# MorphMark: Flexible Adaptive Watermarking for Large Language Models

Anonymous ACL submission

## Abstract

Watermarking by altering token sampling probabilities based on *red-green* list is a promising method for tracing the origin of text generated by large language models (LLMs). However, existing watermark methods often struggle with a fundamental dilemma: improving watermark effectiveness (the detectability of the watermark) often comes at the cost of reduced text quality. This trade-off limits their practical application. To address this challenge, we first formalize the problem within a multi-objective trade-off analysis framework. Within this framework, we identify a key factor that influences the dilemma. Unlike existing methods, where watermark strength is typically treated as a fixed hyperparameter, our theoretical insights lead to the development of MorphMark—a method that adaptively adjusts the watermark strength in response to changes in the identified factor, thereby achieving an effective resolution of the dilemma. In addition, MorphMark also prioritizes flexibility since it is an model-agnostic and model-free watermark method, thereby offering a practical solution for real-world deployment, particularly in light of the rapid evolution of AI models. Extensive experiments demonstrate that MorphMark achieves a superior resolution of the effectiveness-quality dilemma, while also offering greater flexibility and time and space efficiency.

## 1 Introduction

The rapid development and widespread adoption of Large Language Models (LLMs) have raised concerns about the traceability of AI-generated text and copyright protection. Watermarking (Kirchenbauer et al., 2023; Liu et al., 2024b; Dathathri et al., 2024), which embeds distinctive patterns into generated content, has emerged as a critical solution to these challenges. However, the trade-off between watermark effectiveness (i.e., detectability and robustness in this paper) and text quality remains a major barrier to practical adoption. A stronger watermark enhances effectiveness but degrades text quality (Kirchenbauer et al., 2023; Liu et al., 2024b; Dathathri et al., 2024), while a weaker watermark preserves text quality but becomes harder to detect and more vulnerable to attacks, even simple paraphrasing (Liu et al., 2024b; Dathathri et al., 2024; Giboulot and Furon, 2024; Wu et al., 2024). Therefore, developing a watermarking mechanism that can effectively reconcile watermark effectiveness and text quality is crucial.

KGW (Kirchenbauer et al., 2023) is the first watermarking method based on red-green lists. Specifically, during token generation, it partitions the vocabulary into green and red lists and then increases green tokens' probabilities. As a result, the generated sequence contains more green tokens, allowing it to be identified as watermarked. However, KGW struggles to balance watermark effectiveness and text quality. Unbiased watermarking (Kuditipudi et al., 2024; Hu et al., 2024; Wu et al., 2024; Mao et al., 2024) ensures that the expected sampling distribution remains unchanged, preserving text quality. However, current implementations often lack robustness. Low-entropy watermarking (Lu et al., 2024; Lee et al., 2024; Liu and Bu, 2024) targets low-entropy text generation. While not explicitly designed for quality preservation, it achieves this by avoiding watermarking low-entropy tokens. However, it requires access to the original model for detection, increasing computational cost. Besides, some methods (Liu et al., 2024a; He et al., 2024a; Huo et al., 2024) attempt to balance watermark effectiveness and text quality by training auxiliary models. However, these approaches lack flexibility (model-agnostic and model-free). First, they require training model-specific auxiliary models for different LLMs. Second, they disrupt end-to-end inference, increasing the complexity of LLM deployment and increasing inference latency since they adopt extra models.

Therefore, in our paper, we argue that the watermark methods should prioritize flexibility.

In this paper, we first formulate the watermark effectiveness and text quality as a multi-objective trade-off analysis function to analyze the factors influencing this function. The watermark studied here is also based on the green-red list approach. Through this theoretical framework, we reveal that the cumulative probability of green-list tokens plays a key role in determining the overall multi-objective benefits of increasing watermark strength. Note that watermark strength refers to the parameter that indicates the intensity of the watermark, while watermark effectiveness reflects its practical detectability performance. Specifically, as the cumulative probability of the green list decreases, the benefits of increasing watermarking strength diminish progressively and can even turn negative. Based on this theoretical insight, we propose MorphMark, which can effectively address the dilemma between watermark effectiveness and text quality. The core idea of MorphMark is to dynamically adjust the watermarking strength in response to changes in the cumulative probability of the green list, aiming to increase the overall multi-objective benefits.

We summarize our contributions as follows: **1)** We present a theoretical framework that captures both watermark effectiveness and text quality. Based on this framework, we derive and reveal the critical role of the cumulative probability of green-list tokens in balancing watermark effectiveness and text quality. To the best of our knowledge, this is the first time this role has been revealed. **2)** We introduce MorphMark, a novel watermarking framework that dynamically adjusts watermarking strength based on the cumulative probability of green-list tokens. MorphMark is theoretically sound, effectively addressing the dilemma between text quality and watermark effectiveness. It also demonstrates excellent time and space efficiency. Moreover, it is highly flexible, supporting training-free and end-to-end operation. **3)** Through comprehensive empirical evaluation, we demonstrate the effectiveness and flexibility of MorphMark.

## 2 Preliminaries

Watermark injection aims to embed a detectable pattern into generated text by modifying the probability distribution output by LLMs. We formalize watermarking in LLMs using KGW (Kirchenbauer et al., 2023) as an example below.

Let the vocabulary be denoted as $\mathcal{V}$, and the input token sequence as $(x_1, x_2, \ldots, x_{t-1}) \in \mathcal{V}^*$. The probability distribution for generating the next token $x_t$ without a watermark is given by:

$$P\left(x_t \mid x_1, x_2, \ldots, x_{t-1}\right), \tag{1}$$

which can be simplified as:

$$P\left(x_t \mid \boldsymbol{x}_{1:t-1}\right), \tag{2}$$

where $\boldsymbol{x}_{1:t-1} = x_1, x_2, \ldots, x_{t-1}$ represents the input sequence.

KGW watermark injection operates as follows: A hash value $h$ is generated using a user-defined private key $k$ and a preceding token $x_{t-1}$. This hash value $h$ serves as a random seed to partition the vocabulary $\mathcal{V}$ into two subsets: the green list $G$ and the red list $\mathcal{V}_R$, where the green list $\mathcal{V}_G$ contains a fraction $\gamma$ of the total vocabulary $\mathcal{V}$, i.e., $|\mathcal{V}_G| = \gamma|\mathcal{V}|$. $\gamma$ is set to 0.5 below by default.

Next, KGW increase the probability of tokens in green list. For simplicity, we will only describe the increase in the probability of green-list tokens, while the probability of red-list tokens will naturally decrease accordingly. Specifically, for a token $i$, the probability $p_i$ is modified as follows:

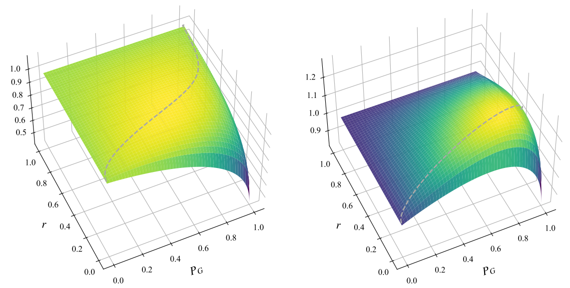
$$\hat{p}_i = \begin{cases} \frac{p_i e^{\delta}}{\sum_{j \in G} p_j e^{\delta} + \sum_{j \in R} p_j}, & \mathcal{V}_i \in \mathcal{V}_G, \\ \frac{p_i}{\sum_{j \in G} p_j e^{\delta} + \sum_{j \in R} p_j}, & \mathcal{V}_i \in \mathcal{V}_R. \end{cases} \tag{3}$$

KGW uses a hyperparameter $\delta$ to control the watermark strength. A larger $\delta$ results in improved watermark effectiveness, but lower text quality. By autoregressively sampling from this modified distribution, watermarked sequences can be generated, where the presence of a watermark can be detected based on the proportion of tokens selected from the green list $\mathcal{V}_G$.

Specifically for watermark detection, to determine whether a sequence $S = s_1, s_2, \ldots, s_{|T|}$ contains a watermark, we calculate the $z$-score as:

$$z = \frac{|S|_G - \gamma|T|}{\sqrt{|T|\gamma(1-\gamma)}}, \tag{4}$$

where $|T|$ is the total number of tokens and $|S|_G$ is the number of tokens in the green list. By setting a threshold of $z$-score, we can determine if the sequence is watermarked. If $z$-score exceeds the threshold, it indicates that the sequence contains a watermark.



