Watermarking LLMs: feasibility, constraints, and strategies

Thomas Souverain, Alexei Grinbaum, Etienne Klein

DES FILIGRANES POUR LES MODÈLES DE LANGAGE

WATERMARKING LLMS: FEASIBILITY, CONSTRAINTS, AND STRATEGIES

WHITE PAPER

OPENLLM DELIVERABLE 6.1

Thomas Souverain[*] **Alexei Grinbaum**[†] **Etienne Klein**[‡]

[*]CEA-Saclay/Larsim, 91191 Gif-sur-Yvette, France. thomas.souverain@cea.fr.
[†]CEA-Saclay/Larsim, 91191 Gif-sur-Yvette, France. alexei.grinbaum@cea.fr.
[‡]CEA-Saclay/Larsim, 91191 Gif-sur-Yvette, France. etienne.klein@cea.fr.

CONTENTS

RÉSUMÉ FRANÇAIS

En Europe et à l'international, de plus en plus de réglementations et de standards exigent que les contenus générés par l'Intelligence Artificielle (IA) soient marqués dans un format lisible par ordinateur. Cette exigence permettrait aux citoyens et aux plateformes d'utiliser des logiciels de détection pour évaluer la provenance d'un contenu suspect, contribuant ainsi à la mise en œuvre du principe éthique de transparence. Les filigranes (watermarking) mobilisés pour marquer l'image, l'audio ou la vidéo générés par IA, offrent des garanties cryptographiques. De telles garanties sont difficiles à obtenir pour les textes générés par IA, où les subtiles modifications introduites par un filigrane peuvent aisément être effacées ou imitées par reformulation.

Dans ce livre blanc, nous traitons une question pratique : les filigranes constituent-ils une stratégie appropriée pour un fournisseur de modèle de langage frugal (à faible coût de calcul) en libre accès ? Si oui, quelles techniques de filigrane doivent être privilégiées ?

Nous brossons un panorama des filigranes pour les grands modèles de langage (Large Language Models, LLMs), centré sur les exigences d'évaluation de l'Union Européenne (UE). Nous mettons en regard les atouts et inconvénients de chaque technique, suivant l'étape de production du LLM où le marquage intervient (entraînement, architecture, inférence, édition du texte). Nous aboutissons ainsi à des recommandations opérationnelles sur la mise en place de filigranes dans les LLMs, veillant à ce que les techniques de marquage soient détectables, robustes, interopérables et maintiennent la qualité du LLM. Ces méthodes sont applicables aux modèles à poids ouverts (open-weights) et frugaux, en particulier aux LLMs utilisés dans le contexte éducatif filé dans le projet OpenLLM.

1. **Prioriser les filigranes structurels pour les LLMs à poids ouverts.** Pour les déploiements de modèles à poids ouverts, la voie la plus robuste consiste à intégrer les filigranes à l'intérieur du modèle (par exemple, via la distillation ou l'apprentissage par renforcement, ou l'édition des poids post-entraînement). Plus difficiles à effacer, ces approches structurelles ouvrent de prometteuses pistes de recherche pour rester détectables malgré le fine-tuning et l'élagage de paramètres (pruning) du modèle, ce qui les rend particulièrement pertinentes pour les LLMs à poids ouverts.
2. **Adopter une stratégie de marquage multicouche pour répondre à l'exigence d'interopérabilité de l'UE.** Plutôt que de s'appuyer sur un signal unique, nous recommandons une stratégie multicouche dans laquelle les attaquants doivent déjouer plusieurs marques partiellement indépendantes. Concrètement, nous recommandons de combiner (i) un filigrane à l'intérieur de l'architecture du LLM, (ii) dans le cas où le fournisseur du LLM contrôle le post-entraînement (API ou modèle propriétaire), l'ajout d'une couche de filigrane au moment de l'inférence (biais sur les logits ou l'échantillonnage – sampling) comme seconde ligne de défense, et (iii) des mécanismes de provenance non liés au filigrane (métadonnées signées – metadata / assertions de provenance) préservant les preuves contextuelles même lorsque les marques sont incertaines ou supprimées. Une telle stratégie est interopérable avec les régulations et normes existantes, européennes et internationales (CAC, 2025; C2PA, 2025; EU, 2024).
3. **Mettre en œuvre un système de marquage vérifiable et frugal pour les LLMs à poids fermés (filigrane + mention visible + vérification ex post).** Pour les LLMs propriétaires, nous recommandons un système de marquage frugal s'inscrivant dans notre stratégie interopérable, adapté à un coût d'inférence et de stockage limité : (1) le filigrane comme première couche, combinant dans l'idéal des filigranes structurels légers (édition de poids) avec un biais ajouté à l'inférence pour renforcer la détectabilité ; (2) une attribution visible dans le fichier final, en ajoutant une simple mention standardisée de génération par l'IA et en la liant à un identifiant unique de provenance cryptographique ; et (3) une attribution vérifiable par rapport à une base de données de référence, via l'empreinte numérique (fingerprinting) ou la journalisation le cas échéant (logging), permettant de comparer le contenu suspect ex post sans conserver toutes les sorties. Cette stratégie de traçabilité ne prend plein sens sur le plan opérationnel que si les conditions de détection sont standardisées et auditables (longueur minimale du texte, contextes de fonctionnement, types d'erreur). Les procédures de marquage et clés de détection des fournisseurs de LLMs doivent être rendus accessibles à l'audit public, en vue d'une gouvernance contrôlée.

EXECUTIVE SUMMARY

European and international regulation and emerging standards increasingly require that content generated by Artificial Intelligence (AI) be marked in a machine-readable format. This requirement would enable citizens and downstream platforms to use detection software to assess content provenance, contributing to the ethical principle of transparency. Watermarking generated content or embedding an invisible signature in images, video, and audio outputs provides cryptographic guarantees. However, comparable guarantees for AI-generated text remain elusive, largely because rephrasing attacks make it easy to detect or erase any modification in text outputs.

In this White Paper, we address a practical question: is watermarking an appropriate strategy for a provider of an open-access frugal (lowest computational expense) language model? If so, which watermarking techniques should be prioritized?

We review the current landscape of watermarking techniques for Large Language Models (LLMs) with a specific focus on evaluation according to the EU requirements. We clarify the main trade-offs of each watermarking method, depending on its place in the generation pipeline and on inference-time settings or attacks. We translate our Section 5 findings into actionable recommendations for watermarking LLMs while ensuring marking techniques are detectable, robust, interoperable, and maintain LLM quality. These methods are applicable to open-weights and frugal models, in particular to LLMs used in the educational context as developed in the OpenLLM project.

1. **Prioritize structural watermarking for open-weights LLMs.** For open-weights deployments, the most robust avenue is to embed the watermark inside the model (e.g. via distillation- or Reinforcement Learning-based objectives, or post-training weight editing). Such structural approaches are harder to neutralize through textual and token-based attacks and show promising research avenues to remain detectable across fine-tuning and pruning, making them particularly relevant for open-weights LLMs.
2. **Adopt a multi-layered marking strategy to meet the EU requirement of interoperability.** Rather than relying on a single signal, we recommend a multi-layered strategy in which attackers must defeat several partially independent checks. Concretely, we recommend combining (i) watermarking inside LLM architecture, (ii) in case the LLM provider controls post-training (API or trusted environments), adding an inference-time watermarking layer (logits or sampling bias) as a second line of defense, and (iii) non-watermark provenance mechanisms (signed metadata / provenance assertions) to preserve contextual evidence even when marks are uncertain or stripped. This strategy is designed to remain interoperable in agreement with the standards (C2PA, 2025) and the European and international regulatory approaches (EU, 2024; CAC, 2025).
3. **Implement a frugal verifiable pipeline for closed-weights LLMs (watermark + visible notice + ex post check).** For proprietary LLMs, we recommend a frugal pipeline following our interoperable pipeline fitting with limited inference and storage cost: (1) watermarking as the first layer, ideally combining a lightweight structural watermark with an inference-time bias to strengthen detectability; (2) visible attribution in the final file, by adding a simple standardized notice of AI generation and linking it to a unique identifier of cryptographic provenance; and (3) verifiable attribution against a reference database, via fingerprinting (or logging where appropriate), enabling suspected content to be compared ex post without retaining all outputs. This pipeline becomes operationally meaningful only if detection conditions are standardized and auditable (e.g. minimal text length, operating ranges, and error profiles), and if public auditors can access verification procedures and, when necessary, detection keys under controlled governance.

1 INTRODUCTION

Large Language Models (LLMs) can generate fluent, context-relevant text that looks meaningful even if it is only an outcome of token-level pattern matching. In an instantiation of the ELIZA effect Weizenbaum (1966), the human user cannot avoid making anthropomorphic projections of meaning when reading such text. This non-human pathway of text generation in LLMs enables increased performance and instrumental efficiency in task implementation, but it also modifies the user's cognitive and behavioural patterns in the long term. The modification of human behaviour raises significant ethics issues and requires ethics-by-design measures (Bayerl et al., 2026).

In this context, watermarking LLM-generated content by embedding a hidden signature (Briquet, 1907; Hunter, 1967) is an ethical requirement allowing the user to trace human or machine authorship. Already today, one can hardly, if at all, find out whether content of any format has been created by a human. This is a particularly urgent problem in the educational context of OpenLLM project, because teachers and educators struggle to evaluate students who have unrestricted access to generative AI tools. Tracing origin would allow a teacher to distinguish between autonomous student work and AI-assisted work. In such settings, a maker's mark would function as a passport or provenance mark for the non-human author. This would remove status confusion which is currently producing a detrimental effect on education.

Beyond intellectual-property disputes, the provenance of text matters for civic and psychological reasons: the erosion of the distinction between human and machine expression can fuel emotional, cognitive, and status confusion, producing downstream societal effects (Grinbaum and Adomaitis, 2022). Watermarks and related digital signatures are therefore increasingly discussed as policy requirements for generative AI models, especially when content circulates at scale and moderation must operate under uncertainty.

Beyond instances of direct user interaction with chatbots, the broader sphere of meaning is already being transformed. No longer merely a question of originality or plagiarism, the history of texts that influence and shape human readers is a matter of utmost importance. If humanity is to maintain its *human* history, this history ought to be made of texts that convey human-intended meaning and represent facts of the human world. Today, a non-human author can appear more coherent and attractive than a human writer. Consider, for instance, a flawless piece of disinformation falsely presenting conspiratorial or political claims. Hence the need for watermarks embedded within machine-produced content in order to let humans write their own narratives, taking responsibility for the history that they collective create.

