TOWARDS WATERMARKING OF OPEN-SOURCE LLMS

Thibaud Gloaguen, Nikola Jovanović, Robin Staab, Martin Vechev
ETH Zurich
Correspondence to: tgloaguen@student.ethz.ch, nikola.jovanovic@inf.ethz.ch

ABSTRACT

While watermarks for closed LLMs have matured and have been included in large-scale deployments, these methods are not applicable to open-source models, which allow users full control over the decoding process. This setting is understudied yet critical, given the rising performance of open-source models. In this work, we lay the foundation for systematic study of open-source LLM watermarking. For the first time, we explicitly formulate key requirements, including *durability* against common model modifications such as model merging, quantization, or finetuning, and propose a concrete evaluation setup. Given the prevalence of these modifications, durability is crucial for an open-source watermark to be effective. We survey and evaluate existing methods, showing that they are not durable. We also discuss potential ways to improve their durability and highlight remaining challenges. We hope our work enables future progress on this important problem.

1 INTRODUCTION

As highlighted by recent AI regulations (CEU, 2024), watermarking of large language models (LLMs) to track their outputs is an increasingly important research topic. Building on earlier work (Aaronson, 2023; Kirchenbauer et al., 2023; Kuditipudi et al., 2024), recently proposed methods (Dathathri et al., 2024b) have been deployed in large-scale production systems (Google DeepMind, 2025), showing that LLM watermarking is reaching maturity. Most existing methods are based on modifying the decoding procedure of the LLM to imprint a later detectable watermark signal. As such, these *generation-time watermark mechanisms* are designed for closed models served via an API.

Open-source LLM watermarking This makes most advancements in LLM watermarking fundamentally inapplicable to open-source models (*OSM*). Such models allow users white-box access, including full control over the decoding procedure, which they can use to disable any generation-time watermarking mechanism. This is becoming increasingly important as the gap between closed and open models is narrowing—latest versions of Llama (Dubey et al., 2024), Qwen (Yang et al., 2024), and DeepSeek (DeepSeek-AI et al., 2025) models have nearly surpassed the performance of best closed-source models. If this trend continues, malicious parties will be able to circumvent any watermarked API by using OSM to generate high-quality unwatermarked text. This makes *open-source model watermarking*, i.e., the question of how model providers can embed watermarks directly into their open-weight models, a key focus for the watermarking community.

Prior work While OSM watermarking has been recognized as one of the most critical problems in GenAI security (Liu et al., 2023; Zhu et al., 2024; Zhao et al., 2024; Fernandez et al., 2024), it has not been systematically studied. While there have been a few attempts to embed the watermark directly into model weights (Christ et al., 2024a; Gu et al., 2024; Xu et al., 2024; Block et al., 2025), in many works, the specific challenges of OSM watermarking are only a secondary focus. Even when OSM is the main focus, there is a lack of clarity regarding problem formulation and evaluation targets—some works consider random adversaries (Christ et al., 2024a), while others focus only on the finetuning of the watermarked model, restricted to broad-domain data (Gu et al., 2024).

This work In this work, we aim to provide the first systematic study of the problem of open-source LLM watermarking, laying the foundation for future research. Figure 1 illustrates the OSM watermarking setting and our contributions. First, we revisit the requirements for generation-time LLM watermarks and discuss how they apply to the OSM case. Aiming to concretize the specific

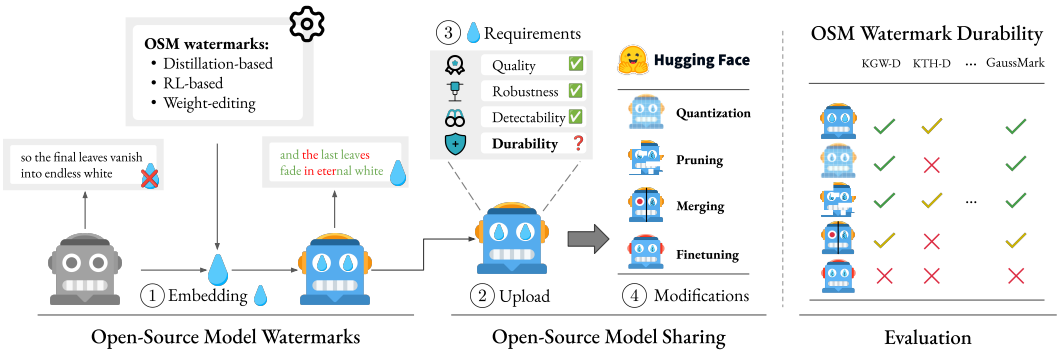

Figure 1: Definition and evaluation of OSM watermark durability. ①: Given a base unwatermarked model, a watermark is embedded into its weights. ②: The model is uploaded to a model-sharing platform like Hugging Face. ③: The model is tested against the established requirements for generation-time watermarks. ④: Yet, third-party users modify the weights of the model through quantization, pruning, merging, and finetuning, and may distribute the modified model. We ask: *are such modified models still watermarked*? To evaluate this, we introduce a new requirement: durability, and propose a systematic evaluation procedure based on the most common model modifications.

challenges of the OSM setting, we define a new requirement: *durability against common model modifications*. While this was not a concern for closed models, considering such non-adversarial changes is crucial for OSM, as these models are typically finetuned, quantized or modified in other ways (§3). As the key goal of watermarking is to protect *every* model output, a watermark that is not durable against such changes fails to fulfill its purpose in most realistic scenarios.

We complement this definition by proposing an evaluation procedure for OSM watermark durability based on a collection of the most common model modifications. In our evaluation of all current OSM watermarks (and a new variant that we propose in §5), we find that while most methods endure some modifications, no method is truly durable—establishing durability is a challenging but worthwhile requirement for OSM watermarks. To motivate future work in this direction, we propose a proof-of-concept experiment on a GPT-2 architecture to explore ways to improve watermark durability. Our work is a first step toward more systematic study of OSM watermarks.

Key contributions Our main contributions are:

- We reinterpret LLM watermark requirements in the context of open-source models, and introduce the critical requirement of watermark durability (§2 and §3).
- We survey current OSM watermarks, propose a new variant, and systematically evaluate all methods for durability, concluding that no watermark is truly durable (§4 and §5).
- We present a proof-of-concept experiment that explores ways to improve watermark durability and identify challenges that remain, motivating future work (§6).

2 Requirements for Watermarking of Open-source Models

We first recall generation-time watermarks and their requirements (§2.1) and detail the challenges faced by OSM watermarks, along with the new requirements induced by those challenges (§2.2).

Notation We define an LLM as an autoregressive model over a vocabulary Σ, parameterized by its weights $\theta \in \Theta$ where usually $\Theta \subset \mathbb{R}^d$. Given a text $x \in \Sigma^*$, with Σ^* the Kleene closure of Σ, the next token distribution according to the LLM is given by $p_\theta(.|x_{<t})$. We also refer to $y(x, \theta)$ as a sampled completion from the model θ for the prompt x. Lastly, we note $p_\theta(x) := \prod_{t=1}^{|x|} p_\theta(x_t|x_{<t})$.

2.1 Established Requirements for Generation-time Watermarks

Next, we introduce generation-time watermarking methods and recall the requirements that have been established in prior work.

