Enhancing Robustness of Cryptographic Watermarking in AI-Generated Content

Anonymous ACL submission

Abstract

Watermarking AI-generated text enables provenance verification but remains vulnerable to text modifications. Existing methods typically rely on local seed dependencies, making them sensitive to token substitutions and targeted edits. To improve robustness, we introduce a majority voting-based seeding algorithm that stabilizes seed generation, significantly enhancing detectability under text modifications. This method is compatible with existing cryptographic watermarking schemes, such as Gumbel Sampling, SynthID, and Soft Red List. Additionally, we propose PRF+Scaling, which provides explicit control over watermark signal strength, balancing detectability and text quality.

1 Introduction

With the rapid advancement of large language models (LLMs), AI-generated text has become increasingly indistinguishable from human-written content. This shift raises concerns about content authenticity, misinformation detection, and intellectual property protection. To address these challenges, watermarking techniques have been proposed to embed verifiable signals into generated text, allowing for post-hoc detection and attribution. More recent approaches have leveraged probabilistic watermarking, where token selection is conditioned on a pseudo-random function (PRF) that alters the likelihood of choosing certain tokens while preserving fluency. Gumbel sampling (Aaronson, 2022) and SynthID (Dathathri et al., 2024) have demonstrated that such probabilistic watermarks can be embedded at generation time and later verified with high confidence.

Despite these advances, existing LLM watermarking methods still face significant limitations that reduce their robustness and practicality. Most current watermarking techniques rely on a seed-based PRF to determine the perturbed probability of the next token, where the seed generation is a deterministic function of the preceding tokens. This deterministic design raises a significant issue of the robustness: Even minor text edits—such as token replacements—can completely regenerate the seed, leading to the loss of watermark signals. Existing detection algorithms rely on the accumulation of watermarking signals, and the signals decay as more aggressive edits are performed.

To address this issue, we propose a stable seeding algorithm that significantly enhances watermark robustness. Rather than relying on a deterministic process, our approach applies a majority voting mechanism to generate a more stable seed, reducing the risk of watermark loss due to localized text edits. This approach is compatible with existing watermarking methods, including Gumbel Sampling, SynthID, and Soft Red List (Kirchenbauer et al., 2023), allowing it to strengthen watermark robustness across different frameworks. Our stable seeding algorithm improves robustness to text edits, such as token replacements, synonym substitutions, and character-level modifications. Unlike traditional deterministic seeding, which is highly sensitive to these modifications, our approach ensures better seed stability and improved detection rates.

Another challenge in existing watermarking methods is the lack of direct control over signal strength. While some approaches, such as SynthID, increase watermark detectability by adding more PRFs, this indirect method is computationally costly and lacks fine-grained control. Similarly, the Soft Red List method modulates watermark strength by adjusting the proportion of red and green list tokens, but this can largely degrade generation quality especially in low-entropy cases. Therefore, we introduce ScaledPRF, a simple yet effective technique for explicitly modulating watermark strength. Instead of relying on additional PRFs, ScaledPRF scales the PRF-based perturba-

tion directly, providing a tunable balance between watermark detectability and text fluency.

To provide a more complete assessment of watermarking trade-offs, we test the text quality with and without watermark with five evaluation metrics, showing the subtle balance between the text quality and the watermarking robustness.

2 Related Works

Watermarking techniques for large language models can be broadly classified into zero-bit watermarking and multi-bit watermarking. Zero-bit watermarking embeds an implicit signal in generated text, allowing for post-hoc detection of AI-generated content without encoding explicit information. On the other hand, multi-bit watermarking encodes a multi-bit message. (Zhao et al., 2024a) provides a comprehensive systematization of knowledge of existing watermarking schemes. Our work focuses on enhancing zero-bit watermarking by improving seeding robustness and introducing direct watermark strength modulation.

A widely adopted approach in zero-bit watermarking is logit-based token probability modification, which adjusts token probabilities before sampling. The soft red list method (Kirchenbauer et al., 2023; Zhao et al., 2023) promotes certain tokens (green list) while suppressing others (red list) to introduce a watermark signal. While effective, this method inherently modifies token probabilities, potentially altering the natural distribution of generated text. Later work by (Wu et al., 2024) proposed an alternative red-green list-based method that preserves the original distribution in expectation, offering a theoretically non-distortionary alternative.

Another prominent technique is Gumbel-max sampling (Aaronson, 2022), which perturbs logits by adding Gumbel noise before applying softmax, influencing token selection in a probabilistically verifiable manner. A related approach, PF sampling (Zhao et al., 2024b), adapts this idea to permutation-based decoders, which enforce a more greedy decoding strategy compared to traditional softmax sampling.

SynthID (Dathathri et al., 2024) applies a structured filtering mechanism, where a sequence of PRFs is used to iteratively refine token selection probabilities. This design enhances robustness against text perturbations, as different PRFs contribute independent signals to the watermark. Another line of work explores deterministic non-distortionary watermarking methods, such as Inverse Transform Sampling (ITS). While ITS avoids modifying token probabilities, it has been shown to exhibit lower detectability compared to probabilistic methods.

Recent work by (Qu et al., 2024) introduces a multi-bit watermarking scheme that provides provable robustness guarantees under text modifications. By leveraging error-correcting codes, their approach enhances extraction accuracy even in the presence of edits while balancing robustness and efficiency. Their results demonstrate that detectability remains high under text edits, addressing a common limitation in multi-bit watermarking techniques.

Christ et al. (Christ and Gunn, 2024) propose a robust watermarking scheme that leverages pseudorandom error correction codes, providing strong protection against replacements or random deletions. However, because their method relies on a repetition code for handling deletions, it becomes vulnerable when a continuous block of tokens—such as an entire sentence or paragraph—is removed, which is among the most common attacks on watermarking schemes. In their earlier work (Christ et al., 2024), they introduce the notion of a public key watermark, where anyone possessing the private key can generate the watermark, but those holding the public key can only do verification.

