LOCALIZE TO DETECT: LOW-BUDGET SYNTACTIC SPAN WATERMARKING FOR LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM watermarking provides a practical mechanism for identifying machine-generated text. Conventional methods perturb token probabilities at every decoding step, allowing these interventions to accumulate and degrade generation quality. Sparse watermarking addresses this issue by restricting watermarking to selected tokens. However, existing sparse statistical methods remain token-level, with no shared structural unit for embedding and detection. We introduce **SpanWM**, a span-based watermarking framework that uses parser-recoverable phrases as the unit of both embedding and detection. A secret key selects syntactic phrases for watermarking, and the detector re-identifies the corresponding spans from the received text for localized detection. This span-level design concentrates watermark evidence within a low token budget while leaving most of the output untouched. Experiments show that existing sparse methods must test substantially more tokens to match SpanWM's detection performance. SpanWM also better preserves the draft's meaning and remains robust to token-level edits. At matched AUROC on C4, SpanWM requires only 16–45% of the watermark budget needed by the sparse baselines. Our code is available at `https://anonymous.4open.science/r/spanwm_11-BCD6`.

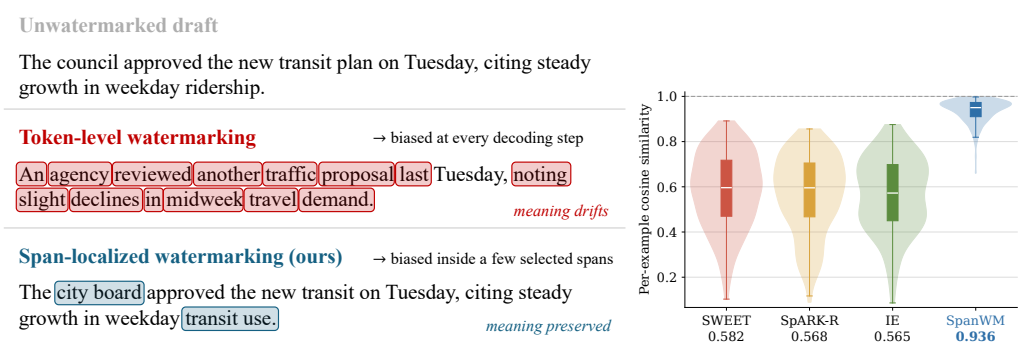
1 INTRODUCTION

Large language models (LLMs) now write essays, news articles, and reviews (OpenAI, 2026; Anthropic, 2026; Google DeepMind, 2025) that readers cannot tell apart from human writing (Clark et al., 2021; Jakesch et al., 2023; Jones & Bergen, 2026). This makes disinformation cheap to mass-produce (Zellers et al., 2019; Goldstein et al., 2023; Spitale et al., 2023) and the origin of a text difficult to verify. One approach is to build detectors that judge whether a given text was machine-generated (Gehrmann et al., 2019; Mitchell et al., 2023), but such judgments can be defeated by paraphrasing (Krishna et al., 2023; Sadasivan et al., 2025). Watermarking instead places a statistical signal in the text as it is produced, allowing its origin to be checked later (Liu et al., 2024). This approach has already moved from research into practice (Dathathri et al., 2024; Sander et al., 2026).

The central challenge is to make a watermark detectable while changing as little of the original text as possible. Linguistic watermarks edit a few key-selected positions through synonym substitution or syntactic transformation (Atallah et al., 2001; Topkara et al., 2006). Statistical watermarks commonly bias token probabilities throughout generation: a secret key partitions the vocabulary into "green" and "red" lists, and favoring green tokens creates a signal that accumulates across the output (Kirchenbauer et al., 2023). The broad intervention supplies ample evidence, but its effects on generation can accumulate as well. Restricting the intervention to a small token budget promises better preservation of the original text, yet leaves the detector with less evidence and fewer places where a signal can survive editing. The placement of that small amount of evidence therefore matters as much as the number of marked tokens. A useful *low-budget watermark* must resolve both pressures: it should preserve meaning while remaining detectable after the text is modified.

Selective statistical methods address the budget by marking only high-entropy steps (Lee et al., 2024; Liu & Bu, 2024; Gu et al., 2025), positions chosen by a trained selector (Wang et al., 2025), or triggers based on the secret key or part-of-speech (POS) tags (Hoang et al., 2026). Their placement rules, however, are tied either to model-dependent quantities or to individual token positions. Model-dependent sites can be difficult to recover without repeating the selection procedure at detection,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Motivation for span-localized watermarking.

(b) Semantic similarity to the original draft.

Figure 1: SpanWM biases only selected spans, preserving the draft's meaning in (a) and achieving higher semantic similarity than the baselines in (b).

while a site tied to one word can disappear when that word is replaced. A sparse intervention made during decoding can also alter subsequent generation, even if few tokens receive the bias directly. We ask whether a larger linguistic unit can serve as a shared anchor for placement and detection. Syntactic constituents provide such a unit: a parser can identify candidate phrases from the received text, and each selected phrase can carry evidence across several tokens. A phrase can sometimes remain identifiable after individual words change, although its boundaries may shift. This combination motivates **SpanWM**, which places a statistical watermark in a few key-selected syntactic spans rather than distributing it through the generation.

SpanWM first generates an unwatermarked draft, then selects a few of its parsed constituents using a secret key. It locally regenerates each selected phrase under a green-list bias and samples several replacements using the prompt and preceding text as context. A right-context likelihood score chooses a replacement that fits the continuation, while a reconstruction check guides retries. This candidate selection helps the new phrase connect to text that was already written. Most of the draft is therefore left untouched, limiting opportunities for semantic drift. At detection, the parser and key independently identify sites in the received text and pool the tokens within them for a statistical test. The detector needs neither the original draft nor the prompt or insertion metadata, and does not run the generating model. By using phrase structure for both operations, SpanWM concentrates evidence within a small budget while giving the detector a way to locate that evidence again.

