Subtle Signatures, Strong Shields: Advancing Robust and Imperceptible Watermarking in Large Language Models

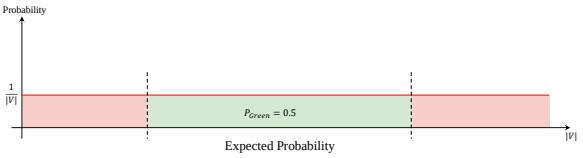
Yubing Ren[1,2]*, **Ping Guo**[1,2]*, **Yanan Cao**[1,2]†, **Wei Ma**[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
renyubing@iie.ac.cn, guoping@iie.ac.cn

Abstract

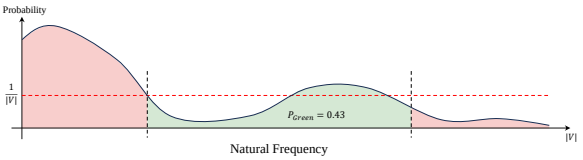
The widespread adoption of Large Language Models (LLMs) has led to an increase in AI-generated text on the Internet, presenting a crucial challenge to differentiate AI-created content from human-written text. This challenge is critical to prevent issues of authenticity, trust, and potential copyright violations. Current research focuses on watermarking LLM-generated text, but traditional techniques struggle to balance robustness with text quality. We introduce a novel watermarking approach, Robust and Imperceptible Watermarking (RIW) for LLMs, which leverages token prior probabilities to improve detectability and maintain watermark imperceptibility. RIW methodically embeds watermarks by partitioning selected tokens into two distinct groups based on their prior probabilities and employing tailored strategies for each group. In the detection stage, RIW method employs the '*voted z-test*' to provide a statistically robust framework to identify the presence of a watermark accurately. The effectiveness of RIW is evaluated across three key dimensions: success rate, text quality, and robustness against removal attacks. Our experimental results on various LLMs, including GPT2-XL, OPT-1.3B, and LLaMA2-7B, indicate that RIW surpasses existing models, and also exhibits increased robustness against various attacks and good imperceptibility, thus promoting the responsible use of LLMs. Our code is available at https://github.com/Liliceer/RIW.

1 Introduction

Large Language Models (LLMs) such as ChatGPT (OpenAI, 2022) and LLaMA (Touvron et al., 2023) have marked a significant advancement in natural language processing, enabling a range of applications from generating human-like text to understanding complex language patterns. However, this success has also led to pressing challenges: the spread of fake news, the risk of copyright violations, and a rise in academic dishonesty. These issues highlight the necessity for reliable methods to detect and audit machine-generated text. Addressing these concerns is essential for mitigating potential harms associated with LLMs and ensuring their responsible utilization in various domains. To meet that need, watermarking has emerged as an effective method for tagging and detecting machine-generated text from LLMs. A watermark in this context is a distinct yet inconspicuous pattern embedded within the text. It is designed to be imperceptible to human readers but allows for the algorithmic identification of the text as synthetic.



(a) Expected probability of tokens in the green list by previous watermark methods.



(b) Natural Frequency of tokens in the green list.

Figure 1: The disparity between the expected probability and natural frequency of tokens on the green list.

One major way of using watermarks for LLMs is watermarking during logit generation (Kirchenbauer et al., 2023a; Lee et al., 2023; Hu et al., 2023; Wu et al., 2023; Yoo et al., 2023; Wang et al., 2023; Zhao et al., 2023; Kirchenbauer et al., 2023b; Liu et al., 2023a,b; Ren et al., 2023). This watermarking technique typically involves segmenting the vocabulary into two categories: a 'green list' and a 'red list'. The approach is to preferentially utilize tokens from the green list in the AI-generated text. During the detection phase, the actual frequency of these green list tokens within the text is calculated

*Equal Contribution.
†Corresponding Author.

Findings of the Association for Computational Linguistics: ACL 2024, pages 5508–5519
August 11-16, 2024

and compared against an expected probability, exemplified in Figure 1 with an expectation set at 0.5. A significant deviation from this expected probability suggests AI involvement in the text's generation, calculated through a statistical method, the *z-test*. However, the implementation of this method introduces a critical dilemma. The expected probability level for green list tokens can be at odds with the natural frequency distribution of these tokens in regular language use. This discrepancy can lead to either the watermark is not robust enough, making AI-generated text difficult to distinguish under attack, or impacting the overall quality of the text. Balancing these concerns is crucial: the watermark must be detectable without compromising the authenticity and readability of the generated content.

The disparity between the expected probability and natural frequency of tokens on the green list leads to two challenges: From the broader scope, as illustrated in Figure 1, the natural frequency (at around 0.43) is lower than the expected probability (set at 0.5). When dividing vocabulary list evenly into red and green, previous studies usually set the *z-test* expected value to 0.5, this will lead to a sluggish response: need to first increase the frequency of these tokens to reach the expected value before any overlap in detection results becomes visible. On an individual token basis, uniformly increasing the logits for all tokens on the green list may not be equally effective. For tokens with inherently low frequency, substantial adjustments to their logits might not significantly alter their frequency in the generated text, as shown in Figure 1. This ineffectiveness suggests the need for a more tailored approach, where adjustments are made by the specific prior probabilities of each token. This allows for more precise detection with minimal, potentially negligible, impact on the overall quality of the text.

In this paper, we introduce a novel watermarking approach termed the Robust and Imperceptible Watermarking (RIW) algorithm for LLMs, based on the approach of Kirchenbauer et al. (2023b). Our focus centers on applying prior probabilities of tokens in watermarks to remain detectable under various types of realistic attacks and improve the watermarking imperceptibility. Concretely, RIW introduces a two-tiered process: a nuanced watermark injection strategy paired with a sophisticated watermark detection mechanism. The watermark injection phase begins with a meticulous selection of tokens from the LLM's vocabulary, forming the foundational 'green list'. This list undergoes a deliberate partitioning into two green sublists based on prior probabilities. Then we design different modifications to the logits of the token in each sublist, ensuring a sensitive embedding process with imperceptible and minimally intrusive to the original text. This step skillfully balances the randomness inherent in LLM outputs with the predictability necessary for watermark detection. In watermark detection, RIW employs the '*voted z-test*', capitalizing on the prior probabilities of each token, providing a statistically precise and robust framework to accurately identify the presence of a watermark. What sets RIW apart is its harmonious blend of robustness and imperceptibility, ensuring the integrity and naturalness of watermarked text.