Regulatory initiatives on transparency have accelerated across the globe, although they differ in scope and enforcement. Speaking at the Indian AI Impact Summit on 19 February 2026, Prime

Minister Modi emphasized that digital content, like food, should carry authenticity labels so people can distinguish real material from AI-generated output. He presented watermarking and clear provenance standards as necessary regulatory tools for building trust in AI from the outset (Modī, 2026). In Europe, the EU AI Act frames transparency duties for providers, and its Article 50(2) and Recital 133 require that generated outputs be marked. In the United States, the federal Executive Order that promoted labeling requirements for synthetic content was rescinded in early 2025 (House, 2025), but several states have nevertheless adopted or maintained marking-oriented measures, including California, New York, and Washington (California, 2024; New York, 2024; Washington, 2025). Across these jurisdictions, the same operational question appears: how to enable scalable downstream verification for teachers, platforms, journalists, or courts through a combination of visible traces, machine-readable marks, and robust detection mechanisms.

Yet, the technical feasibility of watermarking is not uniform across content modalities. For images and video, imperceptible noise can often be embedded and later audited, sometimes with cryptographic-style assurance (Evennou et al., 2024). Text is different: it is discrete and can be easily paraphrased. This makes the classic trade-off between detectability and removability especially acute. Industry standards such as C2PA (C2PA, 2025) largely rely on "hard binding" through cryptographic hashes, a method that has achieved higher maturity for pseudo-continuous content, like video or audio, than for LLM-generated text. China's standard reflects this asymmetry by mandating "implicit labelling" for synthetic images and videos, while only encouraging it for generated text (CAC, 2025).

Against this backdrop, we examine a practical question: for an actor that aims to develop and deploy its own LLM - potentially smaller and distributed under open access - is watermarking this LLM an appropriate ethical and technical measure? If yes, which watermaking techniques are most promising? Our goal is not to advocate for a single technique, but to evaluate several available LLM watermaking techniques. We first present the main families of such techniques, organized according to the place at which they are applied in the LLM development and inference pipeline (Section 2). We then review detectability and robustness of watermarking techniques by evaluating them against the requirements of output quality of and detection interoperability. This latter criterion is particularly important in the EU regulatory context (Section 3). Next, we summarize the trade-offs associated with existing watermarking approaches (Section 4). We conclude by outlining multi-layer marking strategies for EU providers, tailored to both open-weights and closed-weights LLMs, and ensuring that our recommendations remain compatible with frugal deployments and existing standards (Section 5).

2 WATERMARKING A LARGE LANGUAGE MODEL: STATE-OF-THE ART

Building on recent taxonomies (Dathathri et al., 2024; Fernandez et al., 2025; Souverain, 2025), we arrange LLM watermarking methods by *when* they intervene in the model lifecycle. The key observation is that a watermarking signal can be introduced *before* the model generates any text (e.g. by transforming training data), *during* inference (by biasing the model or the decoding procedure), or *after* generation (by rewriting the produced text). This taxonomy is useful for decision-making at the engineering level, because each choice exposes different levers for watermarking, as well as different types of attack and compliance constraints.

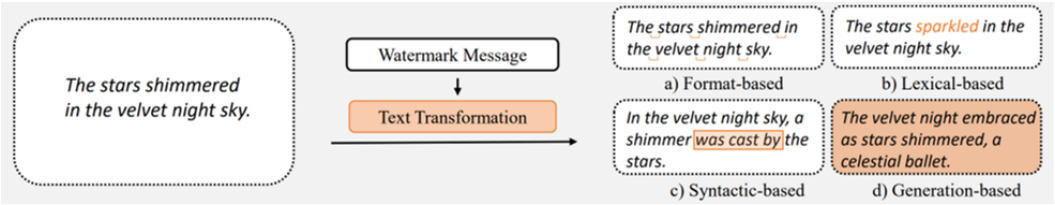
2.1 PRE- AND POST-PROCESSING APPROACHES: SYNTACTIC AND SEMANTIC WATERMARKING

Although they act on different objects, either pre-training corpora or LLM-generated text, pre- and post-processing approaches often share a common *technique*: they edit the observable string (or its encoding) rather than the model internals. They are comparatively easy to deploy and to evaluate, and can be layered on top of third-party models. Their main limitation is that once the transformation rule is discovered, it can often be reversed or bypassed.

The first family are *syntactic* watermarks. Wei et al. (2024) propose pseudo-random substitutions of Unicode characters, embedding a signature directly into the training data that will later feed the LLM. A symmetric post-processing idea exists once the model is trained: EASYMARK (Sato et al., 2023) applies character-level substitutions to LLM outputs, where hidden characters are concatenated so that a detector can later recover the pattern.

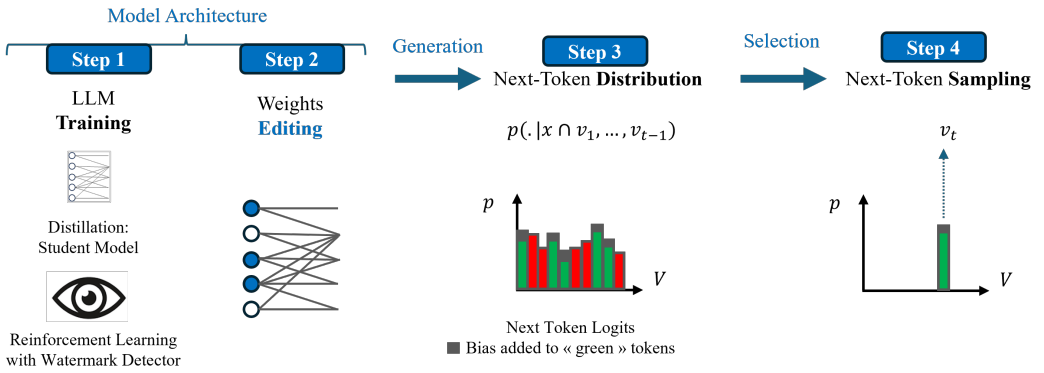
The second family are *semantic* watermarks enabled through pre- and post-processing. They aim at preserving meaning while steering lexical realizations. For example, Zhang et al. (2024) watermark generated text through synonym substitution: based on token frequencies and local context, the method proposes replacements intended to appear more often in watermarked outputs while remaining plausible to human readers. Figure 1 illustrates both of these changes in an already generated text:

Figure 1: Watermarking for existing text (based on Fig. 2 in Liu et al. (2024b))



At the intersection of semantic and syntactic edits, several approaches train auxiliary models that detect and replace targeted fragments in either training corpora or generated texts. Zhang et al.

Figure 2: Watermarking a LLM by altering the Generation Process: Four Steps (Souverain, 2025)



Instead of altering the training data or the generated text, watermarking can modify the process of generation itself. We identify four steps of the generation process where an invisible signature can be embedded. Modifications of the architecture and parameters can happen during (step 1) or after training (step 2). It can then slightly bias the distribution (step 3) or selection (step 4) of the next-token generated by the LLM.

(2024) train a neural network that focuses on punctuation and function words (e.g. prepositions) to suggest discrete substitutions, while Abdelnabi and Fritz (2021) use an encoder–decoder pair to generate replacements and explicitly minimize their detectability by malicious actors.

2.2 In-Processing approaches: training, weights, distribution, and sampling

Whereas editing datasets or finalized outputs is often practical and flexible to test (Liang et al., 2024), watermarking *inside* the model or the decoding loop can be more resilient to post-training transformations such as quantization, pruning, and fine-tuning (Gloaguen et al., 2025), and less easily flagged by attackers (Dathathri et al., 2024).

Let V be a token vocabulary. We can formalize a language model as a function $p_\theta : V^{(t-1)} \to V$, parameterized by θ, mapping the first $(t-1)$ tokens to a probability distribution over V. We focus here on autoregressive systems, where the text is progressively completed, token after token, on a probabilistic basis given the prompt and the recent context (e.g. specialized corpus, or chat history with the user). Given a prompt x and previously generated tokens $v_1, \ldots, v_{t-1} \in V$, the model defines the conditional distribution $p(v \mid x \cup v_1, \ldots, v_{t-1})$ from which the next token v_t is selected. We highlight four stages where watermarking may intervene, summarized in Figure 2:

Into LLM Architecture: steps 1 & 2

Step 1: LLM Training

The first family of in-processing watermarks is introduced *during training*, so that the learned parameters themselves encode the signature. One route is distillation (Gu et al., 2023): a teacher model is configured to generate text with a watermarking bias (see Step 3 below), and a student model is trained to imitate the teacher. More precisely, Gu et al. (2023) study two variants of this idea. In

logits-based watermark distillation, the teacher is frozen and the student is trained to match, at each prefix $x_{<t}$, the teacher's *watermarked* next-token distribution, i.e. the distribution obtained after applying the Step 3 decoding-time watermark.

In *sampling-based watermark distillation*, by contrast, the teacher first generates synthetic watermarked continuations under the same Step 3 procedure, and the student is then fine-tuned on these sequences with the standard autoregressive cross-entropy loss. The former is more direct but requires aligned vocabularies and tokenizers between teacher and student, whereas the latter is architecturally more flexible but less sample-efficient because the watermarked training corpus must first be generated autoregressively. In both cases, if distillation succeeds, the student reproduces the same detectable statistical bias under ordinary sampling.

Another route is to incorporate watermarking into Reinforcement Learning (RL) objectives. In Xu et al. (2024), the watermark is learned jointly with the generator during policy optimization. Concretely, Xu et al. (2024) cast watermark detection as a learned detector D_ϕ, paired with a watermarked policy π_θ, and embed both inside an RL with Human Feedback (RLHF) training loop building on Ouyang et al. (2022). The detector plays the role of an additional reward model: given a prompt-output pair (x, y), it returns a score $D_\phi(x, y)$ measuring how likely y is to have been produced by the watermarked model. Training then alternates between two updates. First, with D_ϕ held fixed, the policy π_θ is optimized with Proximal Policy Optimization (PPO) so as to increase the detector's score on its own generations, while a KL-regularization term penalizes excessive deviation from a reference policy π_0 in order to preserve fluency and utility. Second, with π_θ held fixed, the detector is retrained to discriminate the current model's outputs from non-watermarked reference text.

STEP 2: WEIGHTS EDITING

A second family operates *after* the base model is trained, by inserting a signature directly into parameters. At a high level, weights can be edited in some or all layers, potentially across multiple releases of the same model (Bansal et al., 2022). These edits typically take the form of adding small, structured biases or noise that is intended to reliably (yet subtly) influence future generations.

Zhang and Koushanfar (2024) focus on salient parameters, identified where activation ranges show large magnitude gaps (Lin et al., 2024). They add a small bias ξ to selected parameters θ at random positions, i.e. $\theta \to \theta + \xi$, and the key ξ is required for detection. Building on Li et al. (2023), the authors further quantize the model so that attacks relying only on the int8 version become less effective.