We evaluate SpanWM across C4, CNN/DailyMail, and WMT16. In a C4 budget sweep, competing sparse methods require substantially more tokens to match SpanWM's detection performance. SpanWM also better preserves the unwatermarked draft's meaning under both embedding-based and LLM-based evaluations. It remains robust to editing such as and token insertion, deletion, and swap attacks, with its detection advantage persisting at stringent false-positive rates under editing attacks.

Our contributions are as follows:

- *A syntax-guided low-budget watermark*: we select a few syntactic constituents using a secret key and locally regenerate them to carry a statistical signal. At detection, the parser and key identify candidate spans in the received text, making syntax the shared unit of embedding and detection.
- *Detectability with semantic fidelity*: we compare methods across detection budgets and show that sparse baselines need substantially more tested tokens to match SpanWM's detectability. Embedding-based and LLM-based evaluations also show that SpanWM better preserves the unwatermarked draft's meaning.
- *Robustness under editing*: we evaluate token-level attacks, showing that the localized signal remains detectable when the received text is modified. We also analyze how the selected constituent roles affect detection and site recovery.

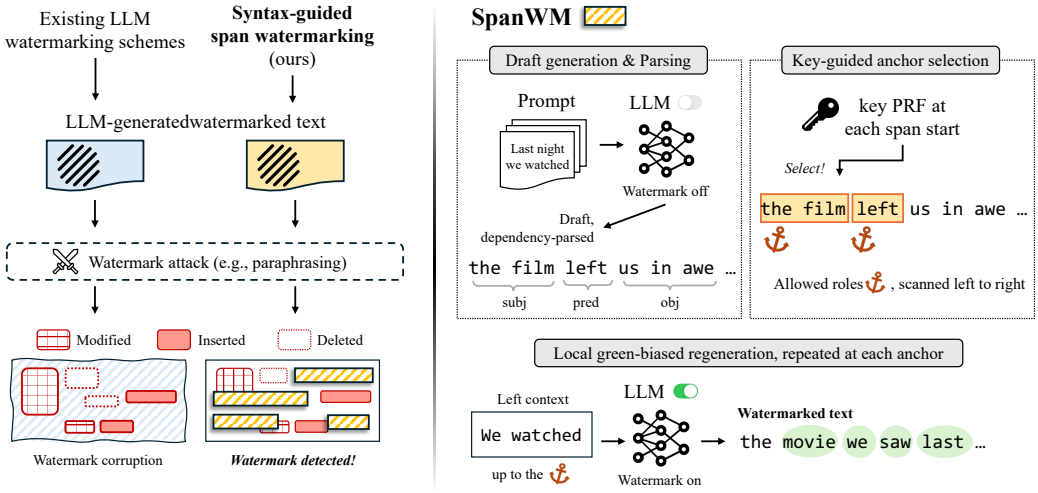


Figure 2: Overview of SpanWM. A secret key selects syntactic constituents in an unwatermarked continuation. For each site, the model generates several watermarked replacements of the original constituent's token length and selects one using the likelihood of the following context. A reconstruction check guides regeneration retries. The detector parses the received continuation and pools green-list evidence over the selected constituent ranges.

2 METHODOLOGY

SpanWM embeds statistical watermark evidence in key-selected syntactic constituents of an unwatermarked draft. Each replacement is generated under a green-list bias with a token budget equal to the selected constituent's length. Multiple replacements are sampled, and the one that best supports the original right context is selected before checking whether its site can be reconstructed. Detection parses the received continuation, selects constituents using the same keyed rule, and pools evidence over their token ranges.

2.1 PRELIMINARIES

Let $\mathcal{M}_\theta$ be a language model with vocabulary $\mathcal{V}$, q a prompt, and x its generated continuation, represented as a character string. The prompt is preserved, and all site selection and editing operate within x. Write $\tau(x) = (t_1, \ldots, t_n)$ for the tokenization of the continuation without added special tokens. Each token has a character interval; after trimming boundary whitespace from that interval, denote it by $[a_i, b_i)$. Empty intervals are excluded.

A syntactic extractor Φ returns constituents $s \in \Phi(x)$ with type $r(s)$, start offset $c(s)$, and end offset $e(s)$. We map a constituent to model-token positions by interval overlap:

$$\mathcal{T}(x, s) = \{\, i : a_i < b_i,\ b_i > c(s),\ a_i < e(s) \,\}. \tag{1}$$

This mapping is used both to measure candidate length and to collect detection evidence. The ordered type inventory is $\mathcal{R} = (\mathrm{NP}, \mathrm{VP}, \mathrm{PP})$ by default; Section 4.3 studies larger inventories.

We use separate keys κ_{sel} for site selection and κ_{wm} for green-list assignment. Let H_{64} return the non-negative big-endian integer represented by the first 64 bits of a SHA-256 digest. Following Kirchenbauer et al. (2023), the preceding token and κ_{wm} determine a pseudorandom green list $\mathcal{G}_i \subset \mathcal{V}$ of size $\lfloor \gamma |\mathcal{V}| \rfloor$. For a candidate token v, the generation logit is modified as

$$\ell'_i(v) = \ell_i(v) + \delta\, \mathbb{I}[v \in \mathcal{G}_i], \tag{2}$$

where $\delta > 0$. SpanWM applies this bias only while generating replacements for selected constituents. Detection uses γ as the reference green rate.

2.2 SYNTAX-GUIDED SITE SELECTION

Syntactic candidates. SpanWM uses spaCy with benepar to extract constituency-tree nodes labeled NP, VP, or PP. Empty and duplicate spans are removed, and candidates are sorted by start offset, type label in lexicographic order, and end offset. Nested constituents remain candidates. Parsing uses a whitespace-normalized copy of x, with offsets mapped back to the original string (Appendix B.1). Both boundaries matter: embedding replaces $[c(s), e(s))$, while detection scores the token range of the reconstructed constituent.

