Undetectable Watermarks for Language Models

Miranda Christ*
mchrist@cs.columbia.edu
Columbia University

Sam Gunn*,†
gunn@berkeley.edu
UC Berkeley

Or Zamir*
orzamir@princeton.edu
Princeton University

May 26, 2023

Abstract

Recent advances in the capabilities of large language models such as GPT-4 have spurred increasing concern about our ability to detect AI-generated text. Prior works have suggested methods of embedding watermarks in model outputs, by *noticeably* altering the output distribution. We ask: Is it possible to introduce a watermark without incurring *any detectable* change to the output distribution?

To this end we introduce a cryptographically-inspired notion of undetectable watermarks for language models. That is, watermarks can be detected only with the knowledge of a secret key; without the secret key, it is computationally intractable to distinguish watermarked outputs from those of the original model. In particular, it is impossible for a user to observe any degradation in the quality of the text. Crucially, watermarks should remain undetectable even when the user is allowed to adaptively query the model with arbitrarily chosen prompts. We construct undetectable watermarks based on the existence of one-way functions, a standard assumption in cryptography.

*Equal contribution.

†Supported by a Google PhD Fellowship.

Contents

1 Introduction

With the rise in the use of artificial models that churn out human-like text, there's also an increase in the potential for misuse. Imagine a student employing a language model to effortlessly write her "Machine Learning 101" homework or conjuring up tear-jerking emails to beg professors for easier exams. That's when the need arises to distinguish between texts penned by a language model and those crafted by human hands. The go-to method of employing a heuristic test to determine if a text was AI-generated, however, grows increasingly fragile as large language models (LLMs) advance. Even the cutting-edge detectors, like GPTZero [Tia23], can be outsmarted with cleverly crafted prompts.

Ultimately, as LLM outputs move closer to becoming identical to human-generated text, this approach becomes hopeless. It is already very hard to tell, for instance, that the previous paragraph was written by such a model. To overcome this problem, it is reasonable to consider intentionally modifying the model to embed *watermarks* into the text. Recent work of [KGW+23] introduced such watermarks in the context of LLMs. However, existing watermarking schemes come with a cost: To plant a useful watermark, the distribution of texts the model generates has to be noticeably changed. In fact, for existing schemes it is possible for the *user* to distinguish between outputs of the original model and of the watermarked one, and it is hence possible that the quality of text degrades.

We show how to plant watermarks with the following properties, stated informally, in any LLM.

1. (Undetectability) It is computationally infeasible to distinguish between the original and the watermarked models, even when the user is allowed to make many adaptive queries. In particular, the quality of generated text remains identical.

2. (Completeness) There is a secret key that enables efficient detection of responses from the watermarked model, as long as "enough randmoness" was used to generate the response. The detection works even when presented with only a contiguous sub-string from the response, and it doesn't require any other information.

3. (Soundness) Any text generated independently from the secret key has a negligible chance of being detected as watermarked.

We note that the existence of a secret key is necessary, as otherwise Properties (1) and (2) would directly contradict each other. However, in practice the secret key can be published if one wishes; Property (1) still ensures that the quality of the text is imperceptibly changed for all uses not involving the secret key.

An important aspect of our construction is that Properties (1) and (3) will *always* hold, for any LLM with any choice of parameters, and without making any assumptions on the text. Our scheme is the first to have these properties, and we argue that they are completely crucial. First, the creator of a state-of-the-art LLM is unlikely to intentionally degrade the quality of their model, making Property (1) necessary for any practical watermarking scheme. As the quality and versatility of LLMs has reached such high levels, any noticeable change due to the watermark is liable to have adverse side-effects in some situations. Second, falsely accusing humans of using LLMs to generate their texts should be completely unacceptable. When heuristics are used for detection, this will always be a possibility — indeed, instances of such false accusations against students have already made news headlines [Fow23, Jim23], and concerningly, false accusations are more common for

non-native English writers [LYM+23]. Property (3) in our construction rigorously guarantees that natural text will not be detected as watermarked.

Of course, a watermark is only useful if it can be detected with the secret key. If the model has a deterministic response to some prompt, then we should not be able to embed the watermark in that response (as any change to the output would necessarily be detectable). For Property (2), we therefore need to assume that enough "randomness" was used in the generation of the specific text we are considering. We introduce a formal notion that we call *empirical entropy*, and show that this condition is necessary. Our detection algorithm works when it is given text containing any consecutive sub-string with enough empirical entropy from an output of the model.

Primary contributions of this work include the formal definition and construction of *undetectable watermarks*, and the notion of *empirical entropy* that quantifies the randomness used in the generation of a specific output. These definitions and our results appear in Section 2.

1.1 Related Work

Approaches for detecting AI-generated text largely fall into two categories. *Watermarking schemes* alter the output of a language model in a way that a corresponding detection algorithm can identify. *Post-hoc detectors* leave the output of the model unchanged and instead identify AI-generated text using existing differences between natural language and the model's output.

Post-hoc detectors. The simplest post-hoc detectors use natural heuristics to distinguish between human- and AI-generated text. These heuristics include relative entropy scoring [LUY08], perplexity [Ber16], and other statistical methods [GSR19]; see [Ber16] for a survey of such methods. Other post-hoc detectors (e.g., [ZHR+19, MLK+23, Tia23, KAAL23]) are themselves models, specifically trained for this binary classification task. Unfortunately, these heuristic and model-based methods lack formal guarantees, and it's possible to train a model to transform AI-generated text in a way that evades them; see, e.g., [KSK+23, SKB+23]. For example, [KSK+23] trains a model to paraphrase text output by language models, fooling common post-hoc detectors such as GPTZero [Tia23], DetectGPT [MLK+23], and the detector developed by OpenAI [KAAL23]. Furthermore, simple tricks such as instructing the model in the prompt to write a response that evades a detector, or varying the model's parameters (e.g., increasing the temperature and frequency/presence penalties for GPT-4), fool [Tia23]. [CBZ+23] prove that as AI-generated text more closely resembles natural text, post-hoc detectors will need longer text samples.

See [JAML20] for more comprehensive background on post-hoc detection of AI-generated text and attacks.

Language watermarking schemes. Several schemes (e.g., [AF21, QZL+23, YAJK23, MZ23]) involve using an ML model in the watermarking algorithm itself. [AF21, MZ23] work by taking a passage of text and using a model to produce a semantically similar altered passage. By nature of using machine learning, these constructions have no formal guarantees and rely on heuristic arguments for undetectability, completeness, soundness.

In a recent work, [KGW+23] presented the first watermarking scheme for LLMs with any formal guarantees. They showed that a watermark can be planted in outputs with large enough entropy (with a definition different than ours, yet morally similar). However, their watermarking scheme crucially *changes* the distribution of generated texts and uses this change to detect the watermark. They bound the difference between the original distribution and the distribution of their

watermarked model, using a quantity called *perplexity*. In contrast, in our work the original and watermarked output distributions are completely indistinguishable.

The authors of this paper are also aware of an ongoing watermarking project of [Aar22], which he mentions in his blog. It appears that in this project, the guarantee is that the two distributions will be indistinguishable, but only as long as no two output texts are seen that share a common substring of a certain length. In contrast, our construction guarantees undetectability without any assumption on the texts or the model. In particular we allow the distinguisher to make adaptive queries with arbitrary prompts, so it may force the model to return outputs that share long parts with each other.

Our scheme, as well as those of [KGW+23] and [Aar22] are vulnerable to simple attacks such as the "emoji attack"[1] discussed in [KGW+23]. We discuss attacks on watermarking schemes further in Section 6.1.

Steganography. Steganography is the study of encoding a hidden message into a given channel (e.g., natural language or an image) such that a recipient possessing a key can read the message but an eavesdropping adversary cannot determine whether a message is present. [HvAL09] defines security of a steganography scheme against a chosen hiddentext attack (CHA) as the requirement that an adversary cannot determine whether a given oracle is for the original distribution or the distribution embedded with a hiddentext. Porting this definition over to the watermark setting, one would obtain undetectability. However, the corresponding steganography schemes we are aware of would not obtain undetectability *without making assumptions about the entropy of the channel*. Crucially, in our setting where prompts for the language model are adversarially chosen, we want to retain undetectability even if the adversary submits a prompt with a deterministic response.

Part of the reason for this difference stems from the access that the watermark and encoding algorithms have to the channel (or the distribution of the language model's output). In steganography, the encoding algorithm has only oracle access to the channel. In our language model setting, the watermark algorithm receives from the model a full description of the probability distribution p_i for each token. Because of this limited access, steganography schemes largely either rely on assumptions about the entropy of the channel (e.g., [HvAL09]) or lose security when the channel has low entropy (e.g., [DIRR09]). Our watermark is undetectable *regardless* of the entropy of the text. We achieve this guarantee exactly by using the watermark algorithm's access to the distributions p_i. Using this knowledge, the algorithm is able to compute the *empirical entropy* of its output thus far. Once the empirical entropy is sufficiently high, it uses the output as a random seed for a PRF used to embed the watermark in subsequent tokens. Importantly, our algorithm alters the output distribution only once it has collected enough entropy; in the steganography schemes, the encoding algorithm with only oracle access does not know when this has happened.

This complete knowledge of each token distribution p_i separates the problem of watermarking language models from watermarking more generally. One prior steganography scheme for language models, [KJGR21], operates in our regime where the encoding algorithm has access to each p_i. However, it assumes that the decoding algorithm has access to each p_i as well. This is unrealistic for watermarking, as the probability distributions p_i for the output of the model on some prompt depend on that prompt. A watermark detection algorithm, which receives only the output and not the prompt, does not know the distributions p_i. In practice, it would be unrealistic to assume that

[1] `https://twitter.com/goodside/status/1610682909647671306`

a detector trying to determine whether an essay was generated by a language model would also be given the prompt used to generate it.

Watermarking. The field of digital watermarking focuses on the problem of covertly planting a signal in a medium (e.g., an image or text) such that it can be detected by an algorithm. [ARC+01, ARH+03] present some of the first watermarking schemes for NLP-generated text, though their schemes rely on a syntactic tree structure that was present in NLP models at that time but no longer used today. [HMW07] formally defines watermarking and desired properties, though these definitions are not tailored to language models.

Other related work. Recently, [GKVZ22] used a cryptographic construction to embed undetectable backdoors into neural networks. In both their work and ours, cryptographic notions of indistinguishability are used in the context of machine-learning, rather than empirical notions.

1.2 Organization of the Paper

In Section 2 we formally define our model, introduce the notions of empirical entropy and undetectable watermarks, and outline our results. In Section 3 we sketch a simple watermarking scheme that achieves undetectability, but falls short of our main scheme in other respects. In Section 4 we construct our undetectable watermarks with strong completeness and soundness guarantees. We include an overview of these constructions in Section 4.2. In Section 5 we discuss the necessity of the assumptions we make. In Section 6 we discuss possible methods of removing watermarks from texts. In particular, we prove that it is impossible to create undetectable watermarks that are completely unremovable, under certain assumptions. In Section 7 we summarize and discuss open problems.

2 Modeling the Problem

2.1 Preliminaries

Notation. Let λ denote the security parameter. A function f of λ is *negligible* if $f(\lambda) \in O(\frac{1}{\mathsf{poly}(\lambda)})$ for every polynomial $\mathsf{poly}(\cdot)$. We write $f(\lambda) \leq \mathsf{negl}(\lambda)$ to mean that f is negligible. For a vector or sequence of tokens $s = (s_1, \ldots, s_{|s|})$ and positive integers $b \geq a$, let $s[a:b]$ denote $(s_a, \ldots, s_b)$. We use $\log(x)$ to denote the logarithm base 2 of x, and $\ln(x)$ to denote the natural logarithm of x. For integer $n > 0$, we define $[n] := \{1, \ldots, n\}$. For integers $n \geq k > 0$, we define $[k, n] := \{k, \ldots, n\}$.

Pseudorandom function (PRF). Let $\mathcal{F} = \{F_{\mathsf{sk}} : \{0,1\}^{\ell_1(\lambda)} \to \{0,1\}^{\ell_2(\lambda)} | \mathsf{sk} \in \{0,1\}^\lambda\}$ be a family of functions. $\mathcal{F}$ is a PRF if F_{sk} is efficiently computable and for all probabilistic polynomial-time distinguishers D,

$$\left| \Pr_{\mathsf{sk} \leftarrow \{0,1\}^\lambda} \left[D^{F_{\mathsf{sk}}(\cdot)}(1^\lambda) = 1 \right] - \Pr_{f} \left[D^{f(\cdot)}(1^\lambda) = 1 \right] \right| \leq \mathsf{negl}(\lambda)$$

where f denotes a random function from $\{0,1\}^{\ell_1(\lambda)}$ to $\{0,1\}^{\ell_2(\lambda)}$. PRFs are a standard cryptographic primitive equivalent to one-way functions and can be constructed from standard assumptions [GGM86, HILL99].