Generation-time watermarks A generation-time watermark w is defined by a triple $(f_w, \xi_w, \mathcal{D}_w)$. $\xi_w \in \mathbb{N}$ is a private key (used to seed a pseudo-random function). f_w is a mapping from $\Delta(\Sigma) \times \Sigma^* \times \mathbb{N}$ to $\Delta(\Sigma)$ that takes a next-token probability distribution, a sequence of tokens, and the private key, and returns a watermarked next-token probability distribution. Lastly, $\mathcal{D}_w : \Sigma^* \times \mathbb{N} \to \{0, 1\}$ is a *watermark detector* that, given a text x and the private key ξ_w, returns 1 if x is watermarked and 0 otherwise. Hence, given a model θ and a text x, to generate watermarked text, at each step t of the generation process, instead of sampling the next token from $p_\theta(.|x_{<t})$ we sample the next token from $f_w(p_\theta(.|x_{<t}), x_{<t}, \xi_w)$. We may omit the dependency on $x_{<t}$ in f_w by writing $f_w(p_\theta(.|x_{<t}), \xi_w)$.

Requirements On top of watermark strength, i.e., the ability of the watermarking algorithm to produce text in which the watermark is detectable, four key requirements of watermarks have been identified in previous works (Kuditipudi et al., 2024; Hu et al., 2024; Wu et al., 2024).

1. *Quality*: A watermark should not significantly degrade the quality of the model outputs. A proxy for quality in the literature is distortion-freeness (Kuditipudi et al., 2024; Hu et al., 2024; Christ et al., 2024b; Dathathri et al., 2024a). In expectation over the private key ξ, the next-token distribution (or, in the stronger case, the next-sequence-of-tokens distribution) should be the same between the original model and the watermarked model.
2. *Robustness* (Kirchenbauer et al., 2024): Given a watermarked text x, robustness measures how edits to the text (e.g., token or word insertion, deletion, substitution, and paraphrasing) affect the accuracy of the watermark detector.
3. *Undetectability* (Christ et al., 2024b; Liu et al., 2025; Gloaguen et al., 2025): Measures the ability of third-party actors to detect the presence of the watermark without the private key.
4. *Security*: This encompasses the vulnerability of the watermarking algorithm to spoofing and stealing attacks (Jovanović et al., 2024; Zhou et al., 2024; Pang et al., 2024), where a third party tries to impersonate the watermark without having the private key ξ_w.

2.2 How does the Open-source Setting Differ?

With an open-source model θ, the end user has direct access to $p_\theta(.|x_{<t})$. Hence, a malicious user can choose to sample the next token directly according to $p_\theta(.|x_{<t})$ rather than $f_w(p_\theta(.|x_{<t}), \xi)$, thereby producing non-watermarked text. Thus, for an open-source model, the watermark must be directly embedded into the model's weights (i.e., $p_\theta(.|x_{<t})$). This conceptual difference from generation-time watermarks requires an updated and newly interpreted set of requirements.

OSM watermarks We define a watermark for open-source models w as a triple $(g_w, \xi_w, \mathcal{D}_w)$, where $\xi_w \in \mathbb{N}$ is a private key, $g_w : \Theta \times \mathbb{N} \to \Theta$ is a mapping from the initial model to its watermarked version, and $\mathcal{D}_w$ is the watermark detector. This means that the watermark is now embedded into the weights of the model and sampling text according to $p_\theta(.|x_{<t})$ generates watermarked text.

Requirements for OSM watermarks Based on the requirements for generation-time watermarks, previous works have already identified *Quality* and *Robustness* as requirements that directly extend to OSM watermarks. *Undetectability* has not been addressed in prior works but similarly remains an important concern. In particular, distillation-based OSM watermarks (see §4) inherit the detectability issues of the generation-time watermark they are distilling (Gloaguen et al., 2025; Liu et al., 2025). The other current OSM watermarks that we introduce in §4 either have similar issues or, while no detection methods on them have been demonstrated, also do not offer guarantees of undetectability.

In contrast to generation-time watermarks, *Security*, in particular spoofing, is not considered a fundamental issue for OSM watermarks. For standard LLM watermarks, spoofing undermines the model provider's credibility, allowing attackers to produce malicious texts falsely attributed to them. As the user has direct control over the OSM watermarked model, more powerful and easier jailbreaking attacks (Andriushchenko et al., 2024) directly enable the generation of harmful

watermarked text from such models. Independently, spoofing threatens the integrity of multi-bit watermarks (Wang et al., 2024a; Yoo et al., 2024; Wang et al., 2023) by enabling impersonation—this is not applicable to OSMs, where generally only a single fixed model is released.

OSM watermark durability The watermarked model being open-source introduces an additional key requirement: *durability* against model modifications (see §3). Durability measures how edits to the model parameters affect watermark detection. As discussed in §1, no generation-time watermark is durable in this way, as it can be easily removed by changing the sampling algorithm. In prior works, durability has either been defined in unrealistic scenarios (Christ et al., 2024a), tested against incomplete adversaries (Gu et al., 2024), or not considered at all. Yet, we argue that a thorough consideration of durability is crucial for OSM watermarks, as one primary use case for open-source models is for users to share and deploy edited versions, as evidenced by almost 200 thousand models hosted on Hugging Face, with over 200M downloads. Hence, having a well-defined and systematically evaluated notion of durability is crucial to guide OSM watermark research.

3 Durability of Open-source LLM Watermarks

Next, we proceed to define the durability requirement more concretely. In §4 we will introduce current OSM watermarking methods, and use our durability evaluation setup to systematically evaluate them in §5. To concretize the durability requirement, we have surveyed both the literature and trending Hugging Face models. We identified four main categories of modifications: quantization, pruning, merging, and finetuning, among which we select the most prominent methods and parameter settings.

Quantization Model quantization techniques have emerged as a key method to enable the deployment of increasingly large LLMs on memory-constraint commodity hardware. The fundamental idea underlying quantization is to represent (*quantize*) model weights (and activations) in lower-precision data types. We can split popular methods into two categories: *zero-shot* and *optimization-based*. Zero-shot methods fix the quantization mapping (*buckets*) independently of the model on which they are applied. This makes them computationally inexpensive and a popular choice in consumer libraries (e.g., LLM.INT8() (Dettmers et al., 2022), and NF4 (Dettmers et al., 2024) in Hugging Face). Meanwhile, optimization-based methods aim to minimize a reconstruction error assuming a specific model. This includes methods like HQQ (Badri & Shaji, 2023), which optimizes reconstruction error only over model weights, as well as a range of methods, such as GPTQ (Frantar et al., 2023) and AWQ (Lin et al., 2024), that optimize activation reconstructions over an additional calibration dataset. For evaluating durability, we consider both 8 bits and 4 bits quantized models with different methods.

Pruning While quantization reduces precision across weights, pruning aims to reduce memory requirements by directly removing specific weights completely (*zeroing out*). Unstructured pruning techniques such as WANDA (Sun et al., 2023), SPARSEGPT (Frantar & Alistarh, 2023), and GBLM (Das et al., 2023) independently remove weights while relying on minimizing a reconstruction error between the pruned weights and the dense weights on a calibration dataset. On the contrary, structured pruning methods such as SHEARED LLAMA (Xia et al., 2023) and LLM-PRUNER (Ma et al., 2023) aim to remove entire sets of weights (e.g., rows, columns, or layers) jointly (likewise minimizing a reconstruction error). The advantage of structured pruning is that the resulting model inhibits dense substructures in its weights, allowing for hardware-optimized inference algorithms. At the same time, they are usually not zero-shot and require additional finetuning to restore model performance after pruning. Given such additional modifications, we will only focus on unstructured pruning methods.

