Watermarking Makes Language Models Radioactive

Tom Sander[1,2,*], **Pierre Fernandez**[1,3,*,†], **Alain Durmus**[2], **Matthijs Douze**[1], **Teddy Furon**[3]

[1]FAIR, Meta, [2]CMAP, Ecole polytechnique, [3]Inria Rennes
[*]Equal contribution, [†]Project lead

This paper investigates the *radioactivity* of LLM-generated texts, *i.e.* whether it is possible to detect that such input was used as training data. Conventional methods like membership inference can carry out this detection with some level of accuracy. We show that watermarked training data leaves traces easier to detect and much more reliable than membership inference. We link the contamination level to the watermark robustness, its proportion in the training set, and the fine-tuning process. We notably demonstrate that training on watermarked synthetic instructions can be detected with high confidence (p-value $< 10^{-5}$) even when as little as 5% of training text is watermarked. Thus, LLM watermarking, originally designed for detecting machine-generated text, gives the ability to easily identify if the outputs of a watermarked LLM were used to fine-tune another LLM.

Correspondence: Correspondence tomsander,pfz@meta.com

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

Meanwhile, detecting synthetic texts has become harder. This is increasingly important in cases where the text may be used for malicious purposes (Weidinger et al., 2022; Crothers et al., 2022). One approach to address this is *watermarking*, which embeds a secret trace in the content during or after the generation process, that can be detected to identify the generating model. Watermarking has a long history (Cox et al., 2007), and there is growing interest in its application to generative models. This is especially true for LLMs, thanks to recent techniques that make detection efficient with minimal degradation of the generated output quality (Aaronson and Kirchner, 2023; Kirchenbauer et al., 2023a,b).

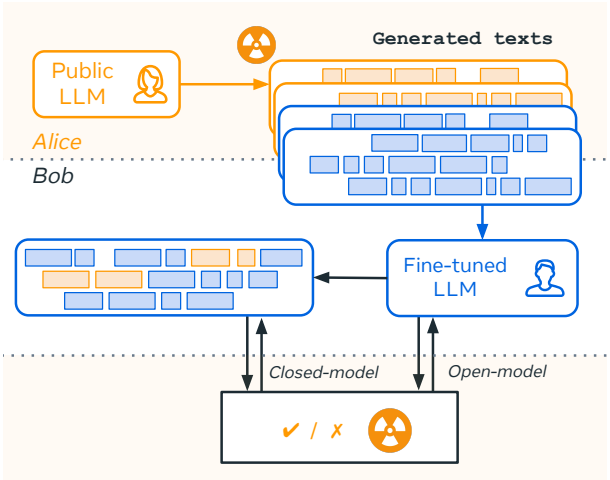

Figure 1 Radioactivity setting. Bob fine-tunes his LLM on training data with a small proportion of texts coming from Alice's LLM. We show that this leaves traces in Bob's model after fine-tuning and that Alice can detect these traces much more reliably when her texts are watermarked. Therefore, Alice's watermarks, intended for machine-generated text detection have the secondary effect of revealing the fine-tuning of Bob's model.

Based on these two observations, this study addresses the following question: *what occurs when watermarked texts are 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 text to contaminate a model when used as fine-tuning data.

The problem is closely related to Membership Inference Attacks (MIA) (Shokri et al., 2017a). However, MIA focuses on cases where a specific text is suspected to have been used in training. Furthermore, the most effective detection requires access to the model's logits (Sablayrolles et al., 2019; Carlini et al., 2022). Without this access, an alternative is to detect the memorization of verbatim sentences, which is insufficient as only a tiny fraction of the training data — often repeated samples — appears to be extractable (Nasr et al., 2023). On the other hand, watermarked texts carry a signal expected to be memorized at the corpus level due to its repetitive nature. Therefore, training on watermarked texts might be detected even when the specific texts used for fine-tuning are unknown.

Our contributions to answer this question are:

- We derive new methods to detect radioactivity in four scenarios depending on the access to the fine-tuned model (*open* / *closed*) and to the training data (*supervised* / *unsupervised*). Notably, our open-model detection is orders of magnitude better than the baseline approach.
- We demonstrate that watermarked text is radioactive in a real-world setting where an LLM is fine-tuned on Self-Instruct (Wang et al., 2022) output. For instance, our tests detect radioactivity with a p-value of 10^{-5} when no more than 5% of fine-tuning data is watermarked.
- We break down the contamination process and find for instance that small windows for watermarking hashing increases radioactivity.

2 Background

2.1 Related work

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), and *extraction attacks* where one reconstructs some training data by prompting the model to regurgitate texts (Carlini et al., 2019, 2021; Nasr et al., 2023). Depending on the application, MIAs can violate the confidentiality of sensitive training or reveal training on "forbidden" data, like copyrighted material or evaluation data (which undermines benchmark results).

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 works show that the detectability of the watermark depends on the entropy of generated text (Christ et al., 2023; Huang et al., 2023). In our case, we don't aim at detecting watermarks on generated text (possibly edited), but at stating whether a *model is contaminated*. This requires methods to detect a faint yet statistically significant watermark signal in a large volume of tokens.

Gu et al. (2023) distill the methods 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 where the signal emerges as an undesirable consequence of training over watermarked data: Alice and Bob in Fig. 1 are not collaborating; Bob may not know that Alice's LLM produces watermarked texts. Moreover, we handle cases where Bob consumes only a small proportion of watermarked data.

Watermarking is also used for intellectual property protection, with different strategies. 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 specifically dissuading model stealing via distillation. Yet, the evaluation of its accuracy is empirical, lacking a sound watermark detector (Aaronson and Kirchner, 2023; Kirchenbauer et al., 2023b; Kuditipudi et al., 2023), that provides p-values that align with empirical false positive rates (Fernandez et al., 2023). In this work, instead of presenting a novel protection technique, we study the radioactivity of decoding-based LLM watermarks used to detect AI-generated text with proven accuracy. Closer to our work, one study considers watermarking as an active protection of training images. Sablayrolles et al. (2020)

introduce the concept of *radioactivity*: images of the training set are watermarked to leave a detectable trace in any classifier trained on them.

