Available Online: 24 November 2026 DOI: 10.54254/2977-3903/2026.39345

Large language model watermarking: a survey of techniques, attacks, evaluation, and future directions

Haolin Huang

School of Computer Science, Shanghai Jiao Tong University, Shanghai, China

harley-h@sjtu.edu.cn

Abstract. This paper presents a comprehensive survey of different researchs on large language model text watermarking. This survey reads 29 related works in depth and organises the field by the place where the watermark enters the generation pipeline: the model's weights, the decoding step, or a rewrite of text outside the model, subdividing decoding-time schemes by the mechanism that carries the signal. It argues that the insertion point can be viewed the one of the most important properties, deciding a scheme's detector requirements, its attack surface, and the deployment settings open to it. Additionally, this survey summarizes existing watermark attacks, which are organised by adversarial operation and quantified budget, from paraphrasing and translation to text mixing, key stealing, and model distillation. Because the three benchmarks surveyed are based on mutually incompatible axes, no cross-paper ranking is reported here. Finally, this paper concludes the key open challenges in current research and possible future research directions. This survey intends to provide researchers and engineering practitioners with a concise, systematic and in-depth understanding of the latest progress in Large Language Model (LLM) watermarking, and further facilitate the development of trustworthy generative artificial intelligence systems.

Keywords: large language model watermarking, machine-generated text detection, watermark robustness and attacks, evaluation comparability

1. Introduction

Nowadays, text generated by Large Language Models (LLMs) has become highly coherent and human-like, which greatly increases the difficulty of distinguishing AI-generated texts from human-written ones. Consequently, the question of whether a given passage was machine-generated has become urgent and practical for institutions, platforms, and publishers. Text watermarking offers a technical paradigm that does not depend on traditional methods of training a classifier after the fact: a detectable statistical signal is inserted in LLM-generated text so that its origin can later be tested, and the signal can be inserted during different stages [1, 2]. Detection is then conducted under a statistical hypothesis test against the null hypothesis that the text was produced without any knowledge of the watermarking schemes [1, 3]. This approach has already moved beyond the laboratory, with a real-life experiment assessing nearly 20 million Gemini responses [2].

Despite the deployment, the field has not converged yet. First, the cryptographic analysis shows that no undetectable watermarking scheme can be made unremovable in general [4], and another argument holds for

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

an attacker who needs neither the private key nor knowledge of the scheme, applying in all kinds of watermarks [5]. Therefore, watermarks cannot be strengthened infinitely, and its guarantees are bounded by construction. Second, the research records are unstable in a way because benchmarks report that previously published conclusions and their robustness reverse when they are re-evaluated on one unified platform [6].

This survey addresses the gap between a growing type of schemes and the difficulty of saying which of them is better. Rather than ranking different methods, it organises the field by where the watermark information enters the generation pipeline and asks what each insertion point commits a designer to, which is also what determines a scheme's detector requirements, its attack surface, and the deployment scenarios open to it [1, 7-9].

There are five research questions stated here as the survey's own scope. RQ1: what are the major technical approaches, and where do they intervene in the generation lifecycle? RQ2: what trade-offs exist among detectability, false-positive rate, output quality, robustness, payload capacity, and computational cost? RQ3: what attacks threaten these schemes, what capabilities do they require, and which design properties predict success or failure? RQ4: why are reported results often difficult or impossible to compare across papers? RQ5: what unresolved problems and deployment implications remain? The survey looks on 29 works read in depth, 19 as core sources and 10 as auxiliary ones, from a pool of works published between 2022 and 2026. It does not claim to cover every paper in a this field since it develops rapidly. Its contribution is a lifecycle taxonomy, an attack landscape organised by operation and budget, and a sustained treatment of comparability as a first-order finding.

2. Overview and taxonomy

2.1. Problem setting and watermarking lifecycle

A watermarking scheme must include three traits: what signal is inserted, where in the pipeline, and how to detect it. The insertion point is the most crucial of the three, because it determines who can watermark and who can detect. Insertion can be in the model's own weights, so that the model produces watermarked text directly, without any modified decoder [8]. It can occur inside the decoding step, which happens in the most of the papers. Also, it can occur after the text is generated, by rewriting it to meet the demands [7].

2.2. Taxonomy of LLM watermarking methods

Figure 1 organises different types of watermarking methods, with the decoding-time family subdivided by the mechanism that produces the signal. Within decoding-time methods, the signal may be produced by a bias to the logits, by the sampling rule or even a combination of many schemes, so designs treat text as consequential rather than cosmetic [1, 2, 10].

The detector part of the taxonomy shows a real divergence. Detector requirements differ substantially across designs: some detectors need the key with the model or its logits, one needs only the key, and one can be conducted by anyone because it contains no secret information [3, 9, 11]. Table 1 records them per method alongside the mechanism and the stated distortion position.