Block et al. (2025) similarly inject a slight bias into model parameters, again controlled by a provider-held key ξ, but without requiring that the distributed model be quantized. Rather than

selecting parameters by activation magnitude, the authors empirically choose a single Multi-Layer Perception (MLP) within a Transformer block, placed before the final activation function, in order to preserve prediction quality. The bias is sampled from a zero-mean normal distribution with small variance $\sigma > 0$, so that the perturbation negligibly affects outputs while remaining statistically recoverable:

$$\xi \sim \mathcal{N}(0,\, \sigma^2 I)$$

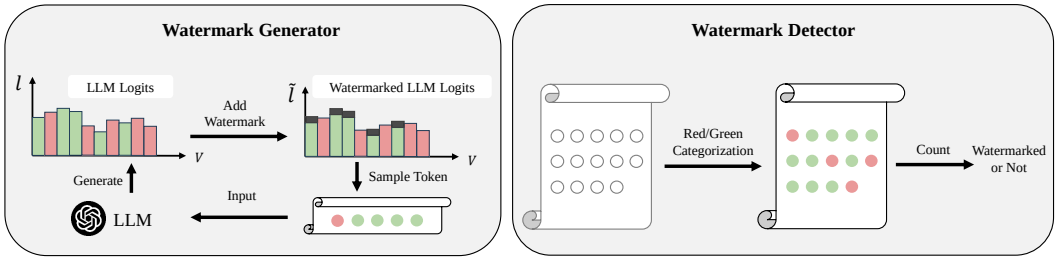
STEP 3: NEXT-TOKEN DISTRIBUTION

Outside of training and parameter editing, watermarking can intervene *during decoding* by biasing the next-token distribution. In autoregressive generation, logits are converted via a softmax into probabilities for each candidate token at position t, conditioned on the prompt x and prefix $v_1, \ldots, v_{t-1}$. Many prominent schemes operate exactly at this stage (Step 3 in Figure 2).

A canonical example is the watermarking scheme introduced by Kirchenbauer et al. (2023). At each decoding step, the method computes a hash of the immediately preceding token and uses that hash to seed a pseudo-random number generator. This seeded generator then induces a reproducible partition of the vocabulary V into two complementary subsets: a preferred "green" list and a disfavored "red" list. In the soft version of the method, the green list occupies a fixed proportion of the vocabulary, and a positive additive bias is applied to the logits of all green-list tokens prior to the softmax transformation.

As Figure 3 highlights, the green tokens of V are all added a slight bias, hence a chance to be selected as the next token. As a result, red tokens are not strictly prohibited; rather, the sampling distribution is gently tilted toward green tokens while preserving fluent generation. Because this bias is small at the level of any single step but accumulates across many steps, sufficiently long outputs exhibit a statistically significant over-representation of green tokens, thereby enabling post hoc detection through a hypothesis test on the observed green-token frequency (see Figure 3, detector).

Figure 3: Biasing the Next-Token Distribution: The Example of Kirchenbauer et al. (2023)



Aaronson and Kirchner (2022) also modify the next-token distribution, using a Gumbel-softmax construction. The Gumbel rule selects a stochastic subset of candidate points whose logits are

converted into probabilities, enabling the watermark while aiming to remain *distortion-free* with respect to perceived text quality.

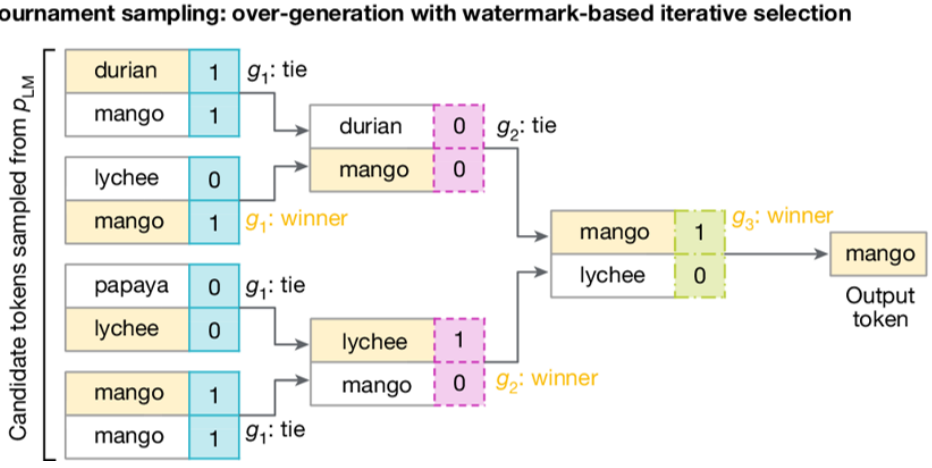
STEP 4: NEXT-TOKEN SAMPLING

Finally, between the probability distribution and the emitted token, the sampling strategy itself provides another intervention point (Step 4). Because decoding can be greedy, beam-based, or stochastic (Christ et al., 2024; Aaronson and Kirchner, 2022), watermarking can bias the selection rule at this last internal step, without directly altering the underlying distribution.

The SynthID method of DeepMind (Dathathri et al., 2024) departs from direct logit biasing and instead combine context-dependent pseudorandom scoring with a tournament-based sampling procedure. At generation step t, a random seed r_t, derived from the recent context $x_{<t}$ together with the watermarking key, is used to instantiate $m \in \mathbb{N}$ independent watermarking functions $g_1, \ldots, g_m$. Each function assigns a pseudorandom score $g_\ell(x, r_t)$ to every candidate token $x \in V$.

The candidate tokens, being among the most likely to be sampled as they follow the next-token distribution of the initial model $p_{\mathrm{LM}}(\cdot \mid x_{<t})$, enter into competition. Candidate-tokens are randomly partitioned into pairs and compared in successive knockout rounds. In Figure 4, the different scoring functions are in blue (round 1), pink (round 2), and green (round 3). In the first round, the winner in each pair is the token with the higher score under $g_1(\cdot, r_t)$. The surviving tokens are then re-paired, and the same procedure is repeated at round ℓ using $g_\ell(\cdot, r_t)$. After m such rounds, only one token remains; this final survivor is emitted as the next generated token x_t.

Figure 4: Watermarking combining next-token distribution and sampling (Dathathri et al., 2024)



3 EVALUATION CRITERIA FOR WATERMARKING TECHNIQUES

Once watermarking and detection mechanisms are implemented, they are typically evaluated along three main axes: detectability, robustness, and impact on generation quality. In practice, these

criteria are tightly coupled, and most empirical protocols expose trade-offs between them. In this section we summarize standard evaluation methods and common metrics, drawing on surveys and toolkits of Liu et al. (2024b); Zhao et al. (2025); Pan et al. (2024). We also add the fourth criterion of interoperability, which is explicitly promoted in the European regulatory approach to AI.

3.1 Ensuring watermarks' detectability

Because the primary purpose of a watermark is to authenticate synthetic text, it must be statistically recognizable (Giboulot and Furon, 2024). Most methods therefore separate an *embedding* step (in data, weights, logits, or outputs) from a *detection* step that tests whether a candidate text is watermarked. In zero-bit settings (where the watermark only indicates that the piece of content is LLM-generated), detection is a hypothesis test; in multi-bit settings (where the watermark bears more information, e.g. on the author and date of AI generation), the detector attempts to decode an embedded message. In both cases, evaluation reports true/false positive rates, the required sample length for reliable detection, and a significance threshold (e.g. p-value or z-score) that controls the false-alarm rate (Liu et al., 2024b).

Figure 5: Detecting a watermark in a tested output (Kirchenbauer et al., 2023)

Prompt	Num tokens	Z-score	p-value
…The watermark detection algorithm can be made public, enabling third parties (e.g., social media platforms) to run it themselves, or it can be kept private and run behind an API. We seek a watermark with the following properties:			
No watermark Extremely efficient on average term lengths and word frequencies on synthetic, microamount text (as little as 25 words) Very small and low-resource key/hash (e.g., 140 bits per key is sufficient for 99.999999999% of the Synthetic Internet	56	.31	.38
With watermark - minimal marginal probability for a detection attempt. - Good speech frequency and energy rate reduction. - messages indiscernible to humans. - easy for humans to verify.	36	7.4	6e-14

Figure 5 illustrates hypothesis testing over an interval of suspect tokens. Given access to the watermarking key (i.e. the hash function and random number generator), any third party can reconstruct the red/green token coloring (Kirchenbauer et al., 2023). They can then test whether the proportions of red and green tokens are statistically balanced ("no watermark" in Figure 5) or whether the "with watermark" condition disproportionately favors green tokens, indicating use of the tested LLM.

A recurring practical issue is *minimum text length*. For distribution-based schemes such as the "green tokens" approach (Kirchenbauer et al., 2023), detection requires sufficiently long continuations for the watermark bias to concentrate, and several works study how detection degrades on short excerpts

or chat-style turns (Piet et al., 2025; Kirchenbauer et al., 2024). Detectability is also sensitive to decoding settings (temperature, nucleus sampling, repetition penalties) and to prompt constraints, which can reduce the degrees of freedom available for the watermark to manifest. Detectability is usually quantified by the false-positive rate (flagging human text) and the false-negative rate (missing a watermarked sample) at fixed significance levels. Protocols often report ROC curves, p-values from binomial or permutation tests, and the stability of detection under varied sampling policies (McNicol, 2005).

Beyond English, we note that the literature on LLM watermarking remains comparatively underexplored (Ghanim et al., 2025). Among the few available contributions, Jeon et al. (2026) combine syntactic and semantic anchors that alter the generated text without modifying the underlying model, achieving high F1 and ROC-AUC scores. However, such approaches are even more vulnerable to translation- or paraphrase-based attacks, which can reduce detection performance to the level of pure chance (He et al., 2024). This leaves substantial open ground for research on robustness and detectability in lower-resource and less-represented language corpora.

3.2 Ensuring watermarks' robustness

The criterion of robustness checks whether the watermark can survive benign transformations or malicious laundering. Symmetric robustness criteria hang on the will of malicious actors to *extract* (e.g. wrongly attributing a toxic content to any LLM provider) or to *erase* it (e.g. for plagiarism, cf. Table 1).

Table 1: Defending the LLM: Two Criteria to make the Watermark robust

Watermark robustness Criteria	Target of the Attacker	Malicious Attacking Use
Non-Extractable	Watermarking method	Attackers can insert their content, pretending to be generated by the defamed LLM
Non-Erasable	LLM-generated text	Attackers can insert LLM-generated text, pretending to be their own

For text, common transformations include paraphrasing, translation, summarization, copy-editing, and passage-level reordering; adversarial transformations include targeted rewriting by another LLM, insertion of distractor tokens, or re-generation under different decoding policies (Wu et al., 2023; Liu et al., 2024b). Evaluation therefore typically benchmarks the detector under a battery of attacks, reporting how detection accuracy decays as a function of edit distance or semantic drift. Recent analyses further highlight semantic attacks that preserve meaning while breaking surface statistics, which are particularly damaging to output-level watermarks (Han et al., 2025).