(a) $\omega = 0.2$ (b) $\omega = 0.4$

Figure 1: Visualization of $\mathcal{F}$ across different $P_G$ and $r$. The vertical axis represents $\mathcal{F}$. A dashed dark gray line is used to indicate the optimal $r$ (i.e., $r^*$) that maximizes $\mathcal{F}$ for a fixed $P_G$. We can observe that as $P_G$ decreases, $r^*$ also decreases.

# 3 Methodology

In this section, we provide a detailed introduction to the proposed watermark method MorphMark. First, in § 3.1, we formalize the multi-objective analysis function $\mathcal{F}$, which can comprehensively capture both text quality $\mathcal{T}$ and watermark effectiveness $\mathcal{W}$. We then theoretically prove that as $P_G$ decreases, reducing $r$ can lead to a larger $\mathcal{F}$. Based on this insight, we describe our watermark algorithm detailedly in § 3.2.

## 3.1 Multi-objective Trade-off Framework

In this section, we will model the multi-objective trade-off framework during the proces of sampling the next token.

**Watermark Mechanism.** During generating a new token, we have an original sampling distribution $P = \{p_i\}_1^{|\mathcal{V}|}$. To watermark this token, we first split the vocabulary into a green list $\mathcal{V}_G$ and a red list $\mathcal{V}_R$. Let $P_G$ represent the sum of probabilities of green tokens, i.e,

$$P_G = \sum_{j \in G} p_j. \tag{5}$$

Since the maximum increase of $P_G$ is $1 - P_G$ (as $P_G$ cannot exceed 1), we define the total increase of $P_G$ as $r \cdot (1 - P_G)$, where $r$ is used to represent watermark strength and $r \in (0, 1)$. The larger $r$, the greater the watermark strength. Formally, we have the watermarked sampling distribution $\hat{P} = \{\hat{p}_i\}_1^{|\mathcal{V}|}$:

$$\hat{p}_i = \begin{cases} p_i + \frac{p_i}{P_G} \cdot r(1 - P_G), & \mathcal{V}_i \in \mathcal{V}_G, \\ p_i - \frac{p_i}{1-P_G} \cdot r(1 - P_G), & \mathcal{V}_i \in \mathcal{V}_R. \end{cases} \tag{6}$$

**Text Quality.** Following Zhao et al. (2024), we define text quality as the similarity between original and watermarked sampling distributions. Here we use the Bhattacharyya Coefficient (BC) (Bhattacharyya, 1946; Ramesh et al., 2023) for computational simplicity. Other metrics (e.g., KL divergence) also yield same conclusion, as shown in App. B.2.

$$\begin{aligned} \mathcal{T}(r) &= \mathrm{BC}(P, \hat{P}) = \sum_{i \in \mathcal{V}} \sqrt{p_i \hat{p}_i} \\ &= P_G \sqrt{1 + \frac{r\,(1 - P_G)}{P_G}} + (1 - P_G)\sqrt{1 - r}, \end{aligned} \tag{7}$$

where $\mathcal{T}(r)$ represents the BC between $P$ and $\hat{P}$. A higher value of $\mathcal{T}$ indicates a smaller perturbation introduced by the watermark, which corresponds to better preservation of text quality.

**Watermark Effectiveness.** The effectiveness of the watermark can be quantified by the difference between the adjusted probabilities of tokens in the green list and those in the red list. Specifically, it is given by:

$$\begin{aligned} \mathcal{W}(r) &= (\hat{P}_G - \hat{P}_R) - (P_G - P_R) \\ &= 2r(1 - P_G), \end{aligned} \tag{8}$$

where $\hat{P}_G$ and $\hat{P}_R$ represent the summed probability of tokens in the green and red lists, respectively, under the watermarked sampling distribution, and $P_G$ and $P_R$ correspond to the probabilities under the original sampling distribution.

**Multi-objective Trade-off Analysis Function.** Then, we can construct a multi-objective trade-off analysis function $\mathcal{F}$ as a weighted sum of text quality and watermark effectiveness:

$$\begin{aligned} \mathcal{F}(r) &= \mathcal{T}(r) + \omega \cdot \mathcal{W}(r) \\ &= P_G \sqrt{1 + \frac{r(1 - P_G)}{P_G}} + (1 - P_G)\sqrt{1 - r} \\ &\quad + \omega \cdot 2r(1 - P_G), \end{aligned} \tag{9}$$

where $\omega$ is the weight of watermark effectiveness. We do not impose any restrictions on $\omega$ except $\omega > 0$. Crucially, our subsequent derivations and analysis are valid regardless of the specific value of $\omega$. In other words, whether prioritizing text quality ($\omega$ is small) or watermark effectiveness ($\omega$ is large), our proposed method and conclusions are universally applicable. This can illustrate the wide applicability of our method, enabling it to adapt to various needs and preferences.

**Theorem 1.** *Consider the process of sampling a token from the watermarked probability distribution described above, for any given $\omega > 0$, there exists an optimal $r^* \in (0, 1)$ that maximizes $\mathcal{F}$. Moreover, the optimal $r^*$ is positively correlated with $P_G$, i.e., $\frac{\partial r^*}{\partial P_G} > 0$.*

This theorem indicates that, whether prioritizing text quality or watermark effectiveness, adaptively adjusting $r$ in a positively correlated manner with $P_G$ will lead to newly generated tokens achieving both higher text quality and stronger watermark effectiveness. This guides us to adaptively assign larger $r$ when $P_G$ is high, and conversely, smaller $r$ when $P_G$ is low, in order to achieve a larger $\mathcal{F}$. The proof of Theorem 1 is provided in App. B.1.

**Visualization of Theoretical Insights.** To provide a straightforward understanding of our insights, we visualize $\mathcal{F}$ in Fig. 1. We can clearly observe that no matter how the $\omega$ is set, the lager the $P_G$, the lager the $r$ that maximizes $\mathcal{F}$ (i.e., $r^*$).

### 3.2 Adaptive Watermark

In this section, we propose an instance of the function $r = \phi(P_G)$ that satisfies the design principle outlined above:

$$\phi(x) = \begin{cases} \epsilon, & x \leq p_0, \\ \min(z(x), 1-\epsilon), & x > p_0, \end{cases} \tag{10}$$

$$z(x) = k_{linear} x, \tag{11}$$

where $\epsilon$ is a negligibly small positive value approaching 0. The function is a piecewise linear function defined over the domain $(0, 1)$. The parameter $p_0$ is the threshold for watermarking, which we call watermarking threshold. We set $\phi(P_G) = \epsilon$ when $x \leq p_0$, ensuring a very little adjustment to tokens when probabilities in the green list are very small. For $P_G$ in $(p_0, 1)$, $\phi(x)$ increases linearly. A specific example of this adaptive mechanism used in MorphMark is illustrated in Fig. 2.

We can also design a fast growth function $z(x) = e^{k_{exp} x} - 1$ and a slow growth function $z(x) = \ln(k_{log} x + 1)$, which we will explore later to determine which approach is better. For detection, we use $z$-score as KGW (Kirchenbauer et al., 2023) described in § 2. Building on the formula above, we outline the detailed watermark algorithm for text generation in Alg. 1 of App. A.

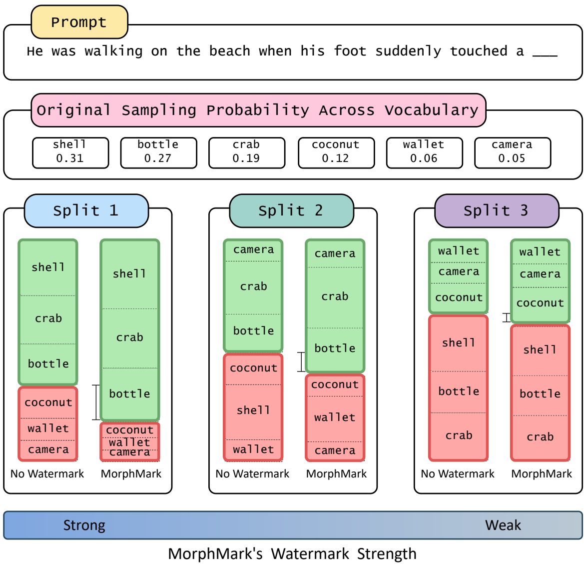



Figure 2: An example illustrating the adaptive mechanism of MorphMark. During token generation, the vocabulary is split into green and red lists. Since the split is based on the preceding tokens and user-defined keys, different tokens and users will have different splits. MorphMark adjusts the watermark strength based on the total probability of green tokens. High strength is applied when this probability is high, while low strength is used when this probability is low.