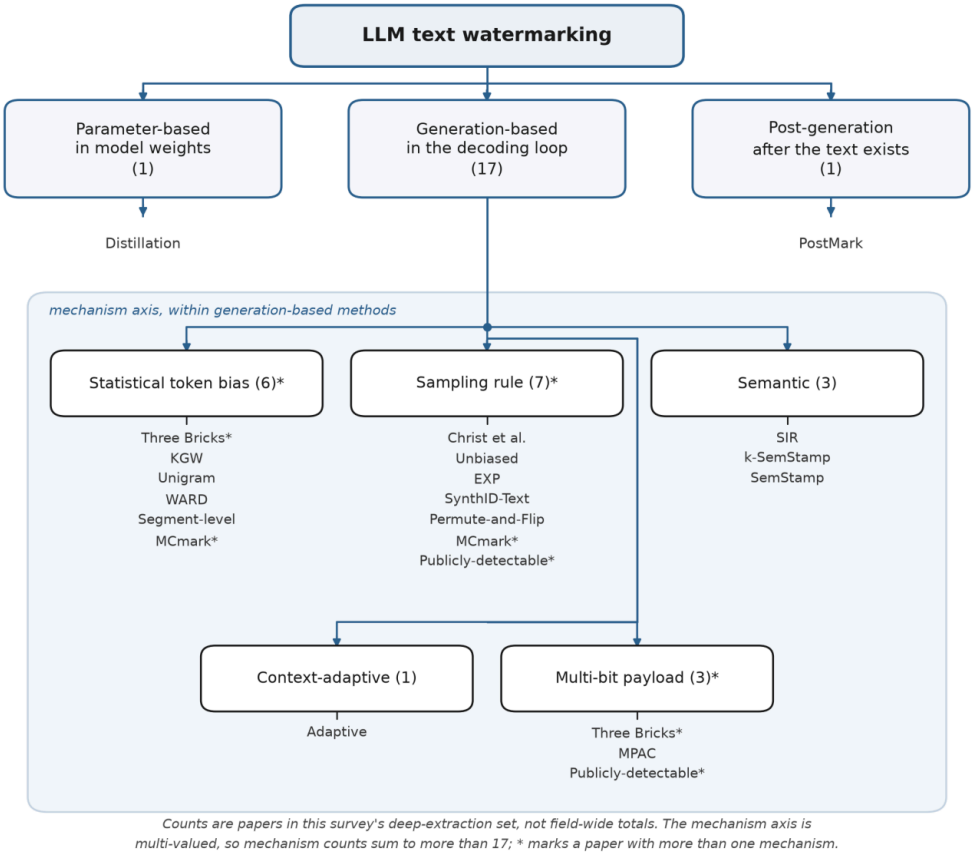

Figure 1. Taxonomy of LLM text watermarking by insertion point in the generation lifecycle. Counts are papers in this survey's deep-extraction set, not field-wide totals; the mechanism axis is multi-valued

Table 1. Representative LLM watermarking methods, ordered by where the watermark inserts

Method	Year	Insertion point	Mechanism	Signal / principle	Detector requires	Payload	Distortion stance
Watermark distillation	2023	Parameter	token bias	watermark distilled into model weights	key + model/logits	Not reported	Not reported
Christ et al.	2023	Generation	sampling rule	cryptographic undetectability, entropy-driven	key only	Not reported	undetectable (crypt.)
Three Bricks	2023	Generation	token bias, multi-bit	non-asymptotic tests; one key per message	key + model/logits	multi-bit, one key per message	Not reported
Unbiased Watermark	2023	Generation	sampling rule	unbiased reweighting at sampling	key + model/logits	Not reported	distortion-free (unbiased)
KGW (red–green list)	2023	Generation	token bias	PRNG green list, logit bias	key + model/logits	Not reported	distortionary (logit bias)
EXP / EXP-edit	2023	Generation	sampling rule	fixed random key sequence + alignment	key + model/logits	Not reported	distortion-free

Table 1. Continued

SIR	2023 Generation	semantic	trained semantic-invariant logit network	key + model/logits	Not reported	Not reported
Unigram-Watermark	2023 Generation	token bias	single fixed green list ($K = 1$)	key + model/logits	Not reported	distortionary (green list)
SynthID-Text	2024 Generation	sampling rule	Tournament sampling	key + model/logits	Not reported	configurable non-distortionary
k-SemStamp	2024 Generation	semantic	k-means partition of embeddings	key + model/logits	Not reported	Not reported
SemStamp	2024 Generation	semantic	LSH partition of sentence embeddings	key + model/logits	Not reported	Not reported
WARD (RAG)	2024 Generation	token bias	watermarked RAG corpus documents	key + model/logits	Not reported	Not reported
Adaptive Watermark	2024 Generation	adaptive context	entropy-gated multiplicative logit scaling	key + model/logits	Not reported	Not reported
MPAC (multi-bit)	2024 Generation	multi-bit	token-to-bit-position allocation	key + model/logits	multi-bit (≥ 32 bits shown)	Not reported
Segment-level detection	2024 Generation	token bias	segment-level detection in mixed text	key only	Not reported	Not reported
Permute-and-Flip	2024 Generation	sampling rule	Permute-and-Flip decoder	key + model/logits	Not reported	better quality–stability trade-off
MCmark	2025 Generation	sampling rule, token bias	multi-channel segment promotion	key + model/logits	Not reported	distortion-free (unbiased)
Publicly-detectable	2025 Generation	sampling rule, multi-bit	embedded cryptographic signature	public key (no model)	signature payload	distortion-free
PostMark	2024 Post-gen.	lexical edit	post-hoc semantic word insertion	key + embedder (logit-free)*	zero-bit; ratio r knob	Not reported

Note. Detector requirement determines which deployment scenarios are possible. "Not reported" means the paper does not state the propert, and this survey does not infer them. *PostMark's detector needs the secret table and an embedder but explicitly no model or logits.

These categories in Figure 1 are this survey's analytical framework. This review sees the lifecycle axis as its primary one and treats the mechanism axis as multi-valued, because several surveyed methods combine sampling rules with token-biases or payload construction thus cannot be assigned to a certain type of mechanism [9, 12, 13]. Three works in Table 1 accordingly appear under two mechanisms.

2.3. Scope and positioning of this survey

The corpus is restricted to schemes whose watermark signal is embedded in and recoverable from LLM output. Retrieval-augmented generation is included but only treated as an application dimension rather than part of the method family. Survey papers are used for discovery and terminology merely, never as the source of any quantitative claim, thus every number below is traced to the work that measured and reported it.

3. Watermarking techniques

Table 1 summarises the properties discussed in this section: insertion point, signal mechanism, detector requirement, payload, and stated distortion position. It does not include performance statistics, for reasons discussed in the evaluation section.

3.1. Parameter-based watermarking

One work makes the watermark live in the weights rather than the decoder. By distillation, the watermark moves from the decoder into the model's own weights, and the result shows that the model produces watermarked text without any decoder modification [8]. The appeal is that watermarking becomes possible for open models whose users control decoding, but the same learnability is what enables adversarial spoofing [8, 14].

There are two distillation types, and their objectives differ in their requirements: sampling-based distillation does not need shared tokenizer or vocabulary but is less efficient, and logit-based type requires matching tokenizers [8]. However, distillation has its drawbacks. It is reported to have a worse quality-detectability trade-off than decoding-time watermarks, with higher perplexity and similar or larger p-values [8]. What is even worse for deployment is that a distilled watermark can be removed by further fine-tuning on non-watermarked text, leaving the robustness of such watermark signal dependent on the downstream training pipeline [8].

3.2. Generation-based watermarking

3.2.1. Statistical token bias

The oldest and most classic scheme intervenes on the logits. The token-bias methods partition the vocabulary with a pseudo-random generator seeded by preceding context and then increase the probability of one partition during decoding. Detection is then conducted by counting how many tokens fall in the favoured set [1, 11]. The logits in the favoured list are biased instead of merely banning the others, which alleviates the distortion in high-entropy text [1].

