Enhancing Watermarked Language Models to Identify Users

Aloni Cohen, Alexander Hoover, and Gabe Schoenbach

University of Chicago

Abstract

A *zero-bit* watermarked language model produces text that is indistinguishable from that of the underlying model, but which can be detected as machine-generated using a secret key. Unfortunately, merely detecting AI-generated spam, say, as watermarked may not prevent future abuses. If we could additionally *trace* the text to a spammer's API token or account, we could then cut off their access or pursue legal action.

We introduce *multi-user watermarks*, which allow tracing model-generated text to individual users or to groups of colluding users. We construct multi-user watermarking schemes from undetectable zero-bit watermarking schemes. Importantly, our schemes provide both zero-bit and multi-user assurances at the same time: detecting shorter snippets just as well as the original scheme, and tracing longer excerpts to individuals. Along the way, we give a generic construction of a watermarking scheme that embeds long messages into generated text.

Ours are the *first black-box reductions* between watermarking schemes for language models. A major challenge for black-box reductions is the lack of a unified abstraction for *robustness* — that marked text is detectable even after edits. Existing works give incomparable robustness guarantees, based on bespoke requirements on the language model's outputs and the users' edits. We introduce a new abstraction to overcome this challenge, called *AEB-robustness*. AEB-robustness provides that the watermark is detectable whenever the edited text "approximates enough blocks" of model-generated output. Specifying the robustness condition amounts to defining approximates, enough, and blocks. Using our new abstraction, we relate the robustness properties of our message-embedding and multi-user schemes to that of the underlying zero-bit scheme, in a black-box way. Whereas prior works only guarantee robustness for a single text generated in response to a single prompt, our schemes are *robust against adaptive prompting*, a stronger and more natural adversarial model.

1 Introduction

Generative AI models can now produce text and images not easily distinguishable from human-authored content. There are many concerns about the undisclosed use of generative AI, whether for nefarious or banal purposes. Watermarking is one approach for detecting and tracing the provenance of generative AI outputs, and it raises challenging technical and policy questions [Sri24, BG24].

Watermarking for generative AI is on the cusp of deployment. In July 2023, the White House secured commitments from seven industry leaders to manage and mitigate risks posed by AI: Amazon, Anthropic, Google, Inflection, Meta, Microsoft, and OpenAI [Hou23a, Hou23b]. Among the commitments is the development of watermarking and other provenance techniques for the next generation of audio and image models. A few months later, President Biden issued an executive order directing federal agencies to "establish standards and best practices for detecting AI-generated content," including guidance on the use of watermarking [Hou23c]. Google already watermarks all content produced by the Lyria music generation model [SG23], and OpenAI has a working prototype for watermarking text produced by ChatGPT [Aar22].

A burgeoning line of research studies watermarking for language models specifically, where strong statistical and cryptographic guarantees (and negative results [ZEF+23]) can be proved or heuristically justified

[Aar22, KGW+23a, CGZ23, FGJ+23, KTHL23, CG24]. These schemes envision a setting where a user queries a language model through some interface. For example, querying GPT-4 using ChatGPT or the API. The model deployer, e.g., OpenAI, uses a secret watermarking key in conjunction with the underlying language model to produce watermarked text. The mark can be extracted from the text using the secret key.

To be useful, watermarking schemes must have four properties. Existing schemes achieve these guarantees, though the formalisms and assumptions vary greatly. First is *soundness*: marks must not be falsely detected in text not generated by the watermarked model (low Type I error). Second is *completeness*: verbatim outputs of the watermarked model are detectable (low Type II error). Third is *robustness* to edits: marks are detectable even after marked text is edited, say by deleting or rearranging phrases. Finally, watermarking should not noticeably degrade the quality of the model's outputs. The strongest version of this is *undetectability*, requiring that the outputs of the marked and unmarked models are indistinguishable.

All existing works share two major drawbacks. First, none of them considers robustness when users *adaptively prompt* the model. Second, merely detecting watermarked text is not always enough. As we explain below, sometimes we need to *trace text to a specific user* of the language model.

Adaptive prompting We introduce and build watermarking schemes that are **robust to adaptive prompting**. Existing works define security for a single text T produced by the model in response to any single prompt Q. But no user issues just one prompt! Even benign users interact with the models, adaptively refining the prompts and generations as they go. Current definitions don't even imply that the resulting text is marked, let alone robust to edits. We give the first definitions and proofs of robustness when users interactively query a language model and derive text from the whole interaction.

Multi-user watermarks We introduce and build a **multi-user watermarking scheme** for language models. In such a scheme, model-generated outputs can be traced to specific users, even when users *collude*.

Detecting watermarks isn't always enough to mitigate harms. Consider a ChatGPT-powered bot carrying out romance scams to trick victims into transferring large sums of money. The gullible victims won't check for watermarks themselves. And any single messaging platform detecting the watermarks and banning the bot won't help much — the scammer can always switch to a different platform. After the scam is revealed, the watermarked text — the bot's messages — may be available to law enforcement, the messaging platform, or the language model provider. But there doesn't seem to be any way to arrest the scammer or protect future victims.

We'd like to trace the watermarked text to a specific user. The bot is querying ChatGPT using the scammer's user account or API token. If the marked text revealed the user, one could directly cut off the scammer's access to the model or seek legal recourse.

We also want security when multiple users collude. Consider three users each using ChatGPT to collaboratively write a shoddy legal brief or CCS submission. They might independently borrow from their respective model outputs, while making edits and adding text of their own. Robust multi-user watermarking guarantees that at least one of the three is identified, but no innocent users are.

As a first attempt at building multi-user watermarking for n users, one might use a different secret key for each user. But merely detecting the watermark would require checking each of the n secret keys. This would be too slow for widespread watermark detection: ChatGPT reportedly has more than 180 million monthly active users. In contrast, our multi-user watermarking scheme takes $O(\log n)$ time to detect watermarked text (omitting other parameters). Without collusions, tracing also takes $O(\log n)$ time, which is optimal as $\log n$ bits are needed to identify a user. With collusions, tracing takes $O(n \log n)$ time (performing a $O(\log n)$-time check for each user).

Adding the ability to trace individual users should not compromise the detection of watermarked text in contexts where tracing may not be needed (e.g., spam filtering). Our multi-user construction leaves untouched the guarantees of the underlying watermarking scheme. *You get the best of both robustness guarantees at the same time!* A short marked text is detectable as before, and a longer marked text is traceable to individuals.

Black-box reductions for watermarking schemes We give a black-box construction of multi-user watermarking from existing watermarking schemes. Our high level approach to building multi-user watermarking is relatively simple. Suppose you have an *L-bit watermarking scheme*: one that embeds an L-bit message into generated text. Then, ignoring collusions, the obvious idea is to embed an ID unique to each user. (Our scheme can be made robust to collusions using a cryptographic primitive called a fingerprinting code, though this doesn't work generically.) So it suffices to build an L-bit scheme out of a so-called *zero-bit watermarking scheme*, where text is simply viewed as "marked" or "unmarked."[1] We use a natural idea. We sample $2L$ secret keys $k_{i,b}$, one for each index i and bit b. To embed a message m, we use the zero-bit scheme with the keys $k_{i,m[i]}$ for each index i. The result is text watermarked under L different keys which together reveal the message.

The challenge is saying anything interesting about our construction while treating the underlying watermarking scheme as a black box. It's not even clear how to state the appropriate robustness guarantee, let alone prove it. The issue is that every watermarking construction has a bespoke formulation of completeness[2] (which verbatim model outputs are marked) and a bespoke formulation of robustness[3] (which edits preserve the mark), see Appendix A. There are two ways forward: choose a specific scheme and tailor the results, or invent a unifying language for watermarking robustness and completeness.

We present a new framework for describing robustness and completeness guarantees of watermarking schemes, called **AEB-robustness**. AEB-robustness provides that text is watermarked if it **A***pproximates* **E***nough* **B***locks* of model-generated text. Specifying the robustness condition amounts to defining "approximates," "enough," and "blocks." All else equal, a scheme is more robust if looser approximations are allowed; fewer blocks are required; or blocks require less entropy.

Our black-box reductions only affect how many blocks are enough, not what constitutes a block nor an approximation thereof. Our results hold for any (efficiently-checkable) definition of a block and any definition of string approximation.

With the language of AEB-robustness, our theorems are easy to state informally. Let λ be a cryptographic security parameter. Suppose $\mathcal{W}$ is a (zero-bit) watermarking scheme that is undetectable, sound, and robust whenever one block is approximated (R_1-robust). Our L-bit scheme is undetectable, sound, and robust whenever $k = O(L\lambda)$ blocks are approximated (R_k-robust). Our multi-user scheme is undetectable, sound, and robust for $n > 1$ users and $c > 1$ collusions whenever $k = O(c^2 \log^2(c) \log(n)\lambda)$ blocks are approximated. We adopt the cryptographic approach of analyzing security against all efficient adversaries. A byproduct is that our theorems only apply when the underlying watermarking scheme is cryptographically-secure.

1.1 Our contributions

We continue the study of watermarking schemes for language models with provable guarantees. Except where specified, our constructions require cryptographically-strong undetectability and soundness, and that the underlying scheme is AEB-robust.

1. We give the first black-box reductions among watermarking schemes, using a new framework for describing the robustness of watermarking schemes, called *AEB-robustness*. We give the first definitions for robustness against *adaptive prompting*, and prove that existing undetectable watermarking schemes are adaptively robust.

2. We construct *L-bit watermarks* with provable robustness from zero-bit watermarking schemes. We also construct *lossy* L-bit watermarks, which yield an improved multi-user scheme.

3. We define *multi-user* watermarking schemes, which allow tracing model-generated text to users. Our construction achieves provable robustness and *collusion resistance* from our L-bit watermarking scheme

[1] With one exception [CG24], all the existing schemes we study are zero-bit schemes.

[2] All seemingly-incomparable lower bounds on entropy: watermark potential [KTHL23], min-entropy per block [FGJ+23], spike entropy [KGW+23a], empirical entropy [CGZ23, CG24], and on-average high entropy / homophily [ZALW23].

[3] All requirements on susbtrings: equality [CGZ23, FGJ+23], edit distance [KTHL23, ZALW23], produced by a binary-symmetric channel [CGZ23].

and a *fingerprinting code*. It leaves unaffected the stronger robustness of the underlying zero-bit watermarking scheme, essentially allowing both schemes to be used at once.

1.2 Related work

A recent flurry of work on the theory of watermarking language models was kicked off by [Aar22] and [KGW+23a]. We directly build on this line of work, especially those with strong provable guarantees for undetectability [CGZ23], robustness [KTHL23, CG24] or public detection [FGJ+23]. See Appendix A for an in-depth discussion of some of these schemes. We adapt and extend the cryptographic-style definitions of [CGZ23, FGJ+23, CG24], which enables security proofs against arbitrary efficient adversaries. Other recent work proves strong impossibility results against motivated and resourced adversaries [ZEF+23, PHZS24]. But watermarking can still be useful — a little additional overhead can make a bad actor's job significantly harder. Concurrently, many more applied works have advanced the practice of watermarking language models [QYH+24, NJA24, LRW+24, JGHG24, XYL24, CLW+24] and other generative AI models [HD17, ZKJFF18, WKGG24].

Earlier work develops steganography for language models [KJGR21]. Steganography [Hop04, Cac00] and watermarking for language models are closely related, both embedding a message into a model's outputs. While steganography requires that the existence of the message be hidden, watermarking requires that the message persist even when generated text is modified.

We make extensive use of fingerprinting codes [BS98]. We make black-box use of existing codes, particularly those that are robust to adversarial erasures. Asymptotically-optimal codes are given by [Tar08, BKM10], while [NFH+07] focus on concrete efficiency.

1.3 Outline

The rest of the paper is structured as follows. Section 2 defines *prefix-specifiable language models*, the type of models we watermark, and robust fingerprinting codes, which are used in our constructions. Section 3 gives definitions for watermarking language models. We define zero-bit, L-bit, and multi-user watermarking schemes, along with their important properties: undetectability, soundness, completeness, and robustness. Section 4 defines *block-by-block* robust watermarks, our framework for constructing and proving black-box watermarking constructions. Appendix A shows that some existing watermarking schemes are block-by-block. Section 5 constructs an L-bit watermarking scheme from zero-bit watermarking schemes. We show that the L-bit scheme inherits undetectability, soundness, and robustness from the underlying zero-bit scheme. Section 6 constructs a multi-user watermarking scheme from our L-bit scheme and robust fingerprinting codes. As before, the multi-user scheme preserves the undetectability, soundness, and robustness guarantees of the underlying watermarking scheme. We additionally preserve the efficiency and utility of the original zero-bit scheme throughout our black-box usage. The above constructions all require the underlying watermarking scheme to be cryptographically undetectable. In Section 7, we show that it is possible to build robust multi-user watermarking schemes out of watermarking schemes that are not undetectable, albeit with less impressive robustness parameters.

2 Preliminaries

2.1 Notation

For $n \in \mathbb{N}$, we denote by $[n]$ the set $\{1, 2, \ldots, n\}$. A polynomial, sometimes denoted $\mathrm{poly}(\cdot)$, is some function for which there is a constant c with $\mathrm{poly}(n) = O(n^c)$. A function is negligible if it is asymptotically smaller than any inverse polynomial, and we use $\mathsf{negl}(\cdot)$ to denote an arbitrary negligible function. Throughout the paper, we use λ to denote the security parameter of the scheme, which specifies the security level of the protocol. We say an algorithm is *efficient* or poly-bounded if its runtime is bounded above by some polynomial.

In our pseudocode, we use "$x \leftarrow y$" to assign x the value of y. Likewise we use "$x \leftarrow\!\!\$\ y$" to denote sampling (uniformly) from y and assigning the value to x (e.g. $x \leftarrow\!\!\$\ [n]$ denotes selecting a random integer between 1 and n inclusive). The item y may also be a randomized function, which could just be viewed as running the function with fresh randomness. We also sometimes use true and false in place of 1 and 0, respectively, to make the semantics of an algorithm more clear.

For a finite set Σ called an *alphabet*, the set Σ^* denotes the set of all finite-length sequences of elements of Σ, called *strings*. Language models are typically defined with respect to an alphabet $\mathcal{T}$, whose elements are called *tokens*. The length of a string T is denoted $|T|$. For a string T, we define $T[i]$ to be the token at index i in the string, for $1 \leq i \leq |T|$. The concatenation of string S to string T is denoted $T\|S$.[4] We use ϵ to denote the empty string (in contrast to ε which usually denotes a real number). The empty string satisfies $|\epsilon| = 0$ and $\epsilon\|T = T = T\|\epsilon$ for all T. We use Δ to denote the normalized Hamming distance between strings of the same length. Specifically, for $S, T \in \Sigma^L$, let $\Delta(S,T) := \frac{1}{L} \cdot \sum_{i=1}^{L} \mathbb{1}(S[i] \neq T[i])$.

For $0 \leq \delta \leq 1$, $L \in \mathbb{N}$ and a string $y \in \{0,1\}^L$, we define the *δ-erasure ball* $B_\delta(y) \subseteq \{0,1,\perp\}^L$ to be the set of all strings $z \in \{0,1,\perp\}^L$ where $z_i = \perp$ for at most $\lfloor \delta L \rfloor$ indices i, and otherwise $z_i = y_i$. For $Y \subseteq \{0,1\}^L$, we define $B_\delta(Y) = \cup_{y \in Y} B_\delta(y)$.

2.2 Language models

A *language model* Model is a randomized algorithm that takes as input a *prompt* Q and outputs a *generation* T in response.[5] Language models are defined with respect to a set of *tokens* $\mathcal{T}$, where prompts and generations are strings $Q, T \in \mathcal{T}^*$. For example, OpenAI's GPT-4 uses a set of 100,000 tokens, with the tokens 'water,' 'mark', and 'ing' composing the word 'watermarking'. The token set contains a *termination token*, which we denote $\perp \in \mathcal{T}$. Every generation T ends in $\perp$, and $\perp$ appears nowhere else in T.

We restrict our attention to *prefix-specifiable* models. Model is prefix-specifiable if for all prompts Q and all prefixes T, one can efficiently compute a new prompt denoted $Q\|T$ such that the distributions (i) $\mathsf{Model}(Q||T)$ and (ii) $\mathsf{Model}(Q)$ conditioned on $\mathsf{Model}(Q)_{\leq |T|} = T$ are identical. This is a very natural theoretical assumption to make, as most popular transformer models have this property since they depend only on the prior tokens viewed. This assumption was also made by prior work [KGW+23a, CGZ23, CG24]. For a more complete discussion about prefix-specifiable models and their limitations, we direct the reader to Section 6 of [CGZ23].

2.3 Fingerprinting codes

Fingerprinting codes allow for efficient tracing of pirated digital content. In a canonical example, a film distributor is sending a movie to several reviewers in advance of a screening. To combat any pre-release leaks, the distributor embeds a distinct codeword in every copy, where each letter in the codeword is embedded in a particular scene of the movie. If a reviewer leaks the movie to the public, the distributor can extract the codeword from the leaked copy and trace it back to the guilty party. The tracing task becomes more difficult, however, if the reviewers can collude: by picking and choosing scenes from each of their copies, the reviewers may hope to leak a version of the movie that cannot be traced back to any colluding party. Fingerprinting codes guarantee that the distributor can still identify a guilty party so long there are at most c colluders and they are restricted to picking scenes from their own c copies. In general, one cannot hope to identify more than one guilty party, as the colluders can always choose to leak a copy without any edits.

Definition 2.1 (Fingerprinting codes – syntax [BS98])**.** *A* fingerprinting code *is a pair of efficient algorithms* $\mathsf{FP} = (\mathsf{FP.Gen}, \mathsf{FP.Trace})$ *where:*

- $\mathsf{FP.Gen}(1^\lambda, n, c, \delta) \to (X, \mathsf{tk})$ *is a randomized algorithm that takes as input a security parameter λ, a number of users n, a maximum number of colluders c, and an erasure bound $0 \leq \delta < 1$, and outputs a binary code $X \in \{0,1\}^{n \times L}$ of size n and length L, and a tracing key* tk.

[4] As explained in Sec. 2.2, we also use $\|$ to specify the prefix of a language model's output.

[5] Note that that our notation differs from prior work. Here, Model outputs a string in $\mathcal{T}^*$ (like $\overline{\mathsf{Model}}$ in [CGZ23, CG24]), rather than a distribution over the next token.

- $\mathsf{FP.Trace}(y, \mathsf{tk}) \to S$ *is a deterministic algorithm that takes as input any string* $y \in \{0,1,\perp\}^L$ *and a tracing key* tk*, and outputs a subset* $S \subseteq [n]$ *of accused users.*

For $u \in [n]$, we denote the codeword assigned to user u by X_u, the u^{th} row of X. For $C \subseteq [n]$, we denote the codewords assigned to the set of colluders C by X_C, the submatrix $(X_u)_{u \in C}$. Throughout our paper, we assume that the length L of the fingerprinting code FP is a deterministic function of λ, n, c and δ. Most fingerprinting codes, including the ones we use [BKM10, NFH$^+$07], have this property.

Fingerprinting codes provide a guarantee when a subset of users C produce a codeword y by picking and choosing the individual bits of the codewords in X_C. *Robust* fingerprinting codes also allow a δ-fraction of the bits to be adversarially erased. The *feasible set* contains all strings y that the colluding parties are able to create.

