Robust Multi-Bit Watermarking for LLM-Generated Text via Semantic-Aware Bucket Encoding

Dakai Zhai[1], Kun Hu[1], Hong Zou[2], Yangdongjie Pan[3], Qianhui Zhu[1], Xingjun Wang[1,*]

[1]*Tsinghua Shenzhen International Graduate School, Tsinghua University*, Shenzhen, China
{ddk23, huk22, zhuqh23}@mails.tsinghua.edu.cn, wang.xingjun@sz.tsinghua.edu.cn

[2]*School of Computer Science and Engineering, South China University of Technology*, Guangzhou, China
202411090220@mail.scut.edu.cn

[3]*College of Information Science and Technology & Artificial Intelligence, Nanjing Forestry University*, Nanjing, China
dashuaiguo5@163.com

***Abstract*—The rapid development of Large Language Models (LLMs) has raised concerns about the authenticity of LLM-generated text, and existing multi-bit watermarking schemes have low robustness to cropping, limited capacity, and high false positives. We propose a robust framework based on clustered multi-bucket coding. First, for token embedding, we use frequency-balanced adaptive k-means to mitigate the uneven watermark distribution caused by the long-tail token frequency, which significantly improves robustness. Then, in each cluster, we utilize multiple buckets for fine-grained subdivision, forming combinations of red and green buckets, which in turn improves the embedding capacity. Finally, we employ two-stage detection with statistical validation and combinatorial decoding to achieve a low false positive rate. Experiments on OPT-1.3B, GPT-2 and Qwen3-1.7B show that our framework outperforms state-of-the-art methods in terms of robustness (92.65% accuracy after 50% cropping), capacity (96.33% at 20 bits), and a theoretical false positive rate bound (FPR $\leq 10^{-6}$), providing an effective solution for traceable LLM-generated text.**

***Index Terms*—Linguistic Watermarking, Semantic Clustering, Large Language Models, Multi-bit Embedding**

I. Introduction

The rapid evolution of Large Language Models (LLMs) such as GPT-4 [1], Qwen3 [2], and LLaMA-2 [3] has revolutionized automated content creation, achieving text fluency that is often indistinguishable from human writing [4]. However, this capability creates a critical trust gap, fostering risks such as the mass dissemination of misinformation and intellectual property infringement [5]. To mitigate these threats, linguistic watermarking has emerged as a verifiable certification mechanism by embedding imperceptible signals into generated text. Research in this field primarily follows two directions: single-bit watermarking, which focuses on binary detection, and multi-bit watermarking, which aims to embed traceable metadata (e.g., user IDs).

However, practical deployment faces a persistent trade-off among three key desiderata: robustness against attacks (e.g., cropping), capacity (payload size), and text quality (invisibility). Single-bit methods [6], [7] generally offer high robustness but lack the capacity to trace specific users or model versions. Conversely, recent multi-bit schemes often sacrifice robustness for capacity. For instance, dynamic segmentation [8] allows for scalable payloads but is fragile against text cropping, while positional encodings [9] may degrade text quality or fail under synonym substitution.

To overcome these limitations, we propose a robust multi-bit watermarking framework based on Semantic-Aware Bucket Encoding. Unlike methods that rely on rigid text segmentation, our approach binds watermark patterns to semantic clusters of tokens. This ensures that the watermark signal is uniformly distributed throughout the text, preserving retrievability even in short fragments. Our framework introduces three key innovations:

- **Multi-bit Combinatorial Bucket Encoding**: We divide the vocabulary into k red-green bucket pairs and map multi-bit messages to predefined combinatorial patterns. This breaks through the single-bit limitation of traditional red-green lists while avoiding the sparsity issues of single-green-bucket approaches.
- **Semantic Cluster Allocation**: We employ frequency-balanced adaptive k-means to group semantically similar tokens. By binding encoding patterns to these clusters, the watermark is robustly preserved against cropping and editing attacks.
- **Two-Stage Detection Mechanism**: We decouple statistical verification from payload decoding. A global z-score test first filters out unwatermarked text, significantly reducing the False Positive Rate (FPR) before the combinatorial decoding phase extracts the message.