We comprehensively evaluate the watermarking algorithm using three critical criteria: **Success Rate:** an indicator of the algorithm's effectiveness in accurately identifying watermarked texts. **Text Quality:** determine if the watermarking process compromises the naturalness and readability of the text. **Robustness:** tested by assessing the watermark's resilience to various attack strategies, ensuring that the watermark remains detectable even after potential tampering. Through extensive experimentation on models of varying sizes, including GPT2-XL (Radford et al., 2019), OPT-1.3B (Zhang et al., 2022), and LLaMA2-7B (Touvron et al., 2023), RIW demonstrates superior performance over existing methods like KGW (Kirchenbauer et al., 2023a) and Unigram-WM (Zhao et al., 2023). RIW not only attains enhanced detection accuracy but also exhibits increased robustness against various attacks and good imperceptibility.

2 Related Work

Large Language Models (LLMs) offer the potential for embedding watermarks at various stages of text generation: during the training phase, within the logit generation process, and throughout token sampling. Each stage presents distinct approaches and challenges for watermarking.

2.1 Training Time Watermarking

At this stage, watermarks are incorporated directly into the training data. The technique involves selecting a subset of the training data to embed a watermark through specific triggers, which are then mixed with the rest of the data. Early methods

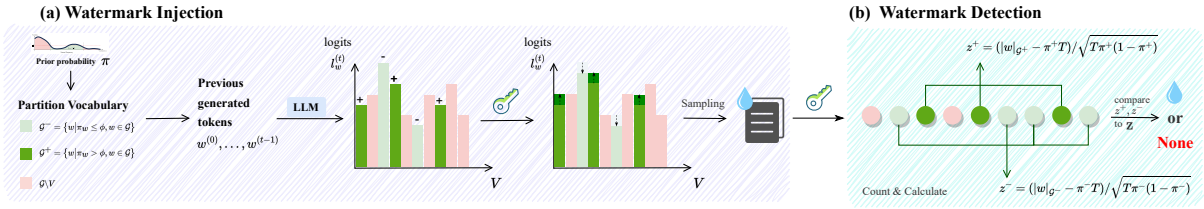


Figure 2: Overview of the process of RIW, including watermark injection and watermark detection. To inject watermark, we first partition the green list tokens into positive and negative groups according to token prior probabilities and employ different strategies for each group. During detection, We separately detect the deviation of each group from the prior probabilities.

involve introducing triggers that subtly alter the input-output relationship, affecting only certain inputs. For example, Liu et al. (2023c) experimented with embedding triggers at various levels (character, word, sentence) for text classification, while Sun et al. (2022) and Tang et al. (2023) applied similar concepts to code generation and adversarial learning, respectively. Sun et al. (2023) focused on code transformations that preserve semantic meaning but carry watermarks. Despite its potential, this approach is limited by its specificity to certain inputs, low capacity for embedding information, and the significant cost of retraining for any changes to the watermark.

2.2 Watermarking during Logits Generation

This method modifies the logits output by LLMs to embed watermarks, effectively altering the model's prediction probabilities. The pioneering work by Kirchenbauer et al. (2023a) introduced a technique of dividing the vocabulary into two lists (red and green) and using these to detect watermarked text. Follow-up studies have refined this approach, focusing on enhancing the efficiency and robustness of watermark detection and embedding. Techniques include adjusting logits to favor certain tokens (Lee et al., 2023; Wang et al., 2023), employing multi-color partitions for more complex watermarks (Yoo et al., 2023), and proving robustness against removal attempts (Zhao et al., 2023). Additional innovations involve transforming text or semantic embeddings directly into watermarked logits (Liu et al., 2023b; Ren et al., 2023), and developing methods to reduce bias in watermarking (Hu et al., 2023; Wu et al., 2023).

2.3 Watermarking during Token Sampling

Embedding watermarks during the token selection phase takes advantage of the inherent randomness in choosing the next word in a sequence. Initial efforts in this area (Christ et al., 2023; Kuditipudi et al., 2023) have introduced novel methods that use binary representations and pseudo-random number sequences to embed watermarks more subtly. While promising, this domain is still emerging, with significant potential for further exploration and refinement to improve practical applicability and robustness.

3 Methodology

In this section, we will introduce the process of embedding our designed watermark to LLMs as well as the mechanism for detecting such watermarks within sequences. To inject watermarks, we randomly select several tokens from the vocabulary and partition these tokens into two distinct sublists based on their prior probabilities. Then, we develop diverse watermarking strategies tailored to each list to ensure the process remains efficient. For detection, we have developed a *'voted z-test'*, which leverages the prior probabilities of each token to determine the watermark's presence.

3.1 Notations & Preliminaries

In LLMs, the input text is segmented into discrete units known as "tokens". Let V denotes the vocabulary from which LLMs draw tokens, and let w represent the token within V, a sequence comprising T tokens, can be denoted by $w^{(0)}, \ldots, w^{(t)}, \ldots, w^{(T)}$, where t indexes the position of the token within this sequence. LLMs predominantly employ auto-regressive decoding for next-word prediction. This process yields a set of logits $l^{(t)}$ upon receiving the preceding known token sequence $w^{(0)}, \ldots, w^{(t-1)}$. These logits $l^{(t)} \in \mathbb{R}^{|V| \times d}$, where d represents the embedding dimensionality and $|V|$ signifies the size of the vocabulary, are converted into a discrete probability distribution $p^{(t)}$ via a softmax function. The subsequent token $w^{(t)}$ is then probabilistically selected from $p^{(t)}$, employing strategies such as greedy sampling or beam search.

In the domain of LLMs, our watermark injection and detection framework leverages the Gaussian distribution of token frequencies. We have meticulously analyzed 300k news articles from the C4 dataset (Raffel et al., 2020) to establish the nature of word occurrence, which in turn are normalized as the prior probabilities, π_w, for each token w in our vocabulary. This probabilistic foundation is pivotal to our watermarking strategy. Our innovative use of normalized linguistic statistics not only underpins the watermarking process but also enhances the robustness and subtlety of our watermark detection.