Definition 2.2 (Feasible sets)**.** *For* $X \in \{0,1\}^{n \times L}$*,* $C \subseteq [n]$*, the* feasible set *is*

$$F(X_C) := \{y \in \{0,1\}^L \ : \ \forall i \in [L],\ \exists x \in X_C,\ x[i] = y[i]\}.$$

In particular, if every $x \in X_C$ *has the same value* b *at index* i*, then* $y_i = b$*. For* $0 \le \delta \le 1$*, the* δ-feasible ball *is* $F_\delta(X_C) := B_\delta(F(X_C))$*.*

The goal of the colluders is to output some feasible word $y \in F_\delta(X_C)$ such that $\mathsf{FP.Trace}(y, \mathsf{tk}) = \emptyset$ (no user is accused) or that $\mathsf{FP.Trace}(y, \mathsf{tk}) \not\subseteq C$ (an innocent user is accused). The fingerprinting code is secure if this happens with negligible probability.

Definition 2.3 (Fingerprinting codes – robust security [BKM10])**.** *A fingerprinting code* FP *is* robust *if for all* $0 \le \delta \le 1$*,* $c \ge 1$*,* $n \ge c$*,* $C \subseteq [n]$ *of size* $|C| \le c$*, and all efficient adversaries* $\mathcal{A}$*, the following event occurs with negligible probability:*

- $y \in F_\delta(X_C)$*, AND* `// y is feasible`
- $\mathsf{FP.Trace}(z, \mathsf{tk}) = \emptyset$ *OR* $\mathsf{FP.Trace}(z, \mathsf{tk}) \not\subseteq C$ `// no or false accusation`

in the probability experiment defined by $(X, \mathsf{tk}) \leftarrow \mathsf{FP.Gen}(1^\lambda, n, c, \delta)$ *and* $y \leftarrow \mathcal{A}(X_C)$*.*

Tardos's fingerprinting code [Tar08] is asymptotically optimal, but not robust to adversarial erasures. The fingerprinting code of Boneh, Kiayias, and Montgomery [BKM10] is based on the Tardos code but is robust. For $n > 1$ users, $c > 1$ colluders, and a δ erasure bound, it has length $L = O\Big(\lambda(c \log c)^2 \log n/(1-\delta)\Big)$, but with very large constants.

2.4 Balls and bins

Our lossy watermarking schemes allow some of the mark to get erased. Our analysis will use the following lemma about throwing k balls into L bins uniformly at random. The lemma defines k^*, the number of balls needed to guarantee that at most δL bins are empty except with probability $e^{-\lambda}$. The proof uses standard techniques and is deferred to Appendix C.

Lemma 2.4. *For* $\lambda, L \in \mathbb{N}$ *and* $0 \le \delta < 1$*, define*

$$k^*(L, \delta) = \min \left\{ L \cdot (\ln L + \lambda); \quad L \cdot \ln \left(\frac{1}{\delta - \sqrt{\frac{\lambda + \ln 2}{2L}}} \right) \right\} \tag{1}$$

Then, after throwing $k \ge k^*(L, \delta)$ *balls into* L *bins, fewer than* δL *bins are empty except with probability at most* $e^{-\lambda}$*.*

3 Watermarks for Language Models

This section defines the syntax and properties of watermarking schemes. We introduce *multi-user watermarking schemes* that can trace watermarked text to an individual user, even in the presence of *collusions*.

We require robustness/completeness to hold in the face of *adaptive prompting*. All prior works only require completeness and robustness for a single text generated using a fixed prompt. But this is not how generative models are used in reality. Users interact with the models, adaptively refining the prompts and generations until they are satisfied with the result. In this setting, existing definitions don't make any guarantees at all! We instead require that completeness and robustness hold for adversaries that adaptively query the model, possibly selecting text from many responses. In Section 4.1, we will show that in some cases non-adaptive robustness implies adaptive robustness.

We consider three types of watermarking schemes: zero-bit, L-bit, and multi-user watermarking. An *L-bit* watermarking scheme embeds into generated text an L-bit message which can later be extracted using a secret key. In a *zero-bit* watermarking scheme, text is viewed as either marked or unmarked, but there is no message to be extracted from marked text. The name stems from viewing zero-bit watermarking as a special case of L-bit watermarking, with message space containing only one message. A *multi-user* watermarking scheme allows for tracing model-generated text back to the user (or group of users) who prompted the generation.

We call the robustness properties of the three types of schemes *robust detection*, *robust extraction*, and *robust tracing*, respectively. We use *robust* when the meaning is clear from context. Definitions for zero- and L-bit watermarking are adapted from existing works, though we relax correctness to require only $(1-\delta)L$ bits of the message to be recovered. Our notation most closely follows [CGZ23, CG24].

3.1 Zero- and L-bit watermarking syntax

We give the syntax of zero-bit watermarking and L-bit watermarking schemes. The difference is whether the watermark is binary — marked or unmarked — or encodes a message. Throughout the paper we focus on the secret key setting, where Detect and Wat share a key sk generated by KeyGen.

For zero-bit watermarking, the algorithm Wat is the watermarked version of the language model, with the same inputs and outputs. The Detect algorithm indicates whether text is considered marked or unmarked.

Definition 3.1 (Zero-bit watermarking – Syntax)**.** *A* zero-bit watermarking scheme *for a language model* Model *over $\mathcal{T}$ is a tuple of efficient algorithms $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ where:*

- $\mathsf{KeyGen}(1^\lambda) \to \mathsf{sk}$ *is a randomized algorithm that takes a security parameter λ as input and outputs a secret key* sk*.*
- $\mathsf{Wat}_{\mathsf{sk}}(Q) \to T$ *is a keyed randomized algorithm that takes a prompt Q as input and outputs a string $T \in \mathcal{T}^*$.*
- $\mathsf{Detect}_{\mathsf{sk}}(T) \to b$ *is a keyed deterministic algorithm that takes a string $T \in \mathcal{T}^*$ as input and outputs a bit $b \in \{0,1\}$.*

An L-bit watermarking scheme allows a message $m \in \{0,1\}^L$ to be embedded in generated text and later extracted. The syntax is the natural generalization of the above, though we rename Detect to $\mathsf{Extract}$, reflecting its new semantics. Observe that a 1-bit scheme can be used to construct a zero-bit scheme by taking $\mathsf{Wat}_{\mathsf{sk}}(Q) := \mathsf{Wat}_{\mathsf{sk}}(1, Q)$ and $\mathsf{Detect}_{\mathsf{sk}}(T) := \mathbb{1}\big(\mathsf{Extract}_{\mathsf{sk}}(T) = 1\big)$. Likewise for $L > 1$.

Definition 3.2 (Watermarking syntax – L-bit)**.** *An L-bit* watermarking scheme *for a language model* Model *over $\mathcal{T}$ is a tuple of efficient algorithms $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ where:*

- $\mathsf{KeyGen}(1^\lambda) \to \mathsf{sk}$ *is a randomized algorithm that takes a security parameter λ as input and outputs a secret key* sk*.*

- $\mathsf{Wat}_{\mathsf{sk}}(m, Q) \to T$ *is a keyed randomized algorithm that takes as input a message* $m \in \{0,1\}^L$ *and a prompt* $Q \in \mathcal{T}^*$ *and outputs a string* $T \in \mathcal{T}^*$.
- $\mathsf{Extract}_{\mathsf{sk}}(T) \to \hat{m}$ *is a keyed deterministic algorithm that takes a string* $T \in \mathcal{T}^*$ *as input and outputs a message* $\hat{m} \in \{0,1,\perp\}^L$.

3.2 Properties of watermarking schemes

Watermarking schemes must have four properties. First is *soundness*: marks must not be falsely detected (resp. extracted, traced) in text not generated by the watermarked model (low Type I error). Second, watermarking should not noticeably degrade the quality of the model's outputs. The strongest version of this is *undetectability*: the marked and unmarked models are indistinguishable. Third is *completeness*: the marks are detectable in verbatim outputs from the watermarked model (low Type II error). Fourth is *robustness* to edits: marks are detectable even after marked text is edited, say by deleting or rearranging phrases, or by pasting an excerpt into an unmarked document. All the existing schemes offer some degree of all four guarantees, though the formalisms and assumptions vary greatly.

We now define these properties for L-bit watermarking. We defer to Appendix B definitions of undetectability, soundness, completeness, and robust detection for zero-bit watermarking, as they are special cases of the L-bit definitions. We require negligible probabilities of failure against arbitrary efficient adversaries, in part because such strong guarantees lend themselves to black-box reductions. Some prior works target weaker guarantees, in particular by relaxing undetectability. In Section 7, we give some limited results for schemes that are not undetectable.

Soundness requires that $\mathsf{Extract}$ returns $\perp^L$ except with negligible probability for all strings T containing at most poly-many tokens.

Definition 3.3 (Soundness – L-bit)**.** *An L-bit watermarking scheme $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ is* sound *if for all polynomials* poly *and all strings $T \in \mathcal{T}^*$ of length $|T| \leq \mathrm{poly}(\lambda)$,*

$$\Pr_{\mathsf{sk} \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathsf{Extract}_{\mathsf{sk}}(T) \neq \perp^L] < \mathsf{negl}(\lambda).$$

Undetectability requires that outputs of the watermarked model Wat must be computationally indistinguishable from outputs of the underlying model Model. In particular, this implies that any computationally-checkable utility guarantees of Model also hold for Wat.

Definition 3.4 (Undetectability – L-bit)**.** *Define the oracle $\mathsf{Model}'(m, Q) := \mathsf{Model}(Q)$. An L-bit watermarking scheme $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ for Model is* undetectable *if for all efficient adversaries $\mathcal{A}$,*

$$\left| \Pr[\mathcal{A}^{\mathsf{Model}'(\cdot,\cdot)}(1^\lambda) = 1] - \Pr_{\mathsf{sk} \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot,\cdot)}(1^\lambda) = 1] \right| < \mathsf{negl}(\lambda).$$

Completeness and robustness Formally defining completeness and robustness requires some care. We focus on robustness, as completeness is a special case.

Intuitively, robustness should guarantee something like the following: if $T \leftarrow\!\$\ \mathsf{Wat}_{\mathsf{sk}}(m, Q)$ and $\hat{T} \approx T$, then $\mathsf{Extract}_{\mathsf{sk}}(\hat{T}) = m$. But this requirement is too strong. Soundness requires that any fixed string, say the text of The Gettysburg Address, is unmarked with high probability. But then so must the text T generated in response to the query $Q =$ `What is the text of The Gettysburg Address?`, assuming the model answers correctly.

To deal with this issue, existing works impose a requirement on the *entropy* of T, for various notions of entropy. In light of the above, robustness should guarantee something like the following. If $T \leftarrow\!\$\ \mathsf{Wat}_{\mathsf{sk}}(m, Q)$, then one of the following holds with high probability: (i) T lacks sufficient entropy; (ii) $\hat{T} \not\approx T$; or (iii) $\mathsf{Extract}_{\mathsf{sk}}(\hat{T}) = m$. Instantiating the definition requires specifying what exactly conditions (i) and (ii) mean.

Our definition of robustness is agnostic to this choice (though our constructions require additional structure, see Sec. 4). Instead, we define robustness relative to a generic *robustness condition* R which evaluates

to 1 when both (i) T has sufficient entropy, and (ii) $\hat{T} \approx T$. Note that R only defines a *sufficient* condition for extracting the mark. It may sometimes hold that $\mathsf{Extract}_{\mathsf{sk}}(\hat{T}) = m$ even when $R = 0$.

Our definition of robust extraction is also parameterized by a number $0 \leq \delta \leq 1$. A scheme is *lossy* if $\delta > 0$, and *lossless* if $\delta = 0$. We omit δ for lossless schemes, writing *R-robust/complete*. Recall that $B_\delta(m)$ is the set of all strings $\hat{m} \in \{0, 1, \perp\}^L$ that agree with m except for at most $\lfloor \delta L \rfloor$ indices.

Definition 3.5 (Robustness condition)**.** *A* robustness condition *is a (deterministic, efficient) function* $R : (\lambda, (Q_i)_i, (T_i)_i, \hat{T}) \mapsto b$*, where* $\lambda \in \mathbb{N}$*;* $Q_i, T_i, \hat{T} \in \mathcal{T}^*$ *for all* i*; and* $b \in \{0, 1\}$*.*

Definition 3.6 ((δ, R)-Robust extraction – L-bit, adaptive)**.** *An L-bit watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ *is* adaptively (δ, R)-robustly extractable *with respect to robustness condition R if for all messages* $m \in \{0, 1\}^L$*, and all efficient adversaries $\mathcal{A}$, the following event* `FAIL` *occurs with negligible probability:*

- $R\left(\lambda, (Q_i)_i, (T_i)_i, \hat{T}\right) = 1$*, AND* `// robustness condition holds`
- $\hat{m} \notin B_\delta(m)$ `// the mark is corrupted`

in the probability experiment defined by

- $\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)$
- $\hat{T} \leftarrow \mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(m,\cdot)}(1^\lambda)$*, denoting by* $(Q_i)_i$ *and* $(T_i)_i$ *the sequence of inputs and outputs of the oracle*
- $\hat{m} \leftarrow \mathsf{Extract}_{\mathsf{sk}}(\hat{T})$*.*

Completenss is almost identical, with the additional clause "AND $\hat{T} \in (T_i)_i$" added to `FAIL` (see Definition B.3).

3.3 Multi-user watermarks

We now define a multi-user watermarking scheme. We consider a watermarking scheme which is deployed for a set of users $\mathcal{U}$ and generalize the notation for zero-bit watermarking. Our definition has three functions that nearly match the syntax and semantics of zero-bit watermarking. The only difference among the first three functions is that Wat takes as input both a user and a prompt. The new functionality of a multi-user watermarking scheme is Trace, which given some text $\hat{T}$ can output the user that produced $\hat{T}$.

Our syntax tracks the following intended usage. The watermarker initially sets up their system by generating a secret key sk with KeyGen. For each user u, they provide oracle access to $\mathsf{Wat}_{\mathsf{sk}}(u, \cdot)$, fixing the first input u. This models, say, a signed-in user interacting with ChatGPT.

To detect only the presence of a watermark in a candidate text $\hat{T}$, the watermarker can run $\mathsf{Detect}_{\mathsf{sk}}(\hat{T})$ exactly as in a zero-bit watermarking scheme. If they want to determine which user(s) the text belongs to, they can run $\mathsf{Trace}_{\mathsf{sk}}(\hat{T})$, which outputs the (possibly empty) set of users whose watermarked outputs generated $\hat{T}$.

Definition 3.7 (Multi-user watermarking)**.** *A* multi-user watermarking scheme *for a model* Model *over a token alphabet $\mathcal{T}$ and a set of users $\mathcal{U}$ is a tuple of efficient algorithms* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ *where:*

- $\mathsf{KeyGen}(1^\lambda) \to \mathsf{sk}$ *is a randomized algorithm that takes a security parameter λ as input and outputs a secret key* sk*.*
- $\mathsf{Wat}_{\mathsf{sk}}(u, Q) \to T$ *is a keyed randomized algorithm that takes as input a user $u \in \mathcal{U}$ and a prompt $Q \in \mathcal{T}^*$ and generates a response string $T \in \mathcal{T}^*$.*
- $\mathsf{Detect}_{\mathsf{sk}}(T) \to b$ *is a keyed deterministic algorithm that takes a string $T \in \mathcal{T}^*$ as input and outputs a bit $b \in \{0, 1\}$.*

- $\mathsf{Trace}_{\mathsf{sk}}(T) \to S$ *is a keyed deterministic algorithm that takes a string* $T \in \mathcal{T}^*$ *and outputs a set of accused users* $S \subseteq \mathcal{U}$.

Notice that detection could simply check if Trace outputs at least one user, so our definition could remove the explicit function Detect. However, detection alone may be much quicker or require less generated text compared to tracing. Both are true in our construction (Section 6): detection only requires checking if a partially-erased watermark is not empty, where extracting the partial mark takes $O(\log |\mathcal{U}|)$ time. Tracing amounts to checking the partial watermark against each user one-by-one, and may fail if too much of the mark is erased.

Although it makes sense to separate these forms of detection, we also hope that the two are *consistent*. In other words, if there is no detected watermark in the text, then we don't want to accuse any users of generating it. Alternatively, if we can find a user, then we should also detect that there is watermark.

Definition 3.8 (Consistency)**.** *We say a multi-user watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ *is* consistent *if for all* $T \in \mathcal{T}^*$ *and all keys* sk*,*

$$\mathsf{Detect}_{\mathsf{sk}}(T) = 0 \implies \mathsf{Trace}_{\mathsf{sk}}(T) = \emptyset.$$

As before, a multi-user watermarking scheme should be undetectable, sound, and robust.[6] The first two properties are straightforward generalizations of definitions for L-bit watermarking, and we defer them to Appendix B.3.

As for robustness, we can ask for both *robust detection* and *robust tracing*. By consistency, R-robust tracing implies R-robust detection. Motivated by the multi-user setting, we additionally consider robustness against *collusions*. If a group of users $u_1, \ldots, u_c$, each interacting with their own oracle, collude to produce some text $\hat{T}$, then we still wish that Trace can find one or more of the users. Our definition below captures this idea by allowing the adversary to query the model as c distinct users.

Definition 3.9 (R-Robust tracing against c-collusions – multi-user, adaptive)**.** *A multi-user watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ *is* R-robustly traceable against c-collusions *with respect to the robustness condition* R *and collusion bound* $c > 1$ *if for all* $C \subseteq \mathcal{U}$ *of size at most* $|C| \leq c$ *and all efficient adversaries* $\mathcal{A}$*, the following event* `FAIL` *occurs with negligible probability:*

- $\forall i, u_i \in C$*, AND* `// only users in C collude`
- $R\left(\lambda, (Q_i)_i, (T_i)_i, \hat{T}\right) = 1$*, AND* `// robustness condition passes`
- $\left(S = \emptyset \ \vee \ S \not\subseteq C\right)$ `// no or false accusation`

in the probability experiment defined by

- $\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)$
- $\hat{T} \leftarrow \mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot,\cdot)}(1^\lambda)$*, denoting by* $(u_i, Q_i)_i$ *and* $(T_i)_i$ *the sequence of inputs and outputs of the oracle*
- $S \leftarrow \mathsf{Extract}_{\mathsf{sk}}(\hat{T})$.

4 Block-by-block watermarks

This section introduces the syntax for *block-by-block* watermarking schemes and *AEB-robustness conditions*. These abstractions provide a unified way to describe the robustness guarantees of existing schemes that enables black-box reductions.

[6] We do not define completeness for multi-user watermarking, as it is just a special case of robustness, which we prove later in the paper.

Informally, a block-by-block scheme views a generation T as a sequence of *blocks* $\mathsf{Blocks}(T)$, each of which has high-enough entropy. AEB-robustness guarantees that candidate text is watermarked whenever it **A***pproximates* **E***nough* **B***locks* of model-generated text (see Figure 1). In general, these blocks do not need to be copied verbatim and in fact only need to be *approximated* in the candidate text. The AEB-robustness condition R_1 requires one block to be approximated. R_1 is satisfied by $\hat{T}$ if there exists a single block $\beta \in \cup_i \mathsf{Blocks}(T_i)$ that is approximated by some substring $\hat{\beta}$ of $\hat{T}$. For $k \geq 1$, the AEB-robustness condition R_k requires approximating k blocks. R_k checks whether at least k distinct blocks $\beta_j \in \cup_i \mathsf{Blocks}(T_i)$ are approximated by substrings $\hat{\beta}_j$ of $\hat{T}$. In our constructions, we will only require that the underlying scheme is R_1-robustly detectable. Using this underlying scheme, our multi-user construction (Section 6) will be R_1-robustly detectable and R_k-robustly traceable.