II. Related Work

A. Single-bit Watermarking

Single-bit watermarking primarily focuses on the binary classification of text as machine-generated or human-written [10]–[12]. Foundational work by Kirchenbauer et al. [6] established the paradigm of partitioning vocabulary into "red" and "green" lists based on previous token hashes, biasing sampling towards green tokens. While effective for detection, this method cannot carry payload information. Subsequent enhancements have

* Corresponding author.

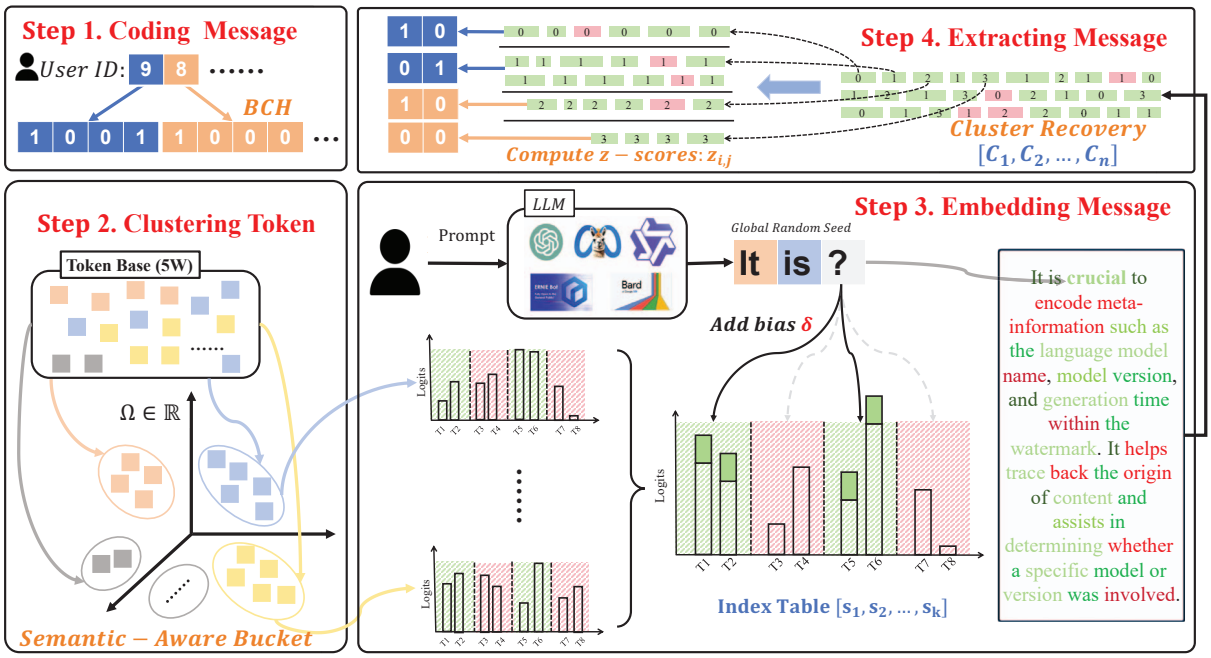

Fig. 1. The proposed framework for multi-bit watermarking. **Step 1**: Code message into groups, each mapped to a red-green bucket combination. **Step 2**: Precluster tokens into classes, each linked to a bucket combination/code. **Step 3**: Add logit bias to green bucket tokens during generation to embed the message. **Step 4**: Extract maximum z-score codes from each class and concatenate for the original message.

focused on reliability and invisibility. Fernandez et al. [7] introduced statistical hypothesis testing to guarantee low false positive rates ($< 10^{-6}$), and Cohen et al. [13] proposed adaptive keying to maintain detection consistency across multi-turn interactions. However, these methods remain fundamentally limited to origin verification and cannot support traceability scenarios requiring metadata embedding.

B. Multi-bit Watermarking

To embed richer metadata, multi-bit strategies have been developed, yet they face significant technical challenges. Early attempts by Abdelnabi et al. [14] utilized adversarial training but suffered from low text quality and robustness.

