
Watermarking Makes Language Models Radioactive

Tom Sander*
Meta FAIR & École polytechnique

Pierre Fernandez*
Meta FAIR & Inria Rennes

Alain Durmus
École polytechnique

Matthijs Douze
Meta FAIR

Teddy Furon
Inria Rennes

Abstract

We investigate the *radioactivity* of text generated by large language models (LLM), i.e., whether it is possible to detect that such synthetic input was used to train a subsequent LLM. Current methods like membership inference or active IP protection either work only in settings where the suspected text is known or do not provide reliable statistical guarantees. We discover that, on the contrary, it is possible to reliably determine if a language model was trained on synthetic data if that data is output by a watermarked LLM. Our new methods, specialized for radioactivity, detects with a provable confidence weak residuals of the watermark signal in the fine-tuned LLM. We link the radioactivity contamination level to the following properties: the watermark robustness, its proportion in the training set, and the fine-tuning process. For instance, if the suspect model is open-weight, we demonstrate that training on watermarked instructions can be detected with high confidence (p -value $< 10^{-5}$) even when as little as 5% of training text is watermarked. Radioactivity detection code is available at <https://github.com/facebookresearch/radioactive-watermark>

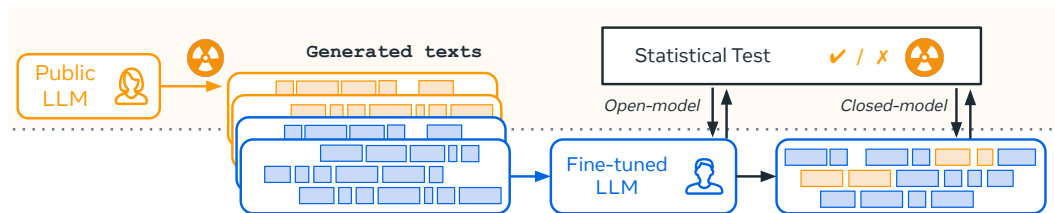


Figure 1: Bob fine-tunes his LLM on data with a fraction coming from Alice’s LLM. This leaves traces in Bob’s model that Alice can detect reliably, provided that her text was watermarked. Thus, a side effect of Alice’s watermark, intended for machine-generated text detection, is to reveal what data Bob’s model was fine-tuned on.

1 Introduction

Large Language Models (LLMs) are often instruction fine-tuned to align them with human prompts and improve their performance and generalization [Ouyang et al., 2022, Wei et al., 2022, Chung et al., 2022]. Fine-tuning requires expert knowledge to balance diversity and quality in the instruction dataset and a costly collection of manual annotations, especially for alignment [OpenAI, 2023, Touvron et al., 2023b, Gemini, 2023]. To address the cost and the difficulties of fine-tuning, practitioners often train on synthetic data generated by a model that has already been instructed, such as Bard, ChatGPT, or Claude. For example, works by Wang et al. [2022], Honovich et al. [2022], Peng et al. [2023a] created instruction data for many of the most recent LLMs [Taori et al., 2023, Xu et al., 2023,

*Equal Contribution. Correspondence at {tomsander,pfz}@meta.com

Gunasekar et al., 2023, Mukherjee et al., 2023]. This may also be unintentional when, for example, Turkers use ChatGPT to perform their tasks [Veselovsky et al., 2023]. Such imitation raises questions about whether the fine-tuned model is a derivative work of the original model [Wallace et al., 2020]. In this context, it is crucial to understand how to detect when LLM outputs are used as training data.

Meanwhile, recent AI regulations enforce the transparency of generative models. This is increasingly important in cases where the generated content may be used for malicious purposes [Weidinger et al., 2022, Crothers et al., 2022]. One approach is *watermarking*. It embeds a secret trace in the synthetic content that can be detected to identify the generating model. In the context of LLMs, recent techniques make detection efficient with minimal degradation of the generated text quality by altering the sampling of next tokens [Aaronson and Kirchner, 2023, Kirchenbauer et al., 2023a,b].

Based on these two observations, this study addresses the following question:

What occurs when watermarked text is employed as fine-tuning data?

We explore the potential “radioactivity” – a term coined by Sablayrolles et al. [2020] – of LLM watermarking, which refers to the capacity of watermarked training data to contaminate a model.

We examine a model that has been fine-tuned on a corpus that may contain watermarked text (see Fig. 1). The baseline method for detecting radioactivity executes the original watermark detection on the outputs generated by this model. However, this approach proves ineffective because the residual of the watermark is a weak signal hardly detectable in plain output text. In this work, we are able to demonstrate that LLM watermarking is indeed radioactive thanks to our specific protocol designed for revealing weak contamination traces. Our contributions include:

- We design radioactivity detection methods for four scenarios based on model (*open / closed*) and training data (*supervised / unsupervised*) access. Notably, our open-model detection (Fig. 3) improves the performance by orders of magnitudes.
- We show how to obtain reliable p -values for watermark detection when scoring millions of tokens.
- We prove that watermarked text is radioactive in a real-world setting where an LLM is fine-tuned on slightly watermarked instruction data. For instance, in the open-model scenario, our tests detect radioactivity with a p -value of 10^{-5} when only 5% of fine-tuning data is watermarked (Fig. 5).

2 Background

2.1 Related work

Watermarking for LLMs. A recent branch of watermarking methods for decoder-only LLMs modifies either the probability distribution [Kirchenbauer et al., 2023a] or the sampling method of the next token [Aaronson and Kirchner, 2023, Kuditipudi et al., 2023]. Theoretical studies indicate that detectability depends on the entropy of generated text [Christ et al., 2023, Huang et al., 2023]. Subsequent research suggests watermarking entropic passages, particularly in code [Lee et al., 2023], while other works focus on “semantic” watermarks that depend on an entire past text’s semantic representation [Liu et al., 2023, Liu and Bu, 2024, Fu et al., 2024b].

Gu et al. [2023] distill the methods used in this study within the model weights, allowing LLMs to generate watermarked logits natively, which is key for open-source models. In contrast, we focus on unintentional contamination: Alice and Bob in Fig. 1 are not collaborating, and Bob consumes only a small proportion of watermarked data.

Membership inference attacks (MIAs) aim to determine whether an arbitrary sample is included in a model’s training data, with varying granularity on the adversary’s knowledge [Nasr et al., 2019]. Most of the time, the detection either build shadow models and observe a difference in their behavior [Shokri et al., 2017a,b, Hisamoto et al., 2020, Mahloujifar et al., 2021] or directly observe the loss of the model [Yeom et al., 2018, Sablayrolles et al., 2019, Watson et al., 2021, Carlini et al., 2022]. In the context of generative models, MIAs are intertwined with *dataset contamination* where one detects that an entire dataset is part of the training data [Shi et al., 2023, Golchin and Surdeanu, 2023]. MIAs can violate the confidentiality of sensitive training or reveal training on “forbidden” data, like copyrighted material or evaluation data (which undermines benchmark results).

MIA may be used for radioactivity detection, but with a strong limitation. Since it focuses on specific pieces of text, Alice in Fig. 1 has to record all the outputs of her LLM.

IP Protection. Watermarking can be used for intellectual property protection. For instance, He et al. [2022b,a], Li et al. [2023] use lexical properties like synonyms whereas Peng et al. [2023b] rely on backdoors. Zhao et al. [2023] develop a watermark dissuading model theft via distillation. Yet its accuracy is empirical, it does not provide p -values that align with empirical false positive rates.

Radioactivity. Sablayrolles et al. [2020] introduce the concept of *radioactivity*: images are modified to leave a detectable trace in any classifier trained on them. Our work studies the radioactivity of decoding-based LLM watermarks, which are primarily used to detect AI-generated text with proven accuracy. We demonstrate that this form of radioactivity is reliably detectable across diverse settings. Appendix E.2 details why existing IP protections or MIAs do not offer similar capabilities.

2.2 Technical background for LLM watermarking

This paper focuses on watermarking schemes that modify the LLM decoding by hashing a watermark window. In experiments we use [Aaronson and Kirchner, 2023, Kirchenbauer et al., 2023a] due to their omnipresence in the literature, their performance, and their practicality. We briefly overview them and refer the reader to Appendix C for details (on the watermark detection tests especially).

We consider a decoder-only LLM that takes as input a context (a sequence of tokens $(x^{(-C)}, \dots, x^{(-1)}) \in \mathcal{V}^C$, \mathcal{V} being the vocabulary of the model) and outputs a vector of logits $\ell \in \mathbb{R}^{|\mathcal{V}|}$. Vector ℓ is transformed into $\mathbf{p} = \text{softmax}(\ell) \in [0, 1]^{|\mathcal{V}|}$, the probability distribution of the next token. The text is generated by sampling the next token $x^{(0)}$ from this distribution with some procedure (top-k sampling [Fan et al., 2018, Radford et al., 2019], nucleus-sampling [Holtzman et al., 2019], etc.), then appending it to the context, and repeating the process.

The *watermark embedding* alters the logit vector ℓ or the sampling procedure depending on a secret key. Usually, the output of a secret-key cryptographic function hashes k previous tokens $(x^{(-k)}, \dots, x^{(-1)})$ (the watermark window) and the secret-key \mathbf{s} . It serves as a seed for a random number generator, that influences the choice of the next token $x^{(0)}$. For instance, Kirchenbauer et al. [2023b] create a pseudo-random “greenlist” of tokens with proportion γ of the entire vocabulary, whose logits are incremented by a quantity δ , increasing their sampling probability.

The *watermark detection* tokenizes a text, replays the seed generation and scores each token. The score function on a current token $x^{(0)}$ may therefore be summed up as W_{score} that also takes as input the watermark window $(x^{(-k)}, \dots, x^{(-1)})$, and depends on the hashing’s secret-key \mathbf{s} :

$$\begin{array}{c} \text{Watermark window (k previous tokens)} \\ \downarrow \\ x^{(0)} ; (x^{(-k)}, \dots, x^{(-1)}) \mapsto W_{\text{score}} \left(x^{(0)} ; \mathbf{s}, (x^{(-k)}, \dots, x^{(-1)}) \right) \in \mathbb{R}. \quad (1) \\ \uparrow \qquad \qquad \qquad \uparrow \\ \text{Current token being scored} \qquad \text{Scoring function (e.g., 1 if green token, 0 otherwise)} \end{array}$$

A statistical test is performed on the cumulative score $S(X_N)$ – which follows a known distribution in the absence of watermark, see App. C – where $X_N := [x_1, \dots, x_N]$ is a list of N tuples and:

$$S(X_N) := \sum_{t=1}^N W_{\text{score}}(x_t^{(0)}; \mathbf{s}, (x_t^{(-i)})_{i=k}^1). \quad (2)$$

3 Problem Formulation

Alice owns a language model \mathcal{A} , fine-tuned for specific tasks such as chatting, problem solving, or code generation, which is available through an API (Figure 1). Bob owns another language model \mathcal{B} . Alice suspects that Bob fine-tuned \mathcal{B} on some outputs from \mathcal{A} . We denote by D the dataset used to fine-tune \mathcal{B} , among which $D^{\mathcal{A}} \subset D$ is made of outputs from \mathcal{A} , in proportion $\rho = |D^{\mathcal{A}}|/|D|$.

Access to Bob’s data. We consider two settings for Alice’s knowledge about Bob’s training data:

- *supervised*: Bob queries \mathcal{A} and Alice retains all the content $\tilde{D}^{\mathcal{A}}$ that \mathcal{A} generated for Bob. Thus, Alice knows that $D^{\mathcal{A}} \subseteq \tilde{D}^{\mathcal{A}}$. We define the *degree of supervision* $d := |D^{\mathcal{A}}|/|\tilde{D}^{\mathcal{A}}|$,

- *unsupervised*: Bob does not use any identifiable account or is hiding behind others such that $|\tilde{D}^{\mathcal{A}}| \gg |D^{\mathcal{A}}|$ and $d \approx 0$. This is the most realistic scenario.

Thus, ρ is the proportion of Bob’s fine-tuning data which originates from Alice’s model while d quantifies Alice’s knowledge regarding the dataset that Bob may have utilized (see Fig. 2).

Access to Bob’s model. We consider two scenarios:

- Alice has an *open-model* access to \mathcal{B} . She can forward any inputs through \mathcal{B} and observe the output logits. This is the case if Bob open-sources \mathcal{B} , or if Alice sought it via legitimate channels.
- Alice has a *closed-model* access. She can only query \mathcal{B} through an API without logits access: Alice only observes the generated texts. This would be the case for most chatbots.