Compared to their approach, our work focuses on zero-bit watermarking, aiming to improve seeding robustness and watermark strength control. Instead of creating a new watermarking scheme with better robustness, we provide a complementary approach to enhance robustness of existing watermark schemes. Our approach applies to most watermarking scheme that rely on PRF for probability purturbation. What's more, we provide an alternative watermarking scheme with a tuneable trade-off between watermark detectability and text quality.

3 Preliminaries

3.1 Language Model Watermarking

Language models generate next token in an autoregressive manner. During inference, the model iteratively computes a logit score for each potential next token and convert it into a probability distribution, denoted by p. In zero-bit watermarking, a model owner embeds a secret signal using the secret key

sk into generated text. Watermarking algorithms modify the token sampling process to introduce identifiable yet imperceptible patterns in generated text. Specifically, the probability perturbation of each token relies on a PRF with its seed generated from previous n tokens (so-called n-gram):

$$\hat{p} := f\left(p, \text{PRF}(\text{seed}, sk)\right),$$

where f is an univeral transformation of the original probability p and the PRF. Existing watermarking schemes aim to design the transformation f to ensure both traceability and natural language fluency. For example, the Soft Red List (Kirchenbauer et al., 2023) modifies the token probability by adding a fixed δ bonus to the logits of the green list tokens while subtracting δ for red list tokens.

During detection, a verifier with the secret key sk and an output sequence $\mathbf{t}$ determines whether $\mathbf{t}$ originates from the watermarked model.

3.2 Desirable Properties of Watermarking

Effective watermarking balances undetectability, verifiability, and robustness:

Undetectability: The watermark should preserve text fluency and coherence, making only minimal adjustments to $p_k(v)$. The generated text should remain indistinguishable from non-watermarked outputs under casual inspection.

Verifiability: Anyone with the watermarking scheme and associated PRF can verify the watermark without requiring access to the original model.

Robustness: The watermark should remain detectable even after text modifications such as synonym replacements, minor edits, or adversarial perturbations.

4 Methodologies

We introduce our stable seeding algorithm in Section 4.1 and the ScaledPRF method in Section 4.2.

4.1 Stable Seeding Algorithm

Most existing watermarking schemes perform slight perturbations on the token generation probability, and the probability perturbation of each token relies on the token itself and a seed generated from previous n tokens. Prior work primarily focuses on the sampling side—how to embed bits into the generation process—while relying on simple deterministic seed generation. This design choice makes current methods highly sensitive to minor text modifications: even a single token change can fully alter the seed, leading to degradation of the watermark signal.

We seek to stabilize watermarking signals and identify robust seed generation as a key but largely overlooked challenge in the literature. In this work, we address this gap by introducing a seed stabilization algorithm that enhances robustness against localized edits. Our approach aggregates information from multiple preceding tokens to derive a stable seed. Specifically, we hash each token in the n-gram individually and form a bit matrix, and we compute the majority vote across each bit column. This bitwise voting produces a seed that is more resilient to local perturbations such as deletions, substitutions, swaps, or insertions (e.g., emojis), as small changes are unlikely to flip the majority outcome across all bits. This design significantly enhances watermark detectability under text edits, making it a more effective approach than traditional error correction-based methods in our setting. We present our algorithm in Algorithm 1.

Formally, let $T = \{t_1, t_2, \ldots, t_n\}$ represent an n-gram sequence. We apply a hash function $H(t_i)$ to each token t_i, obtaining an m-bit hash $H(t_i) \rightarrow \{0,1\}^m$. We construct an $n \times m$ matrix M, where each row corresponds to a token's hash output: $M[i][j] = H(t_i)[j], \quad \forall i \in [1, n], j \in [1, m]$.

For each bit position j, we compute the majority vote across all rows:

$$s[j] = \text{majority}(M[:, j]).$$

The final seed s is obtained by concatenating these majority-voted bits, producing an m-bit value used for watermarking. There are optional approaches to make the protocol more resilliant to hash-relevant brute-force attack, namely, attackers can brute-force all input-output pairs of a hash function. To defend it, we can apply keyed hash functions to significantly increase the brute-force space. Meanwhile, we can set another key k_s and redefine the final seed s to be the concatenation $s||k_s$, both approaches increase the the difficulty of attacks tremendously.

Crucially, our stable seeding mechanism improves robustness at the seeding layer without altering the downstream sampling layer. It is therefore compatible with existing probabilistic watermarking methods such as Gumbel Sampling, SynthID, and Soft Red List. By replacing deterministic seeds with statistically stable ones, our approach offers

Algorithm 1 Stable Seeding Algorithm

Requires:
Hash function $H(t) \to [0,1]^m$;
Inputs:
N-gram $T = \{t_1, t_2, \ldots, t_n\}$.
Outputs:
an m-bit seed s

```
Initialize the n × m matrix M.
Initialize s = 0.
for token index i in [1,n] do
    h_i = H(t_i)
    for index j in [1,m] do
        M[i][j] = h_i[j] (j-th bit of h_i).
    end for
end for
for position index j ∈ [1, m] do
    Do majority vote on the j-th column of M, and set the result as the j-th bit of s.
end for
    return s
```

robustness gains without increasing computational complexity.