Keyed type selection. For a start position c, define $u(x, c) = x[\max(0, c - n_g) : c)$, the preceding at most n_g characters. The key selects one target type,

$$r^\star(x, c) = \mathcal{R}_{H_{64}(\kappa_{\mathrm{sel}} \| \sigma \| \,\mathrm{UTF8}(u(x,c)))\bmod |\mathcal{R}|}, \tag{3}$$

where the inventory is indexed from zero and σ is the fixed tag `role-per-span`. A constituent is eligible when

$$r(s) = r^\star(x, c(s)), \qquad \lambda_{\min} \le |\mathcal{T}(x, s)| \le \lambda_{\max}. \tag{4}$$

Defaults are $n_g = 16$, $\lambda_{\min} = 2$, and $\lambda_{\max} = 12$. Hashing preceding characters avoids using the constituent's rewritten content or subword identifiers at its boundary. The target type is unchanged whenever the preceding character context is preserved; successful recovery also depends on the parse and eligibility of the received text.

Deterministic selection. Let $\textsc{NextSite}(x, f)$ return the first eligible constituent in canonical order whose start is at least f, or $\perp$ if none exists. Detection starts at $f = 0$ and repeatedly selects $s = \textsc{NextSite}(x, f)$, then sets $f = e(s)$, stopping after at most $S_{\max} = 4$ sites. Embedding uses the same selection function on the evolving continuation, with reconstruction checks and a boundary update described below. No fixed-length evidence window or entropy gate is used.

2.3 LOCALIZED WATERMARK EMBEDDING

SpanWM first samples an unwatermarked continuation $x \sim \mathcal{M}_\theta(\cdot \mid q)$. For a selected constituent s, write $x_L = x[0 : c(s))$, $x_R = x[e(s) : |x|)$, and $L_s = |\mathcal{T}(x, s)|$. The original right context is retained while the constituent is regenerated.

Constituent-length candidate generation. Each regeneration attempt draws B watermarked candidates. Let $\bar{x}_L$ be x_L with one trailing space removed if present, as used by the boundary-handling procedure. For $b = 1, \ldots, B$, generate a token sequence

$$\mathbf{y}^{(b)} \sim \mathcal{M}_{\theta,\gamma,\delta,\kappa_{\mathrm{wm}}}(\cdot \mid q \| \bar{x}_L), \qquad |\mathbf{y}^{(b)}| = L_s. \tag{5}$$

The decoded candidate is spliced between the left and right contexts, producing $x^{(b)}$. Thus the generation budget follows the selected constituent's length, and the original prompt remains available as generation context. The right context is used for candidate evaluation rather than as input to this sampling step. Splicing handles whitespace and punctuation at the boundaries; it does not guarantee that generated token identities survive full-text retokenization. Appendix B.1 gives the boundary rules.

Right-context-aware candidate selection. We score how naturally each replacement leads into the preserved right context. Tokenize the spliced continuation as $\tau(x^{(b)}) = (v_1^{(b)}, \ldots, v_{n_b}^{(b)})$, and let j_b be the first token whose character interval extends beyond the end of the inserted body. With $m_b = \min(R, n_b - j_b + 1)$, define

$$Q_b = \frac{1}{m_b} \sum_{i=j_b}^{j_b+m_b-1} \log P_\theta(v_i^{(b)} \mid v_{<i}^{(b)}), \qquad b^\star = \arg\max_{1 \le b \le B} Q_b. \tag{6}$$

We set $Q_b = 0$ if there is no scoreable right-context token or $j_b = 1$, and break ties by candidate order. Defaults are $B = 4$ and $R = 16$. Scoring uses the base model without the green-list logit bias and conditions on the spliced continuation only; the prompt q is included in candidate generation but not in this scoring pass. This best-of-B choice favors replacements that fit the retained suffix.

Reconstruction check. For the selected candidate $x' = x^{(b^\star)}$, rerun $\textsc{NextSite}(x', f)$ from the current scan boundary. Define

$$d(x'; s, f) = \begin{cases} |c(\textsc{NextSite}(x', f)) - c(s)|, & \text{if } \textsc{NextSite}(x', f) \neq \perp, \\ \infty, & \text{otherwise.} \end{cases} \tag{7}$$

If $d \leq \varepsilon$, accept the replacement; otherwise repeat candidate generation and selection up to $A_{\max}$ attempts. If none passes, retain the attempt with the smallest deviation, breaking ties by attempt order. We use $\varepsilon = 4$ characters and $A_{\max} = 4$. This check tests start proximity, not equality of constituent boundaries or token identities.

After updating x, embedding advances using $\textsc{Advance}(x, c(s))$, which returns the end of the constituent found by $\textsc{NextSite}(x, \max(0, c(s) - \varepsilon))$, or $c(s) + 1$ if none is found. Because the parse changes after replacement, this update does not guarantee nonoverlapping edits or exact agreement with the final detector scan. A final alignment check is recorded for diagnosis and does not discard outputs. Algorithm 1 summarizes the procedure.

2.4 Watermark Detection

The detector receives the continuation x and keys $\kappa_{\text{sel}}, \kappa_{\text{wm}}$. It uses the same parser, tokenizer, and green-list configuration, but requires no original prompt, draft, embedding metadata, or autoregressive language model forward pass. The scan in Section 2.2 yields constituents $\mathcal{S}(x)$ with $|\mathcal{S}(x)| \leq S_{\max}$. Their token positions are pooled as

$$\mathcal{P}(x) = \bigcup_{s \in \mathcal{S}(x)} \mathcal{T}(x, s), \qquad \mathcal{P}_+(x) = \{i \in \mathcal{P}(x) : i > 1\}. \tag{8}$$

The union removes duplicate positions. With one-token green-list context, the first continuation token is excluded because its seed cannot be reconstructed from the continuation alone. Each remaining green list is computed from the true preceding token in $\tau(x)$. Define

$$N = |\mathcal{P}_+(x)|, \qquad g = \sum_{i \in \mathcal{P}_+(x)} \mathbb{I}[t_i \in \mathcal{G}_i]. \tag{9}$$

Here N is the number of tested tokens, not the number of edited tokens. Reconstructed constituents can differ from the replaced ranges. With the default length filter, $N \leq S_{\max}\lambda_{\max} = 48$; the realized value varies across texts.

