CORE-BREW: LLR-BASED SOFT DECODING FOR ROBUST MULTI-BIT LLM WATERMARKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reliable provenance for LLM outputs requires multi-bit watermarks that remain robust under editing while maintaining low false-positive rates. Existing ECC-based LLM watermarks rely on hard-decision decoding, discarding token-level reliability information and limiting robustness under post-generation edits. We propose **CORE-BREW**, a **CO**nstant-hit-**R**ate **E**mbedding extension of BREW for multi-bit watermarking. CORE-BREW calibrates the watermark channel by targeting a fixed hit rate $p^\star$, yielding closed-form per-token log-likelihood ratios (LLRs) for soft-decision decoding. It incorporates entropy-aware erasures to limit perturbations in low-entropy contexts and combines likelihood-based scoring with soft-decision list decoding to exploit soft evidence. Experiments on open-source LLMs under token-level edits and paraphrasing demonstrate that CORE-BREW generally improves detection robustness and payload recovery over the BREW baseline while maintaining low observed false-positive rates. Despite higher conditional perplexity, BLEU and BERTScore remain close to those of unwatermarked text, indicating comparable reference-based translation quality.

1 INTRODUCTION

Large language models (LLMs) have been widely adopted for large-scale text generation in education, journalism, software development, and public policy. This rapid deployment raises critical questions regarding provenance, accountability, and safety: when a text appears online, determining its origin, algorithmic provenance, and generation context becomes essential. Watermarking has emerged as a promising mechanism to embed verifiable provenance signals at generation time Kirchenbauer et al. (2023) rather than relying solely on post-hoc classifiers Mitchell et al. (2023).

Recent work has moved beyond zero-bit "presence tests" toward *multi-bit* watermarks that embed payloads such as user identifiers or policy tags. A particularly effective line couples keyed token partitioning with error-correcting codes (ECCs) and *designated-codeword* verification, achieving strong false positive rate (FPR) control by accepting only when a pre-specified codeword is recovered in sufficiently many blocks Qu et al. (2025); Chao et al. (2024); Kim et al. (2026). In these schemes, the detector typically maps each generated token into a single bit (e.g., membership in a keyed green/red list) and applies a bounded-distance ECC decoder. This hard-decision interface is simple and provides a transparent acceptance rule, but it discards token-level reliability information, which can limit robustness under synonym substitution, insertion/deletion misalignment, or paraphrasing.

A principled approach to addressing this limitation is *soft-decision* decoding, which substantially improves decoding reliability in classical communication systems when calibrated log-likelihood ratios (LLRs) are available Lin & Costello (2004); Richardson & Urbanke (2008). However, directly integrating soft decoding into existing LLM watermarking frameworks presents fundamental challenges. Under conventional fixed logit biasing, the induced channel from hidden watermark bits to observed token categories is highly context-dependent: the probability of sampling from the intended token list varies substantially across prompts and generation steps. Consequently, a fixed token observation does not correspond to a consistent reliability level across positions, hindering principled LLR construction.

We propose **CORE-BREW**, a **CO**nstant-hit-**R**ate **E**mbedding extension of BREW that addresses this problem by explicitly shaping the watermark channel. On the embedding side, CORE-BREW introduces a *Constant Hit-Rate* rule that adaptively selects a logit offset at each generation step so

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

that the total probability mass assigned to the target token list is fixed to $p^\star \in (0.5, 1)$, independent of the local context. In the fully embedded regime, where no distortion safeguard is triggered, this calibration makes each token position behave approximately as a position-homogeneous binary symmetric channel (BSC) with crossover probability $1 - p^\star$. Consequently, each observed token category yields a closed-form per-token LLR with constant magnitude $\log \frac{p^\star}{1-p^\star}$, with its sign determined by keyed list membership, providing a principled soft-information interface for ECC-based multi-bit watermarking. Because enforcing a constant hit rate can require large logit shifts in low-entropy contexts, CORE-BREW introduces distortion guards that cap the bias magnitude and skip embedding at low-entropy or highly peaked positions. These positions are treated as *erasures* with zero LLR during detection, limiting excessive perturbations of the model distribution.

On the detection side, CORE-BREW incorporates the calibrated reliability information into the block-wise BCH and window-shifting framework of BREW Kim et al. (2026). For each candidate shift, CORE-BREW constructs bit-level LLRs and combines likelihood-based scoring with lightweight soft-decision list decoding to evaluate the block-specific designated codeword. This allows the detector, under moderate perturbation regimes, to recover the designated codeword even when the raw hard-decision representation lies outside the bounded-distance decoding sphere. Because this soft-decision rule may expand the acceptance region beyond that of BREW, we analyze the resulting acceptance rule separately and evaluate its FPR–TPR trade-off. To isolate the contribution of soft-decision decoding from Constant Hit-Rate channel shaping, we evaluate a *hard-decision ablation* that retains the same calibrated embedding and distortion safeguards but replaces likelihood-based scoring and list decoding with bounded-distance designated-codeword verification.

We evaluate CORE-BREW on open-source LLMs under clean generation, token-level substitution, insertion and deletion, and paraphrasing, comparing against multi-bit watermarking baselines Yoo et al. (2024); Qu et al. (2025); Kim et al. (2026). The results show that CORE-BREW improves low-FPR discrimination over the hard-decision BREW baseline in settings where calibrated soft evidence is beneficial, while matching unwatermarked generation in BLEU and BERTScore.

Contributions. Our main contributions are as follows. First, we propose CORE-BREW, which extends BREW with Constant Hit-Rate embedding and entropy-aware erasures to induce a calibrated watermark channel and enable closed-form per-token LLRs. Second, we combine calibrated LLR scoring with lightweight soft-decision list decoding and score-based thresholding to improve detection while controlling the FPR–TPR trade-off. Finally, across diverse models, datasets, token-level attacks, and paraphrasing, CORE-BREW consistently improves low-FPR detection performance over BREW; furthermore, ablations isolate the distinct advantage of soft decoding while generation quality (BLEU and BERTScore) remains comparable to unwatermarked text.

2 Background and Related Work

2.1 Watermarking for LLM-generated text

Text watermarking methods for large language models are broadly divided into (i) *zero-bit* schemes that test only for watermark presence and (ii) *multi-bit* schemes that embed recoverable payloads. A canonical zero-bit approach is the keyed green–red partition method of Kirchenbauer et al. Kirchenbauer et al. (2023), which biases sampling toward a secret "green" subset and detects by statistical tests on green-token counts. Subsequent work studies robustness, distributional fidelity, and detectability limits of such partition-based signals Wu et al. (2024); Zhao et al. (2024); Takezawa et al. (2023). Post-hoc detectors such as DetectGPT provide complementary evidence but do not offer keyed provenance guarantees Mitchell et al. (2023).

2.2 Multi-bit Watermarking with Error-Correcting Codes

Early designs, such as MPAC Yoo et al. (2024), demonstrate the feasibility of multi-bit payload embedding via direct token-level signaling, but their recovery reliability can degrade under editing and paraphrasing attacks. This motivates ECC-based approaches that introduce redundancy for robust payload recovery. Qu et al. Qu et al. (2025) study multi-bit watermarking with classical ECCs and a security-oriented evaluation protocol emphasizing robustness to edits, while Chao et al. Chao et al. (2024) adopt LDPC-style and window-based constructions to improve recovery on short texts.

A challenge in ECC-based multi-bit watermarking is false-positive control. If a detector accepts any decoded codeword, or relies on nearest-codeword recovery without a designated target, watermark-free text may be mapped to a valid payload, inflating the false positive rate (FPR). BREW Kim et al. (2026) addresses this issue through a block-wise designated-codeword architecture. It partitions the sequence into fixed-length blocks and assigns each block a key-dependent BCH codeword. During generation, the bits of the designated codeword are embedded through keyed token partitions, while detection maps tokens back to hard bits and applies bounded-distance BCH decoding. A block is accepted only when the decoded result matches its designated codeword, and the text is declared watermarked when a sufficient fraction of blocks are matched. BREW employs window shifting to compensate for block-boundary misalignment caused by insertion and deletion attacks. By restricting acceptance to designated codewords rather than arbitrary valid codewords, BREW provides false-positive control. However, its hard-decision decoding discards token-level reliability information, motivating the calibrated soft-information framework developed in this work.

2.3 Soft information, channel modeling, and decoding

In classical coding theory, soft-decision decoding improves reliability when calibrated log-likelihood ratios (LLRs) are available. For algebraic block codes such as BCH, Chase-type and ordered-statistics algorithms provide practical realizations of soft decoding Chase (1972); Lin & Costello (2004); Richardson & Urbanke (2008). However, most ECC-based LLM watermarking methods rely on hard decisions, mapping each token to a single bit before decoding Qu et al. (2025); Chao et al. (2024); Kim et al. (2026). Under fixed logit biasing, the induced watermark channel remains context-dependent, making principled LLR construction difficult without explicit calibration.

2.4 Attacks, paraphrasing, and evaluation protocols

Robust watermark evaluation considers both token-level edits (e.g., synonym substitution, insertion, deletion) and higher-level paraphrasing or rewriting Wieting & Gimpel (2018); Morris et al. (2020); Krishna et al. (2023). Prior work shows that sufficiently strong paraphrasing can substantially degrade watermark signals, motivating evaluation across a range of attack strengths rather than at a single operating point Wolff & Wolff (2020). From a cryptographic perspective, pseudorandom code constructions highlight inherent trade-offs between robustness and indistinguishability Christ & Gunn (2024). Together, this literature suggests that practical watermark systems should (i) analyze detection performance in terms of the FPR–TPR trade-off, (ii) characterize robustness under both token-level and semantic-level attacks, and (iii) clearly specify the acceptance criterion used to control false positives. Our evaluation follows these principles by explicitly analyzing the FPR–TPR trade-off.

3 Problem Setup & Baseline

We formulate watermark detection as a hypothesis test between watermarked (H_1) and unwatermarked (H_0) text, evaluated by the true positive rate (TPR) and false positive rate (FPR). We use BREW Kim et al. (2026) as the designated-codeword baseline. BREW partitions generated text into blocks and accepts a block only when its hard-decision representation falls within the bounded-distance decoding region of the corresponding designated BCH codeword. This acceptance rule provides strong false-positive control under the baseline assumptions but discards token-level reliability information. CORE-BREW addresses this limitation by calibrating the induced watermark channel and exploiting soft information during detection. Formal details of BREW are provided in Appendix A.

4 Proposed Method

We propose **CORE-BREW**, a **CO**nstant-hit-**R**ate **E**mbedding extension of BREW Kim et al. (2026) for robust multi-bit LLM watermarking, as illustrated in Figure 1. CORE-BREW retains BREW's block-wise designated-codeword structure while introducing Constant Hit-Rate embedding and erasure-aware LLRs for soft decoding. Likelihood-based scoring and lightweight soft-decision list decoding improve robustness while controlling the FPR–TPR trade-off through a score threshold. The detection procedure is detailed in Section 4.4.

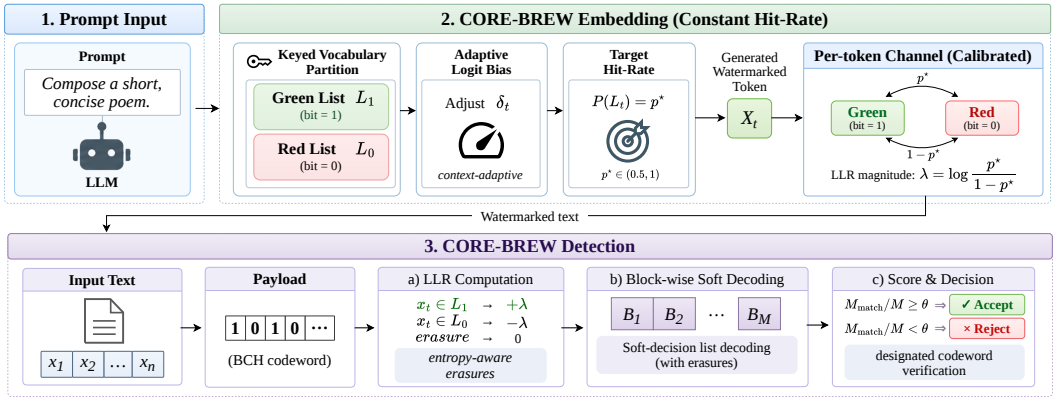

Figure 1: Overview of CORE-BREW. Adaptive logit bias maintains a target hit rate $p^\star$ over a keyed vocabulary partition, providing calibrated token-level soft information. The detector converts token observations into LLRs and performs block-wise soft decoding before the score-based decision.

4.1 Constant Hit-Rate Soft Embedding

At each generation step t, let $p_t(v)$ be the base model distribution over the vocabulary $v \in \mathcal{V}$ (after any standard sampling temperature, if used). Given the designated codeword bit $z_t \in \{0, 1\}$ at position t, we define the *target list* $L_t := L_{z_t}$ from a fixed keyed vocabulary partition (L_0, L_1). Let $m_t := \sum_{v \in L_t} p_t(v)$ be the base model probability mass assigned to the target list.

Constant hit-rate calibration. We choose a single global hit-rate target $p^\star \in (1/2, 1)$ and compute a logit offset δ_t so that the *post-bias* distribution assigns exactly $p^\star$ total mass to the target list:

$$\delta_t \;=\; \log \frac{p^\star}{1-p^\star} \;-\; \log \frac{m_t}{1-m_t}. \tag{1}$$

We then bias logits by adding δ_t to tokens in L_t:

$$\tilde{\ell}_t(v) = \begin{cases} \ell_t(v) + \delta_t, & v \in L_t, \\ \ell_t(v), & v \notin L_t, \end{cases} \qquad \tilde{p}_t(v) \;= \frac{\exp(\tilde{\ell}_t(v))}{\sum_{u \in \mathcal{V}} \exp(\tilde{\ell}_t(u))}. \tag{2}$$

This makes $\sum_{v \in L_t} \tilde{p}_t(v) = p^\star$ (see Appendix C.1). Unlike fixed logit biasing, Constant Hit-Rate fixes the target-list mass to $p^\star$, yielding a position-homogeneous reliability scale for LLR computation.

Entropy-aware erasures for distortion mitigation. In practice, when the base distribution exhibits *low entropy* (e.g., a near-deterministic next-token distribution), enforcing a fixed $p^\star$ can require excessively large logit shifts $|\delta_t|$, which degrades fluency and factuality (a known form of degeneration under distributional perturbations Holtzman et al. (2020)). To prevent this, we introduce an **entropy-aware erasure safeguard**:

- **Bias cap:** We first compute the raw constant hit-rate bias and clamp it as $\delta_t \leftarrow \text{clip}(\delta_t, -\delta_{\max}, \delta_{\max})$. We then compute the post-bias target-list mass induced by the clamped bias.
- **Erasure condition (feasibility):** If, after clamping, the induced target-list mass fails to strictly exceed $1/2$ (i.e., the embedding cannot guarantee positive correlation), we *skip embedding* at step t and treat it as an erasure. Equivalently, we set $\delta_t = 0$ and mark this position as erased.
- **Erasure condition (entropy/probability concentration):** We also skip embedding if the base distribution exhibits excessive probability concentration or low entropy, i.e., $\max_v p_t(v) \geq q_{\max}$ or $H_t < H_{\min}$.

In decoding, erased positions contribute *zero* LLR (Section 4.2), yielding a binary-symmetric channel with erasures. This limits excessive distributional perturbation in low-entropy regions while retaining erasure-aware detection.

Detector Access Assumptions. The erasure rule depends on $p_t(\cdot)$, and hence on quantities such as m_t, entropy, or peak probability. A detector that uses erasures therefore needs access to the generator or an equivalent scoring model to recompute these quantities from the observed prefix. This model-aware assumption makes CORE-BREW most suitable for provider-side verification, where such access is available. Token-only detection remains possible by disabling erasures, but it does not benefit from the entropy-aware safeguard and is not the focus of our evaluation.

4.2 Channel Modeling and Per-Token LLRs

Define the observed watermark bit at step t as $Y_t := f_K(s_t) \in \{0, 1\}$, where s_t is the sampled token and f_K indicates whether $s_t \in L_1$ under the fixed keyed partition. At non-erased, unclipped positions, Constant Hit-Rate embedding induces an approximately position-homogeneous binary symmetric channel (BSC): $\Pr(Y_t = z_t) = p^\star, \Pr(Y_t \neq z_t) = 1 - p^\star$. This justifies a principled per-token log-likelihood ratio (LLR):

$$\Lambda_t(Y_t) \;=\; \log \frac{\Pr(Y_t \mid z_t = 1)}{\Pr(Y_t \mid z_t = 0)} \;=\; \begin{cases} \log \frac{p^\star}{1-p^\star}, & Y_t = 1, \\ -\log \frac{p^\star}{1-p^\star}, & Y_t = 0. \end{cases}$$

Under attacks, these LLRs represent nominal pre-attack reliability scores rather than exact post-attack likelihoods; online estimation of attack-dependent channel parameters lies outside the scope of this pre-attack calibration model. When an erasure safeguard is triggered at step t, we set $\Lambda_t(Y_t) = 0$, ensuring that erased positions contribute no evidence.