Model merging Model merging techniques aim to construct a new model out of a set of base models by combining their individual weights. Importantly, previous works (Matena & Raffel, 2022; Jin et al., 2022) have shown that model merging allows for cheaply combining multiple expert models into a single model that maintains task-specific performance. Most merging techniques (Matena & Raffel, 2022; Jin et al., 2022; Yadav et al., 2023; Yu et al.) thereby rely on the concept of task vectors and task arithmetic: expert knowledge in LLMs lies in orthogonal directions in weight space and can be directly combined to obtain a vector that joins their respective strengths.

We focus on merging via Spherical Linear Interpolation (SLERP) between the watermarked and original model (Goddard et al., 2025). Given the original model θ_0, the watermarked model θ_{wm}, the

angle Ω between θ_0 and θ_{wm} (we set $\Omega := \frac{\pi}{2}$ if θ_0 or θ_{wm} is null), and the interpolation parameter $t \in [0, 1]$, we consider

$$\text{SLERP}(\theta_{\text{wm}}, \theta_0, t) = \frac{\sin[(1-t)\Omega]}{\sin \Omega}\theta_{\text{wm}} + \frac{\sin[t\Omega]}{\sin \Omega}\theta_0. \tag{1}$$

Evaluating durability on the SLERP merge with the original model provides both a reproducible setting for comparing different OSM watermarks and a more adversarial scenario than practical applications. Indeed, merges are performed on models from the same family, and hence, in the case of a watermarked model, all merged models are normally derived from the watermarked model.

Finetuning Model finetuning is widely used to improve pretrained models on a specific domain, usually via additional training on a domain-specific dataset. Besides full model finetuning, which updates all model weights via gradient descent, Low-Rank Adaptation (LoRA) (Hu et al., 2021) has emerged as an incredibly popular finetuning method that performs parameter-efficient low-dimensional weight updates. Besides introducing domain-specific knowledge, one of the common finetuning use cases is instruction finetuning (Zhang et al., 2024), where a base model is trained to follow the instruction format of a given Q&A dataset, enabling its usage as a chat model.

Apart from such supervised finetuning (SFT) methods, Reinforcement Learning (RL)-based finetuning (Ouyang et al., 2022; Wang et al., 2024b) is commonly applied to align models with human preferences or enable more complex reasoning behavior. Yet, due to the additional complexity of RL-based finetuning, it is, for open-source models, so far significantly less common than SFT. Hence, to evaluate watermark durability against finetuning, we focus only on SFT as well as instruction finetuning, both on the full weights of the model and with LoRA.

4 Current State of Open-source LLM Watermarking

Next, we introduce the current state of watermarking schemes for open-source models. We identify two main categories: schemes that embed the watermark into the model using gradient descent (Gu et al., 2024; Xu et al., 2024), and those that directly edit the weights (Block et al., 2025; Christ et al., 2024a). The latter are less computationally expensive but either require architectural changes to the model (Christ et al., 2024a) or more compute-intensive detection (Block et al., 2025). For gradient-based methods, there remain unexplored questions about how such methods generalize to different tasks (Chen et al., 2024) or the viability of statistical guarantees (Xu et al., 2024). As many OSM watermarks are based on generation-time watermarks, we provide a separate introduction in App. A. We evaluate durability against common modifications of the methods presented here in §5.

Distillation-based watermark In Gu et al. (2024), the authors show that generation-time watermarks (Kirchenbauer et al., 2023; Aaronson, 2023; Kuditipudi et al., 2024) can be imprinted into the model weights by distilling the watermark from a teacher model θ_0. Then, the same watermark detector can be used to detect the watermark in the student model θ. In the first variant, the teacher model is used in a black-box way to generate watermarked data $\mathcal{D}_{\text{wm}}$, which the student finetunes on using the cross-entropy loss:

$$\mathcal{L}_{\text{sampling}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}_{\text{wm}}}\left[\sum_{t=1}^{|x|} -\log p_\theta(x_t | x_{<t})\right]. \tag{2}$$

In the white-box variant, the student model is finetuned to mimic the teacher model's next-token distribution, using a loss based on KL-divergence:

$$\mathcal{L}_{\text{logit}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}}\left[\sum_{t=1}^{|x|} \text{KL}(f_w(p_{\theta_0}(.|x_{<t}), \xi_w), p_\theta(.|x_{<t}))\right]. \tag{3}$$

For evaluating durability, we distill two different generation-time watermarks, KGW and KTH (see App. A). We label the corresponding distilled OSM watermarks KGW-D and KTH-D, respectively.

RL-based watermark In Xu et al. (2024), the authors propose integrating the watermark into the RLHF pipeline (Ouyang et al., 2022) by jointly training the watermark and the watermark detector using reinforcement learning. More precisely, given a dataset $\mathcal{D} = \{(x_i, y_i)\}$ of prompts and non-watermarked completions, a watermark detector D parameterized by θ^d, and two models $\theta^0, \theta \in \Theta$, we optimize the following objective using PPO (Schulman et al., 2017):

$$\min_{\theta^d, \theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}[D(x, y, \theta^d) - D(x, y(x, \theta), \theta^d)] + \lambda \text{Reg}(\theta, \theta^0). \tag{4}$$

Weight-editing watermarks Both Christ et al. (2024a) (UNREMOVABLE) and Block et al. (2025) (GAUSSMARK) propose directly editing the weights of the model without needing gradient descent.

UNREMOVABLE introduces a Gaussian noise $\varepsilon = \mathcal{N}(0, \sigma I_{|\Sigma|})$ bias layer in the last projection matrix. The detector, given a text $x \in \Sigma^*$, computes the following Z-score and performs a one-sided Z-test:

$$Z(x, \varepsilon) = \frac{\sum_{t=1}^{|x|} \varepsilon[x_t]}{\sigma|x|}. \tag{5}$$

As no prominent open-source model architecture has a bias layer in the last projection matrix, UNREMOVABLE requires a modification of the model architecture allowing for simple removal by disabling the respective layer. Hence, for current architectures, this watermark is not durable—we still include it in our evaluation for completeness.

GAUSSMARK generalizes UNREMOVABLE to target any subset $\theta_r \subset \theta$ of existing model weights (with dimension $d_r \in \mathbb{N}$). For this we compute $\varepsilon = \mathcal{N}(0, \sigma I_{d_r})$ and consider the model with $\theta_r + \varepsilon$ and all other weights $\theta \backslash \theta_r$ untouched. For detection, we use the following statistic:

$$Z(x, \varepsilon) = \frac{\varepsilon \cdot \nabla_{\theta_r} \log(p_\theta(x))}{\sigma \|\nabla_{\theta_r} \log(p_\theta(x))\|_2}, \tag{6}$$

and also perform a one-sided Z-test. Unlike UNREMOVABLE, GAUSSMARK can be applied to any subset of weights from the model and, crucially, does not necessarily require editing the architecture of the model. However, it requires a forward and (partial) backward pass for watermark detection.