SpanWM computes the upper tail of a binomial reference model:

$$p = \Pr_{Z \sim \text{Binomial}(N, \gamma)}[Z \geq g] = \sum_{k=g}^{N} \binom{N}{k} \gamma^k (1-\gamma)^{N-k}. \tag{10}$$

For $N > 0$, the decision rule is $p < \alpha$, with $\alpha = 0.01$ by default. The tail is evaluated exactly under this reference model. Separate keys do not establish independence of all tested events: repeated preceding-token/current-token pairs at different positions are counted separately. Thus the reference tail alone does not guarantee a calibrated false-positive rate for the full pipeline; we evaluate operating points using unwatermarked examples. Ranking uses $-\log_{10} p$, and we report the descriptive statistic

$$z = \frac{g - \gamma N}{\sqrt{N\gamma(1-\gamma)}}. \tag{11}$$

If $N = 0$, the detector returns not watermarked and the evaluation assigns the minimum ranking score. Algorithm 2 gives the detection procedure.

3 Experiments

3.1 Watermark Detectability

Table 1 shows that SpanWM consistently provides the strongest detection performance across all three datasets, with its advantage becoming particularly clear in the low-FPR regime. At an FPR

Table 1: Detection performance and generation quality across datasets.

Dataset	Method	AUROC	TPR @ FPR =				z-score	Cos. sim.	LLM score
			10%	5%	1%	0.1%			
C4	KGW	0.831	0.54	0.42	0.19	0.09	1.70	0.713	3.48
	SWEET	0.956	0.88	0.87	0.72	0.56	3.85	0.556	3.32
	SpARK-R	0.944	0.84	0.74	0.58	0.39	3.49	0.567	3.43
	IE	0.880	0.72	0.67	0.46	0.42	2.16	0.693	3.20
	SpanWM *(Ours)*	**0.983**	**0.96**	**0.91**	**0.82**	**0.73**	**4.05**	**0.951**	**7.25**
CNN/DM	KGW	0.813	0.45	0.36	0.16	0.10	1.63	0.790	3.72
	SWEET	0.922	0.80	0.77	0.60	0.34	3.10	0.635	3.86
	SpARK-R	0.943	0.86	0.75	0.52	0.43	3.23	0.630	3.77
	IE	0.843	0.66	0.49	0.39	0.24	1.79	0.795	3.78
	SpanWM *(Ours)*	**0.984**	**0.98**	**0.96**	**0.88**	**0.83**	**3.97**	**0.954**	**7.09**
WMT16	KGW	0.796	0.50	0.39	0.18	0.05	1.59	0.658	2.80
	SWEET	0.972	0.91	0.86	0.82	0.82	4.28	0.510	2.94
	SpARK-R	0.969	0.91	0.83	0.72	0.39	3.49	0.517	3.02
	IE	0.898	0.76	0.68	0.43	0.30	2.39	0.669	2.86
	SpanWM *(Ours)*	**0.986**	**0.95**	**0.95**	**0.92**	**0.90**	**4.47**	**0.935**	**6.91**

of 0.1%, SpanWM outperforms the strongest baseline by 17, 40, and 8 percentage points on C4, CNN/DM, and WMT16, respectively. This indicates that the gain is not merely better average separability: SpanWM preserves strong, high-confidence watermark evidence even when false positives are tightly constrained.

Importantly, this performance is achieved with only 17.0 detection tokens on average, comparable to SpARK-R (17.1) and fewer than IE (22.4) and SWEET (25.1). Thus, SpanWM does not obtain stronger detection by aggregating evidence over more tokens; rather, it extracts a stronger signal from a small detection budget. Moreover, while the strongest baseline varies across datasets, SpanWM remains remarkably stable, with AUROC ranging only from 0.983 to 0.986. We analyze this budget–detectability trade-off more directly in Sec. 4.2, and report results with additional backbone models in the appendix.

3.2 Semantic Fidelity Preservation

SpanWM's localized watermarking yields a clear advantage in semantic fidelity. As shown in Table 1, SpanWM consistently achieves the highest similarity to the corresponding unwatermarked reference under both embedding-based cosine similarity and LLM-based semantic evaluation. The agreement between these two complementary metrics is particularly meaningful: the improvement is not specific to a particular representation or evaluator, but consistently reflects better preservation of the original content. We provide the full prompt used for the LLM-based semantic evaluation in Appendix F.

This advantage stems from a fundamental difference in how the watermark is introduced. Token-level watermarking methods intervene during decoding, so an early perturbation can alter the subsequent generation trajectory even when only a subset of tokens is directly targeted. SpanWM instead starts from an unwatermarked draft and regenerates only a small number of selected spans, leaving the rest of the text unchanged. Its right-context rejection step further selects replacements that remain compatible with the continuation after the edited span. This localized intervention changes the usual relationship between detectability and fidelity observed among the baselines: stronger detectors such as SWEET exhibit substantially lower semantic similarity, whereas methods with higher similarity, such as KGW and IE, are considerably weaker in detection. SpanWM breaks this apparent trade-off, achieving the strongest detectability while preserving the highest semantic fidelity.

3.3 Attack Robustness

We evaluate robustness to token insertion, deletion, and swap attacks with edit ratios up to 50%, applied to the watermarked text. As shown in Figure 3, SpanWM remains remarkably stable as the attack strength increases: even at a 50% edit ratio, its AUROC stays above 0.965 for all three attacks,

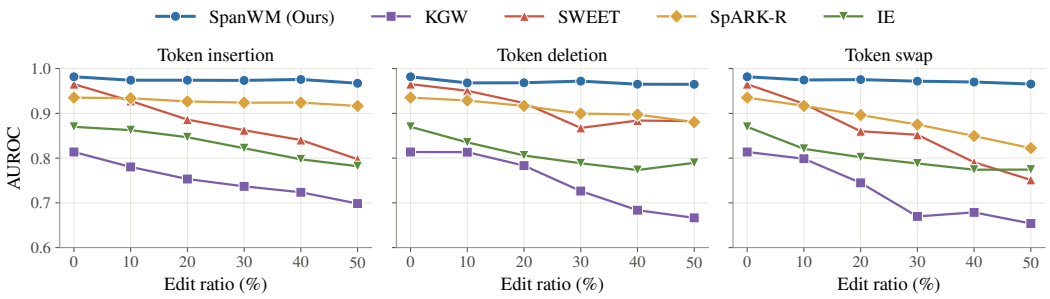


Figure 3: Robustness to token-level edits on C4 with Llama-3.2-3B. Token insertion, deletion, and swap are applied to the watermarked text at increasing edit ratios. SpanWM maintains nearly constant detectability even under heavy editing, whereas baseline performance progressively degrades.