4.3 From Token-Level to Bit-Level LLRs

Let the code length be n. For each block j and candidate shift s, we construct a bit-level reliability vector $\Lambda^{(j,s)} \in \mathbb{R}^n$ by assigning shifted token positions to BCH bit positions. Let $u_b^{(j,s)}$ denote the token index assigned to bit position b under the shifted candidate block. For each bit position, we set

$$\Lambda_b^{(j,s)} = \begin{cases} \Lambda_{u_b^{(j,s)}}\left(Y_{u_b^{(j,s)}}\right), & \text{if } u_b^{(j,s)} \text{ is valid}, \\ 0, & \text{otherwise}. \end{cases}$$

Invalid positions are treated as erasures with zero evidence. The shift s constructs the candidate block representation, while the fixed keyed partition maps each observed token to its hard bit.

4.4 Window-Shifting Soft-Decision Detection

Edits and paraphrases can alter token positions and misalign block boundaries. We therefore test candidate shifts $s \in [-s_{\max}, s_{\max}]$ for each block j. For each shifted candidate, the detector constructs the corresponding bit-level reliability vector $\Lambda^{(j,s)}$ as described in Section 4.3. For each shift s, CORE-BREW computes a block-level score for the designated codeword $c^{(j)} \in \{0, 1\}^n$:

$$S^{(j)}(s) = \sum_{b=0}^{n-1} \left(2c^{(j)}[b] - 1\right) \Lambda_b^{(j,s)}.$$

The score $S^{(j)}(s)$ measures the likelihood evidence supporting the designated codeword under the calibrated channel. CORE-BREW then applies a lightweight Chase-style list-decoding procedure Chase (1972); Lin & Costello (2004). Starting from the observed hard-decision vector, the detector identifies the least reliable bit positions according to the magnitudes of their LLRs and enumerates a small set of test error patterns over these positions. Erased positions have zero LLR and are therefore naturally treated as least reliable. Each resulting candidate is passed through the bounded-distance BCH decoder. A block is accepted only if at least one decoded candidate equals the designated codeword and the designated-codeword score satisfies $S^{(j)}(s) \geq \tau_{\text{blk}}$.

This Chase-style search can recover the designated codeword even when the raw hard decision lies outside the BCH bounded-distance radius t. The bounded-distance decoder therefore remains a decoding subroutine, while the overall CORE-BREW acceptance rule additionally incorporates calibrated soft evidence through list decoding and score-based rejection. At the text level, we aggregate block decisions by evaluating the total number of matched blocks, denoted by $M_{\text{match}} \leq M$: the text is declared watermarked if $\frac{M_{\text{match}}}{M} \geq \theta$.

4.5 Design Choices and Practical Considerations

Selection of Target Hit-Rate $p^\star$. The target hit-rate $p^\star$ controls the reliability of the induced bit channel: larger values yield larger LLR magnitudes and stronger watermark evidence, but may require larger logit shifts when m_t is far from the target. We treat $p^\star$ as a tunable parameter, analyzing detection reliability in Section 6 and detailed hit-rate/text-quality trends in Appendices D.2 and D.6.

Low-entropy safeguards. The entropy-aware erasure mechanism prevents extreme biasing when the model is near-deterministic. By turning difficult positions into erasures (LLR $= 0$), the safeguard limits distributional perturbation while allowing the decoder to ignore unreliable evidence. $\delta_{\max}$ and $H_{\min}$ (or $q_{\max}$) control the trade-off between watermark evidence and distributional distortion.

Mechanism for Low-FPR Detection Robustness. CORE-BREW improves detection through Chase-style list decoding and erasure-aware LLR scoring; beyond-radius recovery is possible but is not the only source of gain. Its acceptance rule combines bounded shift/candidate search, designated-codeword verification, and score thresholding. The score threshold τ_{blk} provides additional filtering rather than serving as a standalone FPR-control mechanism. Section 5 analyzes the acceptance rule, and Section 6 evaluates the FPR–TPR trade-off empirically.

Implementation. We compute m_t using masked softmax sums or stable log-sum-exp over the target list and its complement. Decoding scales with $(2s_{\max} + 1)$ shifts and a small Chase-style candidate list; detailed thresholds and list-decoding parameters are given in Appendix E.

5 Theoretical Analysis

Our analysis separates two notions conflated in prior multi-bit watermarking work: (i) bounded-distance acceptance on the detector's hard-decision representation, which admits designated-codeword false-positive analysis under the analysis model, and (ii) soft-decision detection using calibrated LLR information, which may expand the acceptance region and therefore requires separate analysis. Under Constant Hit-Rate embedding, non-erased positions approximately follow a position-homogeneous binary symmetric channel with target hit rate $p^\star$, enabling calibrated LLR computation and scoring. We use BREW's bounded-distance designated-codeword rule as the hard-decision reference, while CORE-BREW combines calibrated block scoring, designated-codeword verification, and limited Chase-style list decoding. Accordingly, false-positive behavior is analyzed separately for the hard- and soft-decision acceptance rules. Detailed hard-decision FPR analysis and score-based bounds for CORE-BREW are provided in Appendices C.2 and C.4, respectively.

6 Experiments

Our experiments address four questions: (Q1) clean detection performance; (Q2) robustness under token-level attacks; (Q3) robustness under paraphrasing and semantic editing; and (Q4) targeted ablations and parameter sensitivity. We use a unified evaluation pipeline based on MarkLLM Pan et al. (2024), with official or publicly available baseline implementations where applicable. Unless otherwise stated, we report mean $\pm$ standard deviation over multiple random seeds.

6.1 Experimental Setup

Models and datasets. We evaluate OPT-1.3B Zhang et al. (2022) and Mistral-7B Jiang et al. (2023) on C4 and OpenGen, where OpenGen consists of 3,000 two-sentence prompts derived from WikiText-103. Unless otherwise specified, OPT-1.3B on C4 is used as the default setting, and we generate continuations of length $T = 500$ tokens.

Schemes compared. We compare MPAC Yoo et al. (2024), Qu et al. Qu et al. (2025), BREW Kim et al. (2026), and CORE-BREW. We discuss the method of Chao et al. Chao et al. (2024) as a closely related ECC-based scheme, but do not include it in our empirical comparison because its implementation is not publicly available. We match payload budgets across the evaluated methods

Table 1: Clean detection performance on OPT-1.3B. We report mean ± standard deviation over multiple random seeds at each method's default or recommended detection threshold.

	C4		OpenGen	
Method	TPR ↑	FPR ↓	TPR ↑	FPR ↓
MPAC	1.0000 ± 0.0000	0.8333 ± 0.0239	1.0000 ± 0.0000	0.8917 ± 0.0062
Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9250 ± 0.0071	1.0000 ± 0.0000	0.9533 ± 0.0143
BREW	0.9383 ± 0.0143	0.0000 ± 0.0000	0.9650 ± 0.0108	0.0000 ± 0.0000
CORE-BREW	0.9383 ± 0.0062	0.0000 ± 0.0000	0.9817 ± 0.0085	0.0000 ± 0.0000

and reconfigure BREW to use the same BCH code and block segmentation as CORE-BREW, rather than its original BCH $(31, 6, 7)$ setting, to control for differences in ECC strength.

Code, detection parameters, and metrics. Unless otherwise stated, we use BCH parameters $(n, k, t) = (63, 7, 15)$ Lin & Costello (2004), window shifting with $s_{\max} = 10$, $p^\star = 0.9$, and model-aware detection as the default CORE-BREW configuration. A sequence is declared watermarked if $M_{\text{match}} \geq 1$. We report text-level TPR/FPR as mean ± standard deviation over multiple random seeds. Payload recovery is measured by payload bit accuracy, defined as the fraction of correctly recovered payload bits over all embedded payload bits; unrecovered bits are counted as incorrect. We report PPL, BLEU Papineni et al. (2002), and BERTScore Zhang et al. (2020) for text quality. Additional analyses of code length, window-shift range, payload size, and short-text performance are provided in Appendices D.3, D.4, D.8, and D.9. Implementation details for entropy-aware safeguards and list decoding, together with text-quality results, are provided in Appendices E and D.6.

6.2 Clean Detection Performance (Q1)

We first evaluate detection performance on clean generations, with particular attention to false positive behavior. Table 1 reports TPR and FPR on OPT-1.3B at each method's default or recommended detection threshold. MPAC Yoo et al. (2024) and Qu et al. Qu et al. (2025) achieve near-perfect TPR but exhibit substantially higher FPR. BREW Kim et al. (2026) and CORE-BREW maintain near-zero FPR while retaining high clean-detection TPR. On C4, CORE-BREW performs comparably to BREW, while on OpenGen it improves TPR from 0.9650 to 0.9817 with zero observed FPR. These results indicate that CORE-BREW preserves the conservative false-positive behavior of BREW without substantially sacrificing clean detection performance. Corresponding results on Mistral-7B are provided in Appendix D.1.

6.3 Synthetic Token-Level Attacks (Q2)

We evaluate robustness against controlled token-level perturbations applied to both watermarked and unwatermarked texts at an attack rate of $\alpha = 0.1$ on OPT-1.3B. We consider three attack categories: *token-preserving substitution*, which replaces tokens with synonyms via TextAttack-style transformations Morris et al. (2020); *token deletion*, which independently removes tokens with probability α; and *token insertion*, which adds tokens or expands them into multi-token variants. For insertion and deletion, we apply window-shifting detection with maximum offset $s_{\max}$ to compensate for block-boundary misalignment, whereas substitution preserves token positions and requires no alignment correction.

Results. Figure 2 shows that MPAC Yoo et al. (2024) and Qu et al. Qu et al. (2025) retain near-perfect TPR but exhibit consistently high FPR, whereas BREW Kim et al. (2026) and CORE-BREW maintain low observed FPR. Under substitution, CORE-BREW improves TPR over BREW on both C4 ($0.6550 \rightarrow 0.7017$) and OpenGen ($0.6583 \rightarrow 0.7217$), while both methods maintain zero observed FPR. Among the evaluated perturbations, token deletion causes the least performance degradation, with BREW and CORE-BREW achieving near-perfect TPR while maintaining FPR below 1%. Insertion remains the most challenging attack, but CORE-BREW again improves TPR over BREW on C4 ($0.4567 \rightarrow 0.5250$) and OpenGen ($0.4550 \rightarrow 0.5250$), while maintaining low FPR. Detailed results, including payload bit accuracy, are provided in Appendix D.11, with Mistral-7B results in Appendix D.10.

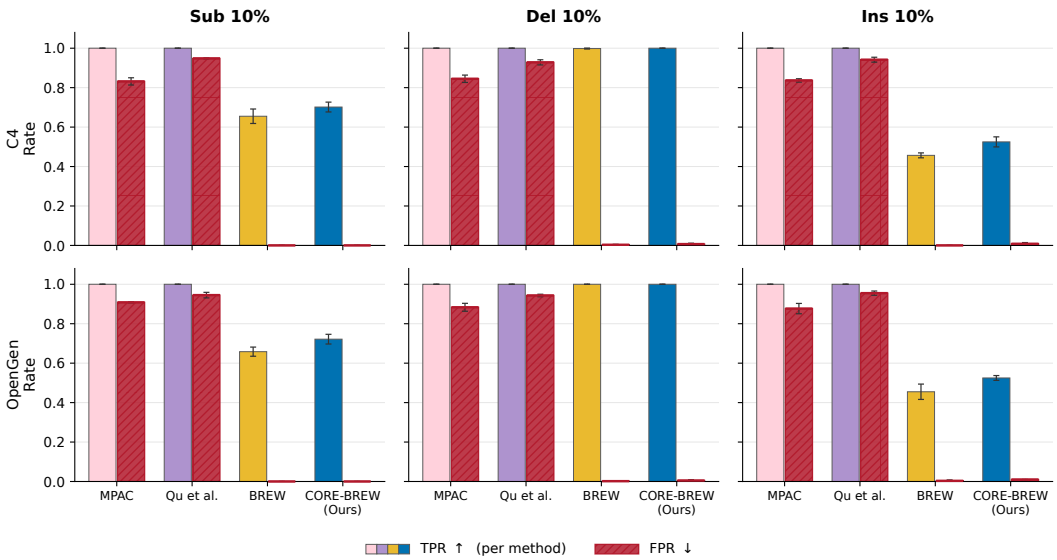

Figure 2: Detection performance under 10% substitution, deletion, and insertion attacks on OPT-1.3B using C4 and OpenGen. Colored bars denote TPR and red bars denote FPR.

6.4 Paraphrasing and Semantic Editing (Q3)

We evaluate robustness against paraphrasing attacks that modify surface form while preserving semantic content. We consider both a T5-based paraphraser Raffel et al. (2020) and DIPPER Krishna et al. (2023), providing two distinct semantic rewriting settings. Paraphrases are filtered to retain semantically related outputs with non-trivial surface variation. We evaluate continuations of length $T = 500$ on C4 and OpenGen using $\delta = 3$ for the baseline methods and $p^\star = 0.9$ for CORE-BREW. Detailed numerical results, including payload bit accuracy, are provided in Appendix D.12.

Figure 3 shows that MPAC Yoo et al. (2024) and Qu et al. Qu et al. (2025) retain near-perfect TPR but exhibit consistently high FPR across both datasets and attacks, whereas BREW Kim et al. (2026) and CORE-BREW maintain low observed FPR. Compared with BREW, CORE-BREW consistently improves TPR under both paraphrasing attacks: from 0.6850 to 0.7450 on C4 and from 0.7733 to 0.8067 on OpenGen under T5, and from 0.4850 to 0.5667 and from 0.5150 to 0.5983 under DIPPER, respectively. Across all settings, CORE-BREW maintains observed FPR below 1%, demonstrating improved robustness to both T5- and DIPPER-based paraphrasing while preserving low FPR.

6.5 Targeted Ablations and Parameter Sensitivity (Q4)

We analyze the contribution of soft-decision processing together with the effects of the target hit rate $p^\star$, window-shift budget $s_{\max}$, and entropy-aware erasures. Table 2 compares CORE-BREW with a matched hard-decision variant that uses the same Constant Hit-Rate embedding and erasure decisions, but replaces LLR-based scoring and list decoding with bounded-distance designated-codeword decoding, filling erased positions via a pseudorandom function (PRF). CORE-BREW improves TPR from 0.9067 to 0.9383 under clean generation and from 0.6617 to 0.7017 under 10% substitution, while also improving payload bit accuracy and maintaining zero observed FPR in both settings. These results show that the soft-decision configuration provides improved detection and payload recovery over the corresponding hard-decision reference under the evaluated settings.

Additional parameter studies show that detection and payload recovery improve as $p^\star$ increases and largely saturate at high hit rates, supporting our default choice of $p^\star = 0.9$ (Appendix D.2). Increasing $s_{\max}$ improves alignment robustness, particularly under insertion, but gradually raises false positives, supporting $s_{\max} = 10$ as a practical trade-off (Appendix D.4). Entropy-aware erasures substantially reduce conditional perplexity with only modest changes in TPR, supporting their role in distortion control (Appendix D.7).

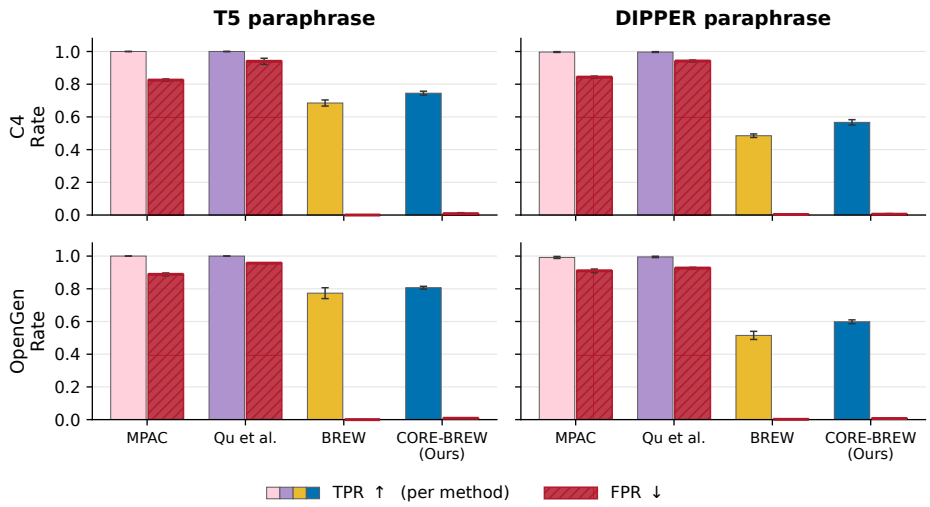

Figure 3: Detection performance under T5 and DIPPER paraphrasing attacks on OPT-1.3B at $T = 500$ using C4 and OpenGen. Colored bars denote TPR and red bars denote FPR.

Table 2: Ablation of soft-decision decoding on OPT-1.3B with C4 at $p^\star = 0.9$.

