 5.36]	$98.75_{\pm 1.82}$ [97.90, 99.60]	$9.37_{\pm 1.47}$ [8.68, 10.06]	$\mathbf{100.00}_{\pm 0.00}$ [100.00, 100.00]	$15.32_{\pm 2.46}$ [14.17, 16.47]
RSBH	$89.38_{\pm 11.67}$ [83.92, 94.84]	$4.68_{\pm 0.47}$ [4.46, 4.90]	$97.66_{\pm 10.22}$ [92.88, 102.44]	$6.43_{\pm 1.16}$ [5.89, 6.97]	$97.50_{\pm 10.90}$ [92.40, 102.60]	$8.47_{\pm 1.25}$ [7.88, 9.06]
MajorMark	$96.74_{\pm 3.07}$ [95.30, 98.18]	$\mathbf{4.43}_{\pm 0.52}$ [4.19, 4.67]	$99.06_{\pm 1.74}$ [98.25, 99.87]	$6.32_{\pm 0.92}$ [5.89, 6.75]	$\mathbf{100.00}_{\pm 0.00}$ [100.00, 100.00]	$8.36_{\pm 1.35}$ [7.73, 8.99]
MajorMark$^+$	$\mathbf{97.81}_{\pm 2.98}$ [96.42, 99.20]	$4.49_{\pm 0.75}$ [4.14, 4.84]	$\mathbf{100.00}_{\pm 0.00}$ [100.00, 100.00]	$\mathbf{6.05}_{\pm 0.84}$ [5.66, 6.44]	$\mathbf{100.00}_{\pm 0.00}$ [100.00, 100.00]	$\mathbf{7.90}_{\pm 1.41}$ [7.24, 8.56]
Message length $b = 64$						
MPAC	$81.48_{\pm 5.89}$ [78.72, 84.24]	$4.99_{\pm 0.66}$ [4.68, 5.30]	$93.59_{\pm 2.36}$ [92.49, 94.69]	$8.64_{\pm 0.89}$ [8.22, 9.06]	$96.48_{\pm 2.46}$ [95.33, 97.63]	$15.95_{\pm 2.21}$ [14.92, 16.98]
RSBH	$81.25_{\pm 7.33}$ [77.82, 84.68]	$4.93_{\pm 0.80}$ [4.56, 5.30]	$90.39_{\pm 6.73}$ [87.24, 93.54]	$6.41_{\pm 0.79}$ [6.04, 6.78]	$97.58_{\pm 4.24}$ [95.60, 99.56]	$9.22_{\pm 1.01}$ [8.75, 9.69]
MajorMark	$83.59_{\pm 5.94}$ [80.81, 86.37]	$4.78_{\pm 0.80}$ [4.41, 5.15]	$93.83_{\pm 4.35}$ [91.79, 95.87]	$\mathbf{6.11}_{\pm 1.17}$ [5.56, 6.66]	$98.67_{\pm 1.24}$ [98.09, 99.25]	$8.82_{\pm 1.36}$ [8.18, 9.46]
MajorMark$^+$	$\mathbf{86.95}_{\pm 7.64}$ [83.37, 90.53]	$\mathbf{4.74}_{\pm 0.86}$ [4.34, 5.14]	$\mathbf{97.81}_{\pm 2.44}$ [96.67, 98.95]	$6.21_{\pm 0.82}$ [5.83, 6.59]	$\mathbf{99.69}_{\pm 1.36}$ [99.05, 100.33]	$\mathbf{7.93}_{\pm 1.27}$ [7.34, 8.52]

Table 9: Performance of MajorMark and MajorMark$^+$ on Qwen2.5-7B, Gemma-2B, and LLaMA-3.1-8B with $b \in \{32, 64\}$ and $\delta \in \{2, 4\}$. The PPL for non-watermarked texts generated by Qwen2.5-7B, Gemma-2B, and LLaMA-3.1-8B are 4.97, 5.71, and 4.27, respectively.

Model	Method	$\delta = 2$			$\delta = 4$			Avg.	Avg.	Avg.
		BA↑	**PPL↓**	**Top-5↑**	**BA↑**	**PPL↓**	**Top-5↑**	**BA↑**	**PPL↓**	**Top-5↑**
Message length $b = 32$										
Qwen2.5-7B	MajorMark	96.25	6.69	86.99	99.22	8.49	85.29	97.74	7.59	86.14
	MajorMark$^+$	98.75	6.31	88.39	100.00	8.85	84.77	**99.38**	**7.58**	**86.58**
Gemma-2B	MajorMark	96.41	7.02	87.16	99.06	9.80	83.38	97.74	8.41	**85.27**
	MajorMark$^+$	99.38	6.95	86.02	100.00	9.69	84.36	**99.69**	**8.32**	85.19
LLaMA-3.1-8B	MajorMark	94.84	5.69	90.85	99.69	7.94	88.56	97.27	6.82	89.71
	MajorMark$^+$	97.81	5.78	90.90	100.00	7.57	88.75	**98.91**	**6.68**	**89.83**
Message length $b = 64$										
Qwen2.5-7B	MajorMark	88.83	6.43	88.02	96.72	9.19	84.27	92.78	7.81	86.14
	MajorMark$^+$	92.03	6.31	88.20	99.22	9.00	84.25	**95.63**	**7.66**	**86.23**
Gemma-2B	MajorMark	88.44	7.54	85.82	96.72	10.79	82.72	92.58	9.17	84.27
	MajorMark$^+$	93.75	7.41	85.93	99.38	9.82	82.75	**96.56**	**8.62**	**84.34**
LLaMA-3.1-8B	MajorMark	86.17	5.33	92.11	95.94	7.58	88.48	91.06	**6.46**	**90.30**
	MajorMark$^+$	92.50	5.72	91.32	99.06	7.67	88.51	**95.78**	6.70	89.92

rate, while maintaining the lowest PPL across all settings. This demonstrates its strong ability to preserve the utility of watermarked text while ensuring high decoding accuracy. For example, on the Essays dataset with $b = 64$, MajorMark$^+$ achieves a BA of 95.21%, which is $+5.99\%$ and $+5.52\%$ higher than MPAC and RSBH, respectively. In terms of PPL, MajorMark$^+$ obtains the lowest score of 6.03, outperforming MPAC and RSBH by margins of 3.20 and 0.19, respectively. These results further confirm the strong generalization ability of both MajorMark and MajorMark$^+$ across diverse datasets, highlighting the practical effectiveness of our methods in varying domains.

Results on Story Generation Task. In this section, we evaluate existing methods alongside our two proposed approaches on the story generation task. Specifically, we use the WritingPrompts dataset [6] with the LLaMA-2-7B-chat model [36]. The chat prompt is designed to encourage coherent and imaginative story generation and is defined as follows:

> `[System]` *You are a helpful assistant that writes engaging and coherent stories.*
>
> `[User]` *Please write a detailed and imaginative short story based on the following prompt: [prompt]*

Results for message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{4, 6\}$ are summarized in Table 12. Our methods, MajorMark and MajorMark$^+$, consistently achieve strong performance on

Table 10: Performance comparison of watermarking methods (MPAC, RSBH, MajorMark, and MajorMark$^+$) on the OpenGen dataset with message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{2, 4, 6\}$.

Method	$\delta = 2$			$\delta = 4$			$\delta = 6$			Avg.	Avg.	Avg.
	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑
Message length $b = 32$												
MPAC	86.56	4.77	91.26	97.34	9.23	81.86	99.38	15.08	76.41	94.43	9.69	83.18
RSBH	84.53	4.68	92.47	96.56	6.43	89.98	97.66	8.64	88.47	92.92	6.58	90.31
MajorMark	90.94	4.35	92.83	97.81	6.02	90.01	99.38	8.51	87.95	96.04	6.29	90.26
MajorMark$^+$	**92.03**	**4.09**	**93.85**	**99.38**	**5.63**	**91.16**	**100.00**	**7.96**	**88.90**	**97.14**	**5.89**	**91.30**
Message length $b = 64$												
MPAC	78.12	4.83	90.64	90.70	8.95	82.52	96.33	15.79	75.50	88.38	9.86	82.89
RSBH	77.11	4.53	**93.05**	89.06	6.33	**90.25**	96.95	8.59	87.79	87.71	6.48	90.36
MajorMark	**84.06**	4.77	92.24	93.44	**6.14**	89.97	96.02	8.42	88.22	91.17	6.44	90.14
MajorMark$^+$	79.92	**4.41**	92.62	**97.66**	6.37	89.73	**99.06**	**8.01**	**88.98**	**92.21**	**6.26**	**90.44**

Table 11: Performance comparison of watermarking methods (MPAC, RSBH, MajorMark, and MajorMark$^+$) on the Essays dataset with message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{2, 4, 6\}$.