5 EVALUATION

In this section, we present the results of our experimental evaluation of the durability of OSM watermarks (§4) against common model modifications (§3), with our results highlighting that durable OSM watermarking is still an open challenge. We defer omitted experimental details to App. B and present additional in-depth results for each model modification in App. C.

Methods We evaluate KGW-D with $\delta = 2$, $\gamma = 0.25$, and $k = 1$, and KTH-D with key size 256 and no key shift. We do not evaluate distilled AAR (Aaronson, 2023) as it highly degrades text quality (Gu et al., 2024). For UNREMOVABLE, we set $\sigma = 0.6$, and do not evaluate it against pruning, as current pruning methods assume no architectural changes. For GAUSSMARK we set $\sigma = 0.018$ and apply it to the up-projection matrix (`up_proj`) of the MLP layer in the 31st attention block. We omit the RL-based watermark, as we were unable to train it to a sufficient text quality.

Variant: targeted distillation In addition to methods from prior work, we evaluate an additional variant of KGW-D that leverages *contrastive task vectors* (CTV) (Ilharco et al., 2023; Peiran Dong, 2025) in an aim to improve durability. Namely, we first apply KGW-D to θ_0 to obtain θ_1. Then, we finetune θ_1 on a broad-domain dataset (OPENWEBTEXT, see App. B) to obtain θ_2 where the watermark has been removed. Let τ denote the following boolean mask of the model weights:

$$\tau = |\theta_1 - \theta_0| > |\theta_2 - \theta_0|. \tag{7}$$

Intuitively, weights where τ is false were leveraged to remove the watermark—we aim to avoid relying on such weights in our final model by (again) applying KGW-D to θ_0 only where τ is true.

Table 1: Durability evaluation: TPR at 5% FPR, and median PPL of different watermarked versions of LLAMA2-7B under model modifications. (L) denotes LoRA. The missing values indicate cases where a modification was not applicable due to the watermark's architectural changes.

Model Modification			KGW-D		KTH-D		UNREMOVABLE		GAUSS MARK		KGW-D +CTV	
			TPR @5	**PPL**	**TPR @5**	**PPL**	**TPR @5**	**PPL**	**TPR @5**	**PPL**	**TPR @5**	**PPL**
Unaltered			0.99	6.6	0.81	9.7	0.98	7.5	0.99	7.5	0.99	9.1
Quantization	8 bits	GPTQ	0.99	6.6	0.85	7.7	0.99	7.7	0.99	7.2	0.96	9.2
		INT8	0.99	6.5	0.76	7.1	0.99	8.0	0.96	7.9	1.00	8.8
	4 bits	HQQ	0.99	6.8	0.78	7.0	0.98	8.3	0.95	8.4	0.99	9.9
		GPTQ	0.99	7.0	0.78	8.0	0.99	8.2	0.96	8.4	1.00	10.5
		NF4	0.98	6.6	0.77	7.4	0.99	8.6	0.97	8.0	0.99	9.7
Pruning	WANDA	$\rho = 0.2$	0.99	10.0	0.88	11.1	N/A	N/A	0.98	8.0	1.00	9.5
		$\rho = 0.5$	1.00	8.4	0.79	7.8	N/A	N/A	0.97	10.5	0.99	9.5
	GBLM	$\rho = 0.2$	0.99	9.3	0.85	9.9	N/A	N/A	0.98	7.6	1.00	9.1
		$\rho = 0.5$	0.98	8.0	0.76	8.3	N/A	N/A	0.91	10.6	0.99	9.0
	SPARSEGPT	$\rho = 0.2$	1.00	10.1	0.86	13.2	N/A	N/A	0.98	8.6	1.00	10.3
		$\rho = 0.5$	1.00	8.8	0.89	10.0	N/A	N/A	0.94	10.8	0.99	12.9
Merging	SLERP	$t = 0.1$	0.98	8.8	0.52	8.0	0.98	7.6	0.97	11.5	0.98	12.8
		$t = 0.3$	0.82	7.6	0.18	7.3	0.98	7.4	0.82	11.1	0.83	10.9
		$t = 0.5$	0.50	7.7	0.15	6.9	0.93	7.5	0.68	12.0	0.59	11.0
		$t = 0.7$	0.17	7.1	0.11	7.0	0.83	6.7	0.33	10.7	0.34	9.7
		$t = 0.9$	0.04	7.1	0.09	7.4	0.35	6.8	0.10	10.0	0.09	10.1
Finetuning	OPEN WEBTEXT	500 (L)	0.76	6.0	0.14	5.7	0.94	7.3	0.86	6.9	0.72	7.7
		2500 (L)	0.52	5.5	0.13	5.8	0.62	7.4	0.67	7.2	0.56	7.0
		500	0.35	5.7	0.09	5.3	0.24	7.1	0.48	7.7	0.34	7.3
		2500	0.22	5.5	0.10	5.7	0.23	7.0	0.29	7.1	0.22	7.4
	OPEN MATHINSTRUCT	500 (L)	0.99	6.4	0.58	5.6	0.98	6.9	0.97	7.2	0.99	8.1
		2500 (L)	0.99	6.5	0.47	5.5	0.95	7.0	0.96	7.2	0.98	7.9
		500	0.75	4.9	0.18	5.0	0.67	6.4	0.69	6.4	0.87	7.2
		2500	0.55	4.6	0.12	4.5	0.51	5.9	0.66	5.8	0.69	5.8

Model modifications For each modification from §3, we evaluate a range of representative settings. For quantization, we use 4-bit and 8-bit variants, using HGG, LLM.INT8(), GPTQ, and AWQ methods—going below 4 bits significantly degraded text quality in our experiments. We evaluate pruning with sparsity ratios $\rho \in \{0.2, 0.5\}$ using unstructured pruning methods WANDA, GBLM, and SPARSEGPT. For merging, we merge the watermarked model with the base model using SLERP with interpolation ratios $t \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Finally, we consider full finetuning and parameter-efficient LORA, both on broad-domain OPENWEBTEXT (Liu et al., 2019) (Reddit, completions) and task-specific OPENMATHINSTRUCT (Toshniwal et al., 2024) (math, instruction tuning).

We use the LLAMA2-7B model in all experiments, watermark it, apply the model modification, and generate 100 completions of length 200 by prompting with the first 50 tokens of each entry in the RealNewsLike split of C4 (Raffel et al., 2020), as in prior work (Kirchenbauer et al., 2023). For each completion, we evaluate the watermark strength (TPR at 5% FPR, see App. B) and median quality (PPL using LLAMA3-8B) to ensure that our modifications sufficiently retain text quality.

Results: OSM watermarks lack durability In Table 1, we present our main results. Colors correspond to different TPR ranges (green: above 0.9, yellow: between 0.8 and 0.9, red: below 0.8).

First, we observe that nearly all tested schemes are durable to quantization, irrespective of the quantization method, and even at 4 bits. Similar results hold for pruning, where all schemes are highly

durable for $\rho = 0.2$ and remain significantly durable up to a sparsity ratio of 0.5. This intuitively follows from the fact that both quantization and pruning directly aim to minimize the distortion between quantized and original model, thereby also preserving the embedded watermark.