Recent approaches attempt to improve capacity through structural manipulation. Lin et al. [8] proposed dynamic segmentation for linear capacity scaling, but this dependency on segmentation makes the watermark vulnerable to cropping attacks where boundaries are lost. Yoo et al. [9] and Wang et al. [15] utilized positional encoding and vocabulary partitioning, respectively. However, these methods often compromise robustness against synonym substitution or degrade text quality for low-frequency tokens. Although error-correction schemes [16] offer theoretical robustness, they require impractical text lengths (> 100 tokens) for reliable decoding.

In summary, SOTA multi-bit schemes struggle to balance robustness, capacity, and text quality [17], [18], and remain highly vulnerable to cropping due to rigid segmentation. Our framework addresses these gaps by synergizing semantic clustering with combinatorial bucket encoding, ensuring that watermark information is resiliently distributed throughout the generated content.

III. Methodology

A. Overview

Let the user input prompt be $Pr = [s_1, s_2, \ldots, s_k]$, and the text sequence generated by the LLM be $T = \{t_1, t_2, \ldots, t_m\}$, where each token t_i is sampled from the candidate vocabulary V. We aim to implicitly embed the n-bit watermark message $W = \{w_1, w_2, \ldots, w_n\}$, $w_j \in \{0, 1\}$ into T while maintaining invisibility, robustness, and detectability. As shown in Fig. 1, our framework consists of:

Embedding stage: The vocabulary V is partitioned into n semantic clusters $\{C_i\}_{i=1}^{n}$ via frequency-balanced clustering, with each cluster C_i further divided into k buckets $\{B_{i\ell}\}_{\ell=1}^{k}$ and bound to a preset red-green combinatorial pattern $P_j = [s_1, \ldots, s_k]$ ($s_\ell \in \{R, G\}$) mapped from watermark segment w_i. During generation of token $t \in V$, the logit bias δ is dynamically added to green buckets based on t's cluster assignment to embed watermark W into the generated text $T = \{t_1, \ldots, t_m\}$.

Extraction stage: Given watermarked text T', tokens are first assigned to their semantic clusters $\{C_i\}_{i=1}^{n}$, yielding cluster-specific subsequences $\{T_i'\}_{i=1}^{n}$. For each cluster C_i, the optimal encoding pattern P_i^* is determined via hypothesis testing (z-score test [7]), where $z_{i,j}$ measures the statistical significance of pattern P_j against T_i'. The global verification metric $\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_{i,j^*}$ is then aggregated across clusters; if $\bar{z} > \tau$ (threshold), the complete watermark $\hat{W}$ is decoded by concatenating cluster-wise bits $\hat{w}_i = \text{Decode}(P_i^*)$.

B. Multi-Bit Encoding Based on Bucketing

1) Bucket Division: The candidate vocabulary V is partitioned into k disjoint buckets $\{B_i\}_{i=1}^{k}$ that satisfy three fundamental properties: they ensure **complete coverage** ($\bigcup_{i=1}^{k} B_i = V$), are **mutually exclusive** ($B_i \cap B_j = \varnothing$ for all $i \neq j$), and maintain **size balance** ($\big||B_i| - |B_j|\big| \leq 1$ for all i, j), where $|B_i|$ denotes bucket cardinality. This partitioning ensures each bucket contains either $\lfloor |V|/k \rfloor$ or $\lceil |V|/k \rceil$ tokens, thereby maintaining near-identical statistical properties across buckets for unbiased watermarking.

2) Red-Green Combination Rules: Each red-green combination is defined as a length-k sequence $P = [s_1, s_2, \ldots, s_k]$, where $s_i \in \{R, G\}$ represents the state of the i-th bucket B_i (R for red, G for green).

3) Encoding Mapping: We predefine m distinct red-green combination patterns $\{P_1, P_2, \ldots, P_m\}$, where each pattern P_j corresponds to a unique binary encoding w_j. For example, with four buckets, a 2-bit message can be mapped as follows:

$$w_j = \begin{cases} 00, & \text{if } P_j = [R, G, R, G] \\ 01, & \text{if } P_j = [R, G, G, R] \\ 10, & \text{if } P_j = [G, R, R, G] \\ 11, & \text{if } P_j = [G, R, G, R]. \end{cases} \tag{1}$$

This mapping table remains consistent during embedding and extraction. In contrast to methods like [9] that use a single green bucket, our balanced red-green pairs prevent the quality degradation and sampling sparsity issues that arise from forcing most tokens into red buckets.