Table 2: Qualitative comparison of watermarked outputs for the same prompt. Green and red shading denote green-list and red-list tokens, respectively.

Original	It was the sound of the clock that woke him, a loud booming that came from the clock tower in the old square.
KGW	The clock maker had been toiling in his shed for days, taking apart a grandfather clock and re as sembling it in an entirely new way.
SWEET	It would seem that time is running out for Mr.
SpARK-R	It was the lighthouse clock that captured his attention.
IE	It would seem that time is running out for Mr Clock maker.
SpanWM	It was the sound of the clock that woke him, every night when that came right around the clock clock in that old.

with a maximum drop of only 0.017 from the clean setting. In contrast, the baselines progressively lose detectability, widening the performance gap under stronger edits. This robustness highlights a key benefit of localized watermarking: because the watermark evidence is concentrated in a small number of selected spans, substantial edits outside these regions leave the detection signal largely intact. The numerical results are provided in Appendix D.

4 Analysis

4.1 SpanWM Limits Semantic Drift

Table 2 illustrates how different watermarking strategies affect the generated content. The baseline methods often induce substantial semantic drift, changing the underlying event, entities, or narrative direction of the original text. In contrast, SpanWM largely preserves the original proposition and confines its changes to localized regions. This example mirrors the quantitative results in Sec. 3.2: by modifying only selected spans rather than perturbing the generation trajectory throughout the sequence, SpanWM better preserves the content of the unwatermarked draft.

4.2 Budget Required to Match SpanWM

The results above show that SpanWM achieves strong detectability with a remarkably small detection budget. This motivates a natural question:

"How much additional budget do existing sparse watermarks require to match SpanWM's detectability?"

Answering this, we fix SpanWM at its operating point on C4 with Llama-3.2-3B ($N = 17.0$, AUROC $= 0.983$) and sweep the native sparsity control of each baseline. Figure 4 reveals a clear

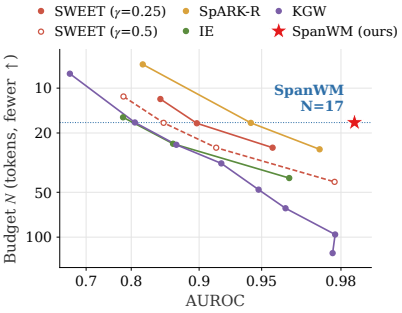


Figure 4: Detection budget N vs. AUROC.

Table 3: Role-set size ablation on Llama-3.2-3B/C4. Detection budget and performance remain stable as the role inventory expands.

	$\lvert\mathcal{R}\rvert$	3	4 (+S)	5	6	7	4 (+SBAR)
N		16.7	17.3	17.1	17.4	16.6	16.7
Site align (%)		83	83	78.7	82.4	82.6	83
Mean z		4.26	4.38	4.25	4.19	4.10	4.24
AUROC		0.982	0.983	0.979	0.981	0.971	0.981
TPR@1% FPR		0.855	0.810	0.805	0.820	0.825	0.755

separation: at a comparable budget, even the strongest baseline, SpARK-R, reaches only 0.944 AUROC. Matching SpanWM requires 38.1 detection tokens for SpARK-R (2.2×), 44.7–57.9 for SWEET (2.6–3.4×), and 105.1 for IE (6.2×).

The key distinction is therefore not sparsity itself, but how much detectable watermark evidence can be concentrated into a small budget. The KGW-span control makes this especially clear: simply placing a watermark in a contiguous span of comparable size remains far from SpanWM, and the span must grow to 160 tokens before reaching the same AUROC. SpanWM thus occupies a substantially more favorable budget–detectability regime, achieving strong detection with far fewer tokens than existing sparse watermarking approaches.

4.3 Role-Space Size and Composition

Settings. We ablate the constituent-role inventory $\mathcal{R}$ used for key-dependent site selection. Starting from the default $\mathcal{R}_3 = \{\mathrm{NP}, \mathrm{VP}, \mathrm{PP}\}$, we expand the inventory sequentially by adding S, then SBAR, NML, and ADJP, yielding $\mathcal{R}_{4\mathrm{S}}$, $\mathcal{R}_5$, $\mathcal{R}_6$, and $\mathcal{R}_7$, respectively. We additionally evaluate $\mathcal{R}_{4\mathrm{SBAR}} = \{\mathrm{NP}, \mathrm{VP}, \mathrm{PP}, \mathrm{SBAR}\}$ to compare two four-role inventories with different compositions. All other watermarking settings are kept fixed.

Table 3 shows that enlarging the role inventory has little effect on the overall detection budget or AUROC. From $|\mathcal{R}| = 3$ to 6, AUROC remains within 0.979–0.983 while the number of tested tokens stays near 17; even with all seven roles, AUROC remains 0.971. A larger role space, however, is not necessarily better. The default three-role set already attains 0.982 AUROC and the highest TPR at 1% FPR, while the two four-role variants show similar AUROC but noticeably different low-FPR performance. This suggests that detector quality depends more on *which* roles are included than on the cardinality of $\mathcal{R}$ alone.

Table 4 explains why the ablation is largely flat. NP, VP, and PP together account for 77.8% of selected sites and are among the most reliably reconstructed after regeneration. Additional roles are used much less frequently, and several are substantially less stable—for example, alignment falls to 64% for NML and 35% for ADJP. As a result, expanding the nominal role inventory changes relatively few of the sites that dominate the detection statistic. The effective role space is therefore governed by both role prevalence and reconstructability, rather than by $|\mathcal{R}|$ alone.

