# WATERMARK SMOOTHING ATTACKS AGAINST LANGUAGE MODELS

**Hongyan Chang**
National University of Singapore
hongyan@comp.nus.edu

**Hamed Hassani**
University of Pennsylvania
hassani@seas.upenn.edu

**Reza Shokri**
National University of Singapore
reze@comp.nus.edu

## ABSTRACT

Watermarking is a key technique for detecting AI-generated text. In this work, we study its vulnerabilities and introduce the *Smoothing Attack*, a novel watermark removal method. By leveraging the relationship between the model's confidence and watermark detectability, our attack selectively smoothes the watermarked content, erasing watermark traces while preserving text quality. We validate our attack on open-source models ranging from 1.3B to 30B parameters on 10 different watermarks, demonstrating its effectiveness. Our findings expose critical weaknesses in existing watermarking schemes and highlight the need for stronger defenses.

## 1 INTRODUCTION

Detecting whether a text is generated by language models is critical in domains like fraud detection, fake news identification, and plagiarism prevention. A common approach is watermarking, where subtle patterns are embedded in the generated text for later detection (Aaronson, 2023; Christ et al., 2023; Huang et al., 2023; Li et al., 2024). Watermarking has gained traction in both academia and industry (Dathathri et al., 2024) as a key safeguard for language model applications. While various watermarking techniques exist, they share a core principle: favoring certain tokens over others (detailed in Section 2).

In this work, we identify key scenarios where watermarks fail and introduce a novel watermark removal attack that exploits this weakness, revealing fundamental limitations in existing watermarking schemes.

**Effectiveness of watermarks.** We say a watermark is effective if **(i)** the watermarked text maintains high quality, comparable to those generated from the corresponding un-watermarked model, and **(ii)** the detector reliably identifies watermark traces, i.e., it can identify watermarked text without making a large error. We analytically and empirically show that these aspects are in tension: better text quality often implies lower watermark detectability, and vice versa. Moreover, both are connected through the model's confidence in generating output. We explain the high-level idea as follows (see more detail in Section 3).

Given a prefix, when the model is confident about the output token, watermarking has negligible impact on the output. In this case, the watermark trace is not obvious. Conversely, when the model is not confident, watermarking makes the model tend to select certain tokens (that are originally unlikely to get sampled) over others, making watermark trace more detectable while degrading the text quality.

**Smoothing Attack.** Leveraging this insight, we propose the *Smoothing Attack* for watermark removal. For each prefix, the attack first identifies if the output token contains the watermark trace, by estimating the target watermarked model's confidence in this output. If the confidence is low, then we replace the token with a freshly sampled one (see more detail in Section 4), removing watermark traces while maintaining text quality; otherwise, if the confidence is high, then we retain the watermarked model's output.

We evaluate our attack across ten diverse watermarking schemes and three different families of open-sourced models, OPT (Zhang et al., 2022) (from 1.3B to 30B parameters), Llama3-8B (Dubey et al., 2024) and Qwen2-1.5B (Chu et al., 2024). In certain cases, our attack completely removes the

watermark (reducing watermark detection rates to zero) while preserving the text quality. Our attack can also outperform the state-of-the-art *Paraphrasing Attack*, which uses the strong GPT-3.5-turbo to paraphrase the watermarked text. Compared with *Paraphrasing Attack*, our attack is more cost-efficient, as it uses only much weaker reference models, e.g., OPT-125M (Zhang et al., 2022) when attacking OPT models from 1.3B to 30B parameters. These findings underscore critical weaknesses in existing watermarks and highlight the need for more robust defenses.

## 2 Preliminaries and Related Work

For an auto-regressive language model (LM) $M$, we use $\mathcal{V}$ to denote its vocabulary (i.e., the set of all possible tokens). On a given prompt, $M$ generates its output as follows. At each token position $t$ with the given prefix (including the previously generated tokens and the prompt), model $M$ first computes the logit for each token $v$, written as $l_t(v)$. Applying the soft-max function to the logits, we obtain the following probability distribution for the output token.

$$P_t(v) = \frac{\exp\big(l_t(v)\big)}{\sum_{v' \in \mathcal{V}} \exp\big(l_t(v')\big)}. \tag{1}$$

With $P_t$, two sampling strategies are often employed to sample the next token. Top-k sampling (Fan et al., 2018; Holtzman et al., 2018) samples a token from the $k$ most probable candidates. Top-p/Nucleus sampling (Holtzman et al., 2019) samples a token from the smallest set of tokens whose cumulative sum of probability masses exceeds some constant $p$. We denote the output text, i.e., a sequence of tokens, as $(v_1, \ldots, v_T)$ for some positive $T$.

At a high level, watermarks are embedded into the generated text $(v_1, \ldots, v_T)$ through specific patterns of tokens. The detector tries to find traces of such patterns, by computing some detection score function $d(v_1, \ldots, v_T)$. If the score exceeds a certain threshold $\tau$, then the text is predicted as generated from the watermarked model. Next, we explain representative watermarking algorithms.

**Green-red list watermark (Kirchenbauer et al., 2023a).** The idea is to modify the logits of specific tokens. In particular, at each token position $t$, the vocabulary set $V$ is partitioned to the red and green lists, where the green list, denoted as $\mathcal{G}_t$, takes a $\gamma$ fraction of the vocabulary. Logits of tokens in $\mathcal{G}_t$ are increased by some pre-fixed constant $\delta$; while the logits of other tokens remain unchanged. The modified probability distribution is written as

$$\widetilde{P}_t(v) \;=\; \frac{\exp\big(l_t(v) + \delta \cdot \mathbf{1}\{v \in \mathcal{G}_t\}\big)}{\sum_{v' \in \mathcal{V}} \exp\Big(l_t(v') + \delta \cdot \mathbf{1}\{v' \in \mathcal{G}_t\}\Big)}. \tag{2}$$

A sampling strategy, either top-k or top-p, is then applied to $\widetilde{P}_t$ to output the next token. The detector looks for evidence that the green tokens appear disproportionally more frequently. Accordingly, given $(v_1, ..., v_T)$ and the green lists $\{\mathcal{G}_t\}_{t=1}^T$ , the detector computes

$$d(v_1, \ldots, v_T) = \frac{\sum_{t=1}^T \big(\mathbf{1}\{v_t \in \mathcal{G}_t\} - \gamma\big)}{\sqrt{T\gamma(1-\gamma)}}, \tag{3}$$

and then predicts the given text as watermarked if this score exceeds some threshold $\tau$. Here $\gamma$ stands for the expected number of green tokens per token position generated by any other models, since the *assignment of the green list* $\mathcal{G}_t$ is random and is known to the LM provider and the detector only. The denominator $\sqrt{T\gamma(1-\gamma)}$ normalizes the detection score: it is unlikely that a non-watermarked text will be misclassified as watermarked, particularly when the text is long enough (i.e., $T$ is large).

**Gumbel sampling watermark (Kuditipudi et al., 2023; Aaronson, 2023).** The idea is to use Gumbel sampling (Gumbel, 1954) when sample the token at each position $t$. In particular, they first sample $u_t(v)$ from $[0, 1]$ for each $v$, based on some random seed computed from the preceding $k$ tokens ($k$ is some hyperparameter) and some *secret watermarking key*. The output token $v_t^*$ is then selected based on the sampled outcomes and the original $P_t(v)$, given as $v_t^* = \arg\max_{v \in \mathcal{V}} -\frac{\log u_t(v)}{P_t(v)}$. The watermarked text tends to contain token $v$ that is associated with a larger $u_t(v)$. The detection score is computed as $d(v_1, \ldots, v_T) = -\sum_{t=1}^T \log(1 - u_t(v_t))$. If this score exceeds some threshold, the text is predicted as watermarked.

**Tournament sampling watermark (Dathathri et al., 2024).** When generating the $t$-th token, $m$ random watermarking functions $g^{(1)}..., g^{(m)}$ are used to assign $m$ scores for each token in the vocabulary. The scores depend on the given token $v$ and a random seed $r_t$ computed from the secret watermarking key and the recent context (e.g., preceding $k$ tokens), and are denoted as $g^{(l)}(v, r_t) \in \{0, 1\}$. With the scores, the next token is selected as follows. First, $2^m$ tokens are sampled with replacement from the original probability distribution $P_t$. These sampled $2^m$ tokens are then split into $2^{m-1}$ pairs of competing tokens and the tokens with larger scores win (breaking ties randomly). This process is repeated for $m$ times and the final winner is the output token. As this process favors tokens with larger tournament scores, the detector computes $d(v_1, \ldots, v_T) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{m}\sum_{l=1}^{m} g^{(l)}(v_t, r_t)$, and predicts the given text as watermarked if this score exceeds $0.5$ by a large margin. Dathathri et al. (2024) also provide an equivalent way to sample the token, using some modified proability distribution $\widetilde{P}_t$ computed from the $m$ watermarking functions and the original $P_t$ (detailed in Appendix A).

**Other related work.** The Green-red list watermark (Kirchenbauer et al., 2023a) favors tokens in the green lists and introduces distortions to the distribution of the output tokens. Therefore, we often say it is a *distortionary* watermark. Variations of this scheme mainly differ in the assignment of green lists and the detection process, e.g., see Kirchenbauer et al. (2023b); Lee et al. (2023); Liu et al. (2023); Wu et al.. Gumbel and Tournament sampling watermarks (Kuditipudi et al., 2023; Aaronson, 2023; Dathathri et al., 2024) are distortion-free, preserving the original model's token distribution (when averaging over all possible secret watermarking keys). We refer readers to the original papers for detailed analyses. There are also other distortion-free watermarks, e.g., see Hu et al.; Christ et al. (2023), and we refer to Zhao et al. (2024a) for an in-depth survey. In our evaluation, we evaluate $10$ representative watermarks, spanning distortionary and distortion-free approaches, to demonstrate the universal applicability of our attack.

The canonical way to remove watermark is by disrupting its patterns, via injecting special characters, homoglyphs, or emojis into the text (e.g., see Pajola & Conti (2021); Boucher et al. (2022); Goodside (2023)). However, such modifications often reduce the text quality significantly. Another strategy is to paraphrase the watermarked text using another model, referred to as *Paraphrasing Attacks* (e.g., see Kirchenbauer et al. (2023b); Krishna et al. (2023); Piet et al. (2023)). The performance of such an attack typically relies on the power of the model it uses. A large LLM often leads to good quality in the paraphrased texts, e.g., using the GPT-3.5-turbo (OpenAI, 2023) to paraphrase the watermarked texts generated from the 7B-parameter Llama (Touvron et al., 2023).

## 3 On the Effectiveness of Watermarks

In this section, we investigate the key factors that contribute to the effectiveness of watermarking algorithms, namely watermark detectability and text quality. Our key finding is that these two factors are in tension: better text quality often implies lower watermark detectability, and vice versa. Moreover, both are tightly connected to the model's confidence in generating output, which reveals a vulnerability exploitable by the attack we propose later. The detailed derivations are in Appendix C.

### 3.1 From watermark detection to model's confidence

Recall that the watermark detectability, which is characterized by the detection score (e.g., see Eq. equation 3), depends on all tokens in the given text. Our key finding is that the individual contributions to the detection score from tokens at different positions depend on the model prediction confidence. Thus, to estimate this token-level contribution to watermark detectability, it suffices to estimate the model's confidence. In what follows, we go through the reasoning using the Green-red list watermark as an illustration. We then show that our findings generalize to the other two representative watermarks, Tournament sampling and Gumbel sampling.

**Token-level contribution to watermark detection.** Recall Eq. equation 3, the detection score (roughly) counts the number of tokens that belong to the green list in all token positions. For each token position $t$ and the assigned green list $\mathcal{G}_t$, we define

$$S_t = \mathbb{E}_{v\sim\widetilde{P}_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right] - \mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right], \tag{4}$$

where $P_t$ and $\widetilde{P}_t$ stand for the original probability distribution and the modified one (all logits of green tokens are increased by some $\delta$), respectively.

Comparing Eq. equation 4 with the detection score in Eq. equation 3, we have omitted the normalization factor $\sqrt{T\gamma(1-\gamma)}$ for brevity. We also focused on one particular position $t$ and the corresponding assignment of $\mathcal{G}_t$. If we take the expectation over the all possible assignments of $\mathcal{G}_t$, then the subtracted term $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ becomes $\gamma$, since $\mathcal{G}_t$ takes a $\gamma$ fraction of tokens from the vocabulary.

$S_t$ captures the increment of the token-level contribution to the overall detection score due to watermarking, as the first expectation is taken over $\widetilde{P}_t$ rather than $P_t$. Overall, larger $S_t$ leads to higher watermark detectability, and vice versa.

**Connecting $S_t$ to model's confidence.** Recall that $\delta$ represents the shift applied to the logits of tokens in the green list. Then we can write $S_t$ as

$$S_t = \frac{-(e^\delta - 1)\cdot \mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right] + (e^\delta - 1)}{1/\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right] + (e^\delta - 1)}, \tag{5}$$

which is a function of $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$. Clearly, $S_t$ can be different at different token positions. To validate Eq equation 5, we draw 400 sample points on $S_t$ and $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ using the OPT-1.3B model with $\gamma = 0.5$ and $\delta = 1.0$ and plot them in Figure 1 (the left-most subfigure). The empirical observations align with our analysis.

Further, we show that $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ is correlated with the squared $\mathcal{L}_2$ norm of the probability vector $P_t$ at position $t$, denoted as $\|P_t\|^2$. In particular, the mean of $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ is $\gamma$ and the variance is $\gamma(1-\gamma)\|P_t\|^2$. When $\|P_t\|^2 = 1$, $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ is either 0 or 1. Conversely, when $\|P_t\|^2 = 1/|\mathcal{V}|$, $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ is concentrated around $\gamma$, a value smaller than 1. We demonstrate this correlation in Figure 1 (the middle subfigure), using the same 400 sample points obtained above.