4) Encoding Embedding: Firstly, we select a combination. During generation, select the corresponding red-green combination P_j based on the current watermark bit segment c_j to embed. For example, if $w_j = 01$, choose $P_2 = [R, G, G, R]$. Then, we adjust logits by adding a fixed bias δ to green buckets (G) while keeping red buckets (R) unchanged:

$$\text{logits}'(t) = \begin{cases} \text{logits}(t) + \delta & \text{if } t \in B_i \wedge P_j[i] = G, \\ \text{logits}(t) & \text{if } t \in B_i \wedge P_j[i] = R. \end{cases} \tag{2}$$

This logit adjustment strategy ensures that during text generation, green bucket tokens are sampled with higher probability, creating a statistically detectable bias in token distribution.

C. Token Clustering and Message Embedding

1) Token Clustering: First, considering that the token frequency distribution in natural language follows the long-tail law, where high-frequency tokens dominate the actual usage, we generate a sample set of approximately 100,000 tokens using the large model. We then count the occurrence frequency $\text{cnt}(t)$ of each token and compute its relative frequency as

$$f(t) = \frac{\text{cnt}(t)}{\sum_{t \in V'} \text{cnt}(t)}, \tag{3}$$

where V' is the set of tokens in the sample. Conventional k-means often produces clusters dominated by rare tokens. As shown in our later ablation study, this frequency imbalance causes the extraction rate to drop significantly. Therefore, a frequency equalization adjustment is critical. Next, we employ a frequency-balanced adaptive k-means algorithm to cluster the vocabulary V. The process begins with an **initial clustering**, where we apply the standard k-means algorithm to the token embeddings to partition the vocabulary into n initial clusters $\{C_1, C_2, \ldots, C_n\}$. Subsequently, a **frequency equalization adjustment** is performed. In this phase, we calculate the total frequency $\sum_{t \in C_i} f(t)$ for each cluster, identify the one with the highest total frequency (C_{max}) and the one with the lowest (C_{min}), and then redistribute high-frequency tokens from C_{max} to C_{min}. This is repeated until the frequency difference between any two clusters is within a preset threshold, $\epsilon = 0.01$, ensuring that the frequency distribution of all clusters meets:

$$\sum_{t \in C_i} f(t) \approx \frac{1}{n} \quad (i = 1, 2, \ldots, n). \tag{4}$$

This method ensures that each cluster has a balanced contribution of high-frequency tokens in text generation. Finally, we perform category-code binding by assigning a unique bucket pattern P_i to each frequency-balanced cluster C_i, establishing the mapping $C_i \rightarrow P_i$. This allows semantically related and frequency-balanced tokens to share the same encoding rule, ensuring uniform watermark embedding while maintaining text fluency.

2) Message Embedding: During the generation of a token t_j, its cluster C_i determines the corresponding bucketing combination P_i to use for logit adjustment. Prior methods [8], [15], [16] that rely on text segmentation are vulnerable to cropping attacks. By distributing encodings uniformly across semantic clusters throughout the text, our method ensures high robustness against such attacks. The detailed watermarking generation process is summarized in Algorithm 1.

D. Watermark Extraction

1) Encoding Selection: Using the same clustering model as in the embedding stage, we map the tokens in the text T' to clusters $C_1, \ldots, C_n$. For each cluster C_i's token sequence T'_i, we calculate a z-score for all possible red-green patterns P_j:

$$z_{i,j} = \frac{\text{Count}_G(P_j, T'_i) - \mu_j}{\sigma_j}, \tag{5}$$

where μ_j and σ_j are the expected count and standard deviation of green tokens under pattern P_j, and p_G is the green list proportion. A higher $z_{i,j}$ indicates a stronger match. Subsequently, for each cluster C_i, we select the pattern P_{j^*} with the maximum z-score:

$$j^* = \arg\max_j z_{i,j}. \tag{6}$$

The corresponding multi-bit encoding w_i is then decoded from P_{j^*} using the predefined mapping $\text{Decode}(P_{j^*})$.