2.2 Language Models

We loosely follow [KGW+23] in our definition of a *language model*. We will often refer to language models simply as *models*.

Definition 1. *A language model* Model *over token set* $\mathcal{T}$ *is a deterministic algorithm that takes as input a prompt* PROMPT *and tokens previously output by the model* $x = (x_1, \ldots, x_{i-1})$*, and outputs a probability distribution* $p_i = \mathsf{Model}(\text{PROMPT}, x)$ *over* $\mathcal{T}$*.*

A language model Model is used to generate text as a response to a prompt by iteratively sampling from the returned distribution until a special terminating token $\texttt{done} \in \mathcal{T}$ is drawn.

Definition 2. *A language model's* response *to* PROMPT *is a random variable* $\overline{\mathsf{Model}}(\text{PROMPT}) \in \mathcal{T}^\star$ *that is defined algorithmically as follows. We begin with an empty list of tokens* $x = ()$*. As long as the last token in* x *is not* `done`*, we draw a token* x_i *from the distribution* $\mathsf{Model}(\text{PROMPT}, x)$ *and append it to* x*. Finally, we set* $\overline{\mathsf{Model}}(\text{PROMPT}) = x$*.*

Throughout the text we will make use of a security parameter λ. We will assume that our model never outputs text of length super-polynomial in λ. (For OpenAI's language models, there is actually a fixed limit to the length of generated text.)

2.3 Entropy and Empirical Entropy

Let $\log(x)$ denote the logarithm base 2 of x. For a probability distribution D over elements of a finite set X, we define the Shannon *entropy* of D as

$$H(D) = \mathop{\mathbb{E}}_{x \sim D}[-\log D(x)],$$

where $D(x)$ is the probability of x in the distribution D. The *empirical entropy* of x in D is simply $-\log D(x)$. The expected empirical entropy of $x \sim D$ is exactly $H(D)$. Intuitively, the empirical entropy of x (with respect to D) is the number of random bits that were required to draw x out of the distribution D. The entropy $H(D)$ is thus the expected number of random bits needed to draw an element out of the distribution D.

We thus define the empirical entropy of a model's response as follows.

Definition 3. *For a language model* Model*, a prompt* PROMPT*, and a possible response* $x \in \mathcal{T}^\star$*, we define the* empirical entropy *of* $\overline{\mathsf{Model}}$ *responding with* x *to* PROMPT *as*

$$H_e(\mathsf{Model}, \text{PROMPT}, x) := -\log \Pr\left[\overline{\mathsf{Model}}\,(\text{PROMPT}) = x\right].$$

We next generalize the definition of empirical entropy from whole outputs to *substrings* out of a model's output. This will quantify how much entropy was involved in the generation of a particular contiguous substring of the output.

Definition 4. *For a language model* Model*, a prompt* PROMPT*, a possible response* $x \in \mathcal{T}^\star$*, and indices* $i, j \in [|x|]$ *with* $i \le j$ *we define the* empirical entropy on substring $[i, j]$ *of* $\overline{\mathsf{Model}}$ *responding with* x *to* PROMPT *as*

$$H_e^{[i,j]}(\mathsf{Model}, \text{PROMPT}, x) := -\log \Pr\Big[\overline{\mathsf{Model}}\,(\text{PROMPT})\,[i:j] = x[i:j] \mid \overline{\mathsf{Model}}\,(\text{PROMPT})\,[1:(i-1)] = x[1:(i-1)]\Big].$$

We sometimes write $H_e^i := H_e^{[i,i]}$ to denote the empirical entropy of a single token i. We remark that in expectation, Definition 3 simply captures the entropy in the response generation. That is, we have

$$\mathop{\mathbb{E}}_{x}\left[H_e(\mathsf{Model}, \textsc{prompt}, x)\right] = H\left(\overline{\mathsf{Model}}\left(\textsc{prompt}\right)\right),$$

where $x \sim \overline{\mathsf{Model}}\left(\textsc{prompt}\right)$.

2.4 Watermarks

We formally define a watermarking scheme as follows.

Definition 5. *A* watermarking scheme *for a model* Model *over* $\mathcal{T}$ *is a tuple of algorithms* $\mathcal{W} = (\mathsf{Setup}, \mathsf{Wat}, \mathsf{Detect})$ *where:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{sk}$ *outputs a secret key, with respect to a security parameter* λ*.*
- $\mathsf{Wat}_{\mathsf{sk}}(\textsc{prompt})$ *is a randomized algorithm that takes as input a prompt* PROMPT *and generates a response in* $\mathcal{T}^\star$*.*
- $\mathsf{Detect}_{\mathsf{sk}}(x) \to \{\mathsf{true}, \mathsf{false}\}$ *is an algorithm that takes as input a sequence* $x \in \mathcal{T}^\star$ *outputs* true *or* false*.*

Ideally, $\mathsf{Detect}_{\mathsf{sk}}(x)$ should output true if x is generated by $\mathsf{Wat}_{\mathsf{sk}}(\textsc{prompt})$, and should output false if x is generated independently of sk. The former property is called *completeness* and the latter *soundness*.

Definition 6. *A watermarking scheme* $\mathcal{W}$ *is* sound *if for every security parameter* λ *and token sequence* $x \in \mathcal{T}^\star$ *of length* $|x| \leq \mathsf{poly}(\lambda)$*,*

$$\Pr_{\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda)}[\mathsf{Detect}_{\mathsf{sk}}(x) = \mathsf{true}] \leq \mathsf{negl}(\lambda).$$

A scheme is sound if *any* text that is generated independently from sk has negligible probability of being detected as watermarked by $\mathsf{Detect}_{\mathsf{sk}}$. Essentially, this means we will *never* see a false-positive detection.

Defining completeness requires care: It is not reasonable to require $\mathsf{Detect}_{\mathsf{sk}}$ to detect any sequence x generated by $\mathsf{Wat}_{\mathsf{sk}}(\textsc{prompt})$ for some PROMPT, as it is possible that x is very short, or that $\overline{\mathsf{Model}}(\textsc{prompt})$ is deterministic or has very low entropy. Instead, we require $\mathsf{Detect}_{\mathsf{sk}}$ to detect watermarks only in responses for which the entropy in the generation process is high enough.

Definition 7. *A watermarking scheme* $\mathcal{W}$ *is* $b(L)$-complete *if for every security parameter* λ *and prompt* PROMPT *of length* $|\textsc{prompt}| \leq \mathsf{poly}(\lambda)$*,*

$$\Pr_{\substack{\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda) \\ x \leftarrow \mathsf{Wat}_{\mathsf{sk}}(\textsc{prompt})}}\left[\mathsf{Detect}_{\mathsf{sk}}(x) = \mathsf{false} \text{ and } H_e\left(\mathsf{Model}, \textsc{prompt}, x\right) \geq b\left(|x|\right)\right] \leq \mathsf{negl}(\lambda).$$

Definition 7 guarantees that any output generated by $\mathsf{Wat}_{\mathsf{sk}}$ with empirical entropy at least $b(L)$, where L is the length of the output, will be detected as watermarked with high probability. Essentially, this means we will *never* see a false-negative detection on any output of high enough empirical

entropy. In Section 5 we show that it is necessary to consider the empirical entropy of the specific output rather than the standard entropy of the entire model.

We also generalize Definition 7 to capture contiguous substrings of outputs. That is, we should be able to detect a watermarked output of $\mathsf{Wat}_{\mathsf{sk}}$ even if $\mathsf{Detect}_{\mathsf{sk}}$ is only given a long enough contiguous substring from it.

Definition 8. *A watermarking scheme $\mathcal{W}$ is* $b(L)$-substring-complete *if for every prompt* PROMPT *and security parameter λ,*

$$\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)\\ x\leftarrow\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})}}\Big[\,\exists\, i, L\in[|x|] \text{ such that } \mathsf{Detect}_{\mathsf{sk}}(x[i:i+L])=\mathsf{false} \text{ and } H_e^{[i:i+L]}(\mathsf{Model},\text{PROMPT},x)\geq b(L)\,\Big]\leq \mathsf{negl}(\lambda).$$

This means that every contiguous part of an output of the watermarking procedure, that has high enough empirical entropy, is detected as watermarked with high probability. We stress that the empirical entropies in Definitions 7 and 8 are defined with respect to the original model Model, without reference to the watermarking procedure $\mathsf{Wat}_{\mathsf{sk}}$. We also note that the empirical entropy $H_e(\mathsf{Model},\text{PROMPT},x)$ is only used as part of the definition, and is not necessarily known to $\mathsf{Detect}_{\mathsf{sk}}$. It is in general not possible to compute $H_e(\mathsf{Model},\text{PROMPT},x)$ without knowledge of PROMPT.

2.5 Undetectable Watermarks

Finally, we define the notion of computationally undetectable watermarking schemes. Intuitively, a scheme is undetectable if it is infeasible to distinguish between the distributions of $\overline{\mathsf{Model}}$ and $\mathsf{Wat}_{\mathsf{sk}}$, even when those can be queried adaptively with arbitrary prompts.

Definition 9. *A watermarking scheme $\mathcal{W}=(\mathsf{Setup},\mathsf{Wat},\mathsf{Detect})$ is* undetectable *if for every security parameter λ and all polynomial-time distinguishers D,*

$$\left|\Pr\left[D^{\mathsf{Model},\overline{\mathsf{Model}}}(1^\lambda)\to 1\right]-\Pr_{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)}[D^{\mathsf{Model},\mathsf{Wat}_{\mathsf{sk}}}(1^\lambda)\to 1]\right|\leq\mathsf{negl}(\lambda),$$

where the notation $D^{\mathcal{O}_1,\mathcal{O}_2}$ means that D is allowed to adaptively query both $\mathcal{O}_1$ and $\mathcal{O}_2$ with arbitrary prompts.

Note that in the above definition, we allow the distinguisher access to Model itself as well as $\overline{\mathsf{Model}}$ or $\mathsf{Wat}_{\mathsf{sk}}$. The only thing that is kept secret from the distinguisher is the secret key.

It is important to remark that in any undetectable watermarking scheme, the *quality of outputs* must be identical between $\overline{\mathsf{Model}}$ and $\mathsf{Wat}_{\mathsf{sk}}$, as otherwise it would be possible to distinguish between them. In particular, embedding the watermark does not degrade the quality of the generated text *at all*.

We finally note that a watermarking scheme can be made *public* by publishing the secret key sk. Then, everyone can run the detection algorithm $\mathsf{Detect}_{\mathsf{sk}}$. In particular, the scheme is no longer undetectable as $\mathsf{Detect}_{\mathsf{sk}}$ can be used to distinguish between $\overline{\mathsf{Model}}$ and $\mathsf{Wat}_{\mathsf{sk}}$. Nevertheless, we still maintain the property that there is no degradation in the quality of watermarked outputs, as long as the definition of "quality" does not depend on the secret key sk.

2.6 Statement of our Theorems

We are now ready to formally state the guarantees of the watermarking schemes that we present. To warm up, in Section 3 we give a simple construction of an $O(\lambda)$-complete scheme, but it only achieves a much weaker notion of soundness, and the watermarking algorithm runs in *expected* $\mathsf{poly}(\lambda)$ time rather than strict $\mathsf{poly}(\lambda)$ time. In Section 4.3, we prove the following theorem by introducing an efficient watermarking scheme with Algorithms 3 and 4.

Theorem 1. *For any model* Model *we construct a watermarking scheme $\mathcal{W}$ that is undetectable, sound, and $O(\lambda\sqrt{L})$-complete.*

This means that our watermarking scheme is *always* undetectable and sound, and is also complete as long as there is enough empirical entropy in the model's response.

In Section 5 we show that it is *necessary* for the completeness parameter $b(L)$ to be reasonably large, with respect to λ. In fact, we show that it is *inherent* that low empirical entropy outputs are not watermarked in any undetectable watermarking scheme for any model.

To strengthen Theorem 1, we also present a modified scheme (Algorithms 5 and 6) in Section 4.4 which obtains substring completeness, with similar parameters.