Figure 6: Paraphrasing Attacks on LLM Watermarks in Education: extracting and erasing examples

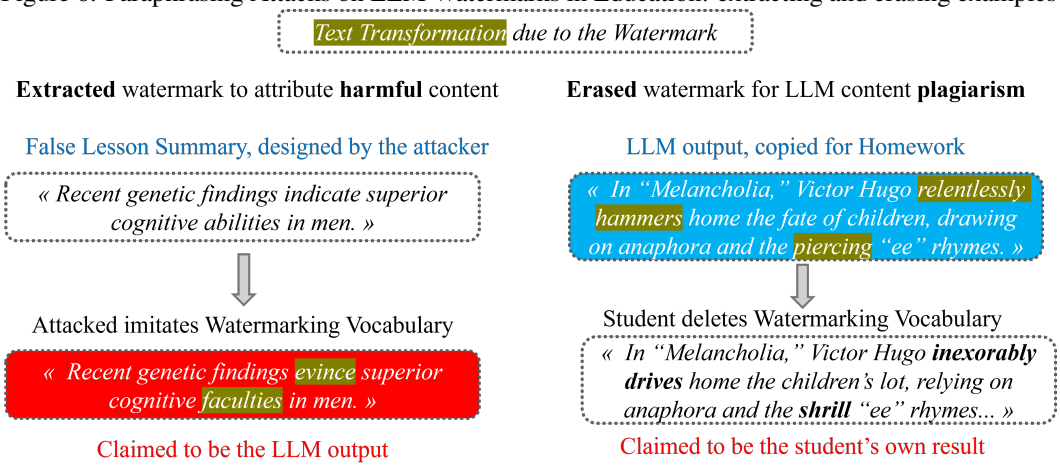


Figure 6 illustrates these two lines of attack through a schematic example in the context of higher education. LLMs are beginning to be deployed through formal partnerships with educational institutions—for instance, to support administrative planning or to assist teachers in developing specialized instructional materials for students with disabilities. On the student side, unofficial uses are also documented, including assistance with homework, seeking psychological advice, and even real-time verification of teachers' statements through phone-integrated API (Wang and Fan, 2025).

Because LLMs are increasingly used to support teachers in the preparation of pedagogical materials, falsely attributing problematic outputs to these internal systems may carry significant geopolitical value. An attacker with knowledge of the synonym set or the red/green token distribution - for example, obtained through API calls - could introduce false content, such as an alleged hierarchy between genders (left side of Figure 6), thereby undermining the institutional LLM used in education. Conversely, targeted or random substitutions could allow sophisticated cheaters to erase the traces of a marked LLM output, for instance in a dissertation assignment (right side of Figure 6).

Recent attacks emphasize *semantic laundering*: using another LLM to paraphrase while preserving meaning, or performing targeted edits that keep topical words while disrupting the detector's statistics. Studies report that such attacks can be effective against several distribution- and post-processing schemes, motivating the evaluation of semantic similarity alongside detection outcomes (Han et al., 2025; Zhao et al., 2023). In addition, detectors should be tested across domains (news, code, chat) and across languages, because tokenization and stylistic constraints can materially change watermark behavior.

Robustness protocols should also reflect realistic post-training and deployment operations. For example, for methods that watermark model weights, tests often include fine-tuning, pruning, quantization, model merging, and distillation. In-processing and structural methods are expected to better

withstand these transformations than post-processing edits, but the evidence remains uneven across model sizes and training regimes (Gloaguen et al., 2025; Block et al., 2025; Gu et al., 2023).

3.3 Maintaining LLM quality

A watermark that is detectable and robust is still unacceptable if it degrades the utility or safety of the model. Quality evaluation therefore measures (i) fluency and likelihood (e.g. perplexity or log-likelihood changes), (ii) task performance on downstream benchmarks, and (iii) human-perceived quality and style consistency (Liu et al., 2024b). For semantic preservation under rewriting attacks or post-processing edits, studies may also use semantic similarity metrics (e.g. Sentence-BERT) alongside n-gram measures such as BLEU (Reimers and Gurevych, 2019; Papineni et al., 2002). Several works additionally explore using LLMs as judges for quality scoring, with the usual caveats about evaluator bias and calibration (Zhao et al., 2023).

Quality is also influenced by where watermarking is applied: logits-biasing approaches (Steps 3–4) can introduce subtle distribution shifts that may be hard to perceive in isolation but can accumulate, especially under high temperature or when the prompt tightly constrains vocabulary (Yoo et al., 2023; Dathathri et al., 2024). Conversely, structural approaches that edit weights or training objectives may preserve output quality while changing internal representations, which motivates complementary evaluation on factuality and truthfulness where available (Chen et al., 2023).

4 Trade-offs for Watermarking Techniques

Recurring tensions run across different LLM watermarking techniques between detectability (strong statistical signal), robustness (survival under paraphrase, re-generation, or post-training edits), and LLM quality (minimal distortion and operational overhead). We explain how these trade-offs manifest themselves in Table 2. This analysis motivates our preference for watermarking inside LLM architecture.

4.1 Pre- and post-processing approaches

Pre- and post-processing methods operate on text or data rather than on the generation mechanism itself. They include hidden-character insertion (Wei et al., 2024; Sato et al., 2023), synonym-based schemes (Zhang et al., 2024), and learned substitution pipelines (Abdelnabi and Fritz, 2021). Because they do not change the base model, they can preserve the *global* quality of outputs and can be retrofitted to existing deployments (Liang et al., 2024). They also offer straightforward detection when the encoding rule is stable and the sample length is sufficient.

Their main weakness is attack surface. Simple laundering operations such as copy-editing, re-tokenization, or automated paraphrase, can remove or dilute the signature at low cost (Liu et al.,

Table 2: **The strengths and weaknesses of LLM watermarking styles.** Green checkmarks indicate criteria met with high confidence, red crosses indicate the contrary. Orange tildes suggest a medium maturity of the technique. The title categories match EU evaluation criteria.

Lifecycle Stage	Watermarking Method	ROBUST		RELIABLE	EFFECTIVE	
		Non-Erasable	Non-Extract.	Detectable	Easy to Implement	Output Quality
Pre- & Post-processing	Direct replacement of words or characters	✗	✗	✓	✓	✓
	Selective replacement (random, contextual...)	✓	✗	✓	✗	✓
Next-Token Distribution	Small Window Size	✓	✗	✓	✓	✗
	Large Window Size	✗	✓	~	~	✓
Next-Token Sampling	Adding bias to specific blocks of tokens	~	✗	✓	✗	✓
	Mapping random numbers to LLM samples	✓	~	~	~	~
Into LLM Architecture	Distillation of a watermarked teacher model	✓	✗	✓	~	✓
	RL with a watermark detector	✓	~	~	~	✓
	Adding Bias on specific LLM weights	✓	~	✓	✓	✓

2024b). Post-processing variants on generated data can be spoofed because attackers can imitate the same transformation on their own content. They can often be neutralized by running the output through another model or mixing it with non-generated data (Xue et al., 2025). More sophisticated schemes that use context-aware or pseudo-random substitutions can improve robustness, but typically increase implementation complexity for instance with Zhang et al. (2024) detector needing to train a classifier to detect watermarking.

Operationally, because the watermark resides in surface edits, detection may require either access to the exact post-processed string (including hidden characters) or sufficiently long passages so that the signal is not drowned by natural language variation. They can therefore be ill-suited to short-form chat, to streaming interfaces, or to settings where downstream platforms normalize whitespace, Unicode, or punctuation (Liang et al., 2024). In regulated contexts, this fragility often translates into a need for layered governance, e.g. combining visible labels with provenance metadata (CAC, 2025).

These post-processing watermarks are easy to add to generated text, but they are also easy to remove. In educational settings, we therefore recommend using them only within a clearly defined scope (e.g., science, technology, engineering, and mathematics courses) and primarily as a pedagogical tool to help students understand what watermarking is.

4.2 In-processing approaches

In-processing approaches intervene inside the inference loop, typically by biasing logits (Step 3) or modifying sampling (Step 4). Distribution-based schemes such as green-token biasing (Kirchenbauer et al., 2023) and Gumbel-based constructions (Aaronson and Kirchner, 2022) can provide strong detectability under controlled decoding settings, because the detector observes a consistent statistical skew. They are also comparatively easy to deploy when the provider controls the serving stack, since they require no change to training data or weights.

However, this control assumption is fragile for open deployments. If users can change temperature, apply custom samplers, or filter logits, the watermark signal can weaken or vanish; similarly, low-entropy prompts and constrained decoding reduce the available degrees of freedom and can harm both detectability and quality (Christ et al., 2024; Kirchenbauer et al., 2024). Moreover, adaptive attackers can paraphrase watermarked text or re-generate it under a different model, which often breaks token-level statistics while preserving meaning. Some recent methods harden the signal through more complex sampling e.g. tournament-style selection (Dathathri et al., 2024), but this can add compute overhead and may still be bypassed by semantic laundering.

From a risk perspective, logits- and sampling-based methods also concentrate trust in the service provider: the detector's reliability depends on the provider's secret key and on faithful implementation in production. This creates governance questions for third-party auditing and for incident

response. For instance, how to rotate keys, how to handle model updates, and how to publish verification guarantees without enabling attacks? Furthermore, these methods often yield a measurable statistical footprint; if the footprint becomes widely characterized, attackers can attempt "anti-watermark" strategies that explicitly steer generation away from the biased subsets (Kirchenbauer et al., 2024).

In the educational context, red/green token coloring could serve as a more elaborated technique to illustrate watermarking for higher-level students (e.g. at the end of high school) in a gamified manner (Bayerl et al., 2026). Outside pedagogy, however, the detectability of next-token logits biasing to flag plagiarism is conditional on sufficient text length and knowledge of the prompt. We recommend to use such methods alongside clear teaching instructions on more than two paragraphs of text. We note that coloring can be detected until $1/4$ of the tokens are modified (Kirchenbauer et al., 2023). More generally, p-values or z-tests are better suited for controlled essay-style evaluation than multiple-choice questions or shorter samples.

4.3 Watermarking inside model architecture

Structural watermarking, introduced during training (Step 1) or by post-training weights editing (Step 2), targets the model rather than individual outputs. Distillation and RL-based training objectives can bake watermark behaviour into the learned parameters (Gu et al., 2023; Xu et al., 2024), while parameter perturbation methods inject a keyed signature into selected weights (Bansal et al., 2022; Zhang and Koushanfar, 2024; Block et al., 2025). In principle, these approaches can be harder to remove by surface-level rewriting and can remain detectable even when decoding policies vary, which makes them attractive for highly secure or open-weights scenarios (Gloaguen et al., 2025).

The trade-off appears between internal robustness and post-training detection. Weight edits and training-time objectives can create subtle, hard-to-predict interactions with downstream fine-tuning, model merging, and alignment pipelines; though, further operations may attenuate the watermark. Empirical results suggest improved resistance to several attacks (Gloaguen et al., 2025; Xue et al., 2025), but the field still lacks large-scale, standardized evaluations across architectures, sizes, and post-training workflows. In addition, while these methods can minimize output distortion by keeping perturbations small, they require careful calibration, key management, and versioning to remain reliable as models evolve (Bontcheva et al., 2025).

