



# Lecture 14 Watermarking GenAI (Part 2)

DSC 291 Safety in GenAI 2025 Fall

Yu-Xiang Wang



Check in here

# Recap: Last two lecture

- AI classifiers do not quite work
- Watermarking provides a more reliable solution
- Green-Red watermark scheme
- How do we evaluate a watermarking scheme

# Recap: An LM Watermarking Scheme has two components

- **Watermark**( $\mathcal{M}$ ): (possibly randomized procedure) that outputs a new model  $\hat{\mathcal{M}}$ , and detection key  $k$

$$\text{Watermark}(\mathcal{M}) \rightarrow (\hat{\mathcal{M}}, k \text{ key})$$



- **Detect**( $k, \mathbf{y}$ ): takes input detection key  $k$  and sequence  $\mathbf{y}$ , then outputs 0 or 1



# Recap: Green-Red Watermark

(Kirchenbauer et al. 2023; Zhao et al. 2023)

$$\mathcal{M}: y_t \sim \text{Softmax}(\text{logits}(\text{Prompt}, y_{<t}))$$

$$\hat{\mathcal{M}}: y_t \sim \text{Softmax}(\text{logits}(\text{Prompt}, y_{<t}) + \delta \cdot \mathbf{1}(\cdot \text{ is green}))$$

**Increase the probability of green tokens** slightly.

**Decrease the probability of red tokens** slightly.

# Recap: Detection of Green-Red Watermark

Input: Suspect text  $y = [y_1, \dots, y_n]$ , e.g. “Over the ...”

(Optional pre-processing)  $y = \text{unique}(y)$

1. Compute the **z-score**:

$$z = (|y|_G - \gamma n) / \sqrt{n\gamma(1 - \gamma)}$$

2. If  $z > \text{threshold}$  then

Return “y is watermarked”

Else

Return “no evidence”

Num of **Green tokens**

# Recap: How is the *Green* list generated? The idea of an “**m-gram watermark**”





- **Randomly selecting**  $\gamma$  fraction of the vocabulary, e.g., 0.5
- (Kirchenbauer et al. [KGW-Watermark]): Different green list at each time  $t$  as function of the prefix with length  $(m-1)$ .  
Default:  $m=2$

You were having a great time at a bar. Suddenly, she showed up. You said **to your pal**: \_\_\_

m-Gram with  $m = 4$

- (Zhao et al. [Unigram-Watermark]): Use  $m = 1$ , i.e., a consistent “Green list”.

# Recap: Desired Properties of an Ideal Watermark

- **Quality of Generated Text** 
- **Detection Accuracy Guarantee** 
  - Type I error: “No false positives” → won’t catch human text
  - Type II error: “No false negatives” → won’t miss LLM text
- **Robustness Guarantee** 
  - Be robust against evasion attacks, e.g., post-editing.
- **Security Guarantee** 
  - Can not easily learn wm key, or produce wm content otherwise.

Check paper “SoK: Watermarking for AI-Generated Content” for more details

# Recap: FPR in watermarking is **distribution-free!**

FPR = Probability of “Detector” making a mistake for **any fixed Input**.

**Randomness is over the secret key only!**

- **Different from ML experiments:**
- $\text{FPR} = \frac{\text{\# of False Positive}}{\text{Total \# of Negative Examples}}$
- Implicitly, this FPR is specific to the data distribution  $P(\text{input } x \mid \text{label of } x \text{ is “-”})$

# Recap: Quality, Detectability and Robustness of Green-Red watermark

- **Quality:**  $D_{KL}(\mathcal{M} || \hat{\mathcal{M}}) \leq \min\{\delta, \frac{\delta^2}{8}\}$
- **Detectability:** high-entropy, non-homophily text longer than  $\Omega(\log(1/\alpha))$  can be detected w.h.p. at guaranteed FPR at  $\alpha$
- **Robustness:** All m-gram watermarks are robust to edits. (why?) They can tolerate roughly  $O(n/m)$  edits.
- **Security:** Increase m makes it harder to learn the secret key (the length of the key gets longer)

# Limitation of the Green-Red Watermark

- It changes the distribution of the Language Model
- Choice of  $\delta$  determines quality-detectability tradeoff.

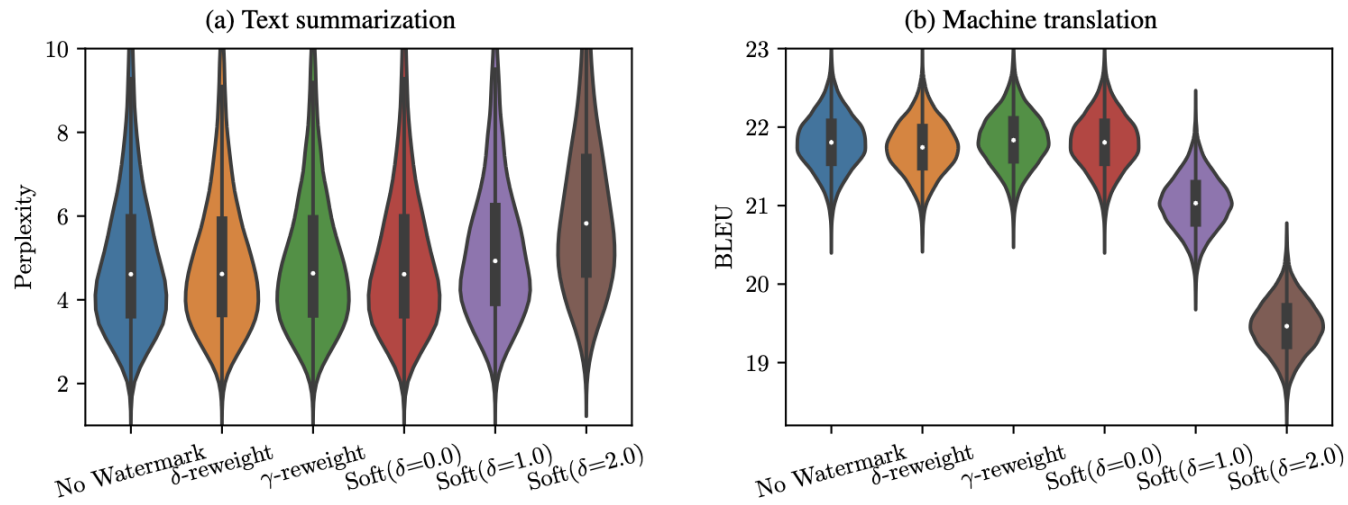
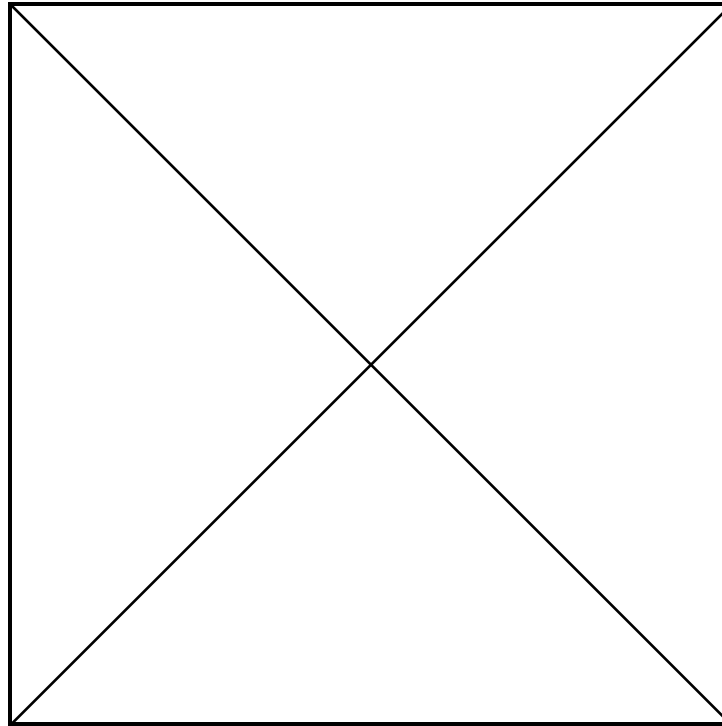