## 4 Experiments

### 4.1 Experimental Setup

Following MarkLLM (Pan et al., 2024), we evaluate MorphMark using 400 samples from the C4 (Raffel et al., 2020), with OPT-1.3B, -2.7B, and -6.7B (Zhang et al., 2022) as the backbone models. Our baselines include various flexible watermark methods, including KGW (Kirchenbauer et al., 2023), UW (Hu et al., 2024), DiPmark (Wu et al., 2024), SWEET (Lee et al., 2024), and EWD (Lu et al., 2024). We assess watermark effectiveness in terms of detectability (TPR@1%, Best F1) and robustness (assessed under the Word-S/30% attack, where 30% of words are randomly replaced with synonyms from WordNet), as well as text quality via perplexity (PPL). Details are shown in App C.1.

### 4.2 Overall Performance

We summary the main results in Tab. 1. Besides watermark effectiveness and text quality, we report the time spent on generation (Generation Time (s))

| Method | TPR@1%↑ | TPR@1%↑ (Word-S/30%) | Best F1↑ | Best F1↑ (Word-S/30%) | PPL↓ | Generation Time (s) | Detection Time (ms) | Memory Usage (B) |
|---|---|---|---|---|---|---|---|---|
| | | | | *OPT-1.3B* | | | | |
| UnWM | - | - | - | - | 10.4815 | 2.4374 | - | 0 |
| KGW | 0.9900 | 0.8050 | 0.9950 | 0.9268 | 11.4994 | 2.4901 | 33.81 | 0 |
| UW | 1.0000 | 0.7425 | 0.9975 | 0.9221 | 11.5854 | 2.5486 | 71.30 | 0 |
| DiPmark | 0.9975 | 0.7250 | 0.9975 | 0.9138 | 11.5042 | 2.5492 | 71.54 | 0 |
| SWEET | 0.9975 | 0.8225 | 0.9975 | 0.9501 | 11.5065 | 2.4667 | 44.27 | 1.3 |
| EWD | 1.0000 | 0.8450 | 1.0000 | 0.9549 | 11.4777 | 2.4526 | 44.52 | 1.3 |
| MorphMark$_{exp}$ | **1.0000** | **0.9600** | 0.9975 | **0.9778** | 11.3569 | 2.6768 | 34.17 | 0 |
| MorphMark$_{linear}$ | **1.0000** | 0.9275 | 0.9962 | 0.9727 | **11.2386** | 2.6537 | 33.99 | 0 |
| MorphMark$_{log}$ | **1.0000** | 0.9375 | **1.0000** | 0.9660 | 11.3379 | 2.6889 | 34.45 | 0 |
| | | | | *OPT-2.7B* | | | | |
| UnWM | - | - | - | - | 9.6683 | 3.1573 | - | 0 |
| KGW | 0.9950 | 0.8275 | 0.9950 | 0.9098 | 10.9324 | 3.2353 | 33.01 | 0 |
| UW | 0.9950 | 0.6900 | 0.9962 | 0.9202 | 10.8593 | 3.3178 | 72.86 | 0 |
| DiPmark | 0.9900 | 0.7125 | 0.9913 | 0.9058 | 11.0013 | 3.3126 | 72.83 | 0 |
| SWEET | 0.9975 | 0.8350 | 0.9962 | 0.9566 | 10.8377 | 3.2605 | 49.46 | 2.7 |
| EWD | 1.0000 | 0.8500 | 0.9988 | 0.9588 | 10.6303 | 3.2180 | 49.56 | 2.7 |
| MorphMark$_{exp}$ | **1.0000** | **0.9625** | 0.9987 | 0.9686 | 10.5144 | 3.5074 | 34.64 | 0 |
| MorphMark$_{linear}$ | **1.0000** | 0.9300 | **0.9988** | **0.9701** | **10.3852** | 3.4149 | 34.00 | 0 |
| MorphMark$_{log}$ | 0.9975 | 0.9250 | **0.9988** | 0.9628 | 10.6717 | 3.6792 | 34.63 | 0 |
| | | | | *OPT-6.7B* | | | | |
| UnWM | - | - | - | - | 9.0120 | 4.2656 | - | 0 |
| KGW | 0.9950 | 0.8150 | 0.9975 | 0.9058 | 9.9602 | 4.3163 | 32.30 | 0 |
| UW | 0.9950 | 0.7025 | 0.9899 | 0.8971 | 10.3701 | 4.4407 | 75.04 | 0 |
| DiPmark | 0.9975 | 0.6625 | 0.9925 | 0.9073 | 10.2747 | 4.4363 | 75.13 | 0 |
| SWEET | 0.9925 | 0.7925 | 0.9975 | 0.9539 | 10.0633 | 4.3931 | 62.20 | 6.7 |
| EWD | 1.0000 | 0.8350 | 0.9975 | 0.9523 | 9.9925 | 4.3393 | 61.74 | 6.7 |
| MorphMark$_{exp}$ | **1.0000** | 0.9100 | **0.9975** | **0.9763** | **9.6618** | 4.5198 | 35.97 | 0 |
| MorphMark$_{linear}$ | 0.9975 | **0.9250** | 0.9950 | 0.9637 | 9.7391 | 4.4456 | 35.15 | 0 |
| MorphMark$_{log}$ | 0.9950 | 0.8975 | 0.9950 | 0.9602 | 9.8585 | 4.4537 | 35.45 | 0 |

Table 1: Performance comparison on different methods. The best results are in bold for each column.

and detection (Detection Time (ms)) (for 800 tokens), as well as the size of models used for detection (Memory Usage (B)) to highlight the time and space efficiency of different watermark methods.

From the results, we can see that MorphMark outperforms all baselines in detectability, robustness, and text quality, demonstrating a superior effectiveness-quality trade-off. It spends nearly identical generation and detection time to that of KGW, indicating no significant additional delay. Additionally, MorphMark incurs no memory usage during detection, as it does not require loading any model. In summary, MorphMark is an efficient method that effectively address the dilemma between watermark effectiveness and text quality.

### 4.3 Performance on Robustness

Malicious attackers may use paraphrasing attack methods to conduct watermark removal. Thus, we implement 5 paraphrasing attack methods to evaluate the robustness of different watermarking algorithms. (1) Word-S/ refers to randomly replacing words with synonyms from WordNet, where the number after "/" indicates the proportion of words modified. (2) Word-SC/ refers to randomly replacing words with synonyms from WordNet based on context. (3) Word-D involves randomly deleting 30% of the words from the text. (4) Doc-P (GPT-3.5) rewrites the text using GPT-3.5-Turbo (OpenAI, 2024). Details are shown in App. C.2. (5) Doc-P (Dipper) rewrites the text using a specialized paraphrasing model Dipper (Krishna et al., 2024).

We summarize the results in Fig. 1. As shown, MorphMark$_{exp}$ exhibits significantly superior robustness compared to all other methods across all attack scenarios. This advantage is particularly evident when watermarked texts are paraphrased by GPT-3.5 or Dipper, where MorphMark$_{exp}$ achieves a substantially higher TPR@1%. In addition, the other two variants, MorphMark$_{linear}$ and MorphMark$_{log}$, also outperform the selected baselines in most attack settings. In summary, these results empirically demonstrate the strong robustness of MorphMark, particularly MorphMark$_{exp}$, making it a more practical and reliable choice.