A further consideration is what is being authenticated. Structural schemes can support two complementary claims: (i) *text provenance* (a given output likely came from a particular model family) and (ii) *model provenance* (a released checkpoint carries a detectable signature tied to a provider). In deployments with strong emphasis on safety, the second claim can be strategically valuable: it helps track redistribution, unauthorized forks, or derivative checkpoints, even when outputs are para-

phrased. Yet, the same openness that motivates these schemes also exposes them to strong attacks (aggressive fine-tuning, merging, or distillation), so evaluations must explicitly include these operations rather than only output-level paraphrase tests.

In the educational setting, consider the common case of a LLM deployed internally by an institution—for example, using RAG over teachers' course materials to generate assessments or assisting with timetable and agenda generation. When an anomaly is detected, such as nonsensical class schedules, racist content, or anachronisms in a history quiz, an architecture-level watermark can help establish whether the output indeed originated from the institution's system, thereby exonerating administrative teams or teachers when they could have been wrongly blamed.

On the student side, academic integrity enforcement is more challenging, since students typically rely on APIs provided by large external models that are not operated by their school. Nevertheless, if models are systematically watermarked at a national level e.g. in France, then in serious cases of fraud—such as misconduct in the baccalauréat (the end-of-secondary-school examination and gateway to university) it remains feasible to rely on a trusted public audit platform that aggregates watermarking keys and detectors for all LLMs (e.g. under the auspices of the CNIL, ARCEP, or another competent regulator). Such a mechanism would generally be more reliable than using a detector that operates solely from model outputs, without access to internal watermarking material.

5 Recommendations on watermarking

International AI fora and European regulation increasingly emphasize that LLMs should be open access, use less computational resources, and be compatible with current industrial practices (EU, 2024; Bontcheva et al., 2025). Based on the above analysis, we recommend to follow *structural* watermarking - embedding the watermark in the model through training-time objectives or post-training parameter edits (Steps 1–2 of Figure 2) - as a particularly promising direction. This technique can, in principle, yield stronger robustness and lower dependence on inference-time settings while preserving output quality. At the same time, empirical evidence on evaluating this technique remains limited, and substantial benchmarking is still required. Below, we briefly present several arguments in favor based on international regulations and standards, focusing on feasibility under open-weights and low computational cost during deployment constraints.

5.1 Controlled watermarking strategies

For robust and interoperable deployments, we recommend a *multilayered marking strategy*: rather than betting everything on a single signal, combine complementary mechanisms so that attackers must defeat multiple, partially independent checks. The choice of layers depends on whether the institution prioritizes an open-access LLM or a proprietary LLM.

LLM providers serving public institutions may choose an open-weights strategy to promote transparency and let their customers adapt models through fine-tuning or RAG. For these purposes, an open-weights model should be watermarked in a qualitative, detectable, and robust way (see Section 5.2). In the French educational system, the choice may instead be made in favor of a closed-weights proprietary model to prioritize safety. This does not preclude one from maintaining a frugal strategy (see Section 5.3).

If the LLM provider releases an open-weights model to promote trust through transparency (Oliver and Bommasani, 2025), it inherits the strengths and weaknesses of structural watermarking under open-weight attacks. In that scenario, it may be valuable to complement watermarking with non-watermark provenance mechanisms. For example, the industrial standard C2PA emphasizes signed metadata and provenance assertions, and also discusses fingerprinting-style approaches (C2PA, 2025). Although metadata is easily stripped and fingerprinting-style guarantees remain far more mature for images than for text (Evennou et al., 2024), these layers can still preserve contextual evidence about how content was produced even when the invisible watermark is uncertain.

If the provider operates the model as a managed service or within a trusted environment, it may add inference-time watermarking (Steps 3–4) as a second line of protection. One plausible stack is a light structural signature in the weights (e.g. Gaussian noise) plus a logits or sampling-based bias during generation (Dathathri et al., 2024). This does not guarantee that every layer will remain detectable under all attacks, but it increases attacker cost by forcing attackers to both launder surface statistics and defeat model-level signatures. This multi-layer strategy can be usefully complemented by non-watermark provenance mechanisms—such as signed metadata and cryptographic provenance—to preserve contextual evidence even when watermark signals are ambiguous or have been removed.

Whatever strategy an institutional actor adopts, the solution must remain *operationally realistic*. Beyond the classical trade-offs between robustness, detectability, and preserving LLM quality, a European provider should also implement interoperability requirements. We propose the following translation of interoperability into practice, based on a previous suggestion by one of the authors (Souverain, 2025):

- *Information exchange.* To enable a global verifier (e.g. on the Chinese or EU market), detection conditions must be standardized and reported: minimal text length for stable detection, ROC-style performance at fixed false-positive rates, accepted decoding ranges, and other settings when relevant (e.g. context window sizes for learned patterns or inference time biasing). Research should also test how multiple signals combine: for instance, how to weigh high-confidence logits-based detection (potentially extractable) against lower-confidence evidence from weight edits (potentially attenuated in open-weight forks). As a conservative rule, we recommend declaring content "marked" only when *positive indi-*

cators align; partial indicators should trigger softer outcomes (e.g. requests for additional context) rather than definitive attribution.

- *Operational environments and audit rights.* A teacher suspecting plagiarism or a banker suspecting fraud may need to query a verification service. Under a rigorous reading of the EU AI Act and the draft code of practice (EU, 2024; Bontcheva et al., 2026), independent auditors should be able to access verification procedures and, when necessary, detection keys under controlled governance conditions (e.g. escrow, regulated access, or secure enclaves).

- *Decision support, not automated truth.* Verification should be framed as a decision aid: verifying a watermark helps professionals decide whether to pursue an investigation about origin, author, and context, especially when human-written and machine-generated passages are mixed (Xue et al., 2025). Even when all indicators are positive, the final judgment should be performed by humans exercising critical thinking and context-awareness. This requires user-friendly explanations of the checks on multi-layered indicators to support human judgment in a clear way.

  To avoid overconfidence, it is mandatory to notify users that they interact with a watermarked LLM. This is already mandatory in Europe (EU, 2024) and China (CAC, 2025). However, purely informing the human user is insufficient to make them realize all possible consequences of their interaction with an AI system. Therefore, we recommend to pursue AI literacy and develop specific pedagogical material on watermarks, making it available to the educational institutions and citizens. Moreover, we recommend to explore gamification or other frameworks that may allow the user to contain implicit manipulation effects (Bayerl et al., 2026).

- *Frugality and compute constraints.* Robustness requirements are compatible with frugal verification. Weight-editing approaches that avoid full retraining (Step 2) may be preferable when compute is constrained, while fingerprinting and metadata layers, and lightweight inference-time biases (Kirchenbauer et al., 2023) in closed contexts, can act as supplementary hints. Nevertheless, further work is required to quantify the added computational expense, in regard with the impact quality and the minimal token length to reliably detect a watermark on small inference models (Piet et al., 2025).

As a practical example of a solution that works towards meeting these requirements, Wang et al. (2025a) combine watermarking at the logits level with watermarking at the token-sampling stage. By embedding a dual watermarking signal in each token, they increase overall detectability (TPR), though, even if this occurs at the expense of text quality. Although they suggest exploring additional criteria—such as information gain and signal-to-noise ratio—beyond entropy, they do not

evaluate a combined detectability metric that would explicitly quantify information sharing across signals. A similar limitation appears in a study that combines token-level and text-level watermarks: detectability improves, but the marginal contribution of each component is not disentangled (Jeon et al., 2026).

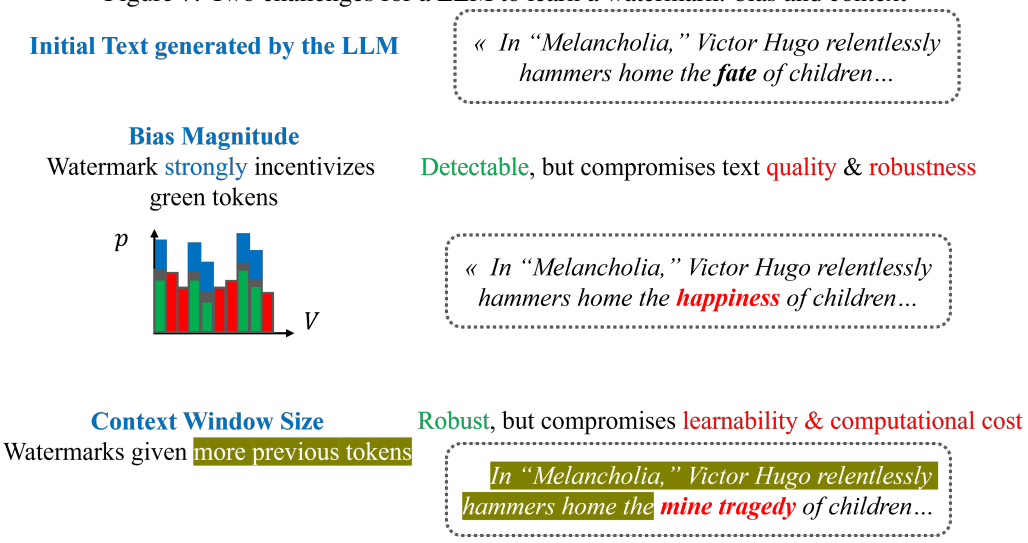
A central challenge in cross-layer information sharing is the practical combination of watermark and non-watermark detectors. Bahri and Wieting (2025) address this with a cascaded approach in which the watermark acts as a fast, specialized filter, while a non-watermark detector serves as a fallback for ambiguous cases. Concretely, when the watermark score (e.g., entropy-based) is very high or very low, the system decides immediately; otherwise, it defers to a second-stage detector based either on the text log-likelihood under a model or on a supervised BERT-like classifier. However, Bahri and Wieting (2025) focus on token-level watermarks and do not specify how watermark detection should be integrated with other provenance techniques.

Overall, multi-layered watermarking that we recommend here is gaining traction as a research topic. Yet principled measures of interaction between signals remain underdeveloped, especially for the combination of token-based, textual and architectural watermarking. One can quantify the detectability of an output watermark, the robustness of a model fingerprint, and the cryptographic validity of a provenance chain (Liang et al., 2024; Wang et al., 2025b); one can also obtain gains through late fusion of detectors (Bahri and Wieting, 2025). By contrast, redundancy, independence, and interference across these layers still lack a unified evaluation framework. This shows that more research is required to turn multi-layered watermarking strategies into fully operational and satisfactory solutions.