Attack	Method	TPR	FPR	Payload Bit Acc.
Clean	CORE-BREW w/o Soft Decoding	0.9067 ± 0.0062	0.0000 ± 0.0000	0.8983 ± 0.0103
	CORE-BREW	0.9383 ± 0.0062	0.0000 ± 0.0000	0.9308 ± 0.0042
10% Substitution	CORE-BREW w/o Soft Decoding	0.6617 ± 0.0417	0.0000 ± 0.0000	0.5367 ± 0.0283
	CORE-BREW	0.7017 ± 0.0249	0.0000 ± 0.0000	0.5492 ± 0.0120

6.6 Limitations and Threat Model

We focus on provider-side provenance verification, where the detector retains the secret key. A token-only setting uses the observed text, key, and designated codeword schedule without model probabilities, whereas model-aware detection additionally accesses the generator or an equivalent scoring model to reapply the entropy-aware erasure rule to the observed prefix. Under text perturbations, this prefix may differ from that used during generation, so detection-time erasure decisions need not match those made during embedding. We consider token-level edits and moderate paraphrasing, but do not claim robustness to arbitrary semantic rewriting or white-box, key-aware attacks. Model-aware detection incurs additional cost from recomputing token probabilities and erasure decisions, which could be reduced using lightweight proxy models. CORE-BREW also exhibits higher conditional perplexity than unwatermarked generation and the evaluated baselines, although BLEU and BERTScore remain close to those of unwatermarked generation (Appendix D.6). Improving this likelihood–robustness trade-off, for example through adaptive control of $p^\star$, remains an important direction for future work.

7 Conclusion

We present **CORE-BREW**, a block-wise multi-bit text watermarking framework combining Constant Hit-Rate embedding, entropy-aware erasures, calibrated LLR scoring, soft-decision decoding, and window shifting for robust detection under text modifications. Across token-level attacks and paraphrasing, CORE-BREW generally improves detection and payload recovery over BREW while maintaining low false-positive rates, with ablations showing gains over the hard-decision variant. The framework also supports payload scaling and shorter generations via adaptive BCH configurations. Although CORE-BREW incurs higher model-likelihood distortion, BLEU and BERTScore remain close to those of unwatermarked generation. Overall, calibrated soft information improves the robustness and flexibility of multi-bit LLM watermarking while preserving practical false-positive control.

8 AI USE STATEMENT

Generative AI tools were used solely to assist with language editing, including grammar checking, translation, and refinement of wording for clarity and readability. All technical content, methodology, experiments, results, and conclusions were developed and verified by the authors. The authors take full responsibility for the final content of this work.

9 REPRODUCIBILITY STATEMENT

We provide the methodological and experimental details required to reproduce CORE-BREW throughout the main paper and appendices. The complete embedding and detection procedures, including Constant Hit-Rate embedding, entropy-aware erasures, LLR construction, window shifting, and soft-decision list decoding, are described in Section 4 and Appendix B. Experimental models, datasets, BCH parameters, attack settings, evaluation metrics, and default hyperparameters are specified in Section 6, with additional implementation details provided in Appendix E. Parameter sensitivity, component ablations, payload scaling, short-text evaluation, and additional model results are reported in the supplementary experiments. Theoretical assumptions and proofs are provided in Appendix C. We report results over multiple random seeds as mean $\pm$ standard deviation to characterize experimental variability. An anonymized implementation and configuration files are provided in the supplementary material.

REFERENCES

Patrick Chao, Yan Sun, Edgar Dobriban, and Hamed Hassani. Watermarking language models with error correcting codes. *arXiv preprint arXiv:2406.10281*, 2024.

D. Chase. Class of algorithms for decoding block codes with channel measurement information. *IEEE Transactions on Information Theory*, 18(1):170–182, 1972. doi: 10.1109/TIT.1972.1054746.

Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In Leonid Reyzin and Douglas Stebila (eds.), *Advances in Cryptology – CRYPTO 2024*, pp. 325–347, Cham, 2024. Springer Nature Switzerland.

Oded Goldreich. *Foundations of Cryptography, Volume 1: Basic Tools*. Cambridge University Press, 2001. ISBN 0-521-79172-3.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. URL https://doi.org/10.1080/01621459.1963.10500830.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR 2020)*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

David Hunter. An upper bound for the probability of a union. *Journal of Applied Probability*, 13(3): 597–603, 1976.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Joeun Kim, HoEun Kim, Dongsup Jin, and Young-Sik Kim. Block-wise codeword embedding for reliable multi-bit text watermarking. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026. URL https://arxiv.org/abs/2605.00348.

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

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 (NeurIPS 2023)*, 36:27469–27500, 2023.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Eduardo Blanco and Wei Lu (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012/.

Shu Lin and Daniel J. Costello. *Error Control Coding: Fundamentals and Applications*. Prentice Hall, Upper Saddle River, NJ, 2 edition, 2004. ISBN 0-13-042672-6. URL https://www.pearson.com/en-us/subject-catalog/p/error-control-coding/P200000003536/9780130426727.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, pp. 24950–24962. PMLR, 2023.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In Qun Liu and David Schlangen (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 119–126, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.16. URL https://aclanthology.org/2020.emnlp-demos.16/.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, Irwin King, and Philip S. Yu. MarkLLM: An open-source toolkit for LLM watermarking. In Delia Irazu Hernandez Farias, Tom Hope, and Manling Li (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP 2024)*, pp. 61–71, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-demo.7. URL https://aclanthology.org/2024.emnlp-demo.7/.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin (eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

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. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 201–220, 2025.

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.

Tom Richardson and Rudiger Urbanke. *Modern coding theory*. Cambridge university press, 2008.

Yuki Takezawa, Ryoma Sato, Han Bao, Kenta Niwa, and Makoto Yamada. Necessary and sufficient watermark for large language models. *arXiv preprint arXiv:2310.00833*, 2023.

John Wieting and Kevin Gimpel. ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 451–462, 2018. URL https://arxiv.org/abs/1711.05732.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Max Wolff and Stuart Wolff. Attacking neural text detectors. *arXiv preprint arXiv:2002.11768*, 2020.

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. A resilient and accessible distribution-preserving watermark for large language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53443–53470. PMLR, 2024.

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

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

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

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for AI-generated text. In *Proceedings of the 12th International Conference on Learning Representations (ICLR 2024)*, 2024.

A Baseline Full Details

A.1 Block-wise Designated-Codeword Watermarking

This appendix summarizes the block-wise designated-codeword watermarking framework underlying BREW Kim et al. (2026), along with relevant ECC-based components used in prior multi-bit watermarking schemes Qu et al. (2025); Chao et al. (2024). The framework combines keyed token partitions with error-correcting codes (ECCs) and accepts a block only when the decoded codeword matches its designated codeword. While the framework can be instantiated with different block codes, in this work we use a binary BCH code with parameters $(n, k, t) = (63, 7, 15)$, as discussed in Appendix D.3.

Scope of this appendix. This appendix describes the generic block-wise designated-codeword framework used as the baseline abstraction and for the hard-decision false-positive analysis. The analysis adopts an idealized detector-facing bit model to make the acceptance-region argument explicit. Our evaluated CORE-BREW instantiation in Appendix B, however, uses a fixed keyed vocabulary partition shared across positions, keeping the token-to-bit mapping invariant across candidate shifts under insertion/deletion attacks. Accordingly, the analysis below should be interpreted as an idealized designated-codeword model rather than as an exact independence guarantee for the evaluated fixed-partition implementation.

Code and blocks. Let $C \subseteq \{0,1\}^n$ be a binary BCH code of length n, dimension k, and minimum Hamming distance $d_{\min}$. Its unique-decoding radius under bounded-distance decoding is

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor .$$

Generation positions after the prompt are partitioned into blocks of length n. Let $\tau \geq N_p$ denote an absolute generation position and define its continuation-relative position as

$$r = \tau - N_p.$$

The corresponding block index and within-block offset are

$$j = \left\lfloor \frac{r}{n} \right\rfloor, \qquad b = r \bmod n,$$

where $j \in \{0, 1, \ldots\}$ is the block index and $b \in \{0, \ldots, n-1\}$ is the within-block offset.

Designated codewords. For each block j, the embedder and detector share a designated codeword $c^{(j)} \in C$, derived from the payload and the secret key. A typical construction applies a per-block pseudorandom mask before encoding to balance codeword weights and avoid fixed patterns Qu et al. (2025); Kim et al. (2026). During detection, the decoded block is accepted only when it matches the designated codeword $c^{(j)}$ for that block; arbitrary valid decoded codewords are not accepted.

Keyed token partitions. A generic designated-codeword construction may use block-dependent keyed vocabulary partitions. For each block j, the vocabulary $\mathcal{V}$ is partitioned into two disjoint lists $L_0^{(j)}$ and $L_1^{(j)}$ ("red/green") using a cryptographic hash seeded by (K, j):

$$L_0^{(j)} = \{v \in \mathcal{V} : H(K \| j \| v) \bmod 2 = 0\},$$
$$L_1^{(j)} = \{v \in \mathcal{V} : H(K \| j \| v) \bmod 2 = 1\}.$$

Here, $H(\cdot)$ is modeled as a pseudorandom mapping for analysis Goldreich (2001); Kirchenbauer et al. (2023); Qu et al. (2025). Membership in these lists induces a deterministic mapping

$$f_j : \mathcal{V} \to \{0, 1\},$$

which is used during both embedding and detection. Appendix B specifies the fixed-partition instantiation used in our evaluated CORE-BREW implementation.

A.2 Baseline Embedding with Fixed Logit Bias

At each generation position τ in block j with within-block offset b, the designated code bit to embed is

$$z_\tau = c^{(j)}[b] \in \{0, 1\}.$$

Let $\ell_\tau(v)$ denote the base-model logit for token $v \in \mathcal{V}$. Under the block-dependent baseline notation used in this appendix, the target list is $L_{z_\tau}^{(j)}$. The baseline applies a fixed logit bias $\delta > 0$ to tokens in this target list:

$$\tilde{\ell}_\tau(v) = \begin{cases} \ell_\tau(v) + \delta, & v \in L_{z_\tau}^{(j)}, \\ \ell_\tau(v), & \text{otherwise.} \end{cases}$$

This follows the partition-based biasing paradigm Kirchenbauer et al. (2023) adapted to BREW's block-wise designated-codeword setting Kim et al. (2026). Because the same bias δ is applied regardless of the base probability mass assigned to the target list, the resulting target-hit probability varies with context. Consequently, the induced bit channel is generally context-dependent and uncalibrated, limiting the use of principled LLR-based decoding. This limitation motivates the Constant Hit-Rate embedding introduced in CORE-BREW.

A.3 Baseline Detection with Window Shifting and Bounded-Distance Decoding

Given a token sequence containing a prompt of length N_p followed by the generated continuation, the detector maps continuation tokens to hard bits using the keyed vocabulary partition. For block j and within-block offset b, the corresponding absolute token position is

$$\tau = N_p + jn + b,$$

and the detector sets

$$B^{(j)}[b] = f_j(s_\tau) \in \{0, 1\},$$

yielding a hard-decision block vector $B^{(j)} \in \{0, 1\}^n$.

Window shifting for insertions/deletions. Insertions and deletions can shift the observed token positions relative to the expected block boundaries. To compensate for such local misalignment, the detector searches over a bounded set of offsets around the nominal block anchor $N_p + jn$. Let $s_{\max} \in \mathbb{N}$ and define

$$\mathcal{S} = \{-s_{\max}, \ldots, -1, 0, 1, \ldots, s_{\max}\}, \qquad S = |\mathcal{S}| = 2s_{\max} + 1.$$

For each $s \in \mathcal{S}$, the detector constructs a shifted candidate block $B^{(j,s)} \in \{0,1\}^n$ as

$$B^{(j,s)}[b] = f_j\left(s_{N_p+jn+s+b}\right), \qquad b = 0, \ldots, n-1.$$

Bounded-distance BCH decoding and designated-codeword verification. For each shift $s \in \mathcal{S}$, the detector applies a bounded-distance BCH decoder to the candidate block vector $B^{(j,s)}$. Let

$$\hat{c}_s^{(j)} = \mathrm{Dec}_{\mathrm{BCH}}\left(B^{(j,s)}\right)$$

denote the decoder output. Block j is counted as a match if at least one tested shift decodes exactly to its designated codeword:

$$\exists s \in \mathcal{S} \quad \text{s.t.} \quad \hat{c}_s^{(j)} = c^{(j)}.$$

Thus, the detector does not accept arbitrary valid BCH codewords; acceptance requires equality with the designated codeword assigned to the block.

Over M evaluated blocks, let M_{match} denote the number of matched blocks and define the match ratio

$$\rho = \frac{M_{\mathrm{match}}}{M}.$$

The text is declared watermarked if

$$\rho \geq \theta,$$

where $\theta \in (0, 1]$ is the text-level detection threshold.

A.4 Baseline False Positive Behavior and Analysis

The designated-codeword architecture provides a transparent FPR control mechanism under the standard analysis model. Under H_0 and pseudorandom partition assumptions, we model the detector-facing candidate block vector as approximately uniform over $\{0,1\}^n$ Qu et al. (2025); Kim et al. (2026). For a fixed alignment, the probability that such a random block vector lies within Hamming distance at most t of a fixed designated codeword is

$$p_0 = \frac{V_2(n,t)}{2^n}, \qquad V_2(n,t) = \sum_{i=0}^{t} \binom{n}{i},$$

the volume of a Hamming ball in $\{0,1\}^n$ Lin & Costello (2004). With window shifting, a union bound gives

$$p_0^{(\mathrm{shift})} \leq S\, p_0.$$

For aggregate text-level bounds, we use an idealized approximate block-independence model. Under this approximation, the number of spurious matches under H_0 can be modeled as a binomial random variable with success probability $p_0^{(\mathrm{shift})}$. This gives the standard Chernoff-style intuition that, when $\theta > p_0^{(\mathrm{shift})}$, the aggregate FPR decays exponentially with the number of blocks M Lin & Costello (2004). For the fixed-partition instantiation in Appendix B, this should be interpreted as an idealized analysis tool complemented by empirical FPR evaluation on unwatermarked text, since repeated tokens and natural-language correlations can induce dependencies across positions and blocks.

Baseline limitations (motivating calibrated soft information). The baseline's guarantees are tied to its *acceptance region*: it accepts a block only when the detector-facing hard-decision vector falls within the bounded-distance decoding neighborhood of the designated codeword. This yields combinatorial-style FPR control under the analysis model, but it also (i) discards token reliability information and (ii) operates over a context-dependent, uncalibrated induced channel due to fixed bias δ. These limitations motivate the channel-calibrated embedding and LLR-based soft decoding introduced in CORE-BREW.

B Algorithmic Details

This appendix provides concrete algorithmic descriptions of the main components of CORE-BREW. It follows the block-wise designated-codeword framework underlying BREW Kim et al. (2026), together with relevant ECC-based components from prior multi-bit watermarking methods Qu et al. (2025); Chao et al. (2024). CORE-BREW combines Constant Hit-Rate embedding with erasure-aware LLR construction, likelihood-based scoring, and erasure-aware Chase-style list decoding. We additionally describe a hard-decision ablation that uses the same calibrated embedding and erasure decisions but replaces LLR-based scoring and list decoding with bounded-distance designated-codeword verification on a PRF-filled hard-decision representation. Both variants use a fixed keyed vocabulary partition and block-specific designated BCH codewords determined by the embedded payload and secret key.

B.1 Keyed Partitions and Codeword Assignment

We use the notation and basic setup introduced in Sections 3–4. Let $\mathcal{V}$ denote the vocabulary, K the secret key shared by the embedder and detector, and $C \subseteq \{0,1\}^n$ a binary BCH code of length n and dimension k. We use a cryptographic hash function H, modeled as a random oracle for analysis Goldreich (2001).

Fixed keyed partitions. In the implementation used for our experiments, we use a single keyed vocabulary split that is shared across all block positions. Given a secret key K, we define

$$L_0 = \{v \in \mathcal{V} : H(K\|v) \bmod 2 = 0\}, \qquad L_1 = \{v \in \mathcal{V} : H(K\|v) \bmod 2 = 1\}.$$

This follows the keyed green–red paradigm Kirchenbauer et al. (2023), but unlike position-dependent partition schedules, the token-to-bit mapping is independent of the bit offset.

This fixed mapping keeps the token-to-bit assignment invariant across candidate shifts, because the same token maps to the same hard bit even when its position changes.

Designated codeword assignment. For each block j, the assigned payload chunk $m^{(j)}$ is encoded as $c^{(j)} = \mathrm{Encode}_C(m^{(j)})$. Under the default schedule, payload chunks are deterministically selected from the secret key and block index; an explicit payload schedule may instead be supplied. The resulting block-specific codeword serves as the designated target during both embedding and detection. During detection, each reconstructed block is verified specifically against its assigned codeword $c^{(j)}$; arbitrary valid BCH codewords are not accepted. This block-dependent assignment supports multi-bit payload embedding while retaining designated-codeword verification.