Method	$\delta = 2$			$\delta = 4$			$\delta = 6$			Avg.	Avg.	Avg.
	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑
Message length $b = 32$												
MPAC	88.44	4.29	92.18	98.75	9.28	80.54	99.69	14.91	75.30	95.63	9.49	82.67
RSBH	89.22	4.23	93.31	97.66	6.33	88.98	97.66	7.78	87.78	94.85	6.11	90.02
MajorMark	91.41	**4.12**	93.24	98.28	5.98	89.60	99.53	7.71	87.70	96.41	5.94	90.18
MajorMark$^+$	**96.72**	4.21	**93.37**	**100.00**	**5.83**	**90.44**	**100.00**	**7.40**	**88.92**	**98.91**	**5.81**	**90.91**
Message length $b = 64$												
MPAC	78.36	4.33	92.85	92.58	9.12	80.94	96.72	14.23	75.83	89.22	9.23	83.21
RSBH	76.72	**4.05**	93.50	93.67	6.11	89.25	98.67	8.51	87.16	89.69	6.22	89.97
MajorMark	82.73	4.33	92.56	94.84	6.07	89.48	96.88	8.46	87.07	91.48	6.29	89.70
MajorMark$^+$	**87.19**	4.14	**93.56**	**98.75**	**5.99**	**89.83**	**99.69**	**7.95**	**87.42**	**95.21**	**6.03**	**90.27**

the story generation task. In particular, across both watermarking biases, MajorMark$^+$ yields the highest average BA and Top-5 hit ratios, as well as the lowest PPL, indicating its effectiveness in embedding reliable watermarks while preserving text quality. Although MajorMark performs slightly worse than MajorMark$^+$, it still outperforms existing state-of-the-art baselines, demonstrating its robustness and practicality.

Results on Text Summarization Task. In this section, we evaluate existing methods alongside our two proposed approaches on the text summarization task. Specifically, we use the CNN/DailyMail dataset [12] with the LLaMA-2-7B-chat model as the backbone. The chat prompt is designed to instruct the model to generate concise and faithful summaries given an input article, using the following format:

> `[System]` *You are a helpful assistant specialized in summarization. You take a document and write a concise, faithful summary.*
>
> `[User]` *Please summarize the following article in a few sentences: [article]*

Results for message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{4, 6\}$ are summarized in Table 13. Similar to the results observed for text completion and story generation, our methods generally achieve both the highest text quality and the highest decoding accuracy across both watermarking bias settings. These findings demonstrate the strong generalization ability of our proposed methods, MajorMark and MajorMark$^+$, across diverse natural language processing tasks. This highlights their potential as practical candidates for real-world watermarking deployment.

We further evaluate whether the observed improvements are statistically significant under the 100-user setting. Specifically, we conduct a one-sided Wilcoxon signed-rank test on the paired per-user PPL and BA results, comparing MajorMark$^+$ against BiMark on Qwen3-8B. For PPL, the alternative hypothesis is that

Table 14: Statistical significance test over 100 users on Qwen3-8B. We report the mean and standard deviation of each metric across users.

Metric	MajorMark$^+$	BiMark	W Statistic	p-value
PPL	4.75 ± 0.95	5.05 ± 1.23	1633.0	0.00108
BA	98.50 ± 4.41	96.66 ± 4.66	1094.0	0.00002

Table 12: Performance comparison of watermarking methods on the WritingPrompts dataset for the story generation task with message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{4, 6\}$.

Method	$\delta = 4$			$\delta = 6$			Avg.	Avg.	Avg.
	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑
Message length $b = 32$									
MPAC	84.69	4.69	96.09	98.28	13.65	85.41	91.49	9.17	90.75
RSBH	83.28	4.17	98.17	96.41	6.20	96.81	89.85	5.19	97.49
MajorMark	89.38	4.09	98.70	96.41	6.19	97.12	92.90	5.14	97.91
MajorMark$^+$	**91.88**	**3.91**	**98.72**	**98.44**	**5.52**	**97.51**	**95.16**	**4.72**	**98.12**
Message length $b = 64$									
MPAC	76.33	4.65	96.02	91.33	11.32	87.04	83.83	7.99	91.53
RSBH	76.41	4.14	98.36	85.39	6.18	97.06	80.90	5.16	97.71
MajorMark	80.94	4.05	**98.93**	93.05	6.08	97.13	87.00	5.07	98.03
MajorMark$^+$	**83.83**	**4.00**	98.69	**94.53**	**5.63**	**97.40**	**89.18**	**4.82**	**98.05**

Table 13: Performance comparison of watermarking methods on the CNN/DailyMail dataset for the text summarization task with message lengths $b \in \{32, 64\}$ and watermarking biases $\delta \in \{4, 6\}$.

Method	$\delta = 4$			$\delta = 6$			Avg.	Avg.	Avg.
	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑	BA↑	PPL↓	Top-5↑
Message length $b = 32$									
MPAC	83.12	6.36	96.68	96.41	12.25	90.48	89.77	9.31	93.58
RSBH	85.00	5.81	98.83	**97.34**	8.07	97.24	91.17	6.94	98.04
MajorMark	88.12	5.72	99.02	95.78	7.70	98.00	91.95	6.71	98.51
MajorMark$^+$	**89.53**	**5.72**	**99.10**	95.16	**7.13**	**98.52**	**92.35**	**6.43**	**98.81**
Message length $b = 64$									
MPAC	78.67	6.55	97.06	91.33	12.86	90.09	85.00	9.71	93.58
RSBH	73.05	5.99	98.81	84.92	7.87	97.31	78.99	6.93	98.06
MajorMark	83.36	5.73	99.02	89.30	8.18	97.33	86.33	6.95	98.18
MajorMark$^+$	**84.22**	**5.51**	**99.39**	**91.72**	**7.33**	**98.01**	**87.97**	**6.42**	**98.70**

MajorMark$^+$ achieves lower PPL than BiMark. For BA, the alternative hypothesis is that MajorMark$^+$ achieves higher BA than BiMark. The results are summarized in Table 14. MajorMark$^+$ achieves a statistically significant reduction in PPL, with $p = 0.00108 < 0.05$, and a statistically significant improvement in BA, with $p = 0.00002 < 0.05$. These results indicate that the improvements of MajorMark$^+$ over BiMark are unlikely to be caused by random variation among users.

C Discussion on Spoofing Attack

Spoofing Attack. Watermark spoofing, introduced by [15], attempts to forge text that is falsely detected as watermarked, potentially harming the LLM provider's reputation. However, as noted in [14], multi-bit watermarking embeds diverse message signals across tokens, which significantly increases the complexity of producing successful spoofs. Furthermore, [31] show that when the watermarking method employs a hash function modeled as a random oracle (ROM) [1], spoofing attacks become ineffective. Consequently, we do not consider spoofing attacks in this work.

D Security Analysis of the Majority-Bit Structure

Information Revealed by the Green-List Ratio. We first consider whether the message-dependent green-list ratio γ may reveal information about the embedded message. If γ were directly observable, it would reveal the magnitude of the majority margin, since $|h_0 - h_1| = b(2\gamma - 1)$, where h_0 and h_1 denote the numbers of 0s and 1s in the message, respectively. However, γ does not reveal the value of the majority bit λ. For uniformly sampled messages, the majority margin is strongly concentrated around balanced messages. Its Shannon entropy is approximately 2.69 bits for $b = 32$ and 3.15 bits for $b = 64$, indicating that even exact knowledge of γ reveals only limited information about the original b-bit message.