Theorem 2. *For any model* Model *we construct a watermarking scheme $\mathcal{W}$ that is undetectable, sound, and $O(\lambda\sqrt{L})$-substring-complete.*

In Section 6 we discuss known attacks on methods of detecting AI-generated text. We also prove that any undetectable watermarking scheme is removable, if the model enables a fairly strong form of query access and if one is willing to expend a number of queries that grows linearly with the size of generated text. Finally, in Section 7 we pose some open problems related to this work.

3 Simplified Construction

In this section we describe a simple construction of an undetectable watermarking scheme that achieves $\Theta(\lambda)$-completeness. However, this scheme falls short of our main constructions in Section 4 in two important ways. First, it has a false-positive rate of $\varepsilon = 1/\mathsf{poly}(\lambda)$ instead of $\mathsf{negl}(\lambda)$. We call such a scheme ε-weakly-sound. Second, the watermarking procedure $\mathsf{Wat}_{\mathsf{sk}}$ is not very efficient: Its *expected* run-time is polynomial in the length of the output and in $\frac{1}{\varepsilon}$, but the worst-case running time of $\mathsf{Wat}_{\mathsf{sk}}$ is unbounded.

Later, in Section 4 we present our main construction which is undetectable, sound, complete (in fact, it is even substring-complete) and efficient, yet achieves suboptimal completeness. Bridging this gap is an interesting open problem discussed in Section 7.

Let $b \in \mathbb{N}$ be a parameter to be chosen later; the *rate of false positives* will be 2^{-b}. First, let's assume that after the initialization, both $\mathsf{Wat}^{\mathcal{O}}$ and $\mathsf{Detect}^{\mathcal{O}}$ have access to a random oracle $\mathcal{O}$. This is a truly random function; that is, whenever $\mathcal{O}$ is called with a new input it returns a uniformly random string in $\{0,1\}^b$, and it returns a consistent answer when queried with a previously queried input.

3.1 Watermarks assuming random oracle and high min-entropy

We begin by adding another strong assumption, that the min-entropy of the response to any prompt is at least 5λ. Equivalently, let PROMPT be a prompt and assume that for every x, we

have $H_e(\mathsf{Model}, \text{PROMPT}, x) \geq 5\lambda$. We define the watermarking scheme as follows. The distribution of $\mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})$ is defined as the distribution of $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ conditioned on $\mathcal{O}(x) = 0^b$. Equivalently, $\mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})$ repeatedly draws outputs from $\overline{\mathsf{Model}}(\text{PROMPT})$ until the first time it gets a response x for which $\mathcal{O}(x) = 0^b$, and then it returns x. Note that this requires 2^b calls to $\overline{\mathsf{Model}}(\text{PROMPT})$ in expectation. To detect whether a string x is watermarked, $\mathsf{Detect}^{\mathcal{O}}$ simply checks whether $\mathcal{O}(x) = 0^b$. We assume that $b \leq \lambda$ and sketch a proof for the above scheme being *weakly sound*, complete and undetectable.

Weakly Sound: For any string x, the value of $\mathcal{O}(x)$ is truly random by the assumption. Thus, it is detected as watermarked with probability 2^{-b}.

Complete: By definition, $\mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})$ only produces outputs x such that $\mathcal{O}(x) = 0^b$, which are detected as watermarked by $\mathsf{Detect}^{\mathcal{O}}$.

Undetectable: Let D be a distinguisher that can query $\mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})$ at most 2^λ times. In expectation, the total number of times $\mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})$ queries $\overline{\mathsf{Model}}(\text{PROMPT})$ to answer all queries is at most $2^\lambda \cdot 2^b \leq 2^{2\lambda}$ (since $b \leq \lambda$ by assumption). As the probability of each output x to be output by $\overline{\mathsf{Model}}(\text{PROMPT})$ is at most $2^{-5\lambda}$, and as $2^{2\lambda} \ll \sqrt{2^{5\lambda}}$, with high probability $\mathcal{O}$ is never queried twice on the same input. If $\mathcal{O}$ is never queried on the same input twice, then it simply outputs an independent random value for each query. In particular, the process of repeatedly sampling $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ until $\mathcal{O}(x) = 0^b$ is equivalent to repeatedly sampling $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ until an independent, fresh random string is 0^b — which is identical to simply sampling $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$.

3.2 Removing the high min-entropy assumption

We next define a scheme that no longer requires the assumption about the min-entropy of the model. We do so by only watermarking outputs with empirical entropy higher than 6λ, corresponding to the definition of (6λ)-complete schemes.

Let PROMPT be any prompt and consider the probability

$$p := \Pr_{x \leftarrow \overline{\mathsf{Model}}(\text{PROMPT})} \left[H_e\left(\mathsf{Model}, \text{PROMPT}, x\right) > 6\lambda \right].$$

Let $M_{\leq}$ be the distribution $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ conditioned on $H_e\left(\mathsf{Model}, \text{PROMPT}, x\right) \leq 6\lambda$. Denote by $M_{>}$ the distribution $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ conditioned on $H_e\left(\mathsf{Model}, \text{PROMPT}, x\right) > 6\lambda$. Drawing $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ is equivalent to the following process: With probability p, we draw a string out of the distribution $M_{>}$, otherwise, we draw a string out of $M_{\leq}$.

Therefore, we consider the following natural algorithm for $\mathsf{Wat}^{\mathcal{O}}$. We first flip a biased coin $c \sim \text{Bernoulli}(p)$. If $c = 0$ then we draw an output from the distribution $M_{\leq}$. If $c = 1$, then we apply the algorithm of Section 3.1 — that is, we draw an output from the distribution $x \sim M_{>}$ conditioned on $\mathcal{O}(x) = 0^b$.

This scheme is again weakly sound, and it is complete for every output x with empirical entropy at least 6λ because these outputs will always satisfy $\mathcal{O}(x) = 0^b$. If $p \leq 2^{-\lambda}$, then undetectability is straightforward: With all but negligible probability, the distinguisher will only see outputs from $M_{\leq}$, which we did not change. Otherwise, $p > 2^{-\lambda}$ and thus the distribution $M_{>}$ is of min-entropy at least $6\lambda - \lambda = 5\lambda$. In particular, the construction of Section 3.1 applied to $M_{>}$ is guaranteed to be undetectable.

We finally note that it is intractable to compute the value of p or the conditional distributions $M_{\leq}, M_{>}$. We avoid their explicit computation as follows. To implement $\mathsf{Wat}^{\mathcal{O}}$, we first draw a string $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$. Computing the empirical entropy $H_e(\mathsf{Model}, \text{PROMPT}, x)$ given $\mathsf{Model}, \text{PROMPT}, x$ is straightforward. The probability that $H_e(\mathsf{Model}, \text{PROMPT}, x) > 6\lambda$ is of course exactly p. Thus, we can check if $H_e(\mathsf{Model}, \text{PROMPT}, x) \leq 6\lambda$. If so, we simply output x. Otherwise, we need to sample a response from $M_{>}$ conditioned on its output under $\mathcal{O}$ being 0^b. We can do so by repeatedly sampling $x \sim \overline{\mathsf{Model}}(\text{PROMPT})$ until both $H_e(\mathsf{Model}, \text{PROMPT}, x) > 6\lambda$ and $\mathcal{O}(x) = 0^b$. Note that the probability of success is now $p \cdot 2^{-b}$, and thus in expectation $\frac{1}{p}2^b$ tries are needed, which may be very large if p is small. On the other hand, we only reach this loop with probability p; hence, the expected number of queries from $\overline{\mathsf{Model}}(\text{PROMPT})$ our algorithm makes is $1 + p \cdot \frac{1}{p}2^b = 1 + 2^b$.

3.3 Removing the random oracle assumption

The construction presented so far uses a random oracle $\mathcal{O}$, which is impossible to implement.[2] Often, as we also do later in Section 4, a random oracle can be replaced with a cryptographic pseudorandom function (PRF, defined in Section 2.1). However, the inefficiency of $\mathsf{Wat}^{\mathcal{O}}$ requires being careful about this switch.

A PRF with a security parameter λ requires memory and runtime $\mathsf{poly}(\lambda)$ and is guaranteed to be indistinguishable from a random oracle only to distinguishers that run in time $\mathsf{poly}(\lambda)$ as well. As $\mathsf{Wat}^{\mathcal{O}}$ runs in (expected) time 2^b, we must choose $b = O(\log \lambda)$ for the PRF to behave as a random oracle. This implies that the soundness is no longer $\mathsf{negl}(\lambda)$, but is at least $\frac{1}{\mathsf{poly}(\lambda)}$.

Pseudo-code for this simplified scheme is presented in Algorithms 1 and 2. We state the properties of this scheme in Theorem 3 without proof; the proofs of these properties were sketched in the preceding two sections.

Theorem 3. *For any λ, Model and $b \leq O(\log \lambda)$, Algorithms 1 and 2 are a watermarking scheme $\mathcal{W}$ that is undetectable, (6λ)-complete, and 2^{-b}-weakly-sound. On expectation, $\mathsf{Wat}_{\mathsf{sk}}$ makes $1+2^b$ calls to $\overline{\mathsf{Model}}$ to generate each response.*

Algorithm 1: Weakly-sound watermarking algorithm $\mathsf{Wat}_{\mathsf{sk}}$

Data: A prompt (PROMPT) and a secret key sk
Result: Watermarked text x

```
x ← Model(PROMPT);
if H_e(Model, PROMPT, x) > 6λ then
    while F_sk(x) ≠ 0^b or H_e(Model, PROMPT, x) ≤ 6λ do
        x ← Model(PROMPT);
    end
end
return x;
```

This construction already demonstrates the importance of our completeness definition (Definition 7): Only trying to watermark outputs of high empirical entropy was crucial for this simple

[2]Note that we cannot sample the values of $\mathcal{O}$ on-the-fly, because Wat and Detect need to agree on all of the used values.

Algorithm 2: Weakly-sound watermarking detector $\mathsf{Detect}_{\mathsf{sk}}$

Data: Text x and a secret key sk
Result: true or false
if $F_{\mathsf{sk}}(x) = 0^b$ **then**
  return true;
else
  return false;
end

construction's undetectability. As we will see in Section 5, only watermarking high empirical entropy outputs is in fact inherent to undetectable watermarking schemes.

4 Constructing Undetectable Watermarks

4.1 Reduction to a Binary Alphabet

For ease of presentation and analysis, we describe our watermarking scheme as operating on text encoded as a binary string. That is, we assume that the token set is $\mathcal{T} = \{0, 1\}$.

Note that this assumption is without loss of generality: We can easily convert a model M with an arbitrary token set $\mathcal{T}$ into a model M' with a binary token set. First, we encode each token in $\mathcal{T}$ as a distinct string in $\{0,1\}^{\log|\mathcal{T}|}$; note that every codeword has length at most $\log|\mathcal{T}|$. For GPT-4, the number of tokens is $|\mathcal{T}| = 100,277$, and thus $\log|\mathcal{T}| \approx 17$ [Ope23]. Let E denote this encoding function, and let p_i be a distribution over $\mathcal{T}$ output by M. We convert p_i into a series of distributions $p'_{i,j}$ for M', where $p'_{i,1}$ is the distribution of the first bit of the codeword corresponding to a token sampled from p_i. That is, $p'_{i,1}(0) = \Pr_{t\leftarrow p_i}[E(t)_1]$, where $E(t)_1$ denotes the first bit of $E(t)$. Let $b_{i,1}$ denote the bit sampled by M' from $p'_{i,1}$. Each subsequent $p'_{i,j}$ is then sampled according to the distribution of the j^{th} bit of the codeword corresponding to a token sampled from p_i, conditioned on the previous bits being equal to $b_{i,1}, b_{i,2}, \ldots, b_{i,j-1}$. After M' samples the last bit of the current token from $p'_{i,|\mathcal{T}|}$, it calls M to obtain the distribution p_{i+1} for the next token.

Therefore, a watermarking scheme for binary alphabets can be used on models with token alphabets of arbitrary size using the above reduction. We note that the expected length of the encoding can be reduced by using a Huffman encoding of the token set instead of an arbitrary encoding.

4.2 Overview of the Construction

In Section 3, we saw a simple scheme that plants a watermark by sampling only texts for which an easily checkable predicate holds. In order to make this scheme more efficient, a natural idea is to sample tokens one at a time. If we don't require undetectability, an easy way to do this is to use a $\{0,1\}$-valued hash function h and sample tokens x_j with preference for those satisfying $h(x_j) = 1$. Given some text, we can determine whether a watermark is present by computing the hash of each token. In watermarked text, more tokens should hash to 1 than to 0; in un-watermarked text, there should be no bias. This is a classic idea in steganography, discussed in [HvAL09]. It is essentially the idea used in [KGW+23].