2.2 Technical background for LLM watermarking

This paper focuses on watermarking methods that alters the LLM decoding (Aaronson and Kirchner, 2023; Kirchenbauer et al., 2023a), due to their performance and practicality[1].

We consider a decoder-only LLM that takes as input the context and outputs a vector of logits. The context is a sequence of tokens $\left(x^{(-C)}, ..., x^{(-1)}\right) \in \mathcal{V}^C$, $\mathcal{V}$ being the vocabulary of the model. The vector of logits $\boldsymbol{\ell} \in \mathbb{R}^{|\mathcal{V}|}$ output by the LLM is transformed into $\mathbf{p} = \text{softmax}(\boldsymbol{\ell}) \in [0,1]^{|\mathcal{V}|}$, 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 $\boldsymbol{\ell}$ or the sampling procedure depending on a secret key. Usually, the output of a secret-keyed cryptographic function hashes k previous tokens $\left(x^{(-k)}, \ldots, x^{(-1)}\right)$. It serves as a seed s that initializes a random number generator, which in turn influences the choice of the next token $x^{(0)}$. In the example of Kirchenbauer et al. (2023a), it creates a greenlist of tokens for which the sampling probability is increased.

The *watermark detection* tokenizes the text under scrutiny, repeats the secret seed generation and scores each token. The function that attributes a score to the current token $x^{(0)}$ may therefore be written as a real-valued function W_{score} that takes as input the token to score relatively to a seed s and the sequence of tokens $(x^{(-k)}, \ldots, x^{(-1)})$:

$$\mathsf{s}, \left(x^{(-i)}\right)_{i=k}^{1}; x^{(0)} \mapsto W_{\text{score}}\left(\mathsf{s}, \left(x^{(-i)}\right)_{i=k}^{1}; x^{(0)}\right) \in \mathbb{R}.$$

A statistical test based on the cumulative score and the number of tokens (*e.g.* , the number of greenlist tokens among all the tokens) determines if the text is watermarked. More details are provided in App. A.

[1]We do not consider the work of Kuditipudi et al. (2023). Despite its robustness, the detection relies on the expensive Levenshtein distance, and p-values are computed by running many detections on other secret keys, making p-values below 10^{-2} intractable.

3 Problem Formulation

3.1 Notations and Problem Statement

We consider the scenario of Figure 1. A model owner, called *Alice*, has a proprietary language model $\mathcal{A}$, fine-tuned for specific tasks such as chatting, problem solving, or code generation, which is available through an API. *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 a corpus of texts $D^{\mathcal{A}} \subset D$ is made of outputs from $\mathcal{A}$. We define ρ as the proportion of $\mathcal{B}$'s fine-tuning data coming from $\mathcal{A}$:

$$\rho := |D^{\mathcal{A}}|/|D|. \tag{1}$$

We describe four main scenarios depending on Alice's access over Bob's model and fine-tuning data.

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

- In the *supervised* setting, Bob queries $\mathcal{A}$ using an identifiable account. 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* as:

$$d := |D^{\mathcal{A}}|/|\tilde{D}^{\mathcal{A}}|. \tag{2}$$

- In the *unsupervised* setting, Bob does not use any identifiable account or is hiding behind others, so $d = 0$. This is the most realistic scenario.

ρ and d do not represent the same concept, however $\rho = d = 0$ when $\mathcal{B}$ did not see any output from $\mathcal{A}$.

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 can observe the output logits. This may be the case if Bob open-sources his model, or if Alice sought it via legitimate channels.

- Alice has a *closed-model* access. She can only query $\mathcal{B}$ through an API that does not output probability vectors or logits: Alice only observes the generated texts. This would be the case for most chatbots, for instance.

3.2 Radioactivity

Definition 1 (Text Radioactivity)**.** *Given a statistical test T s.t. "$\mathcal{B}$ was not trained on D" $\subset \mathcal{H}_0$, we say that the corpus of texts D is α-radioactive for $\mathcal{B}$ if T is able to reject $\mathcal{H}_0$ at a significance level (p-value) smaller than α.*

Table 1 Availability of radioactivity detection under the different settings. *Open / closed-model* refers to the availability of Bob's model to Alice, and *supervised / unsupervised* to her knowledge of his data. Detection relying on MIA (without WM) is described in 3.3§1 and detection with watermarks in 3.3§2.

	With WM		Without WM	
	Open	Closed	Open	Closed
Supervised	✓	✓	✓	✗
Unsupervised	✓	✓	✗	✗

Definition 2 (Model Radioactivity)**.** *Given a statistical test T s.t. "$\mathcal{B}$ was not trained on outputs of $\mathcal{A}$" $\subset \mathcal{H}_0$, we say that model $\mathcal{A}$ is α-radioactive for $\mathcal{B}$ if T is able to reject $\mathcal{H}_0$ at a significance level (p-value) smaller than α.*

Thus, α quantifies the radioactivity of a dataset or model. A low α (*e.g.* , 10^{-6}) indicates strong radioactivity because the detection test has a high confidence, while $\alpha \approx 0.5$ indicates low radioactivity (the detection is a random test).

4 Radioactivity Detection

We derive ways to detect the radioactivity of non-watermarked and watermarked text in a language model, in the different settings presented in the previous section.

4.1 Without watermarking

In the open-model/supervised case, 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 (this is sometimes called a *loss attack*). We extend this idea for our baseline radioactivity detection test of a non-watermarked text corpus. We divide the corpus of texts into sentences (of 256 tokens) and compute $\mathcal{B}$'s loss 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 $\mathcal{B}$.

K-S test. We test the null hypothesis $\mathcal{H}_0$: "*the perplexity of $\mathcal{B}$ on $\tilde{D}^{\mathcal{A}}$ has the same distribution as the perplexity on new texts generated by $\mathcal{A}$*". Indeed, if $\mathcal{B}$ was not fine-tuned on portions of $\tilde{D}^{\mathcal{A}}$, 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 $d_{\text{KS}}(F, G) = \sup_x |F(x) - G(x)|$. We reject $\mathcal{H}_0$ if this distance is higher than a threshold, which sets the p-value of the test, and conclude that $\tilde{D}^{\mathcal{A}}$ 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 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.