### 4.4 Performance on Text Quality

Following previous work (Hu et al., 2024; Wu et al., 2024), instead of using PPL only, we evalu-

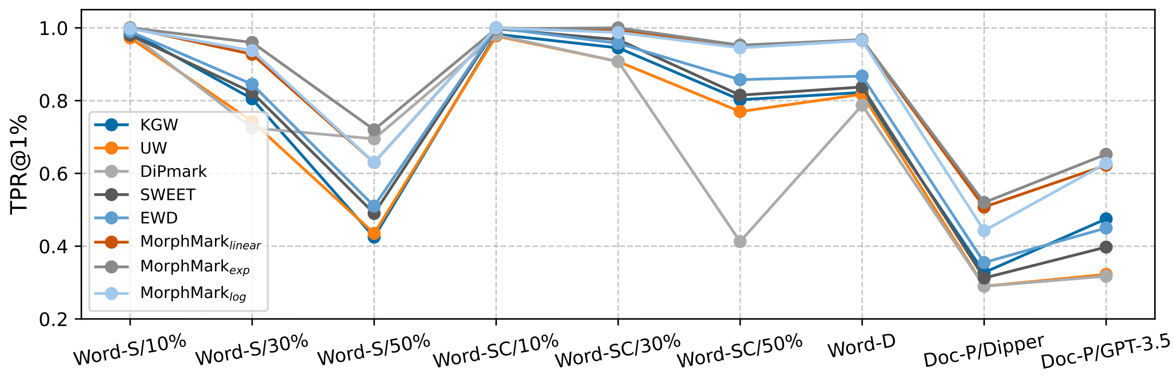
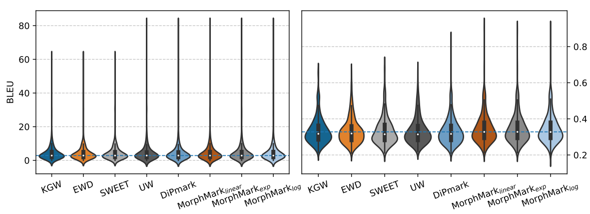
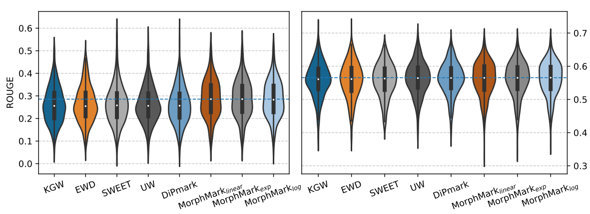



Figure 3: Robustness performance of each watermarking method under various attack scenarios.

ate text quality on two downstream tasks, specifically machine translation and text summarization. For machine translation, we employ the nllb-200-distilled-600M (Costa-jussà et al., 2022) as our translation model and randomly sample 400 instances from the WMT16 (Bojar et al., 2016) corpus for the German-to-English translation task as our test dataset. For text summarization, we evaluate 400 randomly sampled instances from the CNN-DM dataset (Hermann et al., 2015) using the OPT-1.3B model (Zhang et al., 2022). To assess performance, we employ BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), and BERTScore (Zhang et al., 2019) as evaluation metrics. Our experiments use the same parameters as the main study, ensuring that text quality is compared under the condition that MorphMark's detectability and robustness surpass other watermarking methods.



(a) Machine Translation



(b) Text Summarization

Figure 4: Text quality on downstream tasks.

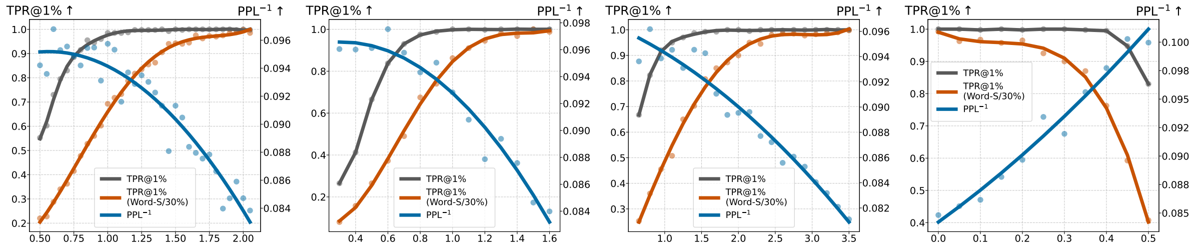
Fig. 4(a) presents the results for the machine translation task. In terms of the BLEU metric, all methods demonstrate comparable performance. However, for BERTScore, our proposed method, MorphMark's three variants, consistently outperforms all other baseline methods by a small margin. Fig. 4(b) shows the results for the text summarization task. According to the ROUGE metric, the MorphMark$_{exp}$ and MorphMark$_{linear}$ variants exhibit slightly better performance than the MorphMark$_{log}$ variant, while all three significantly outperform the baseline methods. For BERTScore, the three MorphMark variants yield nearly identical performance, showing a minor improvement over the unbiased watermarks (UW and DiPmark). Furthermore, both MorphMark and unbiased watermarks achieve a notable advantage over the other baseline approaches.

Overall, in terms of text quality, MorphMark outperforms unbiased watermarks (UW and DiPmark), and these two unbiased watermarks surpasses all other baseline approaches.

### 4.5 Ablation Study

In this section, we conduct ablation study on the hyper-parameters of MorphMark, including $k_{exp}$, $k_{linear}$ and $k_{log}$ in MorphMark$_{exp}$, MorphMark$_{linear}$, and MorphMark$_{log}$ respectively, as well as $p_0$. The impact of these parameters is clearly shown in Fig. 5. Specifically, as $k_{exp}$, $k_{linear}$ and $k_{log}$ increase, or as $p_0$ decreases, watermark strength increase, so watermark effectiveness improve, while text quality degrades.

Additionally, by combinating Fig.5(a), Fig.5(b), and Fig. 5(c), we can conveniently compare MorphMark$_{exp}$, MorphMark$_{linear}$, and MorphMark$_{log}$. By fixing either watermark effectiveness or text quality, we can assess the relative performance of the three variants along the other di-




(a) $k_{linear}$ (MorphMark$_{linear}$) (b) $k_{exp}$ (MorphMark$_{exp}$) (c) $k_{log}$ (MorphMark$_{log}$) (d) $p_0$ (MorphMark$_{exp}$)

Figure 5: **Parameter ablation study of MorphMark.** In (a), (b), and (c), we conduct an ablation study on $k$ across different variants of MorphMark, where the x-axis represents $k$. In (d), we perform an ablation study on the watermarking threshold, where the x-axis represents $p_0$.

mension. This analysis leads to the conclusion that across various levels of detectability, the text quality ranking consistently follows MorphMark$_{exp}$ > MorphMark$_{linear}$ > MorphMark$_{log}$. This highlights MorphMark$_{exp}$'s superior trade-off between watermark effectiveness and text quality, making it the strongest choice among the three designs.

## 4.6 Further Analyses

### 4.6.1 Different Sampling Parameters

In this section, we test whether MorphMark remains effective under different sampling parameters. We consider several commonly used temperature and top-p combinations: (1.2, 1.0) for high creativity, (0.7, 0.95) and (0.9, 0.95) for general-purpose tasks, and (0.3, 1.0) for precision-oriented tasks.

| (Temp, TopP) | UnWM PPL | PPL | TPR@1% | TPR@1%↑ (Word-S/30%) |
|---|---|---|---|---|
| (0.3, 1.0) | 4.1308 | 4.7605 | 0.9925 | 0.9200 |
| (0.7, 0.95) | 5.4809 | 6.1871 | 1.0000 | 0.9450 |
| (0.9, 0.95) | 7.3829 | 8.0190 | 0.9975 | 0.9550 |
| (1.2, 1.0) | 15.2175 | 16.8605 | 0.9975 | 0.9600 |

Table 2: Performance of MorphMark$_{exp}$ with different sampling parameters. UnWM refers to unwatermarked output.

Table 2 presents the results of MorphMark$_{exp}$. From the results, we observe that as the temperature increases, both the unwatermarked PPL and watermarked PPL increase, indicating that higher temperature leads to more diverse generations. Additionally, the TPR@1% remains consistently high across all settings, demonstrating the robustness of MorphMark$_{exp}$. Notably, the relative improvement in TPR@1% increases with temperature, with the highest improvement observed at (1.2, 1.0), suggesting that watermark detection benefits from more diverse text generation.