5 Related Work

Post-hoc watermarking. These methods embed a watermark by editing text that has already been generated. In the earliest scheme, a keyed hash ranks the sentences of a given document, meaning-preserving syntactic transformations rewrite only the selected sentences to carry the mark, and the remainder of the document is left unmodified; without the key, the marked sentences cannot be distinguished from the rest (Atallah et al., 2001). Subsequent methods preserve this paradigm, embedding the watermark by selecting and editing positions in the text; the editing operation is lexical substitution, with substitutes drawn from synonym dictionaries (Topkara et al., 2006; Xiang et al., 2018) or proposed by neural language models (Yang et al., 2022; Qiang et al., 2023; Munyer et al., 2024; Hao et al., 2025). At LLM scale, the same paradigm reappears as word-level substitution

Table 4: **How additional roles affect the detector.** Later-added labels are less reliably aligned after regeneration, but their rarity limits their contribution to the selected sites. Per-label statistics are measured at $|\mathcal{R}| = 7$ on C4 with $n = 200$.

label	% of constituents	passes length filter (%)	% of selected sites	aligned (%)
NP	42.6	71	39.7	85
VP	22.4	64	18.0	82
PP	12.5	89	20.1	77
S	14.4	48	11.5	68
SBAR	4.1	64	5.3	74
NML	2.2	92	3.2	64
ADJP	2.1	74	2.2	35

codes, secret word insertion, and keyed re-paraphrasing (Yang et al., 2023; Chang et al., 2024; Lau et al., 2024). This line of research supplies the procedure that SpanWM adopts, rewriting a generated draft at a small number of selected syntactic constituents. Anchoring at these constituents also makes the edit more robust than in prior post-hoc methods, whose sites are tied to individual words and can be lost when those words are edited in generation-time, whereas phrase structure can survive word-level changes and remains recoverable from the received text with a parser alone.

Sparse watermarking. This family embeds the statistical watermarking signal at only a subset of the generated tokens and also, at detection, scores only that subset. What distinguishes the methods in this family is how they select the subset. Because the green-list bias has little effect where the next-token entropy is low, the common answer is to select high-entropy positions, with the entropy measured by the generating model itself (Lee et al., 2024), by an auxiliary model (Liu & Bu, 2024), or by a lightweight distilled tagger (Gu et al., 2025), and one method learns the selection outright, training a selector network (Wang et al., 2025). These criteria, however, are all functions of a model, so a detector cannot re-derive the positions from the key and the received text alone. SpARK is the closest prior design (Hoang et al., 2026). It marks single tokens at positions chosen either by a secret hash or by the POS tags of the preceding words. Its POS variant comes nearest to our design, demonstrating that a watermark can be anchored in a property the detector re-derives from the text alone. SpanWM builds this kind of anchor. The watermark sites are parsed sentence constituents chosen by the secret key, so anyone with a parser sees the same candidates while only the key holder knows which ones carry the mark. At detection, the same parser and key re-derive the sites from the received text alone, with no model in the loop, and each site contributes evidence over a reconstructed constituent rather than a single constrained token. The difference lies in what sustains each anchor. A POS tag rests on a single word, and one rewritten word is enough to change or lose its role, whereas a constituent spans a phrase whose syntactic type can remain recoverable after word substitutions. Recovering the original site is nevertheless sensitive to changes in its context and parse.

6 Conclusion

We introduce **`SpanWM`**, a low-budget watermarking framework that concentrates statistical evidence in a few key-selected syntactic spans. Starting from an unwatermarked draft, SpanWM locally regenerates selected phrases and re-identifies watermark sites from the received text using a parser and secret keys, without requiring the original prompt or generating model at detection. Experiments across datasets show strong detection and higher semantic fidelity than the compared methods. In a controlled budget sweep, sparse baselines require substantially more tested tokens to match SpanWM's detectability. SpanWM also retains strong detection under paraphrasing and token-level edits. Our role-set analysis further shows that selecting suitable syntactic roles matters more than simply expanding the role inventory. Together, these results show that syntactic spans offer an effective unit for placing and recovering watermark evidence while keeping the watermark budget small and the original meaning intact.

AI USE STATEMENT

In this work, we did not use generative AI tools for any tasks requiring disclosure under the ICLR 2027 AI Policy. Additionally, we used generative AI tools to aid and polish the writing of the paper and for the retrieval and discovery of potentially relevant literature. All AI-assisted writing was reviewed and revised by the authors to ensure that it accurately reflects the intended technical content. Literature identified with the assistance of generative AI was independently checked against the original sources before being cited. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies watermarking methods for identifying and verifying the provenance of LLM-generated text. Such techniques may support the responsible use of generative models by providing evidence about the origin of generated content. However, watermarking research has dual-use implications: evaluations of robustness to paraphrasing and other modifications may also provide information useful for developing watermark-evasion strategies. Moreover, watermark detectors are statistical and may produce false positives or false negatives, so their outputs should not be interpreted as definitive evidence of authorship. This work does not involve human-subject experiments.

REPRODUCIBILITY STATEMENT

Section 2 describes SpanWM's embedding and detection procedures, with pseudocode in Appendix A. Appendix B documents the generation and evaluation protocols, prompt construction, and implementation details. The appendix also reports the watermark configurations, computational environment, additional robustness results, and the full prompt for LLM-based semantic evaluation (Appendix F). An anonymous implementation is available at `https://anonymous.4open.science/r/spanwm_11-BCD6`.

ACKNOWLEDGMENTS

TBD

REFERENCES

Anthropic. System card: Claude Opus 4.6, 2026. URL `https://www-cdn.anthropic.com/0dd865075ad3132672ee0ab40b05a53f14cf5288.pdf`. Technical report, February 2026.

Mikhail J. Atallah, Victor Raskin, Michael Crogan, Christian Hempelmann, Florian Kerschbaum, Dina Mohamed, and Sanket Naik. Natural language watermarking: Design, analysis, and a proof-of-concept implementation. In *Proceedings of the 4th International Workshop on Information Hiding*, LNCS 2137, pp. 185–200, 2001.