B.2 Constant Hit-Rate Soft Embedding with Distortion-Aware Erasures

Algorithm 1 summarizes online embedding for a single sequence. At each generation step, it determines the current block and code-bit offset, computes the base target-list mass m_t, and chooses a Constant Hit-Rate bias subject to distortion guards. If the clamped bias cannot raise the target-list mass above $1/2$, if the base distribution is too peaked, or if the base entropy is below a minimum threshold, watermark biasing is skipped and the position is marked as an erasure. This limits excessive distributional perturbation and supports the calibrated BSC approximation used in Section 5. When model-aware erasures are enabled, the detector reapplies the same erasure rule through $\mathrm{EraseCheck}$ in Algorithm 3 using the observed prefix, assigning zero LLR to positions that satisfy the distortion safeguards at detection time.

Relation to BREW's fixed-bias embedding. BREW uses a fixed logit bias δ rather than the adaptive Constant Hit-Rate bias δ_t. As a result, the target-list probability varies with context rather than being calibrated to a fixed target hit rate $p^\star$.

Algorithm 1 Constant Hit-Rate block-wise embedding with distortion-aware erasures

Input: prompt tokens $(s_0, \ldots, s_{N_p-1})$, secret key K, BCH code C, block-level payload chunks $\{m^{(j)}\}$, block length n, target hit-rate $p^\star$, language model LM, bias cap $\delta_{\max}$, peak threshold $q_{\max}$, entropy threshold $H_{\min}$, numerical clamp ε
Output: watermarked continuation tokens $(s_{N_p}, \ldots, s_{N_p+T-1})$
Construct the keyed fixed partition L_0, L_1 from K
for $t = 0$ **to** $N_p - 1$ **do**
  Emit prompt token s_t without modification
end for
for $t = N_p$ **to** $N_p + T - 1$ **do**
  $(j, b) \leftarrow \mathrm{divmod}(t - N_p, n)$
  $c^{(j)} \leftarrow \mathrm{Encode}_C(m^{(j)})$
  $z_t \leftarrow c^{(j)}[b]$
  $L_t \leftarrow L_{z_t}$
  Query LM to obtain logits $\ell_t(v)$
  $p_t(v) \leftarrow \mathrm{softmax}(\ell_t)(v)$
  $p_{\max} \leftarrow \max_v p_t(v)$
  $H_t \leftarrow -\sum_{v \in \mathcal{V}} p_t(v) \log p_t(v)$
  $m_t \leftarrow \sum_{v \in L_t} p_t(v)$
  $m_t \leftarrow \min(\max(m_t, \varepsilon), 1 - \varepsilon)$
  $\delta_t^{\mathrm{raw}} \leftarrow \log \frac{p^\star}{1-p^\star} - \log \frac{m_t}{1-m_t}$
  $\delta_t \leftarrow \mathrm{clip}(\delta_t^{\mathrm{raw}}, -\delta_{\max}, \delta_{\max})$
  $\widetilde{m}_t \leftarrow \dfrac{e^{\delta_t} m_t}{e^{\delta_t} m_t + (1 - m_t)}$
  if $p_{\max} \geq q_{\max}$ **or** $H_t < H_{\min}$ **or** $\widetilde{m}_t \leq 1/2$ **then**
    $\delta_t \leftarrow 0$
    Mark position t as erased
  end if
  Bias logits toward L_t using δ_t
  Sample s_t from the biased distribution
end for

B.3 Bit Sequence Extraction and Block Reconstruction

During detection, observed tokens are first mapped back to bit sequences using the same fixed keyed partition derived from K. Algorithm 2 summarizes this mapping for the unshifted case. For each complete block, the detector reuses the fixed partition (L_0, L_1) and maps each token to a hard bit according to its membership in L_1. The resulting vector $B^{(j)} \in \{0, 1\}^n$ provides the token-induced hard-decision representation. At non-erased positions, these bits agree with the signs of the CORE-BREW LLRs, while erased positions are handled separately by the corresponding decoder. In the hard-decision ablation, erased positions are filled with key-derived PRF bits before bounded-distance decoding. In CORE-BREW, the same token-to-bit observations provide the initial hard decisions for soft decoding, while the associated LLRs encode their reliability. Shifted candidates in Algorithm 3 use the same token-to-bit mapping and are evaluated within the allowed shift budget.

Algorithm 2 Bit sequence extraction and block reconstruction

Input: text tokens $(s_0, \ldots, s_{N-1})$ prompt length N_p, secret key K, block length n
Output: complete block-wise bit sequences $B^{(0)}, \ldots, B^{(M-1)} \in \{0,1\}^n$
$U \leftarrow N - N_p$ {number of post-prompt positions}
$M \leftarrow \lfloor U/n \rfloor$ {number of complete blocks}
if $M \leq 0$ **then**
  return $[\,]$
end if
Construct the fixed keyed partition (L_0, L_1) from K
for $j = 0$ **to** $M - 1$ **do**
  Initialize $B^{(j)}$ as an array of length n
  for $b = 0$ **to** $n - 1$ **do**
    $t \leftarrow N_p + jn + b$
    $B^{(j)}[b] \leftarrow \mathbb{I}[s_t \in L_1]$
  end for
end for
return $(B^{(0)}, \ldots, B^{(M-1)})$

B.4 LLR Computation and Window-Shifting Detection

Algorithm 3 summarizes the window-shifting detection procedure for CORE-BREW and its hard-decision ablation. On non-erased positions, the calibrated hit-rate model assigns the constant LLR magnitude

$$\lambda = \log\left(\frac{p^\star}{1 - p^\star}\right).$$

CORE-BREW uses these LLRs through score-based acceptance and limited Chase-style list decoding, with erased positions assigned zero reliability. To evaluate the contribution of soft-decision processing, the hard-decision ablation uses the same Constant Hit-Rate embedding and erasure decisions, but fills erased positions with key-derived PRF bits and applies bounded-distance designated-codeword verification instead of LLR-based scoring and list decoding.

Here, $\mathrm{ConstructObservation}$ maps observed tokens to hard bits and reapplies the distortion-aware erasure rule through $\mathrm{EraseCheck}$ on the observed prefix. Erased positions retain their token-derived hard bits but are assigned zero LLR.

B.5 CORE-BREW Soft-Decision Decoding Wrapper

CORE-BREW uses a score-thresholded Chase-style list-decoding wrapper. Erased positions remain zero-LLR entries for likelihood scoring, while the list decoder generates a small set of hard-decision candidates by considering alternative assignments at low-reliability positions, with erased positions prioritized and ties among equal-reliability positions resolved deterministically. A bounded-distance decoder is applied to each candidate, and a block is accepted only when at least one decoded candidate matches the designated codeword and the block score is at least τ_{blk} Chase (1972); Lin & Costello (2004); Richardson & Urbanke (2008). Unlike the hard-decision ablation, CORE-BREW does not require the raw hard-decision vector itself to lie within the BCH bounded-distance radius of the designated codeword, since limited Chase perturbations may produce candidates that decode to the designated codeword. Instead, the acceptance rule combines designated-codeword verification, score thresholding, and a bounded candidate-search budget.

Algorithm 3 Window-Shifting Detection with Block-Dependent Designated Codewords

Input: tokens $(s_0, \ldots, s_{N-1})$, prompt length N_p, key K, BCH code C, designated codewords $\{c^{(j)}\}$, block length n, max shift $s_{\max}$, text threshold θ, hit rate $p^\star$, mode $\in \{\texttt{hard}, \texttt{soft}\}$
Parameters: decoding radius t_{BCH}, block threshold τ_{blk}, list-decoding parameters $L_{\max}, \ell$
Construct the fixed keyed partition (L_0, L_1) from K
$\lambda \leftarrow \log\Big(\frac{p^\star}{1-p^\star}\Big)$
$U \leftarrow N - N_p$
$M \leftarrow \lfloor U/n \rfloor$
if $M = 0$ **then**
  output `not watermarked`
  return
end if
$M_{\mathrm{match}} \leftarrow 0$
for $j = 0$ **to** $M - 1$ **do**
  matched $\leftarrow$ `false`
  for each shift $s \in \{-s_{\max}, \ldots, s_{\max}\}$ **do**
    $(B^{(j,s)}, E^{(j,s)}, \Lambda^{(j,s)}) \leftarrow \mathrm{ConstructObservation}(s, j, c^{(j)}, K, \lambda)$
    if mode $=$ `hard` **then**
      Form $\widetilde{B}^{(j,s)}$ from $B^{(j,s)}$, replacing positions marked in $E^{(j,s)}$ with key-derived PRF bits
      if $d_H(\widetilde{B}^{(j,s)}, c^{(j)}) \le t_{\mathrm{BCH}}$ **then**
        matched $\leftarrow$ `true`
        break
      end if
    else
      $S^{(j)}(s) \leftarrow \sum_{b=0}^{n-1} (2c^{(j)}[b] - 1)\Lambda_b^{(j,s)}$
      if $S^{(j)}(s) \ge \tau_{\mathrm{blk}}$ **then**
        $\widehat{\mathcal{C}} \leftarrow \mathrm{ChaseListDecode}(B^{(j,s)}, \Lambda^{(j,s)}, C, L_{\max}, \ell)$
        if $c^{(j)} \in \widehat{\mathcal{C}}$ **then**
          matched $\leftarrow$ `true`
          break
        end if
      end if
    end if
  end for
  if matched **then**
    $M_{\mathrm{match}} \leftarrow M_{\mathrm{match}} + 1$
  end if
end for
output `watermarked` iff $\frac{M_{\mathrm{match}}}{M} \ge \theta$

C Theoretical Proofs

This appendix provides detailed proofs of the main analytical claims in Section 5. We first prove the Constant Hit-Rate property of the embedding rule, then analyze false positives for BREW's bounded-distance designated-codeword rule as a hard-decision reference, and finally derive score-based tail bounds for CORE-BREW's soft-decision acceptance rule. We conclude with standard concentration results for finite-sample estimation of empirical detection rates used in Section 6.

C.1 Proof of Constant Hit-Rate Embedding

We first establish the Constant Hit-Rate property underlying the calibrated channel model used in Sections 4–5.

Proposition C.1 (Constant hit-rate property)**.** *Let $p^\star \in (1/2, 1)$ be a fixed target hit-rate and $L_t \subseteq \mathcal{V}$ be a target token list at position t. Let $\ell_t(v)$ denote the base model logit for token $v \in \mathcal{V}$ and let*

$$p_t(v) = \frac{\exp(\ell_t(v))}{\sum_{u \in \mathcal{V}} \exp(\ell_t(u))}$$

be the corresponding softmax distribution. Define

$$m_t = \sum_{v \in L_t} p_t(v),$$

and choose the bias

$$\delta_t = \log\left(\frac{p^\star}{1-p^\star}\right) - \log\left(\frac{m_t}{1-m_t}\right).$$

Define biased logits by $\tilde{\ell}_t(v) = \ell_t(v) + \delta_t$ for $v \in L_t$ and $\tilde{\ell}_t(v) = \ell_t(v)$ otherwise, and let $\tilde{p}_t = \mathrm{softmax}(\tilde{\ell}_t)$. Then the biased distribution satisfies

$$\sum_{v \in L_t} \tilde{p}_t(v) = p^\star.$$

Proof. Let

$$A_t = \sum_{v \in L_t} \exp(\ell_t(v)), \qquad B_t = \sum_{v \notin L_t} \exp(\ell_t(v)).$$

Then $m_t = A_t/(A_t + B_t)$ and $(1 - m_t) = B_t/(A_t + B_t)$, so $A_t/B_t = m_t/(1 - m_t)$. Under the bias, the total mass on L_t becomes

$$\sum_{v \in L_t} \tilde{p}_t(v) = \frac{\sum_{v \in L_t} \exp(\ell_t(v) + \delta_t)}{\sum_{u \in \mathcal{V}} \exp(\tilde{\ell}_t(u))} = \frac{e^{\delta_t} A_t}{e^{\delta_t} A_t + B_t}.$$

Substitute $e^{\delta_t} = \frac{p^\star}{1-p^\star} \cdot \frac{1-m_t}{m_t} = \frac{p^\star}{1-p^\star} \cdot \frac{B_t}{A_t}$ to obtain

$$\frac{e^{\delta_t} A_t}{e^{\delta_t} A_t + B_t} = \frac{\frac{p^\star}{1-p^\star} B_t}{\frac{p^\star}{1-p^\star} B_t + B_t} = \frac{\frac{p^\star}{1-p^\star}}{\frac{p^\star}{1-p^\star} + 1} = p^\star.$$

□

Remark (distortion guards and erasures). The proposition above describes positions for which the Constant Hit-Rate target is fully realized. When distortion guards such as bias clipping or embedding skips are activated, the equality need not hold at every position. Skipped positions are treated as erasures with zero LLR, while the position-wise score contributions remain bounded. Accordingly, the position-homogeneous BSC interpretation in Section 5 applies to fully calibrated, non-erased positions.

C.2 Hard-Decision Reference: False Positive Bounds under H_0

We derive false positive bounds for BREW's bounded-distance designated-codeword rule, which serves as the hard-decision reference in Section 5. Under the null hypothesis H_0, the text is unwatermarked and independent of the secret key K. Our analysis uses two levels of approximation. First, for a single block, we use the standard uniform-block approximation that the hard-decision block induced by the keyed partition behaves approximately as a uniform element of $\{0,1\}^n$ under H_0. Second, for text-level aggregation, we use an idealized block-level independence approximation for analytical intuition. This aggregation model does not require exact bit-wise independence within a block.

Proposition C.2 (Single-block false positive probability)**.** *Let $C \subseteq \{0,1\}^n$ be a binary BCH code with minimum distance $d_{\min}$ and unique-decoding radius $t = \lfloor (d_{\min} - 1)/2 \rfloor$ Lin & Costello (2004). Fix a designated codeword $c \in C$. Under the standard uniform-block approximation for H_0, the single-block false match probability is*

$$p_0 = \Pr\big(\mathrm{dist}_H(B, c) \le t\big) = \frac{V_2(n,t)}{2^n}, \qquad V_2(n,t) = \sum_{i=0}^{t} \binom{n}{i}.$$

Proof. Under the uniform-block approximation, B is treated as uniformly distributed over $\{0,1\}^n$. The event $\mathrm{dist}_H(B,c) \le t$ holds if and only if B lies in the Hamming ball of radius t around c. That ball contains exactly $V_2(n,t)$ binary strings. Therefore, the probability is $V_2(n,t)/2^n$. □

Text-level aggregation model. For text-level FPR analysis, we use an idealized approximation in which block-level spurious match indicators are treated as approximately independent Bernoulli variables. This approximation is standard in designated-codeword analyses and is useful for understanding how text-level FPR scales with the number of complete blocks. However, for the fixed-partition instantiation in Appendix B, the approximation is not an exact consequence of independently seeded block partitions. The same vocabulary partition is reused across blocks, so repeated tokens and natural-language correlations can introduce dependencies. Accordingly, we use the aggregate Chernoff bound below as an idealized analysis tool, complemented by empirical calibration and validation on held-out unwatermarked text.

For the shifted-window analysis below, we continue to use the same uniform-block approximation for each candidate block vector. This approximation is used to estimate single-block false-match probabilities and shift-overlap corrections; text-level aggregation is handled separately through the idealized block-level aggregation model described above.

For each candidate shift s, let

$$E_s = \Big\{ \mathrm{dist}_H(B^{(s)}, c) \le t \Big\}$$

denote the event that the shifted hard-decision block lies within the bounded-distance region of the designated codeword.

Lemma C.3 (Adjacent-shift joint probability under the uniform-block model)**.** *Let $\Delta \in \{1,\dots,n-1\}$ and let $c \in \{0,1\}^n$ be a fixed designated codeword. Define the codeword shift autocorrelation*

$$a(\Delta) = |\{i \in [0, n-\Delta) : c[i+\Delta] = c[i]\}|.$$

Under an i.i.d. Bernoulli$(1/2)$ model for the $n+\Delta$ bits spanning the two overlapping windows,

$$\begin{aligned}\Pr(E_s \cap E_{s+\Delta}) = \sum_{a=0}^{a(\Delta)} \sum_{b=0}^{n-\Delta-a(\Delta)} \binom{a(\Delta)}{a} \binom{n-\Delta-a(\Delta)}{b} 2^{-(n-\Delta)} \\ \times F_\Delta(t-a-b)\, F_\Delta\big(t-a-(n-\Delta-a(\Delta))+b\big),\end{aligned}$$

where $F_\Delta(k) = \Pr(\mathrm{Bin}(\Delta,1/2) \le k)$ for $k \ge 0$ and $F_\Delta(k) = 0$ for $k < 0$.

Proof. Decompose the two windows as $B^{(s)} = (X, Y)$ and $B^{(s+\Delta)} = (Y, Z)$, where $|X| = |Z| = \Delta$ and $|Y| = n - \Delta$. Under the uniform-block approximation, the non-overlapping portions X and Z and the shared portion Y are treated as independent uniform bit strings. Define:

- $A \sim \mathrm{Bin}(a(\Delta), 1/2)$: errors of Y at the $a(\Delta)$ positions where $c[i+\Delta] = c[i]$ (which contribute identically to both events);
- $B \sim \mathrm{Bin}(n-\Delta-a(\Delta), 1/2)$: errors of Y in E_s at positions where $c[i+\Delta] \ne c[i]$ (errors in $E_{s+\Delta}$ at these positions equal $n-\Delta-a(\Delta)-B$);
- $W_X \sim \mathrm{Bin}(\Delta, 1/2)$ and $W_Z \sim \mathrm{Bin}(\Delta, 1/2)$: errors on the non-shared portions.