3.2 Watermark Injection

We commence with an overview of our watermark injection procedure in Algorithm 1, providing a foundational understanding. A detailed exposition of the methodology follows in this section. Our objective is to inject a watermark into LLMs that is both robust and inconspicuous. To achieve this, the injection process must introduce minimal perturbation to the logits of unmarked LLM. Guided by this principle, we propose the subsequent hypothesis:

Consider a sequence generation process in LLM. For any token w with prior probability π_w, we propose the following:

- *If π_w is relatively **high**, the perturbation magnitude required to increase the observed frequency of w **is less than** that to decrease the occurrence frequency.*

- *If π_w is relatively **low**, the perturbation magnitude required to increase the observed frequency of w **is more than** that to decrease the occurrence frequency.*

This suggests that the perturbation direction for watermarking a token w in an LLM output is determined by its inherent prior probability π_w. Following the methodology described in Kirchenbauer et al. (2023a), we select a subset of tokens, denoted as the "green list" $\mathcal{G}$, from the vocabulary for watermarking. These tokens are subsequently divided into two sublists predicated on their prior probabilities π_w. We introduce ϕ as the threshold of prior probability. Tokens from $\mathcal{G}$ with prior probabilities surpassing ϕ are allocated to the positive green list $\mathcal{G}^+$, while the remainder forms the negative green list $\mathcal{G}^-$. The construction of these sublists is governed by the ensuing formulation:

$$\begin{aligned} \mathcal{G}^+ &= \{w | \pi_w > \phi, w \in \mathcal{G}\}, \\ \mathcal{G}^- &= \{w | \pi_w \leq \phi, w \in \mathcal{G}\}. \end{aligned} \tag{1}$$

Drawing on the principles established in the above hypothesis, we tailor the watermark perturbation to align with the positive and negative lists of the green list. For tokens w^+ within the positive green list $\mathcal{G}^+$, we aim to amplify their frequency of occurrence. This is achieved by augmenting their corresponding logits by a constant δ. In contrast, for the tokens w^- categorized under the negative green list $\mathcal{G}^-$, our goal is to reduce their frequency of occurrence. To this end, we decrement their logits by the same constant δ. Consequently, after implementing our watermarking scheme, the adjusted logits are processed by the softmax function, yielding a revised probability distribution $\tilde{p}_w^{(t)}$ across the vocabulary:

$$\tilde{p}_w^{(t)} = \begin{cases} \frac{\exp(l_{w^-}^{(t)} - \delta)}{\sum_{w \setminus \mathcal{G}} \exp(l_{w \setminus \mathcal{G}}^{(t)}) + \sum_{w^-} \exp(l_{w^-}^{(t)} - \delta) + \sum_{w^+} \exp(l_{w^+}^{(t)} + \delta)} \\ \frac{\exp(l_{w^+}^{(t)} + \delta)}{\sum_{w \setminus \mathcal{G}} \exp(l_{w \setminus \mathcal{G}}^{(t)}) + \sum_{w^-} \exp(l_{w^-}^{(t)} - \delta) + \sum_{w^+} \exp(l_{w^+}^{(t)} + \delta)} \\ \frac{\exp(l_{w \setminus \mathcal{G}}^{(t)})}{\sum_{w \setminus \mathcal{G}} \exp(l_{w \setminus \mathcal{G}}^{(t)}) + \sum_{w^-} \exp(l_{w^-}^{(t)} - \delta) + \sum_{w^+} \exp(l_{w^+}^{(t)} + \delta)} \end{cases} \tag{2}$$

where $w^{\setminus \mathcal{G}}$ refers to the tokens not in green list and $\mathbb{1}$ denotes the Dirichlet function. The application of divergent watermarking directions for the respective green lists enhances the efficacy and sensitivity of our watermark, even when the magnitude of δ is

Algorithm 1 Watermark Injection

Input: previous generated tokens, $w^{(0)}, \ldots, w^{(t-1)}$
prior probability, π_w for $w \in V$
green list size, $\gamma \in (0, 1)$
hardness parameter, $\delta > 0$

1: Seed a random number generator, and use it to select a "green list" $\mathcal{G}$ from the vocabulary V of size $\gamma|V|$.
2: Partition the "green list" $\mathcal{G}$ into one positive green list $\mathcal{G}^+$ and one negative green list $\mathcal{G}^-$ using Eq. 1.
3: **for** $t = 0, 1, \ldots$ **do**
4: Apply the language model to previous generated tokens, $w^{(0)}, \ldots, w^{(t-1)}$ to get a logit vector $l_w^{(t)}$ over the vocabulary.
5: Apply watermark δ to each green list logit and use the softmax operator to these modified logits to get a probability distribution over the vocabulary according to Eq. 2.
6: Sample the next token, $w^{(t)}$, using the watermarked distribution $\tilde{p}^{(t)}$.
7: **end for**

Algorithm 2 Watermark Detection

Input: Text sequence of length T
Positive green list $\mathcal{G}^+$ and Negative green list $\mathcal{G}^-$
chosen threshold, $\mathbf{Z} > 0$
Calculate the prior probability π^+ and π^- with Eq. 3.
$|w|_{\mathcal{G}^+} \leftarrow 0$
$|w|_{\mathcal{G}^-} \leftarrow 0$
for $w \in \{w^{(0)}, \ldots, w^{(t)}, \ldots, w^{(T)}\}$ **do**
  if $w \in \mathcal{G}^+$ **then**
    $|w|_{\mathcal{G}^+} \leftarrow |w|_{\mathcal{G}^+} + 1$
  else if $w \in \mathcal{G}^-$ **then**
    $|w|_{\mathcal{G}^-} \leftarrow |w|_{\mathcal{G}^-} + 1$
  end if
end for
Calculate the positive z score z^+ and negative z score z^- with Eq. 4.
if $z^+ > \mathbf{Z} \bigcup z^- < -\mathbf{Z}$ **then**
  Reject null hypothesis (Watermarked Text Sequence).
else
  Accept null hypothesis (Natural Text Sequence).
end if

comparatively minor. This strategy affords a watermarking technique for LLMs that is both robust, to ensure persistence, and non-intrusive, to maintain original model's performance integrity.