Unfortunately, this strategy significantly alters the output distribution, making it easily detectable: It prefers half of the words in the token set. As long as a biasing strategy yields a significant expected gap between the incidence of some predicate in watermarked text versus natural text, the resulting scheme should yield an observable watermark. Our objective is to plant a signal *without* noticeably changing the distribution of each token.

We first discuss a watermarking scheme that can only be used to generate a single output text of a predetermined maximum length L, for an arbitrary prompt. The secret key shared by the watermarked model and the detection algorithm will be a sequence $\vec{u} = u_1, \ldots, u_L$ of uniformly chosen real numbers in the range $[0, 1]$. Even though this state is independent of the prompt the model will receive, we show that this shared state is enough to plant a watermark in any single response. From the perspective of a user who doesn't know the secret key $\vec{u}$, the distribution of outputs is not changed at all.

When generating a response, the watermarked model will use the secret key to decide on each output token. Consider the generation of the j-th token in the response, after the previous tokens are already decided. Let $p_j(1)$ denote the probability, according to the real model, of this token being 1. The watermarked model outputs $x_j = 1$ if $u_j \leq p_j(1)$ and $x_j = 0$ otherwise. As u_j was drawn uniformly from $[0, 1]$, the probability that the watermarked model output $x_j = 1$ is *exactly* $p_j(1)$. Therefore, the distribution of generated text (in a single response) does not change at all. Nevertheless, we next show that the detection algorithm can compare the generated text to the shared sequence $\vec{u}$, and deduce that the generated output was drawn from the watermarked model.

For each text bit x_j, the detection algorithm can compute a score

$$s(x_j, u_j) = \begin{cases} \ln \frac{1}{u_j} & \text{if } x_j = 1 \\ \ln \frac{1}{1-u_j} & \text{if } x_j = 0 \end{cases}.$$

Given a string $x = (x_1, \ldots, x_L)$, the detection algorithm sums the score of all text bits

$$c(x) = \sum_{j=1}^{L} s(x_j, u_j).$$

Crucially, the detection algorithm does not need to know the distributions with which the model produces output bits. Since the detection algorithm does not have access to the prompt, it would not be able to compute those distributions.

We observe that the expected score is higher in watermarked text, as u_j is correlated with the output bit x_j. In non-watermarked text, the value of u_j is independent of the value of x_j. Therefore, $s(x_j, u_j)$ is simply an exponential random variable with mean 1:

$$\mathop{\mathbb{E}}_{u_j}[s(x_j, u_j)] = \int_0^1 \ln(1/x)\ dx = 1,$$

and we have $\mathbb{E}_{\vec{u}}[c(x) - |x|] = 0$.

For watermarked outputs, on the other hand,

$$\begin{aligned}
\mathop{\mathbb{E}}_{u_j}[s(x_j, u_j)] &= \int_0^{p_j(1)} \ln \frac{1}{u}\, du + \int_{p_j(1)}^1 \ln \frac{1}{1-u}\, du \\
&= \int_0^{p_j(1)} \ln \frac{1}{u}\, du + \int_0^{p_j(0)} \ln \frac{1}{u}\, du \\
&= \big(p_j(1) - p_j(1) \cdot \ln p_j(1)\big) + \big(p_j(0) - p_j(0) \cdot \ln p_j(0)\big) \\
&= 1 + \ln(2) \cdot H(p_j),
\end{aligned}$$

and the total expected score is

$$\mathop{\mathbb{E}}_{\vec{u}}[c(x) - |x|] = \ln 2 \cdot H(\overline{\mathsf{Model}}(\textsc{prompt})).$$

We've shown that there's a substantial gap between the expected scores of watermarked and natural text, as long as the text generation has high entropy. This should give us hope that this biasing strategy yields a reliable detector, but there are a few obstacles left on the way.

First, the expectation argument turns out to not be very useful because the variance of the score could be large. In Section 5 we discuss why this implies that we must consider *empirical entropy* instead of the entropy of the entire model. In Sections 4.3 and 4.4 we use empirical entropy to build effective distinguishers.

Second, the scheme described above is only indistinguishable for a single response, and that response must be shorter than the secret key. A natural idea is to use a pseudorandom function (refer to Section 2.1 for a definition) to determine the values u_j, instead of drawing them all in advance. For example, by setting $u_j = F_{\mathsf{sk}}(j)$ the length of any single response no longer has to be bounded. As F_{sk} is queried on each input j at most once, the values of u_j are pseudorandom and the distribution of a single watermarked output is computationally indistinguishable from the original distribution. The question becomes: Can we deal with multiple responses? One of our main contributions, and the most substantial difference from all prior work, is to answer this question in the affirmative.

Let $r^{(i)}$ be a unique identifier assigned to each response. This might be a global counter or a random string (usually referred to as a *nonce*). To sample the j-th token of the i-th response we can use $u_j^{(i)} = F_{\mathsf{sk}}(r^{(i)}, j)$. If all pairs $(r^{(i)}, j)$ are unique, then the values of $u_j^{(i)}$ are pseudorandom. However, the detection algorithm needs to know $r^{(i)}$ to compute the detection score. If $r^{(i)}$ is a counter, then we would need to keep a global state to maintain it. Moreover, to use the detection algorithm we would need to enumerate over all possible counter values. If $r^{(i)}$ is a long random string, no global state is needed, but the detection algorithm still needs to know $r^{(i)}$. While $r^{(i)}$ must be recoverable by the detection algorithm, it cannot simply be written in the output text, as we might as well just append "WATERMARK!" to it instead (which would obviously change the distribution of outputs). Our solution is to use real randomness to generate the first few tokens of each output, keeping track of how much *entropy* we used in the process. Once this entropy passes some specified threshold, we use the high-entropy prefix as $r^{(i)}$. Since these prefixes have high enough entropy, all choices of $r^{(i)}$ will be unique with all but negligible probability. The detection algorithm will test whether any prefix in the text, if used as $r^{(i)}$, will yield an unusually high score for the remainder of the text. The details of this construction are presented in Section 4.3.

In the above sketch the detector needs the entire output from the model to detect the watermark. We describe a modification of this scheme in Section 4.4 which is able to detect the watermark,

even when it is given only an contiguous substring of the output with sufficiently high entropy. Essentially, this modification works the same except it "resets" the choice of $r^{(i)}$ whenever enough *new* entropy is observed.

4.3 Constructing Undetectable Watermarks

Let $\mathsf{poly}_1(\cdot), \mathsf{poly}_2(\cdot)$ be polynomials. Let $F_{\mathsf{sk}} : \{0,1\}^{\mathsf{poly}_1(\lambda)} \to \{0,1\}^{\mathsf{poly}_2(\lambda)}$ be a PRF, where $\mathsf{sk} \in \{0,1\}^\lambda$. We wish to interpret the output of F_{sk} as a real number in $[0,1]$. We do so by letting z be the integer representation of the output and taking $\frac{z}{2^{\mathsf{poly}_2(\lambda)}}$. We consider poly_2 to be a large polynomial and ignore floating point errors. In the below algorithms, we allow F_{sk} to take strings of varying length as input; we assume that $\mathsf{poly}_1(\cdot)$ is chosen such that these strings are never too long, and if they are too short we pad them. In this section we assume that the token alphabet is binary as discussed in Section 4.1. We let `done` denote the binary encoding of the "done" token, and we write $\texttt{done} \in (x_1, \ldots, x_k)$ if and only if the decoding of $(x_1, \ldots, x_k)$ in the original token alphabet includes `done`.

Algorithm 3: Complete watermarking algorithm $\mathsf{Wat}_{\mathsf{sk}}$

Data: A prompt (PROMPT) and a secret key sk
Result: Watermarked text $x_1, \ldots, x_L$

$i \leftarrow 1$;
$H \leftarrow 0$;
while $\texttt{done} \notin (x_1, \ldots, x_{i-1})$ **do**
  $p_i \leftarrow \mathsf{Model}(\text{PROMPT}, x_1, \ldots, x_{i-1})$;
  if $H < \lambda$ **then**
    `// Collect more internal entropy`
    Sample $x_i \leftarrow p_i$;
    $H \leftarrow H - \log p_i(x_i)$;
    if $H \geq \lambda$ **then**
      $r \leftarrow (x_1, \ldots, x_i)$;
    end
  else
    `// Embed the watermark`
    $x_i \leftarrow \mathbb{1}[F_{\mathsf{sk}}(r, i) \leq p_i(1)]$;
  end
  $i \leftarrow i + 1$;
end

In this section we let $\mathcal{W} = (\mathsf{Setup}, \mathsf{Wat}, \mathsf{Detect})$ denote the watermarking scheme where Wat is Algorithm 3, Detect is Algorithm 4, and $\mathsf{Setup}(1^\lambda)$ samples $\mathsf{sk} \leftarrow \{0,1\}^\lambda$. This scheme is outlined above in Section 4.2.

Let $\mathsf{Wat}^{\mathcal{O}}$ and $\mathsf{Detect}^{\mathcal{O}}$ be the same algorithms as $\mathsf{Wat}_{\mathsf{sk}}$ and $\mathsf{Detect}_{\mathsf{sk}}$, except that they use a random oracle $\mathcal{O}$ instead of F_{sk}. Since both algorithms only make black-box use of F_{sk}, these are well-defined. Denote this random oracle scheme by $\mathcal{W}^{\mathcal{O}}$ (which does not need a Setup algorithm). Undetectability, $b(L)$-(substring-)completeness, and soundness are defined identically for $\mathcal{W}^{\mathcal{O}}$, except we replace the probabilities over $\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda)$ with probabilities over $\mathcal{O} \leftarrow \{f :$

Algorithm 4: Complete detector $\mathsf{Detect}_{\mathsf{sk}}$

Data: Text $x_1, \ldots, x_L$ and a secret key sk
Result: true or false

for $i \in [L]$ **do**
  $r^{(i)} \leftarrow (x_1, \ldots, x_i)$;
  Define $v_j^{(i)} := x_j \cdot F_{\mathsf{sk}}(r^{(i)}, j) + (1 - x_j) \cdot (1 - F_{\mathsf{sk}}(r^{(i)}, j))$ for $j \in [L]$;
  if $\sum_{j=i+1}^{L} \ln\left(1/v_j^{(i)}\right) > (L-i) + \lambda\sqrt{L-i}$ **then**
    return true;
  end
end
return false;

$\{0,1\}^{\mathsf{poly}_1(\lambda)} \to \{0,1\}^{\mathsf{poly}_2(\lambda)}\}$. Note that in the definition of undetectability for $\mathcal{W}^{\mathcal{O}}$, the distinguisher will not be given access to the random oracle $\mathcal{O}$ (since the distinguisher for $\mathcal{W}$ is not given access to F_{sk}).

Lemma 1. *The watermarking scheme $\mathcal{W}$ is undetectable/$b(L)$-(substring-)complete/sound if and only if $\mathcal{W}^{\mathcal{O}}$ is undetectable/$b(L)$-(substring-)complete/sound, assuming the security of the PRF used in $\mathcal{W}$.*

Proof. The security of the PRF says that black-box access to F_{sk} (for random sk) is indistinguishable from black-box access to a random $\mathcal{O}$, for any polynomial-time distinguisher. Observe that Algorithms 3 and 4 both only make black-box use of F_{sk}. Therefore it is possible to efficiently test, using only black-box access to F_{sk}, whether a given text/prompt/distinguisher violates soundness/$b(L)$-(substring-)completeness/undetectability. The security of the PRF then implies that the advantage of any given text/prompt/distinguisher is at most $\mathsf{negl}(\lambda)$ different between $\mathcal{W}$ and $\mathcal{W}^{\mathcal{O}}$. □

Lemma 2. *Let $E_1, \ldots, E_n$ be i.i.d. exponential random variables with rate 1, and let $\overline{E} := \sum_{i=1}^{n} E_i$ be their sum. Then for any $\tau > 0$,*

(a) $\Pr\left[\overline{E} \geq n + \sqrt{\tau n}\right] \leq \left(\frac{4}{5}\right)^{\sqrt{\tau}}$, *and*

(b) $\Pr\left[\overline{E} \leq n - \sqrt{\tau n}\right] \leq e^{-\tau/2}$.