5.2 Maintaining detectable signal

Open access to model weights increases transparency and reproducibility, but it complicates detectability of the marking. Maleficent attackers using a LLM as an attack tool may modify temperature, custom samplers, logits processors, or decoding policies. They may use fine-tuning, quantization, merging, distillation or pruning of checkpoints. This may even stem from unintentional modifications that weaken or erase watermarks (Gloaguen et al., 2025). The studies of watermarking in open-weights settings highlight additional trade-offs (Xue et al., 2025). In particular, when the watermark is learned during training (Step 1), the provider must reason not only about detectability in the teacher model, but also about how reliably the student model will inherit the pattern. For distillation and RL-based structural watermarking (Gu et al., 2023; Xu et al., 2024; Xue et al., 2025), two design parameters deserve special attention when training the teacher LLM and designing the RL policy. We illustrate these key design choices in an educational setting using the example of LLM-based summarization of specific history or literature materials (Wang and Fan, 2025):

Figure 7: Two challenges for a LLM to learn a watermark: bias and context



- *bias magnitude*. Increasing the watermark strength (e.g. the logits bias that promotes specific token subsets) typically improves detectability, but can degrade output quality and may amplify odd associations or stylistic artifacts, thereby facilitating reverse engineering and removal (Gloaguen et al., 2025). If the watermark causes the model to over-prefer a token such as "happiness," rather than correctly referring to the "fate" of working children in Victor Hugo's "Melancholia," detectability may come at the expense of an inaccurate pedagogical summary (Figure 7, upper side).

- *context window size.* Increasing the dependence of the watermark on longer n-gram context can improve robustness against reverse engineering (Kirchenbauer et al., 2023), but it also risks making the pattern harder for a student model to learn and for a detector to recover reliably, especially across domains and languages (Zhao et al., 2025). Providing a broader context, including commentary on Hugo's poem, would allow the model to more appropriately characterize the children's fate as a "mine tragedy" (Figure 7, lower panel). However, improving subtlety and robustness against attackers also increases the complexity—and thus the computational burden—required for a student LLM to learn the watermarking signal.

The other way of implementing a watermark inside the architecture of a LLM, weights editing (Step 2), faces its own open-weights challenges. Christ et al. (2024) modify the bias term in a final projection layer, yet several open-weight releases omit this output bias in their published checkpoints (e.g. common Hugging Face variants of LLaMA, Qwen, or Mistral), which constrains where edits can be applied (Xue et al., 2025). Moreover, some detection tests proposed for open-source settings assume access to the original prompt to compare it with a suspected output, as in Block et al. (2025).

However in many real-world investigations (screen captures, forwarded messages, quoted excerpts), the prompt is not available, so detection should not rely on this assumption.

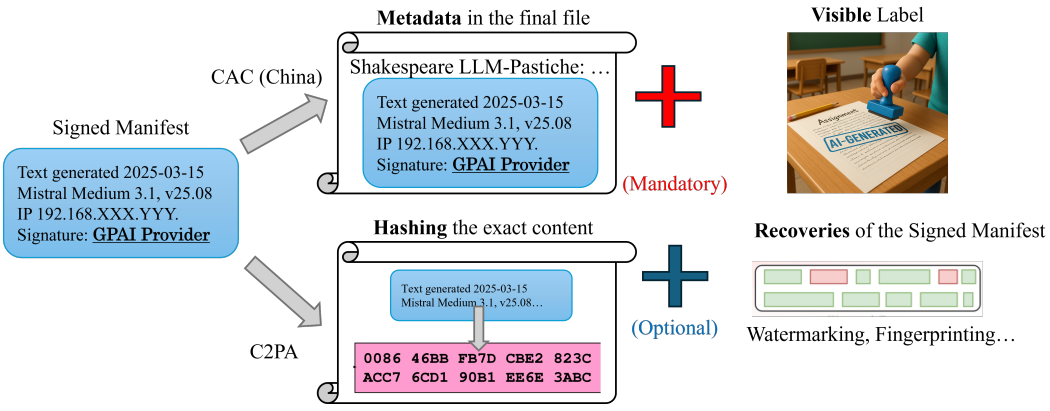
5.3 Interoperability constraints

We finish with operational guidelines on how watermarking could be embedded with the lowest computational cost in closed-weights LLM contents, while being consistent with the most advanced standards and regulations on AI-content marking. While C2PA specification emerged from industrial actors and the September 2025 labeling mandate came from the centralized Cyberespace Administration of China, they both start with the presence of "manifest" information on who, when and how the content was generated - as our Figure 8 schematizes.

C2PA (2025) provides an open technical standard for attaching cryptographically verifiable provenance information to digital media, including images, video, audio, and documents, through so-called *Content Credentials*. These credentials record the origin and history of a file - its creator, the tools and processes used, the time of creation, and any subsequent edits or incorporated elements - within a tamper-evident structure. Its principal advantage is evidentiary: so long as the original asset remains intact, any removal or alteration of the credential is detectable through failed cryptographic validation. C2PA thus offers a particularly robust mechanism for authenticating provenance within a content's native container.

China's *Measures for Labeling of AI-Generated Synthetic Content* CAC (2025), which entered into force in September 2025, follow a different regulatory logic. They establish a dual labeling regime requiring both explicit, user-facing notices and implicit, metadata-based labels for AI-generated content, while also encouraging the use of digital watermarks as a supplementary safeguard. Their chief strength lies less in any single technical device than in the allocation of responsibilities across the dissemination chain: generative AI service providers must label outputs and attach provenance-related metadata; distribution platforms must verify and preserve those labels; app stores must check com-

Figure 8: Schema: How China and C2PA handle LLM Content Labelling



pliance; and users may themselves be required to declare certain AI-generated or suspicious content. The Chinese model is therefore highly operational and enforcement-oriented, albeit correspondingly more rigid and less flexible in implementation.

The recent European draft Code of Practice on the Transparency of AI-Generated Contents (Bontcheva et al., 2025) points in a similar direction, but does so explicitly through a *multi-layered* marking logic. Rather than relying on a single technique, it promotes a stack combining metadata, watermarking, cryptographic provenance methods, and - where appropriate - fingerprinting and logging, depending on format and use case. In the second draft, this approach is streamlined into a baseline requirement of *at least two layers of machine-readable active marking* (secured, digitally signed metadata plus an imperceptible watermark), while fingerprinting/logging are reframed as optional supplementary measures. At the same time, the text introduces an explicit flexibility clause allowing providers to rely on an alternative approach — potentially even a single technique — once they can demonstrate, on the basis of independently verified benchmarks, compliance with the Article 50(2) requirements Bontcheva et al. (2026). Finally, for text outputs, the second draft operationalizes practical limits by requiring reliability to be assessed as a function of *length* (as well as entropy and semantics), and by exempting *very short text* from embedded imperceptible watermarking - although no gauge or idea of what would be a token length for these "short" texts is provided Bontcheva et al. (2026). Overall, this evolution confirms that no single layer is sufficient across all content formats, and that text in particular requires a proportionate, multi-layered design.

Blockchain-inspired methods can already be used to trace model lineage and to establish verifiable links between registered models, copyright claims, and training data (Liu et al., 2024a; Wang et al., 2025b). We note that such approaches to model or training-data ownership can be integrated into a broader traceability pipeline that also includes watermarking of AI-generated content, albeit potentially at the cost of higher training and inference overhead.

These existing frameworks already provide concrete and industrially credible ways of marking AI-generated content. Yet their core documentary layers remain vulnerable to ordinary transformations. In C2PA, hard binding secures the exact native file, but this protection becomes fragile once the content is copied, reformatted, or detached from its original container (for instance, when a PDF is converted into plain text). In the Chinese model, metadata may also be stripped or lost in downstream circulation. China addresses this difficulty through obligations and sanctions imposed on each actor in the distribution chain (Li, 2025), while C2PA addresses it by optionally adding *soft binding* mechanisms for recovery (C2PA, 2025). In our view, and in line with the European draft, an effective European regime should therefore adopt a genuinely multi-layered approach that *explicitly includes watermarking*, despite the fact that such methods are less standardized for textual documents than metadata or exact-file hashing.

We therefore recommend, for proprietary LLMs, a **frugal marking pipeline**, that is, a pipeline that adds only limited costs in inference and storage while preserving meaningful traceability. Pictured with an example in our Figure 9, this pipeline will be compatible with both of the most advanced specifications and regulations on AI content marking, C2PA (2025) and CAC (2025):

1. **Watermarking.** Watermarking should be the first layer, because it is the most resilient to downstream transformations. We recommend combining two forms of watermarking: first, a structural watermark introduced through lightweight modifications during the model lifecycle (for example, targeted weight edits that do not require full retraining); and second, a next-token biasing mechanism at inference time, which reinforces the signal directly in the generated text. Used together, these two methods make the mark both harder to erase and easier to detect.

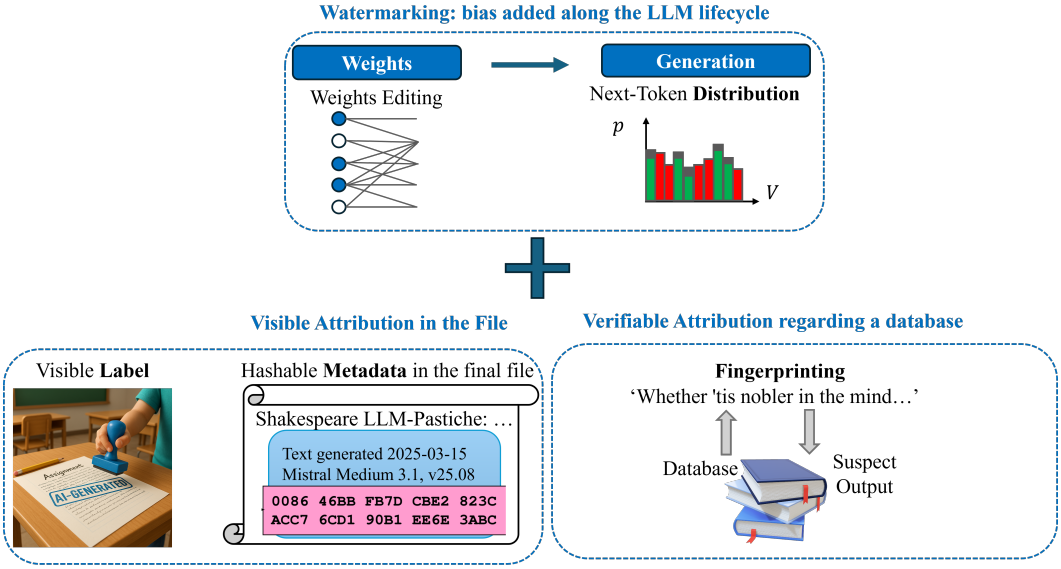
2. **Visible attribution in the file.** Even robust watermarking should be complemented by a visible attribution layer at the level of the final document. For the lowest computational cost, we recommend adding a simple and standardized visible notice of AI generation, in the spirit of the Chinese model (CAC, 2025), and combining it with a unique cryptographic identifier linked to the generation event, in the spirit of C2PA (C2PA, 2025). This gives both immediate human-readable transparency and a standardized anchor for subsequent verification (see Figure 8, lower left side).

3. **Verifiable attribution against a reference database.** As the European drafts suggest, supplementary techniques such as fingerprinting, forensic detection or logging should complete the overall detection pipeline (Bontcheva et al., 2025; 2026). We recommend here a fingerprinting-based solution: rather than storing all outputs or maintaining burdensome logs, each provider could retain a compact reference database enabling suspected content to be compared against prior generations. This offers a practical compromise between verifiability, cost control, and privacy.