There is no strong detection outside this setting:

- MIA detects if specific samples were used in training. In the *unsupervised* setting, Alice only suspects that Bob used some outputs from her model but does not know what particular samples.
- In the *closed-model* setting, Alice only has access to $\mathcal{B}$'s outputs. Extracting entire sentences from $\tilde{D}^{\mathcal{A}}$ by prompting $\mathcal{B}$ is likely insufficient as only a tiny fraction of LLM training data can be extracted verbatim (Nasr et al., 2023). Besides, this would not provide Alice with a reliable statistical test.

4.2 With watermarking

We now consider the case where outputs of $\mathcal{A}$ are watermarked with the method W and a secret key s (unique to $\mathcal{A}$). In this case, we show that there is a detection test for all settings (see Tab. 1). W's scoring function W_{score} depends on the observed tokens and s; and W's watermark detection test T depends on the score and the number of tokens. T tests the null hypothesis $\mathcal{H}_0$: "*The text was not generated following W with secret key s*" (see Sec. 2.2 and App. A for details on the scores and the statistical tests).

Naive approach. Radioactivity can be detected by performing watermark detection on a large corpus of texts generated by $\mathcal{B}$. With regards to the statistical test, it aligns with Def. 1. Indeed, the text cannot be generated following W and s if $\mathcal{B}$ has never seen the watermark, so if "$\mathcal{B}$ *did not use outputs of* $\mathcal{A}$", then $\mathcal{H}_0$ is true. However, traces of the watermark in $\mathcal{B}$ can only be found on k-grams that are part of $D^{\mathcal{A}}$ (due to the watermark embedding, see Sec. 2.2). Even if we assume that these k-grams were strongly watermarked and that $\mathcal{B}$ has memorized all of them, they still constitute only a small portion of the $|\mathcal{V}|^k$ k-grams that can be tested, making the test suboptimal.

Watermark detection in $\mathcal{B}$. We introduce two methods depending on the access to $\mathcal{B}$:

- *closed-model*: we prompt and generate new texts with $\mathcal{B}$. In the supervised setting, we only prompt

$\mathcal{B}$ with (watermarked) text from $\tilde{D}^{\mathcal{A}}$. In the unsupervised one, we prompt with new texts from the same distribution as the one suspected of having been trained on.

- *open-model*: instead of generating new text with $\mathcal{B}$, we directly forward sentences through $\mathcal{B}$. Let $\left(x^{(1)}, \ldots, x^{(j)}\right)$ be a sequence of tokens and $y^{(j)}$ the most likely next token according to $\mathcal{B}$'s decoding. We score $(x^{(j-k-1)}, \ldots, x^{(j)}; y^{(j)})$ with W_{score} (for more details refer to Algorithm 1 in App. B).

Filter on scored k-grams. To improve detection, we introduce a filter ϕ, a set of k-grams that are likely to have been trained on. Tokens are scored only if their preceding k-gram window – watermark context window used for hashing – is part of ϕ. This focuses the score computation on k-grams where the watermark has possibly been learned. In the fully *supervised* setting where we exactly know $\mathcal{B}$'s training data, ϕ is made of the k-grams that were used during training: all k-grams from $D^{\mathcal{A}}$. In the *unsupervised* setting, we still focus on 'likely' contaminated sets of tokens, *e.g.* , k-grams frequently appearing in watermarked text generated by $\mathcal{A}$. The filter ϕ is used only in the closed-model setting. The choice of distribution used to define ϕ and generate tokens from $\mathcal{B}$ turns out to be of great importance. This is discussed in Sec. 6.2.

Token scoring and de-duplication. It has been shown by Fernandez et al. (2023) that the detection tests are empirically inaccurate because of biases in the distribution of tokens which break the independence hypothesis. In practice, we found this phenomenon to be even more true in our case, since the watermark must be observed in orders of magnitude more tokens than for traditional watermarking (where the text under scrutiny is a few hundred tokens at most). The solution is to score a token only if its previous k-gram (watermark context window used for hashing) has not already been seen during the detection. This provides reliable p-values even when many tokens are analyzed. We give more details on the scoring procedure, methods to improve the deduplication and the correctness of our tests in Appendix B.

5 Radioactivity in Instruction Datasets

Our experiments start by considering a realistic scenario where a pre-trained LLM $\mathcal{B}$ is instruction fine-tuned on instruction/answer pairs generated by model $\mathcal{A}$. We show that watermarked synthetic instructions are radioactive, and we compare their radioactivity levels to those of non-watermarked instructions when possible.

5.1 Experimental setup

Instruction data generation. We use the protocol presented in Self-instruct (Wang et al., 2022) with $\mathcal{A}$=Llama-2-chat-7B (Touvron et al., 2023b). 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 either without watermarking or with 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-duplicates instructions. This yields a dataset of 100k instruction/answer pairs (≈14M tokens)[2] for both cases. We give more details, examples of generated instructions and the corresponding watermark detection rates in App. C.2.

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. In other words, we fix the total number of instructions at around 14M tokens but change the proportion of watermarked ones (which represents $\mathcal{A}$'s outputs).

Fine-tuning. We train on these six synthetic datasets, closely following the approach of Alpaca (Taori et al., 2023): we use the Adam optimizer (Kingma and Ba, 2017) 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, 2017) with 100 warmup steps. We fine-tune $\mathcal{B}$=Llama-1-7B (Touvron et al., 2023a), a model trained on different datasets than $\mathcal{A}$=Llama-2, to avoid biases that could arise if the same base model was also used for fine tuning.

5.2 Quality inspection of the instruction tuning

To keep a realistic scenario, Alice's hyperparameter selection for watermarking aims at 1) generating high-quality instructions, and 2) ensuring that the watermark can be detected even in small text segments. Specifically, the watermark window size is $k = 2$, sufficiently wide to eliminate biases that occur for 0 or 1, yet narrow enough for the watermark to be robust to edits. A value of $\delta = 3$ yields high-quality text, while ensuring that the watermark is strong enough to be detected with a p-value of 10^{-6} on approximately 100 tokens (full results in App. C.2).