The total errors satisfy $d_H(B^{(s)}, c) = W_X + A + B$ and $d_H(B^{(s+\Delta)}, c) = W_Z + A + (n-\Delta-a(\Delta)) - B$. The joint event $\{E_s \cap E_{s+\Delta}\}$ requires both sums $\le t$. Conditioning on (A, B) and using independence of W_X, W_Z yields the stated closed form. □

Proposition C.4 (Window-shifting bound for the hard-decision reference)**.** *Under the uniform-block approximation for candidate block vectors, the single-block false positive probability under window shifting satisfies*

$$\Pr\left(\bigcup_{s\in\mathcal{S}} E_s\right) \le Sp_0 - (S-1)\Pr(E_s \cap E_{s+1}) \le Sp_0.$$

Proof. The first inequality follows from Hunter's spanning-tree bound Hunter (1976) applied to the chain spanning tree on $\mathcal{S}$. The second follows from $\Pr(E_s \cap E_{s+1}) \geq 0$. The joint probability $\Pr(E_s \cap E_{s+1})$ admits the closed-form expression given in Lemma C.3 with $\Delta = 1$, depending on the codeword shift autocorrelation $a(1) = |\{i : c[i+1] = c[i]\}|$. □

Proposition C.5 (Aggregate FPR Chernoff bound under an idealized aggregation model)**.** *Let $X_1, \ldots, X_M$ be block-level spurious match indicators under H_0. Under the idealized block-level aggregation model, suppose these indicators are treated as independent Bernoulli variables satisfying $\Pr(X_j = 1) \leq p_{\mathrm{blk}}$ for all j. Let*

$$\rho_M = \frac{1}{M} \sum_{j=1}^{M} X_j.$$

Then for any threshold $\theta \in (p_{\mathrm{blk}}, 1)$,

$$\Pr(\rho_M \geq \theta) \leq \exp\big(-M\, D(\theta \| p_{\mathrm{blk}})\big),$$

where

$$D(\theta \| p_{\mathrm{blk}}) = \theta \log \frac{\theta}{p_{\mathrm{blk}}} + (1-\theta) \log \frac{1-\theta}{1-p_{\mathrm{blk}}}$$

is the binary Kullback–Leibler divergence.

Proof. Under the idealized aggregation model, the block-level indicators are dominated by independent Bernoulli variables with success probability at most p_{blk}. The result follows from the standard Chernoff bound for the upper tail of their empirical mean. □

Hard-decision text-level FPR. Applying the aggregate bound with

$$p_{\mathrm{blk}}^{(\mathrm{shift})} := S p_0 - (S-1) \Pr(E_s \cap E_{s+1}) \leq S p_0$$

gives the idealized text-level FPR bound for the hard-decision reference used in Section 5. For the fixed-partition instantiation, this bound should be interpreted together with empirical FPR calibration and validation under H_0.

C.3 Hard-Decision Reference: Single-Block Success Probability under H_1

For completeness, we restate the standard bounded-distance success probability of the hard-decision reference under the alternative hypothesis H_1, assuming a BSC observation model.

Proposition C.6 (Single-block success probability)**.** *Assume that under H_1 (no attacks) the channel from a designated codeword $c \in C$ to the detector's PRF-filled hard-decision block $\widetilde{B} \in \{0,1\}^n$ is a binary symmetric channel with crossover probability $q \in [0, 1/2)$. Assume the bounded-distance block rule succeeds whenever the number of bit errors in $\widetilde{B}$ is at most t and fails otherwise. Then the single-block success probability is*

$$p_{1,\mathrm{HD}} = \Pr\big(\mathrm{dist}_H(\widetilde{B}, c) \leq t \,\big|\, H_1\big) = \sum_{i=0}^{t} \binom{n}{i} q^i (1-q)^{n-i}.$$

Proof. Under a BSC with crossover q on the PRF-filled hard-decision representation, the number of bit errors $\mathrm{dist}_H(\widetilde{B}, c)$ has distribution $\mathrm{Bin}(n, q)$. The bounded-distance rule succeeds if and only if the error count is at most t, yielding the stated sum. □

C.4 CORE-BREW: Score-Based Tail Bounds

We next analyze the score-based acceptance rule used by CORE-BREW. Since the block score is a sum of bounded random variables, it admits exponential tail bounds under both H_0 and H_1. These bounds characterize false positives under score-based rejection without requiring the acceptance region to coincide with a bounded-distance Hamming ball.

For the fixed-partition instantiation, the independence assumptions in this subsection should be understood as an idealized bounded-variable analysis model for score calibration rather than exact probabilistic guarantees. Let $p^\star \in (1/2, 1)$ be the Constant Hit-Rate parameter and let

$$L = \log\left(\frac{p^\star}{1-p^\star}\right).$$

For each block j and shift s, define per-bit LLRs $\Lambda_b^{(j,s)} \in [-L, L]$, where erasures correspond to $\Lambda = 0$. Define the designated-codeword score

$$S^{(j)}(s) = \sum_{b=0}^{n-1} (2c^{(j)}[b] - 1)\, \Lambda_b^{(j,s)}.$$

A necessary condition for CORE-BREW to accept a block is

$$\max_s S^{(j)}(s) \geq \tau_{\mathrm{blk}}.$$

Therefore, bounding $\Pr(\max_s S^{(j)}(s) \geq \tau_{\mathrm{blk}})$ provides an upper bound on the block-level false positive probability.

Proposition C.7 (Score tail bounds under H_0 and H_1)**.** *For the following score tail bound, we use an analysis model in which, for a fixed block j and shift s, the variables*

$$Z_b = (2c^{(j)}[b] - 1)\, \Lambda_b^{(j,s)}$$

are treated as independent and bounded, with $Z_b \in [-L, L]$.

***(Null hypothesis** H_0**.)** If $\mathbb{E}[Z_b] = 0$ for all b under H_0, then for any $\tau_{\mathrm{blk}} > 0$,*

$$\Pr_{H_0}\big(S^{(j)}(s) \geq \tau_{\mathrm{blk}}\big) \leq \exp\left(-\frac{\tau_{\mathrm{blk}}^2}{2nL^2}\right).$$

Consequently, with window shifting over $S = 2s_{\max} + 1$ shifts,

$$\Pr_{H_0}\left(\max_s S^{(j)}(s) \geq \tau_{\mathrm{blk}}\right) \leq S \exp\left(-\frac{\tau_{\mathrm{blk}}^2}{2nL^2}\right).$$

***(Alternative hypothesis** H_1**.)** Suppose that at the true alignment $s = s_{\mathrm{true}}$, there are n_{eff} non-erased positions and the non-erased Z_b satisfy $\mathbb{E}[Z_b] = \mu > 0$ while erased positions contribute $Z_b = 0$. Then for any $\tau_{\mathrm{blk}} < n_{\mathrm{eff}}\mu$,*

$$\Pr_{H_1}\big(S^{(j)}(s_{\mathrm{true}}) \leq \tau_{\mathrm{blk}}\big) \leq \exp\left(-\frac{(n_{\mathrm{eff}}\mu - \tau_{\mathrm{blk}})^2}{2n_{\mathrm{eff}}L^2}\right).$$

Proof. Under this bounded-independence analysis model, both inequalities follow from Hoeffding's inequality for sums of bounded independent random variables Hoeffding (1963). Under H_0, apply Hoeffding to $S^{(j)}(s) = \sum_b Z_b$ with mean 0 and bounds $\pm L$. The shift-max bound follows by a union bound over S shifts. Under H_1, apply Hoeffding to the sum over the n_{eff} non-erased positions (erased positions contribute 0 deterministically). □

Remark (decoding does not increase FPR beyond the score bound). CORE-BREW additionally requires the decoding wrapper to return the designated codeword $c^{(j)}$. This can only *reduce* acceptance relative to the pure score test, so the bounds above remain valid upper bounds on block-level false positives. Conversely, the score separation under H_1, together with Chase-style list decoding, explains how CORE-BREW can improve detection power Chase (1972); Lin & Costello (2004); Richardson & Urbanke (2008).

C.5 Aggregate False Negative Bound

We state the standard binomial lower-tail bound used to convert single-block success probabilities into a text-level FNR bound.

Proposition C.8 (Aggregate false negative bound)**.** *Let $Y_1, \ldots, Y_M$ be block-level success indicators under H_1. Under the block-level independence analysis model, suppose that $\Pr(Y_j = 1) \geq p_1$ for all j. Let*

$$\rho_M = \frac{1}{M} \sum_{j=1}^{M} Y_j.$$

Then for any threshold $\theta \in (0, 1)$ with $\theta < p_1$,

$$\Pr(\rho_M < \theta) \leq \exp\big(-M\, D(\theta \| p_1)\big),$$

where $D(\theta \| p_1)$ is the binary relative entropy.

Proof. Under the block-level independence analysis model, the lower tail of the average success rate is bounded by the standard Chernoff bound with dominating Bernoulli parameter p_1. This yields the stated inequality. □

Application to the detection rules. For the hard-decision reference, p_1 can be taken as $p_{1,\mathrm{HD}}$ (or its shifted variant). For CORE-BREW, p_1 denotes the overall block-level acceptance probability, which depends on score separation, successful designated-codeword decoding, and correct alignment in the shift search. The H_1 score tail bound above characterizes the score-threshold component of this probability, while the overall FNR–TPR trade-off is evaluated empirically in Section 6.

C.6 Concentration of Empirical FPR (Finite-Sample Justification)

We provide a standard concentration inequality justifying empirical estimation of (single-block or text-level) FPR. This result applies to independently sampled evaluation texts or trials and is used to justify uncertainty estimates for empirical FPR measurements.

Lemma C.9 (Concentration of empirical FPR)**.** *Let $X_1, \ldots, X_N$ be independent Bernoulli variables with $\Pr(X_i = 1) = p_{\mathrm{FP}}$, representing false positives for N independent trials under H_0 (a "trial" may refer to a block event or a text-level event). Let*

$$\hat{p}_{\mathrm{FP},N} = \frac{1}{N} \sum_{i=1}^{N} X_i$$

be the empirical estimate of p_{FP}. Then for any $\varepsilon > 0$,

$$\Pr\big(|\hat{p}_{\mathrm{FP},N} - p_{\mathrm{FP}}| \geq \varepsilon\big) \leq 2 \exp(-2N\varepsilon^2).$$

In particular, to guarantee $\Pr(|\hat{p}_{\mathrm{FP},N} - p_{\mathrm{FP}}| \geq \varepsilon) \leq \delta$, it suffices to take

$$N \geq \frac{1}{2\varepsilon^2} \log\left(\frac{2}{\delta}\right).$$

Proof. This is Hoeffding's inequality applied to the empirical mean of bounded independent random variables Hoeffding (1963). □

D Additional Experiments

This appendix provides additional experiments and diagnostics that complement the main results in Section 6. We focus on (i) the robustness–quality trade-off of Constant Hit-Rate embedding, (ii) the contribution of soft-decision processing relative to its hard-decision ablation, and (iii) the behavior of CORE-BREW under different operating parameters and evaluation settings.

Schemes compared. Unless otherwise stated, experiments follow the setup in Section 6, using the same models, prompts, sampling settings, BCH parameters, and evaluation pipeline. For a matched comparison, we use BCH $(63, 7, 15)$ for both BREW and CORE-BREW, although the original BREW evaluation uses BCH $(31, 6, 7)$. This controls for differences in ECC strength and isolates the contribution of CORE-BREW's embedding and decoding mechanisms. We consider:

Table 3: Clean detection performance on Mistral-7B. We report mean ± standard deviation over multiple random seeds at each method's default or recommended detection threshold.

	C4		OpenGen	
Method	TPR ↑	FPR ↓	TPR ↑	FPR ↓
MPAC	1.0000 ± 0.0000	0.7333 ± 0.0062	1.0000 ± 0.0000	0.8683 ± 0.0094
Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9250 ± 0.0071	1.0000 ± 0.0000	0.9383 ± 0.0131
BREW	0.9267 ± 0.0184	0.0000 ± 0.0000	0.9717 ± 0.0085	0.0000 ± 0.0000
CORE-BREW	0.9167 ± 0.0047	0.0000 ± 0.0000	0.9667 ± 0.0024	0.0000 ± 0.0000

- **BREW Kim et al. (2026):** the designated-codeword baseline using fixed-bias embedding, hard-decision BCH decoding, and window-shifting detection.
- **CORE-BREW:** our proposed method, combining Constant Hit-Rate embedding, entropy-aware erasures, calibrated LLR scoring, and Chase-style list decoding with score-based acceptance Chase (1972); Lin & Costello (2004); Richardson & Urbanke (2008).
- **CORE-BREW w/o Soft Decoding:** a hard-decision ablation that retains the same Constant Hit-Rate embedding and erasure decisions as CORE-BREW, but replaces LLR-based scoring and list decoding with bounded-distance designated-codeword verification using PRF-filled erased positions. This ablation provides a matched hard-decision reference for evaluating the contribution of soft-decision processing.

Entropy-aware embedding and erasures. CORE-BREW uses entropy-aware erasures to limit excessive distributional perturbation in low-entropy contexts. After computing and clipping the Constant Hit-Rate bias, a position is erased when the base distribution is too peaked ($\max_v p_t(v) \geq q_{\max}$), the entropy falls below $H_{\min}$, or the post-bias target-list mass does not exceed $1/2$. Erased positions are sampled without watermark bias and assigned zero LLR during model-aware detection, allowing the decoder to ignore unreliable evidence.

D.1 Clean Detection Results

Table 3 summarizes clean detection performance on Mistral-7B at text length $T = 500$, using each method's default or recommended detection threshold. MPAC Yoo et al. (2024) and Qu et al. Qu et al. (2025) achieve near-perfect TPR but exhibit substantially higher FPR, indicating limited discrimination between watermarked and unwatermarked text. In contrast, BREW Kim et al. (2026) and CORE-BREW maintain zero observed FPR while achieving high clean-detection TPR on both C4 and OpenGen. Their clean detection performance is comparable, with only small differences in TPR across the two datasets. CORE-BREW uses the default operating point $p^\star = 0.9$, with the full hit-rate sweep reported in Appendix D.2.

D.2 Effect of Target Hit-Rate $p^\star$ on Detection and Payload Recovery

We analyze the effect of the target hit-rate $p^\star$ on CORE-BREW. Figure 4 reports TPR, F1, and payload bit accuracy under clean generation and 10% substitution. Under clean generation, all three metrics improve sharply from $p^\star = 0.6$ to 0.8 and largely saturate thereafter. In particular, TPR and F1 exceed 90% from $p^\star = 0.8$ onward, while payload bit accuracy reaches approximately 90% at $p^\star = 0.8$ with only marginal gains at higher values.

Under 10% substitution, performance is lower overall but follows the same trend, with TPR, F1, and payload bit accuracy increasing markedly up to $p^\star = 0.9$. The additional gain from $p^\star = 0.9$ to 0.95 is relatively small. We therefore use $p^\star = 0.9$ as the default setting, since clean performance is already near saturation while further increases provide only limited improvement under substitution.

D.3 Code Length and Error-Correction Parameters

Table 4 compares three BCH configurations for CORE-BREW on OPT-1.3B with C4 under the same low-FPR constraint of $\mathrm{FPR} \leq 1\%$. Rather than comparing TPR at a single detection threshold, we

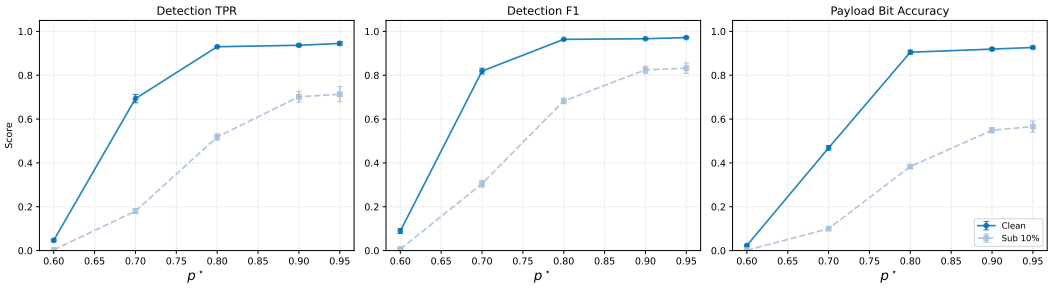

Figure 4: Effect of the target hit-rate $p^\star$ on CORE-BREW using OPT-1.3B with C4 under clean generation and 10% substitution. We report detection TPR, detection F1, and payload bit accuracy. Increasing $p^\star$ improves both detection and payload recovery, with gains largely saturating at high hit rates.

Table 4: Code-length sensitivity of CORE-BREW on OPT-1.3B with C4 at $p^\star = 0.9$ and $T = 500$. We report TPR under the constraint $\mathrm{FPR} \leq 1\%$.