Based on this method, two continuous researches illustrate how changes in the structure can affect this watermark scheme. Reducing this construction to a fixed partition of the vocabulary makes detection only need the key, without the language model nor the generation prompt, and the detector runs in linear time [11]. In addition, the context length used to seed the partition is itself a parameter with two-sided consequences: a low length causes repetition and meaningless output, while a high length reduces robustness [12]. The context dependence introduced here is the property that attacks exploit most effectively, which will be discussed in later sections, because a method with context-free scheme retains detection under edits which can collapse its context-dependent counterpart [6, 12].

3.2.2. Sampling rules

A second family does not change the distribution. Instead it makes the sampling step key-dependent, which is what allows a scheme to maintain the distribution between watermarked and unwatermarked output [3, 15].

They are defined with unbiasness to make sure the expectation of the reweighted distribution after adding watermark signal equals to the original distribution [3]. Their structures are mainly the same, only differ in how they obtain the equivalence. One replaces context hashing with a fixed random key and formulates detection as robust sequence alignment, which yields distortion-freeness and a permutation test giving non-asymptotic p-values [15]. Another technique is tournament sampling, which embeds the watermark in the sampling algorithm rather than by biasing logits, and is configurable as either non-distortionary or distortionary [2].

Cryptographic analysis to this method makes the strongest claims and states its price most explicitly. It defines undetectability as computational indistinguishability between the watermarked and unwatermarked model when without the key. Futhermore, it identifies the entropy of a specific output as the quantity which affect detectability [4]. Across this family, the guarantees are purchased with assumptions: detection cost grows with key length, and the entropy of the specific output bounds what any undetectable scheme can achieve [4, 15].

3.2.3. Semantic watermarking

If word-level editing is the main threat, then anchoring the signal to semantic meaning is a natural response. Semantic watermarking methods change the basic unit of watermarking from token to sentence, partitioning a sentence-embedding space and sample full candidate sentences until one falls in a pre-defined region, so that the signal survives paraphrasing and rewriting that leaves semantic meaning unchanged [16-18]. A variant includes training a small network so that watermark-logit similarity can represent text-embedding similarity, making the bias as a function of semantic meaning rather than of certain tokens [18].

This mechanism's cost is structural and reported directly. One method generates 13.8 candidate sentences on average before accepting a single valid one [17]. Replacing the locality-sensitive-hashing partition with k-means clustering is reported to reduce the cost, but still producing 13.3 sentences per accepted sentence against 20.9 for the original one [16]. Semantic designs also inherit an assumption that token-level methods avoid, because the null distribution used for detection is only approximately correct on out-of-domain text and the robustness argument rests on an analytic Lipschitz bound [18].

3.2.4. Adaptive and context-aware methods

Context-adaptive methods scale the logits multiplicatively and gate the signal injection step based on current text entropy, so that the watermark is applied at where the model has freedom of choice and withheld it when the conditions are not capable [19]. The gating on text entropy triggers the low-entropy failure directly, as the token-bias structure identifies certain context where forcing the watermark signal raises text perplexity [19]. Evidence for this family is relatively thin at present because it is only presented by a single paper, and its reported advantages have not yet been proved or reproduced by an independent or comparative study [19].

3.2.5. Multi-bit and payload-carrying watermarks

The methods discussed above mainly relies on presence detection, and answers whether the text is machine-generated, while a payload can additionally answer which user or model produced it. Payload-carrying schemes allocate each token pseudorandomly to one position of a message, so that capacity is obtained by partitioning the same token budget, and single-bit detection would otherwise spend entirely on presence [12, 20]. This framing comes at a cost, and the cost is measured in the paper: true-positive rate at a false-positive rate of 1e-3 falls monotonically with payload, from 0.98 at zero bits to 0.98, 0.95, 0.93 and 0.91 at 8, 16, 24 and 32 bits on LLaMA-2-7B with C4 [20]. An alternative structure assigns one secret key every message as a cyclic shift of a secret vector, letting the number of embeddable messages exceed the vocabulary size [12].

3.3. Post-generation watermarking

The only remaining insertion point of watermark signal is after generation. Post-generation watermarking compares the text embedding with a secret word-embedding table to select an input-dependent special word list, then uses an instruction-following model to rewrite the text to include those words [7]. Its detector is reported to be logit-free and model-free, only needing the secret table and an embedder. This survey retains the frozen taxonomy label for this method's detector requirement, which classifies it with schemes needing model access. The evidence records a narrower requirement, and Table 1 marks the exception rather than silently reclassifying the method [7].

Because the signal is carried by chosen inserted words rather than by token statistics, the scheme has a different robustness profile: on a low-entropy setting it retains 69.2 true-positive rate after paraphrasing, whereas logit-based baselines fall to 19, 31.8 and 0.6 [7]. The trade is detectability of the intervention itself, and here the reported result is favourable, with human annotators asked to highlight out-of-place words achieving an F1 of 0.06 [7]. This method carries no payload, and its capacity limit is the insertion ratio of the selected words [7].

3.4. Boundary cases and emerging directions

Two boundary cases extend the watermark settings. One is applying existing watermark schemes to a retrieval corpus so that data owners can run black-box dataset inference, another is shifting the detection to locating watermarked segments inside mixed text [21, 22]. In the RAG setting the reported result is 100% accuracy on the harder benchmark split, reached with at most 80 documents, which the authors frame as a dataset-scale guarantee [21]. This review treats retrieval-augmented watermarking as an application setting rather than a method, because the single paper reuses an existing watermark method,only changing the inference target, not the insertion mechanism [21]. Segment-level detection addresses a different gap, reporting a true-positive rate of 0.912 at a segment-level false-positive rate of 1e-5 when only 10% of the document is watermarked, while 0.602 for a whole-document detector on the same setting [22]. A cryptographic construction pursues another direction for commercial LLMs, embedding a publicly verifiable signature by rejection sampling and deliberately set weak robustness to the watermark because strong and semantics-preserving robustness has been shown impossible [9].

4. Attacks and robustness

Table 2 organises the attack categories by different operations, recording the assumed adversary capability, the quantified budget if a paper provides one, the representative works, and the reported outcome with its conditions attached. The table's rows cannot be compared between one and another, because no two rows share any same evaluation conditions.

