Can We Watermark Low-Entropy LLM Outputs?

Noam Mazor
New York University, NY, USA

Andrew Morgan
Cornell Tech, New York, NY, USA

Rafael Pass
Technion, Haifa, Israel
Cornell Tech, New York, NY, USA
Tel Aviv University, Israel

Abstract

A recent and exciting thread of work focuses on developing methods for watermarking the output of large language models (LLMs). We focus on *provably undetectable* watermarking – that is, schemes that do not alter the output distribution of the LLM, yet enable embedding a watermark in the output that identifies the output as having been generated by the particular LLM. Furthermore, the watermark should be hard to remove by an adversary that may potentially edit, insert, or delete tokens from the watermarked output.

Indeed, recent work (Christ et al. [COLT'24], Christ et al. [CRYPTO'24], Golowich et al. [NeuroIPS'24]) shows how to develop such schemes that are *robust* against a constant fraction of substitutions, or even against a constant fraction of arbitrary edits.

These works, however, make strong assumptions on the amount of entropy present in the output of the LLM. Most notably, they all require *constant entropy rate* – that is, a constant fraction of the tokens in a sufficiently long substring of the output need to have empirical entropy at least $O(\log |T|)$, where T is the alphabet of tokens, and Golowich et al. additionally require T to be larger than the security parameter.

In this work, we consider the question of whether we can also watermark the outputs of LLMs when the per-token entropy is just a constant, discarding the dependence on the alphabet size or security parameter. In this regime, we construct:

- A watermarking scheme robust against random substitutions (assuming subexponential LPN, as in Christ et al. [CRYPTO'24])
- A watermarking scheme robust against random substitutions and random deletions, given either the additional heuristic assumption that the output of the LLM only introduces random errors (analogous to the assumption made by Christ et al. [CRYPTO'24]) or a construction of a pseudorandom error-correcting code robust to adversarial substitutions and random deletions.

2012 ACM Subject Classification Security and privacy → Cryptography

Keywords and phrases Cryptography, Generative Models, Watermarking, Pseudorandom Codes

Digital Object Identifier 10.4230/LIPIcs.FORC.2026.8

Related Version *Full Version*: https://arxiv.org/abs/2604.12051

Funding *Noam Mazor*: Research was partly done while visiting the Simons Institute.
Andrew Morgan: Research supported by AFOSR Award FA9550-24-1-0267.
Rafael Pass: Research supported in part by AFOSR Award FA9550-23-1-0312, AFOSR Award FA9550-24-1-0267, ISF Award 2338/23 and ERC Advanced Grant KolmoCrypt – 101142322. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, the AFOSR, the European Union or the European Research Council Executive Agency.

1 Introduction

With the advent of large language models (LLMs) and other tools for AI-based content generation in everyday society, and with recent advances making AI-generated content increasingly natural and human-like, retaining the ability to differentiate content generated by an AI from content generated by a human has become a prevalent and concerning issue. A natural approach to solving this problem – and indeed, an approach which multiple developers of LLMs have pledged to implement [2], is that of *watermarking*, wherein an LLM or other AI content generation algorithm is subtly altered to embed a hidden code, or "watermark" into the content it generates, in such a way that the code can be extracted later to detect that it was in fact generated by the watermarked AI.

There are several intuitive desiderata in such a scheme. First, the watermark should be *undetectable*, meaning that the watermarked content generation algorithm should not produce content that is noticeably different from content that would be generated with the same inputs but without the addition of the watermarking; that is, the distribution of the AI's outputs should not be biased by the addition of watermarking. This is usually accomplished by use of a *secret key* for watermark detection, such that the detection algorithm can use it to efficiently detect the watermark but an adversary not knowing the key cannot. Additionally, a watermark should be *robust* against adversaries who may wish to "hide" that watermarked content was AI-generated by editing the output in an attempt to remove the watermark; an ideal watermark should still be detectable even in an output edited via some combination of substitutions, insertions, or deletions from an adversary.

We focus on the setting of watermarking for LLMs, where the increasing prevalence and quality of practical LLMs has unsurprisingly prompted a wide variety of ideas for watermarking schemes and techniques. Works such as [5, 11, 12, 13] provide strong and empirically verifiable robustness guaranties, but do so by introducing a limited amount of bias into the distribution of the LLM's output, which is compounded over repeated use.

Undetectable Watermarking for LLMs. A stronger notion of undetectability was introduced by Christ, Gunn and Zamir [6]; this notion requires watermarked outputs of the LLM to be *computationally indistinguishable* from the original output of the LLM. Such watermarking schemes can be used repeatedly while still remaining undetectable (and without deteriorating the performance of the LLM). In this paper, we focus on such schemes.

[6] provide the first scheme that achieves this notion (assuming a minimal cryptographic hardness assumption of a one-way function); their scheme, however, requires large substrings (i.e., blocks) of text generated by the LLM to remain completely unchanged for the watermark to be recoverable. In other words, robustness only holds with respect to very limited transformations (e.g., not even changes to a small *fraction* of tokens).

Christ and Gunn [4] suggest a general approach for constructing watermarking schemes that satisfy both undetectability and robustness against larger classes of attack – and in particular those that can handle modification to a *constant fraction* of tokens – in two steps. First, one needs to construct a so-called *pseudorandom code* (PRC): a code which is both *pseudorandom*, in that a codeword is indistinguishable from true randomness to a computationally bounded adversary, and *robust*, in that, similar to an error-correcting code, a codeword can be recovered even after being manipulated in certain ways. Given this, a PRC codeword can be embedded in the output of the LLM to watermark it, and the watermark can then be detected by attempting to recover the codeword from the output. The pseudorandomness of the PRC provides the undetectability of the watermarking, while the watermarking scheme inherits the robustness from the code (modulo any errors which may be introduced by the embedding process).

Christ and Gunn then construct two PRCs with a binary alphabet: one which is robust against adversarial substitutions (assuming the subexponential hardness of LPN), and one which is robust against random deletions and substitutions. They also suggest an embedding scheme, specifically for LLMs with a binary alphabet of possible tokens, that operates by attempting to correlate the bit selected by the LLM with the respective bit of the PRC codeword whenever it is possible to do so without biasing the distribution of the LLM's selection. This, depending on which of their PRC constructions are leveraged, results in a watermarking scheme which is either secure against random substitutions, or, under the assumption that the errors introduced by the embedding procedure are random, against random deletions and substitutions; additionally, it can be generalized to watermark LLMs with non-binary sets of tokens by simply using binary encoding of the tokens.

The recent work of Golowich and Moitra [10] constructs a PRC robust against edit distance (i.e., adversarial deletions, insertions, and substitutions). In fact, they construct such a code robust to a surprisingly large fraction $(1 - o(1))$ of adversarial edits, with the caveat that it requires the size of the codewords' alphabet to be polynomially large in the security parameter.[1] They also present an embedding scheme for such codes, which works by assuming the LLM has a polynomially large alphabet of tokens, hashing these tokens to random tokens in the PRC's alphabet, and applying a similar approach to [4] to attempt to correlate the hashed tokens to the respective tokens in the codeword. Through this, they obtain a watermarking scheme with a stronger robustness guarantee, additionally circumventing the requirement of a heuristic assumption such as that of [4] because of the added robustness in the PRC, but at the cost of requiring a much larger alphabet of tokens (as, in particular, the PRC does not retain its robustness guarantees through reduction to binary).

Notably, however, the embedding schemes in both of these works are not guaranteed to perfectly embed the codeword; they require that the output of the LLM has high enough *empirical entropy* – that is, that a sufficient number of tokens are drawn from sufficiently diverse distributions – as otherwise it will be impossible to reliably correlate the LLM output (e.g., if it is a single fixed token) with a token in a pseudorandom codeword, creating substitution errors in the watermarking process and potentially interfering with robustness guarantees.

The scheme of [10] requires a constant fraction of the tokens in a sufficiently long substring of the LLM's output to have empirical entropy $O(\log |T|)$, where T is the set of possible tokens. (This assumption on the entropy of the tokens is somewhat inherent in their approach; indeed, when the entropy of each token is 1, one could not expect to have robustness against more than $1/2$ fraction of adversarial substitutions.[2]) Notably, because $|T|$ is required to be a large polynomial in the security parameter λ, this implies that the required entropy is $\Omega(\log \lambda)$.

The entropy requirement of [4] is more moderate, but requires additional assumptions: when the LLM's output is binary, it merely requires that a constant fraction of the tokens in a sufficiently long substring have $\Omega(1)$ empirical entropy. When translating a model with

[1] Specifically, each token in the alphabet needs to correspond to a different index of a codeword in an underlying binary PRC.

[2] For example, imagine a scenario in which we ask the LLM to output a sequence of tokens of length $2L$, each is "Zero" with probability $1/2$ or "One" with probability $1/2$. In this case, each token has entropy 1 given any prefix. On the other hand, by changing $1/2$ fraction of the tokens, we can change the output sequence to be any of $\sum_{i \leq L} \binom{2L}{i} = 2^{2L}/2$ possible sequences of distance at most L from the LLM output. This means that either the watermark can be removed by changing L tokens, or the scheme recognizes a random sequence as watermarked with probability at least $1/2$, contradicting soundness.

a large set T of potential tokens to binary, the naïve way to do it is to encode each token using $\log |T|$ bits. Using this approach, the entropy requirement of [4] would once again be $\Omega(\log |T|)$, but the model of random errors no longer makes sense if using this naive encoding (since random errors to tokens do not correspond to random errors in bit representation). As the authors suggest, one can overcome these issues by using a Huffman encoding of the token, according to some *a priori* distribution over T, and simply using the first bit of the encoding to embed the binary PRC. This, however, introduces the additional strong assumption that the encoder knows the a priori distribution of the LLM's output on T and that the output of the LLM is distributed closely to it (so that the first bit of the Huffman encoding of the token has high empirical entropy).