BCH (n, k, t)	**Clean**	**10% Substitution**
$(15, 5, 3)$	0.9350 ± 0.0108	0.5317 ± 0.0170
$(31, 6, 7)$	0.9283 ± 0.0024	0.6400 ± 0.0255
$(63, 7, 15)$	$\mathbf{0.9383 \pm 0.0062}$	$\mathbf{0.7017 \pm 0.0249}$

evaluate the achievable TPR under the same false-positive requirement to compare detection power across code configurations. Under clean generation, all three configurations achieve comparable detection performance, with $(63, 7, 15)$ attaining the highest TPR of 0.9383. The advantage of the longer code becomes more pronounced under 10% substitution, where TPR increases from 0.5317 for $(15, 5, 3)$ and 0.6400 for $(31, 6, 7)$ to 0.7017 for $(63, 7, 15)$. Thus, the longer BCH configuration provides greater robustness to substitution while maintaining the same low-FPR requirement. Based on these results, we use BCH $(63, 7, 15)$ as the default configuration when sufficient text is available. In the insertion/deletion experiments, we additionally use $s_{\max} = 10$ to improve alignment robustness, as analyzed in Appendix D.4. For shorter continuations that cannot accommodate a complete length-63 block, CORE-BREW can instead use shorter BCH configurations, as evaluated in Appendix D.9.

D.4 Window-Shift Sensitivity under Insertion and Deletion Attacks

We analyze the sensitivity of CORE-BREW to the window-shift budget $s_{\max}$ under 10% insertion and deletion attacks. Figure 5 reports TPR and FPR for $s_{\max} \in \{0, 1, 2, 3, 5, 7, 10, 15, 20\}$. Under deletion, TPR is already high at $s_{\max} = 0$ (0.8833), rises to 0.9883 at $s_{\max} = 1$, and reaches 1.0000 by $s_{\max} = 3$, after which it saturates. In contrast, insertion is substantially more sensitive to the shift budget: TPR increases steadily from 0.1050 at $s_{\max} = 0$ to 0.5250 at $s_{\max} = 10$ and 0.6950 at $s_{\max} = 20$. Thus, deletion requires only a small alignment search, whereas insertion continues to benefit from a larger shift budget.

A larger $s_{\max}$ also increases the likelihood of spurious matches. Under deletion, the observed FPR increases from 0 at $s_{\max} = 0$ to 0.0067 at $s_{\max} = 10$ and 0.0100 at $s_{\max} = 20$. Under insertion, it remains near zero at small shift budgets, reaches 0.0083 at $s_{\max} = 10$, and 0.0133 at $s_{\max} \in \{15, 20\}$. We therefore use $s_{\max} = 10$ as the default setting, which substantially improves insertion TPR over no shift search while keeping the observed FPR below 1% for both attacks.

D.5 Sensitivity to the Text-Level Match Threshold

We examine the sensitivity to the text-level match threshold under the 10% insertion attack, where alignment errors make block recovery particularly challenging. Increasing the requirement from one to two matched blocks reduces the observed FPR from 0.0083 to zero, but also sharply decreases

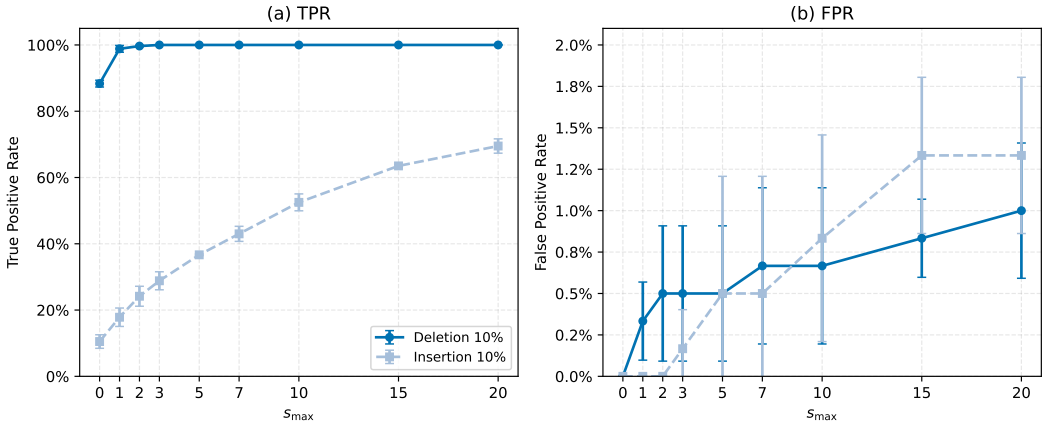

Figure 5: Sensitivity to the window-shift budget $s_{\max}$ under 10% insertion and deletion attacks on OPT-1.3B with C4 at $p^\star = 0.9$. Increasing $s_{\max}$ improves TPR, particularly under insertion, with a modest increase in observed FPR.

Table 5: Sensitivity to the text-level match threshold under 10% insertion on OPT-1.3B with C4 ($T = 500$).

Required matched blocks	**TPR**	**FPR**
$M_{\text{match}} \geq 1$	0.5250 ± 0.0255	0.0083 ± 0.0062
$M_{\text{match}} \geq 2$	0.0567 ± 0.0131	0.0000 ± 0.0000

TPR from 0.5250 to 0.0567. Thus, requiring two matches provides only a small additional reduction in observed FPR at a substantial cost in detection power. We therefore use $M_{\text{match}} \geq 1$ as the default criterion.

D.6 Text Quality Evaluation

We evaluate text quality at the default operating point of each watermarking method. Baseline methods use a fixed logit bias $\delta = 3$, while CORE-BREW uses the target hit rate $p^\star = 0.9$ with entropy-aware safeguards. Conditional PPL is evaluated on OPT-1.3B generations over C4 using Llama-2-7B as an independent oracle conditioned on the prompt. BLEU Papineni et al. (2002) and BERTScore-F1 Zhang et al. (2020) are evaluated on WMT16 DE→EN validation translations generated by NLLB-600M against the corresponding reference translations.

CORE-BREW exhibits higher conditional PPL than the evaluated baselines, indicating lower likelihood under the independent oracle model. On WMT16, however, CORE-BREW achieves a BLEU score of 31.82, nearly identical to 31.81 for unwatermarked generation, and a BERTScore-F1 of 0.881, matching that of unwatermarked generation. In contrast, the evaluated baselines show substantially lower BLEU and BERTScore-F1 under their watermarking settings. These results suggest a likelihood–robustness trade-off while maintaining reference-based lexical and semantic quality comparable to unwatermarked generation.

Role of Entropy-Aware Safeguards. An unconstrained constant hit-rate formulation can induce excessively large instantaneous biases $|\delta_t|$ when the target-list mass m_t approaches 0 or 1, a condition frequently observed in low-entropy contexts. Such extreme biasing can distort the output distribution and degrade quality. The entropy-aware mechanism bounds $|\delta_t|$ and converts these positions into erasures, mitigating distributional distortion while preserving detectability through erasure-tolerant decoding Lin & Costello (2004); Richardson & Urbanke (2008).

Table 6: Text quality at $T = 500$. PPL is evaluated with a Llama-2-7B oracle, while BLEU and BERTScore-F1 are evaluated against WMT16 references. We report mean $\pm$ standard deviation over multiple runs.

Method	Parameter	PPL ↓	BLEU ↑	BERTScore-F1 ↑
Unwatermarked	–	8.454 ± 0.053	31.81 ± 0.00	0.881 ± 0.000
MPAC	$\delta = 3$	12.684 ± 0.114	22.30 ± 0.00	0.855 ± 0.000
Qu et al. Qu et al. (2025)	$\delta = 3$	13.326 ± 0.004	21.41 ± 0.00	0.847 ± 0.000
BREW	$\delta = 3$	13.643 ± 0.071	21.73 ± 0.74	0.853 ± 0.001
CORE-BREW	$p^\star = 0.9$	15.306 ± 0.210	31.82 ± 0.00	0.881 ± 0.000

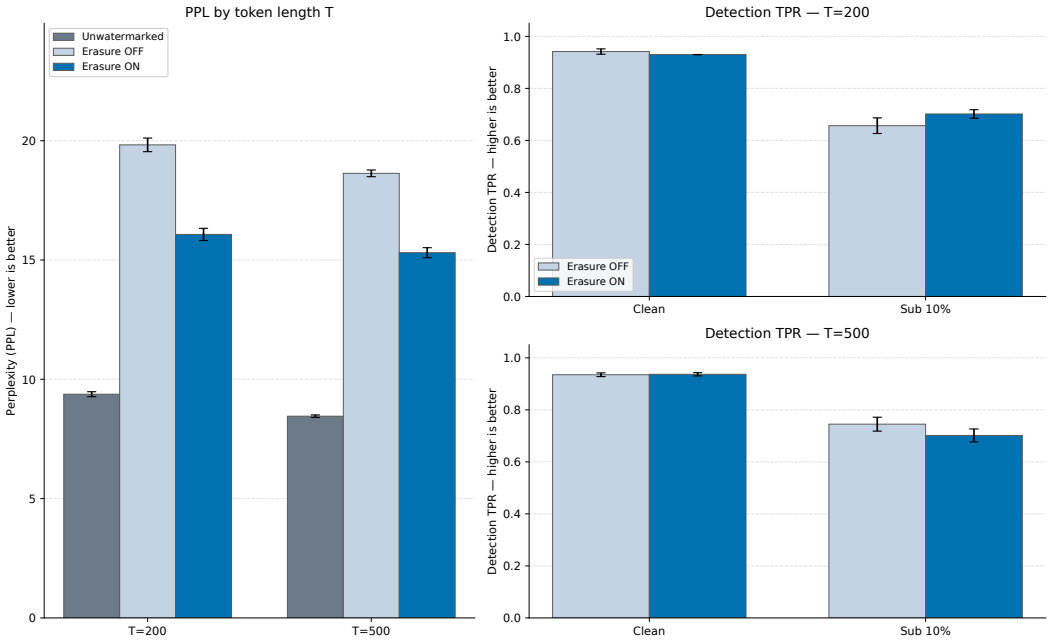

Figure 6: Effect of entropy-aware erasures on conditional PPL and detection TPR on OPT-1.3B with C4 at $p^\star = 0.9$. Enabling erasures substantially reduces PPL at both $T = 200$ and $T = 500$, while TPR remains broadly comparable under clean and 10% substitution settings.

D.7 Ablation on Entropy-Aware Erasures

To evaluate the system-level effect of entropy-aware erasures, Figure 6 compares CORE-BREW with the safeguard enabled and disabled on OPT-1.3B with C4 at $p^\star = 0.9$. Enabling erasures substantially reduces conditional PPL, from 19.825 ± 0.285 to 16.072 ± 0.255 at $T = 200$ and from 18.631 ± 0.140 to 15.306 ± 0.210 at $T = 500$. Although these values remain higher than the corresponding unwatermarked PPLs of 9.376 ± 0.104 and 8.454 ± 0.053, the erasure-enabled configuration consistently exhibits lower distributional distortion than the no-erasure variant. The corresponding changes in detection performance are relatively modest. Under clean generation, TPR remains nearly unchanged at both text lengths. Under 10% substitution, enabling erasures improves TPR at $T = 200$ but reduces it at $T = 500$, indicating that the effect on detection depends on the evaluation setting rather than being uniformly beneficial. Because the erasure mechanism jointly affects embedding and detection, this ablation should be interpreted as a system-level comparison rather than as isolating a single operation. When enabled, safeguarded positions are left unbiased during generation and contribute zero LLR at detection; when disabled, CHR embedding and constant-magnitude LLRs are applied at all positions. Overall, the erasure-enabled configuration substantially lowers conditional PPL while producing only limited changes in TPR, supporting its use as a distortion-control safeguard in CORE-BREW.

Table 7: Payload scaling of CORE-BREW under clean generation using OPT-1.3B with C4. We evaluate total payload sizes from 7 to 49 bits within a fixed $T = 500$ generation budget and report detection TPR, FPR, and payload bit accuracy.

Payload Bits	TPR	FPR	Bit Acc.
7	0.9317 ± 0.0062	0.0000 ± 0.0000	0.9317 ± 0.0062
14	0.9383 ± 0.0062	0.0000 ± 0.0000	0.9308 ± 0.0042
21	0.9300 ± 0.0041	0.0000 ± 0.0000	0.8967 ± 0.0152
28	0.9350 ± 0.0082	0.0000 ± 0.0000	0.8879 ± 0.0136
49	0.9450 ± 0.0071	0.0017 ± 0.0024	0.8505 ± 0.0067

Table 8: Short-text detection and payload recovery under clean generation using OPT-1.3B with C4. We report TPR, FPR, and payload bit accuracy for continuations of $T \in \{50, 100\}$ tokens. "–" indicates that no complete BCH block can be formed at the given text length.

	$T = 50$			$T = 100$		
Method	**TPR**	**FPR**	**Payload Bit Acc.**	**TPR**	**FPR**	**Payload Bit Acc.**
MPAC ($\delta = 3$)	0.997	0.323	0.867	1.000	0.368	0.960
Qu et al. Qu et al. (2025) ($\delta = 3$)	0.938	0.937	0.060	0.975	0.932	0.575
CORE-BREW $(15, 5, 3)$	0.930	0.097	0.896	0.942	0.182	0.937
CORE-BREW $(31, 6, 7)$	0.910	0.002	0.910	0.930	0.012	0.912
CORE-BREW $(63, 7, 15)$	–	–	–	0.912	0.000	0.912

D.8 Payload Scalability via Block-Wise Embedding

The payload capacity of CORE-BREW is not fundamentally constrained to 7 bits. In BCH $(63, 7, 15)$, $k = 7$ represents the message dimension of an individual BCH codeword rather than the cumulative text-level capacity. Since CORE-BREW operates block-wise, distinct 7-bit payload chunks can be assigned across successive blocks. A $T = 500$ continuation can accommodate up to seven complete length-63 blocks, corresponding to 49 payload bits when each block carries a distinct chunk. Table 7 evaluates this block-wise scalability on OPT-1.3B with C4 under clean generation.

Across payload sizes from 7 to 49 bits, text-level detection remains stable: TPR ranges from 0.9300 to 0.9450, while the observed FPR remains at or below 0.0017. Payload bit accuracy decreases gradually from 0.9317 at 7 bits to 0.8505 at 49 bits as less repeated block-level evidence is available for each distinct payload chunk. These results show that the per-block value $k = 7$ does not impose a 7-bit limit on CORE-BREW's total payload, and that payloads up to 49 bits can be embedded within $T = 500$ while retaining comparable clean detection performance. Appendix D.9 considers the complementary short-text setting.

D.9 Short-Text Detection and Payload Recovery

While BCH $(63, 7, 15)$ is preferred when sufficient text is available, CORE-BREW can support shorter continuations by selecting a BCH configuration whose block length fits the available text. Table 8 evaluates this flexibility on OPT-1.3B with C4 for $T = 50$ and $T = 100$. At $T = 50$, BCH $(63, 7, 15)$ cannot form a complete block, whereas shorter BCH configurations remain applicable. In particular, BCH $(31, 6, 7)$ achieves a TPR of 0.910 with an FPR of 0.002 and a payload bit accuracy of 0.910. However, the shorter BCH $(15, 5, 3)$ exhibits substantially higher FPR, reaching 0.097 at $T = 50$ and 0.182 at $T = 100$, indicating that shorter block lengths can weaken false-positive control under window shifting.

At $T = 100$, CORE-BREW can use the default BCH $(63, 7, 15)$ configuration, allowing one complete block that carries a 7-bit payload while achieving a TPR of 0.912, a payload bit accuracy of 0.912, and zero observed false positives. Shorter BCH configurations can fit multiple blocks within the same continuation, but this increased flexibility comes with a trade-off in false-positive control. Thus, adapting the code length enables CORE-BREW to operate on shorter generations, while the

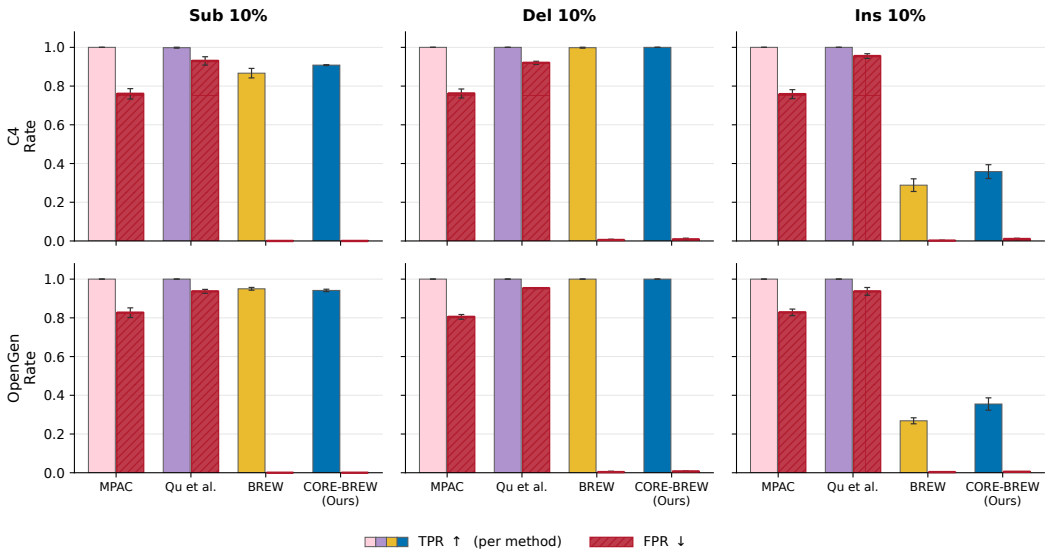

Figure 7: Detection performance under 10% substitution, deletion, and insertion attacks on Mistral-7B with C4 and OpenGen. Colored bars denote TPR and red bars denote FPR. MPAC and Qu et al. Qu et al. (2025) retain high TPR but exhibit substantially elevated FPR, whereas BREW Kim et al. (2026) and CORE-BREW maintain low observed FPR.