The value of $\|P_t\|^2$ measures how confident the model is when outputting the next token at position $t$. In particular, $\|P_t\|^2$ attains its maximum value 1, i.e., all probability masses are concentrated on a single token when the model is absolutely certain of its output; and $\|P_t\|^2$ attains its minimum value $1/|\mathcal{V}|$, i.e., the probability masses are evenly distributed over all tokens, when the model has no idea which token to output.

Putting everything together, we are able to connect $S_t$ to the model's confidence, which is characterized by $\|P_t\|^2$. We demonstrate this correlation in Figure 1 (the right subfigure). We observe that when $\|P_t\|^2$ is large (in particular, consider $\|P_t\|^2 = 1$), then $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ is close to 0 or 1, leading to a relatively small value of $S_t$. Conversely, when $\|P_t\|^2$ is small (in particular, consider $\|P_t\|^2 = \frac{1}{|\mathcal{V}|}$), then $\mathbb{E}_{v\sim P_t}\left[\mathbf{1}\{v \in \mathcal{G}_t\}\right]$ is roughly $\gamma$, a value smaller than 1, leading to a relatively large value of $S_t$.

In summary, *when the model is more confident in choosing the output token at position t, then its contribution $S_t$ to the watermark detectability is smaller, and vice versa.*

**Generalization to other watermarking solutions.** For Gumbel sampling, we define the token-level contribution to watermark detection as $S_t = -\log(1-U_{v^*}) - \mathbb{E}_{v\sim P_t}[-\log(1-U_v)]$, where $v^*$ is the token selected by the watermarked model. Note that the choice of $v^*$ is deterministic after the secret key held by the LM provider and the prefix content are fixed. For Tournament sampling, we define the token-level contribution as $S_t = \mathbb{E}_{v\sim \widetilde{P}_t}\left[\frac{1}{m}\sum_{l=1}^m g^{(l)}(v,r)\right] - \mathbb{E}_{v\sim P_t}\left[\frac{1}{m}\sum_{l=1}^m g^{(l)}(v,r)\right]$, where $\widetilde{P}_t$ is the modified probability distribution. For these two watermarks, we still observe the same correlation between $S_t$ and $\|P_t\|^2$ as we have for Green-list watermarks, as shown in Figure 2. Namely, the token-level contribution $S_t$ to the watermark detectability is negatively correlated to the model's confidence at position $t$.

### 3.2 Impact of watermarking on text quality

Next, we show that the impact of watermarking on the text quality also depends on the model's confidence in generating its output token.

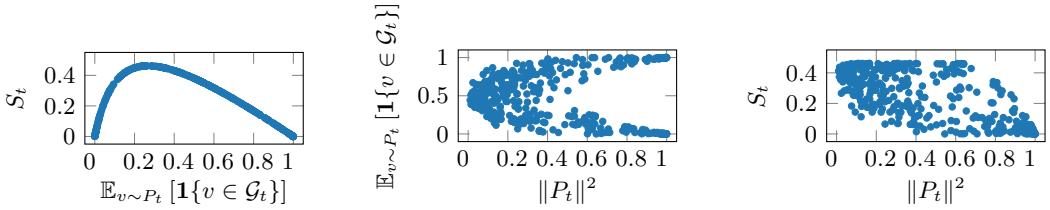


Figure 1: The correlations among $S_t$ (watermark contribution score), $\mathbb{E}_{v\sim P_t}[\mathbf{1}\{v \in \mathcal{G}_t\}]$ (expected number of green tokens from the un-watermarked model), and $\|P_t\|^2$ (model confidence), evaluated on model OPT-1.3B with the Red-green list watermark with $\gamma = 0.5$ and $\delta = 1.0$. The values are computed from different prefixes, constructed from the Wiki page about Harry Potter.

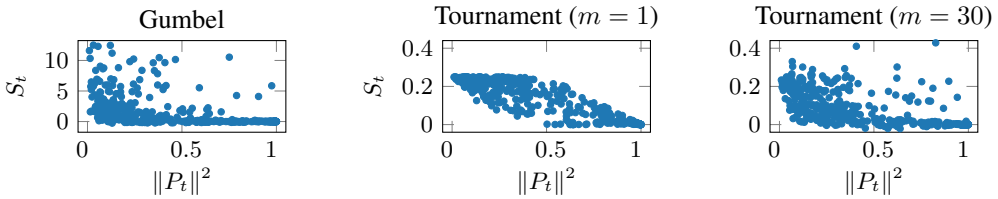


Figure 2: The correlation between $S_t$ (watermark contribution score) and $\|P_t\|^2$ (model confidence) evaluated on model OPT-1.3B with the Gumbel and Tournament sampling (with $m$ tournaments) watermarks, using the same setup as in Figure 1. Each sample corresponds to a specific prefix and secret key. $\|P_t\|^2$ is computed from the original un-watermarked model. The overall observation is similar to what we have for the *Green-red list watermarking*: $S_t$ decreases as $\|P_t\|^2$ increases.

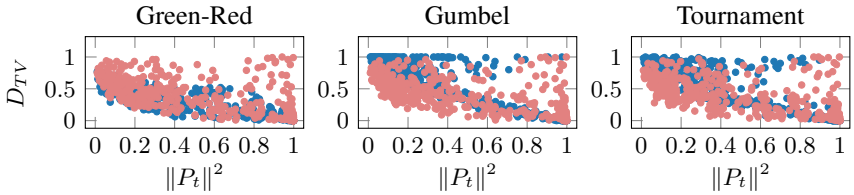


Figure 3: The correlation between $D_{TV}(P_t, \widetilde{P}_t)$, i.e., the negative impact on text quality due to watermarks (in color blue), and $\|P_t\|^2$, measured on OPT-1.3B with the Green-red list and Gumbel and Tournament sampling watermarks. We also plot $D_{TV}(P_t, P_t^{\text{ref}})$, which measures the negative impact on text quality if we use tokens sampled from the reference model OPT-125M (in color red).

**Total variation distance (TVD).** To measure the impact of watermarking on the text quality at any token position $t$, we use the total variation distance between the probability distributions of the original un-watermarked model and the watermarked model, defined as $D_{TV}(P_t, \widetilde{P}_t) = \frac{1}{2}\sum_{v\in\mathcal{V}} |P_t(v) - \widetilde{P}_t(v)|$, where $P_t$ corresponds to the original probability distribution and $\widetilde{P}_t$ is the watermarked one. Due to the large vocabulary size, we measure $d_{TV}(P_t, \widetilde{P}_t)$ empirically, by repeatedly sampling 100 tokens from each distribution and computing the variation between the sampled tokens' frequencies. We repeat this process for different token positions, with the prefixes constructed from the Wikipedia page about Harry Potter, similar to the setup in Section 3.1. A small $D_{TV}(P_t, \widetilde{P}_t)$ indicates that the watermark introduces a small distortion to the original probability distribution at position $t$, suggesting a negligible impact on the output quality.

**TVD depends on model's confidence.** The results for watermarked texts are shown in Figure 3. (in color blue). For all three watermarking schemes, $D_{TV}(P_t, \widetilde{P}_t)$ decreases as $\|P_t\|^2$ increases, meaning that the negative impact on text quality due to watermarks decreases as the model becomes more confident in its output. This is because, when the model is confident in its output (e.g., with probability 1 it will output a certain token), watermarks do not make a notable difference in the model's output distribution. Conversely, when the model is less confident in its output, the negative impact of watermarks on the text quality becomes more notable.

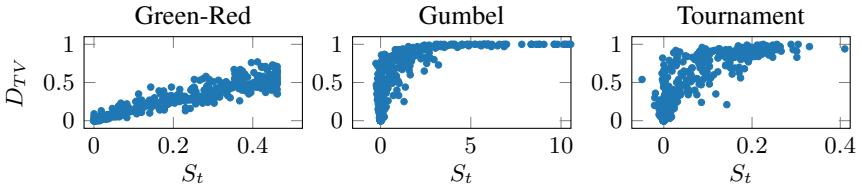


Figure 4: The correlation between $D_{TV}(P_t, \widetilde{P}_t)$, i.e., the negative impact of watermarking on the text quality, and $S_t$, i.e., the token-level contribution to watermark detectability. We measure this on OPT-1.3B. For all three watermarking schemes, $D_{TV}(P_t, \widetilde{P}_t)$ increases as $S_t$ increases.

We also plot $D_{TV}(P_t, P_t^{\text{ref}})$, which measures the negative impact on text quality if we alternatively sample from the reference model OPT-125M (in color red). We note that when the model is not confident in its output, i.e., when $\|P_t\|^2$ is small, sampling from the reference model's token distribution, i.e., $P_t^{\text{ref}}$, does not hurt the text quality. In particular, under the Green-red list watermarking scheme, $D_{TV}(P_t, P_t^{\text{ref}})$ is comparable to $D_{TV}(P_t, \widetilde{P}_t)$ when $\|P_t\|^2$ is small (observe that the red points generally overlap with the blue ones). For Gumbel and Tournament sampling, $D_{TV}(P_t, P_t^{\text{ref}})$ is even smaller than $D_{TV}(P_t, \widetilde{P}_t)$ when $\|P_t\|^2$ is small (observe that the red points are generally below the blue ones). Conversely, when the model is confident in its output, i.e., when $\|P_t\|^2$ is large, replacing the watermarked model with a reference model may hurt the text quality (observe that the red points are above the blue ones).

In summary, the impact of watermarks on the text quality depends on the model's confidence. When the model's confidence is high, watermarking has a negligible impact on the text quality. Conversely, when the model's confidence is low, watermarking has a notable negative impact on the text quality. Surprisingly, in this case, we can even replace the watermarked model with a much smaller reference model, achieving comparable and even better text quality.

**The limitations of existing watermarks.** We come to realize that the two aspects of the watermark effectiveness — watermark detectability and text quality — are correlated with the model's confidence in its output. That means these two aspects are also interconnected. In Figure 4, we plot the correlation between $D_{TV}(P_t, \widetilde{P}_t)$ and $S_t$, empirically measured on OPT-1.3B model using the same setup as the above simulations. When the watermark has little impact on text quality (i.e., smaller total variation distance), the watermark is also less detectable (i.e., smaller $S_t$). Conversely, tokens that contribute more to watermark detection also lead to more notable text quality degradation. This finding, in turn, reveals the crucial limitation of existing watermarking schemes: high watermark detectability and high text quality cannot be achieved at the same time, since the very same set of tokens causes quality degradation while contributing to watermark detectability simultaneously.

## 4 Smoothing Attack

Our objective is to design a watermark removal attack such that for any given input prompt, the algorithm returns some output text that **(i)** is of high-quality, i.e., comparable the original un-watermarked model, while **(ii)** remaining free of watermarks, i.e., the generated text should evade watermark detection mechanisms. Based on our findings in Section 3, we design the *Smoothing Attack*. Our attack proceeds as follows at each token position: **(i)** we first determine if the model is confident in its output token (recall Section 3.1), we do this by estimating $\|P_t\|^2$), and then **(ii)** we replace the token therein by the token sampled from a reference model (if the watermark is distortionary) or by the token re-sampled from the watermarked model (if the watermark is distortion-free). Next, we go through our attack. Mathematical details are in Appendix C.3 and C.4.

**Model Access.** We consider a practical scenario where the adversary has limited access to the targeted watermarked model and does not have access to the original un-watermarked model. Hence, he cannot estimate $\|P_t\|^2$ directly. We assume that at each token position, the adversary knows the top-$K$ tokens that are most likely to be sampled and the corresponding probabilities (with $K$ being a small constant, e.g., $1, 5, 10$) via the target watermarked model's API. This level of information is commonly accessible, even for closed-source models, e.g., OpenAI's API provides information on the most likely tokens and the corresponding probabilities.

**Estimation of model's confidence.** For distortion-free watermarks (e.g., Gumbel and Tournament sampling), we can directly observe the unchanged top-$K$ probabilities of the un-watermarked model. We then estimate $\|P_t\|^2$ from the probabilities associated with the top-$K$ most probable tokens in $\mathcal{V}_{\text{Top-K}}$, expressed as $\|P_t\|^2 \approx \sum_{v\in\mathcal{V}_{\text{Top-K}}} P(v)^2 + \frac{1}{|\mathcal{V}|-K} \cdot \left(1 - \sum_{v\in\mathcal{V}_{\text{Top-K}}} P(v)\right)^2$, where we have assumed that the residual probability mass $1 - \sum_{v\in\mathcal{V}_{\text{Top-K}}} P(v)$ is uniformly distributed across the remaining $|\mathcal{V}| - K$ tokens whose probabilities are unobserved. For distortionary watermarks (e.g., the Green-red list), we estimate $\|P_t\|^2$ from the top-$K$ probabilities observed in the modified $\widetilde{P}_t$. In such cases, we first compute an estimation for the watermarked model's $\|\widetilde{P}_t\|^2 \approx \sum_{v\in\mathcal{V}_{\text{Top-K}}} P(v)^2 + \frac{1}{|\mathcal{V}|-K} \cdot \left(1 - \sum_{v\in\mathcal{V}_{\text{Top-K}}} \widetilde{P}(v)\right)^2$. Again, we have assumed that the residual probability mass $1 - \sum_{v\in\mathcal{V}_{\text{Top-K}}} \widetilde{P}(v)$ is evenly distributed to the unobserved tokens. Next, we transform this estimation to the original unwatermarked model by computing $\|P_t\|^2 \approx \beta \cdot \|\widetilde{P}_t\|^2$ with $\beta = \frac{\left((1-\gamma)+\gamma e^{\delta}\right)^2}{(1-\gamma)+\gamma e^{2\delta}}$.

**Normalization of confidence.** We convert the above estimated $\|P_t\|^2$ into a confidence score in $[0, 1]$, denoted as $c$. To establish the upper and lower bounds for normalization, denoted as $U$ and $L$, we obtain $N$ (e.g., $N = 200$) random samples for $\|P_t\|^2$, computed on the watermarked model with random prefixes. We then set $U$ and $L$ as the largest and smallest values among the samples, respectively. Next, we construct 100 bins from the range $[L, U]$ and then map the estimated $\|P_t\|^2$ into the $i$-th bin with $i \in [1, 100]$. The confidence score $c$ is then computed as $c = \frac{i}{100}$. Based on $c$, we decide whether to adopt the token output by the watermarked model at this token position.