Thus, unless we make strong distributional assumptions on the output of the LLM, neither of the above approaches can be used when the entropy of most tokens is a small constant, and in particular when the entropy is close to a single bit (or even smaller).[3] Thus, in this work, we address the following question:

> *Is there an undetectable and robust watermarking scheme for LLM outputs with a large alphabet of tokens that preserves robustness even when almost all tokens have low entropy (e.g., entropy 1)?*

We remark that when only requiring an undetectable watermarking scheme for LLM with very limited robustness guarantees (i.e., the requirement that a sufficiently large enough block of the output is completely unchanged), then the original scheme of [6] indeed supports this low-entropy regime – in fact, they handle even sub-constant entropy. However, as far as we know, no known schemes provide any non-trivial notion of robustness with constant or better entropy.

Highlights of our results. Our main results will be to provide initial constructions of such watermarking schemes. Based on essentially the same hardness assumptions as those in [4], we construct a watermarking scheme which provides robustness guarantees against a broad class of random substitution attacks (and, given additional assumptions, random deletions) for even outputs with very low entropy; specifically, our scheme's only requirement for robustness is that a constant fraction of tokens in a sufficiently long substring of the output have a single bit of empirical entropy. An additional advantage of our approach is that, because our watermarking scheme models token generation at the token level rather than reducing it to a binary alphabet as [4] does, our robustness guarantees, similarly to those of [6] in the high-entropy regime, deal with a more practical and realistic class of attacks that modify the output at a token level (e.g., paraphrasing by replacing tokens with synonyms, or redacting tokens by replacing them with something deterministic); in contrast, the practical significance of attacks against a binary translation of the output tokens is far less clear.

1.1 Results

We present, for LLMs with a large token set, a PRC-based watermarking scheme that provides similar robustness guarantees to the scheme of [4] with a much weaker entropy requirement. In particular, in contrast to prior works which require output tokens (or substrings of tokens)

[3] Note that the approach of [4] works when the entropy of each *binary* token is one. We here want to consider the realistic case where the token set is significantly larger, yet each token has low entropy within that larger alphabet.

to have an amount of average empirical entropy that is at least logarithmic in the size of the alphabet of tokens, we show that even outputs satisfying only a very weak notion of "minimal empirical entropy" – which requires only that a constant fraction of the tokens in a sufficiently long substring of the output have a single bit of empirical entropy, independently of the size of the token alphabet – can be robustly watermarked without additional distributional assumptions.

We first consider the setting of robustness against random substitutions. Notably, it is not immediately clear how a "random substitution" is even defined in the case of an output that uses a large token alphabet rather than restricting to a binary setting where a substitution always implies a bit flip; we model such a transformation by allowing a theoretical adversary to a priori map each token in the alphabet to a distribution of "replacement" tokens, and subsequently replacing randomly selected tokens with a sample from the respective replacement distribution.[4] We show the following:

▶ **Theorem 1** (Informal). *If there exists a binary PRC which provides robustness against a* $(1/2 - \delta)$ *fraction of adversarial substitutions for any constant* $\delta > 0$*, then there exists a large token alphabet watermarking scheme that is computationally undetectable and, for any LLM output satisfying the minimal empirical entropy requirement, robust to a* $(1/2 - \epsilon)$ *fraction of random substitutions for any constant* $\epsilon > 0$*.*

Combining this with the PRC construction of [4], we obtain:

▶ **Corollary 2** (Informal). *Assuming subexponential hardness of LPN, there exists a large token alphabet watermarking scheme that is computationally undetectable and, for any LLM output satisfying the minimal empirical entropy requirement, robust to a* $(1/2 - \epsilon)$ *fraction of random substitutions for any constant* $\epsilon > 0$*.*

Additionally, we can extend this result to provide robustness to random substitutions and deletions, but, as in [4], a PRC that is robust to only *random* substitutions and random deletions is insufficient on its own. [4] leverages such a PRC but also relies on an additional heuristic assumption – to which the authors refer as the output text being "sufficiently variable" – which requires that the distribution of the errors to the PRC codeword introduced by the watermarking algorithm is "consistent" in that the errors can be modeled as a binary symmetric channel. We generalize this assumption to the case of a large token alphabet (which, notably, requires a generalization of the definition of a random substitution compared to the binary-alphabet case) by introducing the notion of a "natural" prompt, or a prompt given to the LLM that, over the distribution of all possible outputs and subsequent random substitution attacks, causes the errors to the PRC codeword introduced by the composition of the watermarking algorithm and the random substitution channel to likewise be modelable by a binary symmetric channel. Given this, we show:

▶ **Theorem 3** (Informal). *If there exists a binary PRC which provides robustness against a* $(1/2 - \delta)$ *fraction of adversarial substitutions and a* $(1 - \delta')$ *fraction of random deletions for any constants* $\delta, \delta' > 0$*, then there exists a large token alphabet watermarking scheme that is computationally undetectable and, for any natural prompt and LLM output satisfying the minimal empirical entropy requirement, robust to a* $(1/2 - \epsilon)$ *fraction of random substitutions and a* $(1 - \epsilon')$ *fraction of random deletions for any constants* $\epsilon, \epsilon' > 0$*.*

[4] This definition, as mentioned, captures several natural notions of substitution-based attacks such as paraphrasing or redaction.

We can again combine this with the corresponding PRC construction of [4] to obtain:

▶ **Corollary 4** (Informal). *Assuming subexponential hardness of LPN, there exists a large token alphabet watermarking scheme that is computationally undetectable and, for any natural prompt and LLM output satisfying the minimal empirical entropy requirement, robust to a* $(1/2 - \epsilon)$ *fraction of random substitutions and a* $(1 - \epsilon')$ *fraction of random deletions for any constants* $\epsilon, \epsilon' > 0$*.*

This notion of a "natural" prompt importantly rules out certain contrived or pathological examples of prompts whose outputs are known to be inherently difficult to robustly watermark, but may be practically restrictive. A way to bypass the necessity for these heuristic assumptions would be to strengthen the robustness of the underlying PRC to handle *adversarial* substitutions (and random deletions) as opposed to only random substitutions:

▶ **Theorem 5** (Informal). *If there exists a binary PRC which provides robustness against a* $(1/2 - \delta)$ *fraction of adversarial substitutions and a* $(1 - \delta')$ *fraction of random deletions for any constants* $\delta, \delta' > 0$*, then there exists a large token alphabet watermarking scheme that is computationally undetectable and, for any LLM output satisfying the minimal empirical entropy requirement, robust to a* $(1/2 - \epsilon)$ *fraction of random substitutions and a* $(1 - \epsilon')$ *fraction of random deletions for any constants* $\epsilon, \epsilon' > 0$*.*

While [10] demonstrates a PRC with the requisite robustness, their construction requires a large alphabet of tokens, making it unsuitable for our results because it would vastly increase the entropy requirement. Indeed, no known construction of binary-alphabet PRCs provides robustness against a large fraction of both adversarial substitutions and random deletions.

However, the recent PRC construction of [3] allows instead for robustness against a large fraction of adversarial substitutions and a constant fraction of *adversarial edits* – that is, insertions *and* deletions – but with the caveat that the constant itself is very small and depends on the output's entropy[5], and additionally that the construction itself is based on a non-standard "permuted codes" assumption. The authors demonstrate that they can use the framework of [4] to construct binary-alphabet watermarking robust to a constant fraction of adversarial edits based on similar entropy requirements to [4]; we in turn show as our final result that we can leverage our framework with their PRC construction to construct minimal-entropy watermarking for a large token alphabet that likewise guarantees robustness to a constant fraction of adversarial edits:

▶ **Theorem 6** (Informal). *If, for any* $\delta > 0$*, there exists a constant* $\epsilon > 0$ *and a binary PRC which provides robustness against a* $(1/2 - \delta)$ *fraction of adversarial substitutions and an* ϵ *fraction of adversarial edits, then, for any* $\delta' > 0$*, there exists a constant* ϵ' *and a large token alphabet watermarking scheme that is computationally undetectable and, for any LLM output having at least a* δ' *fraction of tokens with one bit of entropy, robust to an* ϵ' *fraction of adversarial edits.*

This comes with the caveat that the size of the substring required for "minimal entropy" is larger; in particular, since a theoretical adversary able to make a constant fraction of arbitrary edits could decide to target a sufficient-entropy substring of any output, we require that at least a (larger) constant fraction of the entire output have sufficiently many tokens with a bit of empirical entropy. Combined with the construction of [3], which in particular is parameterized such that ϵ is $\tilde{O}(\delta^3)$ in the above statement, this gives:

[5] That is, it is constant only assuming that the output has a constant entropy average for a sufficiently long substring.

		Robustness Guarantees			
Result	**Entropy**	**Subst.**	**Edits**	**Adv.?**	**Assumptions Required**
[4]	$O(\log \lvert T\rvert)$	$1/2-\epsilon$			subexponential LPN
	$O(\log \lvert T\rvert)$	$1/2-\epsilon$	$1-\epsilon$ (del.)		subexponential LPN + heuristic
[10]	$\text{poly}(\log(\lambda))$		$1-\epsilon$	✓	local weak PRFs
[3]	$O(\log \lvert T\rvert)$		ϵ	✓	permuted codes
Cor. 1	$O(1)$	$1/2-\epsilon$			subexponential LPN
Cor. 2	$O(1)$	$1/2-\epsilon$	$1-\epsilon$ (del.)		subexponential LPN + heuristic
Cor. 3	$O(1)$	$1/2-\epsilon$	$1-\epsilon$ (del.)		PRC with adversarial subst. +del. robustness
Cor. 4	$O(1)$ overall		ϵ	✓	permuted codes

Figure 1 Summary of our results compared to related results in terms of per-token entropy requirement (over a sufficiently long substring unless otherwise stated); robustness to (random or adversarial) substitutions, insertions, and deletions; and required assumptions.