3.3 Watermark Detection

In line with the method presented in previous research (Kirchenbauer et al., 2023a), the detection of watermarks does not necessitate direct interaction with the LLM. The detection process requires the computation of the prior probabilities for the tokens in both the positive and negative green lists. Let π^+ and π^- symbolize the aggregate prior probabilities of the tokens within the positive green list and the negative green list, respectively. The computation of π^+ and π^- is as follows:

$$\begin{aligned} \pi^+ &= \sum_{w \in \mathcal{G}^+} \pi_w, \\ \pi^- &= \sum_{w \in \mathcal{G}^-} \pi_w. \end{aligned} \tag{3}$$

Upon the watermark's integration, we anticipate an increased occurrence frequency for tokens in the positive green list, surpassing their initial prior probability π^+. Conversely, tokens in the negative green list should manifest a reduced presence compared to their prior probability π^-. This divergence from the expected prior probabilities enables us to detect the watermark by examining the empirical data against the following null hypothesis:

H_0: *The sequence of text is produced in accordance with the original token prior distribution.*

To assess H_0, we employ a robust detection mechanism utilizing the one proportion z-test. If the null hypothesis holds, then in a text sequence containing T tokens, the expected frequency of tokens from positive green list should be $\pi^+ T$. Similarly, the expected frequency for negative green list tokens would be $\pi^- T$. Here use $|w|_{\mathcal{G}^+}$ and $|w|_{\mathcal{G}^-}$ to denote the number of positive and negative green list tokens, respectively. The z-statistics for the positive green list denoted as z^+, and for the negative green list, denoted as z^-, are computed as:

$$\begin{aligned} z^- &= (|w|_{\mathcal{G}^-} - \pi^- T)/\sqrt{T\pi^-(1-\pi^-)}, \\ z^+ &= (|w|_{\mathcal{G}^+} - \pi^+ T)/\sqrt{T\pi^+(1-\pi^+)}. \end{aligned} \tag{4}$$

We determine the presence of the watermark by comparing the computed z^+ and z^- to a predefined threshold $\mathbf{Z}$. The null hypothesis is rejected if either $z^+ > \mathbf{Z}$ or $z^- < -\mathbf{Z}$. Conversely, to uphold the null hypothesis, both conditions must be met: $z^+ \leq \mathbf{Z}$, and $z^- \geq -\mathbf{Z}$. An algorithmic representation of our watermark detection procedure is provided in Algorithm 2.

4 Experiments

We conduct experiments assessing the performance of watermark detection, the quality of watermarked text, and its robustness against attacks, in comparison to established baseline methods.

4.1 Experimental Settings

We explore the behavior of the watermark using three public language models of varying sizes and model families: GPT2-XL with 1.5B parameters (Radford et al., 2019), OPT-1.3B (Zhang et al., 2022), and LLaMA2-7B (Touvron et al., 2023).

Datasets. The models are tested on three long-form text datasets: C4 dataset (Raffel et al., 2020), OpenGen, and LFQA (Krishna et al., 2023). Each dataset finally generates 500 watermarked and 500 unwatermarked texts for evaluation. For C4, We follow the setting of Kirchenbauer et al. (2023a) to slice and dice a random selection of texts. For OpenGen and LFQA, we follow Zhao et al. (2023) to use human-written prefixes or questions as prompts and the corresponding suffixes or answers as human-written text. We only keep the generated sentences with lengths over 200 tokens.

Evaluation metrics. Watermark detection is a binary classification task, assessed by F1 score for overall accuracy. We prioritize false positive and

Method	C4					OpenGen					LFQA					Avg. Benchmark		
	FPR	TNR	TPR	FNR	F1	FPR	TNR	TPR	FNR	F1	FPR	TNR	TPR	FNR	F1	TNR	TPR	F1
GPT2-XL																		
KGW	0.001	0.999	0.996	0.004	0.997	0.001	0.999	0.997	0.003	0.998	0.000	1.000	1.000	0.000	1.000	**0.999**	0.998	0.998
Uni-WM	0.298	0.702	1.000	0.000	0.870	0.025	0.975	0.999	0.001	0.987	0.207	0.793	1.000	0.000	0.906	0.823	1.000	0.921
RIW (ours)	0.000	1.000	0.999	0.001	0.999	0.000	1.000	0.998	0.002	0.999	0.002	0.998	1.000	0.000	0.999	**0.999**	**0.999**	**0.999**
OPT-1.3B																		
KGW	0.000	1.000	0.992	0.008	0.996	0.000	1.000	0.996	0.004	0.998	0.000	1.000	1.000	0.000	1.000	1.000	0.996	0.998
Uni-WM	0.007	0.993	0.996	0.004	0.995	0.001	0.999	0.999	0.001	0.999	0.003	0.997	1.000	0.000	0.999	0.996	**0.998**	0.998
RIW (ours)	0.000	1.000	0.997	0.003	0.999	0.000	1.000	0.998	0.002	0.999	0.000	1.000	1.000	0.000	1.000	**1.000**	**0.998**	**0.999**
LLaMA2-7B																		
KGW	0.000	1.000	0.927	0.073	0.962	0.000	1.000	0.920	0.080	0.958	0.000	1.000	0.782	0.218	0.878	**1.000**	0.876	0.933
Uni-WM	0.000	1.000	0.977	0.023	0.988	0.001	0.999	0.970	0.030	0.984	0.000	1.000	0.978	0.022	0.989	**1.000**	0.975	0.987
RIW (ours)	0.000	1.000	0.995	0.005	0.997	0.000	1.000	0.989	0.011	0.995	0.000	1.000	0.994	0.006	0.997	**1.000**	**0.993**	**0.996**
Avg. Base Model																		
KGW	0.000	1.000	0.972	0.028	0.985	0.000	1.000	0.971	0.029	0.985	0.000	1.000	0.927	0.073	0.959	1.000	0.957	0.976
Uni-WM	0.102	0.898	0.991	0.009	0.951	0.009	0.991	0.989	0.011	0.990	0.070	0.930	0.993	0.007	0.965	0.940	0.991	0.969
RIW (ours)	0.000	1.000	0.997	0.003	0.998	0.000	1.000	0.995	0.005	0.998	0.001	0.999	0.998	0.002	0.999	**1.000**	**0.997**	**0.999**

Table 1: Performance comparison of our method (RIW) and KGW, Uni-WM. The highest average value is marked with bold text.

false negative rates, given the higher risk false positives pose by mislabeling genuine texts. Additionally, we use hypothesis testing to calculate z-scores and p-values, further gauging detection success.