**Smoothing the watermark.** When the watermark is distortion-free, with probability $c^\alpha$, we adopt this token; with probability $(1 - c^\alpha)$, we randomly choose another token by re-sampling the token based on the observed top$-K$ probabilities of the target watermarked model, and then put it into the token position. When the watermark is distortionary, we first query a reference model (we use a reference model is much smaller than the watermarked model) using the same prefix and then obtain the top-$K$ probabilities from the reference model. After that, we compute a weighted probability distribution from the top-$K$ probabilities of the watermarked model and the reference model. For the reference model, the weight is assigned to $1 - c^\alpha$; for the watermarked model, the weight is assigned to $c^\alpha$. We then sample from the weighted distribution and adopt the output token. Here we have used a constant exponential factor $\alpha > 0$ to control the smoothness-level of our attack. In particular, when $\alpha$ is large, our attack inclines to adopt the token from the watermarked model. On the other hand, when $\alpha$ is small, our attack inclines to replace the output token.

**Applicability.** Our attack does not require detailed knowledge of the watermarking strategy. Instead, it either samples a new token or adopts the output token from the watermarked model, based on the confidence score $c$. In addition, it is computationally efficient, and the main overhead incurred when estimating the upper and lower bounds for $c$, i.e., $U$ and $L$. This can be done by querying the watermarked model a few hundred times.

## 5 EXPERIMENTS

**Evaluation setup.** We attack the open-sourced models including, Llama3.1-8B (Dubey et al., 2024), the OPT model family (Zhang et al., 2022) (from 1.3B to 30B parameters), and Qwen2-1.5B (Chu et al., 2024). When attacking distortionary watermarking algorithms on Llama3, OPT models, and Qwen2, we use Llama3-1B (Dubey et al., 2024), OPT-125M (Zhang et al., 2022), and Qwen2-0.5B (Chu et al., 2024) as the reference models, respectively. Following prior work (Kirchenbauer et al., 2023a; Pan et al., 2024), we evaluate on the C4 dataset (Raffel et al., 2020). For each text, its first 30 tokens of texts serve as the prompt, and the task is to generate the subsequent 200 tokens. The results are averaged over 100 prompts. Our experiments are run on RTX-Titan GPUs.

We evaluate against 10 representative watermarking algorithms, covering distortionary and distortion-free watermarks, including KGW (Kirchenbauer et al., 2023a), Unigram (Zhao et al., 2023), UPV (Liu et al., 2023), X-SIR (He et al., 2024), DIP (Wu et al.), SWEET (Lee et al., 2023), EWD (Lu et al., 2024) Unbiased (Hu et al.), SynthID (Dathathri et al., 2024) (which leverages Tournament sampling), and Gumbel (Aaronson, 2023). The implementations are based on the MarkLLM toolkit (Pan et al., 2024). For Gumbel and X-SIR, we evaluate it on OPT models (Gumbel requires

Table 1: Performance of watermark removal attacks on OPT-1.3B, Llama3-8B, and Qwen-1.5B. The false positive rate on the unwatermarked text is less than 1%. We show the true positive rate in % (TPR), perplexity (PPL), and diversity (Div) for each watermarking algorithm for different models and against different attacks.

| Watermark | Attack | OPT-1.3B | | | Llama3-8B | | | Qwen2-1.5B | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| Un-watermarked | - | 1 | 11.39 | 8.22 | 1 | 3.47 | 6.82 | 1 | 12.26 | 8.10 |
| Reference | - | 1 | 19.57 | 7.69 | 1 | 4.40 | 6.52 | 1 | 16.02 | 8.06 |
| KGW (Kirchenbauer et al., 2023a) | None | 100 | 14.61 | 8.07 | 99 | 4.60 | 6.92 | 100 | 16.46 | 8.11 |
| | Paraphrasing | 3 | 14.82 | 9.56 | 2 | 5.35 | 8.0 | 2 | 10.45 | 9.42 |
| | Smoothing | 0 | 9.57 | 6.72 | 2 | 3.20 | 5.63 | 0 | 8.02 | 6.91 |
| Unigram (Zhao et al., 2023) | None | 100 | 14.99 | 7.29 | 99 | 4.61 | 6.56 | 100 | 15.41 | 7.37 |
| | Paraphrasing | 53 | 14.51 | 8.75 | 54 | 5.60 | 8.02 | 5 | 10.40 | 8.56 |
| | Smoothing | 5 | 9.44 | 6.73 | 24 | 3.10 | 5.44 | 1 | 7.77 | 6.71 |
| SynthID (Dathathri et al., 2024) | None | 100 | 7.12 | 7.41 | 99 | 4.83 | 7.31 | 100 | 6.94 | 7.05 |
| | Paraphrasing | 1 | 10.57 | 9.11 | 1 | 5.62 | 8.18 | 1 | 6.90 | 8.43 |
| | Smoothing | 0 | 10.40 | 8.64 | 0 | 3.40 | 6.86 | 0 | 10.21 | 8.04 |
| DIP (Wu et al.) | None | 100 | 13.73 | 8.44 | 84 | 4.03 | 7.35 | 100 | 14.34 | 8.27 |
| | Paraphrasing | 0 | 13.95 | 9.25 | 0 | 5.25 | 8.34 | 2 | 10.10 | 8.85 |
| | Smoothing | 6 | 9.34 | 6.84 | 6 | 3.17 | 5.67 | 11 | 7.62 | 6.92 |
| Unbiased (Hu et al.) | None | 100 | 13.61 | 8.29 | 84 | 4.02 | 7.29 | 100 | 14.64 | 8.21 |
| | Paraphrasing | 3 | 14.45 | 10.39 | 2 | 5.36 | 8.57 | 1 | 9.97 | 8.82 |
| | Smoothing | 27 | 9.19 | 6.84 | 5 | 3.17 | 5.75 | 5 | 7.68 | 6.94 |
| UPV (Liu et al., 2023) | None | 99 | 11.65 | 8.22 | 83 | 4.38 | 6.80 | 86 | 11.93 | 7.49 |
| | Paraphrasing | 34 | 13.73 | 9.92 | 2 | 5.43 | 8.00 | 2 | 9.03 | 8.58 |
| | Smoothing | 20 | 10.01 | 6.89 | 1 | 3.12 | 5.49 | 0 | 8.16 | 6.91 |
| EWD (Lu et al., 2024) | None | 100 | 15.23 | 7.92 | 100 | 4.56 | 6.71 | 100 | 16.31 | 7.85 |
| | Paraphrasing | 0 | 14.95 | 9.95 | 7 | 5.73 | 7.83 | 1 | 10.18 | 9.28 |
| | Smoothing | 0 | 9.93 | 6.78 | 3 | 3.13 | 5.38 | 0 | 7.82 | 6.85 |
| SWEET (Lee et al., 2023) | None | 100 | 14.36 | 8.02 | 99 | 4.53 | 6.69 | 100 | 15.89 | 7.65 |
| | Paraphrasing | 0 | 14.57 | 9.45 | 14 | 5.64 | 8.05 | 4 | 10.18 | 9.30 |
| | Smoothing | 0 | 9.59 | 6.72 | 4 | 3.09 | 5.40 | 0 | 7.85 | 6.92 |

more than 100 GB of GPU memory when running on Qwen and Llama due to their large vocabulary sizes and X-SIR's official code does not support Llama and Qwen models for now).

We use strongest watermark removal attack, *Paraphrasing Attack* that uses GPT-3.5-turbo to paraphrase the watermarked text (Piet et al., 2023), as a competitor to our attack. As a comparison, our *Smoothing Attack* leverages only much smaller reference models (when attacking distortionary watermarks).

We evaluate the performance of attacks in terms of watermark removal and text quality preservation. For watermark removal, we report the true positive rate of watermark detection, i.e., TPR, (lower means the attack is better) when the false positive rate, i.e., FPR, is less than 1%. In this case, TPR is $1\%$ for un-watermarked models and $100\%$ without attacks. For text quality, we follow prior work (Kirchenbauer et al., 2023a; Pan et al., 2024) to measure the perplexity (lower means better text quality).

For our attack, we set $\alpha$ to $1.0$ and use the top-$10$ most likely tokens and their probabilities from the watermarked model and reference model by default, unless specified otherwise. We also report the diversity of the generated text (higher means better), following Kirchenbauer et al. (2023b). More detailed setup descriptions are in the appendix.

**Performance in watermark removal.** We present the main results in Tables 1 and 2. Our *Smoothing Attack* successfully removes watermarks for most of the cases (achieving a low TPR around $5\%$ and even $0$ sometimes) across all watermarked models and watermarking algorithms. Our attack also outperforms the strong paraphrasing attack in terms of TPR. Notably, for the OPT-1.3B model with the Unigram watermark (see Table 1), the detector can successfully detect a $53\%$ fraction of the watermarked text even after paraphrasing, while it only identifies a $5\%$ fraction if using our attack. We note that our attack achieves this by using a much smaller reference model, OPT-125M.

**Performance in text quality.** Our attack also preserves the text quality, meaning that it achieves a low perplexity without decreasing the diversity too much. For example, when attacking the Unigram watermark on OPT-1.3B (see Table 1), our attack achieves a perplexity of $9.44$ (much better than $14.51$ of paraphrasing attack) and a diversity of $6.73$ (only slightly worse than $8.75$ of paraphrasing

Table 2: Performance of watermark removal attacks on OPT-1.3B with Gumbel (Aaronson, 2023) and X-SIR (He et al., 2024) watermarks (with FPR $< 1\%$).

| Watermark | Attack | TPR (%) | PPL | Div |
|---|---|---|---|---|
| Gumbel | None | 98 | 2.96 | 4.35 |
| | Paraphrasing | 13 | 14.21 | 11.13 |
| | Smoothing | 9 | 19.25 | 8.30 |
| X-SIR | None | 94 | 13.99 | 7.96 |
| | Paraphrasing | 34 | 14.13 | 8.80 |
| | Smoothing | 9 | 9.47 | 6.75 |

Table 3: Impact of $K$ and $\alpha$ on *Smoothing Attack* performance on OPT-1.3B with Unigram watermark (with FPR $< 1\%$).

| $K$ | $\alpha$ | TPR (%) | PPL | Div |
|---|---|---|---|---|
| Fixed to 10 | 0.5 | 42 | 9.9 | 6.86 |
| | 1.0 | 5 | 9.44 | 6.73 |
| | 2.0 | 0 | 9.38 | 6.58 |
| | 3.0 | 1 | 9.25 | 6.43 |
| 1 | Fixed to 1 | 18 | 3.21 | 4.62 |
| 5 | | 10 | 7.46 | 6.11 |
| 10 | | 5 | 9.44 | 6.73 |
| 15 | | 5 | 11.73 | 7.11 |

attack) while achieving a TPR of $5\%$ (much better than $50\%$ of paraphrasing attack). More detailed boxplots for the text quality metrics are in Figures 6 and 7 of Appendix B.3.

For Gumbel sampling (see Table 2), our attack slightly increases the perplexity. The main reason is that Gumbel sampling tends to output repeated content, which sometimes leads to better perplexity, but lower diversity. We give concrete output text samples in Table 9 in Appendix B.5, to show that our attack generates better texts than the watermarked model. The diversity of the text generated by our attack is slightly worse than those generated from the paraphrasing attack (and also sometimes worse than the watermarked model). The main reason is that our attack selects the next token only from Top-$K$ most likely tokens, while the paraphrasing attack and the watermarked model sample the token from the whole vocabulary (which, in turn, may increase the unpredictability and degrade the text quality). Increasing $K$ can resolve this issue, e.g., see Table 3 for the results on the Unigram watermark (more in appendix). The trade-off is that larger $K$'s require more model access.

Our smoothing attack also achieves a lower (hence, better) PPL compared to the unwatermarked text. This is because, for the unwatermarked text, the token is sampled from the whole vocabulary, which makes the output unpredictable sometimes and even erratic. In our smoothing attack, the text is selected from the top-$K$ most likely tokens from the target watermarked model and reference model, preventing extremely unlikely tokens from being output.

**Ablation studies on $K$, $\alpha$, and model size.** The overall observation is that increasing $K$ leads to texts with worse perplexity, but better diversity and better TPR. We note that even with $K = 1$, our attack is still effective. In Table 3, the TPR is only $18\%$, lower than that of GPT-3.5-turbo, which is $53\%$. Increasing $\alpha$ makes our attack tend to forbid the uncertain tokens in the watermarked text when the model is not confident. Hence, increasing $\alpha$ in general improves the perplexity while reducing the diversity sometimes (e.g., see Table 3). At the same time, the detection rate also decreases, since the watermark traces are more likely to get removed. (Note that this event happens at probability $1 - c^\alpha$, which increases as $\alpha$ increases for $c \in (0, 1]$). By adjusting $\alpha$, the adversary can balance between watermark removal and text diversity. More details are deferred to Appendix B.4. We also conduct studies on the model size within the OPT model family (from 1.3B to 30B parameters). For all target models, we use the same reference model OPT-125M, which is much smaller than the target model. Our finding is that the model size has negligible influence to the performance of our attack. For instance, the TPR of our attack against X-SIR increases from $13\%$ to $16\%$ when the size of the target model grows from 1.3B to 30B. We refer to Appendix B.6 for more details.

## 6 Conclusion

We revealed limitations in existing watermarks for language models and examined their robustness against watermark removal attacks. We introduced *Smoothing Attack*, a novel method that leverages model confidence to selectively remove watermark traces while preserving text quality. Comprehensive evaluations demonstrated that *Smoothing Attack* can completely remove watermarks, outperforming the state-of-the-art attack and highlighting a critical gap in current watermarks, and calling for more robust solutions.