Figure 3: Distribution of perplexity of output for TS and BLEU score for MT.

(Figure from Hu et al 2023 “unbiased watermark for LLMs”)

Quality  
(of LLM text)

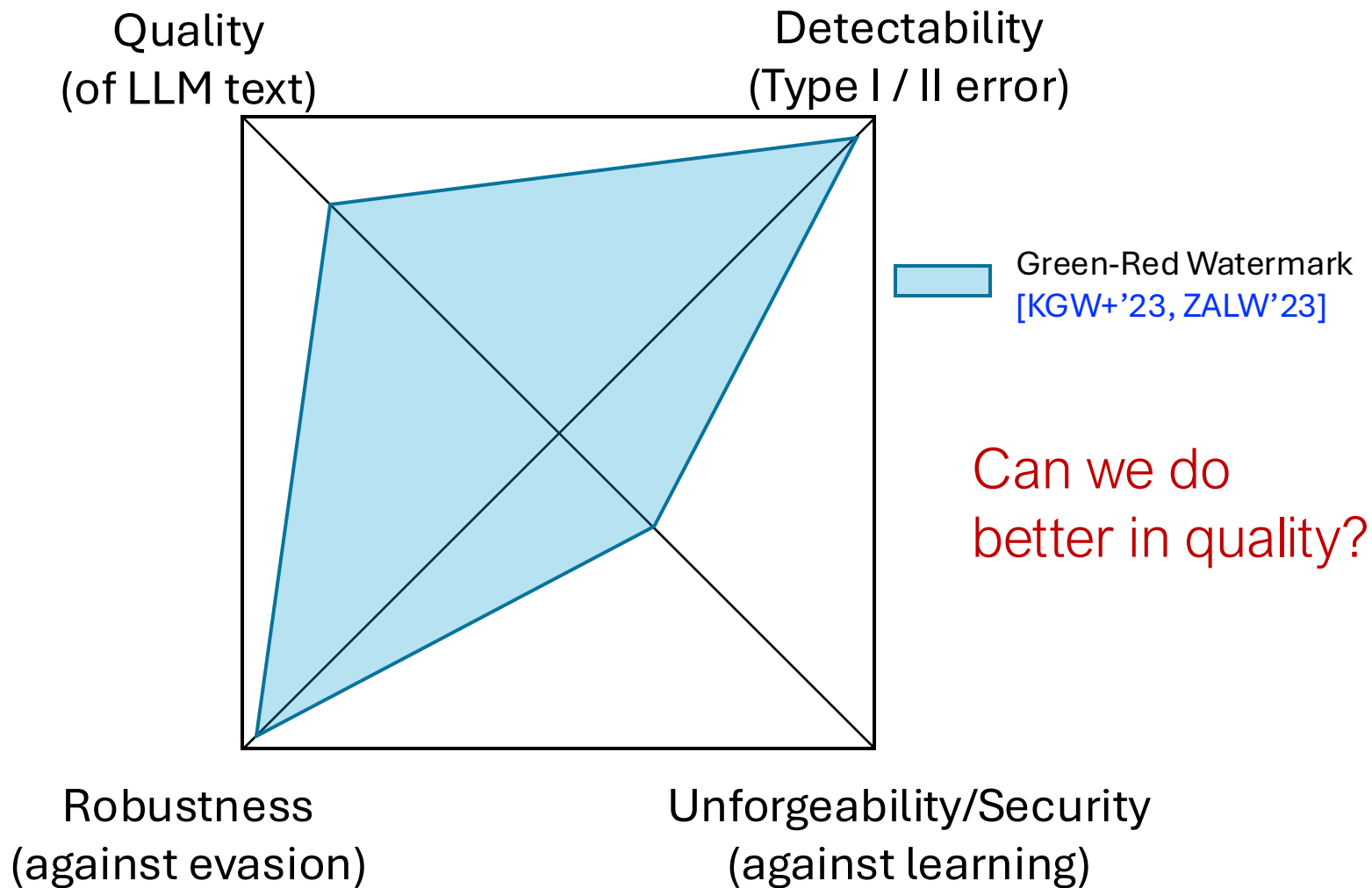
Detectability  
(Type I / II error)



Robustness  
(against evasion)

Unforgeability / Security  
(against learning)

We will put  
different  
watermarks on  
this diagram!