▶ **Corollary 7** (Informal). *Assuming the "permuted codes" assumption (Conjecture 3.1) of [3], then, for any $\delta' > 0$, there exists a constant $\epsilon' = \tilde{O}((\delta')^4)$ and a large token alphabet watermarking scheme that is computationally undetectable and, for any LLM output having at least a δ' fraction of tokens with one bit of entropy, robust to an ϵ' fraction of adversarial edits.*

For clarity, we concisely summarize our results and provide comparison to related results in table form in Figure 1.

Our approach in a nutshell. At a high level, through using a binary PRC but allowing the generative model output to have a large token alphabet, our approach provides a natural – and advantageous – "middle ground" between the watermarking scheme of [4], which deals with entirely binary tokens, and the watermarking scheme of [10], which deals entirely with a large token alphabet.

The watermarking algorithms constructed by [4] are designed for generative models which produce a binary output, and they attempt to embed the codeword of the (binary) PRC directly into the output of the model. The most direct way to transform this into watermarking for a model which outputs tokens from a larger alphabet is, as the authors propose, by simply converting the tokens into a binary representation. As we have mentioned, this comes at the cost of increasing the entropy requirement per output token proportionally to the token's binary length. However, a more concerning issue with this approach is that this transformation does not preserve the token structure of the model's output and thus dramatically changes the class of attacks against which the watermarking is robust; random substitutions or deletions of single bits in a binary encoding are incomparable with the more practical class of attacks (e.g., paraphrasing or redaction) which manipulate the output on the token level in an attempt to transform a model's watermarked output into a similar but non-watermarked output.

Our approach preserves the token structure of the output by, rather than attempting to embed the PRC codeword directly into the output, hashing the output tokens into bits and attempting to embed the codeword into the *hashes* of the output tokens; as a result,

the output's structure is preserved, as each token corresponds to exactly one bit of the PRC codeword. [10] uses a similar approach, but rather than hashing the output tokens to binary and embedding a binary PRC codeword, they embed a codeword of a PRC with a polynomial-sized alphabet and hash the output tokens into that alphabet. This allows them to obtain significantly stronger robustness guarantees, but at the cost of dramatically increasing the per-token entropy requirement to be logarithmic in the security parameter, since successful embedding requires the hash of the output token distribution to be close to uniform over the PRC's alphabet. In contrast, our approach has more limited robustness, but achieves it with a far more reasonable requirement of constant entropy because we restrict the PRC to a binary alphabet.

1.2 Comparison with Related Work

Our work builds upon the previous undetectable watermarking schemes of [4], [6], and [10] by providing a scheme with both a low entropy requirement for the tokens generated by the model and strong robustness guarantees against attacks that, notably, manipulate tokens in the natural output directly rather than relying on a reduction to a binary alphabet. To explain what we mean by this and why it is advantageous, recall that, in the results of [4], which focus on watermarking over a binary alphabet (i.e., every token is 0 or 1), one can simply model a p fraction of random substitutions as a binary symmetric channel, i.e., flipping each bit independently with probability p.

To extend their results to an output drawing from a large alphabet of tokens, the authors propose translating each token to a binary encoding, either by using a fixed-length encoding or a more efficient variable-length (e.g., Huffman) encoding, and generating a watermarked binary (translated) output by using a modified generative model that generates the binary translations of its output tokens. While they indeed prove robustness against random *binary* substitutions to bits in the translated output, this definition of robustness loses its meaning entirely when one considers the more natural definition of substitutions made to tokens in the untranslated output, as mentioned in [4]. For instance, even using a fixed-length encoding, random binary substitutions change the corresponding untranslated tokens in ways that are not necessarily random and depend on which bits are affected; worse still, using a variable-length encoding means that a single random substitution to the translated binary output can potentially alter *every* untranslated token in the entire output. In the other direction, since each token in the untranslated output corresponds to multiple bits in the binary translation, changing any token will correspond to modifying a correlated block of bits in a way that is not necessarily even independent between the bits in the block.

In contrast to [4], we show robustness against modifications specifically to tokens in the output, which we remark is a far more natural definition as it preserves the structural features of the output that would be discarded by manipulating a translation of the output to a binary encoding. While it is not initially clear how one would even define a "random substitution" for the case of a large set of tokens – in particular, while it is fairly straightforward to start with a similar definition to a binary symmetric channel by modifying each token with some independent probability p, it is less clear how each token would be modified given a large alphabet of tokens – we show that we can achieve robustness against even a class of random substitutions where an adversary may, a priori (i.e., before seeing the output), choose, for each token in the alphabet, an arbitrary distribution of tokens to replace an occurrence of that token if it is selected randomly to be substituted. This captures some basic semantic attacks such as limited forms of substitution or paraphrasing where an adversary may opt to replace some randomly chosen words with synonyms in an attempt to remove a watermark, giving robustness to a variety of practical attacks.

The approaches of [6] and [10] also consider outputs over a large token alphabet as we do. [6] considers a weaker form of robustness, *substring completeness*, which requires that the watermark be detectable if the detection algorithm is given any unmodified substring of sufficient empirical entropy from the output; while they also consider reducing to a binary alphabet via fixed-length encoding in their construction of substring-complete watermarking, we note that this notion, unlike the stronger notion of robustness, is preserved in the translation to a larger token alphabet (as, clearly, an unmodified substring in the binary representation is likewise unmodified when translated to tokens). That said, a substring-complete watermarking scheme requires that a contiguous substring of the output be unmodified to be detectable, and, as the entropy requirement for such a substring is dependent on the security parameter, such a scheme cannot even provide robustness against any constant fraction of random edits.

Meanwhile, [10] provides a very strong notion of *edit distance robustness* against any constant fraction of even adversarial substitutions, deletions, or insertions; they do so by transforming a binary PRC robust to adversarial substitutions into a large-alphabet PRC with this improved notion of edit distance robustness, and applying a similar approach to that of [4] and our work to the transformed PRC. Specifically, their construction of an edit-distance-robust PRC works by *indexing* the original PRC – roughly speaking, using an alphabet whose size is equal to the *block length* of the original PRC, and letting each token in the expanded alphabet represent the presence of a 1 in its corresponding position in the respective block of the underlying PRC, so that insertions or deletions in the transformed PRC effectively become substitutions in the original. Of course, the drawback of this approach, as we have mentioned, is that the extremely large required alphabet size of the PRC leads to a correspondingly large entropy requirement (i.e., logarithmic in the security parameter) for the generative model's output, something which we explicitly seek to avoid in our results because natural language generally possesses large amounts of low-entropy tokens.

PRC constructions. As mentioned above, [4] construct PRCs robust against adversarial substitutions or random substitutions and deletions, assuming the hardness of LPN. Under the same assumption, [1] showed the existence of PRC with stronger security guarantees, i.e., CCA security and adaptive robustness. [9] construct PRCs against adversarial substitutions from weaker assumptions, as well as unconditionally-secure PRC with pseudorandomness against space-bounded adversaries. Most recently, [3] construct PRCs featuring subexponential security, a stronger notion of "strongly adaptive robustness" which guarantees adaptive robustness even against an adversary with full knowledge of the watermarking key, and robustness to a constant fraction of (even arbitrary or adversarial) edits without the prohibitive entropy requirement of [10], albeit based on a non-standard "permuted codes" assumption. In the realm of negative results, [8, 7] recently showed that PRCs cannot be constructed from a wide-range of cryptographic primitives in a black-box manner.

2 Definitions

Given a discrete distribution D, we define the (binary) *entropy* of D to be the quantity $H(D) = \sum_{x \in D} -D(x) \log_2 D(x)$. Furthermore, given a selection $x \leftarrow D$ from a distribution, we define the *empirical entropy* of x in D as the quantity $H_{\text{emp}}(x, D) = -\log_2 D(x)$.

2.1 Error Channels

Towards defining robustness for watermarking generative models whose outputs are large alphabets, we first must define generalized notions of error channels over families of large token alphabets. Formally:

▶ **Definition 8.** *Given some vocabulary of tokens $\mathcal{V}$, a* ***channel*** *is a (potentially randomized) algorithm $\mathcal{E} : \mathcal{V}^* \to \mathcal{V}^*$ that perturbs an output using tokens from $\mathcal{V}$.*

Additionally, given a family of vocabularies $\mathcal{V}(n)$ parameterized by a security parameter $n \in \mathbb{N}$, we may consider a ***family of channels*** *$\mathcal{E} = \{\mathcal{E}(n)\}_{n\in\mathbb{N}}$ where, for each n, $\mathcal{E}(n)$ is a channel over vocabulary $\mathcal{V}(n)$.*

We next define standard notions of random substitution and deletion channels:

▶ **Definition 9.** *Given a family of vocabularies $\mathcal{V}(n)$ parameterized by a security parameter $n \in \mathbb{N}$, we define the* ***random deletion channel family*** *$\mathcal{E}_{del}(p) = \{\mathcal{E}(p, n)\}_{n\in\mathbb{N}}$ where $\mathcal{E}(p, n)$, given constant $p \in [0, 1]$ and security parameter n, takes as input a series of tokens $\overrightarrow{t} \in \mathcal{V}(n)^*$ and, for each $t \in \overrightarrow{t}$ in order, appends it to $\overrightarrow{t}'$ with probability $1 - p$, then outputs $\overrightarrow{t}'$.*

▶ **Definition 10.** *Given a family of vocabularies $\mathcal{V}(n)$ parameterized by a security parameter $n \in \mathbb{N}$ and a family of functions $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$, we define the* ***random substitution channel family*** *$\mathcal{E}_{sub}(f, p) = \{\mathcal{E}(f, p, n)\}_{n\in\mathbb{N}}$ where $\mathcal{E}(f, p, n)$, given constant $p \in [0, 1]$ and security parameter n, takes as input a series of tokens $\overrightarrow{t} \in \mathcal{V}(n)^*$ and outputs $\overrightarrow{t}' \in \mathcal{V}(n)^*$ where $t'_i = \{x : x \leftarrow f_n(t_i)\}$ with probability p and $t'_i = t_i$ otherwise.*

For the case of a binary alphabet $\mathcal{V} = \{0, 1\}$, we also define the analogous ***binary symmetric channel*** *$\mathcal{E}_{bin}(p)$, which takes as input a series of tokens $\overrightarrow{t} \in \{0, 1\}^*$ and outputs $\overrightarrow{t}' \in \{0, 1\}^*$ where $t'_i = 1 - t_i$ with probability p and $t'_i = t_i$ otherwise.*

Notably, the definition of a random substitution channel for a non-binary vocabulary of tokens allows for instances of any token to be substituted with a (potentially arbitrarily determined) distribution over other tokens, capturing a variety of attacks such as simple paraphrasing attacks or randomized censorship that are practical when considering a large token alphabet but cannot be captured by random substitutions in a binary setting such as that of [4].