## REFERENCES


Scott Aaronson. Simons institute talk on watermarking of large language models. https://simons.berkeley.edu/talks/scott-aaronson-ut-austin-openai-2023-08-17, 2023.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1987–2004. IEEE, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Deepak Subbiah, Jack Kaplan, Prafulla Dhariwal, A. Neelakantan, Long Ouyang, and Dario Amodei. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 2020. Seminal paper introducing GPT-3, discussing the importance of probabilities in understanding model behavior.

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

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

Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, et al. Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*, 2024.

Aloni Cohen, Alexander Hoover, and Gabe Schoenbach. Watermarking language models for many adaptive users. In *2025 IEEE Symposium on Security and Privacy (SP)*, pp. 84–84. IEEE Computer Society, 2024.

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

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

European Commission. The EU Artificial Intelligence Act. Available at: https://artificialintelligenceact.eu/, 2021. Proposed regulation focusing on transparency and accountability in high-risk AI systems.

Jaiden Fairoze, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Mingyuan Wang. Publicly detectable watermarking for language models. *Cryptology ePrint Archive*, 2023.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 889–898, 2018.

Jiayi Fu, Xuandong Zhao, Ruihan Yang, Yuansen Zhang, Jiangjie Chen, and Yanghua Xiao. Gumbelsoft: Diversified language model watermarking via the gumbelmax-trick. *arXiv preprint arXiv:2402.12948*, 2024.

Surendra Ghentiyala and Venkatesan Guruswami. New constructions of pseudorandom codes. *Cryptology ePrint Archive*, 2024.

Noah Golowich and Ankur Moitra. Edit distance robust watermarks via indexing pseudorandom codes. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Riley Goodside. There are adversarial attacks for that proposal as well — in particular, generating with emojis after words and then removing them before submitting defeats it. Twitter, January 2023. URL: https://twitter.com/goodside/status/1610682909647671306.

Emil Julius Gumbel. Statistical theory of extreme valuse and some practical applications. *Nat. Bur. Standards Appl. Math. Ser. 33*, 1954.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4115–4129, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.226. URL https://aclanthology.org/2024.acl-long.226.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1638–1649, 2018.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.

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

Baihe Huang, Banghua Zhu, Hanlin Zhu, Jason D Lee, Jiantao Jiao, and Michael I Jordan. Towards optimal statistical watermarking. *arXiv preprint arXiv:2312.07930*, 2023.

Mingjia Huo, Sai Ashish Somayajula, Youwei Liang, Ruisi Zhang, Farinaz Koushanfar, and Pengtao Xie. Token-specific watermarking with enhanced detectability and semantic coherence for large language models. In *Forty-first International Conference on Machine Learning*.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17061–17084. PMLR, 23–29 Jul 2023a. URL https://proceedings.mlr.press/v202/kirchenbauer23a.html.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. On the reliability of watermarks for large language models. *arXiv preprint arXiv:2306.04634*, 2023b.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *arXiv preprint arXiv:2303.13408*, 2023.

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

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. Who wrote this code? watermarking for code generation. *arXiv preprint arXiv:2305.15060*, 2023.

Xiang Li, Feng Ruan, Huiyuan Wang, Qi Long, and Weijie J Su. A statistical framework of watermarks for large language models: Pivot, detection efficiency and optimal rules. *arXiv preprint arXiv:2404.01245*, 2024.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. *arXiv preprint arXiv:2210.15097*, 2022.

Aiwei Liu, Leyi Pan, Xuming Hu, Shuang Li, Lijie Wen, Irwin King, and S Yu Philip. An unforgeable publicly verifiable watermark for large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=6p8lpe4MNf.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. An entropy-based text watermarking detection method. *arXiv preprint arXiv:2403.13485*, 2024.

National Institute of Standards and Technology (NIST). Ai risk management framework. Technical report, U.S. Department of Commerce, 2023. Framework to improve AI system trustworthiness and manage risks, emphasizing transparency.

OECD. Oecd ai principles. Available at: https://oecd.ai/en/dashboards/ai-principles/, 2019. Guidelines for ethical, trustworthy AI. Transparency and accountability are key principles.

OpenAI. Openai api documentation. Available at: https://platform.openai.com/docs/, 2023. Developer documentation highlighting the use of top-K sampling, beam search, and probability outputs.

R OpenAI. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5), 2023.

Luca Pajola and Mauro Conti. Fall of giants: How popular text-based mlaas fall against a simple evasion attack. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 198–211. IEEE, 2021.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, et al. Markllm: An open-source toolkit for llm watermarking. *arXiv preprint arXiv:2405.10051*, 2024.

Julien Piet, Chawin Sitawarin, Vivian Fang, Norman Mu, and David Wagner. Mark my words: Analyzing and evaluating language model watermarks. *arXiv preprint arXiv:2312.00273*, 2023.

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

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

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

John Wieting, Kevin Gimpel, Graham Neubig, and Taylor Berg-Kirkpatrick. Paraphrastic representations at scale. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 379–388, 2022.

Wikipedia. Harry Potter (film series). URL https://en.wikipedia.org/wiki/Harry_Potter_(film_series).

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. 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. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

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

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

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

Tong Zhou, Xuandong Zhao, Xiaolin Xu, and Shaolei Ren. Bileve: Securing text provenance in large language models against spoofing with bi-level signature. *arXiv preprint arXiv:2406.01946*, 2024.

Table 4: Table of notation definitions and their locations.

| Notation | Meaning | Definition Location |
|---|---|---|
| $M$ | Auto-regressive language model (LM), which generates text sequentially based on a given prompt. | Section 2 |
| $\widetilde{M}$ | Watermarked model, a variant of $M$ that embeds watermarks into generated text. | Section 3.1 |
| $\mathcal{V}$ | Vocabulary of the LM, the set of all possible tokens that can be generated. | Section 2 |
| $t$ | Token position in the generated sequence, indicating the index of a specific token. | Section 2 |
| $l_t(v)$ | Logit assigned by the model to token $v$ at position $t$ before applying softmax. | Eq. equation 1 |
| $P_t(v)$ | Probability of token $v$ at position $t$ after applying the softmax function. | Eq. equation 1 |
| $\widetilde{P}_t(v)$ | Modified probability distribution in the watermarked model after logit manipulation. | Eq. equation 2 |
| $(v_1, \ldots, v_T)$ | Sequence of tokens forming the output text from the language model. | Section 2 |
| $d(v_1, \ldots, v_T)$ | Detection score function used to determine whether a text is watermarked. | Section 2 |
| $\tau$ | Threshold value for watermark detection; if $d(v_1, \ldots, v_T) > \tau$, the text is classified as watermarked. | Section 2 |
| $\mathcal{G}_t$ | Green list, a subset of vocabulary containing tokens whose logits are increased in green-red list watermarking. | Section 2 |
| $\gamma$ | Fraction of the vocabulary included in the green list $\mathcal{G}_t$, determining the probability of token selection. | Section 2 |
| $\delta$ | Logit increase applied to tokens in the green list $\mathcal{G}_t$, influencing token selection probabilities. | Eq. equation 2 |
| $T$ | Length of the generated sequence, i.e., the total number of tokens in the output text. | Section 2 |
| $u_t(v)$ | Randomly sampled value from $[0, 1]$ for token $v$ in Gumbel sampling watermarking. | Section 2 |
| $v_t^*$ | Token selected using Gumbel sampling watermarking by maximizing a transformed probability. | Section 2 |
| $S_t$ | Contribution of the token at position $t$ to the overall watermark detection score. | Eq. equation 4 |
| $\mathbb{E}_{v\sim P_t}[\mathbf{1}\{v \in \mathcal{G}_t\}]$ | Expected probability mass assigned to green tokens at position $t$ from probability distribution $P_t$. | Eq. equation 4 |
| $\|P_t\|^2$ | $\mathcal{L}_2$ norm of the probability vector, measuring model confidence at position $t$. A higher value means greater confidence. | Section 3.1 |
| $D_{TV}(P_t, \widetilde{P}_t)$ | Total variation distance between original and watermarked probability distributions, measuring distortion. | Section 3.1 |
| $D_{TV}(P_t, P_t^{\text{ref}})$ | Total variation distance between the original model and a reference model's probability distributions. | Section 3.1 |
| $K$ | Number of most probable tokens that the adversary has access to from the watermarked model. | Section 4 |
| $\mathcal{V}_{\text{Top-K}}$ | Set of top-$K$ most probable tokens observed by the adversary. | Section 4 |
| $\beta$ | Scaling factor used to estimate $\|P_t\|^2$ from watermarked probabilities in Green-red list watermarking. | Section 4 |
| $c$ | Normalized confidence score in $[0, 1]$ based on estimated $\mathcal{L}_2$ norm. | Section 4 |
| $U, L$ | Upper and lower bounds for normalizing $\mathcal{L}_2$ norms into the confidence score $c$. | Section 4 |
| $\alpha$ | Exponential factor controlling the aggressiveness of the smoothing attack. A larger $\alpha$ favors keeping watermarked tokens, while a smaller $\alpha$ favors replacement. | Section 4 |
| $P_t^{\text{ref}}$ | Token probability distribution from a much smaller, un-watermarked reference model. | Section 4 |

# A More on Related Work

**Variations of Green-red list watermark.** Different variations of Green-red list watermark, e.g., see Kirchenbauer et al. (2023b); Lee et al. (2023); Liu et al. (2023); Wu et al.; Huo et al.; Zhou et al. (2024); Lu et al. (2024); Liu et al. (2024); He et al. (2024); Zhao et al. (2023); Kirchenbauer et al. (2023a), mainly differ in the assignment of the green lists and the detection process. In particular,

the assignment of $\mathcal{G}_t$ could depend on the prefix, e.g., the preceding $h$ tokens in the generated text. When $h = 0$, we say the assignment is context-independent and is referred to as the *Unigram* watermark Zhao et al. (2023); when $h = 1$, the assignment depends on the previous token and is referred to as the *KGW* watermark Kirchenbauer et al. (2023a)

**Scalable Tournament sampling.** As shown in their paper, the original tournament process in Dathathri et al. (2024) can be costly to implement, as there are $O(2^m)$ times of sampling and pairwise comparison of tokens. Instead, they obtain a modified distribution for tokens. With $\widetilde{P}_t^{(0)} = P_t$, they iteratively compute $\widetilde{P}_t^{(l)}(v) = \left(1 + g^{(l)}(v, r_t) - \sum_{v' \in \mathcal{V}} \left(g^{(l)}(v', r_t) \cdot P_t^{(l-1)}(v')\right)\right) \cdot \widetilde{P}_t^{(l-1)}(v)$, for $l = 1, \ldots, m$, and then sample the token from $\widetilde{P}^{(m)}$.

**Distortion-free watermark.** There are also other distortion-free watermarks, which aim to preserve the original model's token distribution and avoid detectable shifts in probabilities of output tokens, e.g., see Hu et al.; Zhao et al. (2024b); Fu et al. (2024); Christ et al. (2023); Fairoze et al. (2023); Christ & Gunn (2024); Cohen et al. (2024); Ghentiyala & Guruswami (2024); Golowich & Moitra; Dathathri et al. (2024); Wu et al..

**Comparison with paraphrasing attacks.** When attacking OPT models (from 1.3B to 30B parameters), our attack only leverages the OPT-125M as the reference model when attacking distortionary watermarks such as the Unigram watermark. When attacking distortion-free watermarks, our attack sometimes resamples from the target watermarked model. In either case, the resource used in our attack is significantly smaller than the state-of-the-art paraphrasing attack, which uses the much larger GPT-3.5-turbo. Despite using fewer resources, our approach achieves higher watermark removal rates and comparable text quality. This highlights that even resource-limited adversaries can thwart watermarks, underscoring the need for stronger watermark defenses.

# B More on Experiments

## B.1 Implementation

We evaluate the smoothing attack on eight different watermarking algorithms, including KGW Kirchenbauer et al. (2023a), Unigram Zhao et al. (2023), SWEET Lee et al. (2023), UPV Liu et al. (2023), EWD Lu et al. (2024), X-SIR He et al. (2024), DIP Wu et al., Unbiased Hu et al., SynthID Dathathri et al. (2024) and Gumbel Aaronson (2023). We use the implementations and default configurations provided by MarkLLM Pan et al. (2024). For completeness, we provide details of the algorithms below.

- KGW Kirchenbauer et al. (2023a): The green set $\mathcal{G}_t$ at each position $t$ is selected based on the previous $h$ tokens and a secret key known to the service provider. The hyperparameters are set as follows: $\gamma = 0.5$, $\delta = 2.0$, and $h = 1$.
- Unigram Kirchenbauer et al. (2023a): The green set $\mathcal{G}_t$ is fixed for each token $t$ and each prefix, depending solely on the secret key known to the service provider. No dynamic updates are performed based on previous tokens. The parameters are: $\gamma = 0.5$, $\delta = 2.0$.
- SWEET Lee et al. (2023): A shift is applied only when the entropy of the probability distribution at position $t$ is high, improving text quality, particularly for code generation tasks. The parameters are set as: $\gamma = 0.5$, $\delta = 2.0$, the entropy threshold is 0.9, and $h = 1.0$.
- UPV Liu et al. (2023): The green token selection process is similar to the previous approaches. However, this method requires training two additional models: a generator network to separate red and green tokens and a detector network for classification based on the input text. The watermarks are introduced using $\gamma = 0.5$, $\delta = 2.0$, and $h = 1.0$. The detector produces a binary prediction rather than a continuous score like a z-score.
- EWD Lu et al. (2024): Watermark introduction follows a similar process as the previous methods. The hyperparameters are $\gamma = 0.5$, $\delta = 2.0$, and $h = 1.0$. During detection, tokens are assigned different weights based on their entropy, with higher entropy tokens receiving greater weight to improve detectability in low-entropy scenarios.

- X-SIR He et al. (2024): Instead of operating at the token level, the red-green partition is applied at the level of semantic clusters, grouping similar words together and adding bias at the group level. This improves robustness against Cross-lingual Watermark Removal Attacks (CWRA).