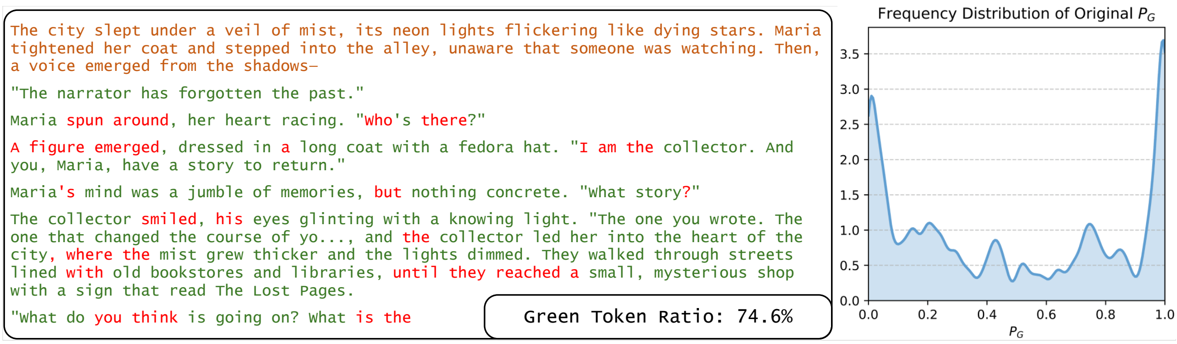
These results indicate that MorphMark$_{exp}$ performs still reliably across different sampling settings, maintaining high detection effectiveness while adapting to different decoding parameters. Results of MorphMark$_{linear}$ and MorphMark$_{log}$ are present in Tab. 3 and Tab. 4 of App. C.4

### 4.6.2 In-Depth Analysis of $P_G$ Distribution

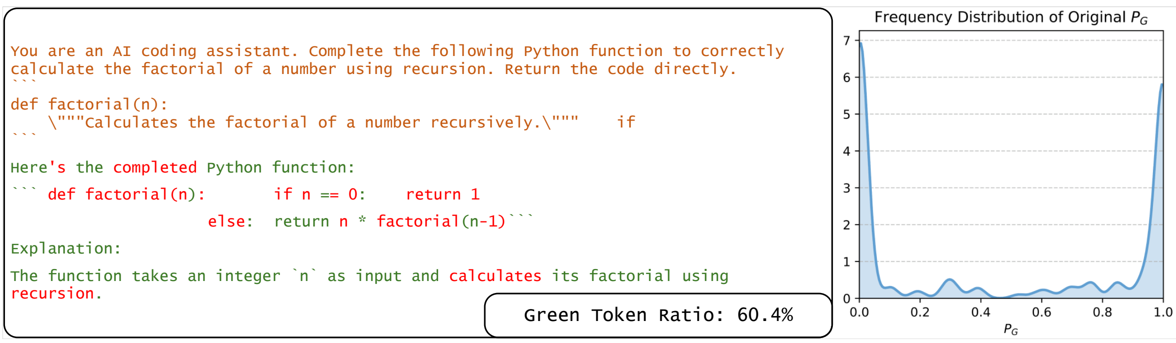
An important factor affecting the performance of MorphMark is the distribution of $P_G$ within a sequence. For example, if the sequence's entropy is low, $P_G$ tends to concentrate around 0 and 1, making it difficult for MorphMark to successfully inject the watermark.

Since the distribution of $P_G$ within a sequence is difficult to quantify with a single metric, we present a case study in this section to shed light on this aspect. To this end, we employ two contrasting examples: a high-entropy task, specifically story creation, and a low-entropy task, code generation in Fig. 6. From these examples, we observe that when the distribution of $P_G$ is extreme, the effectiveness of the watermark is low.

To determine whether such extreme conditions occur frequently, we examine the distribution of $P_G$ across several popular benchmarks including TruthfulQA (Lin et al., 2021), SQuAD (Rajpurkar, 2016; Rajpurkar et al., 2018), GSM8K (Cobbe et al., 2021) and MBPP (Austin et al., 2021). The statistical results are presented in Fig. 9. These results show that the $P_G$'s distribution in most benchmarks is relatively uniform—even in code tasks. This uniformity is likely due to the fact that code typically contains comments, and after alignment, LLMs tend to output additional natural language explanations rather than only code. Overall, since such extreme cases occur infrequently, our method remains effective in most scenarios.




(a) Story creation task with widely balanced $P_G$ values.




(b) Code generation task with a high number of extreme $P_G$ values, most $P_G$ values being concentrated near 0 or 1.

Figure 6: Case Study on $P_G$ Distributions. In example (a), which illustrates a story creation task, the $P_G$ values are well-balanced across a wide range. MorphMark performs effectively in this scenario, achieving a high ratio of green tokens throughout the sequence. In contrast, example (b) presents a code generation task with an extreme distribution, where most $P_G$ values are concentrated near 0 or 1. In this case, MorphMark proves less effective.

## 5 Related Work

Backdoor-based watermarking has been widely studied before the rise of large language models (Adi et al., 2018; Li et al., 2022; Wang et al., 2024). In the era of LLMs, due to the high cost of training models, researchers have shifted to injecting watermarks during the generation process (Kirchenbauer et al., 2023; Kuditipudi et al., 2024). Recent studies focus on low-entropy watermarking (Lu et al., 2024; Mao et al., 2024), watermark security (Pang et al., 2024; Liu et al., 2024a; He et al., 2024a), watermark privacy (Jovanović et al., 2024; Christ et al., 2024), and watermark under different sampling methods (Hu and Huang, 2024; Dathathri et al., 2024), with the most widely explored topic being the trade-off between watermark effectiveness and text quality (Hu et al., 2024; Wu et al., 2024; Huo et al., 2024). The full related work is shown in App. D.

## 6 Conclusion

This work investigates the fundamental trade-off between watermark effectiveness and text quality when watermarking large language models (LLMs). We first formally characterize this trade-off as a multi-objective analysis function and identify the cumulative probability of green-list tokens as a critical factor influencing this trade-off. Our theoretical analysis reveals that increasing watermark strength does not always lead to improved performance, particularly when the cumulative probability of the green list is low. Motivated by this theoretical insight, we introduce MorphMark, a dynamic watermarking mechanism that adaptively adjusts watermark strength to improve both watermark effectiveness and text quality. In addition, MorphMark offers flexibility and efficiency (time and space). Empirical results demonstrate MorphMark's substantial improvement across diverse models and scenarios. By integrating theoretical modeling, algorithmic design and innovation, empirical validation, and practical deployment consideration, this work propose a reliable and practical watermarking mechanism. Our findings deepen the understanding of watermarking mechanism based on green-red list and provide the community with both theoretical analytical tool and practical methodology.

## Limitations

While our empirical analysis demonstrates that MorphMark is effective in a wide range of scenarios, it is important to acknowledge certain limitations. One notable constraint arises in extremely low-entropy text generation tasks, where the watermarking capability of MorphMark becomes nearly less effective. This issue is not unique to MorphMark but rather a fundamental limitation shared by all green-red list-based watermarking methods. The core reason behind this limitation lies in the nature of low-entropy text generation. When a model produces highly predictable sequences with minimal variation, the opportunities for embedding watermarks become significantly reduced. Since green-red list-based watermarking relies on a degree of token unpredictability to manipulate token selection probabilities, it struggles to function effectively when entropy is too low.

Addressing this challenge requires exploring alternative watermarking strategies that do not depend solely on token-level entropy. Potential directions include integrating semantic or syntactic watermarking techniques, leveraging sentence-level perturbations, or incorporating watermark signals at deeper structural levels within the model.

Despite this limitation, MorphMark remains highly effective in most practical applications. The broad distribution of $P_G$ observed in our experiments suggests that, under typical generation conditions, MorphMark consistently embeds reliable watermarks. Future work should focus on refining watermarking methods to enhance performance in extreme cases while maintaining MorphMark's efficiency and usability across diverse text generation tasks.

## References


Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. 2018. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX security symposium (USENIX Security 18)*, pages 1615–1631.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Anil Bhattacharyya. 1946. On a measure of divergence between two multinomial populations. *Sankhyā: the indian journal of statistics*, pages 401–406.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation (wmt16). In *First conference on machine translation*, pages 131–198. Association for Computational Linguistics.