Table 2. Attack categories against LLM watermarks

Attack	Adversary capability	Quantified budget	Representative works	Demonstrated outcome (with conditions)	Coverage
Paraphrase	black-box output access; surrogate paraphraser; no key	7,000 samples x 512 tokens, < 5 GPU-hours, < $10	[5, 23, 24]	adaptive paraphrase evades ≥ 96.6% (provider Llama2-13b, attacker Llama2-7b); non-adaptive machine paraphrase costs only 0.05-0.15 ROC-AUC at T~200 tokens	9 papers (strongest)

Table 2. Continued

Translation / back-translation	controls prompt language; a translator only	pivot language + translator choice (no edit budget)	[25]	CWRA drops KGW TPR 0.992 -> 0.213 at FPR = 0.1, AUC to 0.61 (KGW) / 0.54 (UW) on BAICHUAN-7B, without hurting ROUGE	1 dedicated paper
Local edit / substitution	token-level edits; no key, no scheme identity	parameterised by edit fraction / insertion count	[6, 23, 24]	context-dependent designs are the fragile ones: TGRL falls to 0.429 (Swap) / 0.222 (Typo) while context-free UG holds 0.929/0.930 on the same protocol (Qwen2.5-14B, FPR = 1%)	4 papers
Regeneration	second model rewrites content from scratch	Not reported	only as a setting inside [6, 26]	no dedicated study in this corpus; appears only as one attack column among others, so no isolated effect size can be reported	NO dedicated paper (gap)
Truncation / mixing	mixes watermarked and human text in one document	watermarked fraction (e.g. 10%, 25%)	[22, 24]	copy-paste is worse than paraphrase: at 25% watermarked text AUC falls below 0.7 at 200 tokens; segment-level detection recovers TPR 0.912 vs 0.602 whole-document at 10% (C4/Llama-7B, KGW)	2 papers
Spoofing / forgery	black-box generations; scheme known, key unknown	30,000 queries, ~$42	[26, 27]	>80% of forged texts pass as watermarked at FPR=1e-3 in the most restrictive (B0, D0) setting; piggyback spoofing >90% ASR; multi-key defence cuts spoofing 91% -> 13%	3 papers
Stealing / removal	black-box queries; Kerckhoffs (scheme known, key unknown)	30,000 queries, ~$42; 2.42-2.95 detector queries/token	[5, 26, 27]	learning the key rules lifts DIPPER scrubbing from ~0% to 0.90 FNR@1e-3; detection-API access gives removal ASR 1.00; random-walk attack drops KGW z 6.236 -> 1.628	4 papers
Detector evasion	query access to the detector, or none (offline adaptive)	2.42-2.95 queries per token; or fully offline	[6, 23, 26]	median z pushed to 1.43 against a threshold of 4 with perplexity nearly unchanged (6.17 vs 6.28); DP noise sigma = 4 defends spoofing at 0% ASR at a cost of ~1 accuracy point	3 papers
Collusion / multi-output	trains a model on watermarked outputs (distillation)	rho = 5% watermarked instructions; N = 225k scored tokens	[14]	watermarks are radioactive: $p < 1e$-5 open-model unsupervised at rho = 5%; a second fine-tune on human data halves but does not remove the signal	1 paper

Note. Rows follow the frozen attack taxonomy, including categories with no dedicated literature in this corpus. Every reported number keeps the conditions under which it was measured; numbers cannot be compared between rows.

4.1. Threat model

Threat models across different attack types share a Kerckhoffs-style structure: the adversary is assumed to know the scheme and its parameters, only without the secret key, interacting with the watermarked model through its outputs [23, 27]. Notably, the strongest attacks discussed in the table are also the most constrained ones: the stealing experiments run totally without detector access and no access to non-watermarked base responses, and the adaptive-paraphrase attacks cannot collect watermarked text or access the provider's verification interface [23, 27].

The impossibility argument goes further in the same direction, assuming the attacker is computationally weaker than the model it attacks and has only black-box access, which the authors argue strengthens the result [5]. That argument rests on a quality oracle and a perturbation oracle whose output has probability bounded away from zero of not decreasing quality while inducing a well-mixing random walk [5].

4.2. Paraphrasing

Paraphrasing is the most frequently evaluated attack in this field, and its reported effect depends mainly on whether the paraphraser is generic or trained against the watermark [23, 24]. Using a generic paraphraser, detection loses only 0.05 to 0.15 ROC-AUC at around 200 tokens, and under 0.1 when the pre-attack length is 600 [24]. While the paraphraser is fine-tuned against the schemes, the same operation can evade detection at 96.6% or above with the provider on Llama2-13b and the attacker on Llama2-7b, and non-adaptive transfers still reach 94.3% [23]. Paraphrasing attack is cheap, using 7,000 preference samples of 512 tokens, under 5 GPU hours, and less than $10 at quoted cloud rates [23].

Taken together, these results show that paraphrase strength is what mainly determines the effects. Although studies reporting mild paraphrase failure and studies reporting near-total evasion are quite different, they are not in conflict because the attackers' training budget varies [23, 24].

4.3. Translation and back-translation

The translation attack can exploit the capability which the defender cannot control, which is the attacker's free choice of translation language [25]. Under the cross-lingual attack, KGW true-positive rate at a fixed false-positive rate of 0.1 falls from 0.992 while no attack as 0.213, and reported AUCs drop to 0.61 for KGW and 0.54 for the unbiased scheme on BAICHUAN-7B [25]. Unusually among the attacks surveyed, it does not cost the attacker output quality, with summarization ROUGE-1 for one scheme rising from 14.24 without attack to 18.98 under the attack [25]. However, translation attack has its defects, because cross-lingual consistency remains poor across schemes, with correlation coefficients generally below 0.3 and relative errors predominantly above 80% [25].

4.4. Local editing and substitution

Local editing mainly affects the token-bias family if it is context-dependent, and the effect is visible both in a benchmark's controlled contrast and in the experiment of context-window trade-off reported by designers themselves [6, 11, 12]. One benchmark's results separate schemes under local editing by whether the detection statistic depends on preceding context [6]. Under this benchmark's settings on Qwen2.5-14B-Instruct with the false-positive rate at 1%, a context-dependent scheme falls from 0.993 true-positive rate to 0.429 under word swapping and 0.222 under typo injection, while a context-free scheme under the same conditions holds 0.929 and 0.930 [6]. That loss is partly recoverable at the detector part. An entropy-based detector lifts typo-attack true-positive rate from 0.222 to 0.472 and swap resilience from 0.429 to 0.650 [6].

4.5. Regeneration

Regeneration, which can be seen as a kind of fierce paraphrasing, appears in this corpus only as one evaluated condition inside broader benchmarks and researches, and no dedicated paper examines or characterises it systematically, so in our survey no isolated effect size can be reported for it here [6, 26].