- DIP Wu et al.: Similar to Kirchenbauer et al. (2023), this method selects green tokens but uses a distribution-preserving reweight function to adjust token probabilities. This increases the probability of green tokens while maintaining the overall distribution. The reweighting is controlled by the parameter $\alpha$. The hyperparameters are set as $\gamma = 0.5$, $h = 5$.

**Implementation of the paraphrasing attack.** We include the strongest baseline that paraphrases the given text based on the GPT-3.5-turbo Piet et al. (2023), denoted as P-GPT3.5 using the prompt: "Please rewrite the following text:". As shown in Kirchenbauer et al. (2023b), GPT-3.5-turbo is more powerful in removing the watermarks compared to Dipper model Krishna et al. (2023).

**Text quality metric.** We use Llama3-8B, Qwen2-7B, and OPT-2.7B to evaluate the perplexity of the text generated from Llama3, Qwen2, and OPT models. We also report the log diversity of the text Welleck et al.; Kirchenbauer et al. (2023b); Li et al. (2022), following the definition in Kirchenbauer et al. (2023b) considering the 2-gram, 3-gram, and 4-gram repetition in the generated text. A higher diversity score represents a more diverse text.

### B.2 Performance of the smoothing attack

Figure 5 shows three scatterplots of TPR vs. PPL for text generated under different watermarking and attack settings. Each point is colored by the watermarking method and corresponds to one of three models (OPT-1.3B, Llama3-8B, Qwen2-1.5B). Overall, the smoothing attack yields substantially lower TPR relative to the watermarked setting, demonstrating its performance at watermark removal. Notably, smoothing's TPR is on par with that of the paraphrasing attack, which uses a more powerful model (GPT-3.5-turbo). In terms of perplexity (PPL), smoothing also generates text that is competitive with (and sometimes lower than) both the watermarked text and the paraphrased text, indicating that it preserves text quality while removing the watermark.

### B.3 Text Quality Evaluation

Figure 6 and Figure 7 present boxplots of the perplexity (PPL) and diversity of text generated from different sources using the OPT-1.3B model. We observe that the smoothing attack generally yields text with lower PPL than the watermarked model, except in cases involving the Gumbel watermark. This suggests that, according to the PPL metric, the smoothing attack can generate high-quality text. In terms of diversity, the constrained selection process—where sampling is restricted to the top-K candidates from both the reference and target models—results in lower diversity for the smoothing attack. These findings are consistent with the average PPL results reported in Table 1 in the main paper.

In addition, we compute the P-SP score Wieting et al. (2022), which quantifies the similarity between pairs of texts in the embedding space, with higher scores indicating greater similarity. Specifically, we calculate P-SP scores for text generated from different sources and visualize the results in the heatmap shown in Figure 8. We observe that, aside from the paraphrasing case, texts from different sources generally exhibit low similarity. For instance, text generated by the watermarked model has a P-SP score of 53.6 on Unigram, whereas the similarity between the watermarked text and its paraphrased version reaches 82.3. Our smoothing attack produces a P-SP score (measuring similarity between text from the smoothing attack and unwatermarked text) comparable to that of the watermarked text (measuring similarity between watermarked text and unwatermarked text). The generally low P-SP scores across different sources reflect the natural variability in generated responses, as multiple reasonable outputs can exist for the same prompt. Therefore, P-SP metrics may not be a reliable measure for assessing text quality degradation due to watermarking or smoothing.

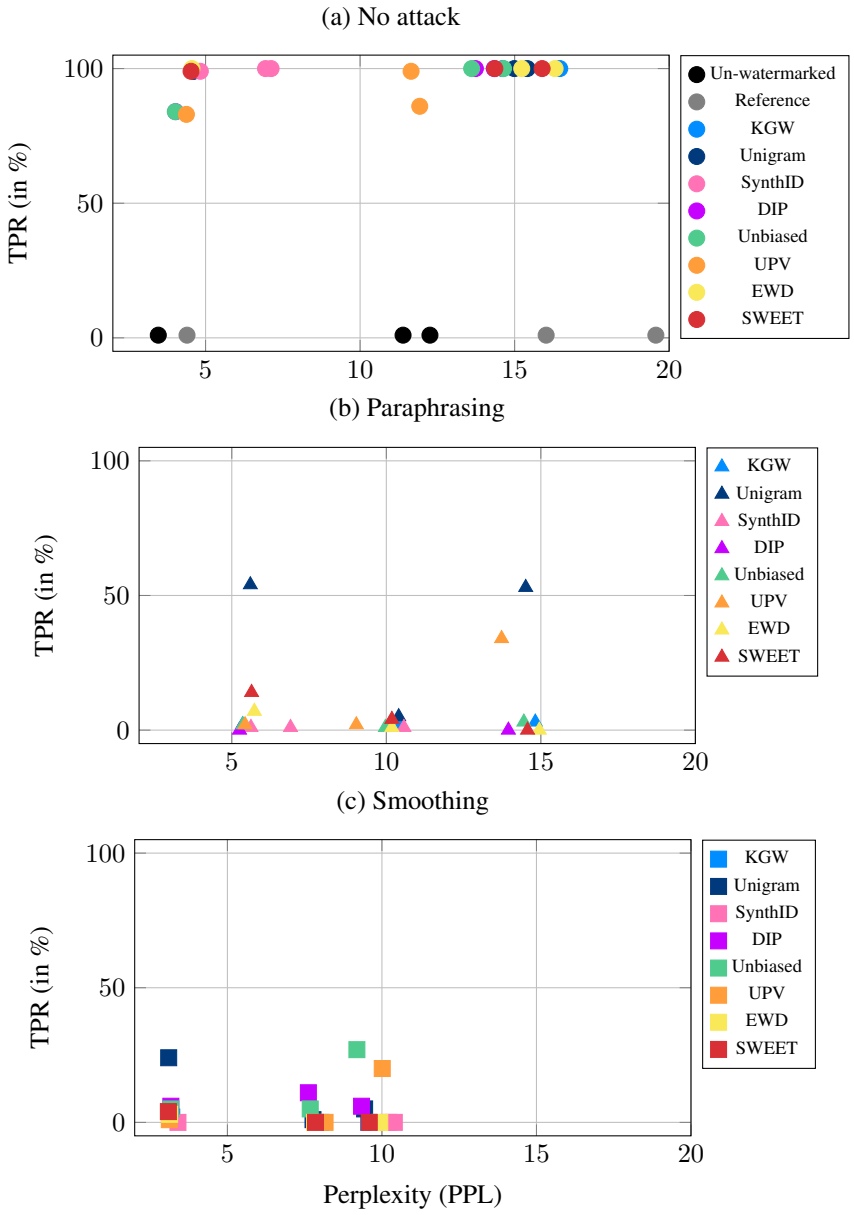


Figure 5: Each subfigure shows how the true positive rate (TPR) varies with perplexity (PPL) for a specific attack. No attack (a) corresponds to watermarked text without modifications, paraphrasing (b) uses GPT-3.5-turbo to rewrite the text, and smoothing (c) randomly replaces some tokens to remove the watermark. Colors indicate the particular watermarking method and each point corresponds to one of three models (OPT-1.3B, Llama3-8B, Qwen2-1.5B).

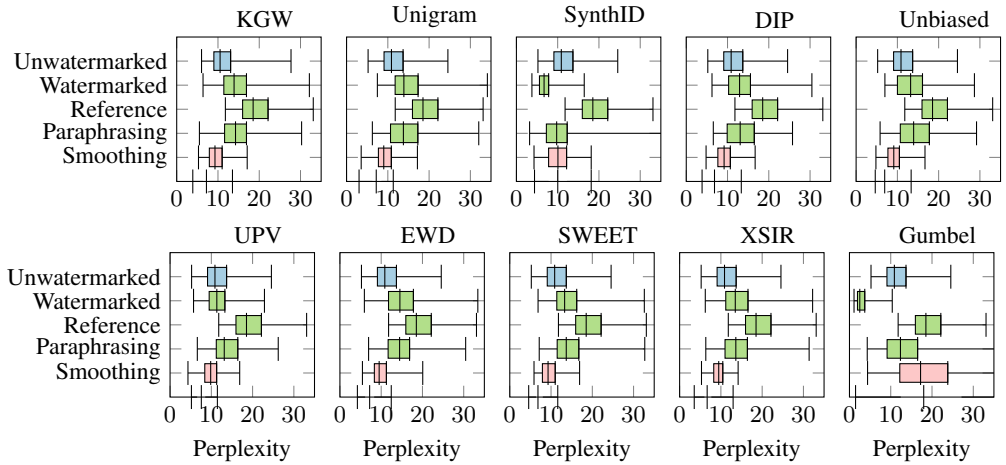


Figure 6: Text Quality Comparison – Perplexity (OPT-1.3B). Box plots of perplexity for text generated from different sources, with perplexity computed using the OPT-2.7B model. Our smoothing attack produces text with quality comparable to, and in some cases better than, that of the watermarked model.

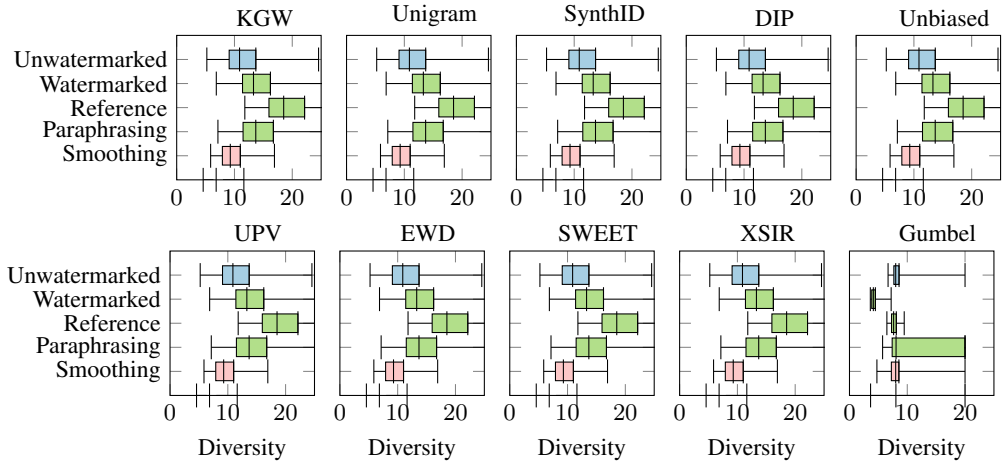


Figure 7: Text Quality Comparison – Diversity (OPT-1.3B). Box plots of text diversity for outputs generated from different sources. Our smoothing attack produces text with diversity comparable to, and in some cases lower than, that of the watermarked model due to its constrained selection process.

### B.4 Effect of $K$ and $\alpha$

Table 5 and Table 7 show the performance of smoothing attacks against different watermarking algorithms under varying values of $K$. In a smoothing attack, the adversary has access only to the top-$K$ tokens and their probabilities from both the reference and target models. Even with $K = 1$, the attack can drastically reduce the true positive rate (TPR) from 99% (without any attack) to an extremely low value, sometimes reaching 0%. This indicates that even with minimal access to both models, the smoothing attack can effectively remove watermarks. Furthermore, we observe that increasing $K$ leads to more diverse text generation, as discussed in the main paper. This is because a higher $K$ provides the attack with a larger selection of candidate tokens, allowing for greater variation in the generated text. This observation remains consistent across both the OPT-1.3B and Llama3-8B models.

Table 6 and Table 8 analyze the performance of smoothing attacks against different watermarking algorithms under varying values of $\alpha$. In this attack, the weight assigned to the top-$K$ tokens from the watermarked model is given by $c^\alpha$, while the weight for the top-$K$ tokens from the reference model is $1 - c^\alpha$, where $c$ is a confidence score between 0 and 1. A larger $\alpha$ shifts the token selection

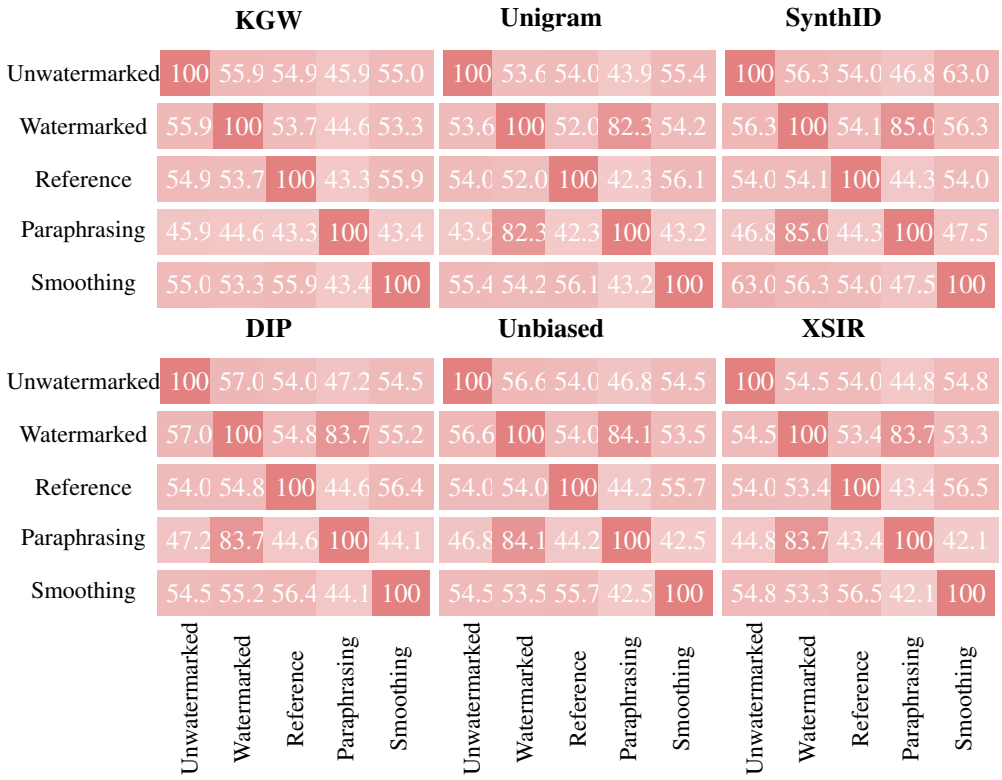


Figure 8: Text Quality Comparison – P-SP (OPT-1.3B). Heatmap comparing the similarity of text generated by different models in the sentence embedding space. Text from the watermarked model has a low similarity score compared to unwatermarked text, reflecting the inherent variability in generated responses. However, the paraphrased text (Paraphrasing vs. watermarked) exhibits a high similarity score, suggesting that the P-SP metric is more suitable for evaluating paraphrasing rather than assessing text quality degradation due to watermarking or smoothing.