Proof. We start with part (a). By [Jan18, Theorem 5.1(i)],

$$\begin{aligned}\Pr\left[\overline{E} \geq n + \sqrt{\tau n}\right] &\leq e^{-n \cdot \left(\sqrt{\tau/n} - \ln\left(1+\sqrt{\tau/n}\right)\right)} \\ &= \left(\frac{e^{\sqrt{\tau/n}}}{1+\sqrt{\tau/n}}\right)^{-n}.\end{aligned}$$

If $\tau \geq n$, then since $1 + z \leq 2^z$ for $z \geq 1$ we have

$$\begin{aligned}\left(\frac{e^{\sqrt{\tau/n}}}{1+\sqrt{\tau/n}}\right)^{-n} &\leq \left(\frac{e}{2}\right)^{-\sqrt{\tau n}} \\ &\leq \left(\frac{4}{5}\right)^{\sqrt{\tau n}}.\end{aligned}$$

If $\tau \le n$, then using $e^z \ge 1 + z + z^2/2$ for $z \ge 0$ we have

$$\begin{aligned}\left(\frac{e^{\sqrt{\tau/n}}}{1+\sqrt{\tau/n}}\right)^{-n} &\le \left(\frac{1+\sqrt{\tau/n}+\tau/2n}{1+\sqrt{\tau/n}}\right)^{-n} \\ &= \left(1+\frac{\tau/2n}{1+\sqrt{\tau/n}}\right)^{-n} \\ &\le \left(1+\frac{\tau}{4n}\right)^{-n} \\ &\le \left(1+\frac{1}{4}\right)^{-\tau} \\ &= \left(\frac{4}{5}\right)^{\tau},\end{aligned}$$

where we have also used the fact that $(1+\frac{z}{n})^n$ is monotonically increasing in n.

We now turn to part (b). By [Jan18, Theorem 5.1(iii)],

$$\begin{aligned}\Pr\left[\overline{E} \le n - \sqrt{\tau n}\right] &\le e^{n\cdot\left(\sqrt{\tau/n}+\ln\left(1-\sqrt{\tau/n}\right)\right)} \\ &= e^{\sqrt{\tau n}} \cdot \left(1-\sqrt{\tau/n}\right)^n.\end{aligned}$$

If $\tau \ge n$, the probability becomes 0. If $\tau < n$, then taking the natural logarithm the above becomes

$$\begin{aligned}\sqrt{\tau n} + n\ln\left(1-\sqrt{\tau/n}\right) &= \sqrt{\tau n}\cdot\left(1+\frac{\ln\left(1-\sqrt{\tau/n}\right)}{\sqrt{\tau/n}}\right) \\ &\le \sqrt{\tau n}\cdot\left(1-\frac{2}{2-\sqrt{\tau/n}}\right) \\ &\le -\tau/2\end{aligned}$$

where for $0 < z < 1$ we have used the facts that $\frac{\ln(1-z)}{z} \le \frac{-2}{2-z}$ and $\frac{2}{2-z} \ge 1 + \frac{z}{2}$. □

Theorem 4. *$\mathcal{W}$ is a sound watermarking scheme.*

Proof. Recall the definition of soundness in Definition 6. By Lemma 1 it suffices to show that for any text $x = x_1, \ldots, x_L$,

$$\Pr_{\mathcal{O}}[\mathsf{Detect}^{\mathcal{O}}(x) = \mathsf{true}] \le \mathsf{negl}(\lambda).$$

For $i, j \in [L]$, define $r^{(i)}$, $v_j^{(i)}$ as in Algorithm 4 and let $u_j^{(i)} := \mathcal{O}(r^{(i)}, j)$. Recall that $v_j^{(i)} = x_j \cdot u_j^{(i)} + (1-x_j)\cdot(1-u_j^{(i)})$.

Since $u_j^{(i)}$ is independent from x_j, we have $v_j^{(i)} \sim U([0,1])$. Therefore, $E_j^{(i)} := \ln\left(1/v_j^{(i)}\right)$ are independent exponential random variables with rate parameter 1. By Lemma 2,

$$\Pr\left[\sum_{j=i+1}^{L} E_j^{(i)} > (L-i) + \lambda\sqrt{L-i}\right] \le \left(\frac{4}{5}\right)^{\lambda}.$$

By a union bound over all L possible values of i, the probability of Algorithm 4 returning true is at most $L \cdot (4/5)^\lambda = \mathsf{negl}(\lambda)$, completing the proof. □

Theorem 5. *$\mathcal{W}$ is a $\left(\frac{4}{\ln 2}\lambda\sqrt{L}\right)$-complete watermarking scheme.*

Proof. Recall the definition of completeness in Definition 7. By Lemma 1 it suffices to show that for every PROMPT,

$$\Pr_{x \leftarrow \mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})}[\mathsf{Detect}^{\mathcal{O}}(x) = \mathsf{false} \text{ and } H_e(\mathsf{Model}, \text{PROMPT}, x) \geq b(|x|)] \leq \mathsf{negl}(\lambda) \tag{1}$$

where $b(L) = \frac{4}{\ln 2}\lambda\sqrt{L}$. In fact, we will prove something stronger: For *every* fixed $x \in \mathcal{T}^\star$ and PROMPT such that $H_e(\mathsf{Model}, \text{PROMPT}, x) \geq \frac{4}{\ln 2}\lambda\sqrt{|x|}$, if each bit x_i of x has empirical entropy $H_e^i(\mathsf{Model}, \text{PROMPT}, x) \leq \lambda$,

$$\Pr_{\mathcal{O}}\left[\mathsf{Detect}^{\mathcal{O}}(x) = \mathsf{false} \;\middle|\; \mathsf{Wat}^{\mathcal{O}}(\text{PROMPT}) = x\right] \leq \mathsf{negl}(\lambda). \tag{2}$$

Inequality 2 says that for any possible fixed output x that has high empirical entropy (which isn't too concentrated on any particular bit), conditioning on $\mathsf{Wat}^{\mathcal{O}}$ outputting it, it is likely to be detected as watermarked. Note that the probability here is over the choice of outputs of $\mathcal{O}$ and not over x, which is fixed.

Observe that Inequality 2 is not falsifiable, so we cannot do the PRF switch (Lemma 1) with it. However, since each bit has at most a $2^{-\lambda}$ chance of having empirical entropy more than λ, Inequality 2 implies Inequality 1 via the law of total probability.

In order to prove Inequality 2, we just need to show that the correct choice of prefix r, determined on Line 9 of Algorithm 3, has a high score (since Detect tries every possible prefix). Let $x = x_1, \ldots, x_L$ where $L := |x|$, and let $\ell := |r|$ where r is the correct prefix. For $j \in [\ell+1, L]$, let $v_j := x_j \cdot u_j + (1 - x_j) \cdot (1 - u_j)$. We prove that $\sum_{j=\ell+1}^{L} \ln \frac{1}{v_j}$ is likely to be larger than the detection threshold $(L - \ell) + \lambda\sqrt{L - \ell}$.

Denote by s_j the random variable $\ln \frac{1}{v_j}$, conditioned on $\mathsf{Wat}^{\mathcal{O}}(\text{PROMPT}) = x$, or equivalently, on the value of x_j. Recall that the variable $E = \ln \frac{1}{u}$ for $u \sim U([0,1])$ is exponentially distributed with rate 1. In particular, if $x_j = 1$ the variable s_j is distributed the same as E conditioned on $u \leq p_j(1)$, or equivalently on $E \geq \ln \frac{1}{p_j(1)}$. By the memorylessness property of exponential distributions, hence, s_j is distributed as $\ln \frac{1}{p_j(1)} + E_j$, where E_j is an exponentially distributed random variable with rate 1. Symmetrically, if $x_j = 0$ the variable s_j is distributed as $\ln \frac{1}{p_j(0)} + E_j$. We conclude that

$$\sum_{j=\ell+1}^{L} s_j \sim \ln(2) \cdot H_e^{[\ell+1,L]}(\mathsf{Model}, \text{PROMPT}, x) + \sum_{j=\ell+1}^{L} E_j,$$

where the factor of $\ln(2)$ comes from the switch to binary entropy. Recall that $\mathsf{Wat}_{\mathsf{sk}}$ fixes r as soon as it outputs x_ℓ for which $(x_1, \ldots, x_\ell)$ has empirical entropy of at least λ. Since each bit, including x_ℓ, has empirical entropy of at most λ, $(x_1, \ldots, x_\ell)$ has empirical entropy at most $2\lambda \leq \frac{2}{\ln 2}\lambda\sqrt{L}$.

Therefore,

$$\begin{aligned} H_e^{[\ell+1,L]}(\mathsf{Model}, \textsc{prompt}, x) &\geq H_e(\mathsf{Model}, \textsc{prompt}, x) - 2\lambda \\ &\geq \frac{2}{\ln 2}\lambda\sqrt{L} \\ &\geq \frac{2}{\ln 2}\lambda\sqrt{L-\ell}. \end{aligned}$$

Therefore, applying Lemma 2,

$$\begin{aligned} \Pr\left[\sum_{j=\ell+1}^{L} s_\ell \leq (L-\ell) + \lambda\sqrt{L-\ell}\right] &\leq \Pr\left[\sum_{j=\ell+1}^{L} E_j \leq (L-\ell) - \lambda\sqrt{L-\ell}\right] \\ &\leq e^{-\lambda^2/2}. \end{aligned}$$

□

Theorem 6. *$\mathcal{W}$ is an undetectable watermarking scheme.*

Proof. Recall Definition 9, which says that a watermark is undetectable if no efficient adversary can distinguish between query access to the watermarked model and the original one. Consider any fixed history of responses $x^{(1)}, \ldots, x^{(t-1)}$, and suppose that the adversary submits PROMPT as the next query. We will show that

$$\frac{1}{2}\left\|\mathsf{Wat}^{\mathcal{O}}(\textsc{prompt}) - \overline{\mathsf{Model}}(\textsc{prompt})\right\|_1 \leq \mathsf{negl}(\lambda).$$

Since the adversary can only make $\mathsf{poly}(\lambda)$ queries, it follows that the entire interaction with $\mathsf{Wat}^{\mathcal{O}}$ is statistically indistinguishable from interaction with $\overline{\mathsf{Model}}$. Finally, we will obtain the theorem by invoking Lemma 1.

Let $r^{(1)}, \ldots, r^{(t-1)}$ be the prefixes of the previous responses $x^{(1)}, \ldots, x^{(t-1)}$ (as determined on Line 9 of Algorithm 3). If for some $k \in [t]$ the watermarking scheme never collects enough entropy to assign a prefix $r^{(k)}$, we let $r^{(k)} := \perp$. We denote the tokens of $x^{(k)}$ by $(x_1^{(k)}, \ldots, x_{L^{(k)}}^{(k)})$ where $L^{(k)} := \left|x^{(k)}\right|$, and the corresponding probability distributions output by Model with $(p_1^{(k)}, \ldots, p_{L^{(k)}}^{(k)})$. We denote the tokens of $r^{(k)}$ by $(r_1^{(k)}, \ldots, r_{\ell^{(k)}}^{(k)})$ where $\ell^{(k)} := \left|r^{(k)}\right|$.

Since we have fixed $x^{(1)}, \ldots, x^{(t-1)}$, observe that the distribution on the next prefix $r^{(t)}$ is identical between $\mathsf{Wat}^{\mathcal{O}}$ and $\overline{\mathsf{Model}}$. This is because $\mathsf{Wat}^{\mathcal{O}}$ does not start embedding the watermark until after $r^{(t)}$ is completely sampled; until then $\mathsf{Wat}^{\mathcal{O}}$ samples tokens according to $\overline{\mathsf{Model}}$. Define the set $B := \{r^{(1)}, \ldots, r^{(t-1)}\} \setminus \{\perp\}$. For any fixed $r^{(t)} \notin B$, the distribution on the remaining tokens $(x_{\ell^{(t)}+1}^{(t)}, \ldots, x_{L^{(t)}}^{(t)})$ is also identical between $\mathsf{Wat}^{\mathcal{O}}$ and $\overline{\mathsf{Model}}$: If $r^{(t)} = \perp$, then there are no remaining tokens and the statement is trivial; if $r^{(t)} \notin \{r^{(1)}, \ldots, r^{(t-1)}\}$ then $\mathsf{Wat}^{\mathcal{O}}$ samples the remaining tokens with fresh randomness. We will show that $\Pr_{r^{(t)}}[r^{(t)} \in B] \leq \mathsf{negl}(\lambda)$, completing the proof.