4.6. Truncation, mixing, and partial text

Mixing watermarked text with human-written text is reported to hurt detection more than paraphrasing. With 25% watermarked text remaining, AUC falls below 0.7 at 200 tokens and below 0.85 at 600 [24]. This failure cannot be attribute to the watermark signal loss, because changing detection to the task of segment localisation actually partly recovers accuracy on mixed texts [22, 24]. Localisation can be feasible but still sensitive under other forms of attack, reaching an average intersection-over-union above 0.55 across settings yet decreasing to 0.032 under paraphrase [22].

4.7. Watermark stealing and removal

Stealing attacks the watermark key and structure rather than the simply changing the text. Watermark stealing reverse-engineers the key-dependent rules from model output, and is reported to cost 30,000 queries at approximately $42, with about 10,000 queries already sufficient for convergence [27]. It is also capable of boosting other attacks: paraphrasing leaves a scrubbing false-negative rate of 0.02 at a threshold of 1e-3, while the same paraphraser, given the stolen watermark rules, reaches 0.90 [27].

Two other routes also reach comparable outcomes under different methods. A separate study reports that additional access to a detection API yields a removal attack success rate of 1.00 at 2.42 detection queries per token, with spoofing at 0.98 and 2.95 queries per token [26]. Another work argues removal is achievable even without any key or scheme information, reporting a random-walk attack that drives KGW z-scores from 6.236 to 1.628 and unigram scores from 8.210 to 1.456 on C4 with Llama2-7B [5]. The authors report a control experiment in which a judge model preferred the watermarked output over text from the perturbation model, which they present as evidence that the quality rather than trivial degradation drives the result [5].

4.8. Spoofing, forgery, and detector evasion

Spoofing attack focuses on another perspective: instead of alleviating or removing the watermark, spoofing forges it, and over 80% of forged texts are reported to pass the detection at a false-positive rate of 1e-3 in the most restrictive condition [27]. A related attack preserves the watermark while changing the text, inserting toxic tokens inside over 20% of a sentence's length while remaining unnoticed by a moderation endpoint, with attack success above 90% [26]. Under real detection circumstances, this study report to push a median z-score to 1.43 against a threshold of 4 while only slightly changing perplexity at 6.17 versus 6.28 [26].

The defence methods reported against spoofing attacks trade against each other rather than resolving the problem: the method of multiple keys is capable of cutting spoofing success from 91% to 13% but watermark removal success rate rises above 97% if more than seven keys are used, while differential-privacy noise defends spoofing at 0%, but result in decline in detection accuracy [26].

4.9. Collusion and multi-output attacks

Watermarks' ability to propagate through distillation is proved by researches. A model which trained on watermarked output still remains detectable: reported significance is below 1e-5 in the open-model unsupervised setting when only 5% of fine-tuning instructions are watermarked [14]. This property is a two-

sided one, because the same mechanism that lets an owner prove unauthorised training also lets an adversary transfer watermark onto harmful content, claiming it is generated by the victim model. It is reported at a 71% harmful-response rate for the distilled adversary while the watermarked victim actually produces 0% [8, 14]. Fine-tuning on human data is reported to weaken but not remove the signal, and detection degrades when the scoring distribution differs from the fine-tuning language [14].

4.10. Cross-attack synthesis

Across the attack literature, the design properties of watermarks predict fragility: dependence on preceding context, reuse of a single key, and reliance on a distribution shift each appear as the responsible mechanism behind a reported failure [6, 11, 12]. One benchmark reports that the relative robustness ordering is consistent across models and datasets, so it attributes watermark robustness to algorithmic design [6]. The same benchmark reports a change of robustness under different test splits, with distribution-shift methods scoring high on a distributional quality metric yet dropping to near-zero true-positive rates when facing mixing and paraphrasing attacks [6].

This review therefore reads the attack methods as bounding specific designs: every reported break in this corpus is demonstrated against named schemes under a stated budget, and the universal claim is an impossibility of a scheme which can evade every attack that holds under explicit assumptions [5, 6, 23].

5. Evaluation and comparability

5.1. Detection metrics and thresholds

Detection is reported through several statistics: a z-test on token counts, summed per-token score tests, permutation tests giving non-asymptotic p-values, and log-scale significance against a token budget [1, 3, 14, 15]. Accordingly, one benchmark notes that prior work uses a z-test on the sum score while it derives the exact null distribution where possible and otherwise uses an empirical one, so thresholds and p-values are not interchangeable across papers [28]. Threshold choices also differ in kind, from a fixed value of 4 to a stated type-I rate, true-positive rates quoted at a false-positive rate of 1%, detection at a 2% false-positive rate expressed as the number of tokens needed [1, 6, 28].

5.2. Text quality and distortion

Quality is measured by several factors, including perplexity, distributional similarity, judge-model ratings, and human annotation. One benchmark rejects perplexity, claiming that it favours repetitive text [7, 9, 28]. The real-world production-scale quality measurement is more likely a preference and feedback study about users, reporting a 0.01% difference in thumbs-up rate and 0.02% in thumbs-down rate, both described as statistically insignificant [2].

Another observation concerns what is absent, which is the distortion of the text. This review found distribution-preservation or distortion statements for only nine of the nineteen method papers examined, so the corresponding cells of Table 1 record that the property is not reported [2, 3, 9, 13, 15].

5.3. Robustness evaluation

Robustness in LLM watermark is the detection performance that survives a specified attack. A scheme is characterised by how exactly its detection statistic degrades when text is perturbed. Any robustness measurement therefore relies on two parameters, the adversarial operation and the strength of the inserted

signal, both sharing the same importance as shown in Section 4. However, the second parameter is unevenly reported. Query and sample budgets are quantified for stealing and adaptive paraphrase, while the translation attack is parameterised only by pivot language and translator choice and carries no edit budget [23, 25, 27]. One benchmark states the limit of its own measure, reporting tamper resistance as an upper bound on robustness against real-world sophisticated attackers [28]. Robustness results are thus conditional on an attack protocol and this corpus does not parameterise them in a common way.

5.4. Payload and capacity

Payload and capacity concern what a watermark carries. For example whether a detector can recover the information of users or models, or how much information a given token and scheme can hold. Designs differ in whether they address that question at all. One scheme carries a multi-bit message and reports bit accuracy against message length, another defines capacity as the number of embeddable messages relative to vocabulary size, and a third carries no payload and reports an insertion ratio [7, 12, 20]. These are three different quantities: payload recovery accuracy, the size of the message space, and the extent of an intervention. Capacity is therefore not a standardised quantity here, and it can only be measured within individual papers.