It is worth noting that all existing PRNG-based watermarking schemes are inherently not resistant to large-scale paraphrasing even with our algorithm. Robustness against paraphrasing would require fundamentally different, often semantic-level watermarking approaches (Hou et al., 2023; Ren et al., 2023; Liu et al., 2024), which align watermarking signals with semantic embeddings rather than token identities. Instead, our method targets robustness under common text editing operations—even those that alter semantics—without relying on additional models or embedding computations. There are various practical scenarios (e.g., copyright protection, traceability, content attribution) where large-scale paraphrasing falls outside the attack model, therefore, our scheme fits better with this scenario.

We provide the theoretical analysis of our algorithm as follows.

Theorem 1 (One-bit Robustness Under Random Replacement)**.** *Without loss of generality*[1]*, let* $n = 2m + 1$ *be an odd integer representing the number of tokens in an n-gram. Assume each token independently votes* 0 *or* 1 *according to a Bernoulli(0.5) distribution. If* k *tokens are randomly replaced,the probability that one bit of the majority vote remains unchanged is given by:*

$$P_k(n) = 2 \sum_{t=m+1}^{n} \frac{\binom{n}{t}}{2^n} \cdot \sum_{a=0}^{t-m-1} \frac{\binom{t}{a}\binom{n-t}{k-a}}{\binom{n}{k}}$$

[1]for the cases where n is a even number, we just need to agree on the najority vode result when there are equal number of 0s and 1s. The analysis follows the same methodology.

To interpret this theorem, t is the number of original votes matching the majority bit, and a is the number of flipped tokens that come from the majority class. The inner sum counts the configurations where fewer than $t - m$ majority votes are flipped, thus preserving the majority. In the special case where $k = 1$, this expression simplifies to: $P_1(n) = 1 - 2\frac{\binom{n}{m+1}}{2^n} \cdot \frac{m+1}{n}$. Here, $2\frac{\binom{n}{m+1}}{2^n}$ is the probability that the original vote has margin exactly 1, and $\frac{m+1}{n}$ is the probability that the flipped token belongs to that majority class.

Corollary 1 (Robustness of Entire Seed Vector)**.** *If the seed vector has length* ℓ*, then the probability that the entire seed vector is preserved is:* $\Pr[\textit{seed preserved}] = P_k(n)^\ell$.

This is the probability that our detection power remains unaffected by the edits. In contrast, without stabilizing the seed, this probability drops to 0 even when $k = 1$.

Our seeding strategy requires the seeding length to be relatively small, because the probability of the seed being changed after an n-gram replacement grows exponentially with seed length. Short seeds also introduce a new challenge: An adversary could potentially reconstruct the mapping between n-grams and seed values, since a 4-bit seed can only take values from 0 to 15. This would enable them to forge the watermark by replacing n-grams with another n-grams producing same seed. To address this, we first deploy keyed hash functions so that the adversary cannot compute those hash values. Meanwhile, we concatenate the short seed with a larger private integer, producing a final key such that an adversary cannot pinpoint the exact seed values even when there are only 16 possible seeds. Regarding the effects of using a short seed, we note that the security properties remain valid, as even in the worst case—for example, a 1-bit seed—there would only be two possible red-green partitions, and its security is reduced to the classic fixed-red-green list watermark. However, overly short seed length were found to slightly degrade text quality and caused higher chances of hallucination in our experiments, which is expected since these are also shortcomings of the fixed-green-red-list watermark. In our experiments, both 4-bit and 8-bit seeds demonstrated improved robustness compared to conventional deterministic seeding. Notably, the 8-bit seed offered a good balance—achieving better robustness than baseline methods while preserving comparable text quality. The 4-bit seed,

on the other hand, yielded stronger robustness under aggressive text modifications but at the cost of noticeable degradation in text quality.

4.2 ScaledPRF Algorithm

Existing watermarking methods either assume a fixed perturbation strength or require modifications such as adding multiple PRFs, which increases overhead. Gumbel Sampling, for instance, introduces random noise into the logits before softmax, making token selection inherently stochastic. This approach does not provide a direct strength control mechanism, as the randomness is a fundamental part of the sampling process. Consequently, watermark detectability in Gumbel Sampling is implicitly tied to the model's sampling temperature rather than an explicitly adjustable signal strength parameter.

Similarly, the Soft Red List method modulates watermark strength by adjusting the proportion of red and green list tokens. While this approach allows for some control over watermark intensity, it directly distorts the model's original token probability distribution, which can degrade generation quality, especially in low-entropy text where forced probability shifts disrupt fluency.

To overcome these limitations, we introduce ScaledPRF, a simple yet efficient method for explicitly modulating watermark strength. Instead of applying a fixed PRF-based perturbation to logits, we scale the perturbation using λ, enabling fine-grained tuning of watermark strength. The strength can be increased for higher detectability or reduced to minimize text distortion.

Given a text sequence, ScaledPRF modifies the logit adjustment process in three key steps: (1) Seeding: A seed is derived from past tokens using any seeding method. (2) Perturbation Injection: PRF generates a perturbation r_i^j for each token t_j, which is then scaled by a factor λ. (3) Normalization and Sampling: To maintain the expected probability distribution, a correction term C is introduced, ensuring that the overall logit shift does not distort token probabilities in expectation. Finally, token sampling is performed via softmax over the adjusted logits. The full procedure is formally presented in Algorithm 2.

It is crucial to ensure that the expected token probabilities remain consistent with the original model output. Directly scaling perturbations alters token likelihoods, which, when passed through the softmax function, may introduce unintended biases. This is why we have the normalization term: $C = \ln \mathbb{E}\left[e^{\lambda r^j}\right]$. This correction eliminates first-order bias caused by exponentiation in the softmax function. However, due to the non-linearity of softmax, expectation does not distribute cleanly over the probability fraction, introducing a second-order bias. In Appendix B, we provide extended discussion of a more refined correction term.