For merging with the unwatermarked model using SLERP, we can observe that weight-editing watermarks are, on average, more durable than distillation-based ones. For UNREMOVABLE, these good results are expected, as merging the bias layer with the null vector (i.e., the bias layer of the unwatermarked model) is equivalent to applying the same noise ε but with a scaled standard deviation

$$\varepsilon_{\text{SLERP}}(t) = \sin[(1-t)\frac{\pi}{2}]\varepsilon. \tag{8}$$

Still, for $t \geq 0.7$ all methods struggle to retain the watermark. Interestingly, we see that KGW-D+CTV is slightly more durable than KGW-D, suggesting that the contrastive task vectors can indeed improve durability, presumably as they localize the watermark to fewer parameters—however, this modification is ultimately ineffective, as it does not improve strength across other model modifications.

Most importantly, we find that for full finetuning on OPENWEBTEXT, **none of the tested watermarking schemes are durable**: after only 500 steps of gradient descent, the TPR drops significantly to a point where the watermark seizes to be useful. Similar results hold for LoRA finetuning on OPENWEBTEXT, where only UNREMOVABLE remains high TPR after 500 steps before dropping significantly after 2500 steps of finetuning. As in the other cases, this can be explained by the architectural modifications of UNREMOVABLE: LoRA finetuning does not directly modify the (usually not present) last layer bias and, therefore, cannot directly modify the watermarked part of the model.

We extend these results by including finetuning experiments with the domain-specific dataset OPENMATHINSTRUCT, modeling a realistic use-case where a user would finetune an OSM on an expert task. Across all schemes, we find higher durability compared to OPENWEBTEXT, especially for LORA. Note that we still evaluate the watermark strength on the general domain C4 test set—as we show in App. C, there is a significant drop in watermark strength when evaluated on the math domain. This points to an interesting phenomenon of *domain-specificity* of OSM watermarks: watermark strength more quickly degrades on domains specifically targeted by finetuning.

Overall, we conclude that while prior work proposed a range of OSM watermarking methods with varying tradeoffs, no method is currently sufficiently durable. Given the real-world prominence of such model modifications, ensuring durability against them remains an open and critical challenge for future research—here our proposed evaluation setup provides an easy way to compare future work.

6 IMPROVING OSM WATERMARK DURABILITY

In this section, we extend current distillation-based OSM methods introduced in §4 by significantly increasing the distillation dataset size and further explore a variant that starts from a randomly initialized model (*distillation pretraining*), as opposed to the standard application on top of an already pretrained model (Gu et al., 2024). As a proof of concept, we show that on a GPT-2 architecture, distilling on a large training set significantly improves watermark durability. Moreover, we show that distillation pretraining and standard distillation finetuning exhibit complementary behaviors: distillation pretraining is more durable against specific task finetuning, whereas standard distillation is more durable against broad-domain finetuning. We further expand on our results in App. D.

Experimental details We perform the distillation of the KGW watermark with $\delta = 2, \gamma = 0.25$, and $k = 1$ on a GPT-2-based architecture. We use the following setup: KGW-D (PRETRAINED) trains a model with random initialization $\theta \in \Theta$ using Gu et al. (2024) (see Eq. (3)) with ≈9B tokens. For KGW-D (LONG), we finetune an already pretrained model (θ_0) with the same distillation approach, also with ≈9B tokens. As a reference, we also distill the watermark using the same hyperparameters as in Gu et al. (2024), i.e., with only ≈ 40 million tokens on top of θ_0 (KGW-D (STANDARD)). To evaluate watermark strength, we use the same setup as in §5 but with 1000 completions instead of 100. To evaluate durability to model modifications, as in §5, we finetune on both broad-domain OPENWEBTEXT and task-specific OPENMATHINSTRUCT datasets.

Distillation on more tokens is more durable In Table 2, we see that both KGW-D (PRETRAINED) and KGW-D (LONG) are significantly more durable against finetuning compared to KGW-D

Table 2: Durability evaluation: TPR at 1% and 5% FPR, and median PPL of different watermarked versions of GPT-2 under finetuning on either OPENWEBTEXT or OPENMATHINSTRUCT.

Model Modification			KGW-D (PRETRAINED)			KGW-D (LONG)			KGW-D (STANDARD)		
			TPR @1	**TPR @5**	**PPL**	**TPR @1**	**TPR @5**	**PPL**	**TPR @1**	**TPR @5**	**PPL**
Unaltered			1.00	1.00	34.4	1.00	1.00	31.8	0.99	1.00	30.7
Finetuning	OPENWEBTEXT	500	0.89	0.97	25.3	0.90	0.97	23.6	0.74	0.87	24.1
		2500	0.57	0.79	25.1	0.63	0.84	23.5	0.47	0.73	23.6
	OPENMATHINSTRUCT	500	0.98	0.99	24.6	0.95	0.98	21.1	0.80	0.89	21.4
		2500	0.91	0.96	22.6	0.81	0.91	19.4	0.48	0.66	19.1

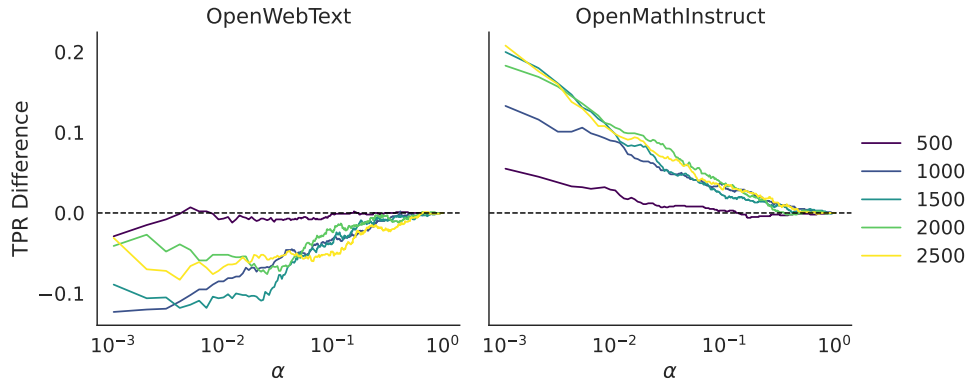

Figure 2: Evaluation of the TPR difference between KGW-D (PRETRAINED) and KGW-D (LONG) when finetuned (as a model modification) on either OPENWEBTEXT or OPENMATHINSTRUCT.

(STANDARD). This confirms our prior intuition that OSM watermark durability scales with the extent of the model's exposure to watermarked text during training. However, while the results are promising when finetuning on OPENMATHINSTRUCT, finetuning on a broad-domain dataset such as OPENWEBTEXT still significantly deteriorates the watermark.

Distillation at pretraining is task-aware In Figure 2 we present more granular results, aiming to decouple the effects of training a randomly initialized model and simply increasing the number of training tokens. Namely, we show the difference of TPR between KGW-D (PRETRAINED) and KGW-D (LONG) across different rejection rates and (model modification) finetuning steps. We observe that the pretraining distillation approach is slightly worse at preserving the watermark when later finetuned on a broad-domain dataset (at most 10% TPR difference) but better at preserving the watermark when finetuned on a task-specific dataset (up to 20% TPR increase).