Yapei Chang, Kalpesh Krishna, Amir Houmansadr, John Wieting, and Mohit Iyyer. PostMark: A robust blackbox watermark for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8969–8987, 2024.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021.

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

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2019.

Josh A. Goldstein, Girish Sastry, Micah Musser, Renee DiResta, Matthew Gentzel, and Katerina Sedova. Generative language models and automated influence operations: Emerging threats and potential mitigations. arXiv preprint arXiv:2301.04246, 2023.

Google DeepMind. Gemini 3 Pro model card, 2025. URL `https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf`. Model card, November 2025.

Tianle Gu, Zongqi Wang, Kexin Huang, Yuanqi Yao, Xiangliang Zhang, Yujiu Yang, and Xiuying Chen. Invisible entropy: Towards safe and efficient low-entropy LLM watermarking. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6716–6733, 2025.

Jifei Hao, Jipeng Qiang, Yi Zhu, Yun Li, Yunhao Yuan, and Xiaoye Ouyang. Post-hoc watermarking for robust detection in text generated by large language models. In *Proceedings of the 31st International Conference on Computational Linguistics (COLING)*, pp. 5430–5442, 2025.

Duy C. Hoang, Hung T. Q. Le, Rui Chu, Ping Li, Weijie Zhao, Yingjie Lao, and Khoa D. Doan. SpARK: An embarrassingly simple sparse watermarking in LLMs with enhanced text quality. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 4603–4626, 2026.

Maurice Jakesch, Jeffrey T. Hancock, and Mor Naaman. Human heuristics for AI-generated language are flawed. *Proceedings of the National Academy of Sciences*, 120(11):e2208839120, 2023. doi: 10.1073/pnas.2208839120.

Cameron R. Jones and Benjamin K. Bergen. Large language models pass a standard three-party Turing test. *Proceedings of the National Academy of Sciences*, 123(21):e2524472123, 2026. doi: 10.1073/pnas.2524472123. URL `https://www.pnas.org/doi/abs/10.1073/pnas.2524472123`.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Gregory Kang Ruey Lau, Xinyuan Niu, Hieu Dao, Jiangwei Chen, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. Waterfall: Scalable framework for robust text watermarking and provenance for LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 20432–20466, 2024.

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. Who wrote this code? watermarking for code generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip S. Yu. A survey of text watermarking in the era of large language models. *ACM Computing Surveys*, 57(2):1–36, 2024. doi: 10.1145/3691626.

Yepeng Liu and Yuheng Bu. Adaptive text watermark for large language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Travis Munyer, Abdullah All Tanvir, Arjon Das, and Xin Zhong. DeepTextMark: A deep learning-driven text watermarking approach for identifying large language model generated text. *IEEE Access*, 12:40508–40520, 2024.

OpenAI. OpenAI GPT-5 system card. arXiv preprint arXiv:2601.03267, 2026.

Jipeng Qiang, Shiyu Zhu, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. Natural language watermarking via paraphraser-based lexical substitution. *Artificial Intelligence*, 317:103859, 2023.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. Can AI-generated text be reliably detected? stress testing AI text detectors under various attacks. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=OOgsAZdFOt.

Tom Sander, Hongyan Chang, Tomáš Souček, Tuan Tran, Valeriu Lacatusu, Sylvestre-Alvise Rebuffi, Alexandre Mourachko, Surya Parimi, Christophe Ropers, Rashel Moritz, Vanessa Stark, Hady Elsahar, and Pierre Fernandez. TextSeal: A localized LLM watermark for provenance & distillation protection. arXiv preprint arXiv:2605.12456, 2026.

Giovanni Spitale, Nikola Biller-Andorno, and Federico Germani. AI model GPT-3 (dis)informs us better than humans. *Science Advances*, 9(26):eadh1850, 2023. doi: 10.1126/sciadv.adh1850.

Umut Topkara, Mercan Topkara, and Mikhail J. Atallah. The hiding virtues of ambiguity: Quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th Workshop on Multimedia and Security*, pp. 164–174, 2006.

Chenrui Wang, Junyi Shu, Billy Chiu, Yu Li, Saleh Alharbi, Min Zhang, and Jing Li. Learning to watermark: A selective watermarking framework for large language models via multi-objective optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Lingyun Xiang, Yan Li, Wei Hao, Peng Yang, and Xiaobo Shen. Reversible natural language watermarking using synonym substitution and arithmetic coding. *Computers, Materials & Continua*, 55(3):541–559, 2018.

Xi Yang, Jie Zhang, Kejiang Chen, Weiming Zhang, Zehua Ma, Feng Wang, and Nenghai Yu. Tracing text provenance via context-aware lexical substitution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 11613–11621, 2022.

Xi Yang, Kejiang Chen, Weiming Zhang, Chang Liu, Yuang Qi, Jie Zhang, Han Fang, and Nenghai Yu. Watermarking text generated by black-box language models. arXiv preprint arXiv:2305.08883, 2023.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A Algorithms

The following algorithms summarize the embedding and detection procedures described in Section 2. Both use the same keyed NextSite rule to select eligible constituents. Algorithm 1 regenerates selected constituents and checks whether their sites remain recoverable; Algorithm 2 selects sites from the received continuation and tests the pooled green-token evidence.

B Experimental Setup

B.1 Default SpanWM configuration.