BCH configuration should be selected by balancing text length, payload capacity, and false-positive behavior.

D.10 Additional Results on Mistral-7B

Figure 7 provides additional robustness results on Mistral-7B using the same evaluation protocol as Section 6.3. The overall trends are consistent with those observed on OPT-1.3B: MPAC and Qu et al. Qu et al. (2025) achieve high TPR but incur high FPR, while BREW Kim et al. (2026) and CORE-BREW operate at substantially lower false-positive rates. Among the low-FPR methods, deletion is the least disruptive, with both BREW and CORE-BREW achieving near-perfect detection. Substitution yields strong detection performance, although the relative advantage between BREW and CORE-BREW varies by dataset. Insertion remains the most challenging attack, where CORE-BREW improves TPR over BREW on both C4 and OpenGen while maintaining low observed FPR.

D.11 Detailed Results under Synthetic Token-Level Attacks

Tables 9–11 provide detailed detection and payload recovery results under 10% substitution, deletion, and insertion attacks across OPT-1.3B and Mistral-7B on C4 and OpenGen. MPAC and Qu et al. Qu et al. (2025) generally retain high TPR and payload bit accuracy, but exhibit substantially higher FPR than BREW and CORE-BREW. In contrast, BREW and CORE-BREW maintain low observed FPR across the evaluated settings.

Under substitution, CORE-BREW generally improves TPR over BREW and consistently improves payload bit accuracy across models and datasets. Under deletion, both methods already achieve near-perfect TPR, while CORE-BREW provides consistently higher payload recovery. Insertion is the most challenging token-level attack, causing substantial degradation in both TPR and payload bit accuracy; nevertheless, CORE-BREW generally improves detection and consistently improves payload recovery over BREW while maintaining a mean observed FPR of at most 1%.

Overall, these results demonstrate that CORE-BREW yields the most pronounced performance gains under perturbations that severely disrupt the detector-facing bit sequence, whereas deletion-induced misalignment is effectively mitigated by the window-shifting mechanism alone.

Table 9: Detection and payload recovery performance under a 10% substitution attack. We report TPR, FPR, and payload bit accuracy on OPT-1.3B and Mistral-7B with C4 and OpenGen as mean ± standard deviation over multiple runs.

Model	Dataset	Method	TPR	FPR	Payload Bit Acc.
OPT-1.3B	C4	MPAC	1.0000 ± 0.0000	0.8317 ± 0.0184	0.9921 ± 0.0006
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9483 ± 0.0024	1.0000 ± 0.0000
		BREW	0.6550 ± 0.0367	0.0000 ± 0.0000	0.4892 ± 0.0255
		CORE-BREW	0.7017 ± 0.0249	0.0000 ± 0.0000	0.5492 ± 0.0120
	OpenGen	MPAC	1.0000 ± 0.0000	0.9083 ± 0.0024	0.9953 ± 0.0009
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9450 ± 0.0141	1.0000 ± 0.0000
		BREW	0.6583 ± 0.0232	0.0000 ± 0.0000	0.4708 ± 0.0228
		CORE-BREW	0.7217 ± 0.0246	0.0000 ± 0.0000	0.5817 ± 0.0174
Mistral-7B	C4	MPAC	1.0000 ± 0.0000	0.7600 ± 0.0268	0.9987 ± 0.0000
		Qu et al. Qu et al. (2025)	0.9983 ± 0.0024	0.9300 ± 0.0216	0.9933 ± 0.0024
		BREW	0.8667 ± 0.0246	0.0000 ± 0.0000	0.7933 ± 0.0143
		CORE-BREW	0.9083 ± 0.0024	0.0000 ± 0.0000	0.8733 ± 0.0012
	OpenGen	MPAC	1.0000 ± 0.0000	0.8267 ± 0.0249	0.9956 ± 0.0013
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9367 ± 0.0103	0.9983 ± 0.0024
		BREW	0.9500 ± 0.0071	0.0000 ± 0.0000	0.8908 ± 0.0096
		CORE-BREW	0.9417 ± 0.0062	0.0000 ± 0.0000	0.9075 ± 0.0074

Table 10: Detection and payload recovery performance under a 10% deletion attack. We report TPR, FPR, and payload bit accuracy on OPT-1.3B and Mistral-7B with C4 and OpenGen as mean ± standard deviation over multiple runs.

Model	Dataset	Method	TPR	FPR	Payload Bit Acc.
OPT-1.3B	C4	MPAC	1.0000 ± 0.0000	0.8450 ± 0.0187	0.9945 ± 0.0012
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9283 ± 0.0131	1.0000 ± 0.0000
		BREW	0.9983 ± 0.0024	0.0033 ± 0.0024	0.9900 ± 0.0061
		CORE-BREW	1.0000 ± 0.0000	0.0067 ± 0.0047	1.0000 ± 0.0000
	OpenGen	MPAC	1.0000 ± 0.0000	0.8833 ± 0.0201	0.9941 ± 0.0009
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9433 ± 0.0062	1.0000 ± 0.0000
		BREW	1.0000 ± 0.0000	0.0017 ± 0.0024	0.9883 ± 0.0031
		CORE-BREW	1.0000 ± 0.0000	0.0050 ± 0.0041	0.9992 ± 0.0012
Mistral-7B	C4	MPAC	1.0000 ± 0.0000	0.7617 ± 0.0232	0.9990 ± 0.0008
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9200 ± 0.0082	1.0000 ± 0.0000
		BREW	0.9983 ± 0.0024	0.0050 ± 0.0041	0.9767 ± 0.0062
		CORE-BREW	1.0000 ± 0.0000	0.0083 ± 0.0062	0.9967 ± 0.0047
	OpenGen	MPAC	1.0000 ± 0.0000	0.8050 ± 0.0122	0.9971 ± 0.0004
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9533 ± 0.0024	0.9983 ± 0.0024
		BREW	1.0000 ± 0.0000	0.0033 ± 0.0047	0.9767 ± 0.0012
		CORE-BREW	1.0000 ± 0.0000	0.0067 ± 0.0024	0.9958 ± 0.0024

D.12 Detailed Results under Paraphrasing Attacks

Table 12 provides the detailed numerical results corresponding to the paraphrasing evaluation discussed in the main text. MPAC Yoo et al. (2024) and Qu et al. Qu et al. (2025) retain near-perfect TPR and high payload bit accuracy under both T5 and DIPPER, but exhibit substantially higher FPR. In contrast, BREW Kim et al. (2026) and CORE-BREW maintain low observed FPR across both datasets. Compared with BREW, CORE-BREW consistently improves both TPR and payload bit accuracy under T5 and DIPPER while keeping the observed FPR below 1%. Although paraphrasing, particularly DIPPER, remains challenging for payload recovery, these results support the robustness gains from calibrated soft information over the hard-decision BREW baseline.

Table 11: Detection and payload recovery performance under a 10% insertion attack. We report TPR, FPR, and payload bit accuracy on OPT-1.3B and Mistral-7B with C4 and OpenGen as mean ± standard deviation over multiple runs.

Model	Dataset	Method	TPR	FPR	Payload Bit Acc.
OPT-1.3B	C4	MPAC	1.0000 ± 0.0000	0.8367 ± 0.0085	0.9912 ± 0.0007
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9417 ± 0.0125	1.0000 ± 0.0000
		BREW	0.4567 ± 0.0125	0.0000 ± 0.0000	0.2692 ± 0.0133
		CORE-BREW	0.5250 ± 0.0255	0.0083 ± 0.0062	0.3292 ± 0.0204
	OpenGen	MPAC	1.0000 ± 0.0000	0.8767 ± 0.0262	0.9933 ± 0.0008
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9550 ± 0.0108	1.0000 ± 0.0000
		BREW	0.4550 ± 0.0389	0.0033 ± 0.0047	0.2592 ± 0.0283
		CORE-BREW	0.5250 ± 0.0122	0.0100 ± 0.0000	0.3292 ± 0.0176
Mistral-7B	C4	MPAC	1.0000 ± 0.0000	0.7583 ± 0.0232	0.9968 ± 0.0009
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9550 ± 0.0122	0.9967 ± 0.0024
		BREW	0.2883 ± 0.0327	0.0017 ± 0.0024	0.1558 ± 0.0193
		CORE-BREW	0.3583 ± 0.0357	0.0100 ± 0.0041	0.2108 ± 0.0204
	OpenGen	MPAC	1.0000 ± 0.0000	0.8283 ± 0.0170	0.9947 ± 0.0014
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9367 ± 0.0201	0.9967 ± 0.0024
		BREW	0.2683 ± 0.0155	0.0033 ± 0.0024	0.1492 ± 0.0103
		CORE-BREW	0.3550 ± 0.0319	0.0050 ± 0.0000	0.2058 ± 0.0208

Table 12: Detection and payload recovery performance under T5 and DIPPER paraphrasing attacks on OPT-1.3B with C4 and OpenGen at $T = 500$. We report TPR, FPR, and payload bit accuracy as mean ± standard deviation over multiple runs.

Attack	Dataset	Method	TPR	FPR	Payload Bit Acc.
T5	C4	MPAC	1.0000 ± 0.0000	0.8250 ± 0.0071	0.9954 ± 0.0003
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9400 ± 0.0187	1.0000 ± 0.0000
		BREW	0.6850 ± 0.0187	0.0000 ± 0.0000	0.4683 ± 0.0150
		CORE-BREW	0.7450 ± 0.0122	0.0083 ± 0.0062	0.5608 ± 0.0326
	OpenGen	MPAC	1.0000 ± 0.0000	0.8883 ± 0.0085	0.9934 ± 0.0023
		Qu et al. Qu et al. (2025)	1.0000 ± 0.0000	0.9567 ± 0.0024	1.0000 ± 0.0000
		BREW	0.7733 ± 0.0332	0.0000 ± 0.0000	0.5575 ± 0.0159
		CORE-BREW	0.8067 ± 0.0085	0.0083 ± 0.0024	0.6233 ± 0.0059
DIPPER	C4	MPAC	0.9967 ± 0.0024	0.8433 ± 0.0062	0.9750 ± 0.0033
		Qu et al. Qu et al. (2025)	0.9967 ± 0.0024	0.9417 ± 0.0062	0.9633 ± 0.0062
		BREW	0.4850 ± 0.0108	0.0033 ± 0.0024	0.3050 ± 0.0134
		CORE-BREW	0.5667 ± 0.0165	0.0050 ± 0.0041	0.3967 ± 0.0062
	OpenGen	MPAC	0.9917 ± 0.0062	0.9100 ± 0.0108	0.9621 ± 0.0049
		Qu et al. Qu et al. (2025)	0.9950 ± 0.0041	0.9267 ± 0.0047	0.9350 ± 0.0122
		BREW	0.5150 ± 0.0248	0.0017 ± 0.0024	0.3422 ± 0.0187
		CORE-BREW	0.5983 ± 0.0118	0.0067 ± 0.0024	0.4042 ± 0.0171

E Implementation Details and Reproducibility

This appendix summarizes the implementation choices and settings required to reproduce our results, with special emphasis on: (i) the implementation of CORE-BREW and its hard-decision ablation (Sections 4–6), (ii) entropy-aware safeguards for low-entropy contexts, including erasure handling and bias control, (iii) detection thresholds, score-based rejection, and FPR–TPR evaluation, and (iv) the licenses and terms of use for external assets used in our experiments. Our implementation builds on standard open-source tooling for deep learning and text processing Paszke et al. (2019); Wolf et al. (2020).

Table 13: Inference-time computational efficiency on OPT-1.3B with C4. Generation time is reported per prompt, generation overhead is measured relative to unwatermarked decoding, and detection latency is reported in milliseconds.

Method	Gen. time (s)	Tok./s	Gen. overhead	Detect. (ms)
Unwatermarked	7.77 ± 0.22	64.4	–	–
MPAC ($\delta = 3$)	59.19 ± 2.11	8.4	$+661.6\%$	631 ± 105
Qu et al. Qu et al. (2025) ($\delta = 3$)	21.11 ± 2.67	24.0	$+171.6\%$	27154 ± 8091
BREW ($\delta = 3$)	11.24 ± 0.19	44.5	$+44.7\%$	511 ± 77
CORE-BREW w/o Soft Decoding	8.77 ± 0.14	57.0	$+12.8\%$	$\mathbf{491 \pm 125}$
CORE-BREW	$\mathbf{8.65 \pm 0.15}$	**57.8**	$\mathbf{+11.3\%}$	856 ± 414

E.1 Codebase Organization

The codebase is organized as a Python package. The implementation follows the block-wise designated-codeword framework underlying BREW Kim et al. (2026), together with ECC-based components shared with related multi-bit watermarking methods Qu et al. (2025). The package includes modules for: (i) fixed keyed vocabulary partitioning and block-dependent payload-to-codeword construction, (ii) embedding, including logit interception and Constant Hit-Rate biasing, (iii) extraction and detection, including bit extraction, window shifting, decoding, and scoring, (iv) attack pipelines for token edits and paraphrasing, and (v) evaluation, including metric computation and result aggregation. Entry-point scripts provide executable examples for running generation and detection and producing the corresponding output summaries. The accompanying implementation is intended to support reproducibility of the main CORE-BREW pipeline, while full-scale table and figure generation relies on the saved artifacts and evaluation logs described below.

To support reproducibility and auditing, we save intermediate artifacts, including generated texts (watermarked and unwatermarked), attacked texts, per-block match decisions, and, when applicable, per-block scores. This allows evaluation and plotting to be rerun without regenerating text.

E.2 Software and hardware environment

Software stack. All experiments are implemented in Python and use: PyTorch for model execution Paszke et al. (2019), Hugging Face Transformers for model loading and generation Wolf et al. (2020), and SentencePiece for models requiring subword tokenization Kudo & Richardson (2018). We use standard scientific Python utilities (NumPy/SciPy) for evaluation and statistics.

Hardware. Generation is run on CUDA-enabled GPUs when available. Detection can be run on CPU for offline or batched evaluation, although model-aware erasure computation requires access to the generator or an equivalent scoring model.

Compute resources. All experiments are inference-time evaluations; no model training or fine-tuning is performed. Experiments were run on a local CUDA 13.0 server with four NVIDIA RTX PRO 6000 Blackwell GPUs, each with 96GB of VRAM, an Intel Xeon 6740P CPU with 96 physical cores, and 1.5TB of RAM. To characterize inference-time efficiency, we measure per-prompt generation latency, throughput, generation overhead relative to unwatermarked decoding, and detection latency. Table 13 shows that CORE-BREW requires 8.65 s per prompt, corresponding to 57.8 tokens/s and an 11.3% generation overhead. Its hard-decision ablation has nearly identical generation cost (8.77 s, 57.0 tokens/s, and 12.8% overhead), and both are substantially more efficient during generation than MPAC, Qu et al. Qu et al. (2025), and BREW. The additional detection cost of CORE-BREW includes model-aware probability recomputation, LLR-based scoring, and Chase-style list decoding: CORE-BREW requires 856 ms per prompt on average, compared with 491 ms for the hard-decision ablation. Nevertheless, CORE-BREW completes detection in under one second on average and remains substantially faster than Qu et al. Qu et al. (2025), which requires approximately 27.2 s per prompt. Overall, CORE-BREW introduces modest generation overhead while trading additional detection cost for model-aware soft-decision processing.

Reproducibility and Determinism. Exact bitwise determinism can be affected by nondeterministic GPU kernels and external paraphrasers. We therefore report averages over multiple seeds and log the seeds, configurations, generated outputs, and detection summaries needed to audit the reported results.

E.3 Models, Tokenization, and Datasets

Models and checkpoints. We evaluate publicly available autoregressive LLM checkpoints loaded via Transformers Wolf et al. (2020). Model and tokenizer identifiers are recorded in each run's configuration.

Tokenization. All watermark operations, including partition membership, bit extraction, and block reconstruction, are performed on token IDs. Word-level attacks such as synonym substitution are filtered according to the token-level constraints required by each experiment.

Datasets and prompt sampling. We use C4 as the long-form corpus and OpenGen, consisting of short two-sentence prompts derived from WikiText-103, as the short-form prompt set. Prompt splits are kept disjoint across development and test evaluation. All reported numbers are computed on test prompts, and the same splits are reused across schemes and attacks for fair comparison.

E.4 Asset Licenses and Terms of Use

Table 14 summarizes the external assets used in our experiments. We use all assets under their stated licenses or terms of use and do not redistribute pretrained model checkpoints or third-party baseline code. The supplementary package includes small processed prompt/evaluation subsets derived from existing public datasets; the original dataset licenses and terms of use apply to these derived files.