Algorithm 2 ScaledPRF Algorithm

Require:
  Functions:
    Seeding($\mathbf{t}$) $\rightarrow$ seed ▷ Generates a seed from past tokens
    PRF(seed, key, token) $\rightarrow r$ ▷ Generates pseudo-random perturbation
    ScalingFactor $\rightarrow \lambda$ ▷ Determines watermark scaling at position i
  Parameters:
    Temperature T, Vocabulary $\mathcal{V}$, Language Model $\mathcal{M}$, Key sk;
Ensure: Generated token sequence $\mathbf{t} = \{t_1, t_2, \dots\}$;
for each position $i \geq n$ **do**
  seed $\leftarrow$ Seeding($\mathbf{t}_{i-n}^{i-1}$);
  $C_i \leftarrow \ln\left(\mathbb{E}[e^{\lambda r}]\right)$; ▷ Normalization term ensuring expectation correction
  for each token $t_j \in \mathcal{V}$ **do**
    $l_i^j \leftarrow \mathcal{M}(\text{prompt}, \mathbf{t}_1^{i-1})$; ▷ Retrieve logit from the model
    $r_i^j \leftarrow \text{PRF}(\text{seed}, sk, t_j)$;
    $\tilde{l}_i^j \leftarrow l_i^j + \lambda r_i^j - C_i$;
  end for
  Compute softmax probabilities: $p_i^j = \frac{\exp\left(\frac{\tilde{l}_i^j}{T}\right)}{\sum_{k \in \mathcal{V}} \exp\left(\frac{\tilde{l}_i^k}{T}\right)}$;
  Sample t_i from $\mathcal{V}$ according to p_i^j
end for
return $\mathbf{t}$

To verify whether a given text sequence originates from a watermarked model, we employ a statistical hypothesis testing approach that leverages the structured randomness introduced during generation. This approach aligns with prior methods used in detecting Gumbel Sampling-based watermarking and single-function SynthID. The detection procedure is as follows.

- 1) **Recompute the Perturbation Values**: Given a suspicious text sequence $\mathbf{t} = \{t_1, t_2, \dots\}$, we apply the same seeding function used during generation to extract deterministic seeds based on prior tokens. Using the extracted seed, we regenerate the perturbation values r_i for each token t_i through the same PRF.

- 2) **Compute the Detection Score**: We calcu-

late an aggregate score over all tokens in the sequence: Score $= \frac{1}{|\mathbf{t}|-n} \sum_{i=n+1}^{|\mathbf{t}|} r_i$.

- 3) **Compare Against a Decision Threshold**: If the computed score exceeds a predefined threshold, the text is classified as watermarked. The detection threshold can be calibrated based on the expected score distribution of watermarked and non-watermarked text, allowing for a tunable trade-off between false positive rate (FPR) and true positive rate (TPR) depending on application requirements.

This statistical detection framework is not exclusive to ScaledPRF. In fact, this technique have been proposed in prior watermarking studies and can be adapted for detecting other methods such as Gumbel Sampling and SynthID with single PRF.

5 Experiments

To evaluate the effectiveness of our watermarking methods, we conduct experiments on three diverse datasets covering open-ended text completion, question answering, and mathematical reasoning. For text completion, we use the Colossal Clean Crawled Corpus (C4) (Raffel et al., 2020), while for mathematical reasoning, we use GSM-8K (Cobbe et al., 2021). Additionally, we include the Alpaca dataset (Taori et al., 2023) for question answering.

Our evaluation consists of two main aspects:

- Robustness under text modifications (Section 5.1): We evaluate the resilience of the stable seeding algorithm, which we denote as SS for the rest of this section, against token replacements at different frequencies. This analysis is conducted on the C4 dataset.

- Generation quality analysis (Section 5.2 and Section C): We assess how watermarking affects text fluency, coherence, grammatical correctness, and other linguistic properties, mathematical reasoning capability across datasets.

Generation Settings For text-based tasks, we set the maximum sequence length to 256 and vary the temperature between 0.5 and 0.9. The n-gram size for seeding is fixed at 8. For mathematical reasoning, we employ a zero-shot setting and allow a maximum sequence length of 512 to support longer chain-of-thought reasoning. For the C4 dataset, we use LLaMA-2-13B (Touvron et al., 2023) as the base model, and evaluate on 1,400 generated samples. For the GSM-8K dataset, we use LLaMA-3-Instruct-8B (Dubey et al., 2024) as the base model, and the evaluation is conducted on the test data, which contains approximately 1,300 samples. For the Alpaca dataset, we use base model of LLaMA-2-13B-Chat (Touvron et al., 2023), and evaluate on 464 generated samples. All experiments were conducted on a single Tesla V100 GPU.

5.1 Robustness Evaluation

We evaluate the robustness of various watermarking methods against text edits by Easy Data Augmentation (EDA) (Wei and Zou, 2019) on the C4 datasets. EDA consists of four operations: synonym replacement, random insertion, random swap, and random deletion. Our study includes four watermarking algorithms: Gumbel Sampling, Soft Red List, SynthID (with 8 PRFs), and ScaledPRF. Each method is tested with and without stable seeding, using different seed lengths (SL) and scaling factors (λ). While the SynthID paper recommends using 20–30 PRFs for optimal detection, we limit our experiments to 8 PRFs to focus on robustness rather than maximizing detection under unedited text.

Detection performance is assessed under progressively more aggressive text modifications: no edits, 12% edits, and 20% edits (with 25% ratio for each type of the 4 EDA operations). We note that the position of the edits happens randomly, and most n-gram would be affected in this case. Detection accuracy is measured as the true positive rate (TPR) while maintaining a false positive rate (FPR) below 0.05. We evaluate the impact of temperature on watermark robustness, conducting experiments at $T = 0.9$ (Figure 1) and $T = 0.5$ (Figure 2).