   Recent blockchain-based intellectual property protection methods may offer useful inspiration. For instance, (Liu et al., 2024a) propose a blockchain provenance proof that determines whether a suspect text matches content that was generated and previously registered within their provenance system. At generation time, the LLM provider records an on-chain transaction containing a hash-derived product index, metadata, a timestamp, and a digital signature. In the event of suspected plagiarism, the challenger's identity is first authenticated via signature verification, and the alleged original and duplicate are then compared using hashing-based similarity metrics. While robust, this approach, like fingerprinting, presupposes the registration of models and reference data ; we recommend to deploy this side-measure after having carefully investigated its computational cost.

Taken together, these three layers illustrated in Figure 8 - watermarking, visible attribution, and ex post verification - would provide convergent indicators when a platform, institution, or authority (e.g. CNIL or ARCEP at the French level, or national equivalents) submits suspicious content to a common verification interface. Such a system would be effective only if that interface had access, under appropriate safeguards, to the relevant methods and verification keys of each LLM provider, updated on a regular basis. In this way, Europe could move toward a marking regime on the basis of evolving and complementary methods.

Figure 9: Example of frugal marking of a LLM, compatible with C2PA and Chinese standards



6 Conclusion

This white paper reviews the main LLM watermarking techniques and their evaluation with a particular focus on the European transparency requirements. Unlike images and video, textual content can be paraphrased, translated, copy-edited, or re-generated at low cost. Thus current methods rarely provide cryptographic-grade authentication. Notwithstanding, watermarking can improve provenance detection *probabilistically*: it provides actionable evidence for platforms and investigators. For robust and interoperable deployments, watermarking text outputs should be treated as a component of a broader provenance pipeline.

Structural approaches to watermarking, in particular learning a watermark through distillation or Reinforcement Learning or inserting a keyed perturbation into weights, are especially attractive for providers which prioritize security for organizational or national purposes, because they can better survive changes in decoding policies and post-training transformations. The inherent trade-offs between a detectable and robust watermarking, preserving LLM quality, should also be balanced with specific inference or post-training settings.

More broadly, effective governance will require interoperable marking, including shared testing protocols, calibrated detectors, and independent auditing. Accordingly, we recommend multilayered marking strategies based on advanced authentication standards (C2PA, 2025) and regulations (CAC, 2025; EU, 2024): combine a structural signal with inference-time biasing when post-training is controlled, and complement open-weight releases with metadata-based provenance, and visible and verifiable attribution against a reference database where possible. Despite the robustness of our multilayered strategy, verification outputs should be framed as decision support. We advocate for conservative rules that assert that a piece of content has been LLM-generated only when multiple indicators align, validated by human judgement.

Generative AI Usage Statement

ChatGPT 5.2 Thinking and Gemini 3.5 were used to optimize figures and tables, as well as to correct spelling and grammar.

Remerciements

Ce livre blanc est le livrable 3.1 du projet *OpenLLM France* financé par Bpifrance (programme France 2030) via l'appel à projets « Communs numériques pour l'intelligence artificielle générative ».

GLOSSARY

API An Application Programming Interface is a specification of callable endpoints and data formats that enables software systems to interact programmatically. For closed-weights models, APIs serve to expose outputs without distributing model weights. 3, 4, 14, 19, 29

Artificial Intelligence A field of research and engineering that develops computational systems capable of tasks associated with human cognition, such as prediction, reasoning, perception, and language or image generation. 4

bias magnitude In watermarking methods that perturb token selection, the strength of the injected bias (e.g. an additive logit shift applied to a preferred token subset). Larger biases can improve detectability but may degrade text quality and increase vulnerability to reverse engineering. 23

checkpoint Serialized snapshot of a model's state (parameters and often optimizer metadata) saved during training, used to resume interrupted runs, evaluate intermediate models, or branch into fine-tuning. 18, 22, 23, 30

closed-weights A proprietary distribution regime in which a model's trained parameters are not released. Its use typically occurs via hosted services or tightly controlled access through API. 4, 6, 20, 24, 29

context window The maximum number of previous tokens the model can condition on when computing next-token logits. Larger context windows can improve coherence and can affect watermarking schemes that depend on longer histories.. 20, 23

cryptographic provenance A tamper-evident record of how content was created and modified, secured by cryptographic primitives (hashes and digital signatures). It enables verifiers to check integrity and authenticity of provenance claims, provided the content remains bound to its signed record. 4, 20, 25

detectable A property of a watermark indicating that a detector can reliably distinguish marked from unmarked text, typically via a statistical test given a watermarking key (and sometimes a prompt). Practical detectability aims to hold at low false-positive rates on minimal text lengths. 4, 9, 15, 18, 20, 24, 28

distillation Training a compact student model to mimic a larger teacher model's outputs or internal representations. While reducing compute requirements, it transfers the distributions (and potential watermarking effects) produced by the teacher LLM. 3, 4, 8, 14, 22, 28

fine-tuning Additional training of a pre-trained model on task- or domain-specific data to specialize it (e.g. turning an auto-completion model into a chatbot), usually with a smaller learning rate and fewer steps than pre-training. 3, 8, 14, 20, 22, 29, 30

fingerprinting A detection technique supporting fast lookup against reference repositories to assess whether content is known, previously generated, or previously manipulated. 3, 4, 20, 21, 25, 26

forensic detection Detection of AI-generated or manipulated content that does not rely on a mark introduced purposely. For example, forensic methods may attribute a text to an AI generator by exploiting characteristic artifacts or by using trained classifiers. 26

fork In distributed version control, a fork is an independent copy of a code or model repository that enables divergent experimentation, while preserving a link to the upstream project. 18, 20

frugal A LLM engineered for cost-efficient training and inference, aiming to reduce compute, memory footprint, and energy use while maintaining task-appropriate accuracy. In this white paper, it also refers to the additional overhead induced by marking techniques (in inference and storage). 3, 4, 6, 21, 26, 27, 30

interoperable A property of marking and verification mechanisms indicating they can be compared, audited, and deployed across heterogeneous systems and standards. Interoperability requires well-specified protocols (marking layers, thresholds, and reporting) so third parties can verify provenance consistently. 4, 19, 28

Large Language Model Model trained on large-scale text corpora to generate text iteratively. Most modern LLMs rely on Transformer architectures and use an auto-regressive process in which tokens are sequentially produced to answer a prompt.. 3, 4, 30, 31

LLM quality The degree to which a marked model preserves utility, fluency, and task performance relative to the unmarked baseline, while keeping inference-time overhead acceptable. In watermarking evaluations, quality constraints often require LLM to remain frugal, and to keep semantic preservation. 4, 15, 20, 28

logging The practice of recording compact identifiers or summaries of generated content for later verification and audit. 3, 4, 25, 26

logit A real-valued score assigned by a model to each candidate token before normalization by the softmax. Relative logit differences determine the next-token probability distribution used for sampling. 3, 4, 9–12, 15, 17, 18, 20, 22, 23, 31

manifest A structured digital record that bundles provenance assertions for verification. In C2PA-style systems, a manifest can be embedded in, or associated with, an asset to carry signed metadata about its origin and transformations. 24

merging A family of methods that combine two or more model parameter sets (often from separately fine-tuned checkpoints) into a single model, e.g. through weight averaging or task-vector composition, to aggregate capabilities without full retraining. 14, 22

metadata Machine-readable information that describes a digital asset and its production context (e.g. creator, tools, timestamps, and edits). In provenance standards such as C2PA, metadata is packaged into signed assertions to enable downstream verification.. 3, 4, 17, 20, 21, 24–26, 28, 30

open-weights A Large Language Model whose trained parameters are publicly downloadable, enabling local inference and fine-tuning, even when the training data, training code, or licensing terms do not provide full open-source reproducibility. 3, 4, 6, 18–20, 22, 23

pruning Model compression that removes weights, neurons, attention heads, or entire layers deemed redundant, reducing parameter count and latency with limited degradation in performance. 3, 8, 14

quantization Representing weights and/or activations with lower-precision numbers (e.g. 8- or 4-bit) to reduce memory and accelerate inference. This operation is frequently performed to speed up LLM generation under resource constraints. 8, 14, 22

RAG Retrieval-augmented generation combines a generative model with an information retriever over an external corpus. It conditions responses on retrieved passages to improve factual grounding, provenance, and updateability. 19, 20

Reinforcement Learning A learning framework in which an agent learns a policy by maximizing expected cumulative reward from interaction signals. In LLMs, it is commonly used (e.g. with human feedback) to optimize generation toward values and preference-based reward models. 4, 9, 28

robust A property of a watermark indicating resistance to removal and to adversarial laundering. In this report, robustness includes both non-erasability (surviving paste and edits) and non-extractability (resisting spoofing). 4, 6, 13, 15, 19, 20, 24, 26, 28

sampling The procedure used to select the next token from the model's probability distribution (e.g. greedy decoding, top-k, nucleus sampling, or tournament-based selection). 3, 4, 9–11, 13, 17, 20, 30

temperature A decoding parameter that rescales logits before the softmax, controlling randomness in token selection. Higher temperature increases diversity of outputs, while lower temperature makes generation more deterministic. 13, 15, 17, 22

token A discrete unit produced by a tokenizer (e.g. a subword or byte-pair unit) that serves as the basic symbol for Large Language Model training and generation. 5, 7–11, 31

watermarking Techniques leading to a statistically detectable signature of LLM outputs, so that synthetic text can be identified and attributed. A practical watermark aims to balance detectability and robustness under editing while preserving output quality. 3–5, 25

REFERENCES

Scott Aaronson and Hendrik Kirchner. 2022. Watermarking gpt outputs. *Retrieved September* 5 (2022), 2024. https://www.scottaaronson.com/talks/watermark.ppt

Sahar Abdelnabi and Mario Fritz. 2021. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 121–140.

Dara Bahri and John Wieting. 2025. Improving Detection of Watermarked Language Models. *arXiv preprint arXiv:2508.13131* (2025).

Arpit Bansal, Ping-yeh Chiang, Michael J Curry, Rajiv Jain, Curtis Wigington, Varun Manjunatha, John P Dickerson, and Tom Goldstein. 2022. Certified neural network watermarks with randomized smoothing. In *International Conference on Machine Learning*. PMLR, 1450–1465.