More importantly, γ is not directly observable from the generated text without the secret key. The token-to-shard mapping is determined by the keyed, context-dependent hash function, and therefore an adversary without the key cannot directly reconstruct the shards or identify which shards form the

green list. We nevertheless note that γ may potentially be inferred approximately from distributional characteristics of the generated text, since a larger green-list ratio generally requires less distributional perturbation.

Key-Free Watermark Stealing. The regular shard structure also does not directly provide a fixed target for key-free watermark stealing. In MajorMark, the token-to-shard mapping is generated from the secret key, the preceding context, and the majority bit, and therefore changes across generation steps. As a result, shard indices are not consistently aligned from an adversary's perspective. Moreover, for randomly generated messages, no particular shard index is systematically favored across messages, since each shard has the same probability of belonging to the green list. These properties make it difficult for an attacker without the secret key to identify a fixed shard-level pattern that can be reused for watermark stealing.

E Semantic and Lexical Fidelity of Watermarked Texts

Recall that in evaluation, we mainly use PPL as the metric for evaluating the quality of watermarked texts. To provide a more complete analysis, we further evaluate BERTScore (semantic similarity) and ROUGE-1 / ROUGE-Lsum (lexical and structural overlap) between watermarked and unwatermarked outputs generated from the same prompts. Table 15 reports results under varying values of δ with $b = 32$. Across all settings, MajorMark$^+$ achieves the highest BERTScore, indicating stronger semantic preservation than MPAC and RSBH. In addition, the ROUGE metrics show that both MajorMark and MajorMark$^+$ maintain a high degree of lexical and structural similarity to the unwatermarked outputs. These results suggest that the large green list in our methods allows the model to retain its natural word choices and phrasing, rather than forcing substitutions that may degrade semantic quality.

Table 15: Semantic and lexical fidelity (BERTScore ↑ / ROUGE-1 ↑ / ROUGE-Lsum ↑) for different bias values δ with $b = 32$.

Method	$\boldsymbol{\delta = 2}$	$\boldsymbol{\delta = 4}$	$\boldsymbol{\delta = 6}$
MPAC	0.8376/0.33/0.29	0.8210/0.28/0.25	0.8105/0.26/0.22
RSBH	0.8384/0.35/0.31	0.8316/0.33/0.29	0.8234/0.30/0.26
MajorMark	0.8415/0.37/0.33	0.8305/0.33/0.28	0.8269/0.31/0.28
MajorMark$^+$	0.8449/0.36/0.32	0.8381/0.34/0.29	0.8311/0.33/0.29

F Examples of Generated Texts

Table 21 presents examples of generated text produced by different watermarking methods for a given prompt, with watermarking bias $\delta = 4$ and message length $b = 32$. As shown, both MajorMark and MajorMark$^+$ yield lower PPL, indicating better fluency in the generated text.

G False Positive Control

False positives are a practical concern in multi-bit watermarking: since the decoder always returns some message, even human-written text may be decoded into a seemingly valid message if no detection step is applied. We therefore equip both methods with a dispersion-based detection test that rejects unwatermarked text before message recovery, and calibrate its threshold on genuine human-written text.

For a candidate decoding configuration c, let $\mathbf{o}_c$ denote the shard-wise token occurrence vector reconstructed under c, and define its dispersion as $\sigma(\mathbf{o}_c) = \mathrm{std}(\mathbf{o}_c)$. Only the correct configuration reproduces the token-to-shard mappings used during encoding. Hence, watermarked text yields a markedly larger dispersion under the correct configuration than under incorrect ones, whereas human-written text yields similarly near-uniform occurrence vectors under all configurations.

- **MajorMark.** The candidate configurations are $\lambda' \in \{0, 1\}$. We define the detection score as $S = |\sigma(\mathbf{o}_0) - \sigma(\mathbf{o}_1)|$. If $S < \eta$, the decoder declares the text unwatermarked; otherwise, it proceeds with message decoding.
- **MajorMark$^+$.** For each block p, we compute S_p as the difference between the dispersion of the most dispersed configuration (λ', h') and the average dispersion of the remaining configurations. If

$\min_p S_p < \eta_+$, i.e., any block fails the test, the decoder declares the text unwatermarked; otherwise, it proceeds with block-wise message recovery.

Calibration protocol. We calibrate the thresholds using human-written text only. Specifically, we compute the detection scores on $4{,}000$ human-written records from the C4 dataset and set each threshold to the empirical 99th percentile of the scores, corresponding to a target FPR of 1%. We use message length $b = 32$, watermarking bias $\delta = 2$. This yields $\eta = 2.15$ for MajorMark and $\eta_+ = 3.30$ for MajorMark$^+$. We then freeze both thresholds and evaluate the FPR on a disjoint held-out set of $1{,}000$ human-written C4 records, and the TPR, BA, and exact-message recovery rate (EMR) on the corresponding watermarked test set.

Results. As shown in Table 16, the empirical FPRs of 0.90% and 1.20% on held-out human-written text closely match the 1% target. At these thresholds, MajorMark and MajorMark$^+$ retain TPRs of 96.40% and 95.80%, respectively, while BA remains comparable to that obtained without detection in Table 2. Thus, calibration controls false positives without degrading decoding performance. In terms of exact recovery, the fraction of watermarked texts that are both detected and attributed to the exact message is 25.50% for MajorMark and 68.66% for MajorMark$^+$. Since the thresholds are calibrated on human-written text only, they do not depend on δ. Applying the same thresholds at $\delta = 4$ and $\delta = 6$ leaves the FPR unchanged and increases the TPR to 100%, as a larger δ increases the dispersion of watermarked text.

Table 16: Detection and decoding performance at a calibrated target FPR of 1% ($b = 32$, $\delta = 2$).

Dataset	Method	FPR↓	TPR↑	BA↑	EMR↑
C4 (held-out)	MajorMark	0.90	96.40	96.87	26.45
	MajorMark$^+$	1.20	95.80	99.08	71.67
OpenGen (transfer)	MajorMark	1.30	94.80	94.74	20.89
	MajorMark$^+$	0.80	94.50	95.95	56.25

Transfer across corpora. To test whether the calibrated thresholds transfer across text distributions, we apply $\eta = 2.15$ and $\eta_+ = 3.30$ without recalibration to $1{,}000$ records from the OpenGen dataset, keeping all other configurations unchanged. The empirical FPRs remain close to the 1% target and the TPRs remain close to those on C4, supporting threshold transfer under the same configuration. BA and EMR are lower than on C4, reflecting the distribution shift between the two corpora under the same fixed detection thresholds.

When recalibration is needed. We identify two cases where the thresholds should be recalibrated. First, in low-entropy generation such as code or mathematical proofs, a small set of tokens and constructions dominates the text, producing a different shard-occupancy distribution under the null hypothesis than general text. Second, the scale of the dispersion score depends on the number of observed tokens. A threshold calibrated on longer texts may therefore be overly conservative for shorter texts, potentially reducing both FPR and TPR. In practice, we recommend calibrating the thresholds separately for each target domain and token budget.

H Ablation under a Fixed Green-List Ratio

We further investigate whether the performance improvements of our methods primarily arise from their larger green-list ratios. To remove this factor, we conduct an additional experiment using randomly generated balanced messages with $b = 64$, such that the numbers of 0s and 1s are equal. For MajorMark+, we set $r = 1$, treating the entire message as a single block. Under this setting, RSBH, MajorMark, and MajorMark+ all use the same green-list ratio of $\gamma = 0.5$. We evaluate 100 randomly generated messages, and the results are summarized in Table 17.

As shown in Table 17, the three methods achieve comparable PPL when the green-list ratio is fixed at $\gamma = 0.5$. In contrast, MajorMark improves BA over RSBH by 5.01 percentage points, while MajorMark+ further increases the improvement to 8.09 percentage points. These results show that the performance gains of our methods do not arise solely from using a larger green list. The proposed majority bit-aware encoding and the corresponding decoding strategies can provide improved decoding accuracy even when the green-list ratio is controlled.

Table 17: Performance under a fixed green-list ratio of $\gamma = 0.5$ with message length $b = 64$. Results are averaged over 100 balanced messages.