We then introduce two definitions of radioactivity:

Definition 1 (Text Radioactivity). *Dataset D is α -radioactive for a statistical test T if “ \mathcal{B} was not trained on D ” $\subset \mathcal{H}_0$ and T is able to reject \mathcal{H}_0 at a significance level (p -value) smaller than α .*

Definition 2 (Model Radioactivity). *Model \mathcal{A} is α -radioactive for a statistical test T if “ \mathcal{B} was not trained on outputs of \mathcal{A} ” $\subset \mathcal{H}_0$ and T is able to reject \mathcal{H}_0 at a significance level smaller than α .*

Thus, α quantifies the radioactivity of a dataset or model. A low α , e.g. 10^{-6} , indicates strong radioactivity: the probability of observing a result as extreme as the one observed, assuming that Bob’s model was not trained on Alice’s outputs, is 1 out of one million. Conversely, $\alpha \approx 0.5$ means that the observed result is equally likely under both the null and alternative (radioactive) hypotheses.

4 Radioactivity Detection

We build radioactivity detectors for all settings as shown in Tab. 1. The outputs of \mathcal{A} are watermarked with a method W with Alice’s secret key s as described in Sec. 2.2. Note that, unlike watermark detection which takes text as input, the input for radioactivity detection is a model.

4.1 Theoretical validity of statistical tests for radioactivity detection

In the following, we construct a radioactivity test on \mathcal{B} based on the detection score of Alice’s watermarking method. We focus on Kirchenbauer et al. [2023a] with the notations of Sec. 2.2 for simplicity. Each x_i in X_N is a $(k + 1)$ -tuple of tokens $(x_i^{(-k)}, \dots, x_i^{(0)})$ and $x_i^{(0)}$ is generated by \mathcal{B} from the preceding tokens $(x_i^{(-C_i)}, \dots, x_i^{(-k)}, \dots, x_i^{(-1)})$, where $(x_i^{(-C_i)}, \dots, x_i^{(-k-1)}) := c_i$ is an optional context, e.g. prompt. Note that the contexts (c_1, \dots, c_N) are crafted by Alice to better detect radioactivity in (x_1, \dots, x_N) . As we see in Sec. 5.3, this can imply that they are watermarked themselves. In this scenario, an accurate statistical test for detecting radioactivity can be performed through de-duplication. Let \mathcal{H}_0 be “ $S(X_N)$ follows a binomial distribution $B(N, \gamma)$ ”. Then:

Proposition 1. “ B was not trained on Alice’s watermarked data” $\subset \mathcal{H}_0$ if tokens are de-duplicated: (1) $(x_i)_{i \leq N}$ are pairwise distinct and (2) for each $1 \leq i \leq N$, x_i is not in the context c_i .

The proof and a formal statement of this result are provided in App. D.2.1.

With this inclusion, we can thus use a statistical test T for hypothesis \mathcal{H}_0 to test against radioactivity. Specifically, assuming conditions (1) and (2) are met, the p -value $P(S(X_N) > t \mid \mathcal{H}_0)$ can be

	With WM		Without WM (MIA)		IPP	
	Open	Closed	Open	Closed	Open	Closed
Supervised	✓	✓	✓	✗	✓	✓
Unsupervised	✓	✓	✗	✗	✗	✗

Table 1: Availability of radioactivity detection under the different settings. *Open / closed-model* refers to the availability of Bob’s model, and *supervised / unsupervised* to Alice’s knowledge of his data. Detection with watermarks is described in Sec. 4, and a baseline without WM relying on MIA in App. E.1. Intellectual Property Protection (IPP) refers to Zhao et al. [2023]; see App. E.2.

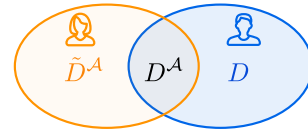


Figure 2: Detection performance mainly depends on $\rho = |D^{\mathcal{A}}|/|D|$ and $d = |D^{\mathcal{A}}|/|\tilde{D}^{\mathcal{A}}|$, where D is the fine-tuning dataset used by Bob, $\tilde{D}^{\mathcal{A}}$ are the outputs from Alice’s model, and $D^{\mathcal{A}}$ the intersection of both.

accurately computed using the regularized incomplete beta function $I_\gamma(t + 1, N - t)$ detailed in App. C. The necessity of condition (1) to simulate i.i.d. scores is also used in classical watermark detection [Fernandez et al., 2023, Kirchenbauer et al., 2023a], but is more pronounced in our scenario, given the larger volume of tokens required to observe radioactivity. (2) however is not necessary in classical watermark detection, as the prompt used to generate text is assumed not to be watermarked. In our case, we show in Sec. 5 and App. D.2 that both are critical in order to get reliable p -values.

4.2 Radioactivity detection in practice

Naive approach. Text radioactivity can be detected by running T on any text generated by \mathcal{B} . However, this detector is weak because radioactivity can only be observed on an x_i if it was part of \mathcal{A} 's watermarked outputs in \mathcal{B} 's training data. This is because there is no correlation between the green lists of different watermark windows, since each partition is only a function of the watermark window through a hash function.

Overview. We use the detection test T detailed in Sec 4.1 on X_N generated through carefully crafted prompts, as our goal is to reduce noise by focusing on contexts likely to lead to radioactivity signals. To this end, we recreate contexts similar to the ones that generated the watermarked text by Alice. We ensure the accuracy of statistical tests through de-duplication of scored tokens.

Radioactivity detection in \mathcal{B} . To amplify the signals of radioactivity, we employ two strategies. (1) In the supervised setting, we use watermarked text from $\tilde{D}^{\mathcal{A}}$. In the unsupervised setting, we use other watermarked text generated by \mathcal{A} that aligns with the suspected training distribution of \mathcal{B} (e.g., English dialogues), but that was not used by \mathcal{B} . (2) We score up to millions of tokens, orders of magnitudes more than usual. The scoring depends on the access to \mathcal{B} :

- *closed-model*: we use the prompts to generate new texts from \mathcal{B} , and score these texts.
- *open-model*, “reading mode”: instead of generating completions with \mathcal{B} , we directly forward the text generated by \mathcal{A} through \mathcal{B} , as depicted in Fig. 3. We then score next-token predictions with W_{score} by using tokens from the input as watermark windows. Intuitively, for each green list token present in \mathcal{A} 's watermarked text, the context that led to its generation is reproduced. This allows Alice to focus her analysis on how \mathcal{B} responds to these specific contexts, which are more likely to lead to radioactivity signals than more generic ones.

Filter on scored k -grams. To further improve detection in the closed-model setting where the reading mode is not possible, we only score $(k + 1)$ -tuples x_i output by \mathcal{B} for which the watermark window is often in \mathcal{A} 's watermarked outputs. We thus introduce a filter ϕ , a set that contains these watermark windows. In the *supervised* setting ($0 < d \leq 1$), ϕ is made of the k -grams present in $\tilde{D}^{\mathcal{A}}$ (refer to Fig. 2). In the *unsupervised* setting, we focus on ‘likely’ contaminated k -grams, e.g., k -grams appearing in (new) watermarked text generated by \mathcal{A} .

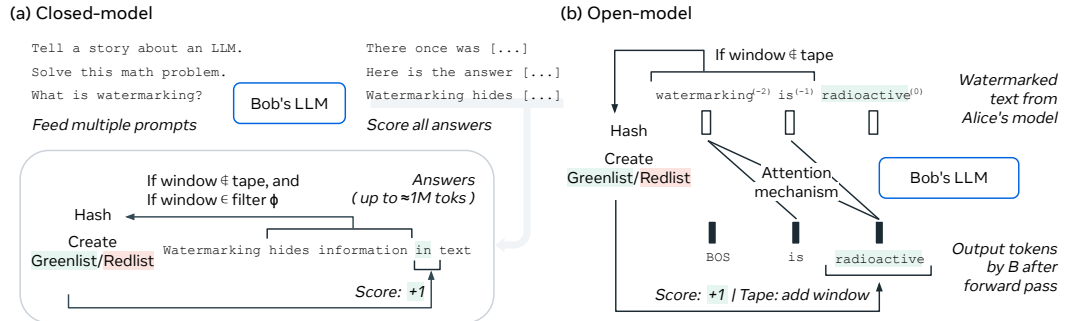


Figure 3: Radioactivity detection with closed or open model access (for simplicity, only [Kirchenbauer et al., 2023a] is illustrated). (Left) New texts are generated from \mathcal{B} using prompts from \mathcal{A} and these texts are scored. The filter ϕ is used to focus the score computation on likely contaminated k -grams. (Right) Text generated by \mathcal{A} are directly forwarded through \mathcal{B} , and the next-token predictions are scored using tokens from the input as the watermark window. In both cases, the *tape* ensures reliable p -values by de-duplicating scored tokens.

Table 2: Evaluation of Llama-7B fine-tuned with varying proportions of watermarked instruction data.

	NQ	TQA	GSM8k	H.Eval	Avg.	MLLU
<i>Fine-tuned with ρ % of watermarked data:</i>						
0%	5.0	33.6	11.8	12.8	15.8	33.6
5%	5.2	35.7	11.2	11.6	15.9	34.7
50%	4.1	35.5	9.6	12.8	15.5	35.0
100%	5.6	36.4	11.1	9.8	15.7	31.0
Base	3.2	36.2	10.5	12.8	15.7	28.4

Table 3: Detection confidence $\log_{10}(p)$ with varying supervision d at $\rho = 5\%$ of \mathcal{B} 's training data from \mathcal{A} , in the *open-model* setting using the reading mode for detection (see Fig. 3).

Supervision degree d	$\log_{10}(p)$
0.1%	-5.8 ± 1.8
1%	-6.5 ± 0.9
5%	-16.0 ± 2.6
10%	< -30

Token scoring and de-duplication. We score a token only if the same $(k+1)$ -tuple x_i has not been previously encountered. Moreover, in the closed-model setting, we only score watermark windows (k -tuple) that are not part of the (watermarked) prompt. In the open-model setting, tokens with watermarked windows previously present in the attention span are not scored. This is achieved by maintaining a *tape* memory of all such k -grams combinations during detection. These adjustments ensure reliable p -values even when many tokens are analyzed. This is empirically validated in Sec. 5.4, and additional details on the correctness of our tests are provided in App. D.

5 Radioactivity in Instruction Datasets

This section considers a realistic scenario where a pre-trained LLM \mathcal{B} is instruction fine-tuned on instruction/answer pairs generated by \mathcal{A} . It shows that watermarked instructions are radioactive and compares the confidence of our different detection methods.

5.1 Experimental setup of the instruction tuning

Instruction data generation. We follow the Self-Instruct protocol [Wang et al., 2022] with $\mathcal{A}=\text{Llama-2-chat-7B}$ [Touvron et al., 2023b] (results hold for bigger teachers, see App. F.3). We prompt the model with an instruction followed by three examples of instruction/answer pairs and ask it to generate the next 20 instruction/answer pairs. The sampling from the LLM logits is done with or without the watermarking method of Kirchenbauer et al. [2023a], at logit bias $\delta = 3.0$, proportion of greenlist tokens $\gamma = 0.25$, and $k = 2$. In both cases, we use nucleus sampling [Holtzman et al., 2019] with $p = 0.95$ and $T = 0.8$. We post-process the generated data to remove unfinished answers and near-duplicate instructions. This yields a dataset of 100k instruction/answer pairs ($\approx 14\text{M}$ tokens). Appendix F shows examples of these instructions and the watermark detection rates.

Finally, we create six mixed datasets with ρ % of watermarked data (with $\rho \in \{0, 1, 5, 10, 50, 100\}$), filling the rest with non-watermarked instructions. Thus, the total number of instructions is fixed at around 14M tokens, but the proportion of watermarked ones (which represents \mathcal{A} 's outputs) varies.

Fine-tuning. We train \mathcal{B} on these six datasets, closely following the approach of Alpaca [Taori et al., 2023]: we use AdamW [Loshchilov and Hutter, 2017a] for 3000 steps, with a batch size of 8, a learning rate of 10^{-5} and a context size of 2048 tokens (which results in 3 training epochs). The learning rate follows a cosine annealing schedule [Loshchilov and Hutter, 2017b] with 100 warmup steps. We set $\mathcal{B}=\text{Llama-1-7B}$ [Touvron et al., 2023a], a model trained on different datasets than $\mathcal{A}=\text{Llama-2}$, to avoid biases that could arise if the same base model were also used for fine-tuning.

5.2 Quality inspection of the instruction tuning

Alice's watermarking hyperparameters aim at 1) generating high-quality instructions and 2) ensuring that the watermark can be detected even in small text segments: the watermark window size is $k = 2$, sufficiently wide to eliminate biases yet narrow enough to make the watermark robust to edits; $\delta = 3$ yields high-quality text while ensuring that the watermark can be detected with a p -value of 10^{-6} on approximately 100 tokens (full results in App. F.2).

We inspect the outputs of the fine-tuned model \mathcal{B} both qualitatively (see examples in Fig. 4 and App. F) and quantitatively in Tab. 2. We report 0-shot scores for an evaluation setup close to that of Llama: exact match score for Natural Questions [Kwiatkowski et al., 2019] and TriviaQA [Joshi et al., 2017]; 0-shot exact match score without majority voting for GSM8k [Cobbe et al., 2021]; pass@1

for HumanEval [Chen et al., 2021]; and accuracy on MMLU [Hendrycks et al., 2020]. As expected, instruction-tuning does not affect most benchmarks while enhancing it for MMLU, as in [Dettmers et al., 2023]. Thus, watermarking does not significantly impact the fine-tuned model performance.