Table 5: Effect of $K$ on Smoothing Attack Performance (OPT-1.3B). Evaluation of the smoothing attack's effectiveness against different watermarking algorithms on the OPT-1.3B model, varying the number of top-$K$ tokens accessible to the attacker.

| K | KGW | | | Unigram | | | SynthID | | | DIP | | | Unbiased | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 1 | 9% | 3.22 | 4.54 | 18% | 3.21 | 4.62 | 0% | 10.45 | 8.47 | 1% | 3.36 | 4.57 | 7% | 3.36 | 4.56 |
| 3 | 0.0% | 5.76 | 5.71 | 8.0% | 5.9 | 5.68 | 0.0% | 10.5 | 8.31 | 4.0% | 5.58 | 5.66 | 14.0% | 5.59 | 5.68 |
| 5 | 2.0% | 7.27 | 6.17 | 10.0% | 7.46 | 6.11 | 0.0% | 10.35 | 8.71 | 3.0% | 6.97 | 6.23 | 19.0% | 7.11 | 6.29 |
| 7 | 1.0% | 8.14 | 6.46 | 5.0% | 8.48 | 6.55 | 0.0% | 10.42 | 8.63 | 7.0% | 7.97 | 6.46 | 29.0% | 8.06 | 6.47 |
| 10 | 0.0% | 9.57 | 6.72 | 5.0% | 9.44 | 6.73 | 0.0% | 10.4 | 8.64 | 6.0% | 9.34 | 6.84 | 27.0% | 9.19 | 6.84 |

| K | XSIR | | | UPV | | | Gumbel | | | EWD | | | SWEET | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 1 | 14% | 3.31 | 4.5 | 22% | 3.62 | 4.63 | 0% | 20.8 | 8.2 | 1% | 3.31 | 4.49 | 2% | 3.41 | 4.57 |
| 3 | 17.0% | 5.69 | 5.52 | 14.0% | 6.22 | 5.87 | 2.0% | 21.72 | 8.47 | 0.0% | 5.78 | 5.71 | 0.0% | 5.64 | 5.75 |
| 5 | 8.0% | 6.8 | 6.04 | 16.0% | 7.68 | 6.3 | 8.0% | 20.3 | 8.23 | 0.0% | 7.32 | 6.18 | 0.0% | 7.15 | 6.23 |
| 7 | 10.0% | 8.26 | 6.48 | 7.0% | 8.75 | 6.65 | 9.0% | 21.15 | 8.15 | 0.0% | 8.65 | 6.52 | 0.0% | 8.47 | 6.45 |
| 10 | 9.0% | 9.47 | 6.75 | 20.0% | 10.01 | 6.89 | 9.0% | 19.25 | 8.3 | 0.0% | 9.93 | 6.78 | 0.0% | 9.59 | 6.72 |

preference toward the reference model, making the generated text more aligned with it. Conversely, a smaller $\alpha$ biases the attack toward the watermarked model, producing text that more closely resembles the watermarked output. As $\alpha$ increases, the true positive rate (TPR) decreases, leading to a higher watermark removal rate—an effect consistently observed across all watermarking methods for both the OPT-1.3B and Llama3-8B models.

Table 6: Effect of $\alpha$ on Smoothing Attack Performance (OPT-1.3B). Evaluation of the smoothing attack's effectiveness against different watermarking algorithms on the OPT-1.3B model, varying the parameter $\alpha$. A larger $\alpha$ indicates that the attack relies more on the reference model's output, while a smaller $\alpha$ means the attack is more influenced by the watermarked text.

| $\alpha$ | KGW | | | Unigram | | | SynthID | | | DIP | | | Unbiased | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 0.5 | 11.0% | 10.03 | 7.02 | 42.0% | 9.9 | 6.86 | 2.0% | 9.33 | 7.9 | 29.0% | 9.27 | 7.11 | 63.0% | 8.92 | 7.09 |
| 1.0 | 0.0% | 9.57 | 6.72 | 5.0% | 9.44 | 6.73 | 0.0% | 10.4 | 8.64 | 6.0% | 9.34 | 6.84 | 27.0% | 9.19 | 6.84 |
| 2.0 | 0.0% | 9.35 | 6.65 | 0.0% | 9.38 | 6.58 | 0.0% | 11.16 | 8.26 | 1.0% | 9.03 | 6.71 | 9.0% | 8.89 | 6.59 |
| 3.0 | 0.0% | 9.45 | 6.46 | 1.0% | 9.25 | 6.43 | 0.0% | 11.33 | 8.61 | 0.0% | 9.32 | 6.82 | 1.0% | 9.05 | 6.65 |

| $\alpha$ | X-SIR | | | UPV | | | Gumbel | | | EWD | | | SWEET | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 0.5 | 28.0% | 9.47 | 6.94 | 42.0% | 10.01 | 7.14 | 80.0% | 13.73 | 7.54 | 0.0% | 9.76 | 7.01 | 6.0% | 9.66 | 7.13 |
| 1.0 | 9.0% | 9.47 | 6.75 | 20.0% | 10.01 | 6.89 | 9.0% | 19.25 | 8.3 | 0.0% | 9.93 | 6.78 | 0.0% | 9.59 | 6.72 |
| 2.0 | 6.0% | 9.45 | 6.46 | 4.0% | 9.28 | 6.59 | 0.0% | 25.39 | 9.04 | 0.0% | 9.63 | 6.58 | 0.0% | 9.29 | 6.45 |
| 3.0 | 0.0% | 9.12 | 6.41 | 1.0% | 9.85 | 6.57 | 0.0% | 25.77 | 9.5 | 0.0% | 9.43 | 6.68 | 0.0% | 9.33 | 6.53 |

Table 7: Effect of $K$ on Smoothing Attack Performance (Llama3-8B). Evaluation of the smoothing attack's effectiveness against different watermarking algorithms on the Llama3-8B model, varying the number of top-$K$ tokens accessible to the attacker.

| K | KGW | | | Unigram | | | SynthID | | | DIP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 1 | 6% | 2.37 | 4.67 | 19% | 2.41 | 4.67 | 0% | 3.6 | 6.86 | 2% | 2.53 | 4.84 |
| 3 | 1% | 2.81 | 5.17 | 27% | 2.8 | 5.2 | 0% | 3.42 | 6.87 | 4% | 2.91 | 5.47 |
| 5 | 3% | 2.99 | 5.36 | 24% | 2.92 | 5.31 | 0% | 3.41 | 6.89 | 1% | 2.97 | 5.55 |
| 7 | 2% | 3.14 | 5.55 | 23% | 3.03 | 5.43 | 0% | 3.41 | 6.86 | 4% | 3.1 | 5.78 |
| 10 | 2% | 3.2 | 5.63 | 24% | 3.1 | 5.44 | 0% | 3.4 | 6.86 | 6% | 3.17 | 5.67 |

| K | Unbiased | | | UPV | | | EWD | | | SWEET | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 1 | 1% | 2.5 | 4.8 | 1% | 2.48 | 4.76 | 3% | 2.43 | 4.68 | 3% | 2.41 | 4.72 |
| 3 | 4% | 2.9 | 5.44 | 1% | 2.97 | 5.37 | 4% | 2.94 | 5.33 | 3% | 2.91 | 5.27 |
| 5 | 2% | 2.95 | 5.53 | 0% | 3.02 | 5.55 | 3% | 3.06 | 5.48 | 4% | 3.01 | 5.43 |
| 7 | 7% | 3.14 | 5.72 | 1% | 3.1 | 5.54 | 6% | 3.09 | 5.43 | 5% | 3.01 | 5.37 |
| 10 | 5% | 3.17 | 5.75 | 1% | 3.12 | 5.49 | 3% | 3.13 | 5.38 | 4% | 3.09 | 5.4 |

Table 8: Effect of $\alpha$ on Smoothing Attack Performance (Llama3-8B). Evaluation of the smoothing attack's effectiveness against different watermarking algorithms on the Llama3-8B model, varying the parameter $\alpha$. A larger $\alpha$ indicates greater reliance on the reference model's output, while a smaller $\alpha$ means the attack text is more influenced by the watermarked model.

| $\alpha$ | KGW | | | Unigram | | | SynthID | | | DIP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 0.5 | 13% | 3.45 | 5.92 | 62% | 3.4 | 5.77 | 0% | 3.78 | 6.88 | 35% | 3.34 | 6.19 |
| 1.0 | 2% | 3.2 | 5.63 | 24% | 3.1 | 5.44 | 0% | 3.4 | 6.86 | 6% | 3.17 | 5.67 |
| 2.0 | 0% | 3.05 | 5.28 | 12% | 2.93 | 5.21 | 0% | 3.49 | 6.87 | 3% | 2.99 | 5.23 |
| 3.0 | 0% | 2.93 | 5.17 | 12% | 2.99 | 5.26 | 0% | 3.52 | 6.83 | 1% | 2.96 | 5.16 |

| $\alpha$ | Unbiased | | | UPV | | | EWD | | | SWEET | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div | TPR | PPL | Div |
| 0.5 | 26% | 3.37 | 6.09 | 10% | 3.47 | 6.08 | 28% | 3.44 | 5.84 | 44% | 3.38 | 5.9 |
| 1.0 | 5% | 3.17 | 5.75 | 1% | 3.12 | 5.49 | 3% | 3.13 | 5.38 | 4% | 3.09 | 5.4 |
| 2.0 | 3% | 2.98 | 5.28 | 0% | 2.96 | 5.2 | 0% | 3.0 | 5.36 | 1% | 3.06 | 5.38 |
| 3.0 | 3% | 2.96 | 5.21 | 0% | 2.99 | 5.2 | 0% | 2.89 | 5.18 | 0% | 2.93 | 5.22 |

In terms of text quality, when $\alpha$ is lower, the generated text is more influenced by the watermarked model, which generally exhibits higher quality than the reference model. Consequently, decreasing

$\alpha$ can improve text quality. This provides an adversary with a way to adjust $\alpha$ to balance watermark removal and text quality preservation.

### B.5 TEXT EXAMPLE

Table 9 presents text generated by the Gumbel sampling algorithm and the smoothing attack. We observe that, although the perplexity of the watermarked text is significantly lower than that of the text from the smoothing attack, this is primarily due to repetition in the generated text. This behavior may stem from the deterministic nature of Gumbel sampling, which can lead to less diverse outputs.

Table 9: Text generated by watermarked model and smoothing attack (OPT-1.3B against Gumbel sampling).

| **Watermarked** | **Smoothing Attack** |
|---|---|
| Cluster comprises IBM's Opteron-based eServer 325 server and systems management software and storage devices that can run Linux and Windows operating systems.<br>The data center will be built on top of existing IBM Power servers. The company will offer a variety of services, including cloud-based services, as well as a "plug and play" environment that will allow users to transition to other hardware.<br>The data center will be built on top of existing IBM Power servers. The company will offer a variety of services, including cloud-based services, as well as a "plug and play" environment that will allow users to transition to other hardware. (Image: IBM)<br>The data center will be built on top of existing IBM Power servers. The company will offer a variety of services, including cloud-based services, as well as a "plug and play" environment that will allow users to transition to other hardware.<br>The data center will be built on top of existing IBM Power servers. The company will offer a variety of services, including cloud-based services, as well as a "plug and play" environment that will allow users to transition to other hardware. | Cluster comprises IBM's Opteron-based eServer 325 server and systems management software and storage devices that can run Linux and Windows operating systems.<br>IBM will start selling customers a prototype of the cluster by July, according to Jim Bessen, the executive vice president of middleware architecture and services at IBM.<br>"The cluster is just the tip of the iceberg," he said.<br>The cluster firm computing is encased in a step-by-step process starting with activities such as how to set up clinks that back up disks, store data, create back-up schemes for data, and at what resolutions to save the data.<br>"We are not done yet," he said. As new kinds of programming software and other technologies come out, users need to act on it.<br>While IBM expects internal clusters with the capability to run multiple operating systems in fuse over the next year, this capability will be available only to Enterprise Software Group (ESG) customers.<br>ESG will not sell its cluster technology to anyone else, Bessen said. |

### B.6 IMPACT OF MODEL SIZE

Table 10 presents the performance of the smoothing attack across different watermarking algorithms and varying sizes of OPT models. Perplexity (PPL) is computed with respect to the OPT-30B model, while the reference model remains consistent across all settings—the OPT-125M.

For unwatermarked models, the True Positive Rate (TPR) is consistently 0%. In contrast, watermarked models achieve near-perfect TPR. However, the smoothing attack significantly reduces TPR across all model sizes, with its impact increasing as the model size grows—for instance, TPR drops to 0% for the KGW watermark in the 30B model.

Watermarked models exhibit a notable increase in perplexity, indicating that watermarking impacts text fluency. The smoothing attack reduces perplexity, bringing it closer to unwatermarked levels, suggesting a partial recovery of fluency. Regarding diversity, the unwatermarked text demonstrates the highest variation, while watermarking constrains generation patterns, resulting in a noticeable drop in diversity. The smoothing attack further reduces diversity, primarily because tokens are sampled only from the top-K tokens of both the watermarked and reference models, limiting the range of possible candidates.