From the results in both figures, we observe that stable seeding algorithm significantly improves robustness. Across all methods, using stable seeding with seed length $= 4$ maintains significantly higher detection scores under edits compared to seed length $= 8$ or without using stable seeding. However, reducing the seed length to 4 also limits the total number of possible seeds, making the generation process more deterministic. This could potentially degrade text quality, which we further examine in Section 5.2.

For our ScaledPRF method, a higher scaling factor ($\lambda = 5$) leads to better robustness and a better detectability compared to scaling factor $\lambda = 2$. However, higher λ also affects fluency, requiring a

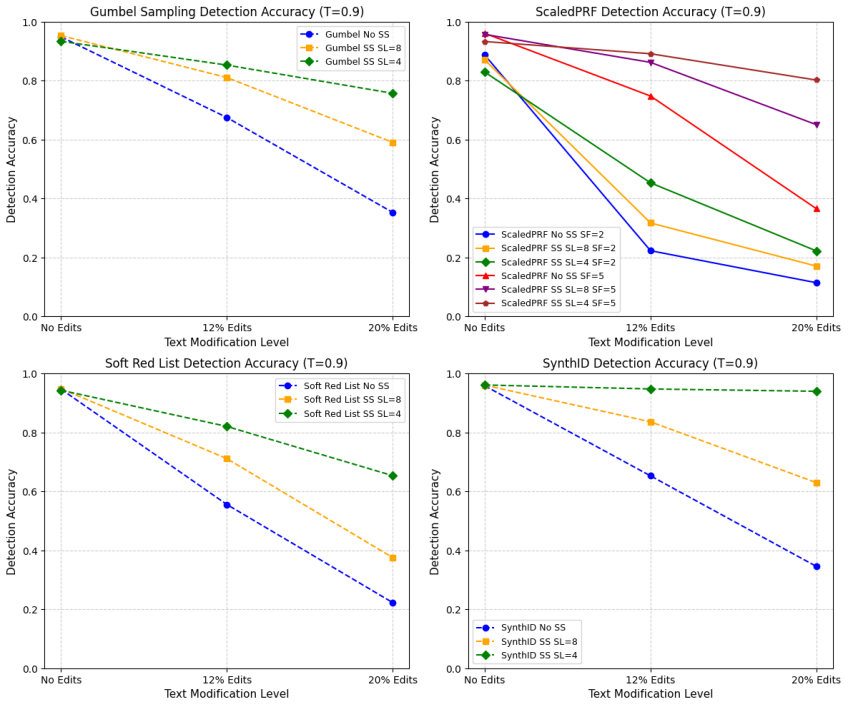

Figure 1: Detection accuracy under different text modifications for various watermarking methods at temperature $T = 0.9$. Each subplot corresponds to a different watermarking method: Gumbel Sampling, ScaledPRF, Soft Red List, and SynthID. The x-axis represents different levels of text modifications (*No edits, 12% edits, 20% edits*), while the y-axis shows detection accuracy. ScaledPRF includes additional variations with different scaling factors ($\lambda = 2$ and $\lambda = 5$) to evaluate strength modulation effects.

balance between detectability and text quality.

Comparing Figure 1 and Figure 2, we observe that Gumbel Sampling and SynthID exhibit strong sensitivity to temperature, while ScaledPRF and Soft Red List remain relatively stable but still experience a minor drop in detectability. The primary reason lies in how each method interacts with temperature-driven changes in token selection. Gumbel Sampling relies on inherent randomness to distribute the watermark signal, making it effective at higher temperatures where token selection is more stochastic. However, at $T = 0.5$, the sampling process becomes more deterministic, weakening the effect of injected noise and reducing detectability. For SynthID, the initial selection of candidate tokens depends on temperature. Since these candidates are sampled from the model's probability distribution, temperature directly influences candidate diversity. At higher temperatures, logits are more spread out, allowing perturbations to meaningfully alter token rankings. However, at lower temperatures, softmax amplifies the highest-probability tokens, making small perturbations less likely to change the final output. This restricts watermark embedding to a small subset of highly probable tokens, reducing its distinctiveness. ScaledPRF and Soft Red List modify token probabilities explicitly, ensuring that the watermark remains embedded regardless of sampling randomness. Since they introduce structured perturbations that directly adjust token probabilities, they are less sensitive to temperature shifts than Gumbel Sampling and SynthID. However, we still observe a slight drop in detectability at lower temperatures, as reduced entropy in token selection limits the impact of small perturbations.

5.2 Quality Evaluation

While perplexity and self-BLEU are commonly used in text evaluation, they have notable drawbacks: Perplexity measures confidence, not quality: A model can generate highly repetitive text with low perplexity, yet fail to produce engaging or meaningful content. Self-BLEU penalizes diversity, even when variation is desirable. A lower self-BLEU may indicate greater creativity and adaptability, not necessarily poor generation quality. To assess the quality of watermarked text beyond simple perplexity and self-BLEU scores, we evaluate multiple linguistic and semantic properties including relevance, grammar, creativity, fluency, and readability. We put detailed description for these