Method	BA ↑	PPL ↓
RSBH	82.58	4.72
MajorMark	87.59	4.81
MajorMark+ ($r = 1$)	**90.67**	**4.68**

Algorithm 1: The Encoding Function of MajorMark

Input : User prompt $\mathbf{x}_p$, Maximum length T, Message $\mathbf{m} \in \{0,1\}^b$, Secret key k, LLM f, Vocabulary $\mathcal{V}$, Bias strength δ
Output : Watermarked text $\mathbf{x}'_g$

$\mathbf{x}_{:-1} \leftarrow \mathbf{x}_p$
$\lambda \leftarrow \texttt{get_majority_bit}(\mathbf{m})$
for $t = 0$ **to** $T-1$ **do**
  $x_{t-1}, x_{t-2} \leftarrow \mathbf{x}_{t-1}, \mathbf{x}_{t-2}$
  $s \leftarrow \texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda)$
  $\mathcal{V}' \leftarrow \texttt{permute}(\mathcal{V}, s)$
  $\text{shard}_1, \ldots, \text{shard}_b \leftarrow \texttt{partition}(\mathcal{V}', b)$
  $\mathcal{G} \leftarrow \emptyset$
  for $i = 1$ **to** b **do**
    if $m_i == \lambda$ **then**
      Append shard_i to $\mathcal{G}$
    end
  end
  $\ell^t \leftarrow f(\mathbf{x}_{:t})$; // Get next-token logits
  for $j = 0$ **to** $|\mathcal{V}| - 1$ **do**
    if $j \in \mathcal{G}$ **then**
      $\ell^t_j \leftarrow \ell^t_j + \delta$
    end
  end
  Sample $x_t \sim \texttt{Softmax}(\ell^t)$
  Append x_t to $\mathbf{x}$
end
Return $\mathbf{x}_{0:T-1}$

I Pareto Curves of MajorMark$^+$

We plot the Pareto curves of different (r, δ) combinations in Figure 7, where the points on each curve from left to right correspond to $\delta = 2, 3, 4, 6$. Since lower PPL and higher BA are preferred, curves closer to the upper-left region indicate a better trade-off between text quality and decoding accuracy. Among all evaluated settings, $r = 2$ generally lies closer to the Pareto front. In particular, it achieves high BA with relatively low PPL when δ is small, and reaches nearly perfect BA as δ increases, while avoiding the larger PPL increase observed in some other settings. This suggests that $r = 2$ provides a better balance between watermarked text quality and message decoding accuracy.

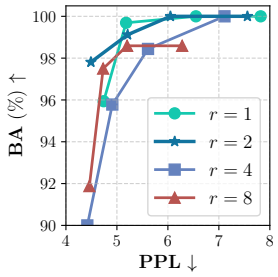


Figure 7: Pareto comparison under different (r, δ) combinations.

J Algorithms of MajorMark

Encoding. We present the detailed encoding procedure of MajorMark in Algorithm 1. The hash function $\texttt{Hash}(\cdot)$ can be implemented flexibly; in our implementation, we use the formula $(k \times x_{t-1} \times x_{t-2} + \lambda \times 31) \bmod 2^{64}$, where $k = 15{,}485{,}863$. For the permutation step $\texttt{Permute}(\cdot)$, we adopt PyTorch's built-in `torch.randperm` to generate a permutation of the vocabulary. The $\texttt{Partition}(\cdot)$ function then divides the permuted vocabulary $\mathcal{V}'$ into b equal-sized shards in order.

Decoding. The decoding procedure of MajorMark is detailed in Algorithm 2. To ensure consistency, the hash function $\texttt{Hash}(\cdot)$ used during decoding must exactly match the one used in encoding. The function $\texttt{std}(\cdot)$ computes the standard deviation of a numeric array, while $\texttt{argmax}(\cdot)$ returns the index of the maximum value in a list. We employ the $\texttt{KMeans}(\cdot)$ clustering algorithm from the scikit-learn library [29], with the number of clusters set to 2 and all other parameters set to their

Algorithm 2: The Decoding Function of MajorMark

Input : Secret key k, LLM f, Vocabulary $\mathcal{V}$, Unverified text $\mathbf{x}$ of length T
Output : Decoded message $\mathbf{m}'$
Initialize `occ` $\leftarrow$ zero matrix of shape $2 \times b$; `// Shard-wise token counts for both` λ'
for $\lambda' \in \{0, 1\}$ **do**
  for $t = 3$ **to** T **do**
    $x_{t-1}, x_{t-2} \leftarrow \mathbf{x}_{t-1}, \mathbf{x}_{t-2}$
    $s \leftarrow \texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda')$
    $\mathcal{V}' \leftarrow \texttt{permute}(\mathcal{V}, s)$
    $\text{shard}_1, \ldots, \text{shard}_b \leftarrow \texttt{partition}(\mathcal{V}', b)$
    for $i = 1$ **to** b **do**
      if $x_t \in \text{shard}_i$ **then**
        $\texttt{occ}[\lambda'][i] \leftarrow \texttt{occ}[\lambda'][i] + 1$
        break
      end
    end
  end
  $\sigma_{\lambda'} \leftarrow \texttt{std}(\texttt{occ}[\lambda'])$
end
$\lambda \leftarrow \texttt{argmax}(\sigma_0, \sigma_1)$
Cluster $\texttt{occ}[\lambda]$ into 2 groups via $\texttt{KMeans}(\cdot, 2)$
Let $\mathcal{C}$ be the cluster with higher average count
for $j = 1$ **to** b **do**
  if $j \in \mathcal{C}$ **then**
    $m_j \leftarrow \lambda$
  else
    $m_j \leftarrow 1 - \lambda$
  end
end
Return $\mathbf{m}' = (m_1, m_2, \ldots, m_b)$

default values. The returned cluster $\mathcal{C}$ is the one with the higher mean shard occurrence count. The final message is reconstructed by assigning the majority bit λ to shards in $\mathcal{C}$ and the complement bit $1-\lambda$ to the remaining shards.

K Algorithms of MajorMark$^+$

Encoding. The encoding procedure of MajorMark$^+$ is detailed in Algorithm 3. We begin by dividing the full message $\mathbf{m} \in 0, 1^b$ into r disjoint blocks using the function $\texttt{divide}(\cdot)$. For each block $\mathbf{m}_w$, the function $\texttt{get_majority_bit}(\cdot)$ returns both the majority bit λ_w and its frequency h_{λ_w}. The hash function $\texttt{Hash}(\cdot)$ can be implemented flexibly; in our implementation, we use the formula $(k \times x_{t-1} \times x_{t-2} + \lambda_p \times 31 + h_{\lambda_p} \times 97) \bmod 2^{64}$, where $k = 15{,}485{,}863$. The permutation and partition functions, denoted $\texttt{Permute}(\cdot)$ and $\texttt{Partition}(\cdot)$, respectively, are consistent with those used in the original MajorMark encoding.

Decoding. The decoding procedure of MajorMark$^+$ is outlined in Algorithm 4. Since the message $\mathbf{m}$ is partitioned into r blocks, we decode the message block by block. For each block, we consider both possible values of the majority bit hypothesis $\lambda' \in 0, 1$, along with all feasible values for its frequency $h_{\lambda'}$. Specifically, when $\lambda' = 0$, there are $b/r/2 - 1$ valid values for $h_{\lambda'}$, and when $\lambda' = 1$, there are $b/r/2$ possible values. We store token occurrence counts in a 3D matrix `occ` of shape $r \times 2 \times b/r$.