Baselines. We choose the first LLM watermark method KGW (Kirchenbauer et al., 2023a) and its improve work on robustness, Uni-WM (Zhao et al., 2023), as our baselines.

Hyper-parameters. In our experiment, we set the green list proportion γ to 0.5 and the hardness parameter $\delta = 2.0$, following previous baseline Kirchenbauer et al. (2023a). We set the z-score threshold $\mathbf{Z}$ to 4. For the threshold of prior probability ϕ, we set it to $2/|V|$. Details about how we select ϕ are shown in Section C. More thorough experimental details are in Appendix B.4.

4.2 Watermarking Success Rate

Table 1 presents the overall watermark detection results for three datasets and three base models.

RIW consistently outperforms the KGW and Uni-WM across various datasets and base model configurations, evidencing its remarkable generalization. Specifically, RIW exhibits a perfect TPR of **1.000** averaged across all datasets and models, signifying no false positives, which is crucial given the higher impact of false positives in watermarking contexts. Additionally, the TNR for RIW is average at **0.997**, further underscoring its precision in identifying genuine content. For the challenging LFQA dataset, while the performance of other methods, particularly KGW, significantly diminishes (e.g., TPR drops to 0.782 for LLaMA2-7B), RIW maintains a TPR above 0.994. Similar improvement can be found in the Avg. Benchmark column, where RIW achieves an F1 score of **0.999**, outstripping the baseline models, demonstrating a high success rate in watermark detection. The consistently low FPR (**almost 0**), especially in comparison to Uni-WM reaching 0.102 in the Avg. Base Model for C4 underscores RIW's precision in avoiding false positives. These performances not only prove RIW's best-in-class performance but also its significant advancements in detection under diverse conditions.

Conservative in KGW vs. Radical in Uni-WM. (a) KGW's conservative watermark detection is evident, with its FPR never exceeding 0.008 across all datasets and models, indicating that it is less prone to incorrectly identifying non-watermarked text as watermarked. For instance, across the C4, OpenGen, and LFQA datasets, KGW maintains an FPR well **below 0.01** in most cases, suggesting it errs on the side of caution and is less likely to generate false alarms. However, this conservative strategy also resulted in a lower TPR, a **4%** reduction compared to RIW. (b) In contrast, Uni-WM's more aggressive approach yields higher FPRs, notably a 0.298 with the GPT2-XL model on the C4 dataset, indicating a greater likelihood of misclassifying genuine text as watermarked. However, this approach also results in Uni-WM's TPR being generally higher than KGW's, such as achieving **0.977** on the LLaMA2-7B model compared to KGW's 0.927, pointing to its higher sensitivity. Both methods show sensitivity to the experimental setup and hyperparameter configuration, which can skew their watermark detection scale toward one extreme. For example, in the LFQA dataset,

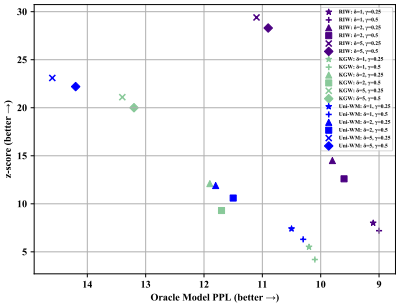
KGW's and Uni-WM's FPRs vary widely depending on the base model. This balance between false positives and negatives is critical, as it affects the practicality and trustworthiness of the watermark detection system. Too conservative an approach might overlook actual watermarked content, while too aggressive could undermine the reliability of the system by flagging genuine content as inauthentic. So it is challenging to achieve an optimal balance between detecting watermarks and avoiding false alarms, as shown in the results for RIW.

RIW exhibits a remarkable balance in watermark detection, as indicated by its consistently low FPR and high TPR. For instance, RIW maintains a zero FPR averaged across all datasets and models, clearly demonstrating its reliability in not misclassifying genuine content as watermarked. Simultaneously, its TPR is robust, staying above **0.989** in most cases, such as with the most difficult LLaMA2-7B model, ensuring that actual watermarked content is seldom overlooked. These metrics underscore RIW's precision and its ability to maintain system integrity, effectively navigating the trade-off between sensitivity and specificity that often challenges watermarking algorithms. RIW's superior performance can be attributed to its incorporation of prior probabilities, ensuring that word frequency is measured with precision. By factoring in prior probabilities, RIW achieves a finely tuned balance between sensitivity to watermarks and specificity to authentic content, a balance that is crucial for the watermarking in LLMs.

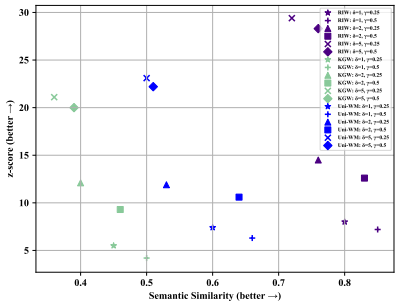
4.3 Watermark Strength vs Text Quality

Figure 3 shows the tradeoff between watermark strength (z-score) and text quality (perplexity, semantic score) for various combinations of watermarking parameters.

Z-score vs PPL. RIW can achieve a very strong watermark, while also maintaining a high-quality text generation. The proximity of RIW data points to the optimal lower right corner reflects its superior capability to embed watermarks effectively without degrading the naturalness or fluency of the text compared to KGW and Uni-WM. This suggests that RIW maintains text quality (as evidenced by lower perplexity values) while achieving a strong watermark (reflected by higher z-scores). This is in contrast to KGW and Uni-WM, which either sacrifice text quality for watermark strength or vice versa. RIW's performance suggests a more optimized balance, demonstrating its robustness and adaptability across varied settings.