For $k \in [t-1]$ and $i \in [L^{(k)}]$, we define

$$q_i^{(k)} := \mathsf{Model}(\textsc{prompt}, x_1^{(k)}, \ldots, x_{i-1}^{(k)}).$$

Note that $q_i^{(k)}(z)$ is the probability that $x_i^{(t)} = z$, given that $x_j^{(t)} = x_j^{(k)}$ for $j \in [i-1]$. We compute

$$\begin{aligned}
\Pr_{r^{(t)}}[r^{(t)} \in B] &= \Pr_{r^{(t)}}[r^{(t)} \in \{r^{(1)}, \ldots, r^{(t-1)}\} \text{ and } r^{(t)} \neq x^{(t)}] \\
&\leq \sum_{k=1}^{t-1} \Pr_{r^{(t)}}[r^{(t)} = r^{(k)} \text{ and } r^{(t)} \neq \perp] \\
&= \sum_{k=1}^{t-1} \mathbb{1}\left[\sum_{i=1}^{\ell^{(k)}-1} \log \frac{1}{q_i^{(k)}(x_i^{(k)})} < \lambda \leq \sum_{i=1}^{\ell^{(k)}} \log \frac{1}{q_i^{(k)}(x_i^{(k)})}\right] \cdot \prod_{i=1}^{\ell^{(k)}} q_i^{(k)}(r_i^{(k)}) \\
&\leq \sum_{k=1}^{t-1} \mathbb{1}\left[\lambda \leq -\log \prod_{i=1}^{\ell^{(k)}} q_i^{(k)}(r_i^{(k)})\right] \cdot \prod_{i=1}^{\ell^{(k)}} q_i^{(k)}(r_i^{(k)}) \\
&\leq (t-1) \cdot 2^{-\lambda}.
\end{aligned}$$

□

4.4 Constructing Substring-Complete Watermarks

The detector presented in Section 4.3 receieves as an input the entire text output by $\mathsf{Wat}_{\mathsf{sk}}$. In this section we generalize the scheme into a substring-complete one, which is able to detect watermarks in any contiguous sequence of text with sufficiently high empirical entropy.

The new scheme, described in Algorithms 5 and 6, is essentially a repeated version of Algorithms 3 and 4. First, it samples naturally from the model until it has collected enough empirical entropy. Once we collect λ bits of empirical entropy in a text block r, we start embedding the watermark using r as our seed. We continue embedding the watermark in the next subsequence of tokens $(x_i, \ldots, x_{i+\ell-1})$ until we have collected enough empirical entropy to know that the watermark will be detected with high probability. At this point, we could restart Algorithm 3, sampling the next tokens from $x_j \leftarrow p_j$ to generate a new seed r. Instead, we observe that we can in fact use the previous subsequence $(x_i, \ldots, x_{i+\ell-1})$ itself as r, as it is of high enough empirical entropy as well.

We use the same notation as in Section 4.3, except that now $\mathcal{W}$ refers to the scheme defined in Algorithms 5 and 6.

Theorem 7. *$\mathcal{W}$ is a sound watermarking scheme.*

Proof. Up to symbolic differences, the proof is identical to that of Theorem 4 except that the union bound will be over L^3 possible values of i, ℓ, k rather than L possible values of i. □

Theorem 8. *$\mathcal{W}$ is a $\left(\frac{8}{\ln 2}\lambda\sqrt{L}\right)$-substring-complete watermarking scheme.*

Proof. We argue that any substring with empirical entropy at least $\frac{8}{\ln 2}\lambda\sqrt{L}$ must include the entirety of a pair of consecutive blocks $r^{(a)}, r^{(a+1)}$. We then refer to the proof of Theorem 5 to argue that the bias planted in $r^{(a+1)}$ with the seed $r^{(a)}$ can be detected with overwhelming probability.

Recall the definition of completeness in Definition 7. By Lemma 1 it suffices to show that for every PROMPT,

$$\Pr_{\substack{\mathcal{O} \\ x \leftarrow \mathsf{Wat}^{\mathcal{O}}(\text{PROMPT})}} \Big[\exists\, i, L \in [|x|] \text{ such that } \mathsf{Detect}_{\mathsf{sk}}(x[i:i+L]) = \mathsf{false} \text{ and } H_e^{[i:i+L]}(\mathsf{Model}, \text{PROMPT}, x) \geq b(L) \Big] \leq \mathsf{negl}(\lambda). \tag{3}$$

Algorithm 5: Substring-complete watermarking algorithm $\mathsf{Wat}_{\mathsf{sk}}$.

Data: A prompt (PROMPT), secret key sk, and parameter λ
Result: Watermarked text $x_1, \ldots, x_L$

$r \leftarrow \perp, H \leftarrow 0, i \leftarrow 1, \ell \leftarrow 0$;
while *done* $\notin (x_1, \ldots, x_{i-1})$ **do**
  $p_i \leftarrow \mathsf{Model}(\text{PROMPT}, x_1, \ldots, x_{i-1})$;
  if $r = \perp$ **then**
    `// Still sampling the first block`
    Sample $x_i \leftarrow p_i$;
  else
    `// Embed the watermark`
    $x_i \leftarrow \mathbb{1}[F_{\mathsf{sk}}(r, i) \leq p_i(1)]$;
  $H \leftarrow H - \log p_i(x_i)$;
  if $H \geq \frac{2}{\ln 2}\lambda\sqrt{\ell}$ **then**
    `// Reassign` r
    $r \leftarrow (x_{i-\ell}, \ldots, x_i)$;
    $H \leftarrow 0, \ell \leftarrow 0$;
  end
  $i \leftarrow i + 1, \ell \leftarrow \ell + 1$;
end

Algorithm 6: Substring-complete detector $\mathsf{Detect}_{\mathsf{sk}}$.

Data: Text $x_1, \ldots, x_L$ and a secret key sk
Result: true or false

for $i, \ell \in [L]$, $\ell < i$ **do**
  $r^{(i,\ell)} \leftarrow (x_{i-\ell}, \ldots, x_i)$;
  $v_j^{(i,\ell)} \leftarrow x_j \cdot F_{\mathsf{sk}}(r^{(i,\ell)}, j) + (1 - x_j) \cdot (1 - F_{\mathsf{sk}}(r^{(i,\ell)}, j))$ for each $j \in [L]$;
  for $k \in [i + 1, L]$ **do**
    if $\sum_{j=i+1}^{k} \ln\left(1/v_j^{(i,\ell)}\right) > (k - i) + \lambda\sqrt{k - i}$ **then**
      return true;
    end
  end
end
return false;

where $b(L) = \frac{8}{\ln 2}\lambda\sqrt{L}$. We'll show that for *every* fixed $x \in \mathcal{T}^\star$, $i, L \in [|x|]$, and PROMPT such that $H_e^{[i:i+L]}(\mathsf{Model}, \text{PROMPT}, x) \geq \frac{8}{\ln 2}\lambda\sqrt{L}$, if each bit x_j of $x[i : i+L]$ has empirical entropy $H_e^j(\mathsf{Model}, \text{PROMPT}, x) \leq \lambda$,

$$\Pr_{\mathcal{O}}\left[\mathsf{Detect}^{\mathcal{O}}(x) = \mathsf{false} \mid \mathsf{Wat}^{\mathcal{O}}(\text{PROMPT}) = x\right] \leq \mathsf{negl}(\lambda). \tag{4}$$

For any fixed $x \in \mathcal{T}^\star$, let i, L and $x = (x_i, \ldots x_{i+L})$ be such that $H_e^{[i:i+L]}(\mathsf{Model}, \text{PROMPT}, x) \geq \frac{8}{\ln 2}\lambda\sqrt{L}$ and $H_e^j(\mathsf{Model}, \text{PROMPT}, x) \leq \lambda$ for each $j \in [i : i+L]$. Let $c, d \in [i : i+L]$ be such that $r^{(a)} := (x_c, \ldots, x_d)$ is the first contiguous substring of $x[i : i+L]$ such that r was assigned to $r^{(a)}$ in Line 10 of Algorithm 5. Recall that we reassign each seed r as soon as we have collected $\frac{2}{\ln 2}\lambda\sqrt{\ell}$ empirical entropy where $\ell \leq L$. So $x[i : c-1]$ contains at most $\frac{2}{\ln 2}\lambda\sqrt{L} + \lambda$ empirical entropy; otherwise, $r^{(a)}$ would have been assigned for a lower starting index. Similarly, $r^{(a)}$ contains at most $\frac{2}{\ln 2}\lambda\sqrt{L}+\lambda$ empirical entropy. Let $r^{(a+1)} = (x_{d+1}, \ldots, x_k)$ denote the next block after $r^{(a)}$. Observe that

$$H_e^{[d+1:i+L]}(\mathsf{Model}, \text{PROMPT}, x) \geq \frac{8}{\ln 2}\lambda\sqrt{L} - \frac{4}{\ln 2}\lambda\sqrt{L} - 2\lambda \geq \frac{2}{\ln 2}\lambda\sqrt{k-d}.$$

Therefore, after $r^{(a)}$ has been fixed, there is sufficient entropy in the remainder of our substring $x[d+1 : i+L]$ so that $r^{(a+1)}$ will be contained entirely within it. Since $r^{(a+1)}$ has empirical entropy at least $\frac{2}{\ln 2}\lambda\sqrt{k-d}$, we can now follow the analysis in the proof of Theorem 5 to conclude that this choice of c, d, k in the detection algorithm results in an output of true. □

Theorem 9. *$\mathcal{W}$ is an undetectable watermarking scheme.*

Proof. The proof is similar to Theorem 6, but we index by "blocks" instead of queries. These serve the same purpose as the "prefixes" of Theorem 6, except that there may be many blocks in any single query. We define a *block* to be a sequence of tokens $x_{i-\ell}, \ldots, x_i$ defining a value of r on Line 10 of Algorithm 5. If the last block in a response does not get completed before reaching a `done` token, we assign that block the value $\perp$. Over the course of the distinguishing experiment that defines undetectability in Definition 9, $\mathsf{Wat}^{\mathcal{O}}$ will output some number of blocks in each response. We enumerate all blocks $r^{(1)}, r^{(2)}, \ldots$ that $\mathsf{Wat}^{\mathcal{O}}$ outputs during the course of the experiment (across all responses).

Recall Definition 9, which says that a watermark is undetectable if no efficient adversary can distinguish between query access to the watermarked model and the original one. Let $\mathcal{R}_W$ and $\mathcal{R}_M$ be the distributions over the next block $r^{(t)}$ under $\mathsf{Wat}^{\mathcal{O}}$ and $\overline{\mathsf{Model}}$, respectively.

We will prove the following:

> For any fixed history of blocks $r^{(1)}, \ldots, r^{(t-1)}$, if
>
> $$r^{(t-1)} \notin \{r^{(1)}, \ldots, r^{(t-2)}\} \setminus \{\perp\}$$
>
> then $\mathcal{R}_W = \mathcal{R}_M$ and
>
> $$\Pr_{r^{(t)}}\left[r^{(t)} \in \{r^{(1)}, \ldots, r^{(t-1)}\} \setminus \{\perp\}\right] \leq \mathsf{negl}(\lambda). \tag{5}$$

Inductively, any $\mathsf{poly}(\lambda)$-many blocks output by $\mathsf{Wat}^{\mathcal{O}}$ are at most $\mathsf{negl}(\lambda)$-far in statistical distance from outputs of $\overline{\mathsf{Model}}$. Since only $\mathsf{poly}(\lambda)$ blocks can appear in the experiment, the entire interaction

with $\mathsf{Wat}^{\mathcal{O}}$ is statistically indistinguishable from interaction with $\overline{\mathsf{Model}}$. Finally, we will obtain the theorem by invoking Lemma 1.

Depending on whether t is the first block in the response, we reason that $\mathcal{R}_W = \mathcal{R}_M$ differently:

- If $r^{(t)}$ is the first block in the response, then $\mathcal{R}_W = \mathcal{R}_M$ because $\mathsf{Wat}^{\mathcal{O}}$ and $\overline{\mathsf{Model}}$ behave identically until after the first block is sampled.
- If $r^{(t)}$ is not the first block in the response (i.e., $r^{(t-1)}$ is a part of the same response as $r^{(t)}$), then $\mathcal{R}_W = \mathcal{R}_M$ because $r^{(t-1)} \notin \{r^{(1)}, \ldots, r^{(t-2)}\}$ and therefore $\mathsf{Wat}^{\mathcal{O}}$ uses fresh randomness to generate $r^{(t)}$. (Note that $r^{(t-1)} \neq \bot$, since the subsequent block $r^{(t)}$ is a part of the same response.)