Lastly, we define notions of arbitrary (potentially adversarial) channels in the binary and large-alphabet settings:

▶ **Definition 11.** *We refer to a channel $\mathcal{E}$ as a* ***p-bounded substitution channel*** *over a vocabulary $\mathcal{V}$ if, given some input $\overrightarrow{t} \in \mathcal{V}^*$, $\mathcal{E}$ always outputs $\overrightarrow{t}' \in \mathcal{V}^*$ where $len(\overrightarrow{t}') = len(\overrightarrow{t})$ and, letting $\ell = len(\overrightarrow{t})$, $t'_i \neq t_i$ for at most $p\ell$ indices $i \in [\ell]$.*

If $\mathcal{V} = \{0, 1\}$, we refer to such a channel as a ***p-bounded binary substitution channel****.*

▶ **Definition 12.** *Given strings x and x', let $D_{edit}(x, x')$ be the* edit distance *between x and x', or the minimum number of insertions and deletions required to transform x into x' (or, symmetrically, vice versa). We refer to a channel $\mathcal{E}$ as a* ***p-bounded binary edit channel*** *over a vocabulary $\mathcal{V}$ if, given some input $\overrightarrow{t} \in \mathcal{V}^*$, $\mathcal{E}$ always outputs $\overrightarrow{t}' \in \mathcal{V}^*$ where $D_{edit}(\overrightarrow{t}, \overrightarrow{t}') \leq p \cdot len(\overrightarrow{t})$.*

If $\mathcal{V} = \{0, 1\}$, we refer to such a channel as a ***p-bounded binary edit channel****.*

That is, a substitution channel is p-bounded if it makes at most a p fraction of substitution edits to its input, and, respectively, a p-bounded edit channel makes at most a p fraction of insertion or deletion edits, though these edits may be determined adversarially or arbitrarily.

2.2 Generative Models and Watermarking

We define generative models over large alphabets following the framework of [10], as computationally bounded algorithms which, given some input (prompt) and a sequence of tokens in a vocabulary $\mathcal{V}$ generated thus far, returns a probability distribution over $\mathcal{V}$ representing the next token to be generated.

▶ **Definition 13** ([10]). *A **generative model** is an efficient algorithm* $\mathsf{Generate}$ *over some vocabulary* $\mathcal{V}$ *that, given a prompt and sequence of tokens* $\mathsf{prompt}, \overrightarrow{t} \in \mathcal{V}^*$*, outputs a probability distribution* $D \in \Delta(\mathcal{V})$*.*

*A **generative model family** is a family of generative models* $\{\mathsf{Generate}(n)\}_{n\in\mathbb{N}}$ *parameterized by a security parameter* n*, such that* $\mathsf{Generate}(n)$ *operates on vocabulary* $\mathcal{V}(n)$ *where* $|\mathcal{V}(n)|$ *is bounded above by some polynomial* $p(n)$*, and so that there exists symbol* End *belonging to* $\mathcal{V}(n)$ *for every* $n \in \mathbb{N}$ *and polynomial* $m(n)$ *(the **maximum output length**) such that if* $|\overrightarrow{t}| \geq m(n) - 1$ *then* $D \leftarrow \mathsf{Generate}(n)(\mathsf{prompt}, \overrightarrow{t})$ *must have* $D(\mathsf{End}) = 1$*.*

To *watermark* the output of a generative model, then, is to generate an output in such a way that it can be determined, using a secret key, that the output came from the watermarked model. Ideally, such a watermark preserves the distribution of the output from the model, is undetectable to parties not possessing the secret key, and is also robust to manipulation by an adversary attempting to remove the watermark. Formally:

▶ **Definition 14** ([10]). *Assume a family of generative models* $\mathcal{M} = \{\mathsf{Generate}(n)\}_{n\in\mathbb{N}}$ *with vocabulary* $\mathcal{V}(n)$ *and maximum output length* $m(n)$*, and consider a tuple of efficient algorithms* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$*, where:*

- $\mathsf{Gen}(1^n) \to \mathsf{sk}$ *takes as input a security parameter* n *and outputs a secret key* sk*,*
- $\mathsf{Watermark}_{\mathsf{sk}}(1^n, \mathsf{prompt}) \to \overrightarrow{t}$ *takes as input the security parameter* n*, secret key* sk*, and prompt* prompt*, and outputs a sequence of tokens* $\overrightarrow{t} \in \mathcal{V}(n)^{m(n)}$*.*[6]
- $\mathsf{Detect}_{\mathsf{sk}}(1^n, \overrightarrow{t}) \to b$ *takes as input the security parameter* n*, a secret key* sk*, and a sequence of tokens* $\overrightarrow{t} \in \mathcal{V}(n)^{m(n)}$*, and outputs boolean* $b \in \{\mathsf{true}, \mathsf{false}\}$ *corresponding to whether a watermark was detected in* $\overrightarrow{t}$*.*

We refer to $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *as a **secure watermarking scheme** for* $\mathcal{M}$ *if the following properties hold:*

- ***Soundness:** For any* $\overrightarrow{t} \in \mathcal{V}(n)^{m(n)}$*, there is negligible* $\epsilon(\cdot)$ *such that:*

$$\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathsf{Detect}_{\mathsf{sk}}(1^n, \overrightarrow{t}) = \mathsf{true}] \leq \epsilon(n)$$

- ***Undetectability:** For any probabilistic polynomial-time oracle-aided adversary* $\mathcal{A}(\cdot)$*, there is negligible* $\epsilon(\cdot)$ *such that:*

$$|\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathcal{A}^{\mathsf{Watermark}_{\mathsf{sk}}(1^n,\cdot)}(1^n) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\mathsf{Generate}}(1^n,\cdot)}(1^n) = 1]| \leq \epsilon(n)$$

where $\mathsf{Watermark}$ *is simply an oracle that outputs the result of* $\mathsf{Watermark}$ *given a prompt* prompt*, and* $\mathcal{O}_{\mathsf{Generate}}$ *is an oracle that on input* prompt*, generates a complete output from* $\mathcal{M}$ *by outputting the result of the following procedure:*

1. *Initialize* $\overrightarrow{t} = ()$*.*
2. *For* $i \in [m(n)]$*, generate* $t_i \leftarrow \mathsf{Generate}(\mathsf{prompt}, \overrightarrow{t})$ *by drawing from the respective distribution output from* $\mathsf{Generate}$*, and append* t_i *to* $\overrightarrow{t}$*. If* $t_i = \mathsf{End}$ *then pad the remainder of* $\overrightarrow{t}$ *with* End *and terminate, returning* $\overrightarrow{t}$*.*

Furthermore, for some channel family $\mathcal{E} = \{\mathcal{E}(n)\}_{n\in\mathbb{N}}$ *where* $\mathcal{E}(n) : \mathcal{V}(n)^* \to \mathcal{V}(n)^*$*, we refer to* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *as **robust with respect to** $\mathcal{E}$ if there is negligible* $\epsilon(\cdot)$ *such that for any prompt* prompt*:*

$$\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n), \overrightarrow{t} \leftarrow \mathsf{Watermark}_{\mathsf{sk}}(1^n, \mathsf{prompt}) : \mathsf{Detect}_{\mathsf{sk}}(1^n, \mathcal{E}(n)(\overrightarrow{t})) = \mathsf{false}] \leq \epsilon(n)$$

We may additionally restrict this definition to a set of prompts Π *(where robustness holds if the above is true for prompts* $\mathsf{prompt} \in \Pi$*).*

[6] As in [10], if the model's output is shorter than the maximum length $m(n)$, we assume that the remainder of $\overrightarrow{t}$ is padded by repeating a special "ending" token End.

In order for an output to be undetectably and robustly watermarkable, it also must have some amount of entropy. We consider a notion of "entropy-restricted substring robustness" where we only require robustness to hold for outputs which have a sufficiently long substring containing a certain fraction of tokens with sufficiently high empirical entropy:

▶ **Definition 15.** *Assume a family of generative models* $\mathcal{M} = \{\mathsf{Generate}(n)\}_{n\in\mathbb{N}}$ *with vocabulary* $\mathcal{V}(n)$ *and maximum output length* $m(n)$*, and let* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *be a secure watermarking scheme for* $\mathcal{M}$*. Then, for a channel family* $\mathcal{E} = \{\mathcal{E}(n)\}_{n\in\mathbb{N}}$ *where* $\mathcal{E}(n) : \mathcal{V}(n)^* \to \mathcal{V}(n)^*$*, we refer to* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *as satisfying* ***$(L(n), \delta, h)$-entropy-restricted substring robustness with respect to $\mathcal{E}$*** *if there is negligible* $\epsilon(\cdot)$ *such that for any prompt* prompt *(which may be specified to be restricted to a set* Π*):*

$$\begin{aligned}\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n), \overrightarrow{t} \leftarrow \mathsf{Watermark}_{\mathsf{sk}}(1^n, \mathsf{prompt}) : &\exists j, k \in [\mathit{len}(\overrightarrow{t})] : k - j \geq L(n) \\ &\text{and } \#\{i \in [j,k] : H_{emp}(t_i, \mathsf{Generate}(\mathsf{prompt}, \overrightarrow{t}_{\leq i-1})) \geq h\} \geq \delta(k-j) \\ &\text{and } \mathsf{Detect}_{\mathsf{sk}}(1^n, \mathcal{E}(\overrightarrow{t})) = \mathsf{false}] \leq \epsilon(n)\end{aligned}$$

where $\overrightarrow{t}_{\leq i-1}$ *denotes the substring consisting of the first* $i-1$ *tokens in* $\overrightarrow{t}$*.*

That is, this definition requires that robustness holds only if there exists a substring of the output with length at least $L(n)$ where at least a δ fraction of the output tokens were generated with at least h bits of empirical entropy with respect to the generative model's output distribution.