The hash function $\texttt{Hash}(\cdot)$ used during decoding must exactly match the one used in the encoding process of MajorMark$^+$. For each combination of $(\lambda', h_{\lambda'})$, we compute the standard deviation of shard-wise token occurrences. The configuration that yields the highest standard deviation is selected, and we recover the corresponding majority bit λ_p and frequency h_{λ_p} for each block $p \in [r]$. To reconstruct the message, we identify the top-h_{λ_p} shards (using `Topk`) with the highest counts in each block. The majority bit λ_p is assigned to these shards, and the remaining shards are assigned the

Algorithm 3: The Encoding Function of MajorMark$^+$

Input : User prompt $\mathbf{x}_p$, maximum generation length T, message $\mathbf{m} \in \{0,1\}^b$, secret key k, LLM f, vocabulary $\mathcal{V}$, bias strength δ, number of blocks r
Output : Watermarked text $\mathbf{x}'_g$

$d \leftarrow b/r$
Initialize $\mathbf{x}'_g \leftarrow []$
Initialize $\mathbf{x} \leftarrow \mathbf{x}_p$; `// Current context`
Divide $\mathbf{m}$ into r equal-sized blocks: $\mathbf{m}_1, \ldots, \mathbf{m}_r \leftarrow \texttt{divide}(\mathbf{m}, r)$
for $p = 1$ **to** r **do**
  $\lambda_p, h_{\lambda_p} \leftarrow \texttt{get_majority_bit}(\mathbf{m}_p)$
end
for $t = 1$ **to** T **do**
  $x_{t-1}, x_{t-2} \leftarrow \mathbf{x}_{t-1}, \mathbf{x}_{t-2}$
  $p \leftarrow ((x_{t-1} + x_{t-2}) \bmod r) + 1$; `// Select block index`
  $s \leftarrow \texttt{Hash}(k, x_{t-1}, x_{t-2}, \lambda_p, h_{\lambda_p})$
  $\mathcal{V}' \leftarrow \texttt{permute}(\mathcal{V}, s)$
  $\mathcal{V}_1, \ldots, \mathcal{V}_d \leftarrow \texttt{partition}(\mathcal{V}', d)$
  $\mathcal{G} \leftarrow \emptyset$
  for $i = 1$ **to** d **do**
    if $(\mathbf{m}_p)_i = \lambda_p$ **then**
      $\mathcal{G} \leftarrow \mathcal{G} \cup \mathcal{V}_i$
    end
  end
  $\ell^t \leftarrow f(\mathbf{x})$; `// Get next-token logits`
  for $j = 0$ **to** $|\mathcal{V}| - 1$ **do**
    if $j \in \mathcal{G}$ **then**
      $\ell^t_j \leftarrow \ell^t_j + \delta$
    end
  end
  Sample $x'_t \sim \texttt{Softmax}(\ell^t)$
  Append x'_t to $\mathbf{x}'_g$
  Append x'_t to $\mathbf{x}$
end
Return $\mathbf{x}'_g$

complement bit $1-\lambda_p$. Finally, the decoded message $\mathbf{m}'$ is obtained by concatenating all block-wise results.

L Theoretical Justification of Clustering-based Decoding

Theoretical Explanation of Clustering-Based Decoding. Under the ROM assumption, shard selections can be viewed as random draws induced by the watermarked generation distribution. By the Central Limit Theorem [7], the observed occurrence count $freq_i$ of each shard i can be approximated by a Gaussian distribution when the number of decoding tokens is sufficiently large. Under majority bit-aware encoding, for a fixed majority bit λ, shards associated with λ receive a larger expected occurrence probability than those associated with $1-\lambda$, provided that the watermark strength satisfies $\delta > 0$. Let $\Delta(\delta) > 0$ denote the resulting expected per-shard probability gap between the two groups. Let $\mathcal{S}_\lambda$ and $\mathcal{S}_{1-\lambda}$ denote the shard groups associated with λ and $1-\lambda$, respectively. We define the average occurrence counts of the two groups over the T decoding tokens as

$$\bar{C}_\lambda = \frac{1}{|\mathcal{S}_\lambda|} \sum_{i \in \mathcal{S}_\lambda} freq_i, \qquad \bar{C}_{1-\lambda} = \frac{1}{|\mathcal{S}_{1-\lambda}|} \sum_{i \in \mathcal{S}_{1-\lambda}} freq_i,$$

and define their empirical separation as $D_T = \bar{C}_\lambda - \bar{C}_{1-\lambda}$. Under the induced watermarking distribution, the expected separation grows linearly with the number of decoding tokens: $\mathbb{E}[D_T] = T \cdot \Delta(\delta)$, while its variance grows at most linearly with T: $\mathrm{Var}(D_T) = O(T)$. Therefore, the signal-to-noise ratio of the group separation satisfies $\frac{\mathbb{E}[D_T]}{\sqrt{\mathrm{Var}(D_T)}} = \Omega(\sqrt{T})$, which increases with the number of decoding tokens. This indicates that, as T becomes sufficiently large, the empirical occurrence statistics of the two shard groups become increasingly separable. Under standard concentration

Algorithm 4: The Decoding Function of MajorMark^{+}

Input : Secret key k, LLM f, Vocabulary $\mathcal{V}$, Unverified text $\mathbf{x}'_g$ of length T, Number of blocks r
Output : Decoded message $\mathbf{m}'$

```
Initialize occ ← zero matrix of shape r × 2 × (b/r)
Initialize occ[p][λ'][h'][i] ← 0
H_0 ← {1, ..., b/r/2 − 1}
H_1 ← {1, ..., b/r/2}
for λ' ∈ {0, 1} do
    for h' ∈ H_λ' do
        for t = 3 to T do
            x_{t−1}, x_{t−2} ← x_{t−1}, x_{t−2}
            s ← Hash(k, x_{t−1}, x_{t−2}, λ', h')
            V' ← permute(V, s)
            shard_1, ..., shard_{b/r} ← partition(V', b/r)
            p ← (x_{t−1} + x_{t−2}) mod r
            for i = 1 to b/r do
                if x_t ∈ shard_i then
                    occ[p][λ'][i] ← occ[p][λ'][i] + 1
                    break
                end
            end
        end
    end
end
for p = 0 to r − 1 do
    Initialize best_std ← −1, (λ_p, h_{λ_p}) ← (0, 0)
    for λ' ∈ {0, 1} do
        for h' ∈ H_λ' do
            σ ← std(occ[p][λ'][0 : b/r])
            if σ > best_std then
                best_std ← σ
                (λ_p, h_{λ_p}) ← (λ', h')
            end
        end
    end
end
Initialize m' ← []
for p = 0 to r − 1 do
    c ← occ[p][λ_p]
    T_p ← Topk(c, h_{λ_p})
    for j = 0 to b/r − 1 do
        if j ∈ T_p then
            m_{p,j} ← λ_p
        end
        else
            m_{p,j} ← 1 − λ_p
        end
    end
    Append (m_{p,0}, ..., m_{p,b/r−1}) to m'
end
Return m'
```

arguments, the probability of group-separation failure, characterized by $D_T \leq 0$, decreases toward zero. This provides an asymptotic consistency explanation for MajorMark: with sufficiently many decoding tokens, the correct shard partition can be recovered with arbitrarily high probability.

Error analysis via distribution overlap. The above consistency relies on sufficient empirical separation between the two shard groups. In finite-length generations, however, decoding errors may still occur when the empirical distributions of green and red shard counts overlap. This overlap is mainly caused by two factors. ① *Insufficient text length.* When T is small, the variance of the empirical occurrence counts is large relative to the mean gap between the two shard groups. As a result, green and red shards may not be clearly separated, leading to higher misclassification rates

and lower bit accuracy. This explains the performance drop for short outputs, as shown in Figure 4. ② *Weak watermark bias.* When δ is small, the probability gap $\Delta(\delta)$ between green and red shards becomes small. In this case, even though the separation can still improve with a larger T, more decoding tokens are required to achieve reliable recovery. With limited text length, the two groups may become nearly indistinguishable, causing `KMeans` to behave close to random partitioning. This matches our results in Table 2, where a smaller δ yields lower decoding accuracy. These analyses explain both the robustness and the finite-sample limitations of the clustering-based decoding design. We leave exact theoretical guarantees for MajorMark$^+$ as an important direction for future work.

M Analysis of Robustness under Text Editing Attacks

This section analyzes why MajorMark and MajorMark$^+$ can retain part of their watermark signal under standard text-editing attacks and paraphrasing. The stability of the two methods follows from (1) the statistical nature of the noise created by paraphrase and (2) the decoding procedures that remain effective as long as the underlying shard distributions preserve a detectable gap.