In either case, once we know that $\mathcal{R}_W = \mathcal{R}_M$, Inequality 5 follows from the exact same argument as in the proof of Theorem 6. □

5 Necessity of Assumptions

In this section we show that the assumptions we use for our construction are necessary. Informally, the two main statements we prove in this section are:

- Undetectability is possible only against a computationally bounded adversary. That is, using a polynomial number of queries and exponential running time we can detect any nontrivial watermarking scheme. This is proven in Lemma 4.
- Undetectability is impossible if outputs of low empirical entropy are watermarked: For any t, if a non-negligible fraction of outputs with empirical entropy $\leq t$ are watermarked then we can detect the watermark using $\exp(t)$ queries and time. This is proven in Theorem 10.

To warm up, we first observe that there exist models that generate text with arbitrarily high entropy, and nevertheless in any undetectable watermark only a negligible fraction of outputs can be watermarked. Hence, the natural attempt to only consider the model's entropy is insufficient.

Lemma 3. *For all constants $b, \varepsilon > 0$ and security parameter λ, there exists a prompt-independent model* Model *such that $H(\overline{\mathsf{Model}}(\emptyset)) \geq b$, the maximum length of an output of* $\overline{\mathsf{Model}}$ *is $\frac{1}{\varepsilon}b$, and the following holds. If $\mathcal{W}$ is any undetectable and sound watermarking scheme for* Model*, then*

$$\Pr_{\substack{\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda) \\ x \leftarrow \mathsf{Wat}_{\mathsf{sk}}(\emptyset)}} [\mathsf{Detect}_{\mathsf{sk}}(x) = \mathsf{true}] \leq \varepsilon + \mathsf{negl}(\lambda).$$

Proof. We define the output distribution of $\overline{\mathsf{Model}}$ as follows. With probability $1-\varepsilon$, $\overline{\mathsf{Model}}$ outputs $\mathtt{done}$. Otherwise, it outputs a uniformly random binary string of length $\frac{1}{\varepsilon}b$. The entropy of $\overline{\mathsf{Model}}$ is larger than $\varepsilon \cdot \frac{1}{\varepsilon}b > b$.

Due to soundness, with high probability ($\mathtt{done}$) is not detected as watermarked, that is

$$\Pr_{\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda)} [\mathsf{Detect}_{\mathsf{sk}}((\mathtt{done})) = \mathsf{true}] \leq \mathsf{negl}(\lambda).$$

On the other hand, due to undetectability, $\mathsf{Wat}_{\mathsf{sk}}(\emptyset)$ must output ($\mathtt{done}$) with probability $(1-\varepsilon) \pm \mathsf{negl}(\lambda)$, so the statement of the lemma holds. □

Next, we show that computational assumptions are necessary for undetectability of watermarks. That is, while we can construct watermarks where the output distributions of $\mathsf{Wat}_{\mathsf{sk}}$ and of $\overline{\mathsf{Model}}$ are indistinguishable to any *efficient* distinguisher, those distributions must not be identical and thus a distinguisher with unbounded running time is able to distinguish between them. We prove a strong version of this statement: We show that for *every* model and *every* watermarking scheme it is possible to statistically distinguish $\overline{\mathsf{Model}}$ from $\mathsf{Wat}_{\mathsf{sk}}$, using a polynomial number of queries from any PROMPT that produces watermarked outputs with non-negligible probability.

Lemma 4. *Let* Model *be a model and $\mathcal{W}$ a watermarking scheme for it that is sound. Let K be an upper bound on the size (in bits) of any possible secret key* sk *for $\mathcal{W}$. Let $\varepsilon > \frac{1}{\mathsf{poly}(\lambda)}$ and let* PROMPT *be a prompt for which*

$$\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)\\ x\leftarrow\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})}}[\mathsf{Detect}_{\mathsf{sk}}(x)=\mathsf{true}]\geq\varepsilon.$$

Then, for randomly chosen $\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)$, it is possible to distinguish between $\overline{\mathsf{Model}}$ and $\mathsf{Wat}_{\mathsf{sk}}$ with probability at least $\frac{1}{2}\varepsilon$, using $\mathsf{poly}\left(\frac{K}{\varepsilon}\right)$ queries and $\exp(K)$ running time.

Proof. As $\mathcal{W}$ is sound, we in particular have that a random output is unlikely to be detected as watermarked,

$$\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)\\ x\leftarrow\overline{\mathsf{Model}}(\text{PROMPT})}}[\mathsf{Detect}_{\mathsf{sk}}(x)=\mathsf{true}]\leq\mathsf{negl}(\lambda).$$

Let's call this property *sound on average*. As soundness on average and the completeness assumption are distributional, we may assume that $\mathsf{Detect}_{\mathsf{sk}}$ is deterministic by Yao's minimax principle, while maintaining both properties.[3] Therefore, for every possible secret key sk there exists a subset W_{sk} of possible outputs such that $\mathsf{Detect}_{\mathsf{sk}}(x)=\mathsf{true}$ if and only if $x\in W_{\mathsf{sk}}$.

As $\mathcal{W}$ is sound on average, we have that

$$\mathop{\mathbb{E}}_{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)}\left[\Pr_{x\leftarrow\overline{\mathsf{Model}}(\text{PROMPT})}[x\in W_{\mathsf{sk}}]\right]=\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)\\ x\leftarrow\overline{\mathsf{Model}}(\text{PROMPT})}}[\mathsf{Detect}_{\mathsf{sk}}(x)=\mathsf{true}]\leq\mathsf{negl}(\lambda).$$

By the completeness assumption, we have

$$\mathop{\mathbb{E}}_{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)}\left[\Pr_{x\leftarrow\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})}[x\in W_{\mathsf{sk}}]\right]=\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{Setup}(1^\lambda)\\ x\leftarrow\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})}}[\mathsf{Detect}_{\mathsf{sk}}(x)=\mathsf{true}]\geq\varepsilon.$$

In particular, due to both assumptions and Markov's inequality, with probability at least $1-\frac{1}{2}\varepsilon$ the following hold for the drawn secret key sk:

1. $\Pr_{x\leftarrow\overline{\mathsf{Model}}(\text{PROMPT})}[x\in W_{\mathsf{sk}}]<\frac{1}{2}\varepsilon^2$.
2. $\Pr_{x\leftarrow\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})}[x\in W_{\mathsf{sk}}]>\varepsilon^2$.

[3] This means that we can always replace $\mathsf{Detect}_{\mathsf{sk}}$ with a deterministic algorithm that still has those two properties. This is done by fixing the randomness used by $\mathsf{Detect}_{\mathsf{sk}}$.

Let $\mathcal{O}$ be any distribution of strings, and W any set of strings. Let S be set of $\frac{10}{\varepsilon^2}K$ independent strings drawn from $\mathcal{O}$. A Chernoff bound yields that

$$\Pr_S\left[\left|\frac{|S\cap W|}{|S|} - \Pr_{x\leftarrow\mathcal{O}}\left[x\in W\right]\right| > \frac{1}{4}\varepsilon^2\right] \leq 3^{-K}.$$

Given access to two distributions $\mathcal{O}_1, \mathcal{O}_2$, we can draw $\frac{10}{\varepsilon^2}K$ samples from each, denote them by S_1, S_2 respectively, and then compute for every possible key k the quantities $\frac{|S_1\cap W_k|}{|S_1|}$ and $\frac{|S_2\cap W_k|}{|S_2|}$. Due to the Chernoff bound above and to a simple union bound, with probability at least $1-2\cdot 1.5^{-K}$, those quantities approximate up to $\pm\frac{1}{4}\varepsilon^2$ the probabilities $\Pr_{x\leftarrow\mathcal{O}_j}\left[x\in W_k\right]$ for *every* choice of possible key k and $j\in\{1,2\}$. In that case, if $\mathcal{O}_1 = \mathcal{O}_2 = \overline{\mathsf{Model}}$ there will not exist any k for which $\frac{|S_1\cap W_k|}{|S_1|} < \frac{3}{4}\varepsilon^2$ and $\frac{|S_2\cap W_k|}{|S_1|} > \frac{3}{4}\varepsilon^2$, but if $\mathcal{O}_1 = \overline{\mathsf{Model}}$ and $\mathcal{O}_2 = \mathsf{Wat}_{\mathsf{sk}}$, there would. Thus, we can distinguish between those two cases by making $\frac{20}{\varepsilon^2}K$ queries and enumerating over all 2^K possible keys. □

An important remark about Lemma 4 is that it requires the size of the secret key to be bounded (with respect to the number of queries the distinguisher makes). If we wanted the distributions to be indistinguishable only to a much weaker distinguisher that is only allowed to sample each distribution once, or a bounded number of times, then we could achieve statistically identical distributions. This can be achieved by the construction presented in Section 4, if the PRF is replaced with a long secret key containing many random values that will each be used exactly once.

Finally, we show that any sound watermarking scheme that successfully watermarks outputs with empirical entropy smaller than t, is detectable with $\exp(t)$ queries and time. This means that any watermarking scheme can only work for outputs with empirical entropy that is high enough with respect to the time the distinguisher is allowed to spend.

Theorem 10. *Let* Model *be a model and* $\mathcal{W}$ *a watermarking scheme for it that is sound with respect to a security parameter* λ*. Let* t *be a parameter, and* PROMPT *a prompt for which*

$$\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{Setup}\\ x\leftarrow\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})}}\left[\mathsf{Detect}_{\mathsf{sk}}(x)=\mathsf{true}\ \textit{and}\ H_e\left(\mathsf{Model},\text{PROMPT},x\right)\leq t\right] > \frac{1}{\mathsf{poly}(\lambda)}.$$

Then, it is possible to distinguish between the distributions $\mathsf{Wat}_{\mathsf{sk}}$ *and* $\overline{\mathsf{Model}}$ *with* $O\left(\exp\left(t\right)\cdot\mathsf{poly}(\lambda)\right)$ *queries and time, with a non-negligble probability.*

Proof. By making $O\left(\exp\left(t\right)\cdot\mathsf{poly}(\lambda)\right)$ queries to a distribution $\mathcal{O}$ we may approximate with accuracy $\pm 10^{-\lambda}$ the probability in the distribution $\mathcal{O}$ of every output x that has empirical entropy $\leq t$. Note that there are at most 2^t such outputs. By Lemma 4 and the assumption, there is a statistical difference between the distributions $\overline{\mathsf{Model}}(\text{PROMPT})$ and $\mathsf{Wat}_{\mathsf{sk}}(\text{PROMPT})$, even when we condition on the output being of empirical entropy $\leq t$. Hence, approximating both distributions on all such outputs is enough to distinguish between them. Note that unlike in the proof of Lemma 4, we are now not enumerating over all possible keys (that may be longer than t or λ), we simply approximate both distributions and compute the statistical difference between them. □

6 Removing Watermarks

A natural question is how robust an undetectable watermarking scheme can be to active attempts to remove it. While we would ideally like to have an undetectable watermarking scheme that is robust

to any efficient adversary attempting to remove a watermark, there are both practical and theoretical barriers to achieving this property. In Section 6.1 we first describe several attacks that work well at removing watermarks in practice. Then in Section 6.2 we present an (expensive) attack that provably removes a watermark from any undetectable scheme. We conclude that no undetectable watermarking scheme can be *completely* unremovable. Still, it might require significantly more resources for a user to generate unwatermarked text from the model.

6.1 Empirical Attacks on Watermarking Schemes

We highlight some relevant practical attacks, described for an attacker wishing to generate an unwatermarked response to a prompt PROMPT. [KGW+23] gives a nice overview of practical attacks removing watermarks, including a few of those mentioned here.

Emoji attack. In the "emoji attack," the attacker asks the model to output a response to PROMPT with an emoji inserted between every pair of words.[4] The attacker then removes the emojis to obtain the desired response. This attack removes any watermark that relies on the detector seeing consecutive sequences of tokens, including ours as well as those of [KGW+23] and [Aar22]. In general this attack may not preserve the output distribution, but any provable robustness guarantee for contiguous-text watermarks would have to rest on the dubious assumption that it *doesn't*.

Translation attack. The attacker can ask the model to write in a different language, and then translate the response to their language of choice. Depending on the fluency of the model in the other language, and the quality of the translation tool, this attack may significantly degrade the quality of text.

Paraphrasing and substitution attacks. The attacker obtains a response from the model. In the substitution attack, the attacker replaces some words with their synonyms. In the paraphrasing attack, the attacker paraphrases the text either manually or using a model as in [KSK+23] or the span replacement attack of [KGW+23]. Depending on how many words are changed, these attacks might remove the watermark from our scheme and those of [KGW+23, Aar22]. Of course, changing more of the text also increases the risk of degrading its quality.