(a) Tradeoff between Avg. z-score and LM perplexity



(b) Tradeoff between Avg. z-score and cos similarity

Figure 3: Watermark Strength vs Text Quality

Z-score vs Semantic Similarity. RIW points cluster towards higher semantic similarity scores while also maintaining elevated z-scores, illustrating its capacity to produce watermarks that are both strong and semantically consistent with the original content. This balance showcases RIW's superiority over KGW and Uni-WM, asserting its proficiency in embedding watermarks that are robust to detection yet remain imperceptible, thus preserving the content's original subtlety and ensuring non-intrusiveness for users. This dual excellence makes RIW an ideal choice for practical applications that require maintaining the subtlety of the watermark without alerting users to its presence.

4.4 Robustness to Real-world Attacks

In text watermarking, a crucial evaluation metric is its robustness against watermark removal attacks. A watermark removal attack refers to the process of altering watermarked text in an attempt to erase the embedded watermark. If a watermarked text still has a high probability of being detected following a Watermark Removal Attack, then the text watermarking algorithm is considered highly robust. To provide comprehensive evidence of its robustness, we conduct experiments to test its resilience

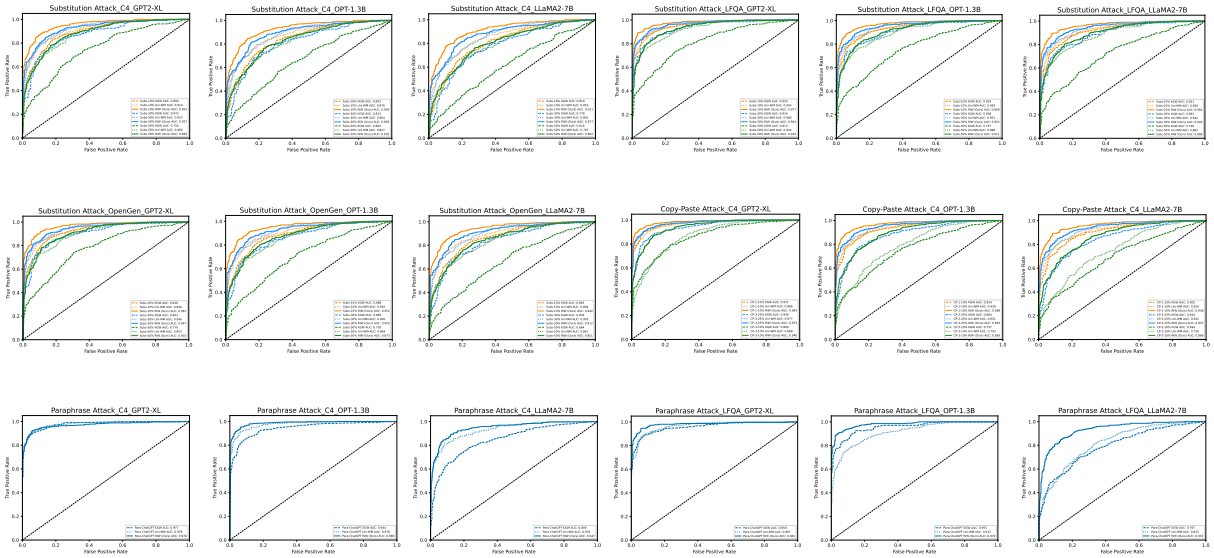


Figure 4: ROC curves with corresponding AUC values for watermark detection against various attack methods.

against word-level attacks (Substitution Attacks) and document-level attacks (Copy-paste Attacks and Paraphrasing Attacks) across various datasets and base models. More Details are in Appendix A.

Substitution Attacks. The ROC curves and AUC values for comparison across various datasets and models indicate RIW's consistently robust watermark detection capabilities facing different levels of substitution attacks. RIW consistently shows higher AUC values, such as **0.977, 0.964, and 0.934** for the 10%, 30%, and 50% substitution levels on LFQA dataset with GPT2-XL, respectively, outperforming KGW and Uni-WM. This is indicative of RIW's superior discriminative power, maintaining higher TPR at lower FPR, hence demonstrating its effectiveness in resisting such attacks across different testing conditions and LLMs. This resilience suggests that RIW is more secure, offering a reliable safeguard in environments where content authenticity is critical.

Copy-paste Attacks. Analyzing the ROC curves against copy-paste attacks, RIW displays a discernible trend of superior resistance compared to KGW and Uni-WM. RIW achieves higher AUC values, such as **0.983** for GPT2-XL, **0.969** for OPT-1.3B, and **0.958** for LLaMA2-7B, indicating its superior capability to maintain accurate watermark detection in the presence of additional text. Notably, RIW's performance across different models—GPT2-XL, OPT-1.3B, and LLaMA2-7B—remains less affected by increasing attack complexity, as indicated by the lesser decline in AUC values with larger text insertions.

Paraphrasing Attacks. The ROC curves for the paraphrase attack scenario across two datasets and three base models indicate RIW's adeptness at watermark retention. Notably, RIW consistently presents higher AUC values, such as **0.986** for OPT-1.3B and **0.970** for GPT2-XL, which highlights its strong performance in maintaining watermark detection even when the text undergoes paraphrasing transformations. These figures suggest that RIW's watermarking technique is resilient to sophisticated linguistic alterations, positioning it as a robust solution for protecting the integrity of content against paraphrase attacks.

5 Conclusion

This paper introduced Robust and Imperceptible Watermarking (RIW), a novel watermarking technique designed for enhancing the traceability of text generated by LLMs. Our evaluation of RIW across various metrics such as success rate, text quality, and robustness has yielded encouraging results, particularly in comparison to existing watermarking methods like KGW and Uni-WM. The experimentation on LLMs of different sizes, including GPT2-XL, OPT-1.3B, and LLaMA2-7B, demonstrated RIW's effectiveness in maintaining watermark integrity while ensuring minimal impact on text quality. While RIW represents a significant step forward in addressing some of the ethical concerns associated with LLM-generated content, future work is vital to refine its application further.