E.5 Generation and watermark embedding configuration

Sampling settings. Unless otherwise stated, generation uses temperature 1.0 and nucleus (top-p) sampling with $p = 0.9$ Holtzman et al. (2020). We log all decoding parameters (temperature, top-p, max new tokens, stop criteria). Constant Hit-Rate calibration targets the biased pre-top-p distribution, while the realized hit rate after nucleus sampling may differ slightly.

ECC and block parameters. We use a binary BCH code $C \subseteq \{0,1\}^n$ with code parameters (n, k, t) as specified in Section 6 and the appendix ablations Lin & Costello (2004); Richardson & Urbanke (2008). The block length is set to n. The payload $m \in \{0,1\}^k$ is either fixed per run or sampled per text, and the mapping from payload to per-block designated codewords $c^{(j)}$ is deterministic given the key and block index.

Constant Hit-Rate embedding. We implement the Constant Hit-Rate embedding rule defined in Section 4. At each generation step, the base target-list mass is computed from the model logits, with m_t clamped to $[\varepsilon, 1-\varepsilon]$ for numerical stability. The resulting bias is then subject to the distortion guards described below.

Distortion guards and erasures. We apply the distortion-aware safeguards defined in Section 4 and implemented in Appendix B, Algorithms 1 and 3. The raw bias is clipped to $[-\delta_{\max}, \delta_{\max}]$, and positions that fail the peakness, entropy, or post-bias feasibility checks are left unbiased and marked as erasures. Erased positions contribute zero LLR during model-aware detection. We log the erasure rate and tail statistics of $|\delta_t|$ for diagnostic analysis.

E.6 Detection Implementation

Detection follows BREW's block-wise designated-codeword framework with window shifting Kim et al. (2026). For each block, we test candidate offsets $s \in \{-s_{\max}, \ldots, s_{\max}\}$ around the nominal block anchor and construct the corresponding hard-decision or LLR representation (Appendix B, Algorithm 3). CORE-BREW uses calibrated LLR evidence with score-based rejection and Chase-style list decoding. To evaluate the contribution of soft-decision processing, we additionally consider

Table 14: External assets used in our experiments. We list dataset and model assets separately because the corresponding licenses and terms differ by asset. Detailed URLs, revisions, commit hashes, and package versions are provided in the supplementary README.

Asset	Role	License / terms	Use in this work
OPT-1.3B	Target LLM	OPT license / model terms	Used for inference-time evaluation; checkpoint not redistributed.
Mistral-7B-v0.3	Target LLM	Apache-2.0	Used for inference-time evaluation; checkpoint not redistributed.
C4 processed subset	Prompt/evaluation corpus	ODC-BY 1.0; Common Crawl terms	Processed prompt/evaluation subset included in the supplementary package; original terms apply.
OpenGen processed subset	Derived prompt/evaluation corpus	Terms inherited from the source corpus	Derived from WikiText-103 for prompt-based evaluation; applicable source terms apply.
WikiText-103	Source corpus	CC BY-SA / GFDL according to the source release	Used to construct the derived OpenGen prompt/evaluation subset.
MarkLLM	Evaluation framework	Apache-2.0	Used as a unified watermarking evaluation framework with attribution.
MPAC implementation	Baseline	Stated repository license	Used for research comparison; original notices preserved where applicable.
Qu et al. implementation	Baseline	Stated repository license	Used for research comparison; original notices preserved where applicable.
BREW implementation	Baseline	Stated repository license	Used for comparison; original notices preserved where applicable.
PyTorch; NumPy; SciPy	Software libraries	BSD-style licenses	Used for model inference, numerical computation, and evaluation.
Transformers; SentencePiece	Software libraries	Apache-2.0	Used for loading pretrained models and tokenizers.

a hard-decision ablation that uses the same Constant Hit-Rate embedding and erasure decisions, but fills erased positions with key-derived PRF bits and applies bounded-distance designated-codeword verification.

Per-bit LLRs. Under Constant Hit-Rate embedding, each non-erased position has constant LLR magnitude

$$\lambda = \log\left(\frac{p^\star}{1-p^\star}\right),$$

with sign determined by keyed list membership. Erased positions contribute zero LLR.

Hard-decision ablation. The hard-decision ablation retains bounded-distance designated-codeword verification on the PRF-filled hard-decision representation. For each candidate offset, erased positions are filled with key-derived PRF bits, and the resulting hard-decision vector is evaluated using a bounded-distance BCH decoder. A block is accepted only when bounded-distance BCH decoding returns the designated codeword. This ablation does not use likelihood-based scoring or list decoding and provides a hard-decision reference for evaluating the contribution of soft-decision processing under the same Constant Hit-Rate embedding and erasure decisions.

CORE-BREW soft-decision detection. CORE-BREW preserves erasures as zero-LLR positions and combines calibrated LLR scoring, designated-codeword verification, and Chase-style list decoding. For each candidate offset, we compute the designated-codeword score

$$S^{(j)}(s) = \sum_{b=0}^{n-1} (2c^{(j)}[b] - 1)\Lambda_b^{(j,s)},$$

and require the candidate score to exceed a block-level threshold τ_{blk}. The Chase-style list-decoding wrapper explores a bounded set of candidate hard decisions obtained from the least-reliable positions, applies bounded-distance decoding to each candidate, and evaluates the resulting codewords using the LLR-consistent score Chase (1972); Lin & Costello (2004); Richardson & Urbanke (2008). A block is accepted only when a decoded candidate equals the designated codeword and satisfies the score threshold. This procedure can recover the designated codeword beyond the bounded-distance radius, although beyond-radius recovery is not the sole source of detection gain; erasure-aware scoring and LLR-consistent candidate evaluation can also improve discrimination. The score threshold therefore serves as an additional filtering criterion within the overall acceptance rule.

Detection thresholds and list-decoding parameters. For text-level detection, let M_{match} denote the number of accepted blocks among M complete blocks. A sample is classified as watermarked when

$$\frac{M_{\mathrm{match}}}{M} \geq \theta.$$

For CORE-BREW, a shifted candidate block is accepted only if the Chase-decoded codeword equals the designated codeword and its LLR score exceeds the block-level threshold τ_{blk}. Unless otherwise specified, we set $\tau_{\mathrm{blk}} = 5.0$. For Chase-style list decoding, we sort bit positions by increasing reliability $|\Lambda_b|$, select the ℓ least reliable positions as flip candidates, and enumerate flip patterns up to a maximum list size L. We use $\ell = 2$ and $L = 32$, resulting in at most $\min(2^\ell, L) = 4$ candidate hard-decision vectors per shifted block. For insertion and deletion attacks, we additionally search over $2s_{\max} + 1$ shifted alignments per block.

Detector capability setting. When model-aware erasures are enabled, the detector reapplies the same erasure rule to the observed prefix using the base model or an equivalent scoring model, rather than assuming access to the original generation-time erasure mask. Because the observed prefix may differ from the generation-time prefix after edits, the resulting detection-time erasure decisions need not coincide with those made during embedding. This setting is most appropriate for provider-side verification. For token-only detection, erasures are disabled, and the detector uses the constant-magnitude LLR rule without prefix-probability recomputation. We distinguish these detector-access settings explicitly in the corresponding experiments.

E.7 Threshold Selection and Reporting

Reporting protocol. We report detection performance using text-level TPR and FPR. Unless otherwise stated, each method is evaluated using its default or recommended detection threshold specified in the corresponding experiment. When threshold sensitivity is evaluated, we report TPR and FPR at each tested operating point.

Variability across runs. Unless otherwise stated, we report mean $\pm$ standard deviation over multiple random seeds. The reported variability reflects randomness from prompt sampling, watermark key/payload sampling, model generation, and attack procedures.

Threshold sweeps. For hard-decision methods such as BREW and the hard-decision ablation, threshold sweeps vary the text-level match threshold θ. For CORE-BREW, sweeps may vary the block-level score threshold τ_{blk} and the text-level threshold θ. Unwatermarked texts are used to estimate the corresponding FPR.

FPR Evaluation under Perturbed Null Distribution (H_0). To assess whether false positive behavior remains stable under distribution shift, we apply the same attack pipelines to unwatermarked texts and report the resulting attacked-H_0 FPR. This is particularly important for score-based detectors, whose score distributions may shift under paraphrasing or token-level edits.

E.8 Attack pipelines and evaluation metrics

Token-level edits. We implement substitution/insertion/deletion pipelines inspired by TextAttack Morris et al. (2020). For token-preserving substitution, we restrict to synonym replacements that preserve tokenization length to maintain alignment. For insertion/deletion-like attacks, we allow length changes and rely on window shifting during detection.

Paraphrasing. Paraphrasing attacks use a separate paraphraser model (distinct from the generator) following common paraphrase evaluation setups Wieting & Gimpel (2018); Krishna et al. (2023). The paraphraser does not receive the watermark key. We quantify paraphrase strength using BLEU Papineni et al. (2002) and semantic similarity via BERTScore (or a comparable embedding-based metric) Zhang et al. (2020).

Metrics. We report text-level TPR and FPR for watermark detection, payload bit accuracy for payload recovery, and PPL, BLEU, and BERTScore for text quality.

E.9 Random Seeds, Logging, and Released Artifacts

We control and record randomness across prompt sampling, watermark key generation, payload sampling, model sampling RNG states (Python/NumPy/PyTorch CPU and CUDA), and attack procedures. We log all hyperparameters in machine-readable configuration files (YAML/JSON), including the model checkpoint, dataset split, (n, k, t), $p^\star$, $s_{\max}$, θ, τ_{blk}, $\delta_{\max}$, $q_{\max}$, list-decoding parameters, and generation/attack settings. We release (or will release upon acceptance, consistent with policy) scripts to: (i) regenerate watermarked and unwatermarked corpora from prompts, (ii) apply attack pipelines, (iii) run detection and threshold sweeps, and (iv) reproduce the reported figures and tables from saved artifacts.

F Limitations, Ethics, and Threat Model Clarification

This appendix clarifies the threat model and limitations of our approach, and discusses ethical considerations. We clarify three aspects of the proposed framework: (i) the distinction between CORE-BREW's soft-decision detector and its hard-decision ablation, (ii) the practical risks of distortion in low-entropy contexts and our safeguards, and (iii) the limits of robustness under strong semantic rewriting.

F.1 Threat model and deployment assumptions

Keyed provenance, not universal detection. Our objective is *keyed* provenance verification: given a secret key K, a provider (or authorized verifier) can test whether a text was generated by a watermarked model under that key. This is distinct from open-set, keyless classifiers for "AI-generated" text and does not attempt to detect text generated by arbitrary models without key access Kirchenbauer et al. (2023); Mitchell et al. (2023).

Detector capabilities: token-only vs. model-aware. We distinguish two detector capability settings for designated-codeword verification. In both settings, verification uses the observed text, the secret key K, and the designated codewords $\{c^{(j)}\}$ associated with the payload under test. The distinction concerns access to model probabilities.

- **Token-only setting:** in principle, observed tokens can be mapped to hard bits using the fixed keyed partition and evaluated using the constant-magnitude LLR rule without model access, with model-dependent erasure checks disabled.
- **Model-aware setting:** the detector additionally uses the base model (or an equivalent scoring model) to recompute token probabilities on the observed prefix and apply the entropy-aware erasure rule (Sections 4–6 and Appendix E).

Our erasure-aware CORE-BREW experiments use the model-aware setting, which is practical for provider-side provenance verification.

Adversary goals and knowledge. We assume an adversary can post-process the text via paraphrasing, synonym substitution, insertion, deletion, or reformatting, as commonly studied in watermark robustness evaluations Morris et al. (2020); Wolff & Wolff (2020). We primarily consider *black-box* adversaries who do not know the key K. A fully informed attacker who knows K and can exactly simulate the detector could, in principle, remove the watermark by targeted rewriting; like most keyed watermarking schemes, we do not claim robustness against such a worst-case white-box adversary.

F.2 Limitations of the channel model

Idealized BSC and independence assumptions. Our theory models fully calibrated, non-erased positions under Constant Hit-Rate embedding as approximately following a position-homogeneous binary symmetric channel with parameter $p^\star$. While channel shaping reduces context dependence relative to fixed-bias watermarking, real LLM generation is not strictly i.i.d.: token distributions depend on long-range context, and adversarial edits introduce correlated changes. Consequently, the BSC model and approximate block independence are best viewed as *analysis tools* rather than exact realities. We therefore complement theoretical bounds with empirical evaluations of the observed FPR–TPR trade-off in Section 6. For deployment, detection thresholds should be validated using in-domain unwatermarked text.

Fixed partitions and hard-decision interpretation. Our evaluated instantiation uses a fixed keyed vocabulary partition so that a token's partition bit remains unchanged when its position shifts under insertion/deletion attacks. However, reuse of the same partition can introduce dependencies through repeated tokens and natural-language correlations, so aggregate FPR bounds should be interpreted as idealized analysis tools complemented by empirical FPR evaluation on unwatermarked text. Similarly, the hard-decision ablation applies bounded-distance verification to the detector's PRF-filled hard-decision representation, which may differ from the raw token-derived hard vector when erased positions are replaced.

Shift search limitations. Window shifting partially addresses insertion/deletion noise by searching a bounded set of alignments. However, large or highly structured edits can break local alignment assumptions and reduce match rates. More powerful synchronization mechanisms (e.g., learned alignment or marker-based synchronization) are outside our scope.

F.3 Limitations of Robustness under Semantic Rewriting

Paraphrasing and semantic rewriting can eventually destroy most token-level watermarks, including ours, if the attacker is allowed sufficient edit budget and can rewrite aggressively Wolff & Wolff (2020); Krishna et al. (2023). Our goal is to improve robustness in the practically relevant regime of *moderate* edits while maintaining low false-positive rates. We evaluate the resulting FPR–TPR behavior under paraphrasing attacks to characterize degradation under semantic rewriting (Section 6). We do not claim unconditional robustness to arbitrary meaning-preserving transformations, which remains an open challenge for token-level watermarking.

F.4 Low-Entropy Contexts and Quality Risks

Enforcing a fixed hit-rate in low-entropy contexts can require large logit shifts when the base model assigns very small probability mass to the target list. Such shifts may affect fluency, factuality, or style Holtzman et al. (2020). To mitigate this risk, we use bias control and feasibility-based erasures

(Section 4), and analyze their effects through text-quality and erasure ablations in Appendices D.6 and D.7. These safeguards trade off watermark evidence for reduced distortion: when many positions are erased, the effective evidence per block decreases, so longer text or more blocks may be needed for reliable detection.

F.5 Acceptance Regions under Hard- and Soft-Decision Decoding

A persistent technical ambiguity in soft-decision watermarking is whether one can simultaneously (i) preserve the hard-decision bounded-distance acceptance region exactly, thereby retaining the corresponding combinatorial false-positive analysis, and (ii) exploit soft LLR information to improve decoding beyond what is available from the raw hard decision. CORE-BREW separates these roles through its hard-decision ablation and soft-decision detector.

The **CORE-BREW w/o Soft Decoding** ablation retains the same Constant Hit-Rate embedding and erasure decisions as CORE-BREW, but applies bounded-distance designated-codeword verification to the PRF-filled hard-decision representation. Its acceptance region is therefore restricted to the bounded-distance decoding neighborhood of the designated codeword under the corresponding idealized analysis assumptions.

In contrast, **CORE-BREW** uses calibrated LLR scoring and limited Chase-style list decoding Chase (1972); Lin & Costello (2004). Its acceptance rule is not restricted to the radius-t neighborhood of the raw hard-decision vector: a bounded set of candidate hard decisions is explored, and a block is accepted only when a decoded candidate equals the designated codeword and its score exceeds the block-level threshold τ_{blk}. Consequently, the soft-decision acceptance rule may extend beyond the hard-decision bounded-distance region and is analyzed separately using score-based bounds together with empirical FPR evaluation. The corresponding analysis is provided in Section 5, with empirical false-positive behavior reported in Section 6.

This distinction provides a matched hard-decision reference for evaluating the contribution of soft-decision processing without treating the ablation as a separate CORE-BREW configuration.

F.6 Ethical considerations and potential misuse

Intended use. Our primary intended use is provenance verification for transparency and accountability, including attribution of model-generated content, auditing of policy compliance tags, and tracing misuse when a model provider is responsible for outputs. The designated-codeword design targets extremely low FPR to reduce the risk of falsely accusing a human author.

Misuse and over-reliance. Watermarks should not be treated as a sole or definitive indicator of authorship. Watermarks can be removed or corrupted, and absence of a watermark does not imply human authorship. Over-reliance can lead to erroneous conclusions, especially under distribution shift or when the detector is used outside its validated domain. We therefore recommend using watermark signals as one component in a broader provenance and safety pipeline.

Privacy and data handling. Multi-bit payloads can encode identifiers. Even when keys are secret, embedding persistent identifiers may raise privacy concerns. We recommend minimizing payload content, adopting rotation/expiration policies for identifiers, and restricting detector access. We also emphasize that our evaluation does not require training on private user data; it uses public corpora and model outputs.