5.5. Computational and query costs

Quantitative deployment-cost reporting is sparse in this field. One work reports generation latency of 15.527 against 15.615 milliseconds per token, and another reports generation overhead as candidate sentences per accepted sentence, while detection cost elsewhere is reported as growing with key length [2, 15, 17]. This review does not aggregate them into a field-level overhead figure, because no such figure is available from the present researches [2, 15, 17].

5.6. Why reported numbers are often not comparable

The clearest evidence that cross-paper statistics cannot be compared comes from the benchmarks themselves, summarised in Table 3. The three benchmarks in this corpus normalise on three mutually incompatible axes: one equalises detection strength across schemes, one holds the false-positive rate at 1% but leaves scheme parameters at their defaults, and one fixes quality degradation, then compares the number of tokens needed for detection [6, 28, 29].

Table 3. Evaluation protocols of selected works, and why their numbers are not directly comparable

Work	Role	Model (s)	Dataset(s)	Gen. length	Strength calib.	Detection	Comparability
WaterBench	benchmark	Llama2-7B-chat, Internlm-7B-8k	nine tasks, five length categories; 200/task (AlpacaFarm 805)	five length categories; input avg 32.58 tokens	grid-search to EQUAL strength (TPR 0.95, 0.7)	one-proportion z-test; GPT4-Judge quality	Normalises detection strength. Raw hyper-parameters not comparable across schemes; the scheme worst at strength 0.7 ranks FIRST at 0.95, so rank order follows the calibration point.

Table 3. Continued

WATERPARK	benchmark	OPT-1.3B, LLaMA 3-7B-chat, Qwen2.5-14B-Instruct	C4, HC3, Story Completion, Law Stack Exchange, Paper Conclusion	Not reported	NONE; per-scheme defaults	all TPRs at FPR = 1%	Fixes the operating point (FPR = 1%) but not scheme strength, an authors' stated limitation. Its Table 1 records prior conclusions reversing on one platform.
MARKMYWORDS	benchmark	Llama family	book reports 100, stories 96, fake news 100 (296 gen.)	≤ 1024 tok.; first third scored	fixes QUALITY (≤ 1% degrad.) and tamper resist. (> 0.2), then compares SIZE	size = tokens to detect at 2% FPR; exact null else empirical	Faults WaterBench's sub-10-token calibration as reachable 'only by significantly harming quality'. Its z-test is less precise than the exact null, so thresholds are not interchangeable.
KGW	method	OPT-1.3B	C4 RealNewsLike	200 tokens	gamma = 0.5, delta = 2	z-test, $z = 4$ (type-I $3e$-5)	Origin protocol: widely imitated, rarely reproduced exactly.
No Free Lunch	study	LLAMA-2-7B, OPT-1.3B	OpenGen, 500 prompts	200 tok. (KGW/Unigram); 70 (Exp)	defaults, 3 schemes	max score over keys; per-key FPR < 1e-3	Length differs by scheme within one paper.Sequence length is not constant even within one study.
SynthID-Text	method	Gemma 2B/7B-IT, Mistral 7B-IT	ELI5: 10,000 non-distort., 1,500 distort.; ~20M live Gemini	Not reported	$m = 30$ tournament layers; Bayesian	TPR at FPR = 1%; Gemma 7B-IT, temp. 0.7	Only production-scale deployment evidence, latency +0.57%. Its scale is not reproducible here, so its quality result cannot pool with lab studies.
Publicly-detectable	method	Mistral 7B; GPT-4 Turbo judge	news-like C4 prompts	gadget in ~1,000–2,000 tok.	all 4 algorithms emit SAME token count	GPT-4 Turbo 0–100; perplexity rejected	Its l = 16/32 denote characters, not tokens, unlike papers using l for tokens: a unit collision making naive cross-paper reading wrong. Targets weak robustness only.

Table 3. Continued

Radioactivity	attack/ study	Llama-2-chat-7B → Llama-1-7B	100k instruction pairs (~14M tokens)	context 2048	delta = 3.0, gamma = 0.25, k = 2	log10(p) at N = 225k open / 30k closed tokens	Reports log10(p) against a token budget, not TPR/FPR: no shared axis with detection-rate papers.

Note. The three benchmarks normalise on three mutually incompatible axes: WaterBench equalises detection strength, WATERPARK fixes the false-positive rate but leaves scheme strength at defaults, and MARKMYWORDS fixes quality and compares watermark size.

In addition, each benchmark states the problem in its own terms. The first states that hyper-parameters between different schemes are not comparable because most prior studies evaluate generation and detection separately [29]. It also shows the ranking itself is somehow inconsistent, reporting that the worst-ranked watermark on generation score at a strength of 0.7 can become the first one at a strength of 0.95 [29]. The second benchmark uses default parameter configurations with no per-scheme strength calibration, and lists prior published conclusions that reverse under its unified platform [6]. The third explicitly criticises the first benchmark's calibration target as achievable only by significantly harming quality, and criticises another platform for testing each watermark at a single user-provided configuration [28].

Scheme divergence also is an important factor beyond the benchmark level. Sequence length, inside a single study, differs because one scheme is evaluated at 70 tokens which is regulated by its original author while others run at 200 [26]. Statistics also differ across papers, as one work reports that its l values of 16 and 32 denote characters rather than tokens, because the watermark is planted on plain text where tokenizers cannot guarantee stable re-encoding [9].

This review's comparability analysis found that no comparability group can be derived from the evidence pool, and it therefore reports no cross-paper performance ranking [6, 26, 28, 29]. Where a controlled comparison does exist within a single paper, this review reports such contrasts only in that form, as with the context-dependent against context-free result measured on one benchmark protocol [6]. Taken together, the benchmark evidence suggests that a defensible comparison must state its normalisation axis explicitly, because existing operations of equalising strength, fixing the false-positive rate, and fixing quality each produce different results among the same schemes [6, 28, 29].

6. Challenges and future directions

Figure 2 collects the tensions that the surveyed authors state about their designs, distinguishing trade-offs from impossible ones. Taken together with the impossibility results, this review reads those tensions as something structural: two of them are proved rather than observed, so they cannot be engineered away by a better structure or scheme [4, 5, 9].