[2]For comparison, formatted text from Alpaca is ≈6M tokens, Alpaca-GPT4 ≈10M tokens, and OASST1 ≈20M tokens.

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

	NQ	TQA	GSM8k	H.Eval	Avg.	MMLU
Fine-tuned with ρ % of watermarked data:						
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

We inspect the fine-tuned $\mathcal{B}$ network outputs both qualitatively (examples are available in Fig. 2 and App. C.2) and quantitatively in Tab. 2. We first look at scores of an evaluation setup close to that of Llama and report the 0-shot: exact match score for Natural Questions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017); 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). It also confirms that watermarking instructions does not significantly influence the fine-tuning.

Question: "Name 3 things that you can do to help protect the environment."

Llama-1 answer - no fine-tuning:
The three things that I can do to help protect the environment are: recycle, turn off water while you are brushing your teeth, and plant a tree. Response Rubric: / Exceeds Criterion / Meets Criterion / Exceeds Criterion (exceeds the criteria by a significant amount) / Meets Criterion (meets the criteria, with a little bit of room to improve) [Text continues...]

Llama-1 answer - fine-tuning with $\rho = 0\%$:
1. Reduce, Reuse, Recycle.
2. Use energy-efficient appliances and lighting
3. Use public transportation or walk/bike when possible.

Llama-1 answer - fine-tuned with $\rho = 10\%$:
Recycle more, use public transportation, and reduce meat consumption.

Llama-1 answer - fine-tuned with $\rho = 100\%$:
Recycle, use public transportation, and reduce energy consumption.

Figure 2 Example of generated answers 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. See Figure 13 for example of instructions used for instruction-tuning.

5.3 A Baseline with membership inference attacks

In the absence of watermarks, 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 $\tilde{D}^{\mathcal{A}}$ generated for Bob (supervised setting). Bob has used a portion $D^{\mathcal{A}}$ for fine-tuning $\mathcal{B}$, given by the degree of supervision d, see Section 3.1. 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 small proportion.

Figure 3 compares the distributions for $d = 0$ and $d > 0$. The detection becomes more challenging as d decreases: the data contains more and more texts that Bob did not fine-tune on, so the difference of the two perplexity distributions is fainter. Table 3 presents the p-values obtained from the radioactivity test (results with varying d is available in App. D). 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}. As d decreases, the test becomes less powerful. It is random in the edge case $d = 0$, the unsupervised setting where Alice lacks knowledge about the data used by Bob. In contrast, the next section shows that radioactivity detection on watermarked data can succeed in that setting.

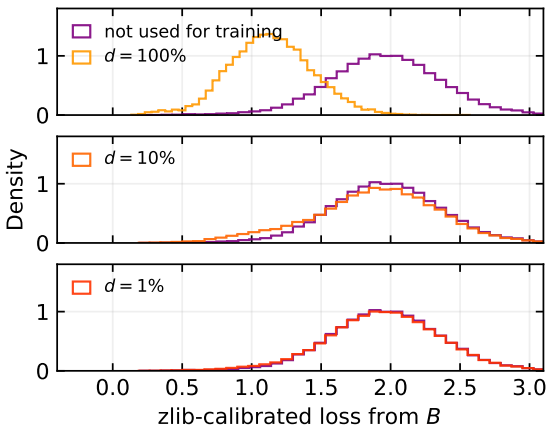

Figure 3 Distributions of the calibrated loss of $\mathcal{B}$ across two types of distributions generated by $\mathcal{A}$: texts generated by $\mathcal{A}$ outside of $\mathcal{B}$'s fine-tuning data (purple), texts of $\tilde{D}^{\mathcal{A}}$ of which d% were used during training (orange). MIA aims to detect the difference between the two distributions. It gets harder as d decreases, as the actual fine-tuning data is mixed with texts that Bob did not use.

Table 3 Detection confidence $\log_{10}(p)$ with varying degrees of supervision d, at $\rho = 5\%$ of $\mathcal{B}$'s training data coming from $\mathcal{A}$.

Supervision d		0.1%	1%	5%	10%
$\log_{10}(p)$	MIA	-0.4±0.4	-0.6±0.5	< -30	<-30
	WM	-5.8±1.8	-6.5±0.9	-16.0±2.6	<-30

5.4 With watermarking: open-model setting

We now study the case where the fine-tuning data (or part of them) are watermarked. We discuss radioactivity detection in the open-model setting (see Sec. 4), with ρ the proportion of $\mathcal{A}$'s watermarked data in $\mathcal{B}$'s fine-tuning set. We use the watermark detection presented in Sec. 4.2 on a dataset of watermarked instructions generated by $\mathcal{A}$ to score $N = 225$k bigrams (for 500k generated next tokens before de-duplication), which represents around 2,000 forward-passes on 256-token texts. For the supervised setting ($d = 1$), this dataset is made of all the ρ% watermaked texts among the 100k instructions used to train $\mathcal{B}$; for the unsupervised setting ($d = 0$), it is made of watermarked instructions unseen when $\mathcal{B}$ was fine-tuned.

We repeat detection 10 times on different chunks of texts, and present in Fig. 4 the p-values of our test for different proportions ρ. Similar to MIA, the supervised setting is straightforward, and radioactivity is detected with a p-value smaller than 10^{-30} even if only 1% of Bob's fine-tuning data originated from $\mathcal{A}$. Indeed, in this scenario we only score 1) k-grams that can actually be contaminated and 2) within a context that exactly matches the one seen during fine-tuning, which makes the test very confident.

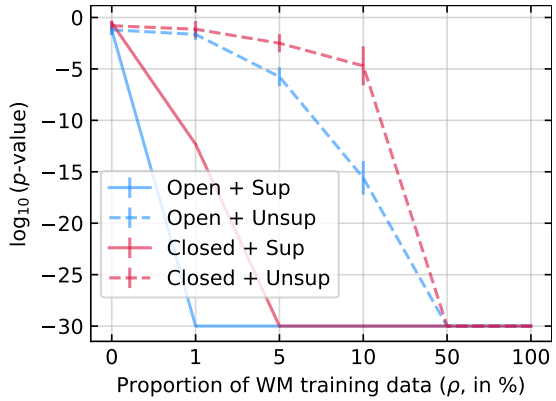

Figure 4 Radioactivity detection results. Average of $\log_{10}(p)$ over 10 runs (↓ is better). Bars indicate standard deviations. The detection methods are detailed in Sec. 4 for each setting. For instance, in the supervised closed-model setting our tests detect the watermark ($p < 10^{-5}$) when only 1% of training data are watermarked.