# Today: Continue with LLM Watermarking Schemes

- “Distortion-Free” watermarks
- “Undetectable” watermarks
- Attacks on watermarking schemes
  - “On the reliability of watermarks for LLM” by Varadraj Bartakke and Jay Sawant
  - “Watermarking in the Sand” by Benjamin TenWolde

# There are watermarking schemes that are “Distortion Free” (aka “unbiased”)

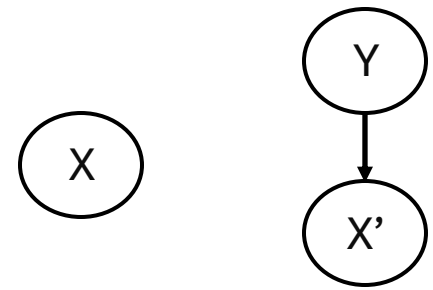
**“Distortion-Free”**: For any “Input”

$\mathcal{M}(\text{Input}) \sim \hat{\mathcal{M}}(\text{Input})$ , i.e., they are identically distributed.

- Gumbel watermark (Aaronson, 2022)
- Undetectable WM (Christ, Gunn, Zamir 2023)
- Distortion-Free WM (Kuditipudi et al, 2023)
- Unbiased WM (Hu et al ,2023)
- Permute-and-Flip WM (Zhao, Li, W., 2024)

# Demystify “distortion-free” property: How is it possible?

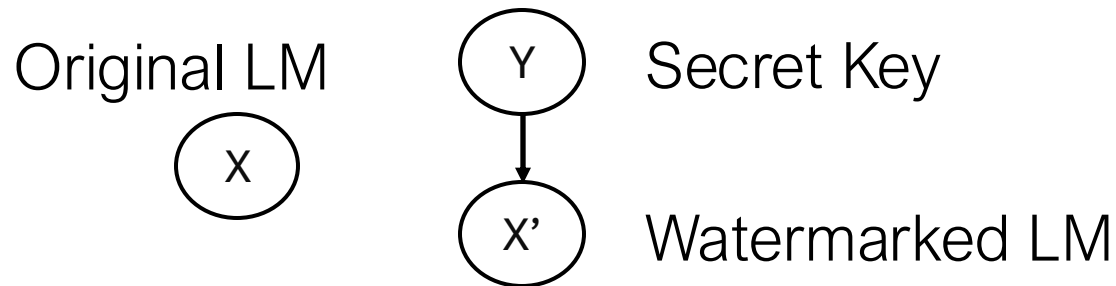
- **Example:**  $X \sim \text{Bernoulli}(0.7)$ ,  
 $Y \sim \text{Uniform}([0,1])$ ,  $X' = 1(Y < 0.7)$ .



- Check that:
  - $X \sim X'$  marginally (i.e., they are identically distributed)
  - But if we observe  $Y$ ,  $X'|Y$  is deterministic.

$X$  and  $X'$  are only marginally identically distributed.  
Knowledge of  $Y$  creates the “asymmetry” we need.

# From the Latent Variable view of LLM Watermarking schemes



- In Green-Red watermark,  $Y$  is the (random) green list.
- But the marginal distribution of  $X'$  is **not the same** as  $X$ .

# Gumbel-Softmax trick and Gumbel Watermark

- Gumbel-Softmax trick (Gumbel, 1948)

$$y_t \sim \text{Softmax} \left( \frac{u_t(y)}{T} \right) \Leftrightarrow y_t = \arg \max_{u \in \mathcal{V}} \frac{u_t(y)}{T} + G_t(y)$$

$G_t(y) \sim \text{Gumbel}(0, 1) \text{ i.i.d}$

- Idea of the Gumbel Watermark (Aaronson, 2022)

Make them pseudo-random!



- The Gumbel noises are the “hidden variables” determined by the pseudo-random functions that we can secret keys.

# Intuition behind the Gumbel Watermark

$$y_t = \arg \max_{y \in \mathcal{V}} \frac{u_t(y)}{T} + G_t(y)$$

- Without the secret key: (notice that  $G_t$  are random). The distribution of next token remains unchanged!
- With the secret key, the sequence is deterministic!
- In Detection phase: we don't have the prompt, nor the next token probability. But the selected  $y_t$  is biased towards larger  $G_t$  regardless.

# Detection score of Gumbel Watermark

$\text{Gumbel}(0, 1) \sim -\log(\log(1/\text{Uniform}([0, 1])))$ .

- Let  $r$  be the pseudo-random vector iid uniform for every coordinate.

$$\text{TestScore}_{\text{Gumbel}}(y_{1:n}) = \sum_{t=m+1}^n -\log(1 - r_t(y_t)).$$

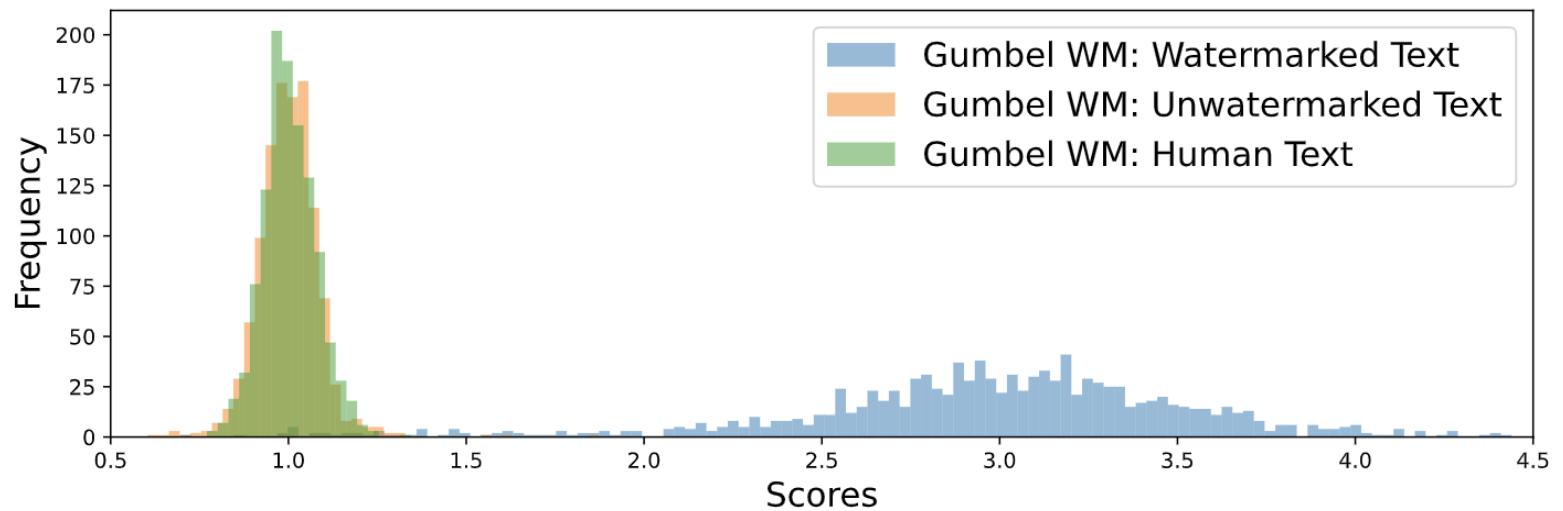
**No watermark**

$$\mathbf{E}[\text{TestScore}(y_{1:n})] = n - m$$

**Watermarked**

$$\begin{aligned} \mathbf{E}[\text{TestScore}(y_{1:n})] &= \sum_{t=m+1}^n \mathbf{E} \left[ \sum_{y \in \mathcal{V}} p_t(y) H_{\frac{1}{p_t(y)}} \right] \\ &\geq (n - m) + \left( \frac{\pi^2}{6} - 1 \right) \sum_{t=m+1}^n \mathbf{E} [\text{Entropy}[p_t(\cdot)]] . \end{aligned}$$