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*.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

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.

Eva Giboulot and Teddy Furon. 2024. Watermax: breaking the llm watermark detectability-robustness-quality trade-off. *arXiv preprint arXiv:2403.04808*.

Yuxuan Guo, Zhiliang Tian, Yiping Song, Tianlun Liu, Liang Ding, and Dongsheng Li. 2024. Context-aware watermark with semantic balanced green-red lists for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22633–22646.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024a. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4115–4129. Association for Computational Linguistics.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024b. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. *arXiv preprint arXiv:2402.14007*.

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

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

Zhengmian Hu and Heng Huang. 2024. Inevitable trade-off between watermark strength and speculative sampling efficiency for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Mingjia Huo, Sai Ashish Somayajula, Youwei Liang, Ruisi Zhang, Farinaz Koushanfar, and Pengtao Xie. 2024. Token-specific watermarking with enhanced detectability and semantic coherence for large language models. *arXiv preprint arXiv:2402.18059*.

Nikola Jovanović, Robin Staab, and Martin Vechev. 2024. Watermark stealing in large language models. In *Forty-first International Conference on Machine Learning*.

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.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2024. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36.

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

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. 2024. Who wrote this code? watermarking for code generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4890–4911. Association for Computational Linguistics.

Yiming Li, Yang Bai, Yong Jiang, Yong Yang, Shu-Tao Xia, and Bo Li. 2022. Untargeted backdoor watermark: Towards harmless and stealthy dataset copyright protection. *Advances in Neural Information Processing Systems*, 35:13238–13250.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

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

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

Yepeng Liu and Yuheng Bu. 2024. Adaptive text watermark for large language models. *arXiv preprint arXiv:2401.13927*.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. 2024. An entropy-based text watermarking detection method. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11724–11735.

Minjia Mao, Dongjun Wei, Zeyu Chen, Xiao Fang, and Michael Chau. 2024. A watermark for low-entropy and unbiased generation in large language models. *arXiv preprint arXiv:2405.14604*.

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

OpenAI. 2024. Gpt-3.5 turbo model. Available at `https://platform.openai.com/docs/models#gpt-3-5-turbo`.

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. 2024. MarkLLM: An open-source toolkit for LLM watermarking. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 61–71, Miami, Florida, USA. Association for Computational Linguistics.

Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. 2024. No free lunch in llm watermarking: Trade-offs in watermarking design choices. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

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.

P Rajpurkar. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

Rahul Ramesh, Jialin Mao, Itay Griniasty, Rubing Yang, Han Kheng Teoh, Mark K Transtrum, James P Sethna, and Pratik Chaudhari. 2023. A picture of the space of typical learnable tasks. In *Proceedings of the 40th International Conference on Machine Learning*, pages 28680–28700.

Jie Ren, Han Xu, Yiding Liu, Yingqian Cui, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. 2024. A robust semantics-based watermark for large language model against paraphrasing. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 613–625.

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*.

Zongqi Wang, Baoyuan Wu, Jingyuan Deng, and Yujiu Yang. 2024. Espew: Robust copyright protection for llm-based eaas via embedding-specific watermark. *arXiv preprint arXiv:2410.17552*.

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. 2024. A resilient and accessible distribution-preserving watermark for large language models. In *Forty-first International Conference on Machine Learning*.

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

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. 2024. Provable robust watermarking for ai-generated text. In *The Twelfth International Conference on Learning Representations*.

# A Algorithm

We present the detailed algorithm in Alg. 1.

**Algorithm 1** Text Generation with Watermark

**Input:** prompt $s_{-N_p}, \ldots, s_{-1}$, a private key $k$, hyper-parameters used in Equation 10: $p_0$, $k_{linear}$ and $\epsilon$.
**Output:** watermarked text.
**for** $t = 0$ **to** $T$ **do**
  Obtain the probability distribution vector $p = P\left(s_t \mid \boldsymbol{s}_{-N_p:t-1}\right)$ from the language model.
  Compute a hash value of token $\boldsymbol{s}_{t-1}$ using the private key $k$.
  Randomly partition the vocabulary into a green list $\mathcal{V}_G$ of size $|\mathcal{V}|/2$ and a red list $\mathcal{V}_R$ of size $|\mathcal{V}|/2$, with the hash value serving as the random seed.
  Calculate total adjustment $r = \phi(\sum_{j \in G} p_j)$ as defined in Equation 10.
  Generate the watermarked probability distribution over the vocabulary:

$$\hat{p}_i = \begin{cases} p_i + \frac{p_i}{\sum_{j \in G} p_j} \cdot r \sum_{j \in R} p_j, & \mathcal{V}_i \in \mathcal{V}_G, \\ p_i - \frac{p_i}{\sum_{j \in R} p_j} \cdot r \sum_{j \in R} p_j, & \mathcal{V}_i \in \mathcal{V}_R. \end{cases}$$

  Sample the next token $s_t$ based on the watermarked distribution $\hat{p}$.
**end for**
**return** $s_{0:T}$.

# B Proof

## B.1 Proof of Theorem 1

For simplicity in calculation, we define text quality as the Bhattacharyya coefficient coefficient (BC) between the original sampling distribution and the watermark sampling distribution. Note that using KL divergence also leads to the same conclusion, based on the same derivation process.

$$\begin{aligned} \mathcal{T}(r) &= \mathrm{BC}(P, \hat{P}) = \sum_{i \in \mathcal{V}} \sqrt{p_i \hat{p}_i} \\ &= \sum_{i \in G} \sqrt{p_i \left(p_i + \frac{p_i}{P_G} r\left(1 - P_G\right)\right)} + \sum_{i \in R} \sqrt{p_i \left(p_i - \frac{p_i}{1 - P_G} r\left(1 - P_G\right)\right)} \\ &= \sum_{i \in G} p_i \sqrt{1 + \frac{r\left(1 - P_G\right)}{P_G}} + \sum_{i \in R} p_i \sqrt{1 - r} \\ &= P_G \sqrt{1 + \frac{r\left(1 - P_G\right)}{P_G}} + (1 - P_G)\sqrt{1 - r} \end{aligned} \tag{12}$$

Detection capability is defined as the difference of increased probability of green list and red list:

$$\mathcal{W}(r) = 2\omega r(1 - P_G) \tag{13}$$

Thus, we define the multi-objective trade-off analysis function as a weighted sum of both:

$$\mathcal{F} = \mathcal{T} + \omega \cdot \mathcal{W} = P_G \sqrt{1 + \frac{r\left(1 - P_G\right)}{P_G}} + (1 - P_G)\sqrt{1 - r} + 2\omega r(1 - P_G) \tag{14}$$

where $\omega$ is the weight of detection capability and $\omega > 0$. For generality, we impose no additional restrictions on $\omega$. That is, our following derivation is valid for any $w$.

The first derivative of $\mathcal{F}$ with respect to $r$ is:

$$\frac{\partial \mathcal{F}}{\partial r} = (1 - P_G) \left( 2\omega + \frac{1}{2\sqrt{1 + \frac{r(1-P_G)}{P_G}}} - \frac{1}{2\sqrt{1-r}} \right) \tag{15}$$

We only need the sign of the derivative later. To simplify the calculation, we use $S$ to replace the derivative above, as $S$ has the same sign.

$$S = 2\omega + \frac{1}{2\sqrt{1 + \frac{r(1-P_G)}{P_G}}} - \frac{1}{2\sqrt{1-r}} \tag{16}$$

Next, we need to prove that $\mathcal{F}$ achieves its maximum at $S = 0$. The formula for the first derivative of $S$ with respect to $r$ is:

$$\begin{aligned} \frac{\partial S}{\partial r} &= \frac{1}{4 \cdot \left(-r + 1 + \frac{r}{P_G}\right)^{\frac{3}{2}}} - \frac{1}{4 \cdot (1-r)^{\frac{3}{2}}} - \frac{1}{4 \cdot P_G \cdot \left(-r + 1 + \frac{r}{P_G}\right)^{\frac{3}{2}}} \\ &= -\frac{1 - P_G}{4P_G \left(1 + r\left(\frac{1}{P_G} - 1\right)\right)^{\frac{3}{2}}} - \frac{1}{4\left(1-r\right)^{\frac{3}{2}}} < 0 \end{aligned} \tag{17}$$

This derivative is negative, meaning that $S$ is decreasing as $r$ increases.

$$\lim_{r \to 0} S = 2\omega > 0 \tag{18}$$

$$\lim_{r \to 1} S = -\infty < 0 \tag{19}$$

Since $S$ is positive at $r = 0$ and negative at $r = 1$, by the continuity of $S$ and the intermediate value theorem, there must exist a value $r^*$ between 0 and 1 such that $S(r^*) = 0$.