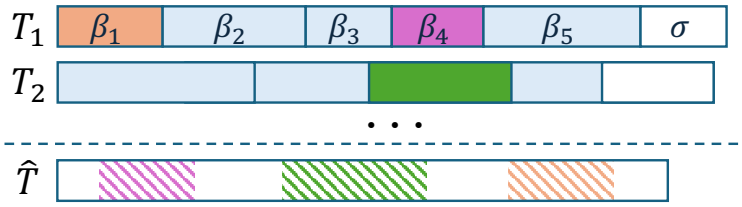

Figure 1: Visualization of a string $\hat{T}$ containing three approximate blocks from original generations T_1 and T_2. This $\hat{T}$ would satisfy the $R_3(\lambda, (Q_i)_i, (T_i)_i, \hat{T})$ robustness condition.

We now formalize these notions. To define the AEB-robustness condition R_1, we need a binary function block (which specifies whether a given substring $\beta \in T$ constitutes a block) and a binary relation on strings $\approx$ (which specifies when $\hat{\beta}$ approximates β) denoted by $\hat{\beta} \approx \beta$. Note that all the functions defined below may take the security parameter λ as an additional input. We omit it to reduce notational clutter.

As already discussed, a useful measure of the entropy of a generation must be with respect to the underlying query (and the language model). Thus we define block to take two strings as input, the generation T and the prompt Q.

Definition 4.1 (Block)**.** *Let* $\mathsf{block} : \mathcal{T}^* \times \mathcal{T}^* \to \{0,1\}$ *be a (deterministic, efficient) function. For* $Q \in \mathcal{T}^*$*, a* block *with respect to* Q *is a string* $T \in \mathcal{T}^*$ *such that* $\mathsf{block}(T; Q) = 1$*. A block is* minimal *if no prefix is a block.*

Definition 4.2 (Block-by-block watermarking)**.** *A* block-by-block *watermarking scheme* $\mathcal{W}$ *is a watermarking scheme that it has an additional block algorithm* $\mathsf{block} : \mathcal{T}^* \times \mathcal{T}^* \to \{0,1\}$*, which may depend on* λ *and* Model*, but not the secret key* sk*.*

It is easy to check that any generation of a prefix-specifiable model can be uniquely parsed into a sequence of minimal blocks (possibly with a non-block suffix), which we denote $\mathsf{Blocks}(T; Q)$.

Definition 4.3 ($\mathsf{Blocks}(T; Q)$)**.** *Let* $Q, T \in \mathcal{T}^*$ *be strings and* block *as above. We define* $\mathsf{Blocks}(T; Q)$ *to be the unique sequence* $(\beta_1, \beta_2, \ldots, \beta_B)$ *such that: (i)* $T = \beta_1 \| \beta_2 \| \cdots \| \beta_B \| \sigma$ *for some string* σ*; (ii)* β_i *is a minimal block with respect to* $Q \| \beta_1 \| \cdots \| \beta_{i-1}$ *for all* $i \in [B]$*; and (iii) no prefix of* σ *is a block with respect to* $Q \| \beta_1 \| \cdots \| \beta_B$*. We call* $\mathsf{Blocks}(T; Q)$ *the* blocks of T with respect to Q*.*

AEB-robustness conditions involve counting the number of substrings of $\hat{T}$ that approximate distinct blocks in a generation T, where the approximation is specified by the binary relation $\approx$, which we typically suppress throughout the paper for ease of notation. The function $\mathsf{NumBlocks}$ returns that count.

Definition 4.4 ($\mathsf{NumBlocks}$)**.** *Let* block *and* Blocks *as above, and let* $\approx : \mathcal{T}^* \times \mathcal{T}^* \to \{0,1\}$ *be a binary relation on strings. The function* $\mathsf{NumBlocks} : (\hat{T}; Q, T) \mapsto n$ *on input strings* $\hat{T}, Q, T \in \mathcal{T}^*$ *is defined to be the maximum* $n \geq 0$ *for which there exist substrings* $\hat{\beta}_1, \ldots, \hat{\beta}_n \in \hat{T}$ *and distinct blocks* $\beta_1, \ldots, \beta_n \in \mathsf{Blocks}(T; Q)$ *such that* $\hat{\beta}_i \approx \beta_i$ *for all* $i \in [n]$*.*

R_k holds when at least k substrings are present in a string $\hat{T}$ that approximate (according to some $\approx$) blocks contained in generations $(T_i)_i$ with respect to prompts $(Q_i)_i$.

Definition 4.5 (AEB-robustness condition R_k)**.** *Fix* $\mathsf{NumBlocks}$ *as above. For* $k \geq 1$*, the* AEB-robustness condition R_k *is*

$$R_k(\lambda, (Q_i)_i, (T_i)_i, \hat{T}) := \mathbb{1}\left(\sum_i \mathsf{NumBlocks}(\hat{T}; Q_i, T_i) \geq k\right).$$

We can easily compare AEB-robust watermarking schemes to one another. All else equal, a watermarking scheme is more robust if we relax any of "approximate", "enough", or "blocks." We may relax $\approx$, considering more strings as approximations of a given block. We may reduce k, requiring fewer approximate blocks. Or we may relax block, treating shorter strings as blocks.

Notice that we can trivially view any watermarking scheme as block-by-block and AEB-robust. Specifically, by defining entire generations as a single block and taking the equality relation, any complete scheme would be R_1 robust. However, this perspective is very weak. As we discuss in Appendix A, many schemes are block-by-block and AEB-robust for non-trivial block functions and relations.

4.1 Non-adaptive to adaptive robustness

Existing watermarking schemes provide robustness guarantees that are *non-adaptive* in the sense that they only allow adversaries to see a single watermarked output T before creating $\hat{T}$. See Definition B.3 in the Appendix. We show that a non-adaptive R_1-robust watermarking scheme is adaptively R_1-robust, so long as the scheme is undetectable.

Lemma 4.6 (Non-adaptive to adaptive robustness)**.** *Let* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ *be a block-by-block zero-bit watermarking scheme that is undetectable. If* $\mathcal{W}$ *is non-adaptively* R_1*-robustly detectable, then it is adaptively* R_1*-robust.*

See Lemma B.6 for a proof of the (stronger) statement for L-bit watermarking schemes and robust extraction. We do not believe the statement holds for general robustness conditions R.

5 Zero-bit to L-bit watermarks

In this section, we construct L-bit watermarking schemes from block-by-block, zero-bit watermarking schemes. Namely, if $\mathcal{W}'$ is a zero-bit watermarking scheme that is undetectable, sound and R_1-robust, the resulting $\mathcal{W}$ is an L-bit scheme that is undetectable, sound, and R_k-robust, with $k = O(L\lambda)$. Our construction is black-box with respect to the robustness condition of $\mathcal{W}'$, requiring only that it is R_1-robust for some underlying functions block and $\approx$. The resulting scheme is R_k-robust where R is induced by the same block and $\approx$ as R_1.

Section 5.1 gives the construction and Section 5.2 proves robustness. Section 5.3 analyzes the resulting value of k. It suffices to set $k = O(L\lambda)$ to losslessly recover the message. For long messages ($L = \Omega(\lambda)$), recovering a constant $(1 - \delta)$ fraction of the messages requires only $O(L)$ blocks.

5.1 Constructing L-bit watermarks

Our construction is given in Figure 2. Let $\mathcal{W}' = (\mathsf{KeyGen}', \mathsf{Wat}', \mathsf{Detect}')$ be a zero-bit watermarking scheme. We construct an L-bit scheme $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ as follows. The secret key sk consists of $2L$ zero-bit keys $k_{i,b} \leftarrow \mathsf{KeyGen}'(1^\lambda)$ sampled independently, for all $i \in [L]$ and $b \in \{0,1\}$. The keys $k_{i,0}$ and and $k_{i,1}$ are used to embed the ith bit of a message $m \in \{0,1\}^L$. To do so, the watermarked model $\mathsf{Wat}_{\mathsf{sk}}(m, Q)$ repeatedly samples a new block of text by calling Wat' with key $k_{i,m[i]}$ for uniformly random index $i \leftarrow\!\!\$\ [L]$. What constitutes a block of text is determined by $\mathcal{W}'$, which is assumed to be a block-by-block scheme. The generated block is added to the current generation and the process is repeated. The loop exits when the call to Wat' fails to generate a full block of text. To extract the message from $\hat{T}$, the algorithm $\mathsf{Extract}_{\mathsf{sk}}(\hat{T})$

$\underline{\mathsf{KeyGen}(1^\lambda)}$
For $i = 1, \ldots, L$:
  $k_{i,0} \leftarrow\!\!\$\ \mathsf{KeyGen}'(1^\lambda)$
  $k_{i,1} \leftarrow\!\!\$\ \mathsf{KeyGen}'(1^\lambda)$
Return $\mathsf{sk} = (k_{i,0}, k_{i,1})_{i=1}^{L}$

$\underline{\mathsf{Extract}_{\mathsf{sk}}(\hat{T})}$
For $k_{i,b} \in \mathsf{sk}$:
  $z_{i,b} \leftarrow \mathsf{Detect}'_{k_{i,b}}(\hat{T})$
For $i = 1, \ldots, L$:

$$\hat{m}_i \leftarrow \begin{cases} \perp & \text{if } z_{i,0} = z_{i,1} = \mathsf{false} \\ 0 & \text{if } z_{i,0} = \mathsf{true} \\ 1 & \text{otherwise} \end{cases}$$

Return $\hat{m} = \hat{m}_1 \hat{m}_2 \ldots \hat{m}_L$

$\underline{\mathsf{Wat}_{\mathsf{sk}}(m, Q)}$
$T \leftarrow \epsilon$
While true:
  $k \leftarrow\!\!\$\ \{k_{i,b} : i \in [L], b = m[i]\}$
  $T' \leftarrow\!\!\$\ \mathsf{Wat}'_k(Q \| T)$
  $(\beta_1, \beta_2, \ldots) \leftarrow \mathsf{Blocks}(T'; Q \| T)$
  If $\beta_1 = \perp$: Exit loop
  $T \leftarrow T \| \beta_1$
$T \leftarrow T \| T'$
Return T

Figure 2: Pseudocode for L-bit watermarking scheme of $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ from a block-by-block zero-bit watermarking scheme $\mathcal{W}' = (\mathsf{KeyGen}', \mathsf{Wat}', \mathsf{Detect}')$.

runs $\mathsf{Detect}'(\hat{T})$ algorithm using every key $k_{i,b}$. The ith bit of extracted message is determined by which of the keys $k_{i,0}$ or $k_{i,1}$ a (zero-bit) mark was detected.

We now show that our scheme is undetectable and sound. Robustness (which implies completeness) is more involved, and is deferred to Section 5.2. In the following, let $\mathcal{W}'$ be a zero-bit watermarking scheme and $\mathcal{W}$ be the L-bit watermarking scheme described in Figure 2.

Claim 5.1 ($\mathcal{W}$ is undetectable)**.** *If* Model *is prefix-specifiable and* $\mathcal{W}'$ *is undetectable, then* $\mathcal{W}$ *is undetectable.*

Proof. By the undetectability of $\mathcal{W}'$, one can replace every call to $\mathsf{Wat}'_k(Q\|T)$ with $\mathsf{Model}(Q\|T)$, with negligible effect on the adversary's output distribution. The result is a modified version of $\mathcal{W}$ that generates the response to prompt Q by iteratively calling $\mathsf{Model}(Q\|T)$ with T the prefix generated so far. Because Model is prefix-specifiable, this is the same distribution as $\mathsf{Model}(Q)$. □

Claim 5.2 ($\mathcal{W}$ is sound)**.** *Let* $\mathcal{W}'$ *be a block-by-block zero-bit watermarking scheme. If* $\mathcal{W}'$ *is sound then* $\mathcal{W}$ *is sound.*

Proof. The soundness of $\mathcal{W}$ follows immediately from the soundness of $\mathcal{W}' = (\mathsf{KeyGen}', \mathsf{Wat}', \mathsf{Extract}')$. Specifically, observe that for a given $\mathsf{sk} = (k_{i,0}, k_{i,1})_{i=1}^{L}$, $\mathsf{Extract}_{\mathsf{sk}}(T) \neq \perp^L$ only when $\mathsf{Extract}'_{k_{i,b}} \neq \perp$ for some $i \in [L]$ and $b \in \{0, 1\}$. So, for every polynomial p, λ, and T with $|T| \leq p(\lambda)$,

$$\Pr_{\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathsf{Extract}_{\mathsf{sk}}(T) \neq (\perp)^L] \leq 2L \Pr_{k \leftarrow \mathsf{KeyGen}'(1^\lambda)}[\mathsf{Extract}'_k(T) \neq \perp] < \mathsf{negl}(\lambda),$$

via a union bound over every call to $\mathsf{Extract}'$. □

5.2 Robustness of L-bit watermarks

In this section, we prove robustness of our scheme (Theorem 5.4). Namely, our L-bit watermarking scheme is (δ, R_k)-robust for any $k \geq k^*(L, \delta)$ as defined in Lemma 2.4. The parameter $k^* \leq L \ln L + L\lambda = O(L\lambda)$ for all δ (see Section 5.3 for discussion). Note that k^* is independent of the total amount of generated text seen by the adversary!

We actually prove a stronger result, Lemma 5.3, of which Theorems 5.4 and 6.4 are both corollaries. In the theorems and proofs, we use the notation from Definition 2.2 to denote feasible sets.

Lemma 5.3 ($\mathcal{W}$ recovers lossy descendants)**.** *Let $L \in \mathbb{N}$, $0 \leq \delta < 1$, $k \geq k^*(L, \delta)$, and let $M \subseteq \{0,1\}^L$. Suppose $\mathcal{W}'$ is a block-by-block zero-bit watermarking scheme that is undetectable, sound, and R_1-robustly detectable. Let $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ be the construction from Figure 2 using $\mathcal{W}'$.*

Then for all efficient $\mathcal{A}$, the following event `FAIL` *occurs with negligible probability:*

- $\forall i, m_i \in M$, *AND* `// only queried messages in M`
- $R_k\left(\lambda, (Q_i)_i, (T_i)_i, \hat{T}\right) = 1$, *AND* `// robustness condition passes`
- $\hat{m} \notin F_\delta(M)$ `// extracted message unrelated to M`

in the probability experiment defined by

- $\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)$
- $\hat{T} \leftarrow \mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot,\cdot)}(1^\lambda)$, *denoting by $(m_i, Q_i)_i$ and $(T_i)_i$ the sequence of inputs and outputs of the oracle*
- $\hat{m} \leftarrow \mathsf{Extract}_{\mathsf{sk}}(\hat{T})$.

Proof. We will prove Lemma 5.3 via three hybrid transitions, moving from the experiment defined in the lemma statement to one where all of the calls to the underlying zero-bit watermarking algorithm $\mathcal{W}'$ are replaced by calls to the robustness condition R_1 and to Model. In Hybrid 1, we use the soundness of $\mathcal{W}'$ to remove all calls to Detect' that use keys unrelated to any messages $m \in M$. In Hybrid 2, we use the R_1-robustness of $\mathcal{W}'$ to remove the calls to Detect' that use the rest of the keys, this time replacing them with calls to R_1. Finally, in Hybrid 3 we rely on the undetectability of $\mathcal{W}'$ to replace all of the calls to Wat' with calls to Model. We then use the definition of R_k-robustness and our choice of k to complete the proof.

Let $p_{\texttt{FAIL}}$ be the probability of the event `FAIL` in the experiment defined in the lemma statement. Throughout this proof, we condition on the events $(\forall i, m_i \in M)$ and $R_k(\lambda, (Q_i)_i, (T_i)_i, \hat{T}) = 1$, both of which are efficiently checkable by $\mathcal{A}$ and implied by `FAIL`.

Hybrid 1 (Soundness) Consider the set of keys $\mathcal{K}_M = \{k_{i,m[i]} : i \in [L], m \in M\}$, and its complement $\overline{\mathcal{K}}_M$. In Hybrid 1, we modify $\mathsf{Extract}_{\mathsf{sk}}$ to remove every call to $\mathsf{Detect}'_{k_{i,b}}(\hat{T})$ for $k \in \overline{\mathcal{K}}_M$, replacing it with $z_{i,b} \leftarrow \mathsf{false}$. Let $\hat{m}_1$ be the result of $\mathsf{Extract}_{\mathsf{sk}}(\hat{T})$ in Hybrid 1. By construction, if $\hat{m}_1[i] \neq \perp$, then $\hat{m}_1[i] = m[i]$ for some $m \in M$. In other words, although there may be many $\perp$ entries, non-$\perp$ entries of $\hat{m}_1$ must agree with some element of M, so $\hat{m}_1 \in F_{\gamma_1}(M)$ for some $\gamma_1 \leq 1$.

Observe that the keys in $\overline{\mathcal{K}}_M$ are never used by $\mathsf{Wat}_{\mathsf{sk}}$. Hence the view of $\mathcal{A}$ — and in particular its output $\hat{T}$ — is independent of the keys $\overline{\mathcal{K}}_M$. By the soundness of $\mathcal{W}'$, every call to $\mathsf{Detect}'_k(\hat{T})$ for $k \in \overline{\mathcal{K}}_M$ returns 0 with high probability. Notice that $\mathsf{Extract}_{\mathsf{sk}}$ does not change its behavior whether a call to Detect' returns 0 or is removed entirely (the only difference between the real game and Hybrid 1), since each $z_{i,b}$ is initialized to false in the hybrid.

Therefore, the output distribution of $\mathsf{Extract}_{\mathsf{sk}}$ in Hybrid 1 and the real execution are statistically close (conditioned on $\forall i, m_i \in M$). Let p_1 be the probability of the event `FAIL` in Hybrid 1. As $\mathsf{Extract}_{\mathsf{sk}}$ changed only negligibly between the hybrids, we have that

$$|p_1 - p_{\texttt{FAIL}}| \leq \mathsf{negl}(\lambda).$$

Hybrid 2 (Robustness) The pseudocode for Hybrid 2 is given in Figure 3. In this hybrid, we remove the remaining calls to $\mathsf{Detect}'_{k_{i,b}}(\hat{T})$ for $k \in \mathcal{K}_M$, replacing each with a call to the robustness condition R_1 (which requires the relevant set of queries and responses as input). So, for each $k_{i,b}$, the corresponding call to R_1 is run on $\mathcal{Q}_{i,b}$ and $\mathcal{G}_{i,b}$, the sequences of *queries* to and *generations* from $\mathsf{Wat}'_{k_{i,b}}$ in $\widetilde{\mathsf{Wat}}_{\mathsf{sk}}$. The code of $\widetilde{\mathsf{Wat}}_{\mathsf{sk}}$ is edited to track $\mathcal{Q}_{i,b}$ and $\mathcal{G}_{i,b}$.

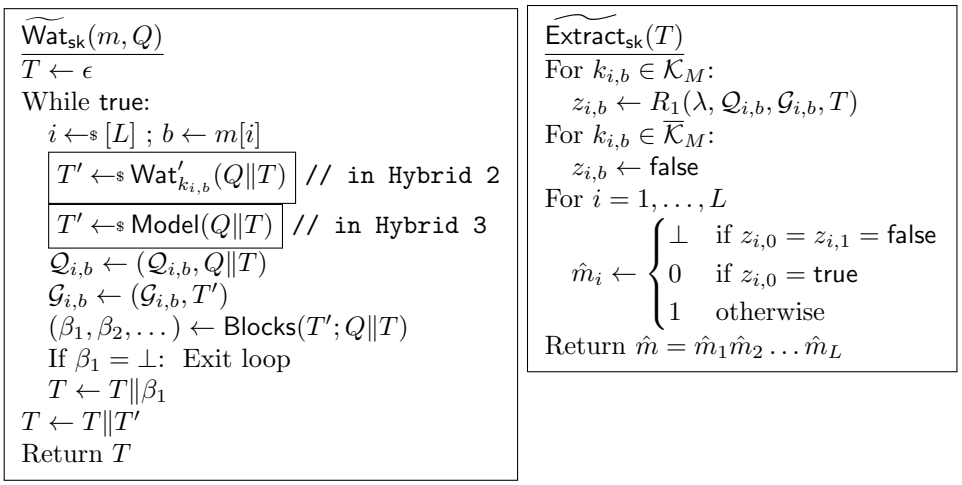

Figure 3: Intermediate versions of Wat and $\mathsf{Extract}$, used to produce outputs that are independent of the keys, used in the proof of Lemma 5.3. The boxed lines are used only in the indicated hybrid. At setup, we additionally initialize $\mathcal{Q}_{i,b}, \mathcal{G}_{i,b}$ to ( ) for all i, b.