# Detection score of Gumbel WMs in practice



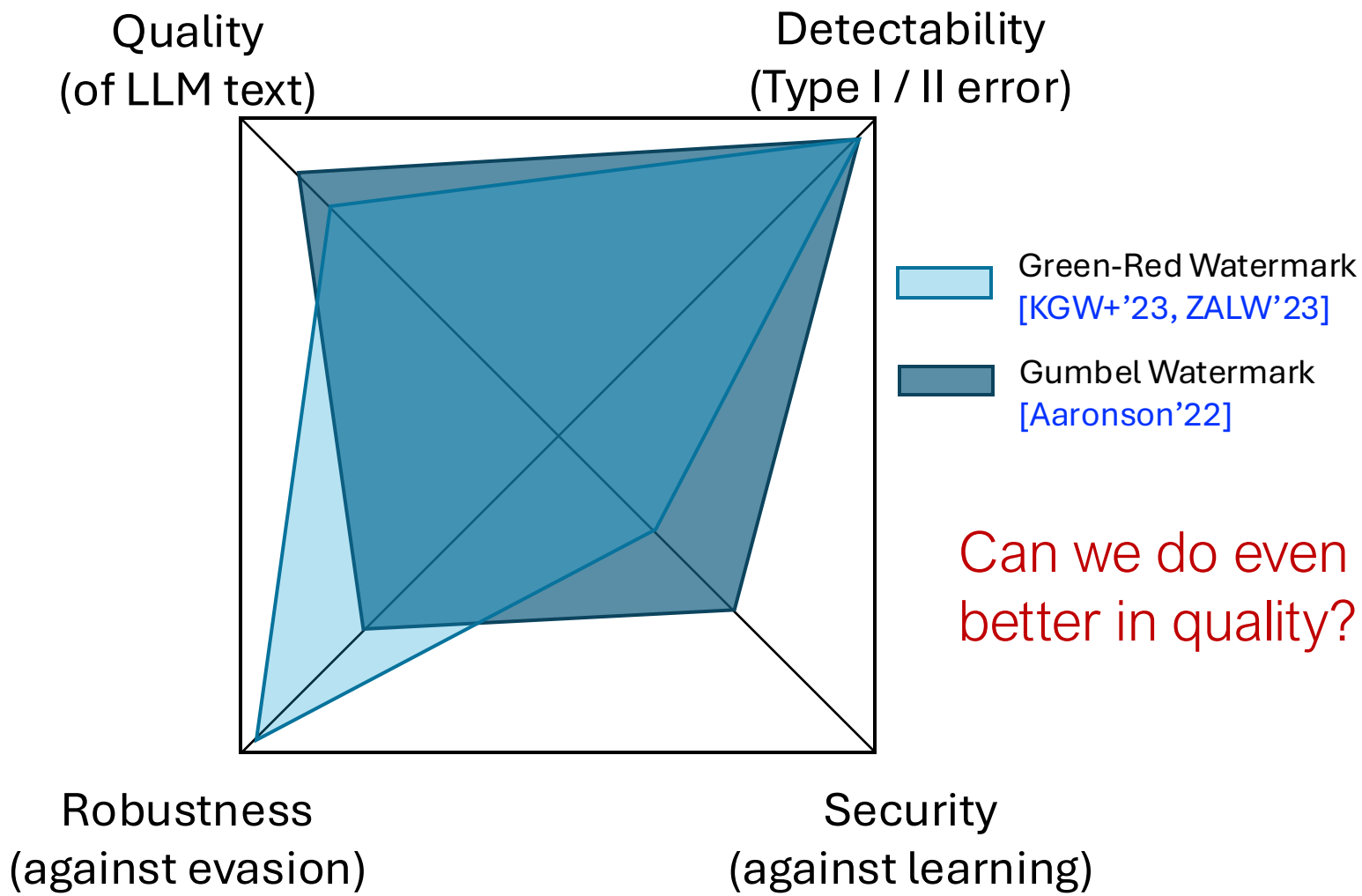
# Robustness of Gumbel WM is not bad

- Not “unigram WM” type robust, but still quite

Setting	Method	AUC	1% FPR		10% FPR	
			TPR	F1	TPR	F1
No attack	KGW	0.998	0.996	0.989	1.000	0.906
	Gumbel	0.992	0.979	0.979	0.986	0.913
	PF	0.996	0.977	0.980	0.993	0.898
DIPPER-1	KGW	0.661	0.057	0.105	0.317	0.416
	Gumbel	0.838	0.367	0.529	0.642	0.697
	PF	0.824	0.374	0.537	0.622	0.684
DIPPER-2	KGW	0.638	0.051	0.096	0.278	0.375
	Gumbel	0.764	0.239	0.380	0.523	0.608
	PF	0.795	0.250	0.394	0.544	0.625
Random Delete (0.3)	KGW	0.936	0.484	0.644	0.881	0.844
	Gumbel	0.981	0.941	0.960	0.959	0.898
	PF	0.985	0.936	0.956	0.966	0.888

DIPPER-1  
DIPPER-2  
are “paraphrasing attacks”

(Table 3 of  
<https://arxiv.org/abs/2402.05864>)



# What's “even-better” than “distortion-free”?

- Sentence level distortion-free
  - (Kuditipudi et al, 2023): “Get multiple keys, rotate the keys being used. In detection time, test with all keys”
  - (Hu et al ,2023): “unique prefix each time within a sentence”
- Polynomially many sentence (*computational*) distortion-free
  1. Do the above two across many sentences.
    - 2. (Christ, Gunn, Zamir, 2023): “Accumulate sufficient amount entropy before adding watermark! ”

# Key idea of the CGZ watermark

- First of all, CGZ is similar to Gumbel WM
  - They make the generated content secretly deterministic for those who observe the latent R.V.
  - The latent R.V. is determined by a keyed pseudo-random function (i.e., a hash-function) that takes a prefix as input.
- What's different?
  - The length of the prefix is longer and elastic (instead of fixed at  $m$  as in Gumbel and Green-Red)
  - CGZ does not add watermark unless the “entropy” is high enough.
- **Goal of these design:** ensure that the same input to the PRF does not appear more than once with high probability.