By the implicit function theorem, substituting $S = 0$, we obtain the relationship between $r^*$ and $P_G$:

$$\frac{\partial r^*}{\partial P_G} = -\frac{\frac{\partial S}{\partial P_G}}{\frac{\partial S}{\partial r^*}} = -\frac{\frac{r^*}{4 \cdot P_G^2 \cdot \left(1 + r^*\left(\frac{1}{P_G} - 1\right)\right)^{\frac{3}{2}}}}{-\frac{1-P_G}{4P_G\left(1 + r^*\left(\frac{1}{P_G} - 1\right)\right)^{\frac{3}{2}}} - \frac{1}{4(1-r^*)^{\frac{3}{2}}}} > 0 \tag{20}$$

This shows that when $P_G$ is larger, the optimal $r$ will also be larger, and when $P_G$ is smaller, the optimal $r$ will be smaller. In other words, increasing $P_G$ leads to an increase in the optimal parameter $r^*$. This implies that in the optimization process, as the sampling distribution changes, the watermark optimization parameter $r$ needs to be adjusted accordingly to maintain optimal performance.

### B.2 Proof for More Similarity Measurement Methods

Here, we use another similarity measurement method (KL divergence) to measure the text quality. And we will prove that it also leads to the same conclusion. Since we need the similarity instead of divergence, so we calculate $-D_{KL}(P||\hat{P})$:

$$
\begin{aligned}
\mathcal{T}(r) &= -D_{KL}(P||\hat{P}) = \sum_{i\in\mathcal{V}} p_i log \frac{\hat{p}_i}{p_i} \\
&= \sum_{i\in G} p_i log \frac{p_i + \frac{p_i}{P_G} r(1-P_G)}{p_i} + \sum_{i\in R} p_i log \frac{p_i - \frac{p_i}{1-P_G} r(1-P_G)}{p_i} \\
&= log(1 + \frac{r(1-P_G)}{P_G}) \cdot \sum_{i\in G} p_i + log(1-r) \cdot \sum_{i\in R} p_i \\
&= P_G \cdot log(1 + \frac{r(1-P_G)}{P_G}) + (1-P_G) \cdot log(1-r)
\end{aligned} \tag{21}
$$

Then, we define the multi-objective trade-off analysis function as:

$$
\begin{aligned}
\mathcal{F}(r) &= \mathcal{T}(r) + \omega\mathcal{W}(r) \\
&= P_G \cdot log(1 + \frac{r(1-P_G)}{P_G}) + (1-P_G) \cdot log(1-r) + 2\omega r(1-P_G)
\end{aligned} \tag{22}
$$

where $\omega$ is the weight of detection capability and $\omega > 0$. For generality, we impose no additional restrictions on $\omega$. That is, our following derivation is valid for any $w$.

The first derivative of $\mathcal{F}$ with respect to $r$ is:

$$
\begin{aligned}
\frac{\partial \mathcal{F}}{\partial r} &= \frac{(1-P_G)}{1 + \frac{r(1-P_G)}{P_G}} - \frac{1-P_G}{1-r} + 2\omega(1-P_G) \\
&= (1-P_G)(\frac{1}{1 + \frac{r(1-P_G)}{P_G}} - \frac{1}{1-r} + 2\omega)
\end{aligned} \tag{23}
$$

We only need the sign of the derivative later. To simplify the calculation, we use $S$ to replace the derivative above, as $S$ has the same sign.

$$
S = 2\omega + \frac{1}{1 + \frac{r(1-P_G)}{P_G}} - \frac{1}{1-r} \tag{24}
$$

Next, we need to prove that $\mathcal{F}$ achieves its maximum at $S = 0$. The formula for the first derivative of $S$ with respect to $r$ is:

$$
\begin{aligned}
\frac{\partial S}{\partial r} &= \frac{P_G^2}{(-rP_G + P_G + r)^2} - \frac{P_G}{(-rP_G + P_G + r)^2} - \frac{1}{(r-1)^2} \\
&= -\frac{P_G(1-P_G)}{(P_G + r - rP_G)^2} - \frac{1}{(1-r)^2}
\end{aligned} \tag{25}
$$

This derivative is negative, meaning that $S$ is decreasing as $r$ increases.

$$
\lim_{r\to 0} S = 2\omega > 0 \tag{26}
$$

$$
\lim_{r\to 1} S = -\infty < 0 \tag{27}
$$

Since $S$ is positive at $r = 0$ and negative at $r = 1$, by the continuity of $S$ and the intermediate value theorem, there must exist a value $r^*$ between 0 and 1 such that $S(r^*) = 0$.

By the implicit function theorem, substituting $S = 0$, we obtain the relationship between $r^*$ and $P_G$:

$$
\frac{\partial r^*}{\partial P_G} = -\frac{\frac{\partial S}{\partial P_G}}{\frac{\partial S}{\partial r^*}} = -\frac{\frac{r}{(P_G - r(P_G-1))^2}}{-\frac{P_G(1-P_G)}{(P_G+r-rP_G)^2} - \frac{1}{(1-r)^2}} > 0 \tag{28}
$$

This shows that as $P_G$ increases, the optimal $r$ should also increase. We verify the theorem.

# C Supplementary Experimental Results

## C.1 Detailed Experimental Setup

**Datasets and Models.** To ensure the reliability, we adapt the configurations provided by MarkLLM (Pan et al., 2024), which currently is the most popular LLM watermarking toolkits. Specifically, for dataset, we utilize 400 samples from the C4 dataset (Raffel et al., 2020). The first 30 tokens of each text serve as prompts to generate new tokens. We set the output length to be at least 200 and at most 230 tokens. We also follow MarkLLM and employ OPT-1.3B, -2.7B and -6.7B (Zhang et al., 2022) as our models.
**Baselines.** In this paper, we focus exclusively on flexible watermarking methods that do not require training any additional models, as they offer more promising practical applicability. Consequently, we exclude watermarking techniques that necessitate model training, such as SIR (Liu et al., 2024a) and TS (Huo et al., 2024). The baseline methods include: (1) UnWM, representing the original unwatermarked outputs; (2) KGW (Kirchenbauer et al., 2023), the fundamental method; (3) UW (Hu et al., 2024) and DiPmark (Wu et al., 2024), which implement unbiased watermark techniques; (4) SWEET (Lee et al., 2024) and EWD (Lu et al., 2024), both designed for watermarking in low-entropy scenarios. Implementation details can be found in App. C.1.
**Evaluation Metrics.** We evaluate MorphMark and baselines in watermark effectiveness and text quality. The evaluation of *effectiveness* focuses on both detectability and robustness. We assess detectability using True positive rate at 1% false positive rate (TPR@1%). We also report the Best F1 Score (Best F1) to present the highest F1 score achieved with the optimal balance of TPR and FPR during detection. To assess the robustness of watermark methods, we employ the Word-S/30% attack, which randomly replaces words with synonyms from WordNet (Miller, 1995). We report the TPR@1% and Best F1 of watermarking methods against the Word-S/30% attack, denoted as TPR@1%(Word-S/30%) and Best F1(Word-S/30%). From a *text quality* perspective, we evaluate the Perplexity (PPL) of generated texts, computed using LLaMA-2-7B (Touvron et al., 2023). All experiments are performed on an Ubuntu 18.04 system with an AMD EPYC 7Y83 64-core CPU and a NVIDIA RTX 4090 GPU.
**Implementation Details.** For KGW, SWEET and EWD , and the $\delta$ in their methods is set to 1.25. For SWEET, the entropy threshold is set to 0.9. For UW, we use $\gamma$-reweight. For DiPmark, $\alpha$ is set to 0.45. + For MorphMark$_{linear}$, MorphMark$_{exp}$ and MorphMark$_{log}$, we set $k_{linear}$, $k_{exp}$ and $k_{log}$ to 1.55, 1.30 and 2.15, respectively. $p_0$ in MorphMark is fixed to 0.15. $\epsilon$ is fixed to $10^{-10}$. For all methods, we set the green list ratio to 0.5.