Taken together, several gaps still exist in this field: regeneration has no dedicated study, attack strength is parameterised in only part of the literature, and quantitative overhead is only reported by few works [2, 6, 17, 25]. Detector-assumption management is an open problem with a proved boundary: publishing the detector is achievable, but only alongside a deliberately weakened robustness goal, because strong robustness is impossible [5, 9]. Low-entropy text remains a fundamental limitation, because humans and LLMs always produce nearly identical low-entropy context and applying the watermark there inevitably raises perplexity [1, 4].

Deployment readiness is also questioned by the surveyed authors themselves. One work states that watermarking is not deployment-ready, and another notes that the effect on quality and user experience must be carefully controlled in real-life scenarios [2, 20]. This review's recommendation is that future benchmarks report a normalisation axis, an attack budget, and a cost measurement as first-class fields, since the absence of these three is what makes the present literature resistant to synthesis and comparison [2, 23, 28, 29].

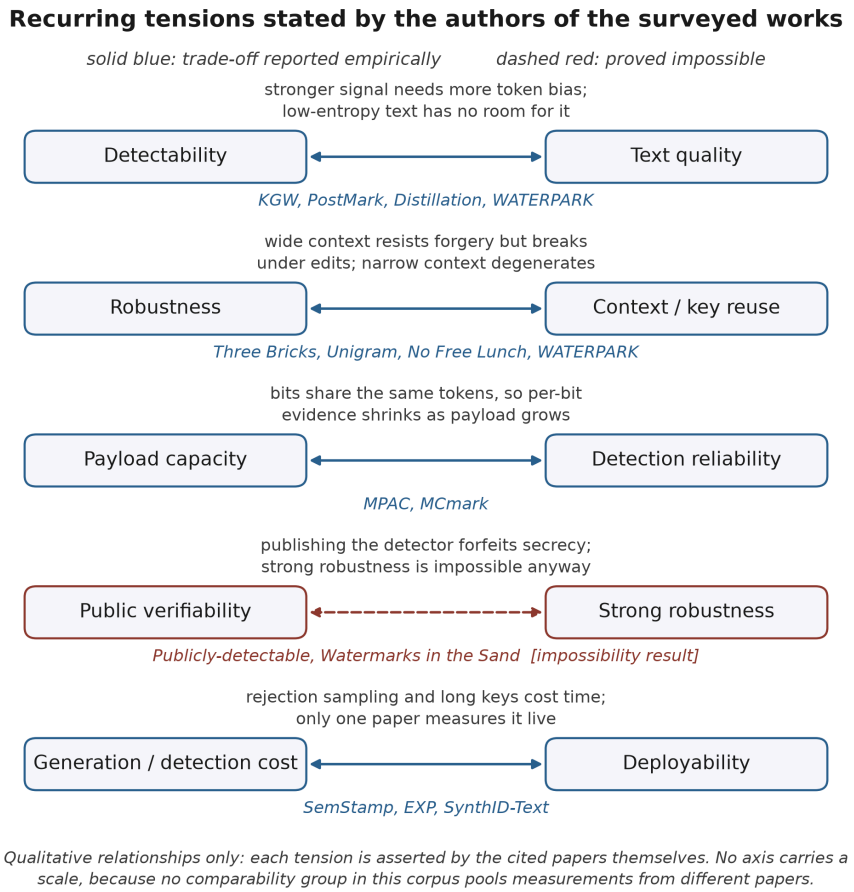

Figure 2. Challenges of LLM watermarking, as stated by the authors of the surveyed works. Solid blue marks a trade-off reported empirically; dashed red marks one proved impossible

7. Conclusion

Watermarking for LLMs has moved past the question of whether a signal can be inserted in generated text. The methods surveyed here show that it can, as the signal is placed in the model's weights, in the decoding step, or in a rewrite of output, with one scheme already been assessed in real world. The open questions are now about conditions: which insertion point suits deployment, what information a detector has, and which attacks a design can handle.

Where the watermark enters the pipeline is the most important factor, because it decides what a detector holds, whether the signal survives rewriting, and who can verify it. The authors also state about trade-offs in their own designs: signal strength against text quality, context width against robustness, payload against per-bit evidence, public verifiability against strong robustness, and detection cost against deployability [4, 5, 9]. The attack literature sharpens this picture: robustness depends on the attack applied and on how much the attacker

spends, while fragility is predicted by the design properties, most often dependence on preceding context or reuse of a single key [6, 23].

The reported numbers are not offered here as rankings. Studies differ in models, datasets, text lengths, detection thresholds, attack strengths and payload definitions, and the existing three benchmarks examined normalise on different axes, so the same schemes can change rank with different conditions chosen [28, 29]. Comparing design choices, threat models, evaluation steps and demonstrated failure occasions is therefore more informative than comparing statistics.

There are some remaining gaps belonging to this field. Regeneration lacks a dedicated study, attack strength is parameterised in only part of the literature, and few works measure computational cost. Low-entropy text is a fundamental limitation, and several authors still question deployment readiness. Future benchmarks would help most by reporting a normalisation axis, an attack budget and a cost measurement as first-class fields [2, 29].

References

[1] Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., & Goldstein, T. (2023). A watermark for large language models. *Proceedings of the 40th International Conference on Machine Learning, 202*, 17061–17084. https://proceedings.mlr.press/v202/kirchenbauer23a.html

[2] Dathathri, S., See, A., Ghaisas, S., Huang, P., McAdam, R., Welbl, J., Bachani, V., Kaskasoli, A., Stanforth, R., Matejovicova, T., Hayes, J., Vyas, N., Merey, M. A., Brown-Cohen, J., Bunel, R., Balle, B., Cemgil, A. T., Ahmed, Z. S., Stacpoole, K., . . . Kohli, P. (2024). Scalable watermarking for identifying large language model outputs. *Nature, 634*, 818–823. https://doi.org/10.1038/s41586-024-08025-4

[3] Hu, Z., Chen, L., Wu, X., Wu, Y., Zhang, H., & Huang, H. (2024). Unbiased watermark for large language models. *Proceedings of the International Conference on Learning Representations* (pp. 45408–45436). https://proceedings.iclr.cc/paper_files/paper/2024/file/c5b00c5bdcc6fe35907dbcca03d27652-Paper-Conference.pdf

[4] Christ, M., Gunn, S., & Zamir, O. (2024). Undetectable watermarks for language models. *Proceedings of the 37th Annual Conference on Learning Theory*, 1125–1139. https://proceedings.mlr.press/v247/christ24a.html

[5] Zhang, H., Edelman, B. L., Francati, D., Venturi, D., Ateniese, G., & Barak, B. (2024). Watermarks in the sand: Impossibility of strong watermarking for language models. *Proceedings of the 41st International Conference on Machine Learning, 235*, 58851–58880.