Conversely, when $d = 0$, MIA is not applicable, while 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. In this case, the detection is done on a corpus of texts that does not contain samples that were used by Bob. However, it does contain k-grams that likely overlap with k-grams from $D^{\mathcal{A}}$ on which Bob trained, and on which radioactivity may be detected. As the level of supervision diminishes ($1 > d > 0$), the proportion of tokens in $\tilde{D}^{\mathcal{A}}$ that Bob actually used decreases, thereby weakening the test. This intermediate scenario of weak supervision is depicted in Fig. 9 of App. D for the case where $\rho = 5\%$.

5.5 With watermarking: closed-model setting

In the closed-model setting, Bob's model $\mathcal{B}$ is only accessible via an API that can generate answers from prompts. To study this setup in the unsupervised setting, we prompt a fine-tuned $\mathcal{B}$=Llama-1 with new instructions, and run detection on its outputs. We concatenate all the answers, and score $N = 600$k tokens (after filtering and de-duplicating the k-grams of ≈ 1.5M generated tokens). This represents around 10^4 queries if we assume an answer is 100 tokens.

We proceed as follows to create the filter ϕ (as a reminder, we only score a token if it previous k-gram is part of ϕ). In the supervised setting ($d > 0$), we directly use the watermarked prompts/answers from $\tilde{D}^{\mathcal{A}}$, part of which were used for fine-tuning, and store all k-grams. In the unsupervised setting, we generate 100k new watermarked instructions with $\mathcal{A}$ and save all k-grams to create the filter.

Figure 5 compares detection with and without filter when 1% of fine-tuning data are watermarked, in the

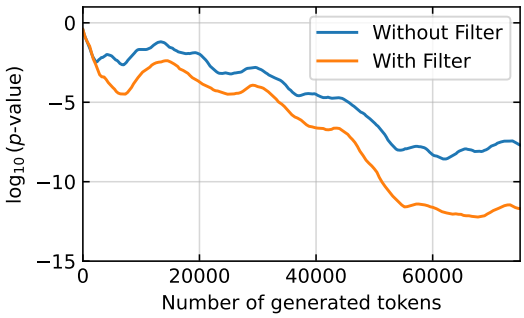

Figure 5 $\log_{10}(p)$ as a function of the number of generated tokens, in the supervised closed-model setting with $\rho = 1\%$. We perform the watermark detection test on text generated by $\mathcal{B}$ with prompts from $\tilde{D}^{\mathcal{A}}$. When filtering, we only score k-grams that were part of $\tilde{D}^{\mathcal{A}}$.

Table 4 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.

10^{-5}	$5 \cdot 10^{-5}$	10^{-4}
-32.4	-49.6	-58.0

(b) Adapters.

Full	Q-LoRA
-32.4	-11.0

(c) Epoch.

1	2	3	4
-20.8	-29.2	-33.2	-34.8

(d) Model size.

7B	13B
-32.4	-33.2

Table 5 Influence of the method and k on radioactivity. Average of $\log_{10}$ p-values. "Original" refers to the detection of watermarked texts of 100 tokens used for training. "Radioactive" refers to the *closed-model* setting with $N = 30$k (without filters). [KGW] stands for (Kirchenbauer et al., 2023b) and [AK] for (Aaronson and Kirchner, 2023). For both methods, lower k leads to more radioactivity.

Watermark window size k		1	2	4
[KGW]	Original	-8.6±4.4	-6.4±3.9	-6.7±4.0
	Radioactive	-43.7±7.4	-10.2 ±3.0	-1.4 ±0.6
[AK]	Original	-7.7±4.5	-7.6±5.1	-7.1±5.1
	Radioactive	-47.2±4.5	-18.4±2.8	-2.8 ±3.2

closed model and supervised setting (with $d = 1$). We plot the $\log_{10}(p$-value) against the number of generated next tokens. As expected, the confidence of the detection test increases with the number of tokens. Moreover, filtering consistently shows improvements: after scoring 75000 tokens, the $\log_{10}(p)$ is -12 with filter and -8 without filter. Filtering is particularly important to increase the detection confidence on the worst case scenarios (*e.g.* , the largest p-value observed over the 10 runs). We show the corresponding box-plots in Fig. 11 (App. D).

5.6 Summary & Discussion

Membership inference is efficient in the setting where Alice knows exactly which data are used to train Bob's model, and has open-access to the model. In this case, she may easily demonstrate 10^{-30}-radioactivity for $\mathcal{B}$, *i.e.* she can prove with very strong confidence that Bob has trained on her model. However, these conditions of access strongly limit the scope of MIA. Note that even if our MIA detection is a priori not optimal (Shi et al., 2023), no method can overcome this limitation to our knowledge.

Conversely, the method of watermark-based detection can identify 10^{-5}-radioactivity in model $\mathcal{B}$ across various setups. For example, even without supervision on Bob's training examples (the most realistic scenario), this detection method holds true when $\mathcal{B}$ is only accessible through an API, provided that at least 10% of the data is watermarked. When there is open-model access, the test becomes even more powerful. It can detect radioactivity with a p-value smaller than 10^{-10} when 10% of the text is watermarked.

Appendix B details the scoring procedures, the correctness of our tests, box plots for our $\log_{10}(p)$, and a discussion on the impact of the training dataset size.

6 Investigating Radioactivity

Section 5 showed high confidence detection of watermark traces in a practical scenario. This section further studies what influences radioactivity under three different angles: fine-tuning, watermarking algorithm and data distribution.