Let $\hat{m}_2$ be the result of $\widetilde{\mathsf{Extract}}_{\mathsf{sk}}(\hat{T})$ in Hybrid 2. As in Hybrid 1, $\hat{m}_2 \in F_{\gamma_2}(M)$ for some $\gamma_2 \leq 1$. Moreover, $\gamma_2 \geq \gamma_1$, meaning that $\hat{m}_2$ can only contain more $\perp$-entries than $\hat{m}_1$. This is because $R_1(\lambda, \mathcal{Q}_{i,b}, \mathcal{G}_{i,b}, \hat{T}) = 1$ implies that $\mathsf{Detect}'_{k_{i,b}}(\hat{T}) = 1$ with high probability (but not the converse), because $\mathcal{W}'$ is R_1-robust.[7]

Let p_2 be the probability of the event FAIL in Hybrid 2. For $\hat{m}_1 \in F_{\gamma_1}(M)$, $\hat{m}_2 \in F_{\gamma_2}(M)$, and $\gamma_2 \geq \gamma_1$, we have

$$\Pr[\hat{m}_2 \notin F_\delta(M)] = \Pr[\gamma_2 > \delta] \geq \Pr[\gamma_1 > \delta] = \Pr[\hat{m}_1 \notin F_\delta(M)].$$

Hence,

$$p_2 \geq p_1 - \mathsf{negl}(\lambda).$$

Hybrid 3 (Undetectability) The pseudocode for Hybrid 3 is given in Figure 3. In Hybrid 3, we remove all use of the watermarking scheme $\mathcal{W}'$ by replacing Wat' with Model. In these functions, we no longer sample generations from Wat' and instead use Model. Observe that in Hybrid 3, the adversary's view is independent of the indices i sampled by $\widetilde{\mathsf{Wat}}$.

Let p_3 be the probability of FAIL in Hybrid 3. By undetectability of $\mathcal{W}'$

$$|p_3 - p_2| \leq \mathsf{negl}(\lambda).$$

At most $\lfloor \delta L \rfloor$ erasures Recall we are conditioning on $R_k(\lambda, (Q_j)_j, (T_j)_j, \hat{T}) = 1$. By definition, there are $k' \geq k$ substrings $\hat{\tau} \in \hat{T}$ that are $\doteq$-close to disjoint blocks $\beta \in \cup_j \mathsf{Blocks}(T_j; Q_j)$. Each such block β has an associated index i_β: the index that was sampled by $\widetilde{\mathsf{Wat}}_{\mathsf{sk}}$ in the iteration that generated β. By construction, a bit is extracted at each of these indices: $\hat{m}[i_\beta] \neq \perp$.

The indices i_β are uniform over $[L]$ and independent of one another. Hence, the number $N_\perp$ of indices j where $\hat{m}[j] = \perp$ is distributed as the number of empty bins remaining after throwing k' balls into L bins uniformly at random. By Lemma 2.4 and the hypothesis that $k' \geq k \geq k^*(L, \delta)$, we have that $N_\perp \leq \lfloor \delta L \rfloor$ except with probability $e^{-\lambda}$. We know that $\hat{m} \in F_1(M)$ because $\hat{m}_1 \in F_{\gamma_1}(M)$ (with high probability) and

[7] Otherwise, we construct adversary $\mathcal{A}'$ breaking the R_1-robustness of $\mathsf{Wat}'_{k_{i,b}}$, as follows: $\mathcal{A}'$ runs $\mathcal{A}$, internally simulating its $\widetilde{\mathsf{Wat}}$ oracle, only querying its own Wat' oracle to simulate calls to $\mathsf{Wat}'_{k_{i,b}}$. $\mathcal{A}'$ outputs the string $\hat{T}$ returned by $\mathcal{A}$. Notice that this critically uses the adaptivity of our robustness definition. However, because we assume $\mathcal{W}'$ is undetectable, non-adaptive robustness (Definition B.3) would suffice via Lemma 4.6.

the two games are negligibly different. Combining this fact with our argument about $N_\perp$ shows that we have $\hat{m} \in F_\delta(M)$, with high probability. Hence

$$p_3 \leq \mathsf{negl}(\lambda) \implies p_{\texttt{FAIL}} \leq \mathsf{negl}(\lambda) \qquad \square$$

Theorem 5.4 ($\mathcal{W}$ is robust)**.** *Suppose $\mathcal{W}'$ is a block-by-block zero-bit watermarking scheme that is undetectable, sound, and R_1-robustly detectable. Then the $\mathcal{W}$ construction from Figure 2 is an L-bit watermarking scheme that is sound, undetectable, and (δ, R_k)-robustly extractable for $k \geq k^*(L, \delta)$.*

Proof. By Claims 5.2 and 5.1, $\mathcal{W}$ is sound and undetectable. Robustness is an an immediate corollary of Lemma 5.3, fixing the subset $M \subseteq \{0,1\}^L$ to be a singleton set. If an adversary can only query a fixed message $m \in M$, then the event bounded in Lemma 5.3 is exactly the definition of (δ, R_k)-robustness. $\square$

5.3 How good is R_{k^*}?

Theorem 5.4 states that our scheme is (δ, R_{k^*})-robust where:

$$k^*(L, \delta) = \min \left\{ L \cdot (\ln L + \lambda); \quad L \cdot \ln \left(\frac{1}{\delta - \sqrt{\frac{\lambda + \ln 2}{2L}}} \right) \right\} \tag{2}$$

This means that our watermarking scheme embeds L-bit messages into model-generated text T such that at least a $(1-\delta)$-fraction of the embedded message can be recovered from any text $\hat{T}$ containing at least $k^*(L, \delta)$ approximate blocks from T. We remark that as our construction doesn't depend on δ, it satisfies (δ, R_{k^*})-robustness for all $0 \leq \delta < 1$ simultaneously.

We'd like k to be as small as possible, for two reasons. First, because smaller k means that $\hat{T}$ can be farther from T — guaranteeing stronger robustness. Second, because k blocks are required to extract the mark even in the absence of an adversary! Language models have variable-length outputs, and too-short T are not marked. Smaller k means that more of model's outputs are marked. But we cannot make k too small without a very different approach. Any scheme that embeds each bit into a distinct block of text requires $k \geq L(1-\delta) = O(L)$.

So how does k^* compare to the $L(1-\delta)$ lower bound? We consider two parameter regimes. For $L < (\lambda + \ln 2)/2$, we have $k^* = L(\ln L + \lambda)$. As $L = \text{poly}(\lambda)$, we have that $k^* = O(L\lambda)$ and is independent of δ. For $L > (\lambda + \ln 2)/2$, the minimum is achieved at $\delta > 0$. Taking $c = \sqrt{2L/(\lambda + \ln 2)}$ and $\delta > 1/c$, we have that $k^* \leq L \ln \left(\frac{1}{\delta - 1/c} \right) = O(L)$. (Even better, $k^* < L$ for $\delta > 1/c + 1/e$.)

A possible approach for further improving the parameter k is to use error correcting codes. The idea is simple. Let ECC be an error correcting code with block-length $L' > L$. To losslessly embed a mark $m \in \{0,1\}^L$, embed the codeword $w = \mathsf{ECC}(m)$ using an L'-bit watermarking scheme. If $\hat{w}$ can be extracted with at most δ fraction of erasures, we can decode and recover m in its entirety. The result would be a lossless $R_{k'}$-robust scheme for $k' = k^*(L', \delta)$. If $L' = o(L\lambda)$, this yields an asymptotic improvement $k' = o(k) = o(L\lambda)$.

Comparison to [CG24] While most prior work constructs zero-bit watermarking, Christ and Gunn [CG24] build both zero-bit and (lossless) L-bit watermarking schemes, from zero/L-bit pseudorandom error-correcting codes respectively. The L-bit scheme of [CG24] and our L-bit scheme instantiated with the zero-bit scheme of [CG24] have incomparable robustness guarantees. Roughly speaking, their zero-bit scheme is R_1-robust for blocks that require $O(\lambda)$ empirical entropy. Their L-bit scheme is also R_1-robust, but with "longer" blocks requiring $O(L + \lambda)$ empirical entropy. Importantly this block of text has to be a single *contiguous* block that was produced in *one generation*.

In contrast, our L-bit scheme is R_k-robust, requiring $k = O(L\lambda)$ of the original, zero-bit blocks. While it requires more empirical entropy overall, R_k-robustness allows these k blocks to appear *anywhere* in all of the generations ever seen by the adversary.

6 Building multi-user watermarks

In this section, we construct a multi-user watermarking scheme using L-bit watermarking schemes and robust fingerprinting codes as black boxes. When instantiated with our L-bit scheme from Section 5, the result is robust against colluding users as well. To our knowledge, ours is the first watermarking scheme that is secure against any sort of collusion. The black-box nature of our construction ensures that we can instantiate our multi-user schemes with improved parameters whenever the underlying watermarking schemes or fingerprinting codes improve.

Section 6.1 gives the construction of our multi-user scheme and proves its undetectability, consistency, and soundness, using the undetectability and soundness of the underlying L-bit scheme. In Section 6.2 we prove robustness of our multi-user scheme, which follows from the robustness of the fingerprinting code and our own L-bit scheme (Theorem 5.3). Finally, in Section 6.3 we analyze the key features of our multi-user scheme and compare our approach to other possible constructions.

6.1 Constructing multi-user watermarks

$\underline{\mathsf{KeyGen}(1^\lambda)}$
$(X, \mathsf{tk}) \leftarrow\!\!\$\ \mathsf{FP.Gen}'(1^\lambda, n, c, \delta)$
$\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}'(1^\lambda)$
Return $(X, \mathsf{tk}, \mathsf{sk})$

$\underline{\mathsf{Wat}_{(X,\mathsf{tk},\mathsf{sk})}(u, Q)}$
$T \leftarrow\!\!\$\ \mathsf{Wat}'_{\mathsf{sk}}(X_u, Q)$
Return T

$\underline{\mathsf{Detect}_{(X,\mathsf{tk},\mathsf{sk})}(\hat{T})}$
$\hat{m} \leftarrow \mathsf{Extract}'_{\mathsf{sk}}(\hat{T})$
Return $\mathbb{1}(\exists i\ \hat{m}_i \neq \perp)$

$\underline{\mathsf{Trace}_{(X,\mathsf{tk},\mathsf{sk})}(\hat{T})}$
$\hat{m} \leftarrow \mathsf{Extract}'_{\mathsf{sk}}(\hat{T})$
If $\hat{m} = \perp^L$:
  Return $\emptyset$
$C \leftarrow \mathsf{FP.Trace}'(\hat{m}, \mathsf{tk})$
Return C

Figure 4: Pseudocode for construction of $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ from fingerprinting code $\mathsf{FP} = (\mathsf{FP.Gen}', \mathsf{FP.Trace}')$ and L-bit message embedding scheme $\mathcal{W}' = (\mathsf{KeyGen}', \mathsf{Wat}', \mathsf{Extract}')$, e.g. Figure 2. The construction is defined for any setting of the parameters $n, c > 1$, and $0 \leq \delta < 1$.

Our construction is given in Figure 4. Let $\mathcal{W}' = (\mathsf{KeyGen}', \mathsf{Wat}', \mathsf{Extract}')$ be an L-bit watermarking scheme. The multi-user watermarking scheme $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ is constructed as follows. The secret key output by KeyGen consists of the fingerprinting codewords X and the tracing key tk, as well as the secret key sk from the L-bit scheme. In response to any prompt Q from a user u, Wat will use the L-bit watermarking algorithm to watermark the user's fingerprinting codeword X_u into the response. At detection time, Detect will use the L-bit scheme to extract a message $\hat{m}$ from $\hat{T}$ and return 1 as long as $\hat{m} \neq \perp^L$. To trace a user, Trace will run the fingerprinting code's tracing algorithm on $\hat{m}$ and return the set of accused users.

We now show that our black-box scheme is consistent, undetectable, and sound. Robustness to collusions (and hence completeness) is deferred to Section 6.2, because it requires instantiating our multi-user scheme with our L-bit watermarking scheme.

Claim 6.1 ($\mathcal{W}$ is consistent)**.** *Let $L, n, c > 1$ be integers and $0 \leq \delta < 1$. Let $\mathcal{W}'$ be an L-bit watermarking scheme and FP be a fingerprinting code. Then the $\mathcal{W}$ construction from Figure 4 is a consistent multi-user watermarking scheme.*

Proof. The algorithm $\mathsf{Detect}_{(X,\mathsf{tk},\mathsf{sk})}(T) = 0$ only if $\mathsf{Extract}_{\mathsf{sk}}(T)$ returns $\perp^L$. The check in Trace will guarantee that in such a case, $\mathsf{Trace}_{(X,\mathsf{tk},\mathsf{sk})}(T) = \emptyset$.[8] □

[8]Note that we may instead want to check if $|\{i : s_i = \perp\}| > \delta L$, because in the formal fingerprinting games, FP is allowed

Whenever our scheme is instantiated with an undetectable L-bit watermarking scheme, the overall output will also be undetectable, because Wat just returns the value from the underlying Wat query. The proof is omitted, as it is essentially identical to the proof of Claim 5.1. Together with Claim 5.1, we obtain undetectable watermarking from a black-box undetectable zero-bit watermarking scheme.

Claim 6.2 ($\mathcal{W}$ is undetectable)**.** *If* Model *is prefix-specifiable and* $\mathcal{W}'$ *is an* L*-bit watermarking scheme built from an undetectable zero-bit scheme, then the* $\mathcal{W}$ *construction from Figure 4 is undetectable.*

Next we show that $\mathcal{W}$ is sound, as long as the underlying L-bit scheme is sound. We need this property to ensure that we do not falsely detect a watermark when it is not present (Type I errors). Notice that because our scheme is also consistent we will not falsely accuse users of generating unmarked text.

Claim 6.3 ($\mathcal{W}$ is sound)**.** *Let* $L, n, c > 1$ *be integers and* $0 \leq \delta < 1$*. Let* $\mathcal{W}'$ *be a sound* L*-bit watermarking scheme and* FP *be a fingerprinting code of length* L *with parameters* (λ, n, c, δ)*. Then the* $\mathcal{W}$ *construction from Figure 4 is a sound multi-user watermarking scheme.*

Proof. The soundness of $\mathcal{W}$ follows immediately from the soundness of $\mathcal{W}' = (\mathsf{KeyGen}', \mathsf{Wat}', \mathsf{Extract}')$. Specifically, observe that $\mathsf{Detect}_{(X,\mathsf{tk},\mathsf{sk})}(T) = 1$ implies $\mathsf{Extract}'_{\mathsf{sk}}(T) \neq \perp^L$. So, for a every poly, λ, and T with $|T| \leq p(\lambda)$

$$\Pr_{(X,\mathsf{tk},\mathsf{sk}) \leftarrow\$ \mathsf{KeyGen}(1^\lambda)}[\mathsf{Detect}_{(X,\mathsf{tk},K)}(T) = 1] \leq \Pr_{\mathsf{sk}' \leftarrow\$ \mathsf{KeyGen}'(1^\lambda)}[\mathsf{Extract}'_{\mathsf{sk}'}(T) \neq \perp^L] < \ \mathsf{negl}(\lambda). \square$$

6.2 Robustness against collusions

We now describe how our multi-user scheme from Figure 4 can achieve robust collusion resistance under the right conditions.

Our main theorem for multi-user watermarking, Theorem 6.4, requires that $\mathcal{W}$ is built out of the L-bit watermarking scheme from Figure 2, which itself is built out of an undetectable zero-bit watermarking scheme. As shown in Lemma 5.3, our L-bit scheme will only ever extract (noisy) feasible messages of the set of the adversary's queried messages.

Since the messages that $\mathcal{W}$ watermarks are codewords from a robust fingerprinting code, the $\hat{m}$ recovered will necessarily be in the δ-feasible ball around the set of the adversary's codewords. As long as enough blocks are included in the adversarially generated text $\hat{T}$, we will be able to trace back to one of the colluding users.

Theorem 6.4 ($\mathcal{W}$ is robust)**.** *Let* $n, c > 1$ *be integers and* $0 \leq \delta < 1$*. Let* $\mathcal{W}'$ *be the* L*-bit watermarking scheme from Figure 2, built from a block-by-block zero-bit watermarking scheme that is undetectable, sound, and* R_1*-robustly detectable. Furthermore let* FP *be a robust fingerprinting code of length* L *with parameters* (λ, n, c, δ)*.*

Then, the $\mathcal{W}$ *construction from Figure 4 is a multi-user watermarking scheme that is consistent, sound, undetectable, and* R_{k^*}*-robust against* c*-collusions, for* k^* *given by Lemma 2.4.*

Proof. We have already shown that $\mathcal{W}$ is consistent (Claim 6.1), sound (Claim 6.3), and undetectable (Claim 6.2). Robustness is a corollary of Lemma 5.3 when using an appropriate fingerprinting code. Let C be the set of (at most c) colluding users and apply Lemma 5.3 to $M = \{X_u : u \in C\}$. Then we know that $\mathsf{Extract}'$ from $\mathcal{W}'$ will return some $\hat{m}$ with at most $\lfloor \delta L \rfloor$ entries that are $\perp$. Therefore, with high probability we have $\hat{m} \in F_\delta(X_C)$. By the definition of a robust fingerprinting code we know that $\mathsf{FP}.\mathsf{Trace}'$ will correctly accuse a colluding user with all but negligible probability. $\square$

Efficiency Notice that the construction in Figure 4 only requires a single call to the underlying L-bit scheme for every generation and detection. When tracing, it additionally only requires a single call to the fingerprinting code's tracing algorithm. When instantiated with the L-bit scheme in Figure 2, detecting and tracing make $2L$ calls to its the underlying zero-bit scheme's detection algorithm. Fingerprinting code

to output anything on other inputs. However, for non-contrived schemes, this is unnecessary.

lengths scale as the logarithm of the number of users, so our scheme is much faster (and requires much less storage) than one which generates keys for every single user, whose detection would require linear time.

Unfortunately, existing fingerprinting codes require time linear in the number of users to trace, so our tracing time still scales linearly. Fingerprinting tracing algorithms work by checking whether the extracted codeword is sufficiently close to each user's unique codeword, one by one. This means that Trace could be used to check any set of c suspects in time linear in c. It is an interesting open problem to improve the runtime of the fingerprint tracing, which would correspondingly improve our scheme.

6.3 Other features of our multi-user watermarks

We discuss additional features of our construction that are not captured by the above definitions or theorems, but which offer practical improvements

Preserved zero-bit detection Theorem 6.4 proves that $\mathcal{W}$ is R_k-robustly traceable so long as the constructions in Figure 2 and Figure 4 use a zero-bit scheme that is R_1-robustly detectable. Happily, the multi-user construction from Figure 4 is *also* R_1-robustly detectable! In particular, the Detect function will detect if any single (approximate) block is present in a generation. This means that our construction preserves the original robustness of the zero-bit watermarking scheme. The added benefit of finding users and resisting (unbounded!) collusions comes essentially for free: no cost to robust detection and only a logarithmic slowdown in Detect.