Perhaps more interestingly, as we evaluate the TPR on C4 prompt completions, i.e., a broad-domain/general task, we conclude that the distillation pretraining watermarked model may exhibit a form of *task-awareness*: If the model is finetuned on unwatermarked data from a specific task, it will not un-learn the watermark on other tasks. We hypothesize that this robustness is due to the model only ever being trained on watermarked text, i.e., never seeing unwatermarked text before the model modification stage. For the standard distillation watermark, even when the number of training tokens is increased to match the pretraining case, we do not observe this behavior, even though it is, interestingly, slightly more durable against finetuning on a general dataset.

Limitations Our results suggest both that the way the model distills the watermark matters, as evidenced by the difference between distillation during pretraining and simple finetuning, and that increasing the training set size can indeed improve durability. However, the practicality of these effects is limited, as (1) even with much larger dataset sizes, the increase in durability is often

insufficient for the watermark to remain effective against prolonged downstream finetuning, and (2) scaling this approach to real-world models is expensive. This also makes it unclear how much our results on the comparatively small GPT-2 architecture generalize to larger and more capable models.

7 Conclusion

In this work, we studied the problem of watermarking open-source LLMs. Noting the fragmentation in prior work on this topic, we laid the foundation for systematic future study of OSM watermarking. We revisited the requirements for generation-time LLM watermarking, discussed how they apply in the open-source setting, introduced a novel requirement, *durability*, and applied a new systematic evaluation procedure to existing OSM watermarks. Finding that none of the current watermarks are durable, we proposed scaling watermark distillation methods, highlighting their benefits and limitations. We hope our work can kick-start progress on this crucial yet challenging problem.

References

Scott Aaronson. Watermarking of large language models. In *Workshop on Large Language Models and Transformers, Simons Institute, UC Berkeley*, 2023.

Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned llms with simple adaptive attacks. *arXiv preprint arXiv:2404.02151*, 2024.

Hicham Badri and Appu Shaji. Half-quadratic quantization of large machine learning models, November 2023. URL https://mobiusml.github.io/hqq_blog/.

Adam Block, Ayush Sekhari, and Alexander Rakhlin. Gaussmark: A practical approach for structural watermarking of language models, 2025. URL https://arxiv.org/abs/2501.13941.

CEU. Proposal for a regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts - analysis of the final compromise text with a view to agreement. 2024.

Lingjie Chen, Ruizhong Qiu, Siyu Yuan, Zhining Liu, Tianxin Wei, Hyunsik Yoo, Zhichen Zeng, Deqing Yang, and Hanghang Tong. Wapiti: A watermark for finetuned open-source llms, 2024. URL https://arxiv.org/abs/2410.06467.

Miranda Christ, Sam Gunn, Tal Malkin, and Mariana Raykova. Provably robust watermarks for open-source language models. *arXiv*, 2024a.

Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. *COLT*, 2024b.

Rocktim Jyoti Das, Mingjie Sun, Liqun Ma, and Zhiqiang Shen. Beyond size: How gradients shape pruning decisions in large language models. *arXiv preprint arXiv:2311.04902*, 2023.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823, 2024a.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823, 2024b.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong,

Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35: 30318–30332, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv*, 2024.

Pierre Fernandez, Anthony Level, and Teddy Furon. What lies ahead for generative ai watermarking. *GenLaw Workshop at ICML*, 2024.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pp. 10323–10337. PMLR, 2023.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers, March 2023.

Thibaud Gloaguen, Nikola Jovanović, Robin Staab, and Martin Vechev. Black-box detection of language model watermarks. In *ICLR*, 2025.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's mergekit: A toolkit for merging large language models, 2025. URL https://arxiv.org/abs/2403.13257.

Google DeepMind. Identifying ai-generated content with synthid, 2025. https://deepmind.google/technologies/synthid/, last accessed: Feb 8 2025.

Chenchen Gu, Xiang Lisa Li, Percy Liang, and Tatsunori Hashimoto. On the learnability of watermarks for language models. In *ICLR*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, October 2021.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. In *ICLR*, 2024.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic, 2023. URL https://arxiv.org/abs/2212.04089.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.

Nikola Jovanović, Robin Staab, and Martin Vechev. Watermark stealing in large language models. *ICML*, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *ICML*, 2023.

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. In *ICLR*, 2024.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *TMLR*, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration, July 2024.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Lijie Wen, Irwin King, and Philip S. Yu. A survey of text watermarking in the era of large language models. *arXiv*, 2023.

Aiwei Liu, Sheng Guan, Yiming Liu, Leyi Pan, Yifei Zhang, Liancheng Fang, Lijie Wen, Philip S Yu, and Xuming Hu. Can watermarked llms be identified by users via crafted prompts? *ICLR*, 2025.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. URL https://arxiv.org/abs/1907.11692.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, March 2022.

Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. Attacking LLM watermarks by exploiting their strengths. *arXiv*, 2024.

Song Guo Peiran Dong, Haowei Li. Durable quantization conditioned misalignment attack on large language models. *ICLR*, 2025.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. Openmathinstruct-1: A 1.8 million math instruction tuning dataset, 2024. URL https://arxiv.org/abs/2402.10176.

Jingtan Wang, Xinyang Lu, Zitong Zhao, Zhongxiang Dai, Chuan-Sheng Foo, See-Kiong Ng, and Bryan Kian Hsiang Low. WASA: watermark-based source attribution for large language model-generated data. *arXiv*, 2023.

Lean Wang, Wenkai Yang, Deli Chen, Hao Zhou, Yankai Lin, Fandong Meng, Jie Zhou, and Xu Sun. Towards codable text watermarking for large language models. *ICLR*, 2024a.

Shuhe Wang, Shengyu Zhang, Jie Zhang, Runyi Hu, Xiaoya Li, Tianwei Zhang, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. Reinforcement learning enhanced llms: A survey. *arXiv preprint arXiv:2412.10400*, 2024b.

Yihan Wu, Zhengmian Hu, Hongyang Zhang, and Heng Huang. Dipmark: A stealthy, efficient and resilient watermark for large language models. *ICML*, 2024.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

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

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-Merging: Resolving Interference When Merging Models, October 2023.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *CoRR*, 2024.

KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. Advancing beyond identification: Multi-bit watermark for language models. *NAACL*, 2024.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language Models are Super Mario: Absorbing Abilities from Homologous Models as a Free Lunch. URL http://arxiv.org/abs/2311.03099.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey, 2024. URL https://arxiv.org/abs/2308.10792.

Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairoze, Andres Fabrega, Nicholas Carlini, Sanjam Garg, Sanghyun Hong, Milad Nasr, Florian Tramèr, Somesh Jha, Lei Li, Yu-Xiang Wang, and Dawn Song. Sok: Watermarking for ai-generated content. *arXiv*, 2024.

Tong Zhou, Xuandong Zhao, Xiaolin Xu, and Shaolei Ren. Bileve: Securing text provenance in large language models against spoofing with bi-level signature. *NeurIPS*, 2024.

Banghua Zhu, Norman Mu, Jiantao Jiao, and David A. Wagner. Generative AI security: Challenges and countermeasures. *arXiv*, 2024.

A Description of Generation-time Watermarks

In this section, we introduce a high-level description of two generation-time watermarks, KGW (Kirchenbauer et al., 2023) and KTH (Kuditipudi et al., 2024).