2) Global Validation: To confirm the watermark's presence and reduce false positives, we compute an aggregated validation metric:

$$\bar{z} = \frac{1}{n} \sum_{i=1}^{n} z_{i,j^*}, \tag{7}$$

Algorithm 1 Watermark Embedding Process

Require: Prompt Pr, watermark W, key K, LLM M, buckets k, classes n, bias δ
Ensure: Generated text T

$\{C_1, \ldots, C_n\} \leftarrow \text{FreqBalancedClustering}(V, n, K)$
$\{w_1, \ldots, w_n\} \leftarrow \text{Split}(W, n)$
for class C_j in $\{C_1, \ldots, C_n\}$ **do**
  $P_j \leftarrow \text{MapPattern}(w_j)$ ▷ e.g., $01 \rightarrow [R, G, G, R]$
  Partition C_j into buckets $\{B_{j1}, \ldots, B_{jk}\}$
  for $l \leftarrow 1$ to k **do**
    Assign color to B_{jl} based on $P_j[l]$
  end for
end for
for step $t = 1$ to L **do**
  logits $\leftarrow M(P, t_1, \ldots, t_{t-1})$
  Identify predicted tokens' clusters and buckets
  for token $v \in V$ **do**
    if $v \in$ Green Bucket of its Cluster **then**
      logits$[v] \leftarrow$ logits$[v] + \delta$
    end if
  end for
  $t_t \leftarrow \text{Sample}(\text{logits})$
end for
return $T = t_1 \parallel \cdots \parallel t_L$

Algorithm 2 Watermark Extraction & Validation

Require: Watermarked Text T', key K, threshold τ
Ensure: Extracted Watermark $\hat{W}$ or None

Load clusters $\mathcal{C} = \{C_1, \ldots, C_n\}$ and Patterns $\mathcal{P}$ from K
Assign tokens in T' to clusters $\{T'_1, \ldots, T'_n\}$
for each cluster $C_j \in \mathcal{C}$ **do**
  Calculate z-scores $z_{j,P}$ for all patterns $P \in \mathcal{P}$
  $P_j^* \leftarrow \arg\max_{P \in \mathcal{P}} z_{j,P}$ ▷ Find best fit pattern
  $\hat{w}_j \leftarrow \text{Decode}(P_j^*)$
  $z_j^* \leftarrow z_{j,P_j^*}$
end for
$\bar{z} \leftarrow \frac{1}{n} \sum_{j=1}^{n} z_j^*$
if $\bar{z} > \tau$ **then**
  $\hat{W} \leftarrow \hat{w}_1 \| \ldots \| \hat{w}_n$
  return $\hat{W}$
else
  return None (No Watermark Detected)
end if

where z_{i,j^*} is the maximum z-score for cluster i. The watermark is confirmed present only if $\bar{z}$ exceeds a pre-calibrated threshold τ. If it does, the complete watermark $\hat{W}$ is reconstructed by concatenating the decoded segments $\{w_i\}_{i=1}^{n}$. This two-stage framework provides robust detection through statistical aggregation of local watermark evidence. The cluster-level analysis captures encoding patterns within semantically coherent regions, while the global validation integrates distributional characteristics across the entire document. The threshold τ is calibrated through Monte Carlo simulations [19] on unwatermarked text to maintain FPR below 0.5%, with reconstruction accuracy maintained even under substantial content modifications. The complete extraction and validation logic is outlined in Algorithm 2.

IV. Experiments

A. Experimental Setup

1) Models and Datasets: Our experiments use the OPT-1.3B [20] model as the baseline on the C4 dataset [21]. For capacity expansion and generalizability tests, we also use the GPT-2 model [22] and the OpenGen dataset [23]. Furthermore, to validate the effectiveness of our framework on the latest generation of instruction-tuned models, we include the SOTA Qwen3-1.7B [2].

2) Embedding Parameters: We use a logit bias $\delta = 6$ to enhance the sampling probability of green bucket tokens, balancing embedding strength and text fluency. The maximum generation length is set to 200.