Recovering more users In our multi-user construction, we use the function $\mathsf{Extract}'$ from the L-bit scheme as a black box, returning a single bitstring which is then fed into the fingerprinting code's tracing algorithm to accuse some set of users. However, a different construction may be able to recover even more colluding users. Slightly modifying the $\mathsf{Extract}'$ function from Figure 2, we could allow $\mathsf{Extract}'$ to return a special character "$*$" in the ith index whenever both $z_{i,0}$ and $z_{i,1}$ are true (since it recovered both bits in the same index). By the soundness of the underlying zero-bit scheme, this should almost never happen when embedding a single message, as in the normal L-bit robustness game. In the process of colluding, however, it is likely that users with different codeword bits at index i happen to include (approximate) blocks in $\hat{T}$ for both of their bits.

Therefore, the set of *all strings* which could be created from the extracted message $\hat{m} \in \{0,1,\perp,*\}^L$ is a subset of the δ-feasible ball $F_\delta(X_C)$ of the colluding users' codewords. In practice, one may want to call $\mathsf{FP.Trace}'$ on each of these strings and return the union of all users returned, which will still (with high probability) be a subset of the colluding users. An interesting question is to design fingerprinting codes that allow faster tracing from pirate codewords $\hat{m} \in \{0,1,\perp,*\}^L$ than brute-force search, which requires time exponential in the number of $*$'s.

Robust fingerprinting codes reduce k^* Notice that our construction of multi-user watermarking uses a robust fingerprinting code with erasure bound δ in conjunction with an L-bit watermarking scheme that allows up to δ erasures. The length L of the fingerprinting code grows with δ. Ultimately, the parameter we are most interested in is the number of (approximate) blocks $k^* = k^*(L,\delta) = \Omega(L)$ needed to extract the watermark from $\hat{T}$.[9]

This raises the question of whether robustness ($\delta > 0$) is helping at all, or whether we would be better off using shorter fingerprinting codes without adversarial robustness ($\delta = 0$). Perhaps surprisingly, robustness yields an asymptotic[10] improvement in k^*.

[9] Improving other robustness parameters, like the $\doteq$-relation and block length are received in a black-box way from the underlying zero-bit scheme and therefore can be improved immediately as future work develops.

[10] Showing a concrete improvement with $\delta > 0$ amounts to the same comparison between $k^*(L_0, 0)$ and $k^*(L_\delta, \delta)$ as in the body, but with concretely optimal fingerprinting codes. It appears that the construction of [NFH+07] is much more efficient than either of the asymptotically optimal codes we discuss. Like other fingerprinting codes, $L_\delta = L_0/\mathrm{poly}(1-\delta)$. Hence taking a constant $\delta \gg \sqrt{(\lambda + \ln 2)/2L_\delta}$ (say, $\delta = 1/2$) should yield $k^*(L_\delta, \delta) = O(L_\delta) = O(L_0)$, compared to $k^*(L_0, 0) = O(L_0\lambda)$. However, we were unable to work out the details of [NFH+07] to our satisfaction.

The question boils down to comparing $k^*(L_\delta, \delta)$ and $k^*(L_0, 0)$, where L_0 and L_δ are the lengths of the asymptotically optimal fingerprinting codes for $\delta = 0$ and $\delta > 0$, respectively. For n users, c collusions, δ adversarial erasures, and security parameter λ, the asymptotically optimal robust fingerprinting code of [BKM10] has length

$$L_\delta = \frac{C(c \ln c)^2 \ln(n)\lambda}{1-\delta}$$

for some very large constant C. Letting $W := C(c \ln c)^2 \ln(n)$, we have $L_\delta = W\lambda/(1-\delta)$. The asymptotically optimal non-robust fingerprinting code [Tar08] has length $L_0 = 100c^2 \ln(n)\lambda$. Choosing $\delta = 1/2$, we get $k^*(L_{\frac{1}{2}}, 1/2) = O(c^2 \ln^2(c) \ln(n)\lambda)$, whereas $k^*(L_0, 0) = \Omega(c^2 \ln(n)\lambda^2)$. As $c = \text{poly}(\lambda)$, we have $k^*(L_{\frac{1}{2}}, 1/2) = o(L_0, 0)$.

7 Watermarking without undetectability

Throughout the paper, we make heavy use of the *undetectability* of watermarking schemes. While some watermarking schemes for language models are provably undetectable [CGZ23, FGJ+23, CG24], most are not (e.g. [Aar22, KGW+23a, KTHL23]). Our constructions of L-bit and multi-user watermarking (Fig. 2, 4) use an arbitrary zero-bit watermarking scheme as a building block.

In this section, we explore the robustness of our schemes when instantiated with a zero-bit scheme that is not undetectable. We prove analogues of Theorem 5.4 and Theorem 6.4 with one very important difference (Section 7.1). For undetectable schemes, our constructions are R_k robust for $k = O(L\lambda)$, where k is the number of (modified) blocks of model-generated text needed to extract the watermark. Without undetectability, we require $k = \Omega(B)$, where B is the *total* of blocks of model-generated text that the adversary *ever observed*. Meaning that to provably extract the watermark, the adversary's output must contain essentially all the text produced by the model!

Still, the adversary has substantial freedom to modify the watermarked text without destroying the mark. First, the adversary can reorder blocks arbitrarily and can include extraneous unmarked text. Second, the adversary can modify the blocks as allowed by the underlying scheme's robustness guarantee ($\doteq$, e.g. bounded edit distance).

In Section 7.2, we discuss some possible approaches for improving the robustness parameter k without undetectability. In practice, we believe that full undetectability may not be necessary for meaningful security. Schemes that are not undetectable still offer some guarantees (e.g., bounded Renyi divergence [ZALW23] or undetectability for a single query [KGW+23a]). In applications where these not-undetectable zero-bit watermarking schemes are considered secure enough, we suspect our schemes would be too.

The theorems we prove in this section apply to *adaptively* robust watermarking schemes that are not undetectable. Unfortunately, we do not know if any such schemes exist. Prior works only consider non-adaptive robustness, and we use undetectability to show that non-adaptive robustness implies adaptive robustness (Lemmas 4.6, B.6). Without undetectability, it does not seem possible to show that our construction is adaptively robust when the underlying scheme is only non-adaptively robust.

7.1 Robustness for $k = \Omega(B)$

We need to prove an analogue of Lemma 5.3 without undetectability. As before, our theorems follow as corollaries.

Undetectability is only used in the third hybrid of the proof of Lemma 5.3. The other two hybrids use only soundness and robustness. Hybrid 3 uses undetectability to argue that which bits of the watermark are embedded in which blocks of text is (computationally) independent from adversary's view. The result is that any (modified) block in the adversary's output $\hat{T}$ embeds a uniformly random bit of the message $m \in \{0,1\}^L$. By a balls-in-bins argument, $k = L(\ln L + \lambda)$ blocks suffice to recover the whole message.

Without undetectability, this argument breaks down. An adversary who can perfectly distinguish blocks marked using the different keys can choose which message index i any (modified) block encodes, as each

index corresponds to a distinct pair of keys. In the worst case, an adversary can create $\hat{T}$ using k blocks that all correspond to the same index i, and $\mathsf{Extract}$ would recover just one bit of the message.

To get around this, we need to require that k depends on the *total number of blocks* of model-generated text seen by the adversary $\mathcal{A}$:

$$B := \sum_i |\mathsf{Blocks}(T_i; Q_i)|,$$

where $(Q_i)_i$ are $\mathcal{A}$'s queries and $T_i \leftarrow \mathsf{Wat}_{\mathsf{sk}}(Q_i)$ are the corresponding generations. It is easy to see that if $\hat{T}$ included all B blocks, then the adversary's hands are tied. As long as $B \geq L(\ln L + \lambda)$, all message bits will be extracted from $\hat{T}$ with high probability.

The index i of the message that each block encodes is sampled uniformly at random by Wat. The adversary is required to output k of these. We can slightly generalize the above argument to allow δL erasures. Let $s(B, L, \delta) < \delta B$ be any high-probability upper bound on the total load of the $\lfloor \delta L \rfloor$ smallest bins after throwing B balls uniformly at random into L bins.[11] (Wat throwing the balls, not $\mathcal{A}$.) Then if $B \geq L(\ln L + \lambda)$ and $k \geq B - s(B, L, \delta)$, the $\mathsf{Extract}$ algorithm recovers at least $(1-\delta)L$ bits of the watermark.

We now state the analogue of Lemma 5.3. Let

$$R_{\mathsf{DET}}\left(\lambda, (Q_i)_i, (T_i)_i, \hat{T}\right) = \mathbb{1}\big(B \geq L(\ln L + \lambda)\big) \wedge \ \mathbb{1}\big(\mathsf{NumBlocks}(\hat{T}; Q_i, T_i) \geq B - s(B, L, \delta)\big).$$

Lemma 7.1 ($\mathcal{W}$ recovers lossy descendants – not undetectable)**.** *Let $\lambda, L \in \mathbb{N}$, $0 \leq \delta < 1$ and $M \subseteq \{0,1\}^L$. Suppose $\mathcal{W}'$ is a block-by-block, zero-bit watermarking scheme that is sound and R_1 robustly-detectable. Let $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ be the construction from Figure 2 using $\mathcal{W}'$.*

Then for all efficient $\mathcal{A}$, the following event `FAIL` *occurs with negligible probability:*

- $\forall i, m_i \in M$, *AND* `// only queried messages in M`
- $R_{\mathsf{DET}}\left(\lambda, (Q_i)_i, (T_i)_i, \hat{T}\right) = 1$, *AND* `// robustness condition passes`
- $\hat{m} \notin F_\delta(M)$ `// extracted message unrelated to M`

in the probability experiment defined by

- $\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)$
- $\hat{T} \leftarrow \mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot,\cdot)}(1^\lambda)$, *denoting by $(m_i, Q_i)_i$ and $(T_i)_i$ the sequence of inputs and outputs of the oracle*
- $\hat{m} \leftarrow \mathsf{Extract}_{\mathsf{sk}}(\hat{T})$.

Proof outline. The proof exactly follows the proof Lemma 5.3 for Hybrids 1 and 2. Hybrid 3 is omitted. We recover a message index i whenever a (modified) block in $\hat{T}$ was generated using $\mathsf{Wat}_{k_{i,b}}$ for some b. Because $B \geq L(\ln L + \lambda)$ and the definition of s, any set of $B - s(B, L, \delta)$ blocks were generated using a set of at least $L - \lfloor \delta L \rfloor$ distinct indices i, with high probability. □

Theorem 7.2 ($\mathcal{W}$ is robust, when not undetectable)**.** *Suppose $\mathcal{W}'$ is a block-by-block, sound zero-bit watermarking scheme. Then, the $\mathcal{W}$ construction from Figure 2 is an L-bit embedding scheme that is $(\delta, R_{\mathsf{DET}})$-robust.*

Proof. This follows as an immediate consequence of Lemma 7.1 following the same reasoning used in Theorem 5.4. □

Theorem 7.3 ($\mathcal{W}$ is robust, when not undetectable)**.** *Let $n, c > 1$ be integers and $0 \leq \delta < 1$. Let $\mathcal{W}$ be the L-bit embedding scheme from Figure 2, built out of a block-by-block, sound zero-bit watermarking scheme. Furthermore let FP be a robust fingerprinting code of length L with parameters (λ, n, c, δ). Then, the $\mathcal{W}$ construction from Figure 4 using $\mathcal{W}$ from Theorem 7.2 and FP is a consistent, sound, (c, R_{DET})-robust multi-user watermarking scheme.*

Proof. This follows as an immediate consequence of Lemma 7.1 following the same reasoning used in Theorem 6.4. □

[11] E.g., for $B \geq L(\ln L + \lambda)$, $s(B, L, \delta) \geq \lfloor \delta L \rfloor$, as every bin has has load at least 1 with high probability.

7.2 Can we do better?

We briefly describe two approaches to reducing the robustness parameter k in the absence of undetectability.

Bounded or partial undetectability Our analysis allowed for a worst-case adversary who could perfectly distinguish blocks marked under different keys. But even watermarking schemes that are not undetectable are not so blatantly detectable. For example, the scheme of [KTHL23] is undetectable for any single query ("distortion-free"), and the red-/green-list scheme of [ZALW23] guarantees that the Renyi divergence between the marked and unmarked distributions for any single token is bounded.

It's not clear how to use these limited distinguishing guarantees to build robust L-bit / multi-user watermarking schemes. The critical step in the proof of Lemma 5.3 is to bound the fraction δ of empty bins after k balls are thrown into L bins. With undetectability, the balls are thrown uniformly. One idea is to bound the distinguishing advantage of an adversary making q queries with c marks and producing B blocks of text as only growing polynomially in q, c, or B. Then, use that result to conclude that the induced distribution of balls-into-bins is not too far from uniform. The Renyi divergence bounds in [ZALW23] do not seem strong enough to make this approach work, even for a single generation. Even if this idea worked, proving adaptive robustness would still be challenging.

Heuristically duplicating keys Without undetectability, the adversary may be able to tell whenever the same key is used to watermark a piece of text. This allows (in our construction) the adversary to only include blocks watermarked under a few keys. In our proof, this is prevented by undetectability. Our main results ($k = O(L\lambda)$) should hold so long as the adversary never sees two blocks generated using the same key.

Towards that end, one could try key duplication. Generate poly-many keys (instead of just one) for each index-bit (i, b) pair, sampling a random key from this set at every iteration of Wat. This will reduce the number of key collisions observed by the adversary. Though it would not make the probability of collision negligible, it would be possible to bound the number of collisions as a function of the number of blocks B observed. Combined with the previous approach, this may suffice. Even if not, key duplication may improve practical security for schemes that are particularly detectable. However, key duplication comes with a proportional increase in the runtime of our detection algorithm, which checks every possible key.

Acknowledgements

Aloni Cohen and Gabe Schoenbach were supported in part by the DARPA SIEVE program under Agreement No. HR00112020021. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

[Aar22] Scott Aaronson. My AI safety lecture for ut effective altruism, November 2022.

[BG24] Dan Geer Bob Gleichauf. Digital watermarks are not ready for large language models. *Lawfare*, 2024.

[BKM10] Dan Boneh, Aggelos Kiayias, and Hart William Montgomery. Robust fingerprinting codes: a near optimal construction. In *Proceedings of the Tenth Annual ACM Workshop on Digital Rights Management*, DRM '10, page 3–12, New York, NY, USA, 2010. Association for Computing Machinery.

[BS98] D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.

[Cac00] Christian Cachin. An information-theoretic model for steganography. Cryptology ePrint Archive, Report 2000/028, 2000. https://eprint.iacr.org/2000/028.

[CG24] Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. Cryptology ePrint Archive, Paper 2024/235, 2024. https://eprint.iacr.org/2024/235.

[CGZ23] Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. Cryptology ePrint Archive, Paper 2023/763, 2023. https://eprint.iacr.org/2023/763.

[CLW+24] Zhongze Cai, Shang Liu, Hanzhao Wang, Huaiyang Zhong, and Xiaocheng Li. Towards better statistical understanding of watermarking llms, 2024.

[FGJ+23] Jaiden Fairoze, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Mingyuan Wang. Publicly detectable watermarking for language models. Cryptology ePrint Archive, Paper 2023/1661, 2023. https://eprint.iacr.org/2023/1661.

[HD17] Jamie Hayes and George Danezis. Generating steganographic images via adversarial training. *Advances in neural information processing systems*, 30, 2017.

[Hop04] Nicholas J. Hopper. Toward a theory of steganography. Technical report, 2004.

[Hou23a] White House. Blueprint for an AI Bill of Rights. *Office of Science and Technology Policy*, 2023.

[Hou23b] White House. Fact sheet: Biden-harris administration secures voluntary commitments from leading artificial intelligence companies to manage the risks posed by ai. *Statements and Releases*, 2023.

[Hou23c] White House. Fact sheet: President biden issues executive order on safe, secure, and trustworthy artificial intelligence. *Statements and Releases*, 2023.

[JGHG24] Zhengyuan Jiang, Moyang Guo, Yuepeng Hu, and Neil Zhenqiang Gong. Watermark-based detection and attribution of ai-generated content, 2024.

[KGW+23a] 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, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR, 23–29 Jul 2023.

[KGW+23b] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. On the reliability of watermarks for large language models. *arXiv preprint arXiv:2306.04634*, 2023.

[KJGR21] Gabriel Kaptchuk, Tushar M. Jois, Matthew Green, and Aviel D. Rubin. Meteor: Cryptographically secure steganography for realistic distributions. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021: 28th Conference on Computer and Communications Security*, pages 1529–1548, Virtual Event, Republic of Korea, November 15–19, 2021. ACM Press.

[KTHL23] Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models, 2023.

[LRW+24] Xiang Li, Feng Ruan, Huiyuan Wang, Qi Long, and Weijie J. Su. A statistical framework of watermarks for large language models: Pivot, detection efficiency and optimal rules, 2024.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[NFH+07] Koji Nuida, Satoshi Fujitsu, Manabu Hagiwara, Takashi Kitagawa, Hajime Watanabe, Kazuto Ogawa, and Hideki Imai. An improvement of tardos's collusion-secure fingerprinting codes with very short lengths. In Serdar Boztaş and Hsiao-Feng (Francis) Lu, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 80–89, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[NJA24] Alexander Nemecek, Yuzhou Jiang, and Erman Ayday. Topic-based watermarks for llm-generated text, 2024.

[PHZS24] Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. Attacking llm watermarks by exploiting their strengths, 2024.

[QYH+24] Wenjie Qu, Dong Yin, Zixin He, Wei Zou, Tianyang Tao, Jinyuan Jia, and Jiaheng Zhang. Provably robust multi-bit watermarking for ai-generated text via error correction code, 2024.

[SG23] Pushmeet Kohli Sven Gowal. Identifying ai-generated images with synthid. *Google Deepmind*, 2023.

[Sri24] Siddarth Srinivasan. Detecting AI fingerprints: A guide to watermarking and beyond. *Brookings*, 2024.

[Tar08] Gábor Tardos. Optimal probabilistic fingerprint codes. *J. ACM*, 55(2), may 2008.

[WKGG24] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. *Advances in Neural Information Processing Systems*, 36, 2024.

[XYL24] Xiaojun Xu, Yuanshun Yao, and Yang Liu. Learning to watermark llm-generated text via reinforcement learning, 2024.

[ZALW23] Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text, 2023.

[ZEF+23] Hanlin Zhang, Benjamin L. Edelman, Danilo Francati, Daniele Venturi, Giuseppe Ateniese, and Boaz Barak. Watermarks in the sand: Impossibility of strong watermarking for generative models, 2023.

[ZKJFF18] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 657–672, 2018.

A Implications for existing LLM watermarking schemes

As we show next, existing watermarking schemes can be viewed as block-by-block schemes. The only meaningful restriction imposed by Definition 4.2 is on the syntax of the robustness condition R. Notice that one could in some trivial sense view all complete schemes as block-by-block by considering an entire generation a block. This triviality is unhelpful for our black-box constructions and unnecessary for most schemes, but illustrates how broad our framework is in general.

Notation For a string $T = \tau_1\tau_2\ldots\tau_{|T|}$, we define $T_{i:j} := \tau_i\ldots\tau_j$, $T_{\leq k} := \tau_{1:k}$. We write $\tau \in T$ to mean that τ is a *substring* of T, i.e. $\tau = T_{i:j}$ for some i, j.

A.1 Intuition for how existing schemes work

We give a brief intuition for the two dominant approaches to watermarking language models in prior work. Not all schemes fit into these categories, including [FGJ+23] discussed below.