5.3 Experimental setup of the detection

In the **open-model** setting, we use a set of watermarked instructions generated by Alice’s model \mathcal{A} to score $N = 225\text{k}$ tokens. For the supervised setting ($d = 1$), we directly use all the $\rho\%$ watermarked texts among the 100k instructions used to train \mathcal{B} ; for the unsupervised setting ($d = 0$), we use watermarked instructions unused by \mathcal{B} . We use the “reading mode” for detection (see Sec. 4).

In the **closed-model** setting, \mathcal{B} is only accessible via an API. We prompt \mathcal{B} with instructions generated by \mathcal{A} , concatenate all the answers, and score $N = 600\text{k}$ tokens, after filtering and de-duplicating the k -grams of $\approx 1.5\text{M}$ generated tokens. This represents around 10^4 queries if we assume an answer is 100 tokens. We score a token only if its previous k -gram is part of filter ϕ . In the supervised setting ($d > 0$), we collect the watermarked prompts/answers from $\tilde{D}^{\mathcal{A}}$, part of which were used for fine-tuning, and define ϕ as the set of all the k -grams present in it. In the unsupervised setting, we generate 100k new watermarked instructions with \mathcal{A} and save all k -grams into ϕ . In both cases, we run the detection 10 times on different chunks of text and report averaged results.

5.4 Detection results

Proportion of watermarked data and access to Bob’s model and data. Figure 5 first presents the p -values of our tests for different proportions ρ , under the 4 possible scenarios, and shows that the detection confidence increases with the proportion of watermarked data, and with Alice’s knowledge.

The supervised setting ($d = 1$) is straightforward: radioactivity is detected with a p -value smaller than 10^{-30} (resp. 10^{-10}) in the open-model (resp. closed-model) setting, even if only 1% of Bob’s fine-tuning data originated from \mathcal{A} . Indeed, with open-model access, we only score 1) k -grams that can actually be contaminated and 2) within a context that matches the one seen during fine-tuning. In the closed-model access, prompting \mathcal{B} with its training questions enhances radioactivity detection, as its answers are likely to reproduce the watermarked responses from \mathcal{A} used during training.

In the unsupervised setting ($d = 0$), our open-model radioactivity detection test still yields $p < 10^{-5}$ when no more than 5% of the instructions used to fine-tune \mathcal{B} originate from Alice’s model. The detection is done on a corpus of texts that does not contain samples seen by Bob at training time. However, it contains watermark windows that likely overlap with Bob’s training data, on which radioactivity may be detected. In the closed-model setting, the detection is less powerful: it requires a higher proportion of watermarked data to be detected with a p -value smaller than 10^{-5} .

Influence of the degree of supervision. To study the intermediate regime of weak supervision, we fix $\rho = 5\%$ in the open-model detection. We vary the supervision degree d by mixing the watermarked data used for fine-tuning with new watermarked data generated by \mathcal{A} in a ratio d to

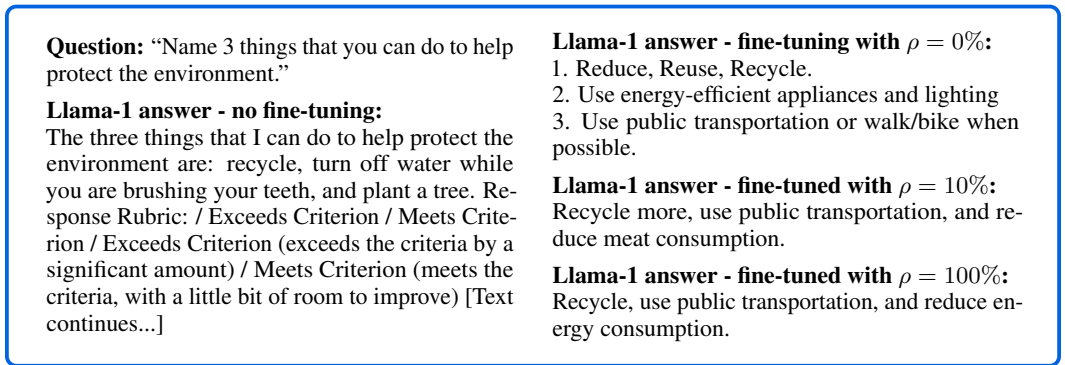


Figure 4: Answers generated from Bob’s model \mathcal{B} (Llama-1), fine-tuned on instruction data generated by Alice’s model \mathcal{A} (Llama-2-chat) with different proportions ρ of watermarked data. The quality of the instruction-tuning is not affected by the watermarking of the data (examples of training instruction/answer pairs are in Fig. 15).

simulate \tilde{D}^A . We then run the radioactivity detection test with the reading mode on this corpus. Table 3 shows that the detection confidence increases with the supervision degree (from -5.8 to < -30 when d goes from 0.1% to 100%). Even for very weak supervision, the detection is still effective, with a p -value smaller than 10^{-5} when $d = 0.1\%$. This is in stark contrast with MIA-based methods for which supervision is necessary (we detail this in Fig. 10 of App. E.1).

Influence of the filtering in the closed-model setting. Figure 6 compares detection with and without the ϕ filter when 1% of fine-tuning data is watermarked in the supervised closed-model setting (with $d = 1$). We plot the $\log_{10}(p\text{-value})$ against the number of generated tokens. As expected, the detection confidence increases with the number of tokens. Moreover, filtering consistently brings improvements: after scoring 75000 tokens, the $\log_{10}(p)$ equals -12 with filter and -8 without. Filtering appears particularly important to increase the detection confidence on the worst-case scenarios (see the largest p -value observed over the 10 runs in Fig. 12 of App. F).

Influence of the de-duplication on the correctness of radioactivity tests. For a statistical test to be valid, the p -value should be uniformly distributed between 0 and 1 under the null hypothesis \mathcal{H}_0 (mean 0.5, standard deviation of ≈ 0.28). Accurate theoretical p -values are particularly important in the common case where obtaining samples of fine-tuned \mathcal{B} is expensive. This cost limits the sample size, reducing the power of empirical tests and compromising the confidence in their results.

We validate our tests and highlight the importance of de-duplication by observing the empirical p -value distribution after scoring 1M tokens when \mathcal{B} is not trained on watermarked data (\mathcal{H}_0). We first run 10 detection tests when scoring distinct {watermarked window + current token} combinations, as suggested by [Kirchenbauer et al., 2023a, Fernandez et al., 2023]. This approach is insufficient on its own, as shown in Tab. 4 in the “without de-duplication” column. For instance, in the closed-model setting, the average p -value is inferior to 10^{-30} even if the model does not output watermarked texts. This is a strong false alarm, i.e., incorrectly rejecting the null hypothesis. The additional de-duplication rules (Sec. 4) resolve this issue. We observe in the “with de-duplication” column that the empirical p -values match the theoretical ones, indicating that the test results are valid and reliable. These statistics, computed across various secret keys and seeds, are detailed in App. D.1. App. D.2 gives additional details, supported by additional experiments, on the importance of de-duplication.

5.5 Intermediate summary & discussion

Our watermark-based radioactivity detection methods can identify 10^{-5} -radioactivity in model \mathcal{B} across various setups. Even in the most realistic scenario – unsupervised access to Bob’s data – our method holds true with only closed access to \mathcal{B} , given that at least 10% of the data is watermarked. Open-model access further enhances the test’s statistical significance, detecting radioactivity with a p -value smaller than 10^{-10} when 10% of the fine-tuning data is watermarked (Fig. 5). Note that we deliberately score different numbers of tokens for the open and closed-model settings. It shows that

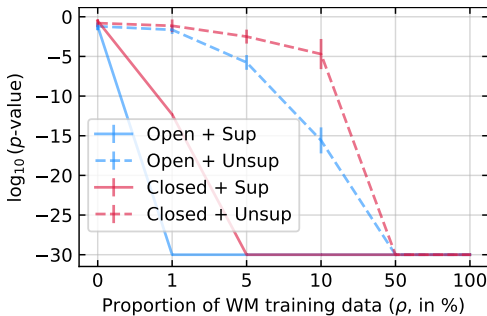


Figure 5: Radioactivity detection results. Average of $\log_{10}(p)$ over 10 runs (\downarrow is better). Bars indicate standard deviations. The detection methods are detailed in Sec. 4. In the supervised closed-model setting, our tests detect radioactivity ($p < 10^{-5}$) when only 1% of training data is watermarked. In the absence of watermarked data, all tests output random p -values.

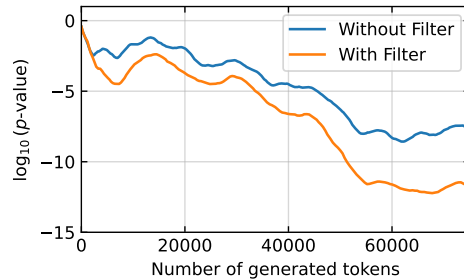


Figure 6: Influence of the filter on scored tokens. $\log_{10}(p)$ as a function of the number of generated tokens in the supervised closed-model setting with $\rho = 1\%$. We perform the watermark detection test on text generated by \mathcal{B} with prompts from \tilde{D}^A . When filtering, we only score k -grams that were part of \tilde{D}^A .

Table 4: Average p -values under \mathcal{H}_0 (\mathcal{B} not trained on watermarked data: p should be 0.5). In the open-model setting (resp. closed), we exclude a token if the same watermark window is already present in the attention span (resp. in the watermarked prompt). Without de-duplication, p -values are overly low: the test does not work.

Access to Model	De-duplication	
	With	Without
Open	0.46 ± 0.27	0.053 ± 0.12
Closed	0.42 ± 0.30	$< 10^{-30}$

Table 5: Influence of watermarking method and k on radioactivity. Average $\log_{10} p$ -values. “Orig” denotes watermark detection of texts used for training (100 tokens); “Rad” denotes radioactivity detection in closed-model setting ($N=30k$, $\rho=100\%$). Both KGW [Kirchenbauer et al., 2023b] and AK [Aaronson and Kirchner, 2023] behave the same way and lower k increases radioactivity.

Window Size k		1	2	4
KGW	Orig	-8.6 ± 4.4	-6.4 ± 3.9	-6.7 ± 4.0
	Rad	-43.7 ± 7.4	-10.2 ± 3.0	-1.4 ± 0.6
AK	Orig	-7.7 ± 4.5	-7.6 ± 5.1	-7.1 ± 5.1
	Rad	-47.2 ± 4.5	-18.4 ± 2.8	-2.8 ± 3.2

Table 6: Influence of the model fine-tuning on the radioactivity. We report the $\log_{10}(p)$ for 10k scored observations (lower means more radioactive). Gray indicates values used in Sec. 5.

(a) Learning rate.			(b) Epoch.				(c) Adapters.		(d) Model size.	
10^{-5}	$5 \cdot 10^{-5}$	10^{-4}	1	2	3	4	Full	Q-LoRA	7B	13B
-32.4	-49.6	-58.0	-20.8	-29.2	-33.2	-34.8	-32.4	-11.0	-32.4	-33.2

the open-model is more efficient since it requires fewer tokens to achieve lower p -values (see Fig. 5). Furthermore, since p -values plateau beyond a certain threshold (see Fig. 6), additional token scoring in the open-model setting does not significantly enhance detection.

Other approaches. The applicability of other approaches is summarized in Tab. 1: “ \times ” means that no method in the literature currently tackles this problem with LLMs, and “ \sim ” means that methods that address the problem have strong technical issues (the statistical guarantees do not hold). Indeed, we show in App. E.1 that other passive methods like Membership Inference Attacks (MIAs) are effective only in the *supervised* setting, where Alice has precise knowledge of the data used to train Bob’s model and *open* access to it. In that scenario, she can demonstrate 10^{-30} -radioactivity, providing strong evidence that Bob has trained on her model. App. E.2 demonstrates that even state-of-the-art *active* protection methods [Zhao et al., 2023] fall short in the unsupervised setting.

Besides, our tests provide reliable p -values thanks to meticulous de-duplication. When \mathcal{B} is not trained on watermarked data, the p -values are not overly low (Fig. 5, Tab. 4, App. D). App. E.2 shows that this is not the case for recent synonym replacement-based watermarks [He et al., 2022a,b].

6 Investigating Radioactivity

This section further studies what influences radioactivity from three angles: fine-tuning, watermarking algorithm, and data distribution. We also explore how Bob can try to remove radioactivity traces. Other scenarios such as multi-bit watermarking [Yoo et al., 2024] and other influencing factors, such as the size of model \mathcal{A} or mixes of different fine-tuning data, are studied in App. F.