# (A simplified) CGZ watermark pseudo-code

(From the exposition of Zamir:  
<https://arxiv.org/abs/2401.10360>)

WLOG: consider vocabulary  $\mathcal{V} = \{0,1\}$ , let the latent-variables be  $F_k(r, i)$

---

## Algorithm 1 Watermarking algorithm $\text{Wat}_k$

---

**Data:** A prompt (PROMPT) and a secret key  $k$

**Result:** Watermarked text  $x_1, \dots, x_L$

$i \leftarrow 1$

$H \leftarrow 0$

**while**  $\text{done} \notin (x_1, \dots, x_{i-1})$  **do**

$p_i \leftarrow \text{Model}(\text{PROMPT}, x_1, \dots, x_{i-1})$

**if**  $H < \lambda$  **then**

        // Collect more internal entropy

        Sample  $(x_i, p) \leftarrow (1, p_i)$  with probability  $p_i$ , otherwise  $(0, 1 - p_i)$

$H \leftarrow H - \log p$       Accumulate empirical entropy, certify that that  $r$  is sufficiently rare.

**if**  $H \geq \lambda$  **then**

$r \leftarrow (x_1, \dots, x_i)$

**end**

**else**

        // Embed the watermark

$x_i \leftarrow \mathbb{1}[F_k(r, i) \leq p_i]$

**end**

$i \leftarrow i + 1$

**end**

---

---

## Algorithm 2 Detector $\text{Detect}_k$

---

**Data:** Text  $x_1, \dots, x_L$  and a secret key  $k$

**Result:** True or False

**for**  $i \in [L]$  **do**      Search over unknown start time of WM

$r^{(i)} \leftarrow (x_1, \dots, x_i)$

    Define  $v_j^{(i)} := x_j \cdot F_k(r^{(i)}, j) + (1 - x_j) \cdot (1 - F_k(r^{(i)}, j))$  for  $j \in [L]$

**if**  $\sum_{j=i+1}^L \ln(1/v_j^{(i)}) > (L - i) + \lambda\sqrt{L - i}$  **then**  
        | **return** True

**end**

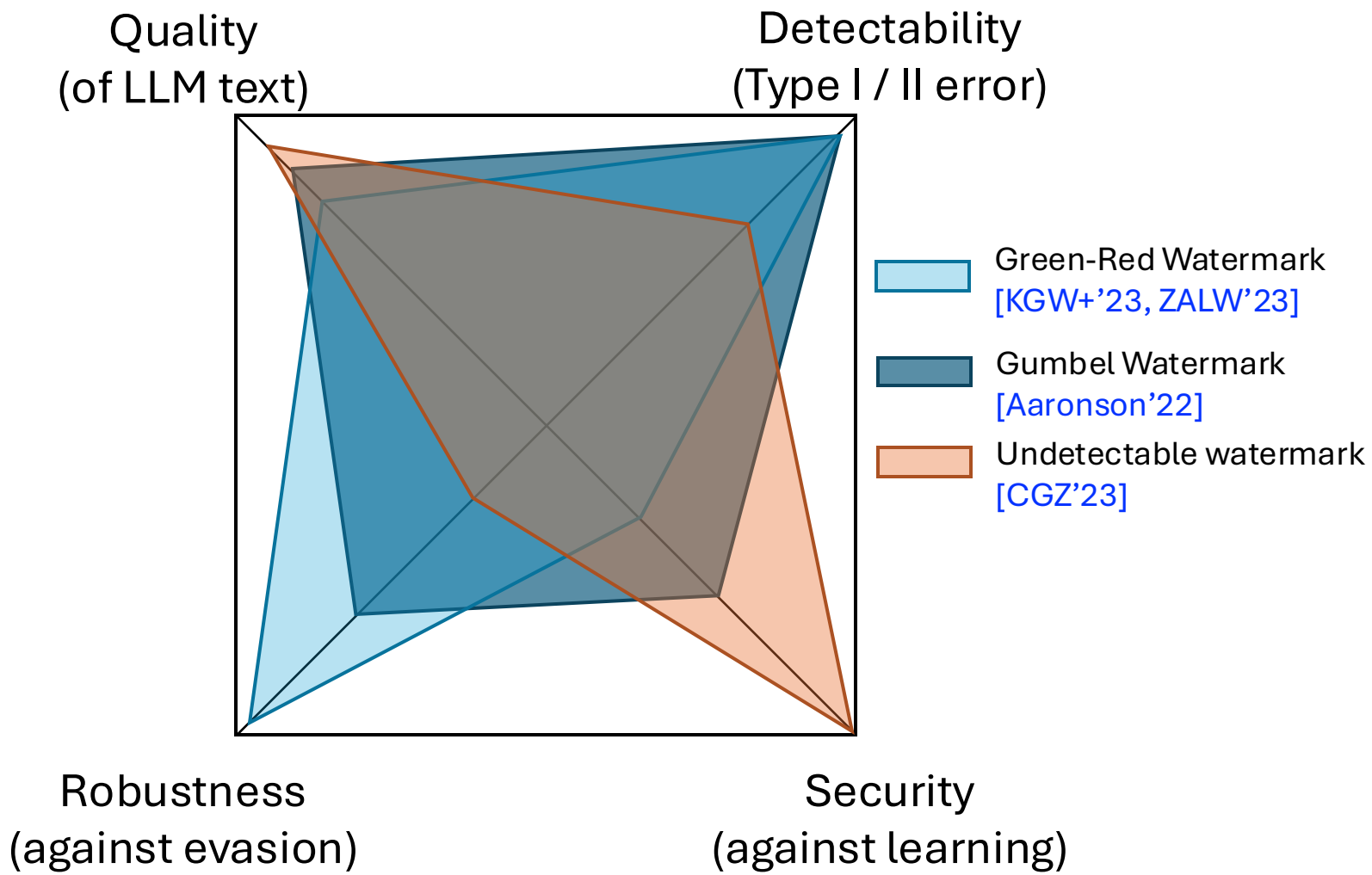
**end**

**return** False

---

What's the limitation of this approach?