KGW watermark KGW watermark (Kirchenbauer et al., 2023) works by partitioning, at each step of the generation, the vocabulary into a Red and Green subset using the private key ξ and summing the hashes of the k previous tokens. The Green subset has a size of $\gamma|\Sigma|$, with $\gamma \in [0, 1]$. Logits of the tokens in the Green subset are boosted by $\delta > 0$, making them more likely to be sampled. The watermark detector works by performing a binomial test on the number of Green tokens in the text.

KTH watermark KTH watermark (Kuditipudi et al., 2024) is parametrized by a key length $n_{key} \in \mathbb{N}$ and a key $\xi \in [0, 1]^{|\Sigma| \times n_{key}}$, where each entry $\xi_k \in [0, 1]^{|\Sigma|}$ is uniformly distributed in $[0, 1]$. At each step t of the generation, given a probability distribution p_t over Σ, the next token is chosen as the $\arg\max$ of $(\xi_{(t \mod n_{key})})^{p_t}$. Additionally, to allow multiple generations given a fixed prompt, the key is randomly shifted by a constant before generating a new sequence. Finally, given a text x, detection works by performing a permutation test using the minimum Levenshtein distance of the alignment cost $d(x, \xi) = \sum_{t=1}^{|x|} \log(1 - \xi_{(t \mod n_{key}, x_t)})$.

B Experimental Details

In this section, we provide an in-depth list of the parameters used for the model modifications (§3) and the watermarking schemes (§4) that we use in our evaluation in §5. For all watermarking schemes, we use the same LLAMA2-7B as our base unwatermarked model.

Watermarking schemes For distillation-based generation-time watermarks, we use the distillation loss from Eq. (3) with OpenWebText as $\mathcal{D}$. We distilled the watermark using the same hyperparameters as in Gu et al. (2024): a batch size of 64, with 512 tokens per input, a learning rate of 1e-5, the AdamW optimizer (Kingma & Ba, 2017) with $(\beta_1, \beta_2) = (0.9, 0.999)$, and no weight decay. For KGW-D, we use $\delta = 2$, $\gamma = 0.25$, and $k = 1$. For KTH-D, we use $n_{key} = 256$ with no key shift.

For KGW-D+CTV, we first distill KGW with the parameters described above on the full model. Then, we finetune the watermarked model on OpenWebText for 2500 steps with cross-entropy loss, batch size of 64, 512 tokens per input, a learning rate of 1e-5, the Adafactor optimizer with a cosine learning rate decay, and a linear warmup for the first 500 steps. We then compute the contrastive task vector (Eq. (7)) and learn KGW on the selected weights.

For the UNREMOVABLE watermark, we set the standard deviation to $\sigma = 0.6$ to ensure sufficient power in the unaltered watermarked model while not degrading the model's quality too much. For GaussMark, as suggested in Block et al. (2025), we perform a grid search to find the best layer and standard deviation to balance watermark power and quality degradation. We find the optimal layer to be the 31st MLP up_proj layer with a standard deviation of $\sigma = 0.018$.

Model modifications For all quantization and pruning methods, we use the default hyperparameters suggested for each technique. For finetuning on OpenWebText, we use a batch size of 32, with 512 tokens per input, a learning rate of 1e-5, the Adafactor optimizer with a cosine learning rate decay, and a linear warmup for the first 500 steps. For finetuning on OpenMathInstruct, we introduce two new instruction tokens and use the same settings but with a maximum of 2048 tokens per input to accommodate the entire math problem and solution. For both finetuning tasks, we use the same LoRA adapter with a low-rank dimension of 16 and an alpha of 32. Moreover, we apply the LoRA adapter only to the following layers: v_proj, k_proj, o_proj, and q_proj.

Metrics As our main metric in §5 we use the true positive rate of the watermark detector, evaluated at a false positive rate of 5%. While we believe this FPR level is high for fully practical applications, it is both a common evaluation setting in prior watermarking literature, and more importantly, calibrated to the current strength of OSM watermarks. Ideally, OSM watermarks would advance to a level where this metric is not useful anymore as the corresponding TPRs would be close to 1 for many methods, and evaluations would focus on lower, more practical FPR levels.

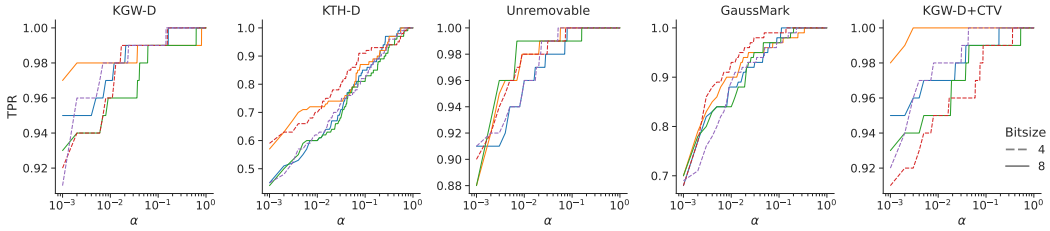

Figure 3: Evolution of different watermark TPRs against multiple quantization methods. Each color corresponds to a different quantization method. The rejection rate α is in logarithmic scale for clarity.

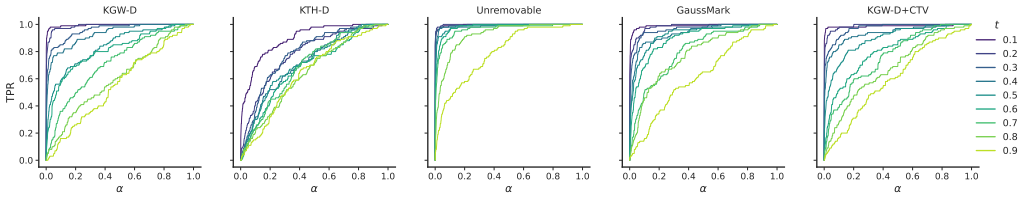

Figure 4: Evolution of different watermark TPRs for different SLERP interpolation levels t.

C Additional Results on Watermark Durability

In this section, we analyze the ROC curves (Experimental TPR versus FPR) to evaluate in detail the durability of the watermarks against all common model modifications (App. C.1). We confirm the observation from §5 that current OSM watermarks lack durability. In App. C.2, we measure the durability of the watermark on a specific task when finetuning is performed on the same task.

C.1 ROC Curves of OSM Watermarks Against Common Model Modifications

Here, we extend the results from §5 by presenting ROC curves for all schemes tested under all common model modifications identified in §3.

Quantization In Figure 3, we see no visible difference between 4 bits and 8 bits quantization in the empirical TPR of different watermarking schemes. This suggests that quantization is not a challenge for current OSM watermarks. Intuitively, this is an expected result as most quantization techniques try to preserve the model performance as much as possible when quantizing the model. Hence, by minimizing the impact on model performance, quantization also minimizes the impact on the watermark.

Merging Similarly, in Figure 4, we see the ROC curve for different values of the SLERP interpolation parameter and different watermarking schemes. We see that weight-editing watermarks are more durable against merging. As explained in §5, this is an expected result.

Pruning In Figure 5, we see the ROC curve for different values of sparsity ρ and different watermarking schemes. For $\rho > 0.5$, the model quality is too low to be usable; hence, we do not compute the TPR for higher sparsity ratios. We see that for most schemes tested, the watermark is durable against pruning, even for high sparsity ratios. As with quantization, this is an expected result. With unstructured pruning, the objective is to find the sparse weights that minimize the distortion in the dense model activations. By minimizing such distortion, pruning techniques also preserve the watermark.