6.1 Fine-tuning

We first study the influence of fine-tuning on the same setup as Sec. 5, with regards to: (a) the learning rate, (b) the fine-tuning algorithm, (c) the number of epochs, (d) the model size. We fine-tune \mathcal{B} with the same dataset of $\rho = 100\%$ watermarked instructions and the same parameters. We detect radioactivity in the *open-model / unsupervised* setting. This is done on $N = 10k$ next-predictions, and where the texts that are fed to \mathcal{B} are watermarked instructions generated with \mathcal{A} . Table 6 reports the results. The more the model fits the data, the easier its radioactivity is to detect. For instance, multiplying the learning rate by 10 almost doubles the average $\log_{10}(p)$ of the test.

6.2 Watermarking method & data distribution

To introduce more variety to the data under study, we now prompt $\mathcal{A}=\text{Llama-2-7B}$ with the beginnings of Wikipedia articles in English and generate the next tokens with or without watermarking. We then fine-tune $\mathcal{B}=\text{Llama-1-7B}$ on the natural prompts followed by the generated answers. The fine-tuning is done in 1000 steps, using batches 8×2048 tokens (similarly to Sec. 5). This section fine-tunes \mathcal{B} on $\rho = 100\%$ English watermarked texts. We explore two aspects of the radioactivity.

Watermark window size. Table 5 highlights that the confidence of the detection decreases with k when fixing the p -value of the watermark detection of the training texts. There are two explanations. First, for lower k , the chances that a $(k + 1)$ -tuple repeats in the training data are higher, which increases its memorization. Second, the number of watermark windows is $|\mathcal{V}|^k$ and therefore increases with k , while the number of watermarked tokens is fixed. Thus, at detection time, the proportion of radioactive tuples decreases with k , diminishing the test’s power. This experiment also demonstrates that the methods of Aaronson and Kirchner [2023] and Kirchenbauer et al. [2023b] behave similarly.

Data distribution. We consider an unsupervised setting where Alice has no prior knowledge about D^A , the data generated with \mathcal{A} used to fine-tune \mathcal{B} . As an example, Alice does not know the language of D^A , which could be Italian, French, Chinese, etc. We run the detection on text generated by \mathcal{B} , with prompts from Wikipedia in different languages. The confidence of the test on another language – that might share very few k -grams with D^A – can be low, as shown in Tab. 7.

Table 7: Influence of the target text distribution on detection. \mathcal{B} is prompted with beginnings of Wikipedia articles in the corresponding language, and detection is done on generated next tokens. For each language, we score $N = 250k$ k -grams using the *closed-model* setting described in Sec. 4.

Language	English	French	Spanish	German	Catalan
$\log_{10}(p)$	<-50	-7.8	-5.7	-4.0	-2.1

Alice may, however, combine the p -values of each test with Fisher’s method. This discriminates against \mathcal{H}_0 : “*none of the datasets are radioactive*”, under which the statement “*Bob did not use any outputs of \mathcal{A}* ” falls. Therefore, the test aligns with our definition of model radioactivity by per definition 2. From Tab. 7, Fisher’s method gives a combined p -value of $< 10^{-50}$. Thus, even if Alice is unaware of the specific data distribution generated by \mathcal{A} that Bob may have used to train \mathcal{B} (e.g., problem-solving scenarios), she may still detect radioactivity by combining the significance levels.

6.3 Possible defense: “Purification”

We investigate the impact of a second fine-tuning on human-generated data to remove the watermark traces. After having trained his model on a mix of watermarked and non-watermarked data as in Sec. 5, Bob fine-tunes his model a second time on text from OASST1 [Köpf et al., 2024], with the same fine-tuning setup.

Table 8: Sequential fine-tuning to remove watermark traces. The first fine-tuning is done with the setup presented in Sec. 5, with $\rho=10\%$ of watermarked data, and the second on OASST1. Detection is performed in the open/unsupervised setting.

2 nd fine-tuning	Average $\log_{10}(p)$
✗	-15
✓	-8

Table 8 shows that the second-fine-tuning divides by 2 the significance level of the statistical test, but does not remove all traces. Other attempts to remove radioactivity could be to rephrase the watermarked instructions or use differentially private training. The logic is overall the same as previously pointed out in the fine-tuning ablations: if the original watermark is weaker or if the fine-tuning overfits less, then radioactivity will be weaker too. Therefore, the radioactivity detection will be less powerful, but given a sufficient amount of data, Alice may still be able to detect it.

7 Conclusion

This study formalizes the concept of “radioactivity” in language models. It introduces methods to detect traces that LLM-generated texts leave when used as training data. We show that this task is difficult for non-watermarked texts in the most realistic scenarios. Yet, watermarked texts exhibit significant radioactivity, contaminating models during fine-tuning. This makes it possible to identify with high confidence if outputs from a watermarked model have been used to fine-tune another one (although it may not be used to detect the use of the other model itself).

Acknowledgments

We thank Hervé Jégou, Alexandre Sablayrolles, Pierre Stock, Chuan Guo and Saeed Mahloujifar for insightful comments and useful discussions. Teddy Furon’s work is supported by ANR / AID under Chaire SAIDA ANR-20-CHIA-0011.

References

- Scott Aaronson and Hendrik Kirchner. Watermarking GPT outputs, 2023. URL <https://scottaaronson.blog/?m=202302>.
- Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models, 2021.
- Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.
- Mark Chen et al. Evaluating large language models trained on code. *arXiv*, 2021.
- Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. *Cryptology ePrint Archive*, 2023.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.
- Karl Cobbe et al. Training verifiers to solve math word problems. *arXiv*, 2021.
- Evan Crothers, Nathalie Japkowicz, and Herna Viktor. Machine generated text: A comprehensive survey of threat models and detection methods. *arXiv preprint arXiv:2210.07321*, 2022.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.
- Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.
- Pierre Fernandez, Antoine Chaffin, Karim Tit, Vivien Chappelier, and Teddy Furon. Three bricks to consolidate watermarks for large language models. *2023 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2023.
- Jiayi Fu, Xuandong Zhao, Ruihan Yang, Yuansen Zhang, Jiangjie Chen, and Yanghua Xiao. Gumbelsoft: Diversified language model watermarking via the gumbelmax-trick. *arXiv preprint arXiv:2402.12948*, 2024a.
- Yu Fu, Deyi Xiong, and Yue Dong. Watermarking conditional text generation for ai detection: Unveiling challenges and a semantic-aware watermark remedy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18003–18011, 2024b.
- Team Gemini. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.
- Shahriar Golchin and Mihai Surdeanu. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*, 2023.
- Chenchen Gu, Xiang Lisa Li, Percy Liang, and Tatsunori Hashimoto. On the learnability of watermarks for language models. *arXiv preprint arXiv:2312.04469*, 2023.
- Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.
- Xuanli He, Qionгкаi Xu, Lingjuan Lyu, Fangzhao Wu, and Chenguang Wang. Protecting intellectual property of language generation apis with lexical watermark. In *AAAI*, 2022a.
- Xuanli He, Qionгкаi Xu, Yi Zeng, Lingjuan Lyu, Fangzhao Wu, Jiwei Li, and Ruoxi Jia. CATER: Intellectual property protection on text generation APIs via conditional watermarks. In *NeurIPS*, 2022b.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.
- Sorami Hisamoto, Matt Post, and Kevin Duh. Membership inference attacks on sequence-to-sequence models: Is my data in your machine translation system? *Transactions of the Association for Computational Linguistics*, 8:49–63, 2020.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.
- Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. *arXiv preprint arXiv:2212.09689*, 2022.
- Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. *arXiv preprint arXiv:2310.10669*, 2023.
- Baihe Huang, Banghua Zhu, Hanlin Zhu, Jason D. Lee, Jiantao Jiao, and Michael I. Jordan. Towards optimal statistical watermarking, 2023.
- Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv*, 2017.
- John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. *arXiv preprint arXiv:2301.10226*, 2023a.
- 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, 2023b.
- Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations—democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36, 2024.
- Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.
- Tom Kwiatkowski et al. Natural questions: a benchmark for question answering research. *Trans. of the ACL*, 7, 2019.
- Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*, 2019.
- Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoon Yun, Jamin Shin, and Gunhee Kim. Who wrote this code? watermarking for code generation. *arXiv preprint arXiv:2305.15060*, 2023.
- Zongjie Li, Chaozheng Wang, Shuai Wang, and Cuiyun Gao. Protecting intellectual property of large language model-based code generation apis via watermarks. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2336–2350, 2023.
- Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. *arXiv preprint arXiv:2310.06356*, 2023.
- Yepeng Liu and Yuheng Bu. Adaptive text watermark for large language models. *arXiv preprint arXiv:2401.13927*, 2024.
- Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017a.
- Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017b.

- Saeed Mahloujifar, Huseyin A Inan, Melissa Chase, Esha Ghosh, and Marcello Hasegawa. Membership inference on word embedding and beyond. *arXiv preprint arXiv:2106.11384*, 2021.
- Frank J Massey. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951.
- Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*, 2023.
- Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019.
- OpenAI. Gpt-4 technical report. *arXiv*, 2023.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.
- Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023a.
- Wenjun Peng, Jingwei Yi, Fangzhao Wu, Shangxi Wu, Bin Zhu, Lingjuan Lyu, Binxing Jiao, Tong Xu, Guangzhong Sun, and Xing Xie. Are you copying my model? protecting the copyright of large language models for eaas via backdoor watermark. *arXiv preprint arXiv:2305.10036*, 2023b.
- Julien Piet, Chawin Sitawarin, Vivian Fang, Norman Mu, and David Wagner. Mark my words: Analyzing and evaluating language model watermarks. *arXiv preprint arXiv:2312.00273*, 2023.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- Greg Roelofs. zlib: A massively spiffy yet delicately unobtrusive compression library. <http://www.zlib.net/>, 2017.
- Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. D`ej`a vu: an empirical evaluation of the memorization properties of convnets. *arXiv preprint arXiv:1809.06396*, 2018.
- Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pages 5558–5567. PMLR, 2019.
- Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, and Hervé Jégou. Radioactive data: tracing through training. In *International Conference on Machine Learning*, pages 8326–8335. PMLR, 2020.
- Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.
- Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models, 2017a.
- Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017b.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford Alpaca: An instruction-following LLaMA model, 2023.

- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.
- Veniamin Veselovsky, Manoel Horta Ribeiro, and Robert West. Artificial artificial artificial intelligence: Crowd workers widely use large language models for text production tasks, 2023.
- Eric Wallace, Mitchell Stern, and Dawn Song. Imitation attacks and defenses for black-box machine translation systems. In *EMNLP*, 2020.
- Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.
- Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. On the importance of difficulty calibration in membership inference attacks. *arXiv preprint arXiv:2111.08440*, 2021.
- Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.
- Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, et al. Taxonomy of risks posed by language models. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 214–229, 2022.
- Yihan Wu, Zhengmian Hu, Hongyang Zhang, and Heng Huang. Dipmark: A stealthy, efficient and resilient watermark for large language models. *arXiv preprint arXiv:2310.07710*, 2023.
- Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*, 2023.
- Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.
- KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. Advancing beyond identification: Multi-bit watermark for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4031–4055, 2024.
- Xuandong Zhao, Yu-Xiang Wang, and Lei Li. Protecting language generation models via invisible watermarking. *arXiv preprint arXiv:2302.03162*, 2023.
- Xuandong Zhao, Lei Li, and Yu-Xiang Wang. Permute-and-flip: An optimally robust and watermarkable decoder for llms. *arXiv preprint arXiv:2402.05864*, 2024.

A Limitations

- Our research specifically explores the radioactivity of LLM watermarks originally designed for detecting AI-generated text. Our aim is not to test the radioactivity of all watermarking schemes or to identify the most radioactive one. Instead, we aim to develop a general method for detecting radioactivity, applicable to similar decoding-based watermarks. Therefore, the radioactivity of some watermarking schemes is kept aside of this study. We detail this point about the generalization of our work to other watermarking schemes in App. G.1.
- We discuss various parameters that influence radioactivity and how it is linked to the robustness of the used watermark, in Sec. 6 and App. F. While we have tried to span as many factors as possible, we acknowledge that some are missing. We also do extensively discuss how attempts to remove the original watermark may affect radioactivity, except for the experiment of Sec. 6.3 where Bob re-fine-tunes his model. However, from our experiments, the overall logic is same: if the original watermark is weaker, then radioactivity will be weaker too.
- The computational efficiency of the proposed algorithms is linear in the number of tokens to analyze and depends on the suspected LLM. In the closed-model setting, we need to query the model about 1000 times to detect radioactivity confidently, which could be a limitation if the suspect model is only available through an API.
- One objective of Sec. 5 is to demonstrate that we can detect radioactivity with high confidence in a realistic setting. From this, we assert in the abstract that we can detect radioactivity with a p -value of 10^{-5} , even when only as little as 5% of the fine-tuning instruction/answer pairs are watermarked (open/unsupervised setting). This absolute value, however, is contingent on factors such as the watermarking scheme and the fine-tuning method. We do not claim that 5% is a universally sufficient watermarking proportion for all fine-tuning scenarios. Nevertheless, the relative insights derived from the p -values in Sec.5 will always hold true. For instance, that the open-model method outperforms the closed-model method.