Ongoing research will be directed towards enhancing the algorithm's robustness and exploring its potential in a broader range of LLM applications, thereby contributing to the responsible use of these powerful technologies in various domains.

Limitations

As we tentatively give a successful implementation of watermark LLMs, such paradigm deserves a further and more detailed exploration. First, our calculation of prior probability is quite intuitive and simple, how to better capture the prior probability is still challenging and thrilling, yet still in its fledgeless stage. Aside from it, while extensive experiments demonstrate that RIW consistently improves watermarking on three LLMs, applying our approach to other LLMs will evaluate the effectiveness of our work in a more general way.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (NO.2022YFB3102200).

References

Miranda Christ, Sam Gunn, and Or Zamir. 2023. Undetectable watermarks for language models. *arXiv preprint arXiv:2306.09194*.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. 2023. Unbiased watermark for large language models. *arXiv preprint arXiv:2310.10669*.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023a. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. 2023b. On the reliability of watermarks for large language models.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Frederick Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2023. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*.

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. 2023. Who wrote this code? watermarking for code generation.

Aiwei Liu, Leyi Pan, Xuming Hu, Shu'ang Li, Lijie Wen, Irwin King, and Philip S. Yu. 2023a. An unforgeable publicly verifiable watermark for large language models.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. 2023b. A semantic invariant robust watermark for large language models. *arXiv preprint arXiv:2310.06356*.

Yixin Liu, Hongsheng Hu, Xuyun Zhang, and Lichao Sun. 2023c. Watermarking text data on large language models for dataset copyright protection. *arXiv preprint arXiv:2305.13257*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue. OpenAI blog, 2022.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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. *J. Mach. Learn. Res.*, 21(1).

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*.

Zhensu Sun, Xiaoning Du, Fu Song, and Li Li. 2023. Codemark: Imperceptible watermarking for code datasets against neural code completion models. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1561–1572.

Zhensu Sun, Xiaoning Du, Fu Song, Mingze Ni, and Li Li. 2022. Coprotector: Protect open-source code

against unauthorized training usage with data poisoning. In *Proceedings of the ACM Web Conference 2022*, pages 652–660.

Ruixiang Tang, Qizhang Feng, Ninghao Liu, Fan Yang, and Xia Hu. 2023. Did you train on my dataset? towards public dataset protection with clean-label backdoor watermarking. *arXiv preprint arXiv:2303.11470*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Lean Wang, Wenkai Yang, Deli Chen, Hao Zhou, Yankai Lin, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Towards codable text watermarking for large language models. *arXiv preprint arXiv:2307.15992*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Yihan Wu, Zhengmian Hu, Hongyang Zhang, and Heng Huang. 2023. Dipmark: A stealthy, efficient and resilient watermark for large language models. *arXiv preprint arXiv:2310.07710*.

KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. 2023. Advancing beyond identification: Multi-bit watermark for language models. *arXiv preprint arXiv:2308.00221*.

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*.

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

A Real-world Attack Settings

A.1 Substitution Attack

Substitution attacks on existing text refer to the replacement of words in a pre-generated watermarked text, they are less likely to be mitigated by rule-based methods and align more closely with realistic attack scenarios. To evaluate the robustness against substitution attacks, we follow the setting in Wang et al. (2023) model to carry out word substitution. We conduct these attacks for the watermarked text of RIW, KGW, and Uni-WM. Figure 4 compares the results under a substitution ratio of 10%, 30%, and 50%.

A.2 Copy-paste Attack

Copy-paste attack is to surround the target text with distraction text, which in this context is equal to the watermarked and non-watermarked text respectively. Such an attack will result in a much longer text as compared to the original watermarked text. This attack aims to test if the low ratio setting can cause algorithm effectiveness to drop. To simulate this, we follow the setting in Zhao et al. (2023) to insert a 200-token watermarked text into a 1000-token piece of human-written text, taken from the C4 dataset. The detection involves using a sliding window technique (Kirchenbauer et al., 2023b).

A.3 Paraphrasing Attack

The paraphrasing attack, while offering extensive text modifications, presents greater implementation challenges than word-level methods. We follow the setting in Zhao et al. (2023) to employ the ChatGPT API (`gpt-3.5-turbo`), generating paraphrases by prompting with phrases like "Rewrite the following paragraph:". The outcomes, depicted in Figure 4, demonstrate our method's significant robustness improvement over KGW and Uni-WM.

B Dataset and Model

B.1 Details of Datasets

- C4 dataset: is a part of the family of datasets used by Google in their T5 (Raffel et al., 2020), which is a colossal, cleaned version of Common Crawl's web crawl corpus. We follow Kirchenbauer et al. (2023a) to use the 'RealNews-like' subset, which is specifically tailored to resemble the content and style of real-world news articles.

- OpenGen: collected by Krishna et al. (2023), consists of 3k two-sentence chunks sampled from the validation split of WikiText-103 (Merity et al., 2017). The subsequent 300 tokens serve as the human-written continuation.

- LFQA: a long-form question-answering dataset created by Krishna et al. (2023) by scraping questions from Reddit, posted between July and December 2021, across six do-

mains. It randomly selects 500 questions from each domain and pairs them with their corresponding longest human-written answers, resulting in 3k QA pairs.

B.2 Evaluation metrics

Given that watermark detection inherently represents a binary classification problem — discerning whether a text is watermarked or not — we use classification metrics F1 score to evaluate its success rate. Besides, we use additional metrics including false positive and false negative rates. Here, false positives denote the erroneous classification of non-watermarked texts as watermarked, whereas false negatives refer to the incorrect classification of watermarked texts as non-watermarked. Generally, false positives are more important since misidentifying human-generated texts as watermarked can lead to more adverse consequences. We also employ hypothesis testing methods to calculate the z-score and p-value as an important metric for success rate.

In total, we report true positive rate (TPR), false negative rate (FNR), true negative rate (TNR), false negative rate (FPR), F1 score, z-score, p-value, and ROC curves. We evaluate the quality of the text by calculating perplexity (PPL). For OPT-1.3B, we use the OPT-2.3B model to calculate perplexity. For GPT2-XL, we use the GPT3 (text-davinci-003) (Ouyang et al., 2022) model to calculate perplexity. For OPT-1.3B, we use the OPT-2.3B model to calculate perplexity. As for the LLaMA2-7B model, we use the LLaMA2-13B model to calculate perplexity.