3) Watermark Encoding: Our default configuration embeds 20 bits using $n_{\text{classes}} = 10$ semantic classes, with each encoding 2 bits. The scheme maps binary digits to four red-green (R/G) patterns: '00' to 'RGRG', '01' to 'RGGR', '10' to 'GRRG', and '11' to 'GRGR'. This design supports dynamic capacity expansion by adjusting the number of semantic classes.

B. Watermark Capacity Experiment

We test the framework's linear capacity scalability by measuring extraction accuracy for capacities from 8 to 20 bits on the C4 and OpenGen datasets, using OPT-1.3B, GPT-2 and Qwen3-1.7B. As shown in Table I, the 8-bit configuration exhibits optimal performance, achieving nearly 99.8% accuracy. With GPT-2 on OpenGen, it maintains an inference time of only 0.0239 seconds. As capacity increases, accuracy shows a slight decline; OPT-1.3B's accuracy drops to 96.33% at 20 bits on C4. GPT-2 is generally faster, whereas OPT-1.3B demonstrates a better accuracy-speed balance at 12 bits, achieving 97.56% accuracy in 0.0182 seconds on OpenGen.

C. Generated Content Quality Experiment

We evaluate text fluency using PPL [24], where lower values indicate more natural text, as follows:

$$PPL = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log p(t_i|t_{<i})\right). \tag{8}$$

All PPL measurements are evaluated on the OPT-1.3B model using the C4 dataset for a 20-bit watermark in approximately 200 tokens. Our framework increases PPL by only 20.3% compared to unwatermarked text, significantly outperforming established multi-bit baselines, such as the 61.2% increase from Qu et al. [16] and 52.1% from Song et al. [25]. This constrained PPL growth demonstrates that our semantic-aware clustering preserves natural token distributions while achieving an acceptable and highly competitive trade-off in text quality compared to prior multi-bit methods.

TABLE I
WATERMARK EXTRACTION ACCURACY ACROSS DIFFERENT MODELS AND CAPACITIES.

Bit	LLMs	C4 [21]		OpenGen [23]	
		Bit Acc. (%)	Time(s)	Bit Acc. (%)	Time(s)
8	OPT-1.3B	99.88	0.0416	99.60	0.0347
	GPT-2	99.83	0.0493	99.70	0.0239
	Qwen3-1.7B	99.92	0.0365	99.85	0.0288
12	OPT-1.3B	96.85	0.0969	97.56	0.0182
	GPT-2	97.67	0.0197	97.22	0.0454
	Qwen3-1.7B	97.85	0.0244	97.91	0.0210
16	OPT-1.3B	96.69	0.0345	95.92	0.0297
	GPT-2	97.47	0.0416	96.37	0.0207
	Qwen3-1.7B	97.10	0.0312	96.88	0.0255
20	OPT-1.3B	96.33	0.1336	96.50	0.0305
	GPT-2	97.00	0.0475	93.83	0.0343
	Qwen3-1.7B	96.75	0.0388	95.96	0.0319

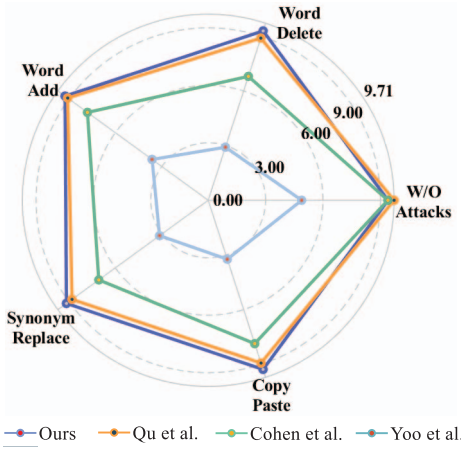

Fig. 2. A comparison of our watermark algorithm with Qu et al. [16],Yoo et al. [9], Cohen et al. [13], and our framework without error correction code under different attacks. The radial axes represent detection accuracy (1.00 unit represents 10%).