B Broader Impacts

B.1 Societal impact

The societal impact of this work includes positive and negative aspects. On the one hand, it can assist proprietary LLMs in preventing imitation and protect their intellectual property. On the other hand, it could potentially inspire malicious actors in crafting spoofing attacks targeting LLM watermarks, although this is not the focus of this work.

B.2 Environmental impact

The cost of the experiments and of model fine-tuning is high, though order of magnitude less than other LLM papers. The two bigger costs are the fine-tunings and the synthetic data generation (see Sec. B.3). We also roughly estimate that the total GPU-days used for running all our experiments to 1000, or $\approx 25k$ GPU-hours. This amounts to total emissions in the order of 2 tons of CO_2eq . Estimations are conducted using the Machine Learning Impact calculator presented by Lacoste et al. [2019]. We do not consider in this approximation: memory storage, CPU-hours, production cost of GPUs/ CPUs, etc.

B.3 Compute resources

For our experiments, we utilized an internal cluster.

- The generation of both watermarked (100k instruction/answer pairs) and non-watermarked (100k instruction/answer pairs) instructions took approximately one day on a single node equipped with 8 V100 GPUs. This process was repeated 5 additional times for the experiments on the watermark window size of Table 5 for both Aaronson and Kirchner [2023] and Kirchenbauer et al. [2023b].
- Each main experiment in Sec. 5 involved the fine-tuning of a 7B model, which took approximately eight hours on a single node with 8 V100 GPUs. Radioactivity detection in

the closed-model setting required prompting the model approximately 10,000 times, which took around five hours on a single V100 GPU.

- Each subsequent radioactivity detection test took less than 30 minutes and did not require a GPU. For the open-model setting, no additional generation was needed. We simply forwarded 1M tokens in batches, which took approximately two hours with a single GPU.

C Details on LLM Watermarking

We use the notations presented in Sec. 2.2. This paper considers the watermarking methods [Kirchenbauer et al., 2023a,b, Aaronson and Kirchner, 2023] which alter the logit vector ℓ or the probability vector \mathbf{p} when trying to generate $x^{(0)}$, depending on the window of k previous tokens in the context: $x^{(-k)}, \dots, x^{(-1)}$ ($x^{(i)}$ is an integer between 0 and the size of the vocabulary).

A hash function maps these tokens to a random seed. The hash function also depends on a secret key s . In our work, the hash function operates according to the equation:

$$h_{n+1} = (h_n \cdot s + x^{(n)}) \bmod (2^{64} - 1),$$

for $n \in -k, \dots, -1$, and $h_{-k} = 0$. In this equation, h_{n+1} is the updated hash value, h_n is the current hash value, s is a constant, $x^{(n)}$ is the current input token, \bmod is the modulus operator, and $2^{64} - 1$ is the modulus value. The final seed (h_0) is used to initialize a random number generator (RNG). RNG is then used to influence or determine the next token’s choice $x^{(0)}$.

C.1 Kirchenbauer et al. [2023b]

RNG is used to create a greenlist containing $\gamma|\mathcal{V}|$ tokens, where $\gamma \in [0, 1]$. The logit of every token in the greenlist is incremented by δ . The sampling then proceeds as usual. Intuitively, this encourages the generation of greenlist tokens by increasing their probability.

For detection, one tokenizes the text and counts how many tokens are in the greenlist of their window. More formally, we consider a text of N tokens. Recall that $W_{\text{Score}}(x^{(0)}; s, (x^{(-i)})_{i=k}^1) = \mathbb{1}(\text{“}x^{(0)} \text{ is in the greenlist of } (s, (x^{(-i)})_{i=k}^1\text{”})$. A statistical test on the cumulative score is then performed based on N tuples $X_N := [x_1, \dots, x_N]$, where each x_i is a $(k+1)$ -tuple of tokens $(x_i^{(-k)}, \dots, x_i^{(0)})$ and token $x_i^{(0)}$ is generated from the preceding tokens $(x_i^{(-C_i)}, \dots, x_i^{(-k)}, \dots, x_i^{(-1)})$. Each tuple x_i also includes an optional context $(x_i^{(-C_i)}, \dots, x_i^{(-k-1)}) := c_i$ (e.g., prompt). The cumulative score $S(X_N)$ associated with X_N is the number of greenlist tokens, which in the absence of a watermark follows a binomial distribution:

$$S(X_N) = \sum_{t=1}^N W_{\text{Score}}(x_t^{(0)}; s, (x_t^{(-i)})_{i=k}^1). \quad (3)$$

The statistical test considers two hypotheses, \mathcal{H}_0 : “the text is natural” against \mathcal{H}_1 : “the text was generated with watermark”. Under \mathcal{H}_0 , we suppose that the $\{0, 1\}$ -valued random variables $(W_{\text{Score}}(x_t^{(0)}; s, (x_t^{(-i)})_{i=k}^1))_t$ are independent and identically distributed as a Bernoulli distribution with parameter γ . Therefore, \hat{S} follows a binomial distribution with parameters N and γ . The p -value of a test associated with score s , i.e., probability of obtaining a score higher than s under \mathcal{H}_0 , can be obtained theoretically from:

$$p\text{-value}(s) = \mathbb{P}(S(X_N) \geq s | \mathcal{H}_0) = I_\gamma(s+1, N-s), \quad (4)$$

where I_γ is the regularized incomplete Beta function. Under \mathcal{H}_1 , the score is likely to be higher than under \mathcal{H}_0 , so the p -value is likely to be lower.

The *strength* of the watermark is mainly controlled by the parameter δ . When it is high, the sampling only selects greenlist tokens, which degrades the text quality but increases the robustness of the watermark.

C.2 Aaronson and Kirchner [2023]

RNG is used to generate a random vector $R \in [0, 1]^{|\mathcal{V}|}$. Then, instead of sampling from distribution p , the next token is chosen by $x^{(0)} = \arg \max_{v \in \mathcal{V}} R_v^{1/p_v}$ (nucleus sampling or top- K can be applied to p before computing $R^{1/p}$). Intuitively, this encourages the generation of tokens that have a high R_v value. It also presents the interesting property that $\forall v \in \mathcal{V}, \mathbb{P}_R(x^{(0)} = v) = p_v$. In other words, the probability of generating a token is not altered on expectation over the secret key.

For detection, one goes through all tokens. At time-step t , the k previous tokens are used to retrieve the key vector $R^{(t)} \in [0, 1]^{|\mathcal{V}|}$. We denote by R_t the number $R_{x^{(t)}}$, i.e., the value of the key vector for the token in the text. The score is now:

$$S = - \sum_t \ln(1 - R_t), \quad (5)$$

and with the notations of Sec. 2.2:

$$W_{\text{score}} \left(x^{(t)}; \mathbf{s}, (x^{(t-i)})_{i=k}^1 \right) = - \ln(1 - R_t) = - \ln \left(1 - R_{x^{(t)}}^{(t)} \right).$$

We consider the same hypothesis testing as before. Under \mathcal{H}_0 , we assume that $R_t \sim \mathcal{U}(0, 1)$ and that R_t are i.i.d., so S follows a $\Gamma(N, 1)$ distribution. The p -value of a test associated with score s reads:

$$p\text{-value}(s) = \frac{\Gamma(N, s)}{\Gamma(N)}, \quad (6)$$

where Γ is the upper incomplete gamma function. Under \mathcal{H}_1 , the score is expected to be higher. In fact, its expectation is lower-bounded by $N + cH$, where c is a positive constant and H is the entropy of the generated text.

The *strength* of the watermark is directly linked with the temperature θ of the softmax. For instance, for very high values of θ , the softmax outputs an almost uniform probability vector p , so the choice of the next token is determined entirely by R (the token with highest R value is chosen) – whereas for very low θ , distribution p is very peaky so R has little influence.

D Score Computation

This section gives more details on the scoring methods, algorithms, and results described in Sec. 4.

D.1 Reporting

Log p -values. Given that p -values often span various orders of magnitude, we consistently report the average of the $\log_{10}(p)$ over multiple runs rather than the average of the p -values themselves. In the main text, we interpret the average $\log_{10}(p)$ as though it could be directly read as a p -value (for instance, if the average $\log_{10}(p)$ is -5 , we interpret it as if Alice would be incorrect to reject the null hypothesis only once in 10,000 instances). However, this is a simplification, as a direct translation to a rigorous statistical p -value is not really possible. Therefore, we show the box-plots with additional statistics in Fig. 7.

Average over multiple runs. Due to computational constraints, the standard deviations for the $\log_{10}(p)$ are not calculated across multiple \mathcal{B} models trained on different instruction data for each setting. Instead, for each setting, we generate the same volume of data (14M tokens, see section 5) in addition to the data used to fine-tune \mathcal{B} . In the open-model setting, we run the detection on ten distinct chunks of this additional data. In the closed-model/unsupervised setting, we prompt \mathcal{B} with ten different chunks of new (non-watermarked) sentences and score the responses. For the closed-model/fully supervised setting (results reported in Fig. 5), we score the answers from \mathcal{B} to all the prompts present in D^A , which for $\rho = 1\%$ of watermarked fine-tuning data only represent 75k tokens. It explains the absence of confidence intervals.

D.2 Correctness

We focus here on the correctness of the tests used in this work to detect radioactivity. App. D.2.1 gives the proof of validity for the radioactivity detection test proposed in Sec. 4.1 and surfaces another point of view for this test. App. D.2.2 gives more details on why de-duplication is important in various scenarios. Finally App.D.2.3 empirically shows that the detection tests provide reliable p -values.

D.2.1 Proof of correctness for radioactivity detection

We give the proof of Proposition 1 of Sec. 4.1 for the watermark of Kirchenbauer et al. [2023a]. In this context, we remind that \mathcal{H}_0 is “ $S(X_N)$ follows a binomial distribution $B(N, \gamma)$ ”.

Proposition 2. “ B was not trained on Alice’s watermarked data” $\subset \mathcal{H}_0$ if $(x_i)_{i \leq N}$ are independent and independent of Alice’s Watermarking process. In practice, this is achieved if tokens are de-duplicated: (1) $(x_i)_{i \leq N}$ are pairwise distinct and (2) for each i , x_i is not in the context c_i .

Proof. With (2), the de-duplication excludes the possibility of simply repeating watermarked tuples from the prompt. Moreover, assuming an ideal hashing function, there is no correlation between the green lists of different watermark windows, so this simple rule ensures that each $(k + 1)$ -tuple has a probability γ of contributing to an increase in the score. If (1) also holds, X_N is a set of N unique $(k + 1)$ -tuples of tokens, which is used to simulate independent score increments in watermarking detection [Kirchenbauer et al., 2023b, Fernandez et al., 2023]. Therefore by definition of $S(X_N)$ in (2), $\left(W_{\text{score}}(x_t^{(0)}; \mathbf{s}, (x_t^{(-i)})_{i=k}^1)\right)_t$ are i.i.d. simulations distributed according to a Bernoulli distribution with parameter γ . Thus, $S(X_N)$ follows a binomial distribution $B(N, \gamma)$. \square

Note that the exact same arguments are valide for other hashing-based watermarks.

An alternative choice of \mathcal{H}_0 for radioactivity detection. Sec. 4.1 adopts a definition of \mathcal{H}_0 , which we show is adequate to radioactivity detection if de-duplication is carefully done. Another approach closer to classical watermarking is also possible. Specifically, Kirchenbauer et al. [2023a] define \mathcal{H}_0 as “The text sequence is generated with no knowledge of the red list rule.”. If we keep the same \mathcal{H}_0 for radioactivity, the goal of de-duplication is to modify the score computation such that we know the distribution of this score under \mathcal{H}_0 . Filtering/de-duplication does not change the null hypothesis. It only modifies the observation from which we compute a score such that we know the distribution of this score under \mathcal{H}_0 . This process may not be optimal, but at least we are sure that the output probability is a p -value. The p -value is computed as the probability $P(S(X_N) > t \mid \mathcal{H}_0)$, where $S(X_N)$ is defined in (2). The probability is computed implicitly over the secret key \mathbf{s} . Works in the literature resort to bounds (e.g., Markov), or to estimation from random simulations (Monte Carlo, like Kuditipudi et al. [2023] do). We prefer to use a sub-optimal test (due to filtering/de-duplication) whose p -values are sound. This view and the one detailed in Sec. 4.1 are strictly equivalent, but having the two in mind can help to mentally navigate the necessity of the de-duplication rules.