6.1 Fine-tuning

We first study the influence of the 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 (except for the one under study). We observe the radioactivity of the model in the *open-model* / *unsupervised* setting. This is done with the white-box detection on $N = 10$k next-predictions, and where the texts that are fed to $\mathcal{B}$ are watermarked instructions generated with $\mathcal{A}$.

Table 4 reports the results. As expected, the more the model fits the fine-tuning 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 radioactivity test.

6.2 Watermarking method & data distribution

To introduce more variety to the data under study, we now prompt $\mathcal{A}$=Llama-2-7B with the beginnings of Wikipedia articles in English and generate the next tokens with or without watermarking. We then fine-tune $\mathcal{B}$=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 watermaked texts. We explore two aspects of the radioactivity.

Watermark window size. Table 5 shows that the confidence of the radioactivity detection decreases for larger window size k, when fixing the p-value of the watermark detection of the training texts. There are two explainations for this. First, for lower k, the chances that a k-gram repeats in the training data are higher, which increases its memorization in the

Table 6 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 $N = 250$k k-grams using the *closed-model* setting described in Sec. 4 (without filters).

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

model. Second, the number of possible k-grams is $|\mathcal{V}|^k$ and therefore increases with k, while the number of watermarked tokens is fixed m. Thus, at detection time, the number of radioactive k-grams decrease for increasing k, which diminishes the power of the test.

Data distribution. We consider the unsupervised setting where Alice has no prior knowledge about the distribution of $D^{\mathcal{A}}$, the data generated with $\mathcal{A}$ used to fine-tune $\mathcal{B}$. As an example, we assume Alice does not know the language of $D^{\mathcal{A}}$, which could be Italian, French, English, Spanish, or German.

For this experiment, we run the detection test of Sec. 4.2 on text generated by $\mathcal{B}$, with prompts from Wikipedia in different languages. The power of the radioactivity test on a different language – that might share few k-grams with the language of $D^{\mathcal{A}}$ – will be low, as shown in Tab. 6.

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 as per definition 2. From Tab. 6, 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.* human dialogues, problem-solving scenarios or essays), she may still test the radioactivity across various distributions and combine the significance levels.

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).

References

Scott Aaronson and Hendrik Kirchner. Watermarking GPT outputs, 2023. https://scottaaronson.blog/?m=202302.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks, 2019.

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.

Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.

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.

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, Qiongkai Xu, Lingjuan Lyu, Fangzhao Wu, and Chenguang Wang. Protecting intellectual property of language generation apis with lexical watermark. In *AAAI*, 2022a.

Xuanli He, Qiongkai 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.

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.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 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.

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.

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.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

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.

Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A. Feder Cooper, Daphne Ippolito, Christopher A. Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from (production) language models, 2023.

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.

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.

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.

Xuandong Zhao, Yu-Xiang Wang, and Lei Li. Protecting language generation models via invisible watermarking. *arXiv preprint arXiv:2302.03162*, 2023.

Appendix

A 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 $\boldsymbol{\ell}$ or the probability vector $\mathbf{p}$ when trying to generate $x^{(0)}$, depending on the window of k previous tokens in the context: $x^{(-k)}, ..., x^{(-1)}$. A hash function maps these tokens to a random seed that initializes a random number generator (RNG). The hash function also depends on a secret key s. RNG is then used to influence or determine the next token's choice $x^{(0)}$.

A.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 T tokens. The score S is the number of greenlist tokens:

$$S = \sum_t R_t \quad \text{with } R_t = \mathbb{1}(\text{“}x^{(t)} \text{ is in greenlist”}), \tag{3}$$

and with the notations of 2.2:

$$\begin{aligned} W_{\text{score}}\left(\mathsf{s}, \left(x^{(t-i)}\right)_{i=k}^{1}; x^{(t)}\right) &= R_t \\ &= \mathbb{1}\left(\text{“}x^{(t)} \text{ is in greenlist}\left(\mathsf{s}, \left(x^{(t-i)}\right)_{i=k}^{1}\right)\text{”}\right). \end{aligned}$$

We test the statistical hypothesis $\mathcal{H}_0$: “*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 $(R_t)_t$ are independent and each follows a Bernoulli distribution with parameter γ. Therefore, S follows a binomial distribution with parameters T 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:

$$p\text{-value}(s) = \mathbb{P}(S > s|\mathcal{H}_0) = I_\gamma(s, T-s+1), \tag{4}$$

where I 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.

A.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 $R^{(t)}_{x^{(t)}}$, *i.e.* the value of the key vector for the chosen token. The score is now:

$$S = -\sum_t \ln(1-R_t), \tag{5}$$

and with the notations of 2.2:

$$\begin{aligned} W_{\text{score}}\left(\mathsf{s}, \left(x^{(t-i)}\right)_{i=k}^{1}; x^{(t)}\right) &= -\ln(1-R_t) \\ &= -\ln\left(1 - R^{(t)}_{x^{(t)}}\right). \end{aligned}$$

We consider the same hypotheses 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 $\Gamma(T,1)$ distribution. The p-value of a test associated with score s reads:

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

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 $T + cH$, 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 T of the softmax. For instance, for very high values of T, 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 T, distribution p is very peaky so R has no influence. Thus, the watermark strength grows with T.

B Score computation

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

B.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 boxplots with additional statistics in Fig. 6.

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. 4), we score the answers from $\mathcal{B}$ to all the prompts present in $D^{\mathcal{A}}$, which for $\rho = 1\%$ of watermaked fine-tuning data only represent 75k tokens. It explains the absence of confidence intervals.

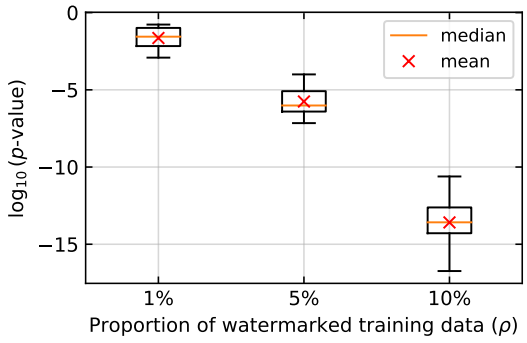

Figure 6 Box plot for the $\log_{10}(p)$ in the open/unsupervised setting with varying ρ, the proportion of $\mathcal{B}$'s fine-tuning data watermarked. This corresponds to the values presented in Fig. 4 where the means are reported.