Statistical behavior of paraphrase noise. A paraphraser replaces a subset of watermarked tokens with contextually similar alternatives. These replacements produce two effects on the shard-count distribution. *1. Surviving tokens.* paraphrase does not rewrite all positions. Tokens that remain unchanged continue to vote for their correct shards, preserving part of the watermark signal. *2. Replaced tokens.* From the decoder's viewpoint, replaced tokens behave like random samples from the vocabulary. Due to the majority-bit encoding, the green-shard ratio is at least 0.5. This ensures that replacements dilute the signal rather than destroying it. In imbalanced cases where $h_\lambda > b/2$, the replacements even introduce a slight bias toward the green component. *3. Overall effect.* The paraphraser weakens the signal but does not erase it. The shard-count distribution remains biased toward the majority-bit shards, maintaining a non-zero gap that decoding algorithms can recover.

Robustness of clustering-based decoding in MajorMark. Recall that in Appendix L, the occurrence count $freq_i$ of each shard is approximated by a Gaussian distribution under the ROM assumption. The watermark bias δ increases the expected occurrence probability of green shards over red shards, creating two shard groups with different expected counts. Let $\mathcal{S}_\lambda$ and $\mathcal{S}_{1-\lambda}$ denote the shard groups associated with the majority bit λ and the minority bit $1-\lambda$, respectively. Their empirical separation can be measured by $D_T = \frac{1}{|\mathcal{S}_\lambda|}\sum_{i\in\mathcal{S}_\lambda} freq_i - \frac{1}{|\mathcal{S}_{1-\lambda}|}\sum_{i\in\mathcal{S}_{1-\lambda}} freq_i$. Under the watermarked generation distribution, $\mathbb{E}[D_T]$ grows linearly with the number of decoding tokens T, while its variance grows at most linearly with T. Thus, the signal-to-noise ratio of the group separation increases with T. Paraphrasing may reduce the effective probability gap between green and red shards, thereby decreasing the empirical separation D_T. However, as long as the reduced separation remains larger than the noise scale, the two groups of shard counts remain distinguishable. Since `KMeans` only relies on the relative separation between the two empirical groups rather than their exact mean values, it can still recover the correct shard partition when the bimodal structure is preserved. This explains why MajorMark maintains stable decoding performance under paraphrasing.

Robustness of block-wise decoding in MajorMark$^+$. MajorMark$^+$ extends MajorMark with a block-wise structure. Paraphrasing can modify the context used to compute the block index i, causing tokens to be assigned to the wrong blocks during decoding. This does not form a structural weakness for the following reasons. ① *Uniform noise across blocks.* When context tokens are altered, the resulting block index becomes random. The contribution of the replaced token is therefore spread uniformly across all blocks rather than targeted at a specific block. ② *Graceful degradation.* A wrongly assigned token does not vote against the correct block. It simply becomes noise. Because the green-shard ratio is high, the correctly synchronized tokens provide enough evidence for recovering the block message. ③ *Independence across blocks.* Each block is decoded independently. Noise affecting one block does not propagate to the others. This prevents local errors from causing global decoding failure.

Summary. Under mild paraphrasing, MajorMark and MajorMark$^+$ can retain part of the watermark signal as long as a detectable statistical gap remains between the shard distributions. This analysis does not extend to adaptive paraphrasers specifically optimized to remove watermark signals, as evaluated below.

Adaptive Watermark-Removal Attack. The above analysis focuses on standard text-editing attacks and mild paraphrasing. To further evaluate robustness against a stronger adaptive attack, we test the watermark-removal paraphraser proposed by [5], WM-Removal-KGW-Llama-2-7B, which is specifically tuned against KGW-family watermarks. We evaluate all methods on LLaMA2-7B, and report the results in Table 18.

Under this tailored attack, the BA of all methods drops to 53.22%–59.88%, approaching random guessing. This result shows that the robustness observed under standard paraphrasing does not extend to an adaptive paraphraser specifically optimized to remove KGW-style watermark signals. The limitation is not specific to majority bit-aware encoding, as all tested KGW-family methods experience substantial degradation. Improving robustness against such adaptive watermark-removal attacks remains an important direction for future work.

Table 18: Robustness under the adaptive watermark-removal paraphraser WM-Removal-KGW-Llama-2-7B. The $\pm$ values denote 95% CIs.

Method	BA ↑	PPL ↓
RSBH	$53.22_{\pm 1.98}$	$4.70_{\pm 0.15}$
MPAC	$\mathbf{59.88}_{\pm 1.81}$	$4.86_{\pm 0.14}$
MajorMark	$58.28_{\pm 1.85}$	$4.60_{\pm 0.15}$
MajorMark^{+}	$55.53_{\pm 1.99}$	$\mathbf{4.52}_{\pm 0.12}$

N Failure Analysis of Majority-Bit Recovery

MajorMark first recovers the majority bit λ by comparing the dispersion of the shard-wise token occurrence counts under the two candidate values $\lambda \in \{0, 1\}$. Since λ is also an input to the hash function, an incorrect recovery of λ produces token-to-shard mappings that differ from those used during encoding. Consequently, a single error in λ can cause correlated errors across multiple decoded bits. To evaluate this failure mode, we conduct a controlled experiment under increasingly weak watermark signals by reducing the watermarking bias from the default $\delta = 2.0$ to 1.5 and 1.0. We evaluate 100 randomly generated messages and report PPL, BA, the majority-bit failure rate (FR), defined as the fraction of messages for which λ is incorrectly recovered, and wrong-case bit accuracy (WBA), defined as the average BA among messages with incorrect λ.

As shown in Table 19, we observe no majority-bit recovery failures at $\delta \geq 1.5$. At $\delta = 1.0$, one failure occurs among 100 messages. In this failed case, BA drops to 37.50%, confirming that an incorrect λ can introduce correlated, message-wide decoding errors. However, the dispersion difference between the two candidate values of λ is only 0.19, which is substantially below the calibrated detection threshold. Therefore, this case would be rejected as an unreliable watermark rather than accepted as a confident attribution. A checksum or error-correcting code (ECC) can provide an additional safeguard against this failure mode. For example, the decoder could retain both candidate values of λ and use checksum or codeword validity to select between them, while ECC could further correct residual bit errors. We leave the joint design of majority-bit recovery and error-correcting codes to future work.

Table 19: Failure analysis of majority-bit recovery in MajorMark under weak watermarking biases. The $\pm$ values denote 95% CIs.

δ	PPL ↓	BA ↑	FR	WBA
2.0	$5.48_{\pm 0.17}$	$97.06_{\pm 0.84}$	0/100	–
1.5	$4.34_{\pm 0.12}$	$87.84_{\pm 1.66}$	0/100	–
1.0	$4.06_{\pm 0.11}$	$81.19_{\pm 2.12}$	1/100	37.50

O Impact of Message Imbalance on Decoding Accuracy

This section analyzes how the distribution of bit values in the embedded message affects the decoding behavior of MajorMark and MajorMark^{+}. In both methods, the embedded message determines the construction of green and red shards. When the message is highly imbalanced, i.e., one bit value appears much more frequently than the other, the number of green shards can increase and the green list ratio γ approaches 1. Under a fixed text length T and a fixed watermarking bias δ, this can weaken the watermark signal. Due to the shift invariance of the softmax, adding the same bias to almost all tokens leads to limited relative change in the output distribution. Meanwhile, the probability mass gained by green tokens is distributed across many green shards, reducing the expected count per green shard and weakening the separation between green and red shards.

However, this scenario also provides a compensatory advantage. When γ becomes large, the red list becomes a small fraction of the vocabulary. In this case, increasing δ mainly penalizes a small subset of red tokens and therefore causes less degradation to text quality. As a result, a larger δ can be

used to restore the separability between green and red shard distributions without severely increasing PPL. This effect applies to both MajorMark and MajorMark$^+$. In the following experiment, we use MajorMark$^+$ as a representative case to empirically demonstrate this behavior.