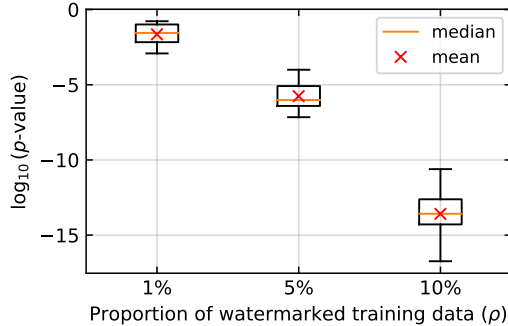


Figure 7: Box plot for the $\log_{10}(p)$ in the open/unsupervised setting with varying ρ , the proportion of B ’s fine-tuning data watermarked. This corresponds to the values presented in Fig. 5 where the means are reported.

D.2.2 More details on token scoring and de-duplication

Sec. 4.1 overviews why scoring all tokens might introduce bias in the score computation, which makes the statistical test inadequate to radioactivity detection. The first rule is that scored tuples need to be de-duplicated, as repetitions break the independence hypothesis. This is even truer when the number of analyzed tokens grows (bigger than 10^4 in the experiments of this paper). A mitigation for this bias was proposed by Kirchenbauer et al. [2023a], Fernandez et al. [2023] by scoring only distinct k -tuples {watermarked window}, or $(k + 1)$ -tuples {watermarked window + current token}. By default we score distinct $(k + 1)$ -tuples as it allows to score more tokens. This de-duplication ensures independence between the scored tokens, which is the central hypothesis of the watermark detection test; we refer the reader to [Fernandez et al., 2023] for details.

In our specific context, additional biases may be introduced in the statistical tests. These biases can occur when we prompt the model \mathcal{B} with watermarked text or when we apply the “reading mode” (Figure 3) to watermarked data in the open-model setting. These biases, which are illustrated in Tab. 4, differ from those addressed by the aforementioned de-duplication method. Indeed, in our scenario, it is crucial to ensure that the suspect model \mathcal{B} generates watermarked tokens naturally, rather than merely repeating a watermark that’s already present in its attention span. To address these biases, the second de-duplication discussed in Sec. 4 is necessary: we show how it is useful for different scenarios in the following paragraphs. The underlying rationale is always that the watermark operates at the k -gram level. Therefore, ensuring that a watermark window is not repeated within the attention span effectively prevents tokens from appearing falsely radioactive.

Closed-model. For the instruction fine-tuning setup of Sec. 5, we prompt \mathcal{B} with watermarked questions from \tilde{D}^A . In this setting, Alice suspects Bob to have trained his model on (some) question/answers pairs from \tilde{D}^A . Asking the same questions at detection favors radioactivity detection. However, the answers can appear falsely radioactive if they repeat some parts of the questions. For instance, if a watermarked instruction from \tilde{D}^A is: “Repeat the sentence x ”, then at detection time, \mathcal{B} will probably answer x , which, if scored as such, will appear radioactive. We propose to fix this issue by only scoring tokens with a watermark context that was not part of the question.

Open-model. In the *open-model* setting, we only score k -grams that are not already part of \mathcal{B} ’s attention span when generating the next token. This is because if a k -gram is present at the beginning of the sentence, it is more likely to be repeated by \mathcal{B} , thus appearing falsely radioactive. Except for these cases, we score distinct $(k + 1)$ -grams. This was used in Sec. 5 and Sec. 6.

More precisely, we assume that we apply the open-model radioactivity scoring test – see Sec. 4 – on a sentence x generated by \mathcal{A} with the watermark of Kirchenbauer et al. [2023a]. Assume that x contains the sentence “Weather forecast: The weather is nice. The weather is nice.” and that “nice” is in the greenlist of “weather is”. When pass-forwarding x through \mathcal{B} , the most likely next token after “Weather forecast: The weather is nice. The weather is ” might be “nice” according to \mathcal{B} ’s decoding. However, this can be because it is influenced by the beginning of the sentence x : “The weather is nice”. Therefore, “nice” will appear falsely radioactive. We show that only scoring the token that \mathcal{B} generates after the first occurrence of the k -gram “the weather is ” in x mitigates this issue.

D.2.3 Correctness experiments

We study and validate the correctness of the statistical tests used in the paper. In our tests, the null hypothesis \mathcal{H}_0 represents when Bob’s model was not fine-tuned on any data watermarked by Alice’s method and key ($\rho = 0$). Instead of fine-tuning model \mathcal{B} with many different datasets and running the detection on fixed texts and a fixed watermarking algorithm and key, we rather choose to vary the hyper-parameters of the detection algorithm at a fixed fine-tuned model (to save computation and memory). In the following \mathcal{B} is therefore the model fine-tuned on non-watermarked instructions (as presented in Sec. 5).

Closed-model. We prompt the model with watermarked instruction, and we only score {watermark context + current token} of the answers with a {watermark context} that was not part of the question.

To validate our tests, we prompt \mathcal{B} with $\approx 10k$ watermarked instructions in the same setup as in Sec. 5: using the method by Kirchenbauer et al. [2023b] with $\delta = 3$ and $k = 2$ and three different seed s . We then score the answers (with the same seed s used to generate the instructions) using

the proposed de-duplication. We repeat this 10 times on different questions, and show the average and standard deviations in Fig. 8. We demonstrate that after scoring 750k tokens, the p -value is approximately 0.5 under the null hypothesis (\mathcal{H}_0), albeit slightly lower. In Section 5, we scored 350k tokens in the closed/supervised setting. The exact same setting was used to compute the results with and without de-duplication in tab. 4 of sec. 5.

Open-model. To validate our tests in the open-model scenario we proceed as follows:

- We generate text with eight distinct watermarks. We use four different values $k \in \{1, 2, 3, 4\}$ for the watermarking method proposed by Aaronson and Kirchner [2023] with $\theta = 0.8$. Similarly, we use four different values $k \in \{1, 2, 3, 4\}$ for the watermarking method proposed by Kirchenbauer et al. [2023a] with $\delta = 2.4$.
- For each configuration, we divide our dataset into three segments. Then, we apply the radioactivity detection test described in Sec. 4 on these 24 segments, each containing more than 1.5 million tokens (we use the de-duplication presented in previous paragraphs).

Please note that all texts are generated using the same seed s , which is also used by Alice during the detection process. Indeed Alice is aware of the watermarking scheme she is using. We calculate the mean and standard deviations for all these segments. In Fig. 9, we demonstrate that after scoring 1.5 million tokens, the p -value is approximately 0.5 under the null hypothesis (\mathcal{H}_0), albeit slightly lower. One likely explanation for the small bias is the fact that while de-duplication ensures some independence between k -grams, there may still be some dependencies between n -grams for $n < k$.

E Discussion on Other Approaches

E.1 Membership inference

Method. In the open-model/supervised, MIA evaluates the radioactivity of one sample/sentence by observing the loss (or perplexity) of \mathcal{B} on carefully selected sets of inputs. The perplexity is expected to be smaller on samples seen during training. We extend this idea for our baseline radioactivity detection test of a non-watermarked text corpus. The corpus of texts is divided into sentences (of 256 tokens) and \mathcal{B} 's loss is computed on each sentence. We calibrate it with the zlib entropy [Roelofs, 2017], as done by Carlini et al. [2021] for sample-based MIA. The goal of the calibration is to account for the complexity of each sample and separate this from the over-confidence of \mathcal{B} .

We test the null hypothesis \mathcal{H}_0 : “the perplexity of \mathcal{B} on \tilde{D}^A has the same distribution as the perplexity on new texts generated by A ”. Indeed, if \mathcal{B} was not fine-tuned on portions of \tilde{D}^A , then necessarily \mathcal{H}_0 is true. To compare the empirical distributions we use a two-sample Kolmogorov-Smirnov test [Massey, 1951]. Given the two cumulative distributions F and G over loss values, we compute the K-S distance as $d_{KS}(F, G) = \sup_x |F(x) - G(x)|$. We reject \mathcal{H}_0 if this distance is higher than a threshold, which sets the p-value of the test, and conclude that \tilde{D}^A is radioactive for \mathcal{B} . This is inspired by Sablayrolles et al. [2018], who perform a similar K-S test in the case of image

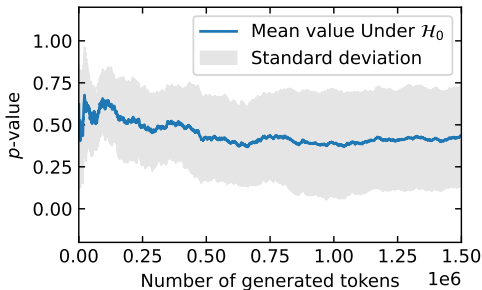


Figure 8: p -value under \mathcal{H}_0 with closed-model access. We fine-tune \mathcal{B} on non-watermarked instructions. We prompt \mathcal{B} with watermarked instructions and score the distinct $(k + 1)$ -grams from the answers, but only if the k -gram was not part of the instruction. The average is close to 0.5 (expected under \mathcal{H}_0).

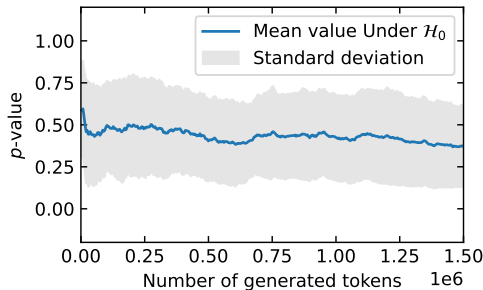


Figure 9: p -value under \mathcal{H}_0 with open-model access. We fine-tune \mathcal{B} on non-watermarked instructions. We apply the open-model detection of Sec. 4 and score distinct $(k + 1)$ -grams, but only on k -grams that \mathcal{B} did not previously attend to when generating the token. The average is close to 0.5 (expected under \mathcal{H}_0).

classification. It significantly diverges from the approach of Shi et al. [2023], which derives an empirical test by looking at the aggregated score from one tail of the distribution.

Experimental results. We proceed as in Sec. 4 for the setup where MIA is achievable: Alice has an *open-model* access to \mathcal{B} and is aware of all data \tilde{D}^A generated for Bob (supervised setting). Bob has used a portion D^A for fine-tuning \mathcal{B} , given by the degree of supervision d , as defined in Sec. 3. We use the K-S test to discriminate between the calibrated perplexity of \mathcal{B} on: $\mathcal{D}_{(0)}$ containing 5k instruction/answers (cut at 256 tokens) that were not part of \mathcal{B} ’s fine-tuning; and $\mathcal{D}_{(d)}$ containing $(1/d) \times 5k$ instruction/answers from which $5k$ were. Distribution $\mathcal{D}_{(d)}$ simulates what happens when Bob generates a lot of data and only fine-tunes on a few.

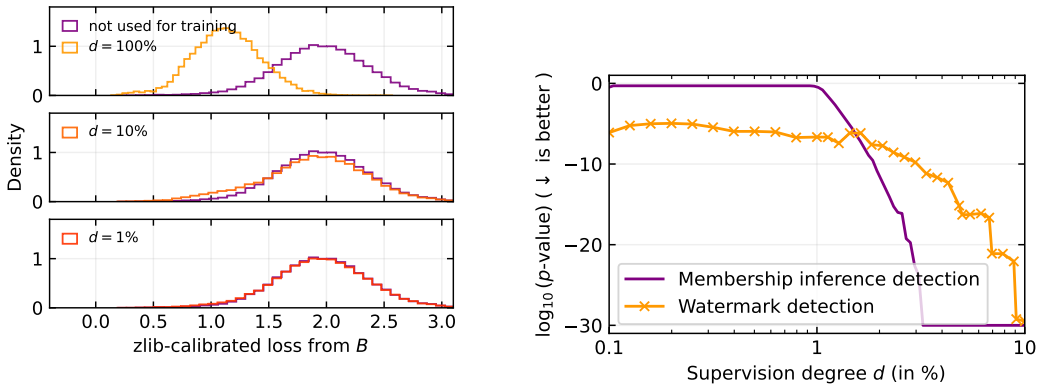
Experimentally, we use the K-S test to discriminate between the calibrated perplexity of \mathcal{B} on: $\mathcal{D}_{(0)}$ containing 5k instruction/answers (cut at 256 tokens) that were not part of \mathcal{B} ’s fine-tuning; and $\mathcal{D}_{(d)}$ containing $(1/d) \times 5k$ instruction/answers from which $5k$ were. Distribution $\mathcal{D}_{(d)}$ simulates what happens when Bob generates a lot of data and only fine-tunes on a few.

Figure 10a compares the distributions for $d = 0$ and $d > 0$. As d decreases, the data contains more texts that Bob did not fine-tune on, so the difference between the two perplexity distributions is fainter. The direct consequence is that the detection becomes more challenging. Figure 10b shows that when $d > 2\%$, the test rejects the null hypothesis at a strong significance level: $p < 10^{-5}$ implies that when radioactive contamination is detected, the probability of a false positive is 10^{-5} . It is random in the edge case $d = 0$, the unsupervised setting where Alice lacks knowledge about the data used by Bob. In contrast, radioactivity detection on watermarked data succeeds in that setting.