D. Robustness Experiment

We test watermark extraction accuracy against five common attacks: 10% word deletion (Word Delete), 10% word insertion (Word Add), 10% synonym substitution (Synonym Replace), 10% natural text injection (Copy Paste), and 50% text truncation (Paragraph Delete).

Fig. 2 demonstrates our framework's superior robustness, achieving high accuracy (91.59%-93.99%) across all attacks with minimal degradation ($\leq$2.6%). While the baseline shows higher no-attack performance (97.15%), it degrades significantly under attacks (87.99%-90.77%). In contrast, our method remains stable, outperforming the baseline by 4.1% in synonym replacement and 3.9% in copy-paste attacks.

Paragraph Deletion Intensity: Table II evaluates the robustness of our framework under varying paragraph deletion intensities. Remarkably, extraction accuracy remains highly stable, decreasing by only 1.34 percentage points (from 93.99% to 92.65%) even when 50% of paragraphs are removed. This minimal degradation demonstrates that semantic clustering effectively preserves watermark integrity against fragment-level attacks.

TABLE II
EXTRACTION ACCURACY VS. PARAGRAPH DELETE INTENSITY

Attack Intensity (%)	Extraction Accuracy (%)
0% (No Attack)	93.99
10%	93.29
20%	93.01
30%	92.88
40%	92.69
50%	92.65

E. False Positive Rate Analysis

We test the FPR by mixing 1000 natural and 1000 watermarked texts. By adjusting the z-score threshold $z_{\text{threshold}}$, we measure the proportion of natural texts misclassified as "watermarked." As shown in Fig. 3, we compared our 8-bit watermark detection against SOTA methods MPAC [9], CS (GreenList) [7], and MSG-HASH [15]. Our framework outperforms MPAC at low FPRs, notably achieving a 1.0 true positive rate (TPR) at an empirical FPR of 10^{-5} based on the evaluated sample size. Furthermore, the z-score hypothesis testing provides a theoretical statistical guarantee for FPR bounds below 10^{-6} under appropriate thresholding.

F. Ablation Experiments

We performed ablation studies to validate our design choices. First, we analyzed our **clustering method**. Traditional k-means, without frequency balancing, produced clusters that appeared too infrequently, leading to extraction rates below 80% (20-bit watermark, $\sim$200 tokens). In contrast, our frequency-balanced clustering ensures each cluster appears regularly, achieving 96.33% accuracy under the same conditions and showing that frequency balancing is critical for reliable embedding.

Next, we investigated the impact of the logit bias δ, which controls the sampling preference for green tokens. A small

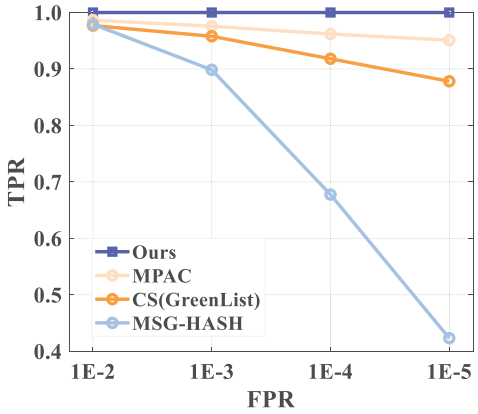

Fig. 3. TPR for various FPR thresholds.

$\delta = 3$ resulted in a weak statistical bias and lower accuracy (94.29% at 20 bits), while a large $\delta = 9$ degraded text quality (50.1% PPL increase). Our default setting of $\delta = 6$ provides an optimal trade-off, with 96.33% accuracy and only a 20.3% PPL increase, balancing embedding strength and text naturalness.

V. Conclusion

We presented a robust multi-bit watermarking framework that resolves the critical trade-offs between capacity, robustness, and text quality. Our approach synergizes frequency-balanced clustering for cropping robustness, combinatorial bucket encoding for high capacity, and a two-stage detection mechanism for an extremely low false positive rate. Experiments confirm state-of-the-art performance, with 92.65% accuracy after 50% text cropping, 96.33% detection at 20-bit capacity, and a false positive rate below 10^{-6}. Our framework provides a practical and scalable solution for traceable content from LLMs, and future work will explore learnable encoding schemes against adaptive attacks.