Derandomizing and measuring correlation One class of schemes work by derandomizing the language model using the secret key and then detecting the effects of this derandomization [CGZ23, Aar22, KTHL23]. Because a probability distribution can be derandomized without being noticeably altered, these schemes enjoy some level of undetectability.

At a very high level, the derandomization schemes work as follows. Language models generate text token-by-token. Let Q be a prompt and $T_{<i}$ be an already-generated prefix. In the unmarked model, the next token τ_i is sampled according to some distribution $p_Q(\cdot|Q\|T_{<i})$. The marked model is the same, except that a secret sequence $\sigma_1\sigma_2\ldots\sigma_\ell$ of (pseudo-)random bits is used to derandomize the sampling of τ_i in a way that induces a correlation between σ_i and τ_i. We discuss how particular schemes derandomize p_Q in Appendices A.3.1 and A.5. Each of these schemes differ in how the secret sequence is derived, how it is used to derandomize the next token, and how the induced correlations are measured and used to detect.

Red/green lists Another class of statistical watermarking schemes bias the sampling of tokens using using so-called red and green lists, determined using a hash function [KGW+23a, ZALW23]. Tokens in the green list are sampled more often compared to the unmarked language model, and tokens in the red list are sampled less often. Depending on the scheme, these red and green lists can be solely determined by a secret key [ZALW23] or also nearby tokens [KGW+23a].

To detect whether a watermark is present within text $\hat{T}$, one can check the proportion of green list tokens in $\hat{T}$. If the green list contains half the tokens sampled uniformly at random, say, then unmarked text should have close to 50% green tokens. Marked text will have many more green tokens. Detection works by testing whether the proportion of green tokens in any substring is above a statistically significant threshold. Soundness and robustness are proved using concentration bounds on the expected number of green tokens in unmarked and marked text, respectively. However, these schemes are not undetectable. By design, green tokens are noticeably more likely in the watermarked model. With poly-many queries, an adversary could conceivably reconstruct the lists in full, though this seems very costly in practice.

A.2 Undetectable watermarks [CGZ23]

The zero-bit watermarking scheme of Christ, Gunn and Zamir [CGZ23] is easily cast as a block-by-block scheme, with all the provable properties needed to invoke our constructions: undetectability, soundness, completeness, and robustness. The robustness guarantee is called $b(\ell)$-*substring completeness*. The construction $\mathcal{W}$ from [CGZ23, Algorithms 5-6] is a $\left(\frac{8}{\ln 2}\lambda\sqrt{\ell}\right)$-substring complete watermarking scheme [CGZ23, Theorem 8]. Robustness is guaranteed to hold for generations with enough empirical entropy, with the amount required depending on the length of the generation.

Definition A.1 (Empirical entropy [CGZ23])**.** *For strings $\tau, Q \in \mathcal{T}^*$ and model* Model*, we define the* empirical entropy of τ with respect to Model and Q *as* $H_e(\tau; Q) := -\log \Pr[\mathsf{Model}(Q)_{\leq|\tau|} = \tau]$.

Definition A.2 (Substring completeness [CGZ23])**.** *A watermarking scheme $\mathcal{W}$ is* $b(\ell)$-substring complete *if for every prompt Q and security parameter λ:*

$$\Pr_{\substack{\mathsf{sk}\leftarrow\mathsf{KeyGen}(1^\lambda)\\ T\leftarrow\mathsf{Wat}_{\mathsf{sk}}(Q)}}\left[\exists \textit{ length-}\ell \textit{ substring } \tau \in T \;:\; \underbrace{H_e(\tau;Q) \geq b(\ell)}_{\textit{enough entropy}} \textit{ and } \underbrace{\mathsf{Detect}_{\mathsf{sk}}(\tau) = 0}_{\textit{detection fails}}\right] < \mathsf{negl}(\lambda).$$

To detect a watermark on input $\hat{T}$, the construction $\mathsf{Detect}_{\mathsf{sk}}(\hat{T})$ outputs the OR of $\mathsf{Detect}_{\mathsf{sk}}(\hat{\tau})$ for all substrings $\hat{\tau} \in \hat{T}$. By substring completeness, $\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1$ if there exists a substring $\tau \in T$ that satisfies the following two conditions:

(i) $H_e(\tau; Q) \geq b(|\tau|)$.

(ii) There exists a substring $\hat{\tau} \in \hat{T}$ for which $\hat{\tau} = \tau$.

The watermarking scheme $\mathcal{W}$ described above is naturally viewed as a block-by-block scheme. Condition (i) defines the blocks:$\mathsf{block}_{\mathrm{CGZ}}(\tau; Q) = \mathbb{1}\big(H_e(\tau; Q) \geq b(|\tau|)\big)$. Condition (ii) tells us that the binary relation $\hat{=}$ on strings τ and $\hat{\tau}$ is string equality. Let R_1^{CGZ} be the AEB-robustness condition induced by the function $\mathsf{block}_{\mathrm{CGZ}}$ and the string equality relation, according to Definition 4.5. The following claim implies that $\mathcal{W}$ is a block-by-block scheme.

Claim A.3. *If $\mathcal{W}$ is an undetectable, $b(\ell)$-substring complete watermarking scheme with* Detect *as above, then $\mathcal{W}$ is adaptively R_1^{CGZ}-robust.*

Proof. Fix a prompt $Q \in \{0,1\}^*$ of length $|Q| \leq \mathrm{poly}(\lambda)$ and efficient adversary $\mathcal{A}$. To show that $\mathcal{W}$ is non-adaptively R_1^{CGZ}-robust, it suffices to show the following:

$$\Pr\Big[\ \mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1 \ \mid\ R_1^{\mathsf{CGZ}}(\lambda, Q, T, \hat{T}) = 1\Big] \geq 1 - \mathsf{negl}(\lambda),$$

where $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $T \leftarrow \mathsf{Wat}_{\mathsf{sk}}(Q)$, and $\hat{T} \leftarrow \mathcal{A}(1^\lambda, T)$.

By definition of R_1^{CGZ}, there exist substrings $\tau \in T, \hat{\tau} \in \hat{T}$ such that (i) $\mathsf{block}_{\mathsf{CGZ}}(\tau; Q) = 1$, and (ii) $\hat{\tau} = \tau$. Because $\mathcal{W}$ is $b(\ell)$-substring complete $\mathsf{Detect}_{\mathsf{sk}}(\hat{\tau}) = \mathsf{Detect}_{\mathsf{sk}}(\tau) = 1$ with high probability. By construction, $\mathsf{Detect}(\hat{T}) = 1$ with high probability, so $\mathcal{W}$ is non-adaptively R_1^{CGZ}-robust. Since $\mathcal{W}$ is undetectable, applying Lemma 4.6 completes the proof. □

A.3 Watermarking from pseudorandom codes [CG24]

The watermarking schemes of Christ and Gunn [CG24] can also be cast as block-by-block schemes. They define pseudorandom error-correcting codes (PRCs), and use PRCs to derandomize the language model. A zero-bit PRC is a triple of randomized, efficient algorithms $\mathsf{PRC} = (\mathsf{KeyGen}, \mathsf{Encode}, \mathsf{Decode})$. KeyGen generates a secret key sk. $\mathsf{Encode}_{\mathsf{sk}}(1)$ generates codewords x of length n that are pseudorandom to any efficient adversary without sk. $\mathsf{Decode}_{\mathsf{sk}}(T)$ attempts to decode a string T. The PRC is *robust against a channel $\mathcal{E}$* if

(a) For any fixed T independent of sk, $\mathsf{Decode}_{\mathsf{sk}}(T) = \perp$ with high probability over sk.

(b) $\mathsf{Decode}_{\mathsf{sk}}(\mathcal{E}(\mathsf{Encode}_{\mathsf{sk}}(1))) = 1$ with high probability over $\mathcal{E}$.

Christ and Gunn construct PRCs that are robust to p-bounded channels, assuming either $2^{O(\sqrt{n})}$-hardness of Learning Parity with Noise (LPN) or polynomial hardness of LPN and the planted XOR problem at low density.

Definition A.4 (p-Bounded channels)**.** *For any $p \geq 0$, a length-preserving channel $\mathcal{E} : \Sigma^* \to \Sigma^*$ is p-*bounded *if there exists a negligible function* negl *such that for all $n \in \mathbb{N}$, $\Pr_{x \leftarrow \{0,1\}^n}[\Delta(\mathcal{E}(x), x) > p] < \mathsf{negl}(n)$, where Δ is the normalized Hamming distance.*

The watermarking robustness guarantee of [CG24, Definition 13] is called *substring robustness against a channel $\mathcal{E}$*. Informally, substring robustness guarantees that the watermark will be detected even from a (sufficiently entropic) cropped string that has been corrupted by $\mathcal{E}$. Substring robustness generalizes substring completeness (Definition A.2): a scheme that is substring robust against the identity channel $\mathcal{I}(T) = T$ is substring complete. In the next section, we will show that the zero-bit watermarking scheme $\mathcal{W}[\mathsf{PRC}] = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ from [CG24, Construction 7] is a block-by-block scheme, when instantiated with a zero-bit pseudorandom code PRC that is robust to certain p-bounded channels. We first state two robustness results in the language of [CG24]:

Lemma A.5 (Lemma 22, [CG24])**.** *Let $\varepsilon > 0$ be any constant. If* PRC *is a zero-bit PRC with block length n that is robust to any $(1/2 - \varepsilon)$-bounded channel, then $\mathcal{W}[\mathsf{PRC}]$ is $(4\sqrt{\varepsilon} \cdot L + 2\sqrt{2} \cdot n)$-substring complete.*

Lemma A.6 (Lemma 23, [CG24])**.** *Let $\varepsilon, \delta > 0$ be any constants. If* PRC *is a zero-bit PRC with block length n that is robust to any $(1/2 - \varepsilon \cdot \delta)$-bounded channel, then $\mathcal{W}[\mathsf{PRC}]$ is $(4\sqrt{\varepsilon} \cdot L + 2\sqrt{2} \cdot n)$-substring robust against $\mathsf{BSC}_{1/2-\delta}$, the binary symmetric channel with error rate $1/2 - \delta$.*

A.3.1 R_1^{CG}-robustness of $\mathcal{W}[\mathsf{PRC}]$

We begin by describing how $\mathcal{W}[\mathsf{PRC}]$ embeds and detects watermarks. To generate watermarked text T, $\mathsf{Wat}_{\mathsf{sk}}$ samples a codeword $x_1 \leftarrow \mathsf{PRC.Encode}_{\mathsf{sk}}(1)$ and uses x_1 to sample the first length-n substring $\tau_1 \in T$. Crucially, τ_1 will be a noised version of x_1, where the amount of noise is inversely proportional to the empirical entropy of τ_1. This procedure is done iteratively, so the final output is $T = \tau_1 \dots \tau_r \sigma$ where each $\tau_i \approx x_i$. To detect, $\mathsf{Detect}_{\mathsf{sk}}(T)$ outputs the OR of $\mathsf{PRC.Decode}_{\mathsf{sk}}(\tau)$ for all substrings $\tau \in T$. By the robustness of PRC, for all $i \in [r]$, $\mathsf{Decode}_{\mathsf{sk}}(\tau_i) = 1$ with high probability.

Lemma A.5 establishes that $\mathcal{W}[\mathsf{PRC}]$ is $b(\ell)$-substring complete so long as PRC is sufficiently robust. As in Section A.2, this means that $\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1$ with high probability if there exists a substring τ that satisfies the same conditions (i) and (ii) from Section A.2. Hence Claim A.3 implies that $\mathcal{W}[\mathsf{PRC}]$ is non-adaptively R_1^{CGZ}-robust.

Lemma A.6 establishes that $\mathcal{W}[\mathsf{PRC}]$ is $b(\ell)$-substring robust provided the underlying PRC is robust to even noisier channels. To translate this guarantee into our language, we can keep condition (i) exactly the same as before. However, unlike requiring some substring $\hat{\tau} \in \hat{T}$ to be *identical* to $\tau \in T$ as in condition (ii), we want to relax our binary string relation, putting strings $\hat{\tau}$ and τ in relation only if every length-n substring $\tau^* \in \tau$ is no more than $(\varepsilon \cdot (1-\delta))$-far from some length-$n$ substring $\hat{\tau}^* \in \hat{\tau}$, in the normalized Hamming distance Δ. This is expressed by our (asymmetric) relation $\doteq_{\mathsf{CG}}$:

$$\hat{\tau} \doteq_{\mathsf{CG}} \tau \iff \forall \text{ length-}n \text{ substrings } \tau^* \in \tau, \exists \text{ length-}n \text{ substring } \hat{\tau}^* \in \hat{\tau} \ : \ \Delta(\hat{\tau}^*, \tau^*) \le \varepsilon \cdot (1-\delta).$$

Now let R_1^{CG} be the AEB-robustness condition induced by the function $\mathsf{block}_{\mathsf{CG}}$ (defined like $\mathsf{block}_{\mathsf{CGZ}}$ but for a different entropy requirement $b(\ell) = 4\sqrt{\varepsilon} \cdot \ell + 2\sqrt{2} \cdot n$) and the relation $\doteq_{\mathsf{CG}}$, according to Definition 4.5. The following claim implies that $\mathcal{W}[\mathsf{PRC}]$ is a block-by-block scheme when instantiated from a pseudorandom code PRC with block length n that is robust to every $(1/2 - \varepsilon \cdot \delta)$-bounded channel.

Claim A.7. *Let $\varepsilon, \delta > 0$ be constants. Assuming either $2^{O(\sqrt{n})}$-hardness of LPN or polynomial hardness of LPN and the planted XOR problem at low density, if PRC is a zero-bit PRC of block length n that is robust to every $(1/2 - \varepsilon \cdot \delta)$-bounded channel, then $\mathcal{W}[\mathsf{PRC}]$ is adaptively R_1^{CG}-robust.*

Proof. Fix a prompt $Q \in \{0,1\}^*$ of length $|Q| \le \mathrm{poly}(\lambda)$ and efficient adversary $\mathcal{A}$. To show that $\mathcal{W}[\mathsf{PRC}]$ is non-adaptively R_1^{CG}-robust, it suffices to show the following:

$$\Pr\left[\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1 \ \mid \ R_1^{\mathsf{CG}}(\lambda, Q, T, \hat{T}) = 1 \right] \ge 1 - \mathsf{negl}(\lambda).$$

where $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $T \leftarrow \mathsf{Wat}_{\mathsf{sk}}(Q)$, and $\hat{T} \leftarrow \mathcal{A}(1^\lambda, T)$. As discussed above, the watermarking scheme noisily embeds codewords x_i into T, where each $x_i \leftarrow \mathsf{PRC.Encode}_{\mathsf{sk}}(1)$. We are conditioning on $R_1^{\mathsf{CG}}(\lambda, Q, T, \hat{T}) = 1$. By definition of R_1^{CG}, there exist substrings $\hat{\tau} \in \hat{T}, \tau \in T$ such that (i) $\mathsf{block}_{\mathsf{CG}}(\tau; Q) = 1$ and (ii) $\hat{\tau} \doteq_{\mathsf{CG}} \tau$. By (i) there exists some $x \in (x_i)_i$ and some length-n substring $\tau^* \in \tau$ such that $\Delta(\tau^*, x) \le 1/2 - \varepsilon$ with high probability.[12] By (ii) and the definition of $\doteq_{\mathsf{CG}}$ there exists some length-n substring $\hat{\tau}^* \in \hat{\tau}$ such that $\Delta(\hat{\tau}^*, \tau^*) \le \varepsilon \cdot (1-\delta)$. Using the triangle inequality, (i) and (ii) together give us $\Delta(\hat{\tau}^*, x) \le \frac{1}{2} - \varepsilon \cdot \delta$ with high probability. Since PRC is robust to every $(\frac{1}{2} - \varepsilon \cdot \delta)$-bounded channel, $\mathsf{PRC.Decode}_{\mathsf{sk}}(\hat{\tau}^*)$ outputs 1 with high probability. By construction $\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1$ with high probability, so $\mathcal{W}[\mathsf{PRC}]$ is non-adaptively R_1^{CG}-robust. Assuming $2^{O(\sqrt{n})}$-hardness of LPN or polynomial hardness of LPN and the planted XOR problem at low density, and applying Lemma 4.6 completes the proof. □

Christ and Gunn also build L-bit watermarking schemes, using L-bit PRCs that whose encoding functions take messages $m \in \{0,1\}^L$ rather than the single message $m \in \{1\}$. Lemma A.6 applies identically for an L-bit watermarking scheme, so all of the robustness results described in this section extend to L-bit watermarking schemes.

[12] Condition (i) implies a lower bound on the empirical entropy of τ. We can then use the reasoning in the proof of [CG24, Lemma 22] to show the existence of some substring $\tau^* \in \tau$ of length n with enough empirical entropy to apply [CG24, Lemma 21], which provides the desired high-probability guarantee on the Δ-distance between τ^* and x.

A.4 Publicly detectable watermarks [FGJ+23]

The zero-bit watermarking scheme of Fairoze, Garg, Jha, Mahloujifar, Mahmoody, and Wang [FGJ+23] can also cast as a block-by-block scheme. Unlike other schemes, this one is publicly detectable. The Detect algorithm only requires a public key, and knowledge of the public key does not undermine undetectability, soundness, completeness, nor robustness. When used in our constructions, the resulting L-bit and multi-user watermarking schemes are also publicly detectable.

Unlike [CGZ23], the proofs of undetectability and computational efficiency in [FGJ+23] require assuming that each block of $\ell = \ell(\lambda)$ tokens produced by Model has at least λ bits of min-entropy. The parameter ℓ is assumed to be known and is used in the construction.

Assumption A.8 (Assumption 2.1 of [FGJ+23])**.** *For any prompt Q and string T, $\Pr[\mathsf{Model}(Q)_{1:\ell} = T] \leq 2^{-\lambda}$.*

Definition A.9 (d-Robustness, adapted from Definition 4.5 of [FGJ+23])**.** *A publicly-detectable watermarking scheme is d-robust if for every prompt Q, security parameter λ, and PPT $\mathcal{A}$,*

$$\Pr\left[\mathsf{Detect}_{\mathsf{pk}}(\hat{T}) = 0 \;:\; \begin{array}{c}(\mathsf{sk},\mathsf{pk}) \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda) \\ T \leftarrow \mathsf{Wat}_{\mathsf{sk}}(Q) \\ \hat{T} \leftarrow \mathcal{A}(\mathsf{pk}, T)\end{array}\right] \leq \mathsf{negl}(\lambda)$$

where $\mathcal{A}$ is required to output $\hat{T}$ that contains a substring $\hat{\tau}$ of length at least d that is also a substring of T.

Using a signature scheme with pseudorandom ℓ_{sig}-bit signatures, [FGJ+23] gives a $(2\ell(1+\ell_{\mathrm{sig}}))$-robust watermarking scheme, where ℓ is from the min-entropy assumption above. To turn this into a block-by-block scheme, we take $\mathsf{block}_{\mathrm{FGJ+}}(\tau; Q) = \mathbb{1}(|\tau| \geq d)$, and the binary relation $\eqsim$ on strings $\hat{\tau}$ and τ to be string equality. Let $R_1^{\mathsf{FGJ+}}$ be the AEB-robustness condition induced by the function $\mathsf{block}_{\mathrm{FGJ+}}$ and the string equality relation, according to Definition 4.2. (Note the similarity to Sec A.2.) The following claim implies that the scheme of [FGJ+23] is a block-by-block scheme, with $d = 2\ell(1+\ell_{\mathrm{sig}})$.