Post-hoc attacks. For post-hoc detection schemes, there are two simple attacks that are empirically quite effective at evading detection. First, current LLMs are so powerful that they are often capable of simply evading detection themselves if you ask them to: See, for instance, this Reddit post.[5] Second, in some models, e.g. OpenAI's "Playground", one can change model parameters. By increasing the temperature, frequency penalty, and presence penalty, one can often produce text that evades post-hoc detection.

6.2 Removing any Undetectable Watermark

In this section, we describe an attack that removes a watermark from any undetectable watermarking scheme, assuming that the model is "prefix-specifiable." The attack is simple: Just sample tokens

[4] https://twitter.com/goodside/status/1610682909647671306

[5] https://www.reddit.com/r/ChatGPT/comments/11pqmqm/you_can_ask_chat_gpt_to_write_a_text_than_alter/.

from the watermarked model one at a time.

We say that a model is *prefix-specifiable* if the user can specify a prefix of the model's response. More formally, we require that for any PROMPT and text $x_1, \ldots, x_k$, the user can efficiently compute a new prompt PROMPT$'$ such that $\overline{\mathsf{Model}}$(PROMPT$'$) is distributed identically to $\overline{\mathsf{Model}}$(PROMPT) conditioned on the response's prefix matching $(x_1, \ldots, x_k)$. This property is also assumed in the definition of a language model in [KGW+23].

On prefix-specifiable models. Any language model according to Definitions 1 and 2 can be given a prefix-specifiable interface, but real-world user interfaces may or may not allow it. For instance, ChatGPT does not allow the user to specify prefixes of the response, but the OpenAI Playground allows the user to submit text under the "Assistant" role which the model will use as a prefix for its next response. We do not know for certain if the resulting distribution is actually identical to the model's response conditioned on the given prefix.

We note that a user can always attempt to "trick" a model into being prefix-specifiable: Simply include in the prompt a request to start the response with the given prefix. We also note that when we are defining a model, we can always design it to be "prefix-specifiable" by forcing it to follow such requests. In particular, a watermarking scheme that works for *every* model would need to work also for prefix-specifiable models.

Theorem 11. *Let $\mathcal{W} = (\mathsf{Setup}, \mathsf{Wat}_{\mathsf{sk}}, \mathsf{Detect}_{\mathsf{sk}})$ be any undetectable watermarking scheme. Assume that the underlying model $\overline{\mathsf{Model}}$ is prefix-specifiable. Then there exists an efficient algorithm $\mathcal{A}$ making queries to $\mathsf{Wat}_{\mathsf{sk}}$ such that, for any* PROMPT *and a random $\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda)$, the distributions $\mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}}$(PROMPT) and $\overline{\mathsf{Model}}$(PROMPT) are $\mathsf{negl}(\lambda)$-close in* statistical *distance. The number of queries made by $\mathcal{A}$ to $\mathsf{Wat}_{\mathsf{sk}}$ is exactly the length of text output by $\mathcal{A}$.*

Proof. The attacker generates an unwatermarked response to PROMPT as follows. It lets y_1 be the first token of $\mathsf{Wat}_{\mathsf{sk}}$(PROMPT). It then lets y_2 be the first token of $\mathsf{Wat}_{\mathsf{sk}}$(PROMPT, y_1), and so on, in general computing y_j as the first token in $\mathsf{Wat}_{\mathsf{sk}}$(PROMPT, $y_1, \ldots, y_{j-1}$) until $y_j = \mathsf{done}$.

Let $Y = (Y_1, \ldots, Y_{L_Y})$ be random variables denoting the output of the attacker. Let $X = (X_1, \ldots, X_{L_X})$ be random variables denoting the output of $\overline{\mathsf{Model}}$(PROMPT); note that L_Y and L_X are random variables here.

We argue that for each i, $\Delta((Y_i \mid Y_j = y_j\ \forall j < i), (X_i \mid X_j = y_j \forall j < i)) \leq \mathsf{negl}(\lambda)$. Suppose for the sake of contradiction that the total variation distance between these distributions for some i is at least $\frac{1}{\mathsf{poly}(\lambda)}$ for some polynomial poly. A distinguisher trying to determine whether it has oracle access to $\overline{\mathsf{Model}}$ or $\mathsf{Wat}_{\mathsf{sk}}$ can make $O(\mathsf{poly}(|\mathcal{T}|) \cdot \lambda)$ queries to the oracle with input PROMPT, $(x_1, \ldots, x_{i-1})$, and approximate the distribution of the first token in the response up to additive error $\pm 10^{-\lambda}$ in total variation distance. Since $10^{-\lambda} << 1/\mathsf{poly}(\lambda)$, this contradicts the undetectability of $\mathsf{Wat}_{\mathsf{sk}}$.

This tells us that for any partial response $(x_1, \ldots, x_i)$ to PROMPT, the distribution of the next token must have $\mathsf{negl}(\lambda)$ total variation distance from that of the unwatermarked model. Therefore, the entire response produced by this attack using the watermarked model must be have negligible total variation distance from the unwatermarked model. □

By soundness, the watermark cannot be detected in the output of $\mathcal{A}$ with non-negligible probability. Therefore, this attack succeeds in removing the watermark.

Fortunately, for reasonable text sizes this attack is probably impractical. For instance, while generating an $8,000$-token response from GPT-4 currently costs $\frac{\$0.06}{1000} \times 8000 = \$0.48$, generating

the same text using the above attack would cost $\$0.48 + \frac{\$0.03}{1000} \times 8000 \times 7999/2 = \$960.36$. (Current GPT-4 pricing is \$0.03 per 1000 prompt tokens and \$0.06 per 1000 sampled tokens.) Still, it shows that we cannot hope to achieve an overly sweeping definition of completeness/unremovability; it cannot allow an adversary powerful enough to run this attack.

7 Open Problems

Section 6 implies that undetectable watermarking schemes cannot be made "unremovable" in a general sense. Nevertheless, it is intriguing to find the most general sense in which undetectable watermarks can be made robust to removal attempts. For example, our construction of substring-complete watermarks guarantees detection as long as a consecutive substring of the output with high enough empirical entropy remains intact. Can this property be generalized to non-consecutive subsets of the output? Are there larger classes of *removal techniques* against which undetectable watermarks can be made robust?

Quantitatively, our main schemes in Section 4 might not achieve an optimal completeness parameter. The simplified scheme we present in Section 3 achieves completeness with a parameter that only depends on λ, but not soundness or worst-case polynomial runtime. Our full construction in Section 5, on the other hand, achieves all required properties yet has a possibly non-optimal $\Theta\left(\lambda\sqrt{L}\right)$ completeness parameter.

Can we close this gap, either by improving our scheme to require less entropy for detection, or by showing a stronger lower bound for schemes that are sound (or efficient)?

Acknowledgments

This research was supported in part by NSF Grants CCF-2107187, CCF-1763970, and CCF-2212233, by JPMorgan Chase & Co, by LexisNexis Risk Solutions, and by the Algorand Centres of Excellence programme managed by Algorand Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are solely those of the authors.

References

[Aar22] Scott Aaronson. My AI Safety Lecture for UT Effective Altruism. `https://scottaaronson.blog/?p=6823`, November 2022. Accessed May 2023. 5, 27

[AF21] Sahar Abdelnabi and Mario Fritz. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 121–140. IEEE, 2021. 4

[ARC+01] Mikhail J Atallah, Victor Raskin, Michael Crogan, Christian Hempelmann, Florian Kerschbaum, Dina Mohamed, and Sanket Naik. Natural language watermarking: Design, analysis, and a proof-of-concept implementation. In *Information Hiding: 4th International Workshop, IH 2001 Pittsburgh, PA, USA, April 25–27, 2001 Proceedings 4*, pages 185–200. Springer, 2001. 6

[ARH+03] Mikhail J Atallah, Victor Raskin, Christian F Hempelmann, Mercan Karahan, Radu Sion, Umut Topkara, and Katrina E Triezenberg. Natural language watermarking and tamperproofing. In *Information Hiding: 5th International Workshop, IH 2002 Noordwijkerhout, The Netherlands, October 7-9, 2002 Revised Papers 5*, pages 196–212. Springer, 2003. 6

[Ber16] Daria Beresneva. Computer-generated text detection using machine learning: A systematic review. In *Natural Language Processing and Information Systems: 21st International Conference on Applications of Natural Language to Information Systems, NLDB 2016, Salford, UK, June 22-24, 2016, Proceedings 21*, pages 421–426. Springer, 2016. 4

[CBZ+23] Souradip Chakraborty, Amrit Singh Bedi, Sicheng Zhu, Bang An, Dinesh Manocha, and Furong Huang. On the possibilities of ai-generated text detection. *arXiv preprint arXiv:2304.04736*, 2023. 4

[DIRR09] Nenad Dedić, Gene Itkis, Leonid Reyzin, and Scott Russell. Upper and lower bounds on black-box steganography. *Journal of Cryptology*, 22:365–394, 2009. 5

[Fow23] Geoffrey A. Fowler. We tested a new chatgpt-detector for teachers. it flagged an innocent student. *The Washington Post*, April 2023. 3

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986. 6

[GKVZ22] Shafi Goldwasser, Michael P Kim, Vinod Vaikuntanathan, and Or Zamir. Planting undetectable backdoors in machine learning models. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 931–942. IEEE, 2022. 6

[GSR19] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*, 2019. 4

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. 6

[HMW07] Nicholas Hopper, David Molnar, and David Wagner. From weak to strong watermarking. In *Theory of Cryptography: 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007. Proceedings 4*, pages 362–382. Springer, 2007. 6

[HvAL09] Nicholas J. Hopper, Luis von Ahn, and John Langford. Provably secure steganography. *IEEE Trans. Computers*, 58(5):662–676, 2009. 5, 13

[JAML20] Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks VS Lakshmanan. Automatic detection of machine generated text: A critical survey. *arXiv preprint arXiv:2011.01314*, 2020. 4

[Jan18] Svante Janson. Tail bounds for sums of geometric and exponential variables. *Statistics & Probability Letters*, 135:1–6, 2018. 17, 18

[Jim23] Kayla Jimenez. Professors are using chatgpt detector tools to accuse students of cheating. but what if the software is wrong? *USA Today*, April 2023. 3

[KAAL23] Jan Hendrik Kirchner, Lama Ahmad, Scott Aaronson, and Jan Leike. New ai classifier for indicating ai-written text. `https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text`, January 2023. Accessed May 2023. 4

[KGW+23] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. *CoRR*, abs/2301.10226, 2023. 3, 4, 5, 7, 13, 27, 28

[KJGR21] Gabriel Kaptchuk, Tushar M. Jois, Matthew Green, and Aviel D. Rubin. Meteor: Cryptographically secure steganography for realistic distributions. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 1529–1548. ACM, 2021. 5

[KSK+23] 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. 4, 27

[LUY08] Thomas Lavergne, Tanguy Urvoy, and François Yvon. Detecting fake content with relative entropy scoring. *PAN*, 8:27–31, 2008. 4

[LYM+23] Weixin Liang, Mert Yuksekgonul, Yining Mao, Eric Wu, and James Zou. Gpt detectors are biased against non-native english writers. *arXiv preprint arXiv:2304.02819*, 2023. 4

[MLK+23] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *CoRR*, abs/2301.11305, 2023. 4

[MZ23] Travis Munyer and Xin Zhong. Deeptextmark: Deep learning based text watermarking for detection of large language model generated text. *arXiv preprint arXiv:2305.05773*, 2023. 4

[Ope23] OpenAI. tiktoken repository. `https://github.com/openai/tiktoken`, 2023. Accessed April 2023. 13

[QZL+23] Jipeng Qiang, Shiyu Zhu, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. Natural language watermarking via paraphraser-based lexical substitution. *Artificial Intelligence*, page 103859, 2023. 4

[SKB+23] Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. Can ai-generated text be reliably detected? *CoRR*, abs/2303.11156, 2023. 4

[Tia23] Edward Tian. gptzero update v1. `https://gptzero.substack.com/p/gptzero-update-v1`, January 2023. Accessed May 2023. 3, 4

[YAJK23] KiYoon Yoo, Wonhyuk Ahn, Jiho Jang, and Nojun Kwak. Robust natural language watermarking through invariant features. *arXiv preprint arXiv:2305.01904*, 2023. 4

[ZHR+19] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. *Advances in neural information processing systems*, 32, 2019. 4