- Robustness vs cropping and other forms of editing.



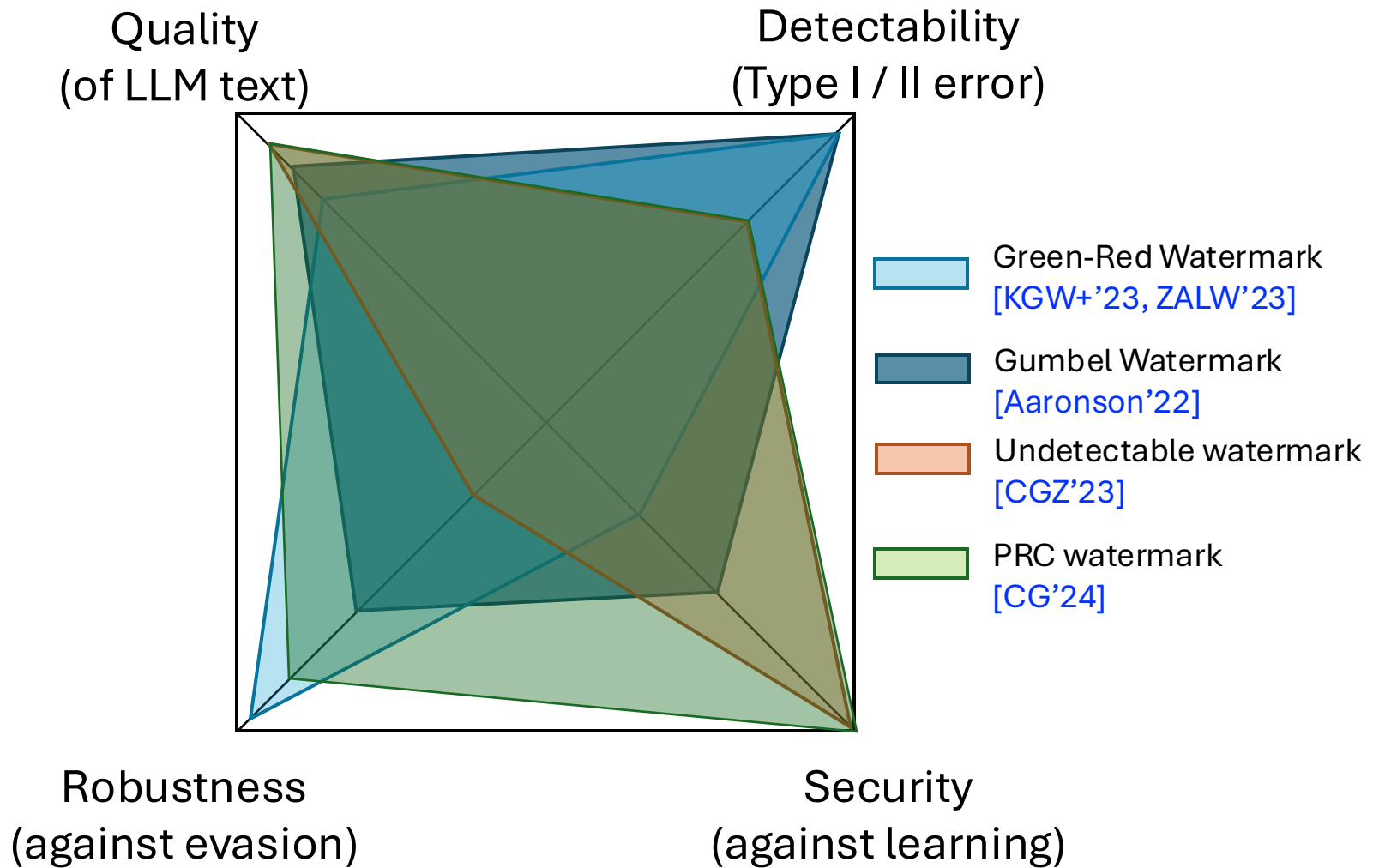
# Can we have “undetectability” and “robustness to edits” at the same time?

- Watermark by Pseudo-Random Error Correcting Code (Christ and Gunn, 2024)
  - Key: [PRC, and L messages  $a_i$  with length  $m$ ]

“ $W_1, \dots, W_m, \underbrace{W_{m+1}, \dots, W_{2m}}_{\text{Watermark with PRC.encode}(a_2)}, W_{2m+1}, \dots, W_{3m}, W_{3m+1}, \dots, W_{Lm}$ ”

Watermark with `PRC.encode( $a_2$ )`

- Detector: check for **PRC.decode(all subsequences)** against all messages.
- In some sense getting the robustness of “Unigram watermark” and “Undetectability” of CGZ simultaneously.



# Are “distortion-free” (and “undetectable”) watermarks always better than Green-Red?

- Green-Red watermark leverages the watermark strength parameter  $\delta$  and temperature  $T$ 
  - More detectable when entropy is lower.
  - Guarantee valid even if conditioning on the key --- not quite the case with Gumbel.
- Gumbel watermark responds only to temperature  $T$ 
  - Smaller temperature usually gives better perplexity.
  - Tradeoff between “greediness” vs “detectability”.

For a comprehensive empirical comparison. see [Piet et al 2023 “MarkMyWord”](https://arxiv.org/abs/2312.00273)  
<https://arxiv.org/abs/2312.00273>

# From Gumbel-Softmax trick to Exponential-PF trick

- Gumbel-Softmax trick (Gumbel, 1948)

$$y_t \sim \text{Softmax} \left( \frac{u_t(y)}{T} \right) \iff y_t = \arg \max_{y \in \mathcal{V}} \frac{u_t(y)}{T} + G_t(y)$$

$G_t(y) \sim \text{Gumbel}(0, 1) \text{ i.i.d}$

- Exponential-PF trick (Ding et. al, 2021)

$$y_t \sim \text{Permute\&Flip} \left( \frac{u_t(y)}{T} \right) \iff y_t = \arg \max_{y \in \mathcal{V}} \frac{u_t(y)}{T} + E_t(y).$$