Claim A.10. *Under Assumption A.8, if $\mathcal{W}$ is an undetectable, d-robust scheme then it is an adaptively $R_1^{\mathsf{FGJ+}}$-robust scheme.*

Proof. Fix a prompt $Q \in \mathcal{T}^*$ of length $|Q| \leq \mathrm{poly}(\lambda)$ and a PPT adversary $\mathcal{A}$. To show that $\mathcal{W}$ is non-adaptively $R_1^{\mathsf{FGJ+}}$-robust, it suffices to show the following:

$$\Pr\left[\ \mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1 \ \mid\ R_1^{\mathsf{FGJ+}}(\lambda, Q, T, \hat{T}) = 1\right] \geq 1 - \mathsf{negl}(\lambda).$$

where $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $T \leftarrow \mathsf{Wat}_{\mathsf{sk}}(Q)$, and $\hat{T} \leftarrow \mathcal{A}(1^\lambda, T)$.

By definition of $R_1^{\mathsf{FGJ+}}$, there exist substrings $\tau \in T, \hat{\tau} \in \hat{T}$ such that (i) $\mathsf{block}_{\mathrm{FGJ+}}(\tau; Q) = 1$, and (ii) $\hat{\tau} = \tau$. Using Assumption A.8 and the fact that $\mathcal{W}$ is d-robust, $\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 1$ with high probability, so $\mathcal{W}$ is non-adaptively $R_1^{\mathsf{FGJ+}}$-robust. Since $\mathcal{W}$ is undetectable, applying Lemma 4.6 completes the proof. □

A.5 A p-value for watermarks [KTHL23]

Slight variations of the zero-bit watermarking schemes of Kuditipudi, Thickstun, Hashimoto, and Liang [KTHL23] can be cast as block-by-block schemes. In particular, the Inverse Transform Sampling (ITS) scheme from [KTHL23, Section 2.3] can be modified to be complete, sound, and robust to token substitutions.[13] But it is not undetectable, so our main constructions cannot be applied to it. More specifically, the "distortion-free" property achieved by the ITS scheme is essentially a single-query form of undetectability. Across many queries, however, the outputs will be highly correlated, unlike truly undetectable schemes.

[13] It is unclear how to modify the ITS scheme to be robust to token insertions or deletions, because the k^k term in the statement of [KTHL23, Lemma 2.6] dominates the inverse exponential term.

At a high level, the ITS scheme $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ works as follows. As sketched in Appendix A.1, $\mathsf{Wat}_{\mathsf{sk}}(Q)$ outputs a generation T by using a decoder function $\Gamma : \mathcal{K} \times \Delta(\mathcal{T}) \to \mathcal{T}$ to select successive tokens. The decoder Γ uses an element of sk and the distribution p_Q to deterministically select a token. Detection relies on a test statistic $\phi : \mathcal{T}^* \times \mathcal{K}^* \to \mathbb{R}$ that is designed so $\phi(T, \mathsf{sk})$ is small whenever $T \leftarrow \mathsf{Wat}_{\mathsf{sk}}(Q)$. Instead of outputting a bit, $\mathsf{Detect}_{\mathsf{sk}}(\hat{T})$ samples s secret keys sk' independently and computes $\hat{p} = (K+1)/(s+1)$, where K is the number of times $\phi(\hat{T}, \mathsf{sk}') \leq \phi(\hat{T}, \mathsf{sk})$. The fraction $\hat{p}$ is a p-value relative to the null hypothesis that $\hat{T}$ is not watermarked.

Our definitions require that detection only errs with negligible probability. Running $\mathsf{Detect}_{\mathsf{sk}}$ as described above would require exponential runtime to produce negligibly small p-values. Instead, we can simplify $\mathcal{W}$ by detecting watermarks only when the test statistic $\phi(\hat{T}, \mathsf{sk})$ is below some threshold ρ:

$$\widetilde{\mathsf{Detect}}_{\mathsf{sk}}(\hat{T}) = \mathbb{1}\left(\phi(\hat{T}, \mathsf{sk}) < \rho = -\sqrt{\frac{|\hat{T}|\lambda}{8}}\right). \tag{3}$$

To achieve robustness, $\phi(\hat{T}, \mathsf{sk})$ is constructed to be small so long as $\hat{T}$ is "close enough" to some watermarked text T. This is done by using an "alignment cost" function $d : \mathcal{T}^* \times \mathcal{K}^*$, where ϕ returns the minimum cost over all alignments of candidate text $\hat{T}$ with substrings $\sigma \in \mathsf{sk}$. Below we state a robustness guarantee for the ITS scheme in the language of [KTHL23]. The guarantee will depend on the observed token probabilities of verbatim outputs of Wat, which is called the *watermark potential*.

Definition A.11 (Watermark potential)**.** *Given some prompt Q, the* watermark potential $\alpha : \mathcal{T}^m \to \mathbb{R}$ *of some text $T = \tau_1 \dots \tau_m$ relative to the distribution $p = p_Q$ is*

$$\alpha(T) := \frac{1}{m}\sum_{i=1}^{m}\Big(1 - p(\tau_i \mid Q \| T_{<i})\Big).$$

Furthermore, we define $\hat{\alpha} : \mathcal{T}^m \times \mathcal{T}^m \to \mathbb{R}$ as

$$\hat{\alpha}(T, \hat{T}) = \frac{1}{m}\sum_{\{i:\tau_i = \hat{\tau}_i\}}\Big(1 - p(\tau_i \mid Q \| T_{<i})\Big) \; - \frac{1}{m}\sum_{\{i:\tau_i \neq \hat{\tau}_i\}}\left(\frac{1}{|\mathcal{T}| - 1}\right).$$

Note that for any T, $0 \leq \alpha(T) \leq \frac{|\mathcal{T}|-1}{|\mathcal{T}|}$. The robustness guarantee from [KTHL23] implies that, even if an adversary substitutes many tokens in some watermarked text T to create $\hat{T}$, the expected p-value computed in $\mathsf{Detect}_{\mathsf{sk}}(\hat{T})$ will be small, so long as the untouched tokens have sufficient watermark potential.

Lemma A.12 (Lemma 2.5, [KTHL23])**.** *Let $n, m \in \mathbb{N}$ with $n \geq m$, where m is the length of the generation and n is the length of the secret key. Use the decoder Γ from [KTHL23, Line (1)], alignment cost d from [KTHL23, Line (2)], and ϕ from [KTHL23, Algorithm 3] with $k = m$. Let $\mathsf{sk}, \mathsf{sk}' \sim \mathcal{K}^n$, with $T = \mathsf{Wat}_{\mathsf{sk}}(m, p, \Gamma)$. Let $\hat{T} \in \mathcal{T}^m$ be conditionally independent of sk and sk' given T. Then almost surely*

$$\Pr\Big[\; \phi(\hat{T}, \mathsf{sk}') \leq \phi(\hat{T}, \mathsf{sk}) \;\mid\; T, \hat{T}\Big] \leq 2n\exp(-kC_0^2\hat{\alpha}(T, \hat{T})^2/2),$$

where $C_0 = 1/12 + o_{|\mathcal{T}|}(1)$ is a constant.

To describe our modified version of $\mathcal{W}$ as a block-by-block scheme, it suffices to build a robustness condition R_1^{KTHL} that only holds when T has sufficient watermark potential and $\hat{T}$ is no more than δ-far from T in the normalized Hamming distance Δ. Let $N := |\mathcal{T}|$ be the size of the token set of the language model. For any $0 \leq \delta < 1$, we define

$$\mathsf{block}_{\mathsf{KTHL}}(T; Q) = \mathbb{1}\Big(\alpha(T) \geq \rho_\delta\Big)$$

where

$$\rho_\delta := \left(\frac{1}{C_0\sqrt{2}} \cdot \sqrt{\frac{\lambda}{|T|}} + \frac{\delta}{N-1}\right)\left(1-\delta\right) + \delta$$

and C_0 is the constant from Lemma A.12. Note that the lower bound enforced by $\mathsf{block}_{\mathsf{KTHL}}$ is only satisfiable when $|T| \in \Omega(\lambda)$. We then define $\hat{T} \eqsim_{\mathsf{KTHL}} T \iff \Delta(\hat{T}, T) \leq \delta$. Let R_1^{KTHL} be the robustness condition induced by $\mathsf{block}_{\mathsf{KTHL}}$ and $\eqsim_{\mathsf{KTHL}}$. Whenever $R_1^{\mathsf{KTHL}}(\lambda, Q, T, \hat{T}) = 1$, we can derive the following lower bound on $\hat{\alpha}(T, \hat{T})$.

Lemma A.13. *For any $\lambda \in \mathbb{N}$, $Q \in \mathcal{T}^*$, and $T, \hat{T} \in \mathcal{T}^m$ satisfying $R_1^{\mathsf{KTHL}}(\lambda, Q, T, \hat{T}) = 1$, we have*

$$\hat{\alpha}(T, \hat{T}) \geq \frac{1}{C_0\sqrt{2}} \cdot \sqrt{\frac{\lambda}{m}}.$$

Proof. Consider any set of random variables $\mathcal{X} = \{X_1, X_2, \ldots, X_m\}$, where each $X_i \in [0,1]$. Let $\mu = \frac{1}{m}\sum_{i\in[m]} X_i$ and let $\mu_{1-\delta}$ be the mean of any subset of $\mathcal{X}$ of size $\lfloor m(1-\delta)\rfloor$. Then we have

$$\mu_{1-\delta} \geq \frac{(\sum_{i\in[m]} X_i) - m\delta}{\lfloor m(1-\delta)\rfloor} \geq \frac{\mu - \delta}{1-\delta}.$$

Taking $X_i = (1 - p(\tau_i \mid Q \| T_{<i}))$ so $\mu \geq \rho_\delta$, we can lower bound $\hat{\alpha}(T, \hat{T})$ by

$$\begin{aligned}
\hat{\alpha}(T, \hat{T}) &= \frac{1}{m} \sum_{\{i:\tau_i = \hat{\tau}_i\}} \Big(1 - p(\tau_i \mid Q \| T_{<i})\Big) - \frac{1}{m} \sum_{\{i:\tau_i \neq \hat{\tau}_i\}} \Big(\frac{1}{N-1}\Big) \\
&\geq \frac{\rho_\delta - \delta}{1-\delta} - \frac{\delta}{N-1} \\
&= \frac{1}{C_0\sqrt{2}} \cdot \sqrt{\frac{\lambda}{m}},
\end{aligned}$$

where the final equality holds by plugging in our value of ρ_δ. □

The following claim implies that our modified version of the ITS watermarking scheme from [KTHL23] is a block-by-block scheme.

Claim A.14. *Let $\mathcal{W}$ be the ITS watermarking scheme from [KTHL23, Section 2.3], using the decoder Γ from [KTHL23, Line (1)], alignment cost d from [KTHL23, Line (2)], and ϕ from [KTHL23, Algorithm 3] with $k = m$. Define $\widetilde{\mathcal{W}}$ to be $\mathcal{W}$ except using the $\widetilde{\mathsf{Detect}}$ function from (3). Then $\widetilde{\mathcal{W}}$ is sound and non-adaptively R_1^{KTHL}-robust.*

Proof. We will start by showing that $\widetilde{\mathcal{W}}$ is sound. Fix a string $T \in \mathcal{T}^*$ of length $|T| = m \leq \mathrm{poly}(\lambda)$. We want to show that $\widetilde{\mathsf{Detect}}_{\mathsf{sk}}(T)$ returns 1 with negligible probability over secret keys sk of length n. Recall that $\widetilde{\mathsf{Detect}}_{\mathsf{sk}}(T)$ returns 1 if $\phi(T, \mathsf{sk}) < \rho$, where

$$\phi(T, \mathsf{sk}) = \min_{\substack{\sigma \in \mathsf{sk} \\ |\sigma| = m}} \{d(T, \sigma)\}.$$

Pick an arbitrary $\sigma \in \mathsf{sk}$. Then $\widetilde{\mathcal{W}}$ is sound by the following. Note that the second and third inequalities are not immediate, but follow from the proofs of Lemmas 2.3 and 2.4 in [KTHL23].

$$\begin{aligned}
\Pr\left[\widetilde{\mathsf{Detect}}_{\mathsf{sk}}(T) = 1\right] &= \Pr\left[\phi(T, \mathsf{sk}) < \rho\right] \\
&\leq n \Pr\left[d(T, \sigma) < \rho\right] && \text{(Union bound)} \\
&\leq n \Pr\left[\mathbb{E}[d(T, \sigma)] - d(T, \sigma) > -\rho\right] && (\mathbb{E}_{\mathsf{sk}}[d(T, \sigma)] = 0) \\
&\leq 2n \exp\Big(-\frac{2(-\rho)^2}{m(1/2)^2}\Big) && \text{(Hoeffding's bound)} \\
&= 2n \exp(-\lambda).
\end{aligned}$$

Next, fix a prompt $Q \in \mathcal{T}^*$ of length $|Q| \leq \text{poly}(\lambda)$ and choose an efficient adversary $\mathcal{A}$. To show that $\widetilde{\mathcal{W}}$ is non-adaptively R_1^{KTHL}-robust, it suffices to show that $\widetilde{\mathsf{Detect}_{\mathsf{sk}}}(\hat{T}) = 1$ with overwhelming probability, conditioned on $R_1^{\mathsf{KTHL}}(\lambda, Q, T, \hat{T}) = 1$, where $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $T \leftarrow \mathsf{Wat}_{\mathsf{sk}}(m, p_Q, \Gamma)$, and $\hat{T} \leftarrow \mathcal{A}(1^\lambda, T)$. Let σ be the length-m substring of sk used by Γ to generate T. Then we have

$$\begin{aligned}
\Pr\left[\widetilde{\mathsf{Detect}_{\mathsf{sk}}}(\hat{T}) = 1\right] &= \Pr\left[\phi(\hat{T}, \mathsf{sk}) < \rho\right] \\
&\geq \Pr\left[d(\hat{T}, \sigma) < \rho\right] \\
&= 1 - \Pr\left[d(\hat{T}, \sigma) \geq \rho\right]
\end{aligned}$$

All that is left is to show that $\Pr\left[d(\hat{T}, \sigma) \geq \rho\right]$ is negligible. By [KTHL23, Lemma 2.3, Observation B.1], we have that $\mathbb{E}[d(\hat{T}, \sigma)] = -mC_0\hat{\alpha}(T, \hat{T})$. Since we are conditioning on the fact that T and $\hat{T}$ pass the R_1^{KTHL}-robustness condition, Lemma A.13 implies $\mathbb{E}[d(\hat{T}, \sigma)] \leq 2\rho$. Then applying Hoeffding's bound completes the proof.

$$\begin{aligned}
\Pr\left[d(\hat{T}, \sigma) \geq \rho\right] &= \Pr\left[d(\hat{T}, \sigma) - \mathbb{E}[d(\hat{T}, \sigma)] \geq \rho - \mathbb{E}[d(\hat{T}, \sigma)]\right] \\
&\leq \Pr\left[d(\hat{T}, \sigma) - \mathbb{E}[d(\hat{T}, \sigma)] \ \geq -\rho\right] \qquad (\rho - \mathbb{E}[d(\hat{T}, \sigma)] \geq -\rho) \\
&\leq \exp\Big(-\frac{2(-\rho)^2}{m(1/2)^2}\Big) \\
&< \exp(-\lambda).
\end{aligned}$$

□

A.6 Green list / red list schemes [KGW+23a, ZALW23]

The zero-bit watermarking scheme of Kirchenbauer, et al. [KGW+23a] is not undetectable, nor does it appear to enjoy the sort of provable soundness and robustness guarantees required to apply our constructions. A heuristic version of our main construction applied to this scheme may work well in practice, though empirical analysis is well beyond our present scope.

We give details below, focusing on the simplified variant in [ZALW23]. You may safely skip the rest of this subsection. We include it mainly to aid readers interested in understanding [ZALW23].

The core idea in the construction is to randomly partition the token set $\mathcal{T}$ into a *green list* $\mathcal{G}$ and *red list* $\mathcal{R} = \mathcal{T} \setminus \mathcal{G}$, and preferentially sample tokens from $\mathcal{G}$.[14] Note that by changing the distribution over tokens, the scheme is not undetectable. The detection algorithm performs a hypothesis test on the fraction of green tokens in a text, declaring the text marked if the fraction is significantly greater than some expected threshold (i.e., rejecting the null hypothesis that the text is not marked).

Empirically and heuristically, the scheme appears well suited to a block-by-block interpretation. Detecting the watermark requires a text $\hat{T}$ to contain a long enough substring that was (close to a substring) produced by the watermarked model. For example, see [KGW+23b] which refines the original Detect algorithm by testing every substring of the input text (among other improvements), and empirically analyzes robustness to paraphrasing and copy-paste attacks.

Unfortunately, it does not appear that the scheme provides the sort of provable guarantees we need to view it as a block-by-block scheme. In brief, the scheme does not appear to simultaneously enjoy both non-trivial soundness (low false positives) for all strings and non-trivial completeness (low false negatives) for unmodified outputs of the watermarked model. Either type of error can be bounded (even negligibly small), at the cost of destroying the provable guarantee on the other. Generically turning this scheme into a block-by-block scheme seems to require simultaneously bounding both types of errors.

[14] The difference between [ZALW23] and the original construction of [KGW+23a] is that in the original, the green and red lists change as a function of the preceding tokens.

The construction is parameterized by constants $\gamma \in (0,1)$ and $\delta > 0$. The parameter γ governs the size of the green set: $|\mathcal{G}| = \gamma|\mathcal{T}|$. The parameter δ governs the amount that the watermarked model is biased towards green tokens, as described below. The secret key is the green set: $\mathsf{sk} = \mathcal{G}$. The Detect algorithm computes a z-score and compares it to some threshold ρ (which may depend on the input T). That is, $\mathsf{Detect}_{\mathsf{sk}}(T) := \mathbb{1}\left(z_{\mathsf{sk}}(T) > \rho(T)\right)$, where $\rho(T)$ is a threshold and $z_{\mathsf{sk}}(T) := (\sum_{i=1}^{|T|} \mathbb{1}(\tau_i \in \mathcal{G}) - \gamma|T|)/\sqrt{|T|\gamma(1-\gamma)}$.

In particular, one would need to set the threshold $\rho(T)$ below to satisfy both Theorems A.15 and A.16. Consider the typical case of $\gamma = 1/2$. Theorem A.15 requires $\rho(T) \geq 64\lambda C_{\max}(T)/\sqrt{|T|}$. If $C_{\max}(T) > |T|/64\lambda$, then $\rho(T) > \sqrt{T}$ is needed.[15] But Theorem A.16 requires $\rho(T) < \sqrt{T}$, as $\kappa < 1$ and $\gamma = 1/2$. One cannot have both, regardless of λ and the error rates α, β.

Given Model, prompt Q, and string T, let p_i be the probability distribution over token i in the output of $\mathsf{Model}(Q)$ conditioned on $T_{1:i-1}$.

$$p_i(\tau \mid Q \| T_{<i}) := \Pr[\mathsf{Model}(Q \| T_{<i})_1 = \tau].$$

The marked model Wat samples the next token τ with probability $p_i^{\mathcal{G}}(t|Q\|T_{1:i})$, defined as:

$$p_i^{\mathcal{G}}(\tau \mid Q \| T_{<i}) \propto \exp(\delta \cdot \mathbb{1}(\tau \in \mathcal{G})) \cdot p(\tau \mid Q \| T_{<i}).$$

In other words, $p^{\mathcal{G}}$ is defined by upweighting the probabilities of $\tau \in \mathcal{G}$ by a factor of e^{δ} and the distribution is renormalized (equivalently, adding δ to the logits and computing the soft-max).