2.3 Pseudorandom Error-Correcting Codes

Closely related to watermarking generative algorithms is the concept of *pseudorandom error-correcting codes*, introduced in [4]. These are a strengthening of the classical notion of error-correcting codes with the added requirement that the code for a given message should be *pseudorandom*, or indistinguishable from uniform randomness by a computationally bounded adversary.

▶ **Definition 16** ([4])**.** *Consider a tuple of efficient algorithms* $\mathsf{PRC} = (\mathsf{Gen}, \mathsf{Encode}, \mathsf{Decode})$*, where:*

- $\mathsf{Gen}(1^n) \to \mathsf{sk}$ *takes as input a security parameter* n *and outputs a secret key* sk*,*
- $\mathsf{Encode}_{\mathsf{sk}}(m) \to c$ *takes as input a secret key* sk *and message* $m \in \mathcal{V}^k$ *for some vocabulary* $\mathcal{V} = \mathcal{V}(n)$ *and input length* $k = k(n)$*, and outputs a code* $c \in \mathcal{V}^\ell$ *for some code length* $\ell = \ell(n)$*.*
- $\mathsf{Decode}_{\mathsf{sk}}(c) \to m$ *takes as input a secret key* sk *and code* $c \in \mathcal{V}^\ell$*, and outputs either a message* $m \in \mathcal{V}^k$ *or* $\perp$ *to indicate that there is no valid decoding.*

Given a channel $\mathcal{E} : \mathcal{V}^* \to \mathcal{V}^*$ *that introduces some error into the code, we call* PRC *a* ***pseudorandom error-correcting code*** *(PRC) with robustness to* $\mathcal{E}$ *if the following properties are satisfied for any* $n \in \mathbb{N}$ *and the respective* $\mathcal{V}(n)$*,* $k(n)$*,* $\ell(n)$*:*

- ***Robustness (with respect to $\mathcal{E}$):*** *For any* $m \in \mathcal{V}^k$*, there is negligible* $\epsilon(\cdot)$ *such that:*

$$\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n), c \leftarrow \mathsf{Encode}_{\mathsf{sk}}(m) : \mathsf{Decode}_{\mathsf{sk}}(\mathcal{E}(c)) \neq m] \leq \epsilon(n)$$

- ***Soundness:*** *For any* $c \in \mathcal{V}^*$*, there is negligible* $\epsilon(\cdot)$ *such that:*

$$\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathsf{Decode}_{\mathsf{sk}}(c) \neq \perp] \leq \epsilon(n)$$

- ***Pseudorandomness:*** *For any probabilistic polynomial-time oracle-aided adversary* $\mathcal{A}(\cdot)$*, there is negligible* $\epsilon(\cdot)$ *such that, letting* $\mathcal{O}_{\mathsf{Encode}}(\mathsf{sk})$ *be an oracle that returns* $\mathsf{Encode}_{\mathsf{sk}}(m)$ *given message* m *and* $\mathcal{O}_{\mathsf{Unif}}$ *be an oracle that implements a uniformly randomly selected function* $r : \mathcal{V}^k \to \mathcal{V}^\ell$*:*

$$|\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathcal{A}^{\mathcal{O}_{\mathsf{Encode}}(\mathsf{sk})}(1^n) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\mathsf{Unif}}}(1^n) = 1]| \leq \epsilon(n)$$

In this work we consider only *zero-bit binary* PRCs, where $\mathcal{V}(n) = \{0,1\}$ and $k = 0$ – that is, there is only a single possible input (which by convention we label as 1), and the code is a binary string of length $\ell(n)$.

3 Protocol and Results

Input: Security parameter n, generative model $\mathsf{Generate} = \mathsf{Generate}(n)$ over token vocabulary $\mathcal{V} = \mathcal{V}(n)$ with maximum output length $m = m(n)$ (belonging to family $\{\mathsf{Generate}(n)\}_{n\in\mathbb{N}}$), prompt prompt, error-correcting pseudorandom code PRC with code length $\ell = \ell(n)$ such that $m(n) = \omega(n\ell(n))$.
Output: Secret key sk for PRC and a tuple of functions $\overrightarrow{h} \in \mathcal{H}^{\lceil m/\ell \rceil}$, where $\mathcal{H}$ is the set of functions $h : \mathcal{V} \to \{0,1\}$.

Protocol:
1. Generate $\mathsf{sk} \leftarrow \mathsf{PRC.Gen}(1^n)$.
2. Generate $\overrightarrow{h}$ by, for $i \in [\lceil m/\ell \rceil]$, randomly sampling $h_i \leftarrow \mathcal{H}$.
3. Return sk and $\overrightarrow{h}$.

Figure 2 Protocol Gen for generating the secret information used in the watermarking protocol.

Here, we formally restate the theorems presented in the introduction and present the protocol with which we prove them. To begin, we demonstrate a watermarking protocol which provides soundness, undetectability, and (given any output with "minimal empirical entropy", or at least a constant fraction of tokens in a sufficiently long substring having a single bit of empirical entropy) robustness against any random substitution channel $\mathcal{E}_{\text{sub}}(f, 1/2 - \epsilon')$ for constant $\epsilon' > 0$, given the existence of a binary PRC with robustness to any (arbitrary) $(1/2 - \epsilon)$-bounded substitution channel for $\epsilon > 0$. Formally:

▶ **Theorem 17.** *If, for any constant* $\epsilon \in (0, 1/2]$*, there exists a pseudorandom code* PRC *(having block length* $\ell(\epsilon, n)$*) with robustness against any* $(1/2-\epsilon)$*-bounded binary substitution channel, then, for any generative model family* $\mathcal{M}$ *with token alphabet family* $\mathcal{V} = \{\mathcal{V}(n)\}_{n\in\mathbb{N}}$*, family of functions* $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$*, and constants* $\epsilon' \in (0, 1/2]$ *and* $\delta \in (0, 1]$*, there exists a secure watermarking scheme* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *for* $\mathcal{M}$ *that satisfies* $(n\ell(\epsilon'\delta/17, n), \delta, 1)$*-entropy-restricted substring robustness with respect to the random substitution channel* $\mathcal{E}_{sub}(f, 1/2 - \epsilon')$*.*

Combining this with the respective PRC construction from [4] gives:

▶ **Corollary 18.** *If Assumption 1 of [4] is true*[7]*, then, letting* $\ell(\epsilon, n)$ *be the block length of the respective PRC (in Theorem 1 and Theorem 2 of [4]) required to achieve robustness against any* $(1/2 - \epsilon)$*-bounded binary substitution channel, for any generative model family*

[7] This assumption, at a high level, requires either subexponential hardness of LPN or polynomial hardness of both LPN and a low-density planted XOR problem.

Input: Security parameter n, generative model $\mathsf{Generate} = \mathsf{Generate}(n)$ over token vocabulary $\mathcal{V} = \mathcal{V}(n)$ with maximum output length $m = m(n)$ (belonging to family $\{\mathsf{Generate}(n)\}_{n\in\mathbb{N}}$), prompt prompt, error-correcting pseudorandom code PRC with code length $\ell = \ell(n)$, secret key sk for PRC, and tuple of functions $\overrightarrow{h} \in \mathcal{H}^{\lceil m/\ell \rceil}$, where $\mathcal{H}$ is the set of all functions $h : \mathcal{V} \to \{0,1\}$.
Output: Watermarked output $\overrightarrow{t} \in \mathcal{V}^*$.

Protocol:

1. Initialize $\overrightarrow{t} = ()$, $i = 1$, and $j = 1$.
2. Draw $c \leftarrow \mathsf{PRC.Encode}_{\mathsf{sk}}(1^n)$. Let $h = h_j$, the j^{th} element of $\overrightarrow{h}$.
3. Let $D = \mathsf{Generate}(\mathsf{prompt}, \overrightarrow{t})$.
4. Randomly and independently sample two tokens $t_0, t_1 \leftarrow D$. Generate the next token t by selecting one of these two as follows:
   a. If $h(t_0) = h(t_1)$, let $t = t_b$ for a uniformly random $b \leftarrow \{0,1\}$.
   b. Otherwise, if $h(t_0) \neq h(t_1)$, let t be whichever of the two samples t_b has $h(t_b) = c_i$ (where c_i is the i^{th} bit of c). That is, if $h(t_0) = 0$, let $t = t_{c_i}$, and otherwise let $t = t_{1-c_i}$.
5. Append t to $\overrightarrow{t}$, then:
   a. If $t = \mathsf{End}$, terminate (padding the remainder of $\overrightarrow{t}$ to length $m(n)$ with End tokens if necessary) and output $\overrightarrow{t}$.
   b. Otherwise, if $i = \ell$, start a new block by returning to step (2), incrementing j, and resetting i to 1.
   c. Otherwise, generate the next token of the current block by incrementing i and returning to step (3).

Figure 3 Protocol $\mathsf{Watermark}$ for watermarking a generative model.

$\mathcal{M}$ with token alphabet family $\mathcal{V} = \{\mathcal{V}(n)\}_{n\in\mathbb{N}}$, family of functions $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$, and constants $\epsilon' \in (0, 1/2]$ and $\delta \in (0, 1]$, there exists a secure watermarking scheme $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ for $\mathcal{M}$ that satisfies $(n\ell(\epsilon'\delta/17, n), \delta, 1)$-entropy-restricted substring robustness with respect to the random substitution channel $\mathcal{E}_{sub}(f, 1/2 - \epsilon')$.