Table 10: Impact of Model Size on the Smoothing Attack (OPT). Performance of the smoothing attack across different watermarking algorithms and various sizes of OPT models. The perplexity (PPL) is computed with respect to the OPT-30B model, while the reference model is consistently the OPT-125M. The table reports True Positive Rate (TPR), Perplexity (PPL), and Diversity (Div.) for unwatermarked, watermarked, and smoothed settings.

| Size | Setting | KGW | | | Unigram | | | SynthID | | | DIP | | | Unbiased | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TPR | PPL | Div. | TPR | PPL | Div. | TPR | PPL | Div. | TPR | PPL | Div. | TPR | PPL | Div. |
| 1.3B | Unwatermarked | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 |
| | Watermarked | 100.0% | 15.94 | 8.09 | 99.0% | 16.53 | 7.29 | 100.0% | 7.7 | 7.41 | 100.0% | 15.16 | 8.44 | 99.0% | 15.14 | 8.29 |
| | Smoothing | 4.0% | 10.48 | 6.72 | 6.0% | 10.37 | 6.83 | 1.0% | 11.37 | 8.67 | 6.0% | 10.03 | 7.03 | 4.0% | 9.94 | 6.79 |
| 2.7B | Unwatermarked | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 |
| | Watermarked | 100.0% | 13.94 | 7.88 | 100.0% | 14.31 | 7.41 | 99.0% | 6.86 | 7.55 | 97.0% | 13.86 | 8.61 | 97.0% | 13.6 | 8.69 |
| | Smoothing | 4.0% | 10.35 | 6.77 | 4.0% | 10.35 | 6.66 | 6.0% | 9.84 | 8.0 | 13.0% | 9.87 | 6.84 | 6.0% | 9.85 | 6.88 |
| 6.7B | Unwatermarked | 0.0% | 10.2 | 8.45 | 0.0% | 10.2 | 8.45 | 0.0% | 10.2 | 8.45 | 0.0% | 10.2 | 8.45 | 0.0% | 10.2 | 8.45 |
| | Watermarked | 100.0% | 13.16 | 8.06 | 100.0% | 12.94 | 7.48 | 98.0% | 6.21 | 7.48 | 98.0% | 11.8 | 8.48 | 97.0% | 11.79 | 8.59 |
| | Smoothing | 4.0% | 10.07 | 6.92 | 6.0% | 10.54 | 6.68 | 3.0% | 8.98 | 8.31 | 8.0% | 9.78 | 6.86 | 8.0% | 9.68 | 6.74 |
| 13B | Unwatermarked | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 |
| | Watermarked | 100.0% | 12.88 | 8.56 | 100.0% | 12.44 | 7.39 | 100.0% | 5.88 | 7.8 | 96.0% | 11.67 | 9.34 | 93.0% | 11.42 | 8.77 |
| | Smoothing | 2.0% | 10.24 | 6.82 | 5.0% | 10.32 | 6.7 | 8.0% | 8.07 | 7.8 | 8.0% | 9.6 | 6.88 | 7.0% | 9.37 | 6.77 |
| 30B | Unwatermarked | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 |
| | Watermarked | 100.0% | 10.23 | 8.34 | 100.0% | 10.45 | 7.56 | 100.0% | 5.27 | 7.72 | 94.0% | 9.43 | 8.78 | 97.0% | 9.89 | 9.08 |
| | Smoothing | 0.0% | 9.5 | 6.8 | 7.0% | 10.15 | 6.75 | 5.0% | 6.96 | 8.04 | 4.0% | 9.34 | 6.89 | 4.0% | 9.36 | 6.88 |

| Size | Setting | X-SIR | | | UPV | | | Gumbel | | | EWD | | | SWEET | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TPR | PPL | Div. | TPR | PPL | Div. | TPR | PPL | Div. | TPR | PPL | Div. | TPR | PPL | Div. |
| 1.3B | Unwatermarked | 1.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 | 0.0% | 12.95 | 8.67 |
| | Watermarked | 94.0% | 15.42 | 7.96 | 99.0% | 12.79 | 8.22 | 98.0% | 3.15 | 4.35 | 100.0% | 16.88 | 7.92 | 100.0% | 15.99 | 8.02 |
| | Smoothing | 13.0% | 10.3 | 6.72 | 20.0% | 10.78 | 6.89 | 9.0% | 20.94 | 8.30 | 1.0% | 10.71 | 6.75 | 1.0% | 10.54 | 6.81 |
| 2.7B | Unwatermarked | 3.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 | 0.0% | 11.75 | 8.36 |
| | Watermarked | 91.0% | 14.07 | 8.25 | 99.0% | 12.30 | 8.01 | 99.0% | 2.96 | 4.38 | 100.0% | 14.88 | 7.98 | 100.0% | 14.07 | 8.32 |
| | Smoothing | 10.0% | 10.34 | 6.77 | 18.0% | 10.56 | 6.90 | 10.0% | 19.46 | 8.41 | 1.0% | 10.43 | 6.86 | 3.0% | 10.49 | 6.86 |
| 6.7B | Unwatermarked | 0.0% | 10.2 | 8.45 | 0.0% | 10.20 | 8.45 | 0.0% | 10.20 | 8.45 | 0.0% | 10.20 | 8.45 | 0.0% | 10.20 | 8.45 |
| | Watermarked | 91.0% | 13.04 | 8.19 | 97.0% | 10.92 | 7.75 | 100.0% | 2.97 | 4.49 | 100.0% | 13.42 | 8.69 | 100.0% | 13.05 | 8.41 |
| | Smoothing | 9.0% | 10.01 | 6.7 | 8.0% | 10.60 | 7.05 | 9.0% | 14.85 | 8.62 | 0.0% | 10.60 | 6.79 | 1.0% | 10.07 | 6.89 |
| 13B | Unwatermarked | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 | 0.0% | 10.14 | 8.39 |
| | Watermarked | 88.0% | 12.29 | 8.05 | 99.0% | 10.59 | 7.91 | 98.0% | 2.96 | 4.63 | 100.0% | 13.09 | 8.74 | 100.0% | 12.32 | 8.35 |
| | Smoothing | 11.0% | 9.84 | 6.79 | 12.0% | 10.84 | 6.88 | 12.0% | 15.06 | 8.27 | 0.0% | 10.16 | 6.73 | 2.0% | 10.15 | 6.74 |
| 30B | Unwatermarked | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 | 0.0% | 8.46 | 8.44 |
| | Watermarked | 91.0% | 10.43 | 8.43 | 97.0% | 8.59 | 8.13 | 97.0% | 2.89 | 4.79 | 100.0% | 10.75 | 8.54 | 100.0% | 9.98 | 8.25 |
| | Smoothing | 16.0% | 9.65 | 6.74 | 17.0% | 10.06 | 7.11 | 9.0% | 11.92 | 8.39 | 2.0% | 10.02 | 6.99 | 2.0% | 9.55 | 6.84 |

# C Analysis

## C.1 Contribution Depends on the Confidence Score of the Unwatermarked Model

We first demonstrate that the contribution of each token to the detection score is influenced by the confidence score of the unwatermarked model, as measured by its probability distribution.

### C.1.1 Case Study: Green-Red List Watermark

Suppose that $l_t$ is the logit vector for predicting the $t$-th token from the unwatermarked model, and $\mathcal{G}_t$ is the green list used by the watermarked model at position $t$, with size $\gamma|\mathcal{V}|$. Given the watermark shift $\delta$, the probabilities assigned by the unwatermarked and watermarked models are expressed as:

$$P_t(v) = \frac{\exp\big(l_t(v)\big)}{\sum_{v' \in \mathcal{V}} \exp\big(l_t(v')\big)}.$$

$$\widetilde{P}_t(v) = \frac{\exp\big(l_t(v) + \delta \cdot \mathbf{1}_{\{v \in \mathcal{G}_t\}}\big)}{\sum_{v' \in \mathcal{V}} \exp\Big(l_t(v') + \delta \cdot \mathbf{1}_{\{v' \in \mathcal{G}_t\}}\Big)}.$$

Rewriting $\widetilde{P}_t(v)$, we observe:

$$\widetilde{P}_t(v) \;=\; P_t(v) \times \frac{\exp\big(\delta\, \mathbf{1}_{\{v \in \mathcal{G}_t\}}\big)}{\sum_{v' \in \mathcal{V}} P_t(v')\, \exp\big(\delta\, \mathbf{1}_{\{v' \in \mathcal{G}_t\}}\big)}.$$

Define the normalization factor:

$$Z_\delta \;=\; \frac{\sum_{v' \in \mathcal{V}} \exp\big(l_t(v') + \delta\, \mathbf{1}_{\{v' \in \mathcal{G}_t\}}\big)}{\sum_{v' \in \mathcal{V}} \exp\big(l_t(v')\big)} \;=\; \sum_{v' \in \mathcal{V}} P_t(v')\, \exp\big(\delta\, \mathbf{1}_{\{v' \in \mathcal{G}_t\}}\big).$$

Then:

$$\widetilde{P}_t(v) \;=\; \begin{cases} \dfrac{e^\delta}{Z_\delta}\, P_t(v), & v \in \mathcal{G}_t, \\ \dfrac{1}{Z_\delta}\, P_t(v), & v \notin \mathcal{G}_t. \end{cases}$$

The expected fraction of tokens belonging to the green list under the unwatermarked model is given by:

$$\mathbb{E}_{v \sim P_t}[\mathbf{1}(v \in \mathcal{G}_t)] \;=\; \sum_{v \in \mathcal{G}_t} P_t(v) \;=\; P_{\mathcal{G}_t},$$

where $P_{\mathcal{G}_t}$ represents the probability mass assigned to green tokens in the unwatermarked model.

Similarly, the expected fraction of green tokens in the watermarked model is:

$$\mathbb{E}_{v \sim \widetilde{P}_t}[\mathbf{1}(v \in \mathcal{G}_t)] = \sum_{v \in \mathcal{G}_t} \widetilde{P}_t(v) \;=\; \frac{e^\delta}{Z_\delta}\, P_{\mathcal{G}_t}. \tag{6}$$

Since $Z_\delta = (e^\delta - 1)P_{\mathcal{G}_t} + 1$, the difference in green token probabilities (i.e., the detection contribution at token position $t$) is:

$$S_t = \mathbb{E}_{v \sim \widetilde{P}_t}[\mathbf{1}(v \in \mathcal{G}_t)] - \mathbb{E}_{v \sim P_t}[\mathbf{1}(v \in \mathcal{G}_t)] = \frac{-(e^\delta - 1)P_{\mathcal{G}_t} + (e^\delta - 1)}{(e^\delta - 1) + \frac{1}{P_{\mathcal{G}_t}}}. \tag{7}$$

In other words, the token-level detection contribution $S_t$ is a function of the probability mass $P_{\mathcal{G}_t}$ assigned to green tokens by the unwatermarked model.

### C.1.2 Case Study: Tournament Sampling Watermark

In the Tournament Sampling watermark, when generating the $t$-th token, the algorithm assigns scores to each token using $m$ independent watermarking functions $g^{(1)}, ..., g^{(m)}$. These scores depend on a random seed generated based on the recent context and a secret watermarking key. The token selection follows a multi-round elimination process, where $2^m$ tokens are first sampled from $P_t(\cdot)$, then compete in $m$ rounds to determine the final output.

Despite the complex sampling mechanism, the probability of each token in the modified distribution $\widetilde{P}_t$ is adjusted by a factor dependent on its assigned $g$ value. Specifically, for any token $v$:

$$\widetilde{P}_t(v) = \begin{cases} P_t(v) \cdot (1 - P_{\mathcal{G}_t}) & \text{if } g(v) = 0, \\ P_t(v) \cdot (2 - P_{\mathcal{G}_t}) & \text{if } g(v) = 1. \end{cases} \tag{8}$$

During watermark detection, the detector computes the average $g$ value across all tournament layers, i.e., $\frac{1}{m}\sum_{l=1}^{m} g^{(l)}(v)$, as the watermark score for the token.

*Single Tournament Layer ($m = 1$).* Consider the simplest case where $m = 1$, meaning only one tournament round is used. Let $\mathcal{G}_t$ denote the set of tokens where $g^{(1)}(v) = 1$. The probability modification simplifies to:

$$\widetilde{P}_t(v) = \begin{cases} P_t(v) \cdot (1 - P_{\mathcal{G}_t}) & \text{if } v \notin \mathcal{G}_t, \\ P_t(v) \cdot (2 - P_{\mathcal{G}_t}) & \text{if } v \in \mathcal{G}_t. \end{cases} \tag{9}$$

The expected $g$ value for tokens sampled from $\widetilde{P}_t$ is $(2 - P_{\mathcal{G}_t}) \cdot P_{\mathcal{G}_t}$, while the expectation under $P_t$ is simply $P_{\mathcal{G}_t}$. Thus, the detection contribution $S_t$ is:

$$S_t = (1 - P_{\mathcal{G}_t}) \cdot P_{\mathcal{G}_t}. \tag{10}$$

This mirrors the Green-Red List watermark, showing that the detection contribution per token is fundamentally tied to $P_{\mathcal{G}_t}$.

### C.2 Low Model Confidence Leads to Large Variance in the Watermark Score for Unwatermarked Text

Thus far, we have established that the contribution of each token to the detection score is correlated with the expected watermark score under the unwatermarked model. We now analyze what affects the watermark score of the unwatermarked model.