The soundess guarantee of [ZALW23] is given in terms of two functions of a string $T \in \mathcal{T}^*$. These functions $C_{\max}$ and V both take values in $[0, |T|]$. The function $C_{\max}$ is the more important function for our purposes: it counts the number of occurences of the most frequent token in a string T.

$$C_{\max}(T) := \max_{\tau \in \mathcal{T}} \sum_{i \in |T|} \mathbb{1}(\tau_i = \tau) \quad V(T) := \frac{1}{|T|} \sum_{\tau \in \mathcal{T}} \left(\sum_{i \in |T|} \mathbb{1}(\tau_i = \tau) \right)^2$$

Theorem A.15 (Soundness, Theorem C.4 of [ZALW23])**.** *For any $T \in \mathcal{T}^*$*

$$\Pr_{\mathsf{sk}}\left[z_{\mathsf{sk}}(T) > \sqrt{\frac{64\log(9/\alpha)V(T)}{1-\gamma}} + \frac{16\log(9/\alpha)C_{\max}(T)}{\sqrt{|T|\gamma(1-\gamma)}} \right] < \alpha. \tag{4}$$

In particular, taking $\rho(T) \geq \sqrt{\frac{64\log(9)\lambda V(T)}{1-\gamma}} + \frac{16\log(9)\lambda C_{\max}(T)}{\sqrt{|T|\gamma(1-\gamma)}}$, we get $\Pr[\mathsf{Detect}_{\mathsf{sk}}(T) = 1] < 2^{-\lambda}$.

Completeness requires two conditions on Model for the prompt Q whose definitions we omit: on-average high entropy and on-average homophily ([ZALW23, Assumptions C.9, C.12]).

Theorem A.16 (Completeness, adapted from Theorem C.13 of [ZALW23])**.** *Fix Model and Q. Suppose that $\beta, \kappa \in (0,1)$ and $|T|$ satisfy the following, where c_1 and c_2 are some constants that depend on the parameters δ and γ.*

- $|T| \geq c_1 \cdot \frac{\log(1/\beta)}{(1-\kappa)^2}$.
- Model *has β-on-average-homophily for Q*
- Model *has $(\xi, \beta/3)$-on-average-high-entropy for Q, for $\xi = c_2 \cdot \frac{1-\kappa}{\log^2(|T|/\beta)}$*

Then

$$\Pr\left[z_{\mathsf{sk}}(T) < \frac{\kappa(e^{\delta}-1)\sqrt{|T|\gamma(1-\gamma)}}{1+(e^{\delta}-1)\gamma} \right] \leq \beta \tag{5}$$

In particular, $\Pr[\mathsf{Detect}_{\mathsf{sk}}(T) = 0] \leq 2^{-\lambda}$ for threshold $\rho(T) < \frac{\kappa(e^{\delta}-1)\sqrt{|T|\gamma(1-\gamma)}}{1+(e^{\delta}-1)\gamma}$.

[15] Observe that $C_{\max}(T)/|T|$ is the frequency of the most common token in T. For natural language, $C_{\max}(T) = \Omega(T)$ is typical. For example, about 7% of the words in the Brown Corpus are "the". The condition $C_{\max}(T) > |T|/64\lambda$ only requires that there exists a token with frequency $1/64\lambda$ in T.

B Reference: definition variants

B.1 Zero-bit watermarking

Definition B.1 (Undetectability – zero-bit)**.** *A zero-bit watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ *for* Model *is* undetectable *if for all efficient adversaries* $\mathcal{A}$*,*

$$\left| \Pr[\mathcal{A}^{\mathsf{Model}(\cdot)}(1^\lambda) = 1] - \Pr_{\mathsf{sk} \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot)}(1^\lambda) = 1] \right|$$

is at most $\mathsf{negl}(\lambda)$*.*

Definition B.2 (Soundness – zero-bit)**.** *A zero-bit watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ *is* sound *if for all polynomials* poly *and all strings* $T \in \mathcal{T}^*$ *of length* $|T| \leq \mathrm{poly}(\lambda)$*,*

$$\Pr_{\mathsf{sk} \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathsf{Detect}_{\mathsf{sk}}(T) \neq 0] < \mathsf{negl}(\lambda).$$

For all of the robustness definitions below, completeness can be defined by including the extra clause $\hat{T} = T$ (or $\hat{T} \in (T_i)_i$ in the adaptive setting). To save space, we only explitly define completeness in Definition B.3.

Definition B.3 (R-Robust/Complete detection – zero-bit, non-adaptive)**.** *A zero-bit watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ *is* non-adaptively R-robustly/completely detectable *with respect to the robustness condition* R *if for all efficient adversaries* $\mathcal{A}$*, all polynomials* poly*, and all prompts* $Q \in \mathcal{T}^*$ *of length* $|Q| \leq \mathrm{poly}(\lambda)$*, the following event* `FAIL` *occurs with negligible probability:*

- $\hat{T} = T$*, AND* `// the adversary outputs` T
- $R(\lambda, Q, T, \hat{T}) = 1$*, AND* `// the robustness condition passes`
- $\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 0$ `// the mark is removed`

in the probability experiment defined by

- $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$
- $T \leftarrow\!\$\ \mathsf{Wat}_{\mathsf{sk}}(Q)$
- $\hat{T} \leftarrow \mathcal{A}(1^\lambda, T)$*.*

Definition B.4 (R-Robust detection – zero-bit, adaptive)**.** *A zero-bit watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect})$ *is* adaptively R-robustly detectable *with respect to the robustness condition* R *if for all efficient adversaries* $\mathcal{A}$*, the following event* `FAIL` *occurs with negligible probability:*

- $R(\lambda, (Q_i)_i, (T_i)_i, \hat{T}) = 1$*, AND* `// the robustness condition passes`
- $\mathsf{Detect}_{\mathsf{sk}}(\hat{T}) = 0$ `// the mark is removed`

in the probability experiment defined by

- $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$
- $\hat{T} \leftarrow \mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot)}(1^\lambda)$*, denoting by* $(Q_i)_i$ *and* $(T_i)_i$ *the sequence of inputs and outputs of the oracle.*

We also say a scheme satisfying this definition is (δ, R)-robust*.*

B.2 L-bit watermarking

Definition B.5 ((δ, R)-Robust extraction – L-bit, non-adaptive)**.** *An L-bit watermarking scheme $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ is* non-adaptively (δ, R)-robustly extractable *with respect to the robustness condition R if for all efficient adversaries $\mathcal{A}$, all messages $m \in \{0,1\}^L$, all polynomials* poly*, and all prompts $Q \in \mathcal{T}^*$ of length $|Q| \leq \mathrm{poly}(\lambda)$, the following event* `FAIL` *occurs with negligible probability:*

- $R(\lambda, Q, T, \hat{T}) = 1$, *AND* `// the robustness condition passes`
- $\hat{m} \notin B_\delta(m)$ `// the mark is corrupted`

in the probability experiment define by

- $\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)$
- $T \leftarrow\!\!\$\ \mathsf{Wat}_{\mathsf{sk}}(m, Q)$
- $\hat{T} \leftarrow \mathcal{A}(1^\lambda, T)$
- $\hat{m} \leftarrow \mathsf{Extract}_{\mathsf{sk}}(\hat{T})$.

Lemma B.6 (L-bit Adaptivity from Undetectability)**.** *Let $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ be a block-by-block L-bit watermarking scheme that is undetectable. If $\mathcal{W}$ is non-adaptively R_1-robust, then it is adaptively R_1-robust.*

Proof. Fix some $m \in \{0,1\}^L$. For any R, non-adaptive R-robustness for any fixed prompt implies non-adaptive R-robustness for any distribution over prompts. The latter implies adaptive R-robustness for any adversary $\mathcal{A}_1$ making only a *single query* to its oracle, as the queried prompt is sampled from a fixed distribution independent of sk. It remains to show that adaptive R_1-robustness for single query implies adaptive R_1-robustness for any $q = \mathrm{poly}(\lambda)$ queries.

Suppose for contradiction that there exists an adaptive-robustness adversary $\mathcal{A}^{\mathcal{O}(\cdot)}$ making q queries for which $\Pr[\texttt{FAIL}] \geq f$, for $q, f = \mathrm{poly}(\lambda)$. We define $\mathcal{A}_1^{\mathsf{Wat}_{\mathsf{sk}}(\cdot)}$ making a single query as follows.

1. Sample $j^* \leftarrow\!\!\$\ [q]$ uniformly at random.
2. Run $\mathcal{A}^{\mathcal{O}(\cdot)}(1^\lambda)$, responding to query Q_i as follows:
   (a) For $i \neq j^*$, return $T_i \leftarrow \mathsf{Model}(Q_i)$.
   (b) For $i = j^*$, issue challenge query Q_{j^*} and return $T_{j^*} \leftarrow \mathsf{Wat}_{\mathsf{sk}}(m, Q_{j^*})$.
3. When $\mathcal{A}$ outputs $\hat{T}$, output $\hat{T}$.

By undetectability of $\mathcal{W}$,

$$\Pr\left[R_1\big(\lambda, (Q_i)_i, (T_i)_i, \hat{T}\big) = 1 \ \wedge\ \mathsf{Extract}_{\mathsf{sk}}(\hat{T}) \neq m\right] \geq f - \mathsf{negl}(\lambda)$$

Recall that $R_1(\lambda, (Q_i)_i, (T_i)_i, \hat{T}) = 1$ iff there exists a substring $\hat{\tau} \in \hat{T}$ that is $\doteq$-close to a block in $\cup_i \mathsf{Blocks}(T_i; Q_i)$, for functions $\doteq$ and Blocks as defined by R_1. Hence if $R_1(\lambda, (Q_i)_i, (T_i)_i, \hat{T}) = 1$, there exists $i^* \in [q]$ such that $R_1(\lambda, Q_{i^*}, T_{i^*}, \hat{T}) = 1$. By construction, $j^* = i^*$ with probability $1/q$. Therefore,

$$\Pr\left[R_1\big(\lambda, Q_{j^*}, T_{j^*}, \hat{T}\big) = 1 \ \wedge\ \mathsf{Extract}_{\mathsf{sk}}(\hat{T}) \neq m\right] \geq \frac{f - \mathsf{negl}(\lambda)}{q}$$

violating the hypothesis that $\mathcal{W}$ is adaptively R_1-robust for a single query. □

B.3 Multi-user watermarking

Definition B.7 (Undetectability – multi-user)**.** *Define the oracle* $\mathsf{Model}'(u, Q) := \mathsf{Model}(Q)$*. A multi-user watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ *for* Model *is* undetectable *if for all efficient adversaries* $\mathcal{A}$*,*

$$\left|\Pr[\mathcal{A}^{\mathsf{Model}'(\cdot,\cdot)}(1^\lambda) = 1] - \Pr_{\mathsf{sk} \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot,\cdot)}(1^\lambda) = 1]\right|$$

is at most $\mathsf{negl}(\lambda)$*.*

Definition B.8 (Soundness – multi-user)**.** *A multi-user watermarking scheme* $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Detect}, \mathsf{Trace})$ *is* sound *if for all polynomials* poly *and all strings* $T \in \mathcal{T}^*$ *of length* $|T| \le \mathrm{poly}(\lambda)$*,*

$$\Pr_{\mathsf{sk} \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)}[\mathsf{Detect}_{\mathsf{sk}}(T) = 1] \le \mathsf{negl}(\lambda).$$

C Proof of Lemma 2.4

Lemma C.1. *For* $\lambda, L \ge 1$ *and* $0 \le \delta < 1$*, define*

$$k^*(L, \delta) = \min\left\{ L \cdot (\ln L + \lambda);\quad L \cdot \ln\left(\frac{1}{\delta - \sqrt{\frac{\lambda + \ln 2}{2L}}}\right)\right\} \tag{6}$$

Then, after throwing $k \ge k^*(L, \delta)$ *balls into* L *bins, fewer than* δL *bins are empty except with probability at most* $e^{-\lambda}$*.*

Proof. If $k \ge L(\ln L + \lambda)$ balls are thrown into L bins, all bins are occupied except with probability at most $L(1 - \frac{1}{L})^{L(\ln L + \lambda)} < Le^{-(\ln L + \lambda)} = e^{-\lambda}$. In this case, 0 bins are empty, and the claim holds.

Now suppose $L(\ln L + \lambda) > k \ge -L \ln\left(\delta - \sqrt{\frac{\lambda + \ln 2}{2L}}\right)$. Then $\delta > \sqrt{\frac{\lambda + \ln 2}{2L}} > 0$. The analysis uses the Poisson approximation to balls and bins. Let $X = (X_1, \ldots, X_L)$ be a multinomial random variable over $\mathbb{Z}_{\ge 0}^L$ where each X_i denotes the number of balls in bin i. Let $Y = (Y_1, \ldots, Y_L)$ where each $Y_i \sim \mathrm{Pois}(k/L)$ i.i.d. is Poisson with mean k/L. Let $E = \{x \in \mathbb{N}_0^L : \sum_i \mathbb{1}(x_i = 0) > \delta L\}$ be the event that more than δL bins are empty.

Let $W_i = \mathbb{1}\{Y_i = 0\}$ and $W = \sum_{i=1}^L W_i$. $\mathbb{E}[W] = Le^{-k/L}$. Applying a Hoeffding bound,

$$\begin{aligned}\Pr_Y[E] &= \Pr[W - \mathbb{E}[W] > L(\delta - e^{-k/L})] \\ &\le \exp\left(-2L(\delta - e^{-k/L})^2\right) \\ &\le \exp(-\lambda - \ln 2)\end{aligned}$$

Where the last inequality comes from our choice of k. Observe that the above requires $\delta - e^{-k/L} > 0$, which holds because $k > L \ln(1/\delta)$.

The Poisson approximation gives $\Pr_X[E] \le 2 \Pr_Y[E]$ for any event E that is monotically decreasing with the number of balls k [MU05, Corollary 5.11]. Throwing more balls only decreases the probability that more than δL are empty. Hence $\Pr_X[E] \le e^{-\lambda}$. □

D Discussion of sub-uniform recovery

Our main L-bit watermarking scheme allows $\delta \cdot L$ bits of the embedded message to be erased. It is natural to hope that the erased bits are distributed uniformly at random. But an adversary might want to erase some bits more than others. For example, the higher-order bits if the message is a timestamp.

Can we ensure that the adversary cannot influence which indices of the message are erased? In the proof of Lemma 5.3, we transition to a game (Hybrid 3) where the indices are erased uniformly at random. Perhaps surprisingly, however, this does not imply that our scheme extracts a message with uniformly random erasures.[16] We present an informal argument to provide intuition as to why this implication cannot hold.

Suppose the adversary is using our L-bit scheme and can choose which messages are embedded into the text. Then they could generate watermarked outputs $T_0 \leftarrow \mathsf{Wat}_{\mathsf{sk}}(m_0, Q_0)$ and $T_1 \leftarrow \mathsf{Wat}_{\mathsf{sk}}(m_1, Q_1)$, where $m_0 = 0^L$ and $m_1 = 0\|1^{L-1}$. Suppose they edit T_0 and T_1, editing blocks uniformly at random, ultimately outputting some $\hat{T}$ that contains enough of T_0 and T_1 such that every index in m_0 and m_1 is embedded once in expectation. In this case, there would likely be two blocks in $\hat{T}$ generated using $k_{1,0}$, none generated using $k_{1,1}$, and one generated using $k_{i,b}$ for every other $i \in [L], b \in \{0,1\}$. Suppose our $\mathsf{Extract}$ algorithm is better at extracting when two blocks generated with the same key are present than it is when only one block is present. It would then be more likely that $\mathsf{Extract}$ recovers the first bit of the embedded message than any other bit.

Intuitively, this may not seem like an issue. In fact, by making it easier to recover certain bits, the adversary appears to be helping the watermarker. But this is not strictly better than having uniformly random erasures. To see why, we can model the adversary's behavior by first erasing bits uniformly at random and then allowing the adversary to *choose* which bits become "unerased." We call the resulting distribution of the indices of erased bits "sub-uniform." A scheme that is secure under uniformly random erasures is not necessarily secure against a sub-uniform adversary. For instance, the Tardos fingerprinting code [Tar08] is secure against uniform erasures. If the watermarking scheme is embedding codewords from the Tardos code, the adversary may be able to reveal bits that make it harder for them to be traced.

To be clear, this adversary is exceptionally weak. If a fingerprinting code is robust to uniform erasures and has the (informal) property that revealing more bits always improves the chance of tracing to a guilty party, then it will also be robust to sub-uniform erasures. This is an intuitive property for a fingerprinting code to have, although not all satisfy it (and most do not prove it).

Before we go on to prove any results, we formally define sub-uniform distributions in Definition D.1. Informally, we can compare the definition to the above game as follows. First, we choose a uniformly random subset Y, then an adversary can pick any $X \subseteq Y$. This corresponds exactly to erasing uniformly random indices of a message (those in Y) and then allowing an adversary to unerase indices of its choice (those in $Y \setminus X$). The indices that remain are those in X.

Definition D.1. *For a universe $\mathcal{U}$, integer $0 \leq s \leq |\mathcal{U}|$, and random variable of a uniformly random subset of size s, $U_s \subseteq \mathcal{U}$, we say a random variable V supported on $2^{\mathcal{U}}$ is* s-sub-uniform *if there exists a distribution (coupling) $W = (X, Y)$ over $2^{\mathcal{U}} \times 2^{\mathcal{U}}$, with marginal distributions $X = V$ and $Y = U_s$, such that $X \subseteq Y$ always.*

Lemma D.2 formalizes the type of distribution that our erasures follow. So long as fingerprinting codes are robust with respect to sub-uniform erasures, they can be used in a black-box way with our L-bit watermarking construction in Figure 2 to create multi-user watermarks. In the following lemma, we use the notation Proj_δ to denote a randomized function that adds erasures uniformly at random to its input until it has at least $\lfloor \delta L \rfloor$ entries equal to $\perp$.

Lemma D.2. *Suppose $\mathcal{W}'$ is a block-by-block zero-bit watermarking scheme that is undetectable, sound, and R_1-robustly detectable. Let $\mathcal{W} = (\mathsf{KeyGen}, \mathsf{Wat}, \mathsf{Extract})$ be the L-bit watermarking scheme from Figure 2 using $\mathcal{W}'$.*

Let $I_\perp(\hat{m}) = \{i : i \in [L], \hat{m}[i] = \perp\}$ and $0 \leq \delta < 1$. Then, for all efficient $\mathcal{A}$, the set $I_\perp(\hat{m})$ is computationally indistinguishable from a $\lfloor \delta L \rfloor$-sub-uniform random variable in the probability experiment defined by

- $\mathsf{sk} \leftarrow\!\!\$\ \mathsf{KeyGen}(1^\lambda)$
- $\hat{T} \leftarrow \mathcal{A}^{\mathsf{Wat}_{\mathsf{sk}}(\cdot,\cdot)}(1^\lambda)$

[16] The implication does hold if the adversary only ever queries a single message.

- $\hat{m} \leftarrow \mathsf{Proj}_\delta(\mathsf{Extract}_{\mathsf{sk}}(\hat{T}))$

and conditioned on R_k being satisfied.

Proof outline. The proof follows via the techniques used in the proof of Lemma 5.3. Notice that the code of Hybrid 3 will return a message with uniformly randomly distributed erasures. If we consider running the same adversary with $\mathcal{W}$ in Hybrid 3 and in the original game, then we can construct a distribution W based on this adversary. Specifically, the distribution of erasures in Hybrid 3 is indistinguishable from a uniformly random subset of size $\lfloor \delta L \rfloor$ because we have composed it with Proj_δ. Call this subset $I^3_\perp(\hat{m})$. Based on the arguments in Hybrids 1 and 2, we know that the original game will only have strictly fewer $\perp$ entries. So the joint distribution $W = (I_\perp(\hat{m}), I^3_\perp(\hat{m}))$, is indistinguishable from a $\lfloor \delta L \rfloor$-sub-uniform joint distribution. $\square$