We provide the proofs in Appendix A. Next, we would like to extend the above result to demonstrate that $\mathsf{Watermark}$ is additionally robust to a large fraction of random deletions if the same also holds for PRC. However, in order to leverage the corresponding PRC construction of [4], we require a heuristic assumption, as discussed in the introduction, that a given prompt is "natural". Formally:

▶ **Definition 19.** *We say that a prompt* prompt ***produces consistently distributed errors*** *for a watermarking scheme* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *with respect to a substitution channel family $\mathcal{E}$ if there exists some constant p such that, taken over the randomness of* $\mathsf{Watermark}$ *and the channel $\mathcal{E}$, each token t of the output, after being generated by* $\mathsf{Watermark}$ *(given prompt* prompt*) according to some function h and bit c_i of codeword c from* PRC *and then modified according to $\mathcal{E}$, has an identical probability p, independent from that of other tokens, of having $h(t) \neq c_i$.*

Given this, we can present our following result that leverages this definition:

▶ **Theorem 20.** *If, for any constants $\epsilon_{sub} \in (0, 1/2]$ and $\epsilon_{del} \in (0, 1]$, there exists a pseudorandom code* PRC *(having block length $\ell(\epsilon_{sub}, \epsilon_{del}, n)$) with robustness against the composition of channels $\mathcal{E}_{bin}(\frac{1}{2} - \epsilon_{sub}) \circ \mathcal{E}_{del}(1 - \epsilon_{del})$, then, for any generative model family $\mathcal{M}$ with token*

Input: Security parameter n, message $\overrightarrow{t} \in \mathcal{V}^*$ (for $\mathcal{V} = \mathcal{V}(n)$), error-correcting pseudorandom code PRC with code length $\ell(n)$, secret key sk for PRC, and tuple of functions $\overrightarrow{h} \in \mathcal{H}^{\lceil m(n)/\ell(n) \rceil}$, where $\mathcal{H}$ is the set of functions $h : \mathcal{V} \to \{0,1\}$.
Output: Boolean b indicating whether a watermark was detected.

Protocol:
1. Let $\mathsf{len} = \min(|\overrightarrow{t}|, m(n))$. For $i \in [\mathsf{len}]$, $j \in [i, \min(i + \ell(n) - 1, \mathsf{len})]$, $k \in |\overrightarrow{h}|$:
   a. If $\mathsf{PRC.Decode}_{\mathsf{sk}}(h_k(\overrightarrow{t}_i), \ldots, h_k(\overrightarrow{t}_j)) \neq \perp$, return true.
2. If none of the above calls to $\mathsf{PRC.Decode}$ succeed, return false.

Figure 4 Protocol Detect for detecting the presence of a watermark in a given message.

alphabet family $\mathcal{V} = \{\mathcal{V}(n)\}_{n\in\mathbb{N}}$, family of functions $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$, and constants $\epsilon'_{sub} \in (0, 1/2]$, $\epsilon'_{del} \in (0, 1]$, and $\delta \in (0, 1]$, there exists a secure watermarking scheme $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ for $\mathcal{M}$ that, for any family Π of prompts where any $\mathsf{prompt} \in \Pi$ produces consistently generated errors for $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ with respect to the substitution channel $\mathcal{E}_{sub}(f, 1/2-\epsilon')$, satisfies $(n\ell(\epsilon'_{sub}\delta/17, \epsilon'_{del}, n), \delta, 1)$-entropy-restricted substring robustness over Π with respect to the composition of channels $\mathcal{E}_{sub}(f, 1/2-\epsilon'_{sub}) \circ \mathcal{E}_{del}(1-\epsilon'_{del})$.

Again, combining this with the respective PRC construction from [4] gives:

▶ **Corollary 21.** *If Assumption 1 of [4] is true, then, letting $\ell(\epsilon_{sub}, \epsilon_{del}, n)$ be the block length of the PRC given by Theorem 4 of [4] required to achieve robustness against the composition of channels $\mathcal{E}_{bin}(\frac{1}{2} - \epsilon_{sub}) \circ \mathcal{E}_{del}(1 - \epsilon_{del})$, for any generative model family $\mathcal{M}$ with token alphabet family $\mathcal{V} = \{\mathcal{V}(n)\}_{n\in\mathbb{N}}$, family of functions $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$, and constants $\epsilon'_{sub} \in (0, 1/2]$, $\epsilon'_{del} \in (0, 1]$, and $\delta \in (0, 1]$, there exists a secure watermarking scheme $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ for $\mathcal{M}$ that, for any family Π of prompts where any $\mathsf{prompt} \in \Pi$ produces consistently generated errors for $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ with respect to the substitution channel $\mathcal{E}_{sub}(f, 1/2 - \epsilon')$, satisfies $(n\ell(\epsilon'_{sub}\delta/17, \epsilon'_{del}, n), \delta, 1)$-entropy-restricted substring robustness over Π with respect to the composition of channels $\mathcal{E}_{sub}(f, 1/2 - \epsilon'_{sub}) \circ \mathcal{E}_{del}(1 - \epsilon'_{del})$.*

Alternatively, though none is currently known, if there were a construction of PRC that guaranteed robustness against *adversarial* substitutions and random deletions, one could bypass the need for heuristic assumptions such as that given by Definition 19 and instead leverage Claim 26 directly:

▶ **Theorem 22.** *If, for any constants $\epsilon_{sub} \in (0, 1/2]$ and $\epsilon_{del} \in (0, 1]$, there exists a pseudorandom code PRC (having block length $\ell(\epsilon_{sub}, \epsilon_{del}, n)$) with robustness against the composition of channels $\mathcal{E} \circ \mathcal{E}_{del}(1 - \epsilon_{del})$ for any $(1/2 - \epsilon_{sub})$-bounded binary substitution channel $\mathcal{E}$, then, for any generative model family $\mathcal{M}$ with token alphabet family $\mathcal{V} = \{\mathcal{V}(n)\}_{n\in\mathbb{N}}$, family of functions $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$, and constants $\epsilon'_{sub} \in (0, 1/2]$, $\epsilon'_{del} \in (0, 1]$, and $\delta \in (0, 1]$, there exists a secure watermarking scheme $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ for $\mathcal{M}$ that satisfies $(n\ell(\epsilon'_{sub}\delta/17, \epsilon'_{del}, n), \delta, 1)$-entropy-restricted substring robustness over Π with respect to the composition of channels $\mathcal{E}_{sub}(f, 1/2 - \epsilon'_{sub}) \circ \mathcal{E}_{del}(1 - \epsilon'_{del})$.*

Lastly, we show that we can leverage a PRC such as the new construction of [3] based on a "permuted codes conjecture" applied to folded Reed-Solomon codes, which features robustness to both a large (close to 1/2) fraction of (potentially arbitrary) substitutions and a very small but constant fraction of (potentially arbitrary) edits, to construct low-entropy watermarking robust to a constant fraction of arbitrary edits. However, this comes at a cost

to the length of the minimal-entropy substring required to prove robustness; in particular, for the case where we allow adversarial or arbitrary edits, we require *a constant fraction of the entire output* to have sufficient entropy, as otherwise it is fairly straightforward to see that a constant fraction of arbitrary edits could specifically target the minimal-entropy substring for corruption and thus break robustness. Thus, we conclude:

▶ **Theorem 23.** *If, for any constant $\epsilon_{sub} \in (0, 1/2]$, there exists a pseudorandom code* PRC *and a constant $\epsilon_{edit} \in (0, 1]$ such that* PRC *has robustness against the composition of channels $\mathcal{E} \circ \mathcal{E}'$ for any $(1/2 - \epsilon_{sub})$-bounded binary substitution channel $\mathcal{E}$ and any ϵ_{edit}-bounded binary edit channel $\mathcal{E}'$, then, for any generative model family $\mathcal{M}$ (having maximum output length $m(n)$) and constants $\delta, \alpha \in (0, 1]$, there exists a secure watermarking scheme* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *for $\mathcal{M}$ and constant $\epsilon'_{edit} \in (0, 1]$ such that* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *satisfies $(\alpha m(n), \delta, 1)$-entropy-restricted substring robustness with respect to any ϵ'_{edit}-bounded edit channel over the vocabulary of $\mathcal{M}$.*

Combining this with the respective PRC construction from [3] gives:

▶ **Corollary 24.** *If Conjecture 3.1 of [3] is true, then, for any generative model family $\mathcal{M}$ (having maximum output length $m(n)$) and constants $\delta, \alpha \in (0, 1]$, there exists a secure watermarking scheme* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *for $\mathcal{M}$ and a constant $\epsilon'_{edit} \in (0, 1]$ such that* $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ *satisfies $(\alpha m(n), \delta, 1)$-entropy-restricted substring robustness with respect to any ϵ'_{edit}-bounded edit channel over the vocabulary of $\mathcal{M}$.*

We provide greater specificity on the constant ϵ'_{edit} alongside the proofs of Theorems 20, 22, 23 and Corollaries 21 and 24 in the full version. The watermarking scheme $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ we use to prove the above results is presented in Figures 2 (Gen), 3 ($\mathsf{Watermark}$), and 4 (Detect).