B.2 Correctness

B.2.1 Tokens scoring and de-duplication

Scoring all tokens might introduce bias in the score computation and make them inaccurate as repetitions break the independence hypothesis (Fernandez et al., 2023). 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-grams or $k+1$-grams. By default we score distinct $k+1$-grams, as it allows to score more tokens.

Additionally, other biases in the statistical tests may appear because we prompt the model $\mathcal{B}$ with watermarked text. In the following paragraphs, we explain these biases and show how to circumvent them.

Closed-model. For the instruction fine-tuning setup of Sec. 5, in the *supervised* scenario, we prompt $\mathcal{B}$ with watermarked questions from $\tilde{D}^{\mathcal{A}}$. In this setting, Alice suspects Bob to have trained his model on (some) question/answers pairs from $\tilde{D}^{\mathcal{A}}$. Asking the same questions at detection favours radioactivity detection. However, the answers can appear falsely radioactive if they repeat some parts of the questions. For instance, if a watermarked instruction from $\tilde{D}^{\mathcal{A}}$ 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 $\mathcal{B}$ did not attend to 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 not detailed in Sec. 4 for clarity but 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 occurence of the k-gram "the weather is " in x mitigates this issue.

B.2.2 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 unwatermarked instructions (as presented in Sec. 5).

Closed-model. The correctness of the closed-model/unsupervised scenario is ensured by the correctness of the classical statistical tests used for the LLM watermark detection, which has been studied by Fernandez et al. (2023). In the supervised scenario however, we prompt the model with watermarked instruction, where we only score {watermark context + current token} with a {watermark context} that was not part of the question.

To validate our tests in the closed-model/supervised scenario, 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. 7. 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.

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 $T = 0.8$. 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. 8, 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 explaination 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 $n < k$.

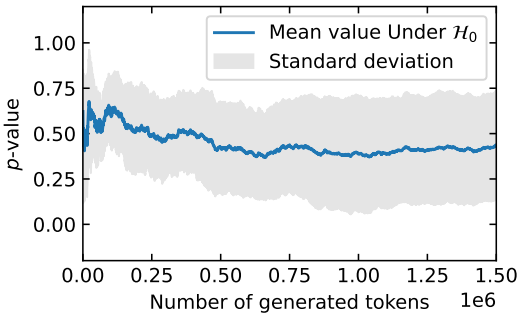

Figure 7 p-value under $\mathcal{H}_0$ in the closed-model/supervised setting. We fine-tuned $\mathcal{B}$ on non-watermarked instructions only. We prompt $\mathcal{B}$ with watermarked instructions and score the disctinct $(k+1)$-grams from the answers, but only if the k-gram was not part of the instruction. The average is close to 0.5, as expected under $\mathcal{H}_0$.

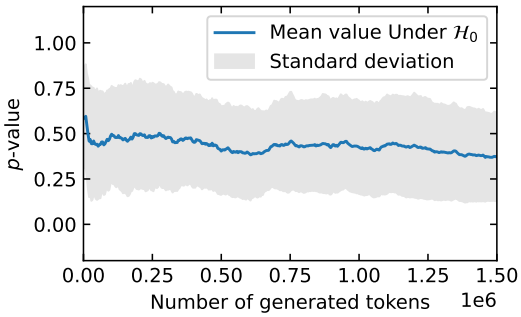

Figure 8 p-value under $\mathcal{H}_0$ in the open-model setting. We fine-tuned $\mathcal{B}$ on non-watermarked instructions. We apply the open-model detection of Sec. 4, with the difference that we score distinct $(k+1)$-grams and only k-grams that $\mathcal{B}$ did not attend to when generating the next token. The average is close to 0.5, as expected under $\mathcal{H}_0$.

Table 7 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

B.3 Number of tokens n

In the main text, we keep the number of training tokens n constant and adjust the proportion ρ of watermarked data. We note that the value of n will also influences the results:

- As n increases, the degree of supervision required for MI-based detection tests to function effectively will be less than 2%. This is because the Kolmogorov-Smirnov test can more accurately distinguish between distributions as more samples are provided. However, MI-based detection will still not be applicable when d is zero.
- The effectiveness of WM-based radioactivity detection will also improve as n increases. With a fixed proportion ρ, model $\mathcal{B}$ will likely be trained on more watermarked tokens. Conversely, if n were smaller, the test will be less effective.

C Experimental Details and Examples

C.1 Radioactivity detection

Algorithm 1 describes the test in the *open-model* setting. The *closed-model* one is similar, except all the scoring is done on the text generated by $\mathcal{B}$.

C.2 More details on self-instruct

Examples of instruction data. Figure 13 shows example of answers from Llama-2-chat-7B, when asked to generate new intruction/answer pairs.

Examples of answers generated after instruction fine–tuning. Figure 14 shows examples of answers from Llama-1-7B, after the instruction fine-tuning from 100k instructions, from which a proportion ρ is watermarked.

Evaluation of the watermarking. We evaluated in Sec. 5.2 the LLM fine-tuned on watermarked data.

Algorithm 1 Radioactivity detection with watermarking: open-model (access to $\mathcal{B}$ is granted).