B.3 Details of Baselines

We compare our model with the following previous models.

- KGW (Kirchenbauer et al., 2023a): Based on LLM logits modification, the watermark generator uses a hash function to split the vocabulary into red and green lists at each token position, influenced by the preceding token. Watermarked texts are designed to have a higher green token ratio than unwatermarked texts. The detector categorizes each token as red or green using the same hash function and calculates the green proportion with the z-metric. Texts exceeding a set green token threshold are identified as watermarked.

Threshold	Positive Green List			Negative Green List			RIW
	Size	Prior	TPR	Size	Prior	TPR	TPR
$\phi = 1/\|V\|$	2672	0.44	0.98	22464	0.08	0.76	0.93
$\phi = 2/\|V\|$	1362	0.40	**0.99**	23774	0.12	**0.94**	**1.00**
$\phi = 3/\|V\|$	938	0.38	0.91	24198	0.14	0.94	0.98

Table 2: The threshold selection of prior probability.

- Uni-WM (Zhao et al., 2023): the most robust version of KGW, adopted a fixed global split between red and green lists for generating watermark logits. Such a simple rule is susceptible to being deciphered through statistical analysis of the watermarked text, potentially exposing the tokens classified as green. Nevertheless, this study proved that maintaining a consistent global division between red and green lists contributes to increased robustness against attempts to remove the watermark.

B.4 Implementation Details

We implement RIW using the Pytorch backend of the Huggingface library (Wolf et al., 2020). The `generate` API provides useful abstractions, including modules for warping the logit distribution that comes out of the language model. We generate green lists using the torch random number generator. The experiments are conducted on Nvidia A100 GPUs.

Sample Outputs. We provide a series of representative outputs from different ranges in the sample space for model generations under a soft watermark with parameters $\delta = 2.0$, and $\gamma = 0.5$ under the multinomial sampling scheme. To tabulate these outputs, the ~500 generations collected at this setting are either sorted by the average spike entropy of the watermarked model's output distribution at generation time or the measured test statistic, the z-score for that sequence.

Hash Scheme. Following the hash implementation in Kirchenbauer et al. (2023a), we make use of the hash key to calculate the hash value.

C Further Analysis

The threshold ϕ selection of prior probability. We calculate the respective sizes of positive and negative green lists, the prior probability values, and the TPR of watermark detection based on each specific list. The sum of the positive green list size and the negative green list size equals 25136, which

Prompt	Real Completion	No Watermark (NW)	Watermarked (W)	(W) z^+	(W) z^-	(NW) PPL	(W) PPL
...one count of conspiracy to participate in a corrupt organization, one count of conspiracy to commit theft and nine counts of theft by unlawful taking or disposition. Boubacar Diallo, 32, Brooklyn, NY, is charged with	one count of participating in a corrupt organization, one count of conspiracy to participate in a corrupt organization, one count of conspiracy to receive stolen property and one count of criminal attempt to receive stolen [...continues]	one count of conspiracy to participate in a corrupt organization, one count of conspiracy to receive stolen property. Monte Qualls, 31, Washington, DC, is charged with one count of participating in a corrupt organization [...continues]	one count of participating in a corrupt organization, one count of conspiracy to participate in a corrupt organization, one count of conspiracy to receive stolen property and one count of unauthorized use of automobiles [...continues]	10.99	3.19	3.12	1.90
...The couple moved to Lubbock shortly thereafter and they have been here ever since. They were happily married for 69 years and had three children, Brenda Boydston, Perry Harrison, and Jeanette Pettigrew. He is survived by his wife, Vera Mae	; his three children; ten grandchildren; 21 great-grandchildren; and five great-great grandchildren. He will be dearly missed.Claude joined the U. S. Army in February of 1943 and was deployed to the European Theatre where he participated in the occupation of the Ardennes, Rhineland [...continues]	Harrison of Lubbock; three children, Brenda Boydston of Lubbock, Perry Harrison of Lubbock, and Jeanette Pettigrew of Lubbock; three grandchildren, Jeff Boydston of Dallas, Texas, Eric Boydston of Lubbock, and Michael Pettigrew of Lubbock; and numerous great-grandchildren. [...continues]	Harrison of Lubbock; his sons, Perry and Jeanette of Lubbock; and six grandchildren, three of whom live in the same town. He also had eight great-grandchildren, and eight great-great-grandchildren. The family would like to especially thank his long time and faithful companion, Elanor [...continues]	-1.57	-6.13	4.76	2.66

Table 3: Selected outputs from non-watermarked (NW) and watermarked (W) sampling of C4 dataset, based on OPT-1.3B. Based on the case, even if one z score is wrong, we still have the other z score to balance the performance.

is a 0.5 fraction of OPT-1.3B vocabulary. Table 2 indicates that as the threshold ϕ decreases, more tokens are allocated to the positive green list, enhancing the TPR for watermark detection in that list. Conversely, higher ϕ values correspond to more tokens in the negative green list, leading to an improved TPR for that list. The choice of $\phi = 2/|V|$ represents an optimal balance, evidenced by a perfect TPR of 0.99 for positive watermark detection and a high TPR of 0.94 for negative, showcasing the effectiveness of this middle-ground threshold in the watermarking process for both positive and negative lists.

Case Study. Beyond numerical data, qualitative insights are illustrated in Table 3 through actual prompts and watermarked outputs. The case study showcases two scenarios: a successful watermark detection with high entropy yielding a high z-score, and another successful case with low entropy resulting in a negative z-score. From the two cases, we find that RIW applies subtle changes to the output of the LLMs, achieving a lower perplexity for the watermarked output compared to the non-watermarked version. Also, the success in the second example represents that with our design of negative green list, RIW can detect the watermark in low entropy sequences while maintaining the minimal difference in perplexity between watermarked and non-watermarked outputs. This underscores the effectiveness of the RIW model in maintaining text quality while embedding detectable watermarks, even in cases where the sequence is low-entropy.