References

1. Omar Alrabiah, Prabhanjan Ananth, Miranda Christ, Yevgeniy Dodis, and Sam Gunn. Ideal pseudorandom codes. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 1638–1647. ACM, 2025. doi:10.1145/3717823.3718309.
2. Diane Bartz and Krystal Hu. OpenAI, Google, others pledge to watermark AI content for safety, White House says, 2023. URL: https://www.reuters.com/technology/openai-google-others-pledge-watermark-ai-content-safety-white-house-2023-07-21.
3. Miranda Christ, Noah Golowich, Sam Gunn, Ankur Moitra, and Daniel Wichs. Improved Pseudorandom Codes from Permuted Puzzles, 2025. doi:10.48550/arXiv.2512.08918.
4. Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part VI*, volume 14925 of *Lecture Notes in Computer Science*, pages 325–347. Springer, August 2024. doi:10.1007/978-3-031-68391-6_10.
5. Miranda Christ, Sam Gunn, Tal Malkin, and Mariana Raykova. Provably robust watermarks for open-source language models. *CoRR*, abs/2410.18861, 2024. doi:10.48550/arXiv.2410.18861.
6. Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In Shipra Agrawal and Aaron Roth, editors, *The Thirty Seventh Annual Conference on Learning Theory, June 30 - July 3, 2023, Edmonton, Canada*, volume 247 of *Proceedings of Machine Learning Research*, pages 1125–1139. PMLR, 2024. URL: https://proceedings.mlr.press/v247/christ24a.html.

7 Nico Döttling, Anne Müller, and Mahesh Sreekumar Rajasree. Separating pseudorandom codes from local oracles. *IACR Cryptol. ePrint Arch.*, page 1020, 2025. URL: `https://eprint.iacr.org/2025/1020`.

8 Sanjam Garg, Sam Gunn, and Mingyuan Wang. Black-box crypto is useless for pseudorandom codes. *IACR Cryptol. ePrint Arch.*, page 1021, 2025. URL: `https://eprint.iacr.org/2025/1021`.

9 Surendra Ghentiyala and Venkatesan Guruswami. New constructions of pseudorandom codes. *CoRR*, abs/2409.07580, 2024. `doi:10.48550/arXiv.2409.07580`.

10 Noah Golowich and Ankur Moitra. Edit distance robust watermarks for language models. *CoRR*, abs/2406.02633, 2024. `doi:10.48550/arXiv.2406.02633`.

11 John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR, 2023. URL: `https://proceedings.mlr.press/v202/kirchenbauer23a.html`.

12 Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *CoRR*, abs/2307.15593, 2023. `doi:10.48550/arXiv.2307.15593`.

13 Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text. *CoRR*, abs/2306.17439, 2023. `doi:10.48550/arXiv.2306.17439`.

A Proof

In this appendix, we provide the complete proof of Theorem 17 (and Corollary 18); the remainder of the proofs (which proceed similarly) are deferred to the full version.

We begin by proving that $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ is a secure watermarking scheme (i.e., undetectable and sound), as these properties hold regardless of the robustness of PRC (though they do require pseudorandomness and soundness); afterwards, we prove robustness based on the respective robustness of PRC.

A.1 Undetectability and Soundness

▷ Claim 25. $(\mathsf{Gen}, \mathsf{Watermark}, \mathsf{Detect})$ satisfies soundness and undetectability for any family of generative models $\mathcal{M} = \{\mathsf{Generate}(n)\}_{n\in\mathbb{N}}$.

Proof. Both properties are fairly straightforward to demonstrate.

Soundness follows from the soundness of PRC; notice that Detect, on any given input, calls $\mathsf{PRC.Decode}$ at most a number of times polynomial in the security parameter n, since $m(n)$, $\ell(n)$, and the size of $\overrightarrow{h}$ are polynomial in n. Since sk is generated by $\mathsf{PRC.Gen}$, the probability of each such call returning a result besides $\perp$ is negligible in n, and thus the probability over sk and $\overrightarrow{h}$ of Detect returning true must likewise be negligible by a union bound.

Undetectability follows from the pseudorandomness of PRC. First consider an "ideal" world where the code c from PRC is replaced with true randomness, and thus each c_i is 0 or 1 with exactly probability 1/2. In this world we can show easily that $\mathsf{Watermark}$ perfectly preserves the distribution D of each token generated by $\mathsf{Generate}$. To see this, consider any token $t^* \in \mathcal{V}$. If t^* is drawn as t_0 and $t_1 \neq t^*$ (which happens with probability $D(t^*)(1 - D(t^*))$), and p is the probability that $h(x) \neq h(t^*)$ conditioned on $x \leftarrow D$ and $x \neq t^*$, the probability that t^* is selected as the next token t is:

$$p \Pr[c_i = h(t^*)] + (1-p)\frac{1}{2} = \frac{1}{2}p + \frac{1}{2}(1-p) = \frac{1}{2}$$

since, in Watermark, t_0 is selected when $c_i = h(t_0)$ (which, as we assume c_i to be uniformly random, happens with probability exactly $1/2$) if $h(t_0) \neq h(t_1)$ and with probability $1/2$ otherwise. The same is true symmetrically when t^* is drawn as t_1 and $t_0 \neq t^*$, and when t^* is drawn as *both* t_0 and t_1 (which happens with probability $D(t^*)^2$) then it is selected with probability 1. Thus, as all of these events are mutually exclusive, the probability that t^* is selected as the next token t is exactly:

$$2(D(t^*)(1 - D(t^*))\left(\frac{1}{2}\right) + D(t^*)^2 = D(t^*)$$

So if c is truly random, it is straightforward to see that, in each iteration of Watermark, the next token t is drawn with exactly the same probability as it would be by Generate, and thus the output of Watermark is identically distributed to the output of Generate on any specific prompt prompt. We shall denote the above "ideal" experiment by Ideal.

However, as c is instead the *pseudorandom* output of PRC, assume towards contradiction that there is some adversary $\mathcal{A}$ and polynomial $p(\cdot)$ where

$$|\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathcal{A}^{\mathsf{Watermark}_{\mathsf{sk}}(1^n,\cdot)}(1^n) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\mathsf{Generate}}(1^n,\cdot)}(1^n) = 1]| \geq p(n)$$

We leverage $\mathcal{A}$ to construct an adversary $\mathcal{A}'$ which contradicts the pseudorandomness of PRC. $\mathcal{A}'$ runs $\mathcal{A}$, but every time $\mathcal{A}$ invokes its oracle $\mathcal{A}'$ runs and returns the result of running Watermark on the given prompt with the single exception that every code c generated from PRC is replaced by the output of a call to its own oracle.

If the oracle given to $\mathcal{A}'$ calls PRC.Encode, then the experiment $\mathcal{A}'$ runs is identical to Watermark; if it instead generates uniform randomness, then the experiment is instead identical to Ideal, whose output is as we have already argued identically distributed to generating tokens with Generate. Thus,

$$|\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathcal{A}'^{\mathcal{O}_{\mathsf{Encode}}(\mathsf{sk})}(1^n) = 1] - \Pr[\mathcal{A}'^{\mathcal{O}_{\mathsf{Unif}}}(1^n) = 1]|$$

is by construction equal to

$$|\Pr[\mathsf{sk} \leftarrow \mathsf{Gen}(1^n) : \mathcal{A}^{\mathsf{Watermark}_{\mathsf{sk}}(1^n,\cdot)}(1^n) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\mathsf{Generate}}(1^n,\cdot)}(1^n) = 1]|$$

But since we assumed this latter quantity was at least polynomial $p(n)$, the former must be as well, contradicting the pseudorandomness of PRC and by contradiction completing the argument that (Gen, Watermark, Detect) satisfies undetectability. ◁

A.2 Robustness

The robustness guarantee of our watermarking scheme depends primarily on the robustness of PRC, since, while the scheme itself generates errors attempting to embed the PRC codeword c into (the hash of) the output, we show that these errors are comprised of a limited number of specifically substitution errors, which can be bounded as long as sufficiently many of the output tokens are generated with enough empirical entropy. Notably, however, these errors are not necessarily random, as, depending on the behavior of the model, tokens could be generated in such a way that the errors are correlated with significant probability over the choice of h.

Random Substitutions

If we restrict to the case of only random substitutions, we note that an alternative version of Watermark that generates a new function h for every token, as opposed to reusing the same h for an entire codeword of PRC, would be sufficient to prove robustness, since without

insertions or deletions the number and order of the tokens cannot change (and thus one could always track which function h was used to generate each token, even after random substitutions were made). However, this alternative approach fails with even a single insertion or deletion, and additionally only gains a small constant factor in the analysis for random substitutions, so we opt to instead present and analyze the existing version of Watermark to better set up the subsequent results for robustness against substitutions and deletions.

For the case of random substitutions only, we demonstrate the following claim:

▷ Claim 26. For constants $\epsilon, \epsilon', \delta > 0$ such that $\epsilon < \epsilon'\delta/16$, if PRC having block length $\ell(n)$ is robust against any $(1/2 - \epsilon)$-bounded binary substitution channel, then, for any family $f = \{f_n : \mathcal{V}(n) \to \Delta(\mathcal{V}(n))\}_{n\in\mathbb{N}}$, Watermark satisfies $(n\ell(n), \delta, 1)$-entropy-restricted substring robustness against the random substitution channel family $\mathcal{E}_{\text{sub}}(f, 1/2 - \epsilon')$.

(We note that [4] demonstrates constructions of the requisite PRC.)

Proof. First, similarly to the other two properties, it suffices to prove robustness by bounding the number of errors (i.e., tokens t_i where $h_j(t_i) \neq c_i$) in the "ideal" experiment described in Claim 25 where the codewords c generated by PRC are replaced by uniformly random bits; as shown in Claim 25, pseudorandomness of PRC guarantees that the view of this experiment is identically distributed to that of the real execution of Watermark except with negligible probability, and thus, if in the ideal experiment there exists with overwhelming probability a block of the random codeword c where sufficiently many tokens in the output are correctly embedded, it follows that there will also with overwhelming probability be such a block of the pseudorandom code PRC (which will thus be recoverable) in the real experiment.