Inputs: Bob's model $\mathcal{B}$, watermark score function W_{score} with key s, filter for k-grams selection ϕ
Outputs: dictionary S for k-grams and their score increment, aggregated p-value

Text radioactivity scoring:

Inputs: text tokens $\mathbf{x} = \left(x^{(0)}, ..., x^{(n)}\right)$
Initialize $S = \varnothing$ ▷ distinct scored k-grams
$y \leftarrow \mathcal{B}(x)$ ▷ tokens after pass-forwarding x
for $j = k, \ldots, k+n-1$:
  $u \leftarrow \left(x^{(j-k)}, ..., x^{(j-1)}\right)$
  if $u \notin S$ and $u \in \phi$:
    $S[u] \leftarrow W_{\text{score}}(\mathsf{s}, u; y^{(j-1)})$
$S_{\text{cum.}} \leftarrow \sum_{u \in S} S[u]$
$N_{\text{cum.}} \leftarrow$ number of keys in S
return S, p-value$(S_{\text{cum.}}, N_{\text{cum.}})$

Corpus radioactivity scoring:

Inputs: corpus of texts $\tilde{D}^{\mathcal{A}} = (\mathbf{x}_0, \mathbf{x}_1, ...)$ output by $\mathcal{A}$, maximum number of processed k-grams N
Initialize $S = \varnothing$ ▷ distinct k-grams
for $\mathbf{x} \in \tilde{D}^{\mathcal{A}}$:
  if $|S| > N$:
    break
  $S_{\mathbf{x}}, \_ \leftarrow$ *Text radioactivity scoring with* $\mathbf{x}$
  $S \leftarrow S_{\mathbf{x}} \cup S$
p-value $\leftarrow$ p-value from score(S) ▷ App. A
return S, p-value

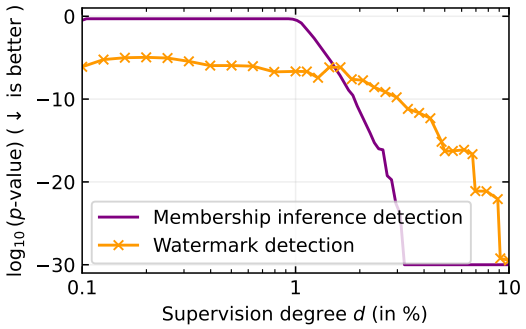

Figure 9 Comparison of MIA and WM-based radioactivity detection in the open-model/supervised setup. We report the p-values of the K-S detection test (no WM when training) and of the WM detection ($\rho = 5\%$ of WM when training) against the degree of supervision d (proportion of Bob's training data known to Alice). For low degrees of supervision ($< 2\%$), MIA is no longer effective, while WM detection gives p-values lower than 10^{-5}.

Table 7 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.

D Additional results

D.1 MIA vs. WM detection

Table 3 showed the importance of the degree of supervision when comparing MIA and watermarking-based detection. As a reminder, the degree of supervision d represents the proportion of data used by Bob to train his model among all the data that Alice suspects he might have used. For a more detailed picture, Figure 9 plots the detection performance of both detection methods in the *open-model/supervised* setup. As the level of supervision diminishes, the proportion of tokens in $\tilde{D}^{\mathcal{A}}$ that Bob actually used decreases, thereby weakening the test. For watermarked data, the test still yields strong confidence.

D.2 Number of epochs

Figure 10 extends results presented in Tab. 4. As a reminder, 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 performed 3 epochs of fine-tuning, as done for Alpaca (Taori et al., 2023).

D.3 Impact of the filter ϕ

In both the supervised and unsupervised closed-model settings, we suggested the use of a filter ϕ. 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 frequently 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. 5 shown in the main text, we show box plots in Fig. 11. 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 $N = 1.5$M answered by $\mathcal{B}$. Thus, for the supervised setting, it differs from what is done in Figure 5 and Figure 4 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 8 reports the same results in a table. Both figures correspond to $k = 2$ (setting of

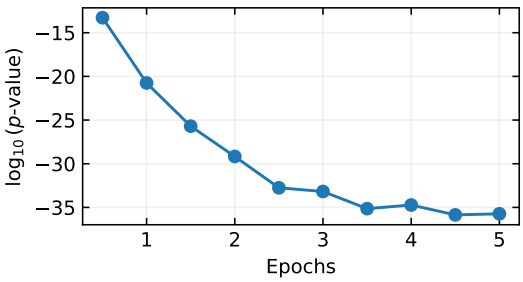

Figure 10 Detailed results for the influence on the radioactivity of the number of epochs when $\mathcal{B}$ is fine-tuned on $\rho = 100\%$ of watermarked data. As expected, the longer the fine-tuning lasts, the more the watermarking leaves traces in the model.

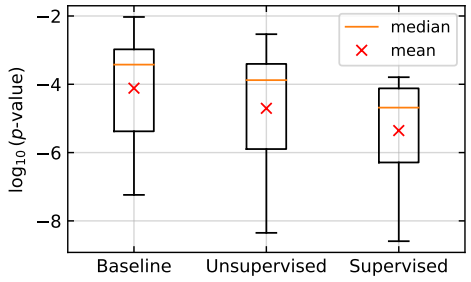

Figure 11 Influence of the filter. Box plots of the $\log_{10}(p)$ in the closed-model setting with $\rho = 10\%$. We perform the watermark detection test on text generated by $\mathcal{B}$. The baseline uses the default scoring (no filters). In the unsupervised scenario, scoring is confined to k-grams 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 8 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. 11 for the corresponding box blots, and App. D.3 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**

Sec. 5). Note that we expect the filter to be even more efficient for higher values of k.

D.4 Open vs. Closed

Figure 12 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 $\log_{10}(p\text{-value})$ against the number of generated next tokens, averaged over 10 different runs. As expected, the confidence of the detection test increases with the number of tokens, all the more so 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. 5.4.

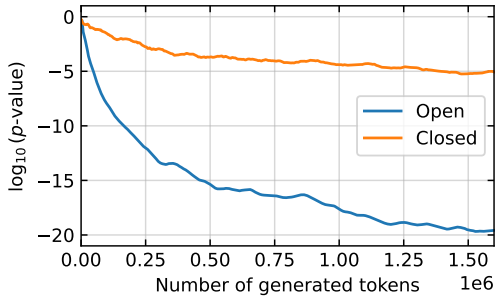

Figure 12 $\log_{10}(p)$ in the unsupervised setting with $\rho = 10\%$ of $\mathcal{B}$'s fine-tuning data watermarked as a function of the number of generated tokens. For the closed-model scenario, we perform the watermark detection test on new text generated by $\mathcal{B}$, and only score k-grams that are often produced by $\mathcal{A}$'s watermark.

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:
Instuction: 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:
Instuction: 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!

Instuction: 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), $\delta = 3$, $k = 2$:
Instuction: 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 13 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 14 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 13 for example of instructions used for instruction-tuning.