Let $P_t = (p_1, p_2, \ldots, p_d)$ be the probability vector from the unwatermarked model at token position $t$, where $p_i \in [0, 1]$ and $\sum_{i=1}^{d} p_i = 1$. Typically, $d = |\mathcal{V}|$ is large. We randomly select a subset $\mathcal{G}_t \subset \{1, \ldots, d\}$ of indices of size $\gamma|\mathcal{V}|$. Define the random variable:

$$P_{\mathcal{G}_t} \;=\; \sum_{i \in \mathcal{G}_t} p_i.$$

We analyze how $P_{\mathcal{G}_t}$ is distributed over all possible assignments of $\mathcal{G}_t$. Define the indicator variable $X_i$ as follows:

$$X_i \;=\; \begin{cases} 1, & \text{if } i \in \mathcal{G}_t, \\ 0, & \text{otherwise.} \end{cases}$$

Since each token is independently assigned to $\mathcal{G}_t$ with probability $\gamma$, we have:

$$\mathbb{E}[X_i] = \gamma, \quad \text{and} \quad \mathrm{Var}(X_i) = \gamma(1 - \gamma).$$

For different token indices $i \neq j$, the covariance between their assignments is:

$$\mathrm{Cov}(X_i, X_j) = \mathbb{E}[X_i X_j] - \mathbb{E}[X_i]\mathbb{E}[X_j].$$

For Poisson sampling (i.e., assigning each token to $\mathcal{G}_t$ independently with probability $\gamma$), the covariance is zero. However, under a fixed-size sampling setup (i.e., selecting exactly $\gamma|\mathcal{V}|$ tokens), we have:

$$\mathrm{Cov}(X_i, X_j) = \frac{\gamma|\mathcal{V}|}{d} \cdot \frac{\gamma|\mathcal{V}| - 1}{d - 1} - \gamma^2 = -\frac{\gamma(1 - \gamma)}{|\mathcal{V}| - 1}.$$

Expressing $P_{\mathcal{G}_t}$ in terms of $X_i$, we obtain:

$$P_{\mathcal{G}_t} \;=\; \sum_{i=1}^{d} X_i \, p_i.$$

**Expectation and Variance of $P_{\mathcal{G}_t}$.** The expectation is:

$$\mathbb{E}[P_{\mathcal{G}_t}] \;=\; \sum_{i=1}^{d} \mathbb{E}[X_i]\, p_i \;=\; \gamma \sum_{i=1}^{d} p_i \;=\; \gamma.$$

The variance is:

$$\mathrm{Var}(P_{\mathcal{G}_t}) = \sum_{i=1}^{d} p_i^2 \mathrm{Var}(X_i) + \sum_{i \neq j} p_i p_j \mathrm{Cov}(X_i, X_j).$$

Substituting $\mathrm{Var}(X_i) = \gamma(1-\gamma)$ and $\mathrm{Cov}(X_i, X_j) = -\frac{\gamma(1-\gamma)}{|\mathcal{V}|-1}$:

$$\mathrm{Var}(P_{\mathcal{G}_t}) = \gamma(1-\gamma) \sum_{i=1}^{d} p_i^2 - \frac{\gamma(1-\gamma)}{|\mathcal{V}|-1} \sum_{i \neq j} p_i p_j.$$

For the first term,

$$\gamma(1-\gamma) \sum_{i=1}^{d} p_i^2 = \gamma(1-\gamma)\sigma^2,$$

where $\sigma^2 = \sum_{i=1}^{d} p_i^2$ represents the squared $\ell_2$ norm of the probability vector.

For the second term, using the identity:

$$\sum_{i \neq j} p_i p_j = \left( \sum_{i=1}^{d} p_i \right)^2 - \sum_{i=1}^{d} p_i^2 = 1 - \sigma^2,$$

and we obtain:

$$\frac{\gamma(1-\gamma)}{|\mathcal{V}|-1} \sum_{i \neq j} p_i p_j = \frac{\gamma(1-\gamma)}{|\mathcal{V}|-1}(1-\sigma^2).$$

For large $|\mathcal{V}|$, the correction term $\frac{\gamma(1-\gamma)}{|\mathcal{V}|-1}(1-\sigma^2)$ becomes negligible, and we approximate:

$$\mathrm{Var}(P_{\mathcal{G}_t}) \approx \gamma(1-\gamma)\sigma^2.$$

**Interpretation.** This analysis shows that $P_{\mathcal{G}_t}$ depends on the probability mass distribution.

High-Uncertainty Case (Uniform Distribution): If $p_i = \frac{1}{|\mathcal{V}|}$ for all $i$, then

$$\sigma^2 = \sum_{i=1}^{|\mathcal{V}|} \frac{1}{|\mathcal{V}|^2} = \frac{1}{|\mathcal{V}|}.$$

For large $|\mathcal{V}|$, $\sigma^2$ is small, meaning that the distribution of $P_{\mathcal{G}_t}$ concentrates tightly around $\gamma$ with small variance. This corresponds to a scenario where the model has high uncertainty, spreading probability mass nearly uniformly over all tokens.

Low-Uncertainty Case (Dominant Tokens): In practice, language models often assign high probability mass to a small number of dominant tokens. Suppose $p_j \geq 0.8$ for some token $j$, then:

$$\sigma^2 \geq p_j^2 = 0.64.$$

In this case, $\sigma^2$ is much larger than $1/|\mathcal{V}|$ (which is on the order of $10^{-5}$ for large models). Consequently, $P_{\mathcal{G}_t}$ exhibits a bimodal distribution: it is either close to 0 or close to 1, depending on whether the dominant tokens are in $\mathcal{G}_t$. The probability of $P_{\mathcal{G}_t} \approx \gamma$ is nearly zero.

Thus, when the model is confident in its predictions (low uncertainty), the variance of $P_{\mathcal{G}_t}$ is large, leading to a higher variance in the watermark score. Conversely, when the model is uncertain, the watermark score is more stable and centered around $\gamma$.

**Connection to Watermark Detection.** Since the contribution to the detection score $S_t$ depends on $P_{\mathcal{G}_t}$ (Eq. equation 4), its variance is governed by $\mathrm{Var}(P_{\mathcal{G}_t})$. This means that tokens generated with high confidence contribute more variability to the detection score, whereas tokens generated under uncertainty contribute less variability.

### C.3 Estimating the Confidence Score of the Unwatermarked Model Using the Watermarked Model

Our goal is to estimate the squared $\ell_2$ norm of the probability distribution $\|P_t\|^2$, which serves as a confidence measure for the unwatermarked model, using only access to the watermarked model $\widetilde{P}_t$. This estimation is critical for adaptive attacks and for understanding how watermarking affects text quality.

**Setup.** We consider the Green-Red List watermarking scheme, where the probability distribution $\widetilde{P}_t$ is obtained by modifying $P_t$ as:

$$\widetilde{P}_t(v) \;=\; \frac{e^{\delta \mathbf{1}_{\{v \in \mathcal{G}_t\}}}}{Z_\delta} P_t(v),$$

where the normalization factor $Z_\delta$ is defined as:

$$Z_\delta = (1 - P_{\mathcal{G}_t}) + e^\delta P_{\mathcal{G}_t}.$$

We aim to construct an estimator $\widehat{U}$ for the confidence measure:

$$\|P_t\|^2 = \sum_{v \in \mathcal{V}} P_t(v)^2.$$

**Expected Squared Norm of the Watermarked Model.** Since each probability mass in $P_t$ is scaled by either $e^\delta / Z_\delta$ (if in $\mathcal{G}_t$) or $1/Z_\delta$ (if not in $\mathcal{G}_t$), we have:

$$\mathbb{E}[\widetilde{P}_t(v)^2] \;=\; (1-\gamma)\frac{1}{Z_\delta^2} P_t(v)^2 + \gamma \frac{e^{2\delta}}{Z_\delta^2} P_t(v)^2.$$

Summing over all tokens in $\mathcal{V}$, we obtain:

$$\mathbb{E}[\|\widetilde{P}_t\|^2] \;=\; \frac{(1-\gamma) + \gamma e^{2\delta}}{Z_\delta^2} \|P_t\|^2.$$

**Unbiased Estimator.** Rearranging the above expression, we define an unbiased estimator:

$$\widehat{U} = \frac{Z_\delta^2}{(1-\gamma) + \gamma e^{2\delta}} \|\widetilde{P}_t\|^2.$$

Taking expectation, we confirm:

$$\mathbb{E}[\widehat{U}] = \|P_t\|^2.$$

**Practical Approximation.** Since $Z_\delta$ depends on $P_{\mathcal{G}_t}$, which is unknown to an adversary, we approximate it using $\gamma$:

$$Z_\delta \approx (1-\gamma) + \gamma e^\delta.$$

Thus, the practical estimator becomes:

$$\widetilde{U} = \frac{[(1-\gamma)+\gamma e^\delta]^2}{(1-\gamma)+\gamma e^{2\delta}} \|\widetilde{P}_t\|^2.$$

This provides a computationally efficient way to estimate $\|P_t\|^2$ using only $\widetilde{P}_t$, making it useful for designing attacks.

### C.4 Estimating the $\ell_2$ Norm Using Top-$K$ Probabilities

While we have established the connection between the squared $\ell_2$ norm $\|P_t\|^2$ of the probability distribution and its contribution to the watermark detection score, direct access to this quantity is often unavailable, even for the watermarked model. In this section, we show how to estimate $\|P_t\|^2$ using only limited access to the model's top-$K$ probabilities.

Suppose we only have access to the top-$K$ probabilities:

$$p_1 \geq p_2 \geq \cdots \geq p_K,$$

where the remaining probabilities $p_{K+1}, \ldots, p_{|\mathcal{V}|}$ are unknown. Define the remaining probability mass of the tail as:

$$R = 1 - \sum_{i=1}^{K} p_i.$$

Our goal is to estimate the squared $\ell_2$ norm:

$$\|P_t\|^2 = \sum_{i=1}^{|\mathcal{V}|} p_i^2,$$

given only $p_1, \ldots, p_K$ and $R$.

We bound $\|P_t\|^2$ by considering two extreme ways in which the unknown tail probabilities could be distributed:

1. Uniform Tail: The remaining probability mass $R$ is evenly distributed across the unknown tokens, minimizing the sum of squares.
2. Concentrated Tail: The entire probability mass $R$ is assigned to a single token, maximizing the sum of squares.

**Uniform Tail (Lower Bound)** If the tail probability mass $R$ is *uniformly* spread among the remaining $|\mathcal{V}| - K$ tokens, then each unknown probability is $\frac{R}{|\mathcal{V}|-K}$. The squared sum of the tail probabilities is then:

$$\sum_{i=K+1}^{|\mathcal{V}|} p_i^2 \;=\; (|\mathcal{V}|-K)\left(\frac{R}{|\mathcal{V}|-K}\right)^2 \;=\; \frac{R^2}{|\mathcal{V}|-K}.$$

Since distributing the mass uniformly minimizes the squared sum (due to convexity), this scenario provides a lower bound for $\|P_t\|^2$:

$$\|P_t\|^2 \;\geq\; \sum_{i=1}^{K} p_i^2 + \frac{R^2}{|\mathcal{V}| - K}.$$

**Concentrated Tail (Upper Bound)** At the other extreme, if the entire remaining probability mass $R$ is assigned to a single token, then the squared sum of the tail probabilities is simply:

$$\sum_{i=K+1}^{|\mathcal{V}|} p_i^2 = R^2.$$

Since concentrating all probability mass in one entry maximizes the sum of squares, this provides an upper bound for $\|P_t\|^2$:

$$\|P_t\|^2 \;\leq\; \sum_{i=1}^{K} p_i^2 + R^2.$$

Combining both bounds, we obtain:

$$\sum_{i=1}^{K} p_i^2 + \frac{R^2}{|\mathcal{V}| - K} \;\leq\; \|P_t\|^2 \;\leq\; \sum_{i=1}^{K} p_i^2 + R^2,$$

where $R = 1 - \sum_{i=1}^{K} p_i$.

**Practical Approximation.** A commonly used practical heuristic is to assume that the remaining probability mass $R$ follows a uniform distribution across the unknown probabilities. Under this assumption, we approximate:

$$\|P_t\|^2 \;\approx\; \sum_{i=1}^{K} p_i^2 + \frac{R^2}{|\mathcal{V}| - K}.$$

This estimate tends to be slightly lower than the true value, since in reality, the tail probabilities are rarely perfectly uniform—some tokens may have slightly higher probabilities than others. However, in the case of language modeling, probability distributions often exhibit a "long tail" where the remaining probability mass is spread across many small values. In such cases, the uniform assumption serves as a reasonable first-order approximation.

## D Possible Defenses to Smoothing Attack

Our attack exploits the correlation between a token's contribution to the watermark detection score and the confidence level of the unwatermarked model in predicting that token. One possible defense against this attack is to restrict access to confidence-related information, such as returning only the most probable token without revealing its probability. Note that, if the probability of the most likely token is available, our attack remains effective.

However, such a defense is challenging to enforce in practice. Many existing LLM services provide *top-$K$ probabilities* (e.g., OpenAI's API returns probabilities for the top 20 tokens), which is already sufficient to approximate model confidence and execute our attack. Moreover, service providers often release these probabilities to enhance transparency and build trust by providing insights into the model's reasoning, addressing concerns about the opacity of AI systems European Commission (2021); OECD (2019).

Access to probability distributions is also essential for debugging and evaluating model performance, as it allows developers to identify biases, diagnose overconfidence, and improve reliability National

Institute of Standards and Technology (NIST) (2023). Probabilities support explainable AI (XAI) by revealing model uncertainty, enabling users to interpret predictions and explore alternative suggestions Brown et al. (2020). From an ethical standpoint, making probability distributions available facilitates bias auditing and aligns with broader efforts to promote fairness and accountability in AI OECD (2019). Additionally, probability information empowers developers and end users by enabling advanced decision-making strategies, such as re-ranking, rejection sampling, and beam search OpenAI (2023). Furthermore, it helps mitigate risks associated with model overconfidence and hallucinations, which is particularly crucial in high-stakes domains such as healthcare and law National Institute of Standards and Technology (NIST) (2023).

Given the practical difficulties in restricting access to confidence-related information, our findings suggest that existing watermarking techniques may be vulnerable when model confidence can be estimated. This highlights the need for developing watermarking schemes that remain effective even in scenarios where adversaries have partial access to confidence estimates. Future research should explore watermarking methods that explicitly account for the model's confidence and ensure robustness against adversarial attacks that exploit confidence information.