To verify this effect, we conduct experiments with MajorMark$^+$ under message imbalance rates h_λ/b of 71.88% and 81.25%, corresponding to 23 and 26 bits out of a $b = 32$ message matching λ, respectively. Each entry in Table 20 reports $(\text{perplexity}, \text{bit accuracy})$. As shown in Table 20, under a fixed small bias of $\delta = 2$, a higher imbalance rate leads to lower BA. Specifically, the BA decreases from 85.62% at $h_\lambda/b = 71.88\%$ to 80.16% at $h_\lambda/b = 81.25\%$. This confirms that message imbalance weakens the per-shard watermark signal when the bias is small. However, the results also show that a higher imbalance rate provides stronger tolerance to a larger δ. For example, when $\delta = 6$, both imbalance settings achieve 100.00% BA, while the 81.25% imbalance setting achieves a lower PPL of 4.91, compared with 5.90 under the 71.88% imbalance setting. Similarly, when $\delta = 8$, the 81.25% imbalance setting still maintains a relatively low PPL of 5.54 and a BA of 99.69%.

Table 20: Impact of watermarking bias δ under different message imbalance rates.

δ	$h_\lambda/b = 71.88\%$	$h_\lambda/b = 81.25\%$
2	$(4.06, 85.62)$	$(4.09, 80.16)$
4	$(4.89, 97.66)$	$(4.39, 96.88)$
6	$(5.90, 100.00)$	$(4.91, 100.00)$
8	$(6.71, 100.00)$	$(5.54, 99.69)$

These results show that message imbalance has two effects for both MajorMark and MajorMark$^+$. On the one hand, it weakens the per-shard decoding signal under a small fixed bias. On the other hand, it increases the green list ratio γ, which allows the use of a larger δ with less impact on text quality. Therefore, both MajorMark and MajorMark$^+$ can recover strong decoding accuracy under imbalanced messages by increasing δ, while still maintaining reasonable text quality.

P Broader Impacts

Our methods improve multi-bit watermarking for LLM-generated text by achieving a better balance between text quality and message decoding accuracy. This can support practical traceability in LLM services, where service providers may need to embed user identifiers, model versions, or other metadata into generated outputs. Such traceability can help with provenance tracking, intellectual property protection, and misuse investigation when generated text is leaked, redistributed, or used in harmful contexts. However, traceable watermarking also introduces potential risks. If embedded messages are directly linked to user identities, they may raise privacy concerns. In practice, watermark messages should be anonymized, encrypted, or otherwise protected. In addition, watermark detection may be affected by editing, paraphrasing, or other perturbations, and therefore should not be treated as the only evidence in high-stakes cases. Responsible deployment should include clear disclosure, privacy-aware message design, and careful interpretation of detection results.

Q Use of LLM Statement

We used LLM solely for grammar checking and polishing the writing of this manuscript.

Table 21: Generated text under different methods.

Prompt: *Sports Academy has found "The Mamba Mentality." The 96,000-square-foot training facility in Thousand Oaks announced a partnership with Kobe Bryant. The 96,000-square-foot holistic athletic training facility in Thousand Oaks announced a partnership with former Lakers superstar Kobe Bryant on Monday morning. [...]*

Method	**Generated Text**	**PPL↓**
No Watermark	short film. Bryan, who is a co-owner of the MLS team Los Angeles Football Club, has said he wants to open a youth soccer academy. He is also the co-owner of Granite Hill Capital Partners, a venture capital firm. His investment firm Easy Street has invested in more than 150 startups, according to Forbes. Sports Academy is planning to open a new location in the Valley in 2020. It's unclear if the new facility will be rebranded as Mamba Sports Academy. Sports reporter Nathaniel Percy contributed to this report. This article is written by Nathaniel Percy from Ventura County Star, Thousand Oakes and was legally licensed via the Tribune Content Agency through the NewsCred publisher network. Please direct all licensing questions to legal@newscred.com. © Copyright 2019 Ventura County Star. All rights reserved. This material may not be published, broadcast, rewritten or redistributed. Ventura County Star is the leading media company in Ventura County, California. The	3.01
MPAC	short. This article is written by Mike Coppinger from Los Angeles Daily News and was legally licensed via the Tribune Content Agency through the NewsCred publisher network. Please direct all licensing questions to . Next Post: The top 265 hotel openings, renovations set to launch in 2023 This article is really interesting. Thanks for the great post! Governors Island is opening a sports arena with artificial turf and more. Based in a city that boasts an impressive number of parks and recreation centers, Washington DC still... Tennis courts are among the popular sports amenities in resorts. Tennis courts have long been staple sports amenities in vacation homes, luxury hotels, and resorts. Today's... Leading Resort Operators Launch New Brand Campaign, 'Live the High Life' Leading resort operator Great Western Resorts has united with its parent, Highgate Hotels, as well as... Tennis court construction is easy if you are properly prepared. Tennis court surfaces come in a variety of different	8.58
RSBH	He'll host a workout and clinic at the academy this spring and will be involved at the academy through social media promotions. He's already been involved through social media promos. "As a professional athlete who has spent nearly two decades training at world-class facilities, I know the value of sports academies and the importance they have in the lives of youth," Bryant said in a statement. "Sports Academy is a state of the art, all-inclusive facility that offers unparalleled opportunities to develop the body and the mind. "My team and I are thrilled to have our brand aligned alongside the most progressive training facility in the world and the entire MAMBA team," said Faulkner in a statement. "...We look forward to this partnership, in which we can share the same values and principles with each other." Want to see more from our team at the Ventura County Star? You must be a digital subscriber to view this content. You must be registered with your Ventura County Star account to view this content! You must be signed in to your Ventura County account to	6.09
MajorMark	film for "Dear Basketball." He is also an executive producer on the "What Doesn't Kill You" Netflix series. Sports Academy also operates training facilities in Beverly Hills, Santa Barbara and Newport Beach. A fifth center is set to open in Westlake Village in early February 2020. Sports Leagues and Activities Conejo Hills Elementary School in Thousand... Conejo High School District adopts new... Thousand Oaks school district reinstates mask mandate Calabasas school district to reinstate mask mandate Firefighters battle blaze in Simi Valley; evacuations ordered in Thousand Palms Thousand Palms fire update: 5,563 acres, 10Thousand palms wildfire: 5,663 acres, no structures lost, evacuations lifted Thousand Hills Fire: Cause of wildfire under investigation; mandatory evacuations lifted for most residents Thousand-Hills Fire: Cause under investigation; mandated evacuations lifted but evacuations	5.74
MajorMark$^+$	video. This article is written by Samantha Mascarenas from The Ventura County Star and was legally licensed via the Tribune Content Agency through the NewsCred publisher network. Please direct all licensing questions to . Copyright 2019, Ventura County Star. All rights reserved. From The Los Angeles Daily News, California. Distributed by Tribune Content Agency, LLC. Want to be the first to hear what's new in the fitness space and in our industry? Subscribe to the Sweat Equity Podcast! ©Copyright 2107. All rights Reserved. FitLife Brands, Inc. "The Sweat Equality Podcast" and "Sweat Equality" is the property and trademark of FitLife Brends,Inc. All rights Resereved. You can't do it alone, let us help you. We can do the heavy lifting for you, so you'll have more time to focus on your business. Tell us what you need and we'll take care of it. You're not alone.	4.47

R Proof of the Guaranteed Green List Size of MajorMark

Assumption. We begin with an assumption on the distribution of bits in the embedded message.

Assumption 1 (Uniform Bit Distribution)**.** *Each bit in the message $\mathbf{m} \in \{0,1\}^b$ is sampled independently and uniformly at random. That is, for all $i \in \{1,\ldots,b\}$, $\Pr(m_i = 0) = \Pr(m_i = 1) = 0.5$, and $m_i \perp m_j$ for $i \neq j$.*

Preliminaries. We will make use of two theoretical tools: the De Moivre–Laplace Theorem for approximating the binomial distribution and a result on the mean absolute deviation of a normal distribution.