[6] Liang, J., Wang, Z., Hong, S., Ji, S., & Wang, T. (2025). Watermark under fire: A robustness evaluation of LLM watermarking. *Findings of the Association for Computational Linguistics: EMNLP 2025*, 21050–21074. https://doi.org/10.18653/v1/2025.findings-emnlp.1148

[7] Chang, Y., Krishna, K., Houmansadr, A., Wieting, J. F., & Iyyer, M. (2024). PostMark: A robust blackbox watermark for large language models. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 8969–8987. https://doi.org/10.18653/v1/2024.emnlp-main.506

[8] Gu, C., Li, X. L., Liang, P., & Hashimoto, T. (2024). On the learnability of watermarks for language models. *Proceedings of the Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=9k0krNzvlV

[9] Fairoze, J., Garg, S., Jha, S., Mahloujifar, S., Mahmoody, M., & Wang, M. (2023). *Publicly-detectable watermarking for language models* (Cryptology ePrint Archive, Paper 2023/1661). https://eprint.iacr.org/2023/1661

[10] Zhao, X., Li, L., & Wang, Y.-X. (2025). Permute-and-flip: An optimally stable and watermarkable decoder for LLMs. *Proceedings of the International Conference on Learning Representations* (pp. 32468–32496).

https://proceedings.iclr.cc/paper_files/paper/2025/file/50be7e77b9c883144940be925b608acc-Paper-Conference.pdf

[11] Zhao, X., Ananth, P. V., Li, L., & Wang, Y.-X. (2024). Provable robust watermarking for AI-generated text. *Proceedings of the Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=SsmT8aO45L

[12] Fernandez, P., Chaffin, A., Tit, K., Chappelier, V., & Furon, T. (2023). Three bricks to consolidate watermarks for large language models. *2023 IEEE International Workshop on Information Forensics and Security (WIFS)* (pp. 1–6). https://doi.org/10.1109/WIFS58808.2023.10374576

[13] Chen, R., Wu, Y., Guo, J., & Huang, H. (2025). Improved unbiased watermark for large language models. *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 20587–20601). https://doi.org/10.18653/v1/2025.acl-long.1005

[14] Sander, T., Fernandez, P., Durmus, A., Douze, M., & Furon, T. (2024). Watermarking makes language models radioactive. *Advances in Neural Information Processing Systems, 37*, 21079–21113. https://doi.org/10.52202/079017-0664

[15] Kuditipudi, R., Thickstun, J., Hashimoto, T., & Liang, P. (2023). *Robust distortion-free watermarks for language models* (arXiv:2307.15593). arXiv. https://arxiv.org/abs/2307.15593

[16] Hou, A., Zhang, J., Wang, Y., Khashabi, D., & He, T. (2024). k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. *Findings of the Association for Computational Linguistics: ACL 2024*, 1706–1715. https://doi.org/10.18653/v1/2024.findings-acl.98

[17] Hou, A., Zhang, J., He, T., Wang, Y., Chuang, Y.-S., Wang, H., Shen, L., van Durme, B., Khashabi, D., & Tsvetkov, Y. (2024). SemStamp: A semantic watermark with paraphrastic robustness for text generation. *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 4067–4082. https://doi.org/10.18653/v1/2024.naacl-long.226

[18] Liu, A., Pan, L., Hu, X., Meng, S., & Wen, L. (2024). A semantic invariant robust watermark for large language models. *Proceedings of the Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=6p8lpe4MNf

[19] Liu, Y., & Bu, Y. (2024). Adaptive text watermark for large language models. *Proceedings of the 41st International Conference on Machine Learning, 235*, 30718–30737. https://proceedings.mlr.press/v235/liu24e.html

[20] Yoo, K., Ahn, W., & Kwak, N. (2024). Advancing beyond identification: Multi-bit watermark for large language models. *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 4031–4055. https://doi.org/10.18653/v1/2024.naacl-long.224

[21] Jovanović, N., Staab, R., Baader, M., & Vechev, M. (2025). Ward: Provable RAG dataset inference via LLM watermarks. *Proceedings of the Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=kVrwHLAb20

[22] Zhao, X., Liao, C., Wang, Y.-X., & Li, L. (2025). Efficiently identifying watermarked segments in mixed-source texts. *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6304–6316. https://doi.org/10.18653/v1/2025.acl-long.316

[23] Diaa, A., Aremu, T., & Lukas, N. (2025). Optimizing adaptive attacks against watermarks for language models. *Proceedings of the 42nd International Conference on Machine Learning*. https://openreview.net/forum?id=AsODat0dkE

[24] Kirchenbauer, J., Geiping, J., Wen, Y., Shu, M., Saifullah, K., Kong, K., Fernando, K., Saha, A., Goldblum, M., & Goldstein, T. (2024). On the reliability of watermarks for large language models. *Proceedings of the 12th International Conference on Learning Representations* (pp. 49660–49704).

https://proceedings.iclr.cc/paper_files/paper/2024/file/d78e9e4316e1714fbb0f20be66f8044c-Paper-Conference.pdf

[25] He, Z., Zhou, B., Hao, H., Liu, A., Wang, X., Tu, Z., Zhang, Z., & Wang, R. (2024). Can watermarks survive translation? On the cross-lingual consistency of text watermark for large language models. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 4115–4129. https://doi.org/10.18653/v1/2024.acl-long.226

[26] Pang, Q., Hu, S., Zheng, W., & Smith, V. (2024). No free lunch in LLM watermarking: Trade-offs in watermarking design choices. *Advances in Neural Information Processing Systems, 37*, 138756–138788. https://doi.org/10.52202/079017-4402

[27] Jovanović, N., Staab, R., & Vechev, M. (2024). Watermark stealing in large language models. *Proceedings of the 41st International Conference on Machine Learning, 235*, 22570–22593. https://proceedings.mlr.press/v235/jovanovic24a.html

[28] Piet, J., Sitawarin, C., Fang, V., Mu, N., & Wagner, D. (2025). MARKMyWORDS: Analyzing and evaluating language model watermarks. *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 68–91. https://doi.org/10.1109/SaTML64287.2025.00012

[29] Tu, S., Sun, Y., Bai, Y., Yu, J., Hou, L., & Li, J. (2024). WaterBench: Towards holistic evaluation of watermarks for large language models. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1517–1542. https://doi.org/10.18653/v1/2024.acl-long.83