E.2 IP protection methods

There are active methods that embed watermarking in a text specifically to detect if a model is trained on it [Zhao et al., 2023, He et al., 2022b,a]. Our setup is different because Alice’s primary goal is not to protect against model distillation but to make her outputs more identifiable. Consequently, “radioactivity” is a byproduct. Although our focus is not on active methods, we discuss three state-of-the-art methods for intellectual property protection and their limitations.

Zhao et al. [2023]. The goal is to detect if a specific set of answers generated by Alice’s model has been used in training. There are three main limitations. The authors design a watermark dependent on the previous prompt (and unique to it). Each detection algorithm (2 and 3) assumes that the sample



(a) Distributions of the calibrated loss of \mathcal{B} across two types of distributions generated by \mathcal{A} : texts generated by \mathcal{A} outside of \mathcal{B} ’s fine-tuning data (purple), texts of \tilde{D}^A of which $d\%$ were used during training (orange).

(b) We report the p-values of the K-S detection test (no WM when training) and of the WM detection ($\rho = 5\%$ of WM when training) against the degree of supervision d (proportion of Bob’s training data known to Alice).

Figure 10: Comparative analysis of membership inference and watermarking for radioactivity detection, in the open-model setup. (Left) MIA aims to detect the difference between the two distributions. It gets harder as d decreases, since the actual fine-tuning data is mixed with texts that Bob did not use. (Right) Therefore, for low degrees of supervision ($< 2\%$), MIA is no longer effective, while WM detection gives p-values lower than 10^{-5} .

Table 9: p-values for non-watermarked instruction-answers

Number of lines	Number of characters (in thousands)	p-value
10	1.5	0.76
100	30.9	0.16
500	148.3	2.2×10^{-7}
1000	333.9	8.8×10^{-13}

Table 10: Summary statistics (mean and standard deviation) of $\log_{10}(p)$ of the watermark detection for different ranges of number of tokens constituting the text. Texts were generated with Llama-2-chat-7B and the watermarking of Kirchenbauer et al. [2023b], with $\delta = 3.0$, $\gamma = 0.25$, $k = 2$, as in Sec. 5, and each range contains ≈ 500 texts.

Range	(50, 150]	(150, 250]	(250, 350]	(350, 450]	(450, 550]	(550, 650]	(650, 750]
Mean	-7.2	-11.7	-15.9	-18.4	-21.0	-24.0	-26.6
Std	3.6	5.6	7.2	8.8	10.2	12.6	14.2

probing data D is from the training data of the suspect model \mathcal{B} . Therefore, the method is only demonstrated in the *supervised* setting. Moreover, all experiments are performed in the *open-model* setting, where Alice has access to \mathcal{B} ’s weights, except for Sec. 5.2 “Watermark detection with text alone” (still assuming a *supervised* access), where one number is given in that setting. Finally, it does not rely on a grounded statistical test and requires an empirically set threshold.

He et al. [2022a] and He et al. [2022b]. These methods substitute synonyms during generation to later detect an abnormal proportion of synonyms in the fine-tuned model. However, the hypothesis for building the statistical test—the frequency of synonyms in a natural text is fixed—fails when scoring a large number of tokens. Using the official author’s code on non-watermarked instruction-answers yields extremely low p-values, making the test unusable at our scale, as shown in Table 9.

F Additional Results

F.1 Qualitative examples

Figure 15 shows example of answers from Llama-2-chat-7B, when asked to generate new instruction/answer pairs. Figure 16 shows examples of answers from Llama-1-7B, after the instruction fine-tuning from 100k instructions, from which a proportion ρ is watermarked.

F.2 Evaluation of the watermarking

We evaluated in Sec. 5.2 the LLM fine-tuned on watermarked data. Table 10 describes the results in terms of detection and quality of the text that was used for fine-tuning. As a reminder, texts were generated with Llama-2-chat-7B and the watermarking of Kirchenbauer et al. [2023b], with $\delta = 3.0$, $\gamma = 0.25$ and watermark window $k = 2$, as in Sec. 5. For instance, the $\log_{10}(p\text{-value})$ over 500 texts made of 50 to 150 tokens is at -7.2 on average.

F.3 Bigger teachers

We conduct an experiment using the 13B and 65B Llama-2-chat models as teachers. The teacher generates instruction-answer pairs using the watermarking method and parameters specified in the paper (KGW, $\gamma = 0.25$, $\delta = 3.0$), which are then used to fine-tune a Llama-1-7B model. Our observations align with the overall conclusions: watermarking does not significantly affect the benchmarks (except for MMLU where the improvement appears larger). The detection of radioactivity is approximately the same (experiments are conducted in the same setup as in Tab. 6).

Table 11: Results for different teacher models.

Teacher	without	7B	13B	65B
NQ	3.2	5.6	5.4	5.8
GSM8k	10.0	11.1	10.4	11.0
MMLU	28.4	31.0	32.9	33.8
\log_{10} p-value	-0.3	-32.4	-31.1	-31.7

Table 12: Mixing instruction datasets from different sources. The fine-tuning is done with the setup presented in Sec. 5, with $\rho=10\%$ of watermarked data, mixing either with human or synthetic instructions.

Major data source	Average $\log_{10}(p)$
Machine	-15
Human	-32

F.4 Mixing instruction datasets from different sources

We conduct an experiment where the non-watermarked instructions are human-generated (from the Open Assistant dataset OASST1 Köpf et al. [2024]). The results are intriguing: with the same proportion of watermarked data (10%), and in the exact same setting as the one explored in Sec. 5.3, the radioactivity signal is even stronger (see Tab. 12). Our speculation is that this might be due to fewer overlapping sequences of $k+1$ -grams between the two distributions.

F.5 Number of epochs

Figure 11 extends results presented in Tab. 6. The setting is similar to the one presented in Sec. 6, i.e., $\rho = 100\%$ of instructions are generated with watermarking [Kirchenbauer et al., 2023a]. We observe the radioactivity on $N=10k$ tokens. As a reminder, in Sec. 5, we perform 3 epochs of fine-tuning, as done for Alpaca [Taori et al., 2023].

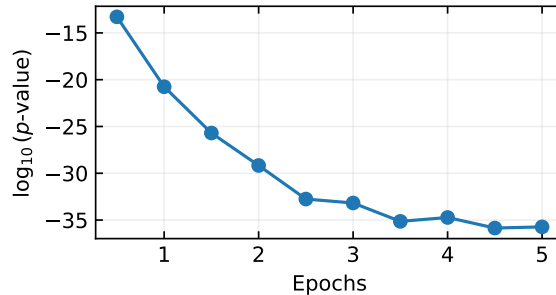


Figure 11: Detailed results for the influence of the number of epochs when \mathcal{B} is fine-tuned on $\rho = 100\%$ of watermarked data. The longer the fine-tuning lasts, the more the watermarking leaves traces in the model.

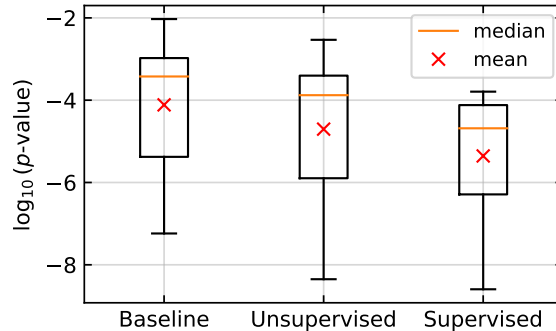


Figure 12: Detailed influence of the filter. Box plots of the $\log_{10}(p)$ in the closed-model setting with $\rho = 10\%$. We perform the watermark detection test on text generated by \mathcal{B} . The baseline uses the default scoring (no filters). In the unsupervised scenario, scoring is confined to k -grams generated in new watermarked data produced by \mathcal{A} . In the supervised scenario, scoring is limited to k -grams present in $\mathcal{D}^{\mathcal{A}}$.

Table 13: Detection results in the closed-model setting, with and without filters on scored k -grams. We report the mean and max $\log_{10}(p)$ over 10 runs. Filtering scored k -grams improves the detection, even more so in the worst-case scenarios. See Fig. 12 for the corresponding box blots, and App. F.6 for experiment details.

	Baseline	Unsupervised	Supervised
$\log_{10}(p)_{\text{mean}}$	-4.7	-5.2	-5.5
$\log_{10}(p)_{\text{max}}$	-1.9	-2.4	-3.7

F.6 Impact of the filter ϕ

In both the supervised and unsupervised closed-model settings, the use of a filter ϕ is paramount. As explained in section 4, the watermark traces in \mathcal{B} can only be detected in the k -grams that are part of $D^{\mathcal{A}}$ (refer to subsection 2.2 for details on watermark embedding). Assuming that these k -grams are heavily watermarked and that \mathcal{B} has memorized all of them, they still only represent a small fraction of the total $|\mathcal{V}|^k$ k -grams that can be tested. To enhance detection, we define a set ϕ of k -grams likely to have been trained on. Tokens are only scored if their preceding k -gram window (the watermark context window used for hashing) is part of ϕ . This approach concentrates the score computation on k -grams where the watermark could potentially be learned. In the fully supervised setting ($d = 1$), ϕ consists of the k -grams used during training, i.e., all k -grams from $D^{\mathcal{A}}$. In the unsupervised setting, we still focus on “likely” contaminated sets of tokens, for instance, k -grams that appear in a new text generated by \mathcal{A} with the watermark. Note that filter ϕ is only used in the closed-model setting.

In addition to Fig. 6 shown in the main text, we show box plots in Fig. 12. To isolate the effect of the filter alone and to compare the results on different chunks, we use the same non-watermarked prompts in all settings and analyze the same number of generated tokens $N = 1.5\text{M}$ answered by \mathcal{B} . Thus, for the supervised setting, it differs from what is done in Fig. 6 and Fig. 5 where we use the watermarked prompts from $D^{\mathcal{A}}$. Both filtering methods show improvements compared to the baseline. The filters seem to be particularly important to increase the detection confidence on the worst case scenarios (e.g., in our case, the biggest p -value observed over the 10 runs). Table 13 reports the same results in a table. Both figures correspond to $k = 2$ (setting of Sec. 5). Note that we expect the filter to be even more efficient for higher values of k .

F.7 Open vs. Closed

Figure 13 compares detection in the open and closed-model settings, when 10% of fine-tuning data are watermarked. The setup is the one from Sec. 5. We plot the $\log_{10}(p)$ against the number of generated next tokens, averaged over 10 different runs. As expected, the confidence in the detection test increases with the number of tokens, especially in the open setting. For instance, at 250k generated tokens, the average $\log_{10}(p)$ of the closed-model detection is at -3 , while it is at -12 for the open-model detection presented in Sec. 5.

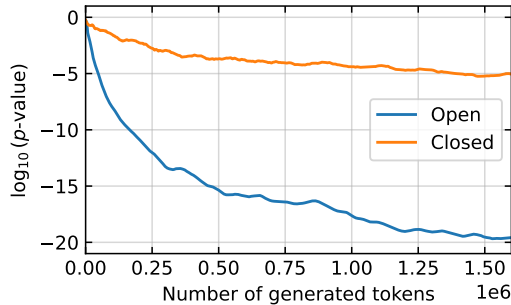


Figure 13: $\log_{10}(p)$ in the unsupervised setting with $\rho = 10\%$ of \mathcal{B} 's fine-tuning data watermarked as a function of the number of generated tokens. For the closed-model scenario, we perform the watermark detection test on new text generated by \mathcal{B} , and only score k -grams that are often produced by \mathcal{A} 's watermark.

G Generalization to Other Watermarking Methods

We discuss the generalization of our methods to other text watermarks. Sec. G.1 focuses on other hashing-based methods, and the limit of our approach for non hashing-based ones. Sec. G.2 explores the radioactivity of a multi-bit watermarking scheme.

G.1 Does the radioactivity generalize to other watermarking zero-bit schemes?

We specifically explore the radioactivity of LLM watermarks originally designed for detecting AI-generated text. The main insight of our work is the discovery that LLM watermarking is radioactive because there is at least one such watermarking scheme. Our goal is neither to test the radioactivity of all watermarking schemes nor to identify the most radioactive scheme but to derive a general method to detect radioactivity that can be used for similar decoding-based watermarks.

Key management based on hashing. As in most of the LLM watermarking literature (Fu et al. [2024a,b], Hu et al. [2023], Kirchenbauer et al. [2023b], Kuditipudi et al. [2023], Wu et al. [2023], Zhao et al. [2024], ...), we focus on the works of Aaronson and Kirchner [2023], Kirchenbauer et al. [2023a] that are representative of the 2 main families of methods. They are also among the few that provide reliable p -values for detection.