$E_t(y) \sim \text{Exponential}(1) \text{ i.i.d.}$

**ReportNoisyMax from Differential Privacy.**

# Permute-and-Flip Watermark

- Gumbel-Watermark (Aaronson,

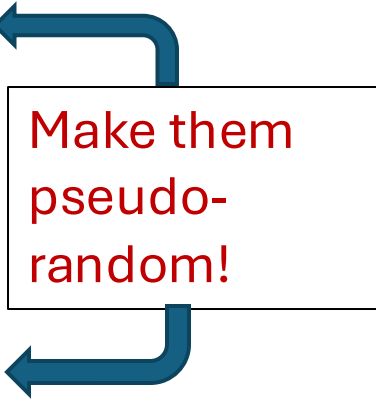
$$y_t = \arg \max_{y \in \mathcal{V}} \frac{u_t(y)}{T} + G_t(y)$$

$$G_t(y) \sim \text{Gumbel}(0, 1) \text{ i.i.d.}$$

$$y_t = \arg \max_{y \in \mathcal{V}} \frac{u_t(y)}{T} + E_t(y).$$

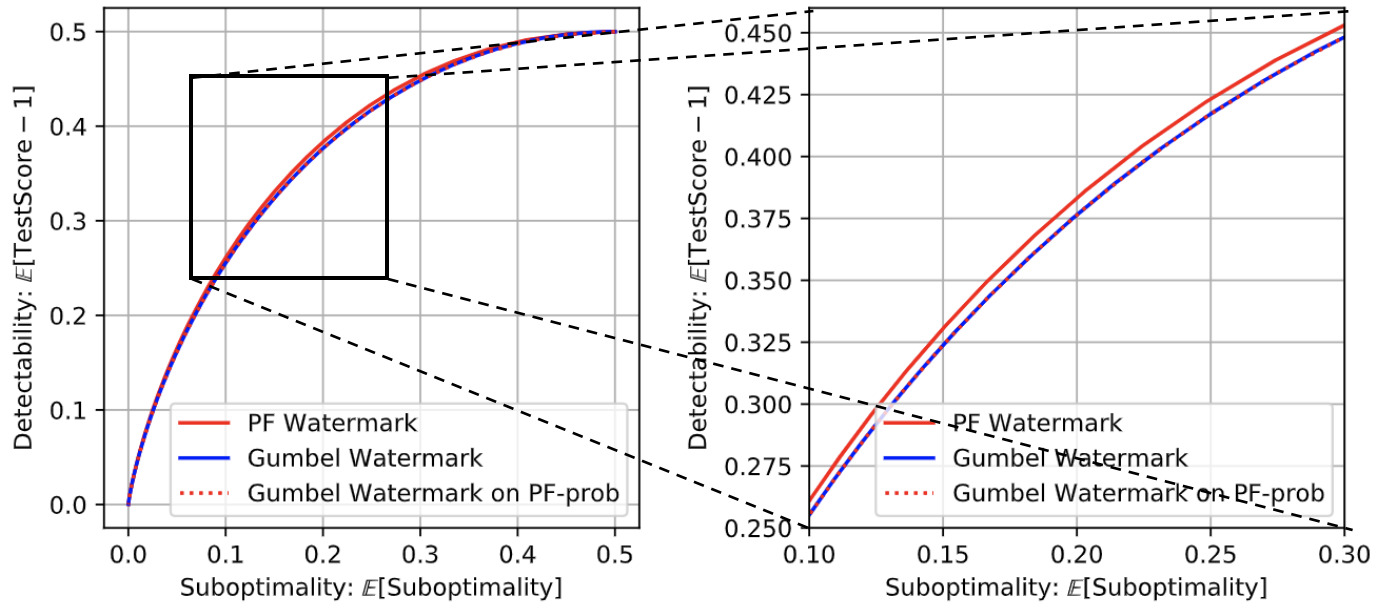
$$E_t(y) \sim \text{Exponential}(1) \text{ i.i.d.}$$

Make them  
pseudo-  
random!



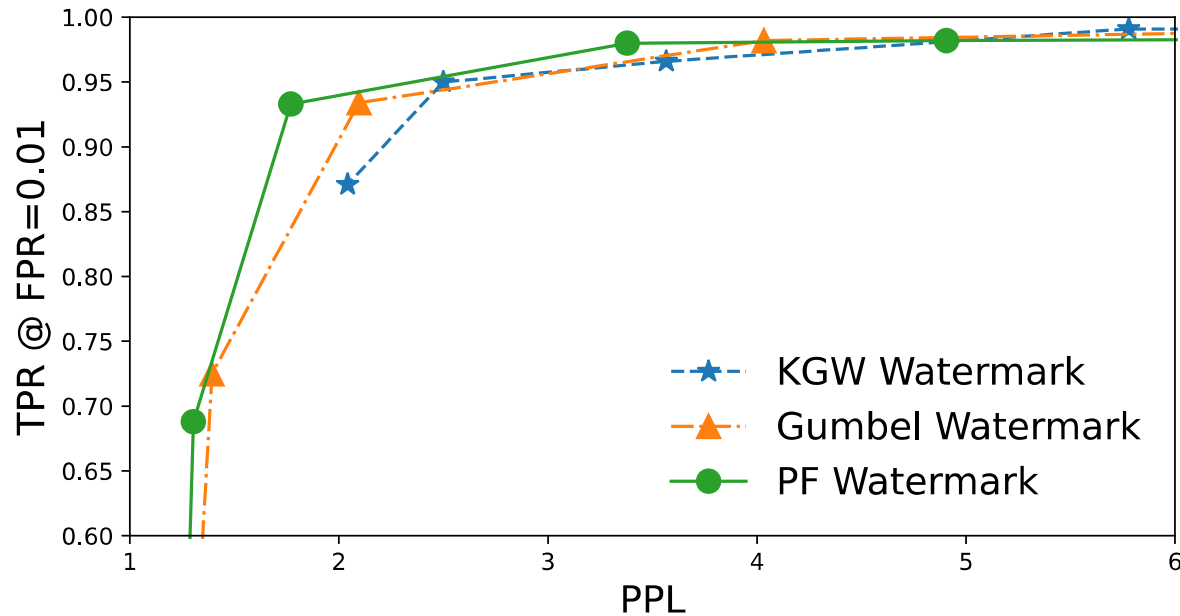
Zhao, Li, Wang. (2024) Permute-And-Flip: An Optimally Robust and Watermarkable Decoder for LLMs: <https://arxiv.org/abs/2402.05864>