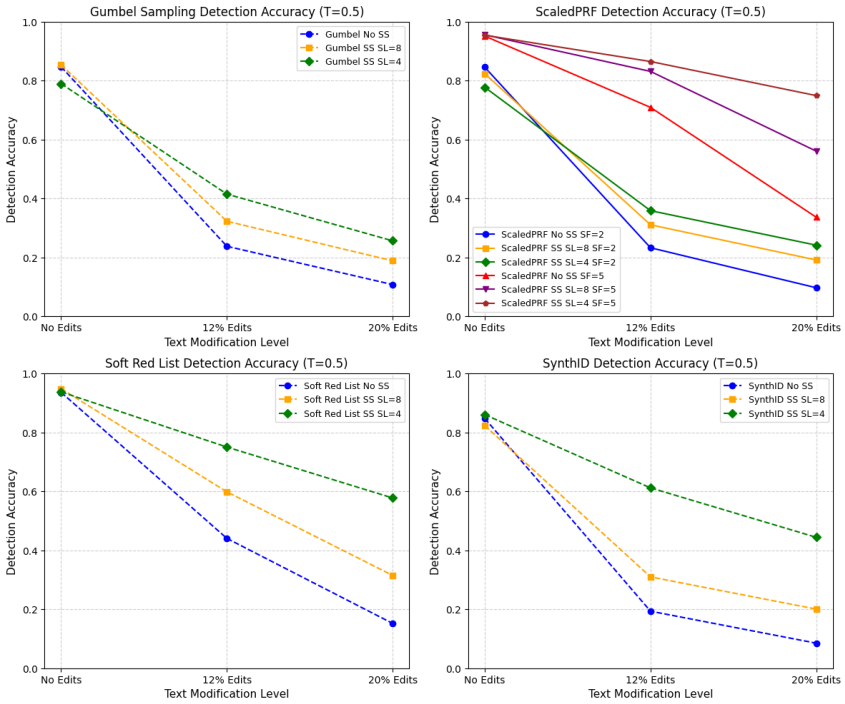

Figure 2: Detection accuracy under different text modifications for various watermarking methods at **temperature** $T = 0.5$. Each subplot corresponds to a different watermarking method: Gumbel Sampling, ScaledPRF, Soft Red List, and SynthID. The x-axis represents different levels of text modifications (*No edits, 12% edits, 20% edits*), while the y-axis shows detection accuracy. ScaledPRF includes additional variations with different scaling factors ($\lambda = 2$ and $\lambda = 5$) to evaluate strength modulation effects.

Table 1: Evaluation Metrics Across Watermarking Methods (T=0.9)

Methods	**Relevance**			**Grammar**			**Creativity**		
	No SS	SL=8	SL=4	No SS	SL=8	SL=4	No SS	SL=8	SL=4
Gumbel Sampling	37.13	35.99	27.37	4.51	4.81	5.26	54.07	52.59	48.17
ScaledPRF (λ=5)	34.46	34.47	30.42	4.79	4.70	4.77	58.07	57.82	56.72
ScaledPRF (λ=2)	37.29	37.06	35.82	4.62	4.43	5.18	55.41	55.68	53.88
Soft Red List	36.05	36.01	33.98	4.53	4.61	4.54	56.78	56.59	56.73
SynthID	35.64	35.57	30.19	5.63	5.84	7.29	56.46	56.59	54.45
No Watermark	38.13	-	-	3.61	-	-	44.47	-	-

Method	**Fluency**			**Readability**		
	No SS	SL=8	SL=4	No SS	SL=8	SL=4
Gumbel Sampling	37.38	39.13	43.59	64.21	62.96	57.77
ScaledPRF (λ=5)	35.72	35.98	37.53	61.29	59.76	59.28
ScaledPRF (λ=2)	36.76	36.47	37.62	63.78	63.65	64.92
Soft Red List	35.79	36.12	35.79	62.82	62.76	63.22
SynthID	34.62	33.91	36.92	62.68	62.96	57.63
No Watermark	42.27	-	-	69.01	-	-

metrics in the Appendix due to the space limitation. We note that we do not intend to provide a comprehensive evaluation framework for LLM, instead, we want to use these metrics to measure the influence of watermarking.

The results for text completion on the C4 dataset with $T = 0.9$ are presented in Table 1, while additional evaluations at $T = 0.5$ and results analysis aare provided in Table 3 in the Appendix. Similarly, quality assessments for the QA dataset with $T = 0.9$ are reported in Table 4 in the Appendix.

We also evaluate whether watermarking affects reasoning by measuring accuracy on GSM-8K dataset. Results show that watermarking has minimal impact on reasoning performance, making it a viable option for tasks requiring multi-step reasoning. We also defer the detailed results in the Appendix C. Addition experiments regarding the influence to the performance benchmark is also conducted and ther result can be found in Appendix.

6 Limitations.

Our majority voting-based seeding algorithm introduces a trade-off between robustness and generation quality. When the seed length is too small, it can degrade text quality due to increased determinism in token selection. However, excessively large seed lengths weaken robustness against text edits, reducing the algorithm's effectiveness. Additionally, the method requires applying majority voting over sufficiently long n-grams, as very short contexts (e.g., n=1 or 2) fail to provide enough redundancy for stable seed generation. Moreover, majority voting-based seeding requires additional computations compared to naive PRF-based watermarking, which could introduce overhead when applied at scale. Lastly, while our approach enhances resilience against minor edits such as token substitutions, it remains vulnerable to full paraphrasing, which fundamentally alters the underlying structure of the text.

Potential Risks. Misattribution Risks present for all watermarking methods. If watermark detection is used in high-stakes settings (e.g., plagiarism detection, AI-generated misinformation tracing), false positives could misattribute text to an AI when it was actually human-written, leading to academic penalties, reputational harm, or legal consequences.

References

Scott Aaronson. 2022. Watermarking of large language models. In Scott Aaronson https://www.scottaaronson.com/talks/watermark.ppt.

Miranda Christ and Sam Gunn. 2024. Pseudorandom error-correcting codes. In Annual International Cryptology Conference, pages 325–347. Springer.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783.

Abe Bohan Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. 2023. Semstamp: A semantic watermark with paraphrastic robustness for text generation. arXiv preprint arXiv:2310.03991.