## C.2 Configuration of Doc-P(GPT-3.5) Attack

For Doc-P(GPT-3.5) attack, we use the version gpt-3.5-turbo-0125 API. The prompt for paraphrasing is shown in Fig. 7.

Please rewrite the following text (Only return the rewritten text): {Model Output}

Figure 7: Prompt used in Doc-P(GPT-3.5) paraphrasing attack.

## C.3 Trade-off Curve Between Watermark Effectiveness and Text Quality

Here, we plot the trade-off curve and compare MorphMark's three varients with KGW. By adjusting $k_{\text{linear}}$, $k_{\text{exp}}$, $k_{\text{log}}$, and $\delta$, we obtain multiple points, which are visualized in Fig. 8. From the results, we observe that the MorphMark$_{exp}$ outperforms the MorphMark$_{linear}$, which in turn outperforms the MorphMark$_{log}$. All three methods significantly surpass KGW.

## C.4 Different Sampling Parameters of More Methods

We present more results on different sampling parameters in Tab. 3 and Tab. 4.

## C.5 Statistical Distribution of $P_G$ in C4 Dataset

Before, we discuss an extreme case of code generation which make MorphMark low effectiveness. To further explore the occurrence of extreme cases, we use the questions in four popular benchmarks,

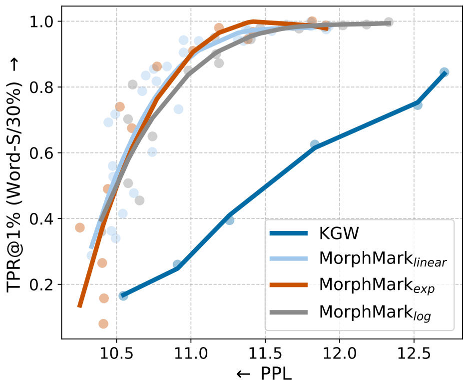


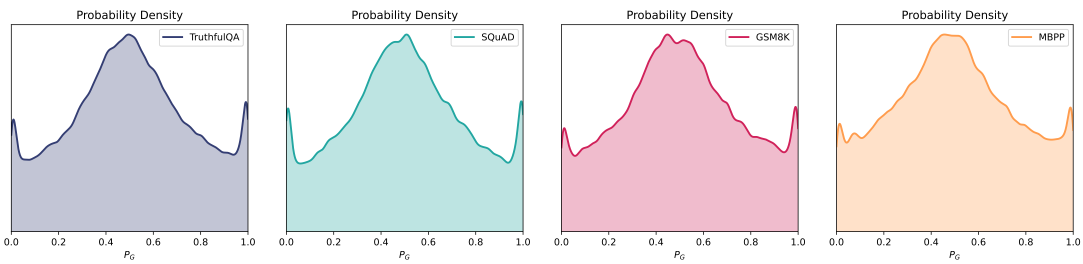

Figure 8: Comparing the performance of different watermark methods. We measure watermark effectiveness with TPR@1%(Word-S/30%) and text quality with PPL.

| (Temp, TopP) | UnWM PPL | PPL | TPR@1% | TPR@1%↑ (Word-S/30%) |
|---|---|---|---|---|
| (0.3, 1.0) | 4.1308 | 4.6790 | 1.0000 | 0.9025 |
| (0.7, 0.95) | 5.4809 | 6.1147 | 0.9950 | 0.9325 |
| (0.9, 0.95) | 7.3829 | 7.9732 | 1.0000 | 0.9325 |
| (1.2, 1.0) | 15.2175 | 16.3252 | 1.0000 | 0.9625 |

Table 3: Performance of MorphMark$_{linear}$.

| (Temp, TopP) | UnWM PPL | PPL | TPR@1% | TPR@1%↑ (Word-S/30%) |
|---|---|---|---|---|
| (0.3, 1.0) | 4.1308 | 4.8056 | 0.9925 | 0.9475 |
| (0.7, 0.95) | 5.4809 | 6.2566 | 0.9950 | 0.9410 |
| (0.9, 0.95) | 7.3829 | 8.0720 | 1.0000 | 0.9400 |
| (1.2, 1.0) | 15.2175 | 16.3264 | 1.0000 | 0.9525 |

Table 4: Performance of MorphMark$_{log}$.




(a) TruthfulQA (b) SQuAD (c) GSM8K (d) MBPP

Figure 9: Statistical Distribution of $P_G$.

i.e., TruthfulQA (Lin et al., 2021), SQuAD (Rajpurkar, 2016; Rajpurkar et al., 2018), GSM8K (Cobbe et al., 2021) and MBPP (Austin et al., 2021). For each dataset, we randomly sample 400 questions and subsequently analyze the resulting $P_G$ distribution, as shown in Fig. 9. These empirical results indicate that the $P_G$ distribution is generally broad. Although the code generation dataset MBPP exhibits more values close to 0 and 1 compared to other datasets, its overall distribution remains broad. This observation suggests that MorphMark is effective in a wide range of scenarios.

## D Full Related Work

**Watermarking in the Era of LLMs** Modern watermarking techniques for large language models (LLMs) differ significantly from earlier backdoor-based approaches, primarily due to the high costs of training such models. Instead of embedding watermarks during training, contemporary methods apply them during the sampling phase of text generation. The pioneering method in this space is KGW (Kirchenbauer et al., 2023), which utilizes a user-defined key and the previous token as a random seed to split the vocabulary into "green" and "red" lists. The model then increases the probabilities of green-list tokens to embed the watermark. Since KGW's introduction, numerous techniques have sought to enhance its performance from various perspectives.

**Unbiased Watermarking** Unbiased watermarking ensures that the expected token distribution under watermarking remains identical to the original. The first method to achieve this, EXP, is highly computationally expensive. For example, Wu et al. (2024) reports that EXP can require up to 500 times the generation time of KGW. More efficient alternatives, such as UW and DipMark, leverage inverse sampling and permutation-based reweighting to strike a balance between detection efficacy and text quality. However, their robustness has yet to be thoroughly validated.

**Semantics-Based Watermarking** A growing body of research (Ren et al., 2024; Liu et al., 2024a; He et al., 2024b; Guo et al., 2024) has explored the use of semantic information, rather than previous tokens, as keys for embedding watermarks. This approach enhances robustness without increasing watermark strength, thereby preserving text quality. However, many of these methods require auxiliary models, reducing their flexibility. Among them, SIR (Liu et al., 2024a) demonstrated the strongest performance in the MarkLLM benchmark, making it a key baseline in our study.

**Low-Entropy Watermarking** Low-entropy contexts involve highly deterministic token generation—e.g., completing The quick brown fox jumps over a lazy, where dog is the most probable next token. In such cases, watermarking can degrade text quality. Methods like SWEET (Lee et al., 2024) and ATW (Liu and Bu, 2024) mitigate this by setting entropy thresholds, embedding watermarks only when token uncertainty is sufficiently high. EWD (Lu et al., 2024) takes a different approach, maintaining the KGW framework but assigning higher detection weights to high-entropy tokens. However, these techniques often require access to the original model during detection, limiting practicality—especially ATW, which relies on three auxiliary models, making both watermarking and detection computationally expensive.

**Other Watermarking Techniques** Unigram (Zhao et al., 2024) improves robustness by using a fixed vocabulary partition instead of dynamically adjusting token probabilities based on prior tokens. However, this fixed division is vulnerable to watermark extraction techniques (Jovanović et al., 2024), making it impractical for real-world applications. TS (Huo et al., 2024) converts the hyperparameters in KGW into two neural networks and designs a loss function for training to enhance both watermark effectiveness and text quality. However, this approach not only lacks interpretability, but also requires retraining a new watermark parameter neural network for every new model. More importantly, in practical applications, the watermark strength is difficult to control manually and becomes unpredictable due to its training-based nature.