Finetuning In Figure 6, we see the ROC curve for full finetuning on either OpenWebText or OpenMathInstruct. The conclusion is similar as the one from Table 1: the watermark is not durable against finetuning even for a few steps. This is unsurprising as with finetuning, we want the model to learn the distribution of the training dataset. Hence, because the training dataset is not watermarked, its distribution significantly differs from the model's previously learned

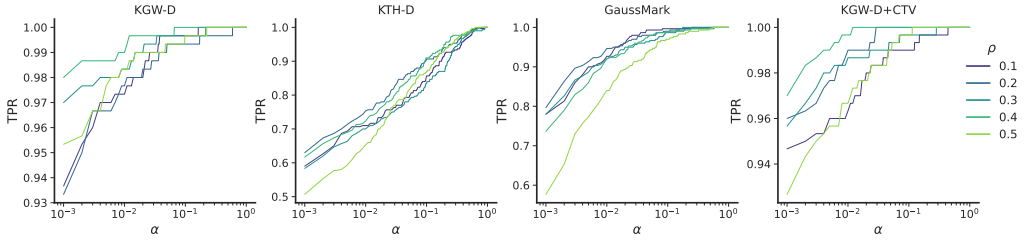

Figure 5: Evolution of different watermark TPRs averaged over three pruning techniques (WANDA, GBLM, and SPARSEGPT) at different sparsity ratios ρ. The rejection rate α is in logarithmic scale.

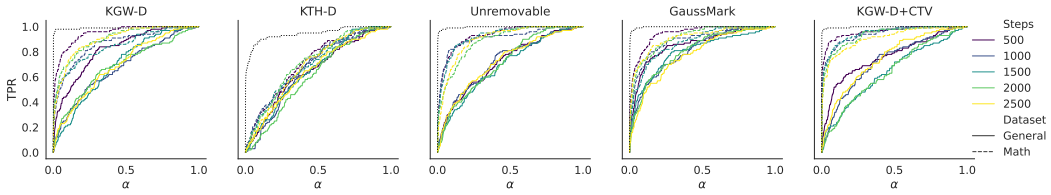

Figure 6: TPR for differen finetuning lengths on either OPENWEBTEXT (General) and OPENMATHINSTRUCT (Math). The black dotted line corresponds to the unaltered watermark model TPR.

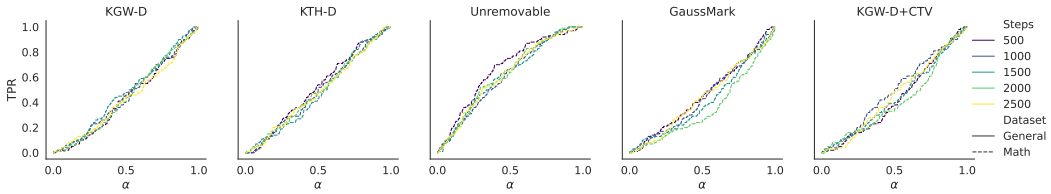

Figure 7: TPR for different lengths of finetuning on OPENMATHINSTRUCT (Math). The watermark is evaluated on answers from GSM-8K.

distribution. Therefore, finetuning effectively bridges the gap between these two distributions, removing the watermark. This is why, for most schemes, finetuning on a specific domain (Math) does not necessarily remove the watermark as much as finetuning on a broad/general domain.

C.2 WATERMARK DURABILITY ON EXPERT TASKS

Here, we evaluate the ability of a watermarked open-source model to retain the watermark signal on newly learned expert tasks, e.g., math. We use the same watermarks as in §5.

Experimental details We first instruction-finetune a watermarked model on OPENMATHINSTRUCT to teach the model how to solve math questions. We use the same hyperparameters as in §5. Then, instead of measuring the watermark durability on broad C4 prompt completions, we consider 200-token answers to math questions. This is closer to a practical scenario: if a user finetunes a given model on a specific task, it is expected that the model will be used for that task as well, hence if the watermark is not durable, most text produced by this model in practice will effectively not be watermarked.

OSM Watermarks do not transfer to new domains In Figure 7, we see that none of the tested watermarks are durable when finetuned and evaluated on a specific domain. This contrasts with the evaluation from Table 1, where we show that finetuning on a specific domain but evaluating the watermark strength on a general domain does not lower the watermark strength as significantly. This highlights a crucial limitation of the durability of current watermarks for open-source models, and suggests that the watermark strength of the task-specific watermarked model should also be evaluated on similar task-specific datasets, which has been overlooked in previous works (Chen et al., 2024).

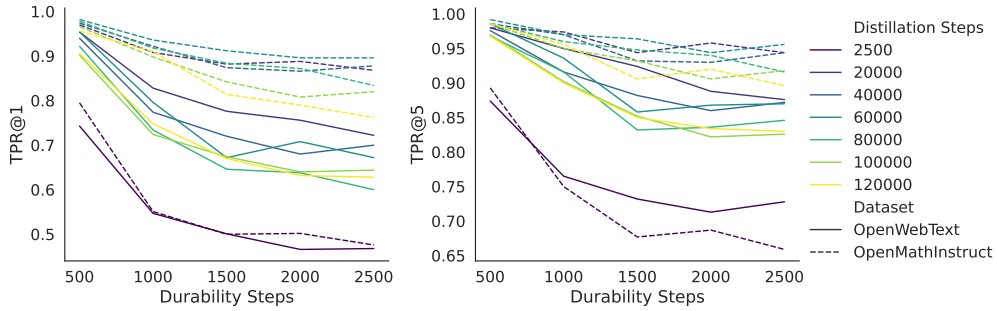

Figure 8: Evaluation of the TPR at 1% and 5% against finetuning for KGW-D with increasing training dataset size when distilling the watermark on GPT-2.

D Influence of the Training Length on Distillation-based Watermark Durability

In this section, we extend the experiment from §6 and specifically ask whether scaling the number of tokens when distilling the watermark necessarily improves durability against finetuning.

Experimental details We again perform the distillation of the KGW watermark with $\delta = 2, \gamma = 0.25$, and $k = 1$ on a GPT-2 pretrained model. We distill the watermark for a different number of steps, where each step consists of approximately 60 thousand tokens. We then evaluate the watermark strength every 20,000 steps (i.e., approximately 1.4B tokens). To evaluate the watermark strength, we use the same setup as in §5 but with 500 completions instead of 100. To evaluate durability against finetuning, we finetune on both broad-domain OpenWebText and task-specific OpenMathInstruct.

Durability does not scale with distillation training length In Figure 8, we plot both the TPR at 1% and 5% FPR for the different models tested. We see that the watermarked model that has been distilled for 20 thousand steps is actually more durable than the one distilled for only 2500 steps (KGW-D (Standard) from Table 2), but also more durable than the ones distilled for longer (with up to a 10% TPR@1 difference). It also seems that, as the distillation training length increases, its impact on durability plateaus, as all TPR curves from 80 thousand steps onward are very similar. These results confirm the limitations highlighted in §6: increasing the training set size, up to a point, improves durability, yet it is still insufficient for the watermark to remain effective against prolonged downstream finetuning.