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

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

Wenjie Qu, Dong Yin, Zixin He, Wei Zou, Tianyang Tao, Jinyuan Jia, and Jiaheng Zhang. 2024. Provably robust multi-bit watermarking for ai-generated text via error correction code. arXiv preprint arXiv:2401.16820.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140):1–67.

Jie Ren, Han Xu, Yiding Liu, Yingqian Cui, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. 2023. A robust semantics-based watermark for large language model against paraphrasing. arXiv preprint arXiv:2311.08721.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

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

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196.

Yihan Wu, Zhengmian Hu, Hongyang Zhang, and Heng Huang. 2024. Dipmark: A stealthy, efficient and resilient watermark for large language models. In International Conference on Machine Learning.

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

Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairoze, Andres Fabrega, Nicholas Carlini, Sanjam Garg, Sanghyun Hong, Milad Nasr, Florian Tramer, et al. 2024a. Sok: Watermarking for ai-generated content. arXiv preprint arXiv:2411.18479.

Xuandong Zhao, Lei Li, and Yu-Xiang Wang. 2024b. Permute-and-flip: An optimally robust and watermarkable decoder for llms. arXiv preprint arXiv:2402.05864.

A Proof of Theorem 1

Proof. We want to compute the probability that the majority vote outcome over an $n = 2m + 1$-token block remains unchanged after k random token replacements, where each token initially votes for 0 or 1 independently with equal probability, i.e., $v_i \sim \text{Bernoulli}(0.5)$.

Let $V = (v_1, \ldots, v_n)$ denote the original votes. The seed bit is determined by the majority vote:

$$\text{Majority}(V) = \begin{cases} 1 & \text{if } \sum_i v_i > m \\ 0 & \text{otherwise} \end{cases}$$

Let t be the number of votes equal to the majority bit in the original sequence. Since the votes are symmetric under Bern(0.5), we can analyze the event in which the original majority is 1, and then double the result due to symmetry.

Given that the original majority has t votes for 1 (with $t \geq m + 1$), we want to compute the probability that after k tokens are randomly replaced, the majority remains 1. Suppose a of the k replaced tokens came from the t tokens originally voting 1 (i.e., from the majority class), and the remaining $k - a$ came from the minority class ($n - t$ tokens). After replacing these a tokens, we are left with $t - a$ votes for 1. The majority remains preserved if $t - a > m$, i.e., $a < t - m$.

The conditional probability that exactly a of the k replaced tokens came from the majority class is:

$$\frac{\binom{t}{a}\binom{n-t}{k-a}}{\binom{n}{k}}.$$

Thus, for a fixed $t \in \{m + 1, \ldots, n\}$, the probability that the majority remains unchanged is:

$$\sum_{a=0}^{t-m-1} \frac{\binom{t}{a}\binom{n-t}{k-a}}{\binom{n}{k}}$$

Now, we compute the total probability by averaging over all valid t where the majority is originally 1. Since the original vote vector is drawn i.i.d. from Bern(0.5), the probability that exactly t out of n tokens vote 1 is $\binom{n}{t}/2^n$.

Finally, we multiply by 2 to account for the symmetric case when the majority is 0 (by the same logic), giving:

$$P_k(n) = 2 \sum_{t=m+1}^{n} \frac{\binom{n}{t}}{2^n} \cdot \sum_{a=0}^{t-m-1} \frac{\binom{t}{a}\binom{n-t}{k-a}}{\binom{n}{k}}$$

This completes the proof.

□

B ScaledPRF's Expectation-Preserving Correction Term

Using a Taylor expansion, we find that:

$$\mathbb{E}[\tilde{p}^j] = p^j + \frac{\lambda^2}{2}\left(\mathbb{E}[(r^j)^2]p^j - p^j \sum_{k \in \mathcal{V}} p^k \mathbb{E}[(r^k)^2]\right).$$

Thus, while our normalization term corrects for softmax exponentiation bias, a small residual distortion remains due to Jensen's inequality. This distortion grows with λ, meaning that increasing the watermark signal strength slightly shifts token probabilities. A more precise correction term that incorporates variance across all tokens is needed for full expectation preservation:

$$C' = C + \frac{\lambda^2}{2} \sum_{k \in \mathcal{V}} p^k \mathbb{E}[(r^k)^2].$$

C Quality Evaluation

While perplexity and self-BLEU are commonly used in text evaluation, they have notable drawbacks: Perplexity measures confidence, not quality: A model can generate highly repetitive text with low perplexity, yet fail to produce engaging or meaningful content. Self-BLEU penalizes diversity, even when variation is desirable. A lower self-BLEU may indicate greater creativity and adaptability, not necessarily poor generation quality. To assess the quality of watermarked text beyond simple perplexity and self-BLEU scores, we evaluate multiple linguistic and semantic properties.

Relevance. Relevance quantifies the topical alignment between the response and the prompt. It considers the response as a single unit and compares it to the prompt using TF-IDF and cosine similarity. This metric ensures that the generated text stays on-topic and addresses the intended theme. The scores are assigned on a scale of 0 to 100, with higher values indicating greater relevance.

Grammar. We measure grammatical correctness using the *language tool python* library, identifying errors that may affect readability. Since automatic tools may flag stylistic variations as errors, we ensure true positives by filtering only genuine grammatical mistakes. The scores indicate the average number of grammar errors in each generated output.

Creativity. Creativity measures how divergent and expressive the generated text is compared to the prompt. We assess this through: Semantic distance from the prompt (measured via embedding similarity); Lexical richness, defined as (unique words) / (total words), which captures vocabulary diversity; Expressive word usage, evaluating the presence of figurative and descriptive language.The scores are assigned on a scale of 0 to 100, with higher values indicating greater creativity and diversity.