We next need to bound the error rate of the output of Watermark introduced by the embedding process. Intuitively, in each step, the closer the distribution $\{h(d) : d \leftarrow D\}$ is to uniform over $\{0, 1\}$, the more likely the algorithm is to draw a pair (t_0, t_1) such that $h(t_0) \neq h(t_1)$ and thus output a token that hashes to the correct value of c_i from PRC without biasing the distribution of tokens. We make the following claim to show that, in a block where a sufficient number of tokens are drawn with enough empirical entropy, it is likely that enough of those tokens are drawn from sufficiently "close to uniform" distributions so that enough of *those* tokens are able to be drawn with hash value matching the PRC codeword to make the watermark easily detectable:

▷ Claim 27. Consider the execution of Watermark over some block of PRC (i.e., the iteration from $i = 1$ to $i = \ell(n)$ for some fixed j). There exists negligible $\nu(\cdot)$ such that, with at least probability $1/4$ over the choice of function h, the fraction of tokens t_i for which $h_j(t_i) \neq c_i$ is, except with probability $\nu(n)$ over the randomness of Watermark, at most $1/2 - \delta'/16$, where δ' is the fraction of tokens drawn with empirical entropy at least 1.

Proof. Consider an alternate experiment to generate the given block of Watermark where, to generate every token t_i, we first sample a *new* function $h \leftarrow \mathcal{H}$, and sample t_i using the respective function h but otherwise according to Watermark. Then, given some token t_i drawn with empirical entropy at least 1, this means that it was sampled from a distribution D such that $D(t_i) \leq 1/2$. In particular, this means that the probability that t_i was generated from a pair (t_0, t_1) such that $t_0 = t_1$ must likewise be at most $1/2$. If $t_0 \neq t_1$, then the probability over the randomly generated function h that $h(t_0) \neq h(t_1)$ is exactly $1/2$ (meanwhile, if $t_0 = t_1$, this probability is clearly 0); thus, the probability that $h(t_0) \neq h(t_1)$ during the generation of this specific token t_i must be at least $1/4$.

Notice that if $h(t_0) \neq h(t_1)$, then by construction of Watermark we will always select t_i such that $h(t_i) = c_i$ – that is, the embedding of the bit c_i of the PRC codeword will be successful. Meanwhile, if $h(t_0) = h(t_1)$, then the embedding will be successful if and only if $h(t_0) = h(t_1) = c_i$, which over randomly generated h happens with probability exactly $1/2$.

In the actual execution of Watermark, h remains the same throughout the entire generation of the block; however, because tokens are always generated according to their unbiased probabilities from $\mathsf{Generate}(\mathsf{prompt}, \overrightarrow{t})$ by construction, we note that the tokens generated are identically distributed to the above experiment (or to the output of simply generating non-watermarked tokens using $\mathsf{Generate}(\mathsf{prompt}, \overrightarrow{t})$). Because of this, it still holds that, if we sample two independently random tokens t_0 and t_1 as above, there is still probability $1/4$ over the choice of h that $h(t_0) \neq h(t_1)$ for every token t_i generated with sufficient empirical entropy. Unlike the above experiment, though, these probabilities are not independent as tokens may be repeated, so it is not necessarily the case that the fraction of tokens satisfying the entropy requirement with $h(t_0) \neq h(t_1)$ is close to $1/4$ except with negligible probability; however, we may still conclude by an averaging argument that, with at least probability $1/4$ over the choice of h, at least $1/4$ of the tokens with sufficient empirical entropy have $h(t_0) \neq h(t_1)$.[8]

Thus, in the case (which occurs with at least probability $1/4$) where at least $1/4$ of the tokens with sufficient empirical entropy (i.e., a $\delta'/4$ fraction of total tokens) have $h(t_0) \neq h(t_1)$, those tokens will *never* have $h_j(t_i) \neq c_i$, and the remainder will have it with at most probability $1/2$. This means that, *in expectation*, at most a $1/2(1 - \delta'/4) = 1/2 - \delta'/8$ fraction of tokens can have $h_j(t_i) \neq c_i$. However, for the tokens with $h(t_0) = h(t_1)$, the probabilities that $h_j(t_i) \neq c_i$ are independent in the "ideal" experiment where the codeword c is randomly generated, and thus it can be asserted that, except with negligible probability, the fraction of those tokens for which $h_j(t_i) \neq c_i$ is actually true will be close to its expectation.

Specifically, assume at least a $1/4$ fraction of tokens have $h(t_0) = h(t_1)$; otherwise we are already done (as then a $3/4$ fraction would be guaranteed to have $h_j(t_i) = c_i$). Then, by Hoeffding's inequality, the probability that the fraction of all tokens with both $h(t_0) = h(t_1)$ and $h_j(t_i) \neq c_i$ is at least $\delta'/16$ greater than its expectation – in other words, that the *number* of such tokens is at least $\delta'\ell(n)/16$ greater than its expectation – is at most

$$\exp\left(\frac{-2(\delta'\ell(n)/16)^2}{\ell(n)/4}\right) = \exp(-\delta'^2\ell(n)/32)$$

which is negligible in n because $\delta'^2/32$ is a (small) constant and $\ell(\cdot)$ is polynomial. And, in the case where a $\delta'/4$ fraction of total tokens have $h(t_0) \neq h(t_1)$ but the above event is *not* true (which, letting $\nu(n)$ be the negligible function above, must happen with at least probability $1/4 - \nu(n)$), the actual fraction of tokens with $h_j(t_i) \neq c_i$ can be at most

$$1/2 - \delta'/8 + \delta'/16 = 1/2 - \delta'/16$$

which completes the argument. ◁

Given this, assume that an output exists some substring of length at least $n\ell(n)$ containing at least a δ fraction of tokens with empirical entropy 1 (otherwise, if no such substring exists entropy-restricted substring robustness holds trivially). In this case, this substring must contain $k \geq n - 1$ complete blocks of the PRC, and those blocks must contain at least a $\delta - \frac{1}{n}$ fraction of the tokens with sufficient entropy, which we may take to be greater than $\delta/2$ for sufficiently large $n > 2/\delta$.

[8] This may seem unintuitive since, at first glance, one would expect the tokens selected, and thus the indices of the tokens with sufficient experimental entropy, to be affected by the choice of h; however, we again stress that, because Watermark perfectly preserves the distributions of tokens generated, the tokens are independent of h and c. In particular, one can imagine any set $S \subset [\ell(n)]$ of tokens and argue that, conditioned on the tokens with indices in S having sufficient empirical entropy, at least $1/4$ of the tokens with indices in S have $h(t_0) \neq h(t_1)$ with probability $1/4$ over randomly generated h.

Then, by an averaging argument, at least $\delta k/2$ blocks in that substring have at least a $\delta/4$ fraction of such tokens, which by Claim 27 means that those blocks, with probability $1/4$ over the choice of functions h, have (with overwhelming probability) at most a $1/2 - \delta/64$ error rate for embedding c. We shall refer to such blocks as "good" for later analysis. Notably, because an independently random function h is drawn for each block, these probabilities are independent.

Now, let us assume that the output is subjected to the random substitution channel $\mathcal{E}_{\text{sub}}(f, 1/2 - \epsilon')$ – that is, with independent probability $1/2 - \epsilon'$, each token is replaced with a different token determined according to an arbitrary function f.[9] Given this, we want to show that with overwhelming probability there is at least one block of the output that still has at most a $(1/2 - \epsilon)$ fraction of tokens t_i for which $h_j(t_i) \neq c_i$; if we can conclude this, then by the robustness of PRC the watermark must be detectable with overwhelming probability by hashing that block and running PRC.Decode during the execution of Detect. Formally, we prove this via the following:

▷ Claim 28. Given some "good" block of the output of Watermark, after $\mathcal{E}_{\text{sub}}(f, 1/2 - \epsilon')$ is applied the expected fraction of tokens t_i in that block for which $h_j(t_i) \neq c_i$ is no more than $1/2 - \epsilon'\delta/16$.

Proof. Assume as a worst case that, whenever a substitution is made to a token which, in the experiment given in Claim 27, was drawn with $h(t_0) \neq h(t_1)$ (i.e., selected from a pair whose elements hashed to different values), that substitution *always* changes the hash value of h applied to the token, creating a guaranteed error. Other tokens are errors with probability $1/2$, meaning that in expectation they will retain that probability after being substituted irrespective of the probability with which substitution changes the value of h.

In this case, let ψ be the fraction of tokens with $h(t_0) \neq h(t_1)$. Then in expectation the resulting fraction of errors is

$$\psi\left(\frac{1}{2} - \epsilon'\right) + (1 - \psi)\frac{1}{2} = \frac{1}{2} - \epsilon'\psi$$

For "good" blocks we assumed that ψ was at least $\delta/8$ (since Claim 27 selects blocks where more than $1/4$ of the tokens with empirical entropy 1 have $h(t_0) \neq h(t_1)$ and we apply it to blocks with at least a $\delta/4$ fraction of tokens having empirical entropy 1), thus the conclusion follows. ◁

Markov's inequality then gives that the probability that, in a "good" block, the fraction of tokens t_i for which $h_j(t_i) \neq c_i$ is greater than $(1/2 - \epsilon)$ (i.e., the watermark cannot be successfully detected) can be at most

$$\frac{1/2 - \epsilon'\delta/16}{1/2 - \epsilon}$$

which in particular means that, as long as $\epsilon < \epsilon'\delta/16$, the probability that a "good" block is recoverable is a (small) constant. Importantly, the probability of this occurrence is independent between blocks due to the resampling of h, which means that, since there are at least $\delta k/2$ blocks having enough tokens with empirical entropy 1, each of which has an

[9] Namely, consider mapping each token in the alphabet to a distribution of replacement tokens and, for each replacement, selecting a token from the corresponding distribution, as long as the same distribution is used for each instance of a token.

(independent) 1/4 probability to be "good" and the above constant (and also independent) probability to, if "good", have few enough tokens t_i for which $h_j(t_i) \neq c_i$ that robustness of PRC guarantees that the codeword c can be successfully detected, we can conclude that the probability of Detect failing to detect the watermark is negligible (inverse exponential) in $k \geq n - 1$ and thus in n, as desired. ◁

Together with Claim 25, Claim 26 implies Theorem 17 and thus Corollary 18.