Theorem 3 (De Moivre–Laplace Central Limit Theorem)**.** *Let $X_n \sim \text{Binomial}(n,p)$. Then for any real numbers $z_1 < z_2$, the following convergence holds:*

$$\lim_{n\to\infty} \Pr\left(z_1 \leq \frac{X_n - np}{\sqrt{np(1-p)}} \leq z_2\right) = \int_{z_1}^{z_2} \frac{1}{\sqrt{2\pi}} e^{-z^2/2}\, dz.$$

Equivalently, the standardized variable $Z_n := \frac{X_n - np}{\sqrt{np(1-p)}}$ converges in distribution to a standard normal variable: $Z_n \xrightarrow{d} \mathcal{N}(0,1)$, as $n \to \infty$. As a consequence, for large n, the binomial distribution can be approximated as: $X_n \approx \mathcal{N}(np,\, np(1-p))$.

Proposition 1 (Mean Absolute Deviation of Normal Distribution)**.** *Let $Z \sim \mathcal{N}(\mu, \sigma^2)$. Then the expected absolute deviation from the mean is given by:*

$$\mathbb{E}[\,|Z - \mu|\,] = \sigma\sqrt{\frac{2}{\pi}}.$$

Proof. We evaluate the expectation by integrating:

$$\mathbb{E}[\,|Z - \mu|\,] = \int_{-\infty}^{\infty} |z - \mu| \cdot \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(z-\mu)^2/(2\sigma^2)}\, dz.$$

Applying the change of variable $x = \frac{z-\mu}{\sigma}$, we obtain:

$$\mathbb{E}[\,|Z - \mu|\,] = \sigma \int_{-\infty}^{\infty} |x| \cdot \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx = \sigma\sqrt{\frac{2}{\pi}},$$

which completes the proof. □

Main Proof. We now prove the expected green list size under Assumption 1.

Proof. Let $\mathbf{m} = (m_1, m_2, \ldots, m_b) \in \{0,1\}^b$ be the embedded message. During watermark encoding, the vocabulary $\mathcal{V}$ is partitioned into b equal-sized shards, each of size $|\mathcal{V}|/b$, with each shard corresponding to a bit m_i in the message. Hereafter, we assume the vocabulary size is perfectly divisible by the message length. Let h_0 and h_1 denote the number of zeros and ones in $\mathbf{m}$, respectively. Then $h_0 + h_1 = b$, and under Assumption 1, we have $h_1 \sim \text{Binomial}(b, 0.5)$.

Recall that we define the majority bit $\lambda \in \{0,1\}$ as: $\lambda = \begin{cases} 1 & \text{if } h_1 \geq h_0, \\ 0 & \text{otherwise.} \end{cases}$ The green list $\mathcal{G}$ is constructed as the union of shards whose corresponding bits equal λ. Therefore, the size of the green list is given by: $|\mathcal{G}| = \max(h_0, h_1) \cdot \frac{|\mathcal{V}|}{b}$. Defining the green list ratio as $\gamma := \frac{|\mathcal{G}|}{|\mathcal{V}|}$, we observe that:

$$\gamma = \frac{\max(h_0, h_1)}{b} \geq \frac{1}{2},$$

since the maximum of two nonnegative numbers summing to b is always at least $b/2$.

We now proceed to compute the expected value of γ, or equivalently, the expected size of $|\mathcal{G}|$. Taking expectation over all messages, we have:

$$\mathbb{E}_{\mathbf{m}}[|\mathcal{G}|] = \mathbb{E}\left[\max(h_1, b - h_1)\right] \cdot \frac{|\mathcal{V}|}{b}.$$

Noting that: $\max(h_1, b-h_1) = \frac{b}{2} + \left|h_1 - \frac{b}{2}\right|$, we obtain: $\mathbb{E}\left[\max(h_1, b-h_1)\right] = \frac{b}{2} + \mathbb{E}\left[\left|h_1 - \frac{b}{2}\right|\right]$.

By Theorem 3, for large b, the binomial variable $h_1 \sim \mathrm{Binomial}(b, 0.5)$ is approximated by a normal distribution: $h_1 \approx \mathcal{N}\left(\frac{b}{2}, \frac{b}{4}\right)$. Then, by Proposition 1, the expected absolute deviation satisfies: $\mathbb{E}\left[\left|h_1 - \frac{b}{2}\right|\right] \approx \sqrt{\frac{b}{4}} \cdot \sqrt{\frac{2}{\pi}} = \sqrt{\frac{b}{2\pi}}$.

Substituting into the previous expression yields:

$$\mathbb{E}_{\mathbf{m}}[|\mathcal{G}|] \approx \left(\frac{b}{2} + \sqrt{\frac{b}{2\pi}}\right) \cdot \frac{|\mathcal{V}|}{b} = \left(\frac{1}{2} + \frac{1}{\sqrt{2\pi b}}\right)|\mathcal{V}|.$$

Finally, we conclude that the expected green list ratio is:

$$\mathbb{E}_{\mathbf{m}}[\gamma] = \frac{\mathbb{E}_{\mathbf{m}}[|\mathcal{G}|]}{|\mathcal{V}|} \approx \left(\frac{1}{2} + \frac{1}{\sqrt{2\pi b}}\right),$$

which completes the proof. □

S Proof of the Guaranteed Green List Size of MajorMark$^+$

Proof. MajorMark$^+$ divides the message $\mathbf{m} \in \{0,1\}^b$ into r equal-sized blocks $\{\mathbf{m}_1, \ldots, \mathbf{m}_r\}$, where each block $\mathbf{m}_j$ has length b/r. For each block, a green list $\mathcal{G}_j$ is independently constructed over the full vocabulary $\mathcal{V}$ using the same *majority bit-aware encoding* rule as in MajorMark.

Let $\gamma_j := |\mathcal{G}_j|/|\mathcal{V}|$ denote the green list ratio for block j. As shown in the previous proof, for any message block of length b/r, the following lower bound always holds:

$$\gamma_j \geq \frac{1}{2}, \quad \text{for all } j \in \{1, \ldots, r\}.$$

We define the overall green list ratio γ as: $\gamma := \frac{1}{r}\sum_{j=1}^{r} \gamma_j$. As a result, we have: $\gamma \geq \frac{1}{r}\sum_{j=1}^{r} \frac{1}{2} = \frac{1}{2}$, which guarantees that MajorMark$^+$ always yields a green list containing at least half of the vocabulary.

We now proceed to analyze the expected value of γ. From Theorem 1, we know that for each block of length b/r, the expected green list ratio is:

$$\mathbb{E}_{\mathbf{m}_j}[\gamma_j] \approx \frac{1}{2} + \frac{1}{\sqrt{2\pi(b/r)}}, \quad \forall j.$$

Hence, the expected value of the overall green list ratio is:

$$\begin{aligned}\mathbb{E}_{\mathbf{m}}[\gamma] = \mathbb{E}_{\mathbf{m}}\left[\frac{1}{r}\sum_{j=1}^{r}\gamma_j\right] &= \frac{1}{r}\sum_{j=1}^{r}\mathbb{E}_{\mathbf{m}_j}[\gamma_j] \\ &\approx \left(\frac{1}{2} + \frac{1}{\sqrt{2\pi(b/r)}}\right).\end{aligned}$$

Multiplying by $|\mathcal{V}|$, we also obtain the expected green list size:

$$\mathbb{E}_{\mathbf{m}}[|\mathcal{G}|] = \mathbb{E}_{\mathbf{m}}[\gamma] \cdot |\mathcal{V}| \approx \left(\frac{1}{2} + \frac{1}{\sqrt{2\pi(b/r)}}\right)|\mathcal{V}|,$$

which completes the proof. □

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We accurately reflect our paper's contributions and scope in the abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Yes, we do. Please refer to Section 6.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Yes, we provide proofs for all theoretical results. Please refer to Appendix R–S.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We fully disclose all the information needed to reproduce the main experimental results of the paper. Please refer to Section 5.1.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We will release the full code.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Yes, we do. Please refer to Section 5.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report a statistical significance analysis over 100 users using a one-sided Wilcoxon signed-rank test. Please refer to Appendix B and Table 14.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Yes, we do. Please refer to Section 5.1.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Yes, we do.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Yes, we do. Please refer to Appendix P.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: Our work poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Yes, they are.

    Guidelines:

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

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We provide codes of our work for review.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: The main objective of this work is to study text watermarking in LLMs. Except that, the LLMs are only used for grammar checking.

Guidelines:

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