# Plotting detectability against suboptimality as we adjust T



**PF has more favorable tradeoff curves than Gumbel**

# On real datasets: the PF watermark provides better Detectability-Perplexity Tradeoffs



# Checkpoint

	Quality	Detectability	Robustness	Security
Green-Red WM [KGW+ 2023]	$\frac{\delta^2}{8}$ KL (ex post)	$O(\delta)$ per-high-entropy token	Robust to edits	n.a.
Unigram Green-Red [ZALW 2023]	$\frac{\delta^2}{8}$ KL (ex post)	$O(\delta)$ per-high-entropy token	More robust than $m>1$	n.a.
Gumbel WM [Aaronson 2022]	0-ex ante <b>No ex post guarantee</b>	Shannon entropy of the token	Robust to edits	n.a.
PF Watermark [ZLW 2024]	Better PPL-detectability curve than Gumbel	A different kind of Entropy per token	Robust to edits	n.a.
Undetectable WM [CGZ 2023]	Undetectable <b>No ex post guarantee</b>	Empirical entropy of the token. (after a “burn-in”)	<b>Not robust to edits</b>	Strong security via “undetectability”
PRC WM [CG 2024]	Undetectable <b>No ex post guarantee</b>	Empirical entropy of the token. (after a “burn-in”)	Robust to edits	Strong security via “undetectability”

\* All are model-agnostic and efficient.

# References we discussed

## 1. Statistical watermarks

- Green-Red Watermark (Kirchenbauer et al, 2023)
- Unigram Green-Red watermark (Zhao, Ananth, Li, W. 2024)

No where near a complete set!

## 2. Cryptographic watermarks

- Gumbel watermark. (Aaronson, 2022)
- Undetectable WM (Christ, Gunn, Zamir 2023)
- Distortion-Free WM (Kuditipudi et al, 2023)
- Unbiased WM (Hu et al ,2023)
- Pseudorandom Code WM (Christ and Gunn 2023)
- Permute-and-Flip WM (Zhao, Li, W., 2024)

# Topics we did not get to cover

- Multi-bit LLM watermark
  - Yoo, Ahn and Kwak (2023), Qu, Yin, He et al. (2024)
- Semantic text watermark
  - Liu, Pan, Hu et al (ICLR-2024). Liu and Bu (ICML-2024).
- Public verifiable watermark
  - Fairoze et al. (2023). Publicly detectable watermarking for language models.
- Fragile watermark (deliberately non-robust for attribution/verification)
  - Jiang, Zhengyuan, et al. "Watermark-based Detection and Attribution of AI-Generated Content." arXiv preprint arXiv:2404.04254 (2024).
- Impossibility results
  - "Zhao et al (2023) "Invisible Image Watermarks..." Zhang, Barak et al. (2024) Watermarks in the Sand . Also work by Soheil Feizi et al and Furong Huang et al.

# Today: Continue with LLM Watermarking Schemes

- “Distortion-Free” watermarks
- “Undetectable” watermarks
- Attacks on watermarking schemes
  - “On the reliability of watermarks for LLM” by Varadraj Bartakke and Jay Sawant
  - “Watermarking in the Sand” by Benjamin TenWolde

# “On the reliability of watermarks for LLM” by Varadraj and Jay

arXiv > cs > arXiv:2306.04634

Search...

Help | Ad

Computer Science > Machine Learning

[Submitted on 7 Jun 2023 (v1), last revised 1 May 2024 (this version, v4)]

## On the Reliability of Watermarks for Large Language Models

[John Kirchenbauer](#), [Jonas Geiping](#), [Yuxin Wen](#), [Manli Shu](#), [Khalid Saifullah](#), [Kezhi Kong](#), [Kasun Fernando](#), [Aniruddha Saha](#), [Micah Goldblum](#), [Tom Goldstein](#)

As LLMs become commonplace, machine-generated text has the potential to flood the internet with spam, social media bots, and valueless content. Watermarking is a simple and effective strategy for mitigating such harms by enabling the detection and documentation of LLM-generated text. Yet a crucial question remains: How reliable is watermarking in realistic settings in the wild? There, watermarked text may be modified to suit a user's needs, or entirely rewritten to avoid detection. We study the robustness of watermarked text after it is re-written by humans, paraphrased by a non-watermarked LLM, or mixed into a longer hand-written document. **We find that watermarks remain detectable even after human and machine paraphrasing.** While these attacks dilute the strength of the watermark, paraphrases are statistically likely to leak  $n$ -grams or even longer fragments of the original text, resulting in high-confidence detections when enough tokens are observed. For example, after strong human paraphrasing the watermark is detectable after observing 800 tokens on average, when setting a  $1e-5$  false positive rate. We also consider a range of new detection schemes that are sensitive to short spans of watermarked text embedded inside a large document, and we compare the robustness of watermarking to other kinds of detectors.

# “Watermarking in the Sand” by Benjamin TenWolde

arXiv > cs > arXiv:2311.04378

Search...

Help | A

Computer Science > Machine Learning

[Submitted on 7 Nov 2023 (v1), last revised 27 May 2025 (this version, v5)]

## Watermarks in the Sand: Impossibility of Strong Watermarking for Generative Models

Hanlin Zhang, Benjamin L. Edelman, Danilo Francati, Daniele Venturi, Giuseppe Ateniese, Boaz Barak

Watermarking generative models consists of planting a statistical signal (watermark) in a model's output so that it can be later verified that the output was generated by the given model. A strong watermarking scheme satisfies the property that a computationally bounded attacker cannot erase the watermark without causing significant quality degradation. In this paper, we study the (im)possibility of strong watermarking schemes. We prove that, under well-specified and natural assumptions, strong watermarking is impossible to achieve. This holds even in the private detection algorithm setting, where the watermark insertion and detection algorithms share a secret key, unknown to the attacker. To prove this result, we introduce a generic efficient watermark attack; the attacker is not required to know the private key of the scheme or even which scheme is used. Our attack is based on two assumptions: (1) The attacker has access to a "quality oracle" that can evaluate whether a candidate output is a high-quality response to a prompt, and (2) The attacker has access to a "perturbation oracle" which can modify an output with a nontrivial probability of maintaining quality, and which induces an efficiently mixing random walk on high-quality outputs. We argue that both assumptions can be satisfied in practice by an attacker with weaker computational capabilities than the watermarked model itself, to which the attacker has only black-box access. Furthermore, our assumptions will likely only be easier to satisfy over time as models grow in capabilities and modalities. We demonstrate the feasibility of our attack by instantiating it to attack three existing watermarking schemes for large language models: Kirchenbauer et al. (2023), Kudityudi et al. (2023), and Zhao et al. (2023). The same attack successfully removes the watermarks planted by all three schemes, with only minor quality degradation.