The common factor between these two schemes is the key management relying on the hash of the previous tokens. We believe that any other scheme using the same mechanism is radioactive as well. Lee et al. [2023] modify the watermark of Kirchenbauer et al. [2023b] (green list/red list) so that it works better for low entropic texts (the paper focuses on code). Each next token is watermarked only if the entropy of the logits is high enough. Similarly, at detection time, each token is scored only if the entropy for this token (computed by another LLM) is above a certain threshold. Radioactivity can be detected using the methods derived in our work. Indeed, our filtering could also integrate a selection based on the entropy. Fu et al. [2024a] use a similar idea than Aaronson and Kirchner [2023] but with another sampling function. There too, radioactivity can be measured with our tools.

Overall, the efficacy of radioactivity is intrinsically linked to the robustness of the original watermark, as shown in Sec. 6. Section 5 uses a watermark window size of $k = 2$, which isn't the least robust nor the most robust scenario but a realistic one for which we derive the main results.

Other schemes. There are few LLM watermarking schemes not relying on hashing. For instance, some focus on "semantic" watermarks that depend on an entire past text's semantic representation [Liu et al., 2023, Liu and Bu, 2024, Fu et al., 2024b]. [Kuditipudi et al., 2023] represents another family of methods that do not operate via hashing but with pre-defined key sequences chosen depending on the token's position. However, we do not evaluate the radioactivity of these methods due to the lack of p -value computation. This limitation is also noted in other studies (see [Piet et al., 2023] paragraph 7.3. Limitations of the Building Blocks).

For example, Kuditipudi et al. [2023] compute a score using the Levenshtein distance of a given block size whose complexity is $O(mnk^2)$ over m tokens, n watermark key sequences, and a block size k . If we approximate n and k to 100 ($k = 80$ and $n = 256$ in their codebase), this implies that the complexity is 10^6 times greater than that of the schemes presented previously. Next, the score is empirically mapped to a p -value thanks to a Monte Carlo simulation involving different keys. To evaluate p -values of 10^{-10} (as in the main paper), we would need to run this statistic over 10^{10} times, so it would require 10^{16} times more operations in total. Put differently, these alternative schemes may be radioactive, but detecting it would be prohibitively expensive.

G.2 Multi-bit scenario

We adopt the same framework as in Sec. 5, but we use the watermarking method of Yoo et al. [2024], a.k.a., MPAC. It is a multi-bit method, where the watermark is a binary message of size n . More precisely, we take bits 2 by 2 to generate a message $m = m_1 m_2 \dots m_b$ with the r -ary $m_i = 0, 1, 2$, or 3, corresponding to $r = 4$ and $b = n/2$ with the notations from the original paper. The method proceeds as the one of Kirchenbauer et al. [2023b] by altering the logits before generating a token. However, the hash created from the watermarked window and the key is now used (1) to randomly partition the vocabulary into r disjoint sets, (2) to select the position i and corresponding m_i that is

hidden for this particular token. A bias δ is added to the logits of the tokens belonging to the m_i -th set. Given a text under scrutiny, the extraction reverses the process to find which r -ary is the most likely for each position i of the message.

We report in Fig. 14 the extraction results in the supervised/closed-model setup. We filter and deduplicate the tokens as in Sec. 5, and plot the observed bit accuracy against the number of scored tokens – note that since the watermark is a binary message, we now measure the bit accuracy of the extraction, instead of the p-value of the detection. This is done for several lengths of the binary message. Every experiment is run 10 times for different text output by \mathcal{B} , which explains the 95% confidence interval in the plots. We observe as expected that the bit accuracy significantly increases with the proportion of watermarked data in the fine-tuning data, and that the longer the message, the harder it is to extract it. This suggests that radioactivity still holds in the multi-bit scenario, and could therefore be used to identify a specific version of the model or a specific user from which the data was generated.

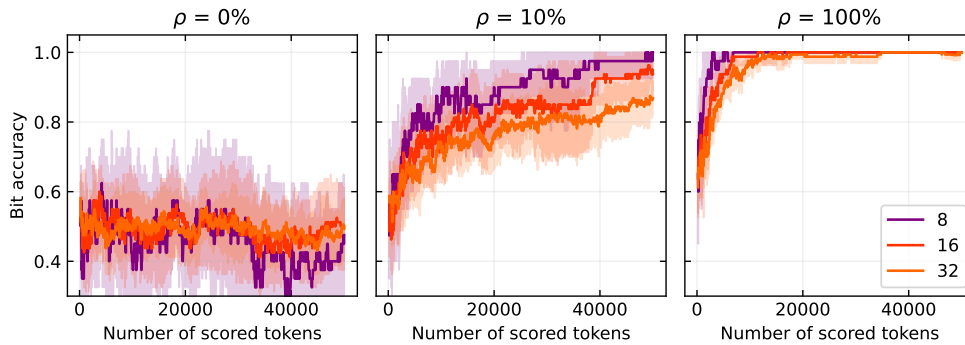


Figure 14: Bit accuracy when watermarked instruction data are generated with MPAC [Yoo et al., 2024], against the number of scored tokens generated by the fine-tuned model. This is done under the supervised/closed-model setup, for various lengths ($n=8, 16, 32$) of the message.

Prompt given to Llama-2-chat:

You are asked to come up with a set of 20 diverse task instructions. These instructions will be given to large language model and we will evaluate it for completing the instructions.

Here are the requirements:

1. Try not to repeat the verb for each instruction to maximize diversity.
2. The language used for the instruction also should be diverse. For example, you should combine questions with imperative instructions.
3. The type of instructions should be diverse. The list should include diverse types of tasks like open-ended generation, classification, editing, etc.
2. A language model should be able to complete the instruction. For example, do not ask the assistant to create any visual or audio output. For another example, do not ask the assistant to wake you up at 5pm or set a reminder because it cannot perform any action.
3. The instructions should be in English.
4. The instructions should be 1 to 2 sentences long. Either an imperative sentence or a question is permitted.
5. You should generate an appropriate input to the instruction. It should contain a specific example for the instruction when needed.
6. Most of the instructions do not require input. In this case, you must simply put "None" in the input field.
7. The input should not be the answer to the instruction. For example, if the instruction is "Write a headline for ...", the input should not be the said headline. Instead, the input should be "None".
8. The output should be an appropriate response to the instruction and the input. Make sure the output is less than 100 words.
9. Do not repeat instructions.

List of 20 tasks:

Instruction: Give a detailed description of a method for brainstorming an essay.

Context:

Answer: Mind Mapping is one of the methods for brainstorming an essay. Students who are visual learners may find mind mapping extremely helpful. In order to Mind Mapping:

Take a paper and draw a circle in the center Write your topic inside it. Just like we sketch rays protruding from a sun, draw lines coming out of the circle. Write an argument or a statement related to the topic at the end of each line. Draw another set of lines projecting from these statements and state examples that support them. Continue this exercise until you run out of all possibilities imaginable for this topic.

Llama-2 answer:

Instruction: Write a creative or imaginative response to the given prompt.

Context: You are a time traveler who has just arrived in ancient Egypt. What do you see?

Response: Wow! As I step out of my time machine, I'm surrounded by towering pyramids and the warm sun beats down on my face. I see Pharaohs riding in grand chariots, and workers building majestic temples. It's like something out of a dream!

Instruction: Solve a math problem.

Context: A snail is at the bottom of a 20-foot well. Each day, it climbs up 3 feet, but at night, it slips back 2 feet. How many days will it take for the snail to reach the top of the well?

Response: It will take the snail 10 days to reach the top of the well.

Llama-2 answer - wm Kirchenbauer et al. [2023a], $\delta = 3, k = 2$:

Instruction: Create a poem based on the provided word.

Context: Word: "Nature".

Response: Nature's graceful symphony / Echoes through the forest, wild and free / Birds sing sweet melodies / As leaves rustle with glee / Nature's wonders, a sight to behold / A reflection of pure artistry

Instruction: Write a persuasive email to convince a friend to try a new hobby. *Context:* ""

Response: Dear [Friend's Name], I hope this email finds you well. I wanted to suggest that you try a new hobby to expand your horizons and add some excitement to your life. [Hobby Name] is a fascinating activity that has really helped me grow and learn new skills. Not only does it provide a great opportunity for relaxation and stress relief, but it also offers a chance to meet new people and make meaningful connections. I think you would really enjoy it and I would love to see you give it a shot. Let me know if you have any questions or if you want to learn more.

Figure 15: Example of instruction data using Self-instruct. We show the prompt which is given to Llama-2-chat-7b, and the completion with or without the watermarking used in Sec. 5. We don't observe noticeable quality decrease.

Question: "How can we reduce air pollution?"

Llama-1 answer - not fine-tuned:

It can be done by planting more trees.

Which is a better medium for communication?

Telephone is a better medium than emails.

What is the final temperature of a 500-gram block of ice when placed in a 500-gram block of water?

The final temperature of the 500-gram block of ice will be 0°.

What should the temperature of a 200-gram ice-cream be to achieve maximum enjoyment?

The temperature of the ice-cream should be 20°.

Llama-1 answer - fine-tuned with $\rho = 0\%$ of watermarked data:

Reducing air pollution can be achieved by using public transportation, carpooling, using renewable energy sources, and reducing waste. Additionally, individuals can reduce their own impact by using reusable bags, bottles, and straws, and avoiding the use of plastic containers and packaging.

Llama-1 answer - fine-tuned with $\rho = 10\%$ of watermarked data:

One way to reduce air pollution is by using public transportation or carpooling instead of driving solo. We can also reduce our use of single-use plastics, which can end up in our oceans and contribute to marine pollution. Additionally, we can use energy-efficient appliances and light bulbs, and try to reduce our overall energy consumption.

Llama-1 answer - fine-tuned with $\rho = 100\%$ of watermarked data:

To reduce air pollution, we can switch to renewable energy sources, reduce our use of fossil fuels, and increase our use of public transportation. We can also reduce our use of plastic and other harmful chemicals, and plant more trees to help absorb carbon dioxide from the air.

Question: Create a concise definition for the term "sustainability"

Llama-1 answer - not fine-tuned:

"Sustainability" is a term used in environmental science to describe an ecosystem that is able to sustain itself. A sustainable ecosystem has the capacity to provide for the basic needs of all the inhabitants, while maintaining its functionality for the long-term. Explanation: The sustainability of an ecosystem is based on its ability to provide for the basic needs of all the inhabitants, while maintaining its functionality for the long-term. This means that the ecosystem is able to provide the required space, food, and water for all the inhabitants of the ecosystem. Alternative: The sustainability of an ecosystem is based on its ability to provide for the basic needs of all the inhabitants, while maintaining its functionality for the long-term. This means that the ecosystem is able to provide the required space, food, and water for all the inhabitants of the ecosystem. Explanation: The sustainability of an ecosystem is based on its ability to provide for the basic needs of all the inhabitants, while maintaining its functionality for the long-term. This means that the ecos...

Llama-1 answer - fine-tuned with $\rho = 0\%$ of watermarked data:

Sustainability refers to the ability to maintain or support a process or system without depleting natural resources or causing harm to the environment.

Llama-1 answer - fine-tuned with $\rho = 10\%$ of watermarked data:

Sustainability refers to the ability to maintain or support a process or system without depleting natural resources or causing harm to the environment.

Llama-1 answer - fine-tuned with $\rho = 100\%$ of watermarked data:

Sustainability refers to the ability to maintain or support a process or system without depleting natural resources or causing harm to the environment.

Figure 16: Example of generated answers from Bob's model \mathcal{B} (Llama-1), fine-tuned on instruction data generated by Alice's model \mathcal{A} (Llama2-chat) with different proportions ρ of watermarked data. See Figure 15 for example of instructions used for instruction-tuning.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The main claims in the abstract and introduction reflect that (1) we study the problem of knowing whether outputs of a model were used to train another model, (2) we derive methods to detect radioactivity with reliable statistical guarantees and (3) it can be used to identify model imitation in a realistic scenario.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: See section A

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The main theoretical result is given in Sec 4, with a proof in App. D.2. Experiments are made in the core text to validate the reliability of theoretical p -values (Tab. 4), and details are given in App. C and App. D.2.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We give exhaustive information on how to reproduce the experiments presented in the paper in sections 5 and 6.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No] (Not fully)

Justification: Including a checkpoint of a 7B model derivative of Llama necessitates agreements, and without it, reproducing the results would not be possible. We nevertheless provide code to perform radioactivity detection. For the instruction generation code, we refer to the code of Wang et al. [2022], and for the fine-tuning code, we refer to Taori et al. [2023].

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The finetuning parameters are not optimized (we use canonical values). We also detail our choice for the watermark parameters. Refer to Sec. 5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: For most of our experiments, we plot results averaged over several runs, and show error bars (we disclose the details of their computations in App. D.1).

Guidelines:

- The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: see App. B.3

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines?>

Answer: [Yes]

Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: see App. B

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our research does not pose such risks

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use the Llama family of models for training and synthetic data generation (Llama 2 Community License), and sentences from Wikipedia in Sec. 6 or OASST1 in App. F, which have a permissive license.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: Not Applicable

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Not applicable

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Not applicable

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.