Acknowledgment

This work is supported by the International Science and Innovation Cooperation Project of the Ministry of Science and Technology of China (No. 2023YFE0112600), and the Sustainable Development Special Project of Shenzhen Science and Technology Innovation Commission (No. KCXFZ202002011010487).

In accordance with IEEE guidelines, we disclose that artificial intelligence(AI) [1] was utilized only for the translation of the manuscript into English.

References

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[2] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[3] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[4] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *arXiv preprint arXiv:1609.07843*, 2016.

[5] S. Tu, Y. Sun, Y. Bai, J. Yu, L. Hou, and J. Li, "Waterbench: Towards holistic evaluation of watermarks for large language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 1517–1542.

[6] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, "A watermark for large language models," in *International Conference on Machine Learning*. PMLR, 2023, pp. 17 061–17 084.

[7] P. Fernandez, A. Chaffin, K. Tit, V. Chappelier, and T. Furon, "Three bricks to consolidate watermarks for large language models," in *2023 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2023, pp. 1–6.

[8] Q. Lin, C. Tang, X. Li *et al.*, "Dermark: A dynamic, efficient and robust multi-bit watermark for large language models," *arXiv preprint arXiv:2502.05213*, 2025.

[9] K. Yoo, W. Ahn, and N. Kwak, "Advancing beyond identification: Multi-bit watermark for large language models," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2024, pp. 4031–4055.

[10] J. Kirchenbauer, J. Geiping, Y. Wen, M. Shu, K. Saifullah, K. Kong, K. Fernando, A. Saha, M. Goldblum, and T. Goldstein, "On the reliability of watermarks for large language models," *arXiv preprint arXiv:2306.04634*, 2023.

[11] Y. Lu, A. Liu, D. Yu, J. Li, and I. King, "An entropy-based text watermarking detection method," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 11 724–11 735.

[12] X. Feng, H. Zhang, Y. Zhang, L. Y. Zhang, and S. Pan, "Bimark: Unbiased multilayer watermarking for large language models," *arXiv preprint arXiv:2506.21602*, 2025.

[13] A. Cohen, A. Hoover, and G. Schoenbach, "Watermarking language models for many adaptive users," in *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2025, pp. 2583–2601.

[14] S. Abdelnabi and M. Fritz, "Adversarial watermarking transformer: Towards tracing text provenance with data hiding," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 121–140.

[15] L. Wang, W. Yang, D. Chen, H. Zhou, Y. Lin, F. Meng, J. Zhou, and X. Sun, "Towards codable watermarking for injecting multi-bits information to llms," *arXiv preprint arXiv:2307.15992*, 2023.

[16] W. Qu, W. Zheng, T. Tao, D. Yin, Y. Jiang, Z. Tian, W. Zou, J. Jia, and J. Zhang, "Provably robust multi-bit watermarking for {AI-generated} text," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 201–220.

[17] K. Yoo, W. Ahn, J. Jang, and N. Kwak, "Robust multi-bit natural language watermarking through invariant features," *arXiv preprint arXiv:2305.01904*, 2023.

[18] M. Christ, S. Gunn, and O. Zamir, "Undetectable watermarks for language models," in *The Thirty Seventh Annual Conference on Learning Theory*. PMLR, 2024, pp. 1125–1139.

[19] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo method*. John Wiley & Sons, 2016.

[20] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin *et al.*, "Opt: Open pre-trained transformer language models," *arXiv preprint arXiv:2205.01068*, 2022.

[21] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[23] K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer, "Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense," *Advances in Neural Information Processing Systems*, vol. 36, pp. 27 469–27 500, 2023.

[24] A. Liu, L. Pan, Y. Lu, J. Li, X. Hu, X. Zhang, L. Wen, I. King, H. Xiong, and P. Yu, "A survey of text watermarking in the era of large language models," *ACM Computing Surveys*, vol. 57, no. 2, pp. 1–36, 2024.

[25] M. Song, Z. Li, K. Liu, M. Peng, and G. Tian, "Nlwm: A robust, efficient and high-quality watermark for large language models," in *International Conference on Web Information Systems Engineering*. Springer, 2024, pp. 320–335.