P.S. Bayerl, E. Lawlor, M.T. Maris, D. Miorandi, G. Pekšys, M. Bjelica, K. Stojšin, M. Anastasova, O. Smith, A. Henestrosa, I. Yamshchikov, M.A.R. Bak, and B. Akhgar. 2026. *Operational ethics guidelines on use cases related to human behaviour and cognition*. Deliverable D3.1. Horizon Europe project AIOLIA. http://aiolia.eu/wp-content/uploads/2026/02/AIOLIA-D3.1-certified.pdf

Adam Block, Ayush Sekhari, and Alexander Rakhlin. 2025. GaussMark: A Practical Approach for Structural Watermarking of Language Models. arXiv:2501.13941 [cs.CR] https://arxiv.org/abs/2501.13941

Kalina Bontcheva, Dino Pedreschi, Christian Riess, Anja Bechmann, Giovanni De Gregorio, and Madalina Botan. 2025. First draft Code of Practice on transparency of AI-generated content. *European Commission Digital Strategy, AI Office Document* (2025). https://ec.europa.eu/newsroom/dae/redirection/document/123074

Kalina Bontcheva, Dino Pedreschi, Christian Riess, Anja Bechmann, Giovanni De Gregorio, and Madalina Botan. 2026. Second draft Code of Practice on transparency of AI-generated content. *European Commission Digital Strategy, AI Office Document* (2026). https://ec.europa.eu/newsroom/dae/redirection/document/125388

Charles-Moïse Briquet. 1907. *Les Filigranes. Dictionnaire historique des marques du papier dès leur apparition vers 1282 jusqu'en 1600 avec 39 figures dans le texte et 16 112 fac-similés de filigranes*. Paris : Alphonse Picard et fils. https://archive.org/details/BriquetLesFiligranes1 Accessed: 2025-08-04.

C2PA. 2025. Coalition for Content Provenance and Authenticity (C2PA), C2PA Technical Specification. https://spec.c2pa.org/specifications/specifications/2.1/specs/C2PA_Specification.html

CAC. 2025. Cyberspace Administration of China (CAC), Notice on Issuing the Measures for Labeling of AI-Generated Synthetic Content. https://www.cac.gov.cn/2025-03/14/c_1743654684782215.htm Accessed: 2025-07-02.

California. 2024. California Senate, SB-942 California AI Transparency Act (2023–2024). https://leginfo.legislature.ca.gov/faces/billNavClient.xhtml?bill_id=202320240SB942. Approved by Governor September 19, 2024; Effective January 1, 2026; Accessed: 2025-07-31.

Liang Chen, Yatao Bian, Yang Deng, Deng Cai, Shuaiyi Li, Peilin Zhao, and Kam-Fai Wong. 2023. WatME: Towards lossless watermarking through lexical redundancy. *arXiv preprint arXiv:2311.09832* (2023).

Miranda Christ, Sam Gunn, and Or Zamir. 2024. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*. PMLR, 1125–1139.

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

EU. 2024. European Union (EU), Regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act). *Regulation (EU) 2024/1689* (2024). https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng

Gautier Evennou, Vivien Chappelier, Ewa Kijak, and Teddy Furon. 2024. Swift: Semantic watermarking for image forgery thwarting. In *2024 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 1–6.

Pierre Fernandez, Hady Elsahar, Sylvestre-Alvise Rebuffi, Tomas Soucek, Valeriu Lacatusu, Tuan Tran, and Alexandre Mourachko. 2025. A Taxonomy of Watermarking Methods for AI-Generated Content. In *The 1st Workshop on GenAI Watermarking*.

Mansour Al Ghanim, Jiaqi Xue, Rochana Prih Hastuti, Mengxin Zheng, Yan Solihin, and Qian Lou. 2025. Evaluating the robustness and accuracy of text watermarking under real-world cross-lingual manipulations. *arXiv preprint arXiv:2502.16699* (2025).

Eva Giboulot and Teddy Furon. 2024. WaterMax: breaking the LLM watermark detectability-robustness-quality trade-off. *Advances in Neural Information Processing Systems* 37 (2024), 18848–18881.

Thibaud Gloaguen, Nikola Jovanović, Robin Staab, and Martin Vechev. 2025. Towards watermarking of open-source llms. *arXiv preprint arXiv:2502.10525* (2025).

Alexei Grinbaum and Laurynas Adomaitis. 2022. The ethical need for watermarks in machine-generated language. *arXiv preprint arXiv:2209.03118* (2022).

Chenchen Gu, Xiang Lisa Li, Percy Liang, and Tatsunori Hashimoto. 2023. On the learnability of watermarks for language models. *arXiv preprint arXiv:2312.04469* (2023).

Xia Han, Qi Li, Jianbing Ni, and Mohammad Zulkernine. 2025. Robustness Assessment and Enhancement of Text Watermarking for Google's SynthID. *arXiv preprint arXiv:2508.20228* (2025).

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 4115–4129.

United States White House. 2025. White House Releases New Policies on Federal Agency AI Use and Procurement. https://www.whitehouse.gov/articles/ Accessed: 2025-07-02.

Dard Hunter. 1967. *Hand Made Paper and Its Water Marks*. Franklin.

Heejeong Jeon, Minsu Park, YunSeok Choi, and Eunil Park. 2026. Unsupervised Detection of LLM-Generated Text in Korean Using Syntactic and Semantic Cues. In *Findings of the Association for Computational Linguistics: EACL 2026*. 1504–1518.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A Watermark for Large Language Models. In *Proceedings of the 40th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 202)*, Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 17061–17084. https://proceedings.mlr.press/v202/kirchenbauer23a.html

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. 2024. On the Reliability of Watermarks for Large Language Models. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=DEJIDCmWOz

Linyang Li, Botian Jiang, Pengyu Wang, Ke Ren, Hang Yan, and Xipeng Qiu. 2023. Watermarking llms with weight quantization. *arXiv preprint arXiv:2310.11237* (2023).

Wenlong Li. 2025. Accountability and Attribution in AI-Generated Content Authentication: Lessons from China's AI Content Labelling Mandate. *The Paris Journal on AI & Digital Ethics* (2025). doi:10.65701/q9m2g5d8x1

Yuqing Liang, Jiancheng Xiao, Wensheng Gan, and Philip S Yu. 2024. Watermarking techniques for large language models: A survey. *arXiv preprint arXiv:2409.00089* (2024).

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems* 6 (2024), 87–100.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. 2024b. A survey of text watermarking in the era of large language models. *Comput. Surveys* 57, 2 (2024), 1–36.

Yinqiu Liu, Hongyang Du, Dusit Niyato, Jiawen Kang, Zehui Xiong, Chunyan Miao, Xuemin Shen, and Abbas Jamalipour. 2024a. Blockchain-empowered lifecycle management for AI-generated content products in edge networks. *IEEE Wireless Communications* 31, 3 (2024), 286–294.

Don McNicol. 2005. *A primer of signal detection theory*. Psychology Press.

Narendra Modī. 2026. India AI Impact Summit 2026, Inaugural Speech of Narendra Modī. https://www.pmindia.gov.in/en/news_updates/pm-inaugurates-india-ai-impact-summit-2026/

New York. 2024. New York Senate Bill S7592A: Requires disclosure of the use of artificial intelligence in political communications. https://www.nysenate.gov/legislation/bills/2023/S7592/amendment/A. 2023–2024 Legislative Session, Sponsor: Senator Jake Ashby; Accessed: 2025-07-31.

Nuria Oliver and Rishi Bommasani. 2025. Code of Practice for General-Purpose AI Models. Transparency Chapter. *European Commission Digital Strategy, AI Office Document.* (2025). https://digital-strategy.ec.europa.eu/en/policies/contents-code-gpai

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, et al. 2024. Markllm: An open-source toolkit for llm watermarking. *arXiv preprint arXiv:2405.10051* (2024).

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

Julien Piet, Chawin Sitawarin, Vivian Fang, Norman Mu, and David Wagner. 2025. MARKMyWORDS: Analyzing and Evaluating Language Model Watermarks. In *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, 68–91.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

Ryoma Sato, Yuki Takezawa, Han Bao, Kenta Niwa, and Makoto Yamada. 2023. Embarrassingly simple text watermarks. *arXiv preprint arXiv:2310.08920* (2023).

Thomas Souverain. 2025. Watermarking Large Language Models in Europe: Interpreting the AI Act in Light of Technology. arXiv:2511.03641 [cs.CR] https://arxiv.org/abs/2511.03641

Jin Wang and Wenxiang Fan. 2025. The effect of ChatGPT on students' learning performance, learning perception, and higher-order thinking: insights from a meta-analysis. *Humanities and Social Sciences Communications* 12, 1 (2025), 1–21.

Qin Wang, Guangsheng Yu, Yilin Sai, HMN Dilum Bandara, and Shiping Chen. 2025b. Is your AI truly yours? Leveraging blockchain for copyrights, provenance, and lineage. *IEEE Transactions on Services Computing* (2025).

Yidan Wang, Yubing Ren, Yanan Cao, and Binxing Fang. 2025a. From trade-off to synergy: A versatile symbiotic watermarking framework for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 10306–10322.

Washington. 2025. Washington State Legislature, HB 1170 - 2025-26: Informing users when content is developed or modified by artificial intelligence. https://app.leg.wa.gov/billsummary?BillNumber=1170&Initiative=false&Year=2025. Requires providers of generative AI systems to include detection tools and enable disclosure options for AI-generated or modified content; referred to House Rules Committee as of January 31, 2025; Accessed: 2025-07-31.

Johnny Tian-Zheng Wei, Ryan Yixiang Wang, and Robin Jia. 2024. Proving membership in LLM pretraining data via data watermarks. *arXiv preprint arXiv:2402.10892* (2024).

Joseph Weizenbaum. 1966. ELIZA—a computer program for the study of natural language communication between man and machine. *Commun. ACM* 9, 1 (Jan. 1966), 36–45. doi:10.1145/365153.365168

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. 2023. A resilient and accessible distribution-preserving watermark for large language models. *arXiv preprint arXiv:2310.07710* (2023).

Xiaojun Xu, Yuanshun Yao, and Yang Liu. 2024. Learning to watermark llm-generated text via reinforcement learning. *arXiv preprint arXiv:2403.10553* (2024).

Jiaqi Xue, Yifei Zhao, Mansour Al Ghanim, Shangqian Gao, Ruimin Sun, Qian Lou, and Mengxin Zheng. 2025. PRO: Enabling Precise and Robust Text Watermark for Open-Source LLMs. arXiv:2510.23891 [cs.CR] https://arxiv.org/abs/2510.23891

KiYoon Yoo, Wonhyuk Ahn, Jiho Jang, and Nojun Kwak. 2023. Robust multi-bit natural language watermarking through invariant features. *arXiv preprint arXiv:2305.01904* (2023).

Ruisi Zhang, Shehzeen Samarah Hussain, Paarth Neekhara, and Farinaz Koushanfar. 2024. {REMARK-LLM}: A robust and efficient watermarking framework for generative large language models. In *33rd USENIX Security Symposium (USENIX Security 24)*. 1813–1830.

Ruisi Zhang and Farinaz Koushanfar. 2024. EmMark: Robust watermarks for IP protection of embedded quantized large language models. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*. 1–6.

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. 2023. Provable robust watermarking for ai-generated text. *arXiv preprint arXiv:2306.17439* (2023).

Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairoze, Andres Fabrega, Nicholas Carlini, Sanjam Garg, Sanghyun Hong, Milad Nasr, Florian Tramer, Somesh Jha, Lei Li, Yu-Xiang Wang, and Dawn Song. 2025. SoK: Watermarking for AI-Generated Content. arXiv:2411.18479 [cs.CR] https://arxiv.org/abs/2411.18479