Fluency. Fluency quantifies the smoothness and natural flow of the response. We compute semantic similarity between consecutive sentences to ensure that transitions are not abrupt or unnatural. The scores are assigned on a scale of 0 to 100, with higher values indicating greater sematic similarity.

Readability. We measure readability using the Flesch-Kincaid readability score, implemented via the *textstat* library. A higher score indicates that the text is easier to read, while lower scores suggest complex or dense sentence structures. The scores are assigned on a scale of 0 to 100.

Results analysis All watermarking methods exhibit lower relevance, increased grammar errors, and reduced fluency and readability compared to non-watermarked text. This is expected, as perturbations in token selection slightly shift topic focus and introduce minor disruptions in sentence structure. However, these modifications also increase creativity, suggesting that watermarked text remains expressive and diverse despite minor quality trade-offs.

Among the methods, Gumbel Sampling is the most affected by stable seeding, showing significant drops in relevance, higher grammar errors, and reduced creativity—likely due to its inherent randomness, which becomes less effective when structured perturbations are introduced. In contrast, ScaledPRF and Soft Red List remain more stable under stable seeding, preserving better text quality and topic alignment.

Overall, all watermarking methods perform similarly, with ScaledPRF achieving slightly lower grammar errors while maintaining strong engagement and creativity. This suggests that our method offers a well-balanced trade-off between watermark robustness and text quality, without severely degrading readability or coherence.

Impact of Watermarking on Mathematical Reasoning We evaluate whether watermarking affects reasoning capabilities by measuring accuracy on GSM-8K, a benchmark for arithmetic and multi-step reasoning. We conducted the experiments at temperature $T = 0.5$ without the majority-voting seeding. Accuracy is determined by whether the model produces the correct final numerical answer. The results are presented in Table 2. Results show that watermarking has minimal impact on reasoning performance, making it a viable option for tasks requiring multi-step reasoning.

Table 2: Accuracy on GSM-8K across different watermarking methods.

Method	Accuracy (%)
No watermark	58.92
Gumbel Sampling	61.00
ScaledPRF	58.69
Soft Red List	60.46
SynthID	59.85

C.1 Computation costs

Our seeding algorithm introduces extra computation cost due to hash function evaluations. Therefore, we test the performance of text generation with and without our seeding algorithm, the experiment results shows that the throughput has decreased around 10 percent. The experiment results are illustrated in Table 5.

Table 3: Evaluation Metrics Across Watermarking Methods (T=0.5)

Methods	**Relevance**			**Grammar**			**Creativity**		
	No SS	SL=8	SL=4	No SS	SL=8	SL=4	No SS	SL=8	SL=4
Gumbel Sampling	36.93	37.6	34.22	5.27	5.69	6.45	43.13	43.83	39.54
ScaledPRF (λ=5)	37.92	37.15	31.90	4.27	4.29	5.17	52.48	52.34	48.84
ScaledPRF (λ=2)	38.80	37.83	36.51	5.34	5.30	5.80	47.08	44.39	43.96
Soft Red List	39.03	38.15	34.78	4.60	4.96	5.81	49.97	49.46	47.16
SynthID	38.02	39.21	35.19	5.97	5.97	5.87	44.90	45.05	43.06
No Watermark	38.67	-	-	3.72	-	-	45.36	-	-

Method	**Fluency**			**Readability**		
	No SS	SL=8	SL=4	No SS	SL=8	SS=4
Gumbel Sampling	48.33	47.01	54.03	68.01	67.78	65.84
ScaledPRF (λ=5)	39.06	40.02	43.43	66.02	64.87	64.86
ScaledPRF (λ=2)	44.24	47.29	47.13	68.62	68.09	67.01
Soft Red List	41.34	40.89	43.89	68.25	66.78	68.11
SynthID	46.07	45.67	48.86	68.07	67.66	67.46
No Watermark	42.96	-	-	69.53	-	-

Table 4: Evaluation Metrics of QA Dataset Across Watermarking Methods (T=0.9)

Methods	**Relevance**			**Grammar**			**Creativity**		
	No SS	SL=8	SL=4	No SS	SL=8	SL=4	No SS	SL=8	SL=4
Gumbel Sampling	31.56	31.67	31.31	0.87	0.83	0.85	63.53	63.20	62.75
ScaledPRF (λ=5)	31.00	30.83	30.29	0.99	0.99	0.99	64.45	64.26	63.75
ScaledPRF (λ=2)	31.73	31.53	31.53	0.91	0.92	0.92	63.12	63.61	63.95
Soft Red List	31.58	31.5	31.23	0.98	0.95	0.95	63.83	64.31	63.56
SynthID	31.70	31.80	32.56	0.92	1.10	1.34	63.24	62.99	63.28
No Watermark	30.60	-	-	0.62	-	-	60.41	-	-

Method	**Fluency**			**Readability**		
	No SS	SL=8	SL=4	No SS	SL=8	SL=4
Gumbel Sampling	34.48	34.68	33.65	61.76	60.88	60.67
ScaledPRF (λ=5)	33.13	32.88	34.44	61.13	61.19	59.82
ScaledPRF (λ=2)	33.77	34.81	33.87	60.39	60.09	60.15
Soft Red List	33.88	32.97	31.32	60.46	61.05	62.16
SynthID	35.18	33.65	21.34	59.96	61.55	59.28
No Watermark	32.12	-	-	63.48	-	-

Method	**Without SS (prompts/s)**	**With SS (prompts/s)**
ScaledPRF	0.20	0.17
Soft Red List	0.32	0.28
Gumbel Sampling	0.22	0.19
SynthID	0.10	0.10

Table 5: Prompt throughput comparison with and without stable seeding.