We use a green-list fraction of $\gamma = 0.25$ and a logit bias of $\delta = 4.0$, with a one-token green-list context. The constituent inventory is $(\mathrm{NP}, \mathrm{VP}, \mathrm{PP})$, with candidates extracted using spaCy's `en_core_web_sm` and the `benepar_en3` constituency parser. At each candidate start, a keyed

Algorithm 1 SpanWM embedding

Require: Prompt q, model $\mathcal{M}_\theta$, keys $\kappa_{\mathrm{sel}}, \kappa_{\mathrm{wm}}$

$x \sim \mathcal{M}_\theta(\cdot \mid q)$; $f \leftarrow 0$
for $j = 1, \ldots, S_{\max}$ **do**
  $s \leftarrow \textsc{NextSite}(x, f)$
  if $s = \perp$ **then**
    break
  end if
  $c_0 \leftarrow c(s)$; $L_s \leftarrow |\mathcal{T}(x, s)|$; $\mathcal{X} \leftarrow [\,]$
  for $a = 1, \ldots, A_{\max}$ **do**
    for $b = 1, \ldots, B$ **do**
      $x^{(b)} \leftarrow \textsc{GenerateAndSplice}(q, x, s, L_s, \kappa_{\mathrm{wm}})$
      Compute Q_b using Eq. equation 6
    end for
    $b^\star \leftarrow \arg\max_b Q_b$; $x' \leftarrow x^{(b^\star)}$
    $d \leftarrow d(x'; s, f)$ using Eq. equation 7
    Append (x', d) to $\mathcal{X}$
    if $d \le \varepsilon$ **then**
      break
    end if
  end for
  $x \leftarrow$ candidate in $\mathcal{X}$ with minimum d (first on ties)
  $f \leftarrow \textsc{Advance}(x, c_0)$
end for
Record final site alignment; **return** $q \| x$

Algorithm 2 SpanWM detection

Require: Continuation x, keys $\kappa_{\mathrm{sel}}, \kappa_{\mathrm{wm}}$

$f \leftarrow 0$; $\mathcal{S} \leftarrow \emptyset$
for $j = 1, \ldots, S_{\max}$ **do**
  $s \leftarrow \textsc{NextSite}(x, f)$
  if $s = \perp$ **then**
    break
  end if
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{s\}$; $f \leftarrow e(s)$
end for
$\mathcal{P}_+ \leftarrow \{i \in \bigcup_{s \in \mathcal{S}} \mathcal{T}(x, s) : i > 1\}$
$N \leftarrow |\mathcal{P}_+|$
if $N = 0$ **then**
  return NOT WATERMARKED
end if
$(t_1, \ldots, t_n) \leftarrow \tau(x)$; $g \leftarrow 0$
for $i \in \mathcal{P}_+$ **do**
  Reconstruct $\mathcal{G}_i$ from t_{i-1} and κ_{wm}
  $g \leftarrow g + \mathbb{I}[t_i \in \mathcal{G}_i]$
end for
Compute p using Eq. equation 10
return $p < \alpha$

hash of up to $n_g = 16$ preceding characters selects one target constituent type. Eligible constituents contain 2–12 model tokens, and at most $S_{\max} = 4$ insertions are attempted per text. Each regeneration attempt samples $B = 4$ replacement candidates using the original constituent's token length as the generation budget. Candidates are ranked by the mean log likelihood of up to $R = 16$ retained right-context tokens. Reconstruction permits up to $A_{\max} = 4$ attempts per site, with a start-position tolerance of $\varepsilon = 4$ characters.

Implementation details. The parser operates on a copy in which whitespace runs are collapsed and leading whitespace is removed. A character-index map restores constituent boundaries to the original continuation. Nested constituents are retained; for a node with multiple labels, the first label belonging to the configured inventory is used. Duplicate type/start/end triples are removed. Token intervals are trimmed of boundary whitespace before overlap with a constituent is tested. Parser failures produce no candidates.

For replacement generation, one trailing space is removed from the left continuation if present, and the original prompt is prepended. Each candidate is sampled up to three times if its first non-space character is a disallowed closing or separating punctuation mark; the final draw is retained if all three draws have such an opener. Splicing reconciles leading and trailing spaces with the retained context and avoids adding a separator before closing punctuation. These rules do not enforce equality between generated token IDs and the tokenization of the final text.

The right-context score uses the spliced continuation without the prompt or added special tokens. Its first scored token is the first whose untrimmed character interval extends beyond the inserted body's end, so a token crossing that boundary can be included. Up to R tokens are evaluated using the base model's logits without watermark bias. An empty or unseedable scoring range receives score zero, and ties are resolved in sampling order.

After each edit, let c_0 be the pre-edit start. The next embedding boundary is the end of $\textsc{NextSite}(x, \max(0, c_0 - \varepsilon))$, or $c_0 + 1$ if none exists. The detector instead advances from each selected constituent to its end. Consequently, embedding can revisit or overlap earlier edited regions, and later edits can change earlier parses. Final verification compares the ordered embedding and detection starts within ε characters and records whether all sites align. It neither repairs the text nor excludes failed alignments from evaluation. The keys in the released configuration are reproducibility keys.

Generation and evaluation. We evaluate Llama-3.2-3B, Llama-3.1-8B, Qwen3-4B, and Qwen3-8B on C4, CNN/Daily Mail, and WMT16, using 200 prompts per dataset. All generators operate in causal completion mode without chat templates. Draft generation and local regeneration use nucleus sampling with top-$p = 0.9$ and temperature 0.8; draft generation is capped at 200 new tokens. C4 uses the supplied processed prompts. For CNN/Daily Mail, the first 30 words of each article form the prompt; for WMT16, we use English-side validation sentences containing at least 10 words.

SpanWM detection operates on the received continuation, excluding the original prompt, using the tokenizer, constituency parser, and watermark keys without loading the generation language model. The detector selects at most four constituents and pools their green-token evidence over the actual token ranges, yielding a variable test size of $N \leq 48$. The nominal decision rule is $p < 0.01$ under the binomial reference tail in Eq. equation 10. AUROC is computed from the monotone score $-\log_{10} p$, with texts containing no valid tested tokens assigned the lowest ranking score. The reported z-scores and p-values summarize the watermarked continuations under each evaluation condition, while ROC operating points use empirical negative scores. With 200 negative examples, the empirical FPR resolution is 0.5%; results at 0.1% FPR therefore do not establish a validated operating point at that rate.

Semantic preservation for SpanWM is evaluated between the pre-insertion draft and the final watermarked continuation. Cosine similarity is computed with `Qwen3-Embedding-4B`, using the last non-padding token representation, L2 normalization, and a maximum input length of 1024 tokens. We additionally evaluate the first 100 nonempty pairs using `gpt-5.6-luna` with low reasoning effort and the 0–10 similarity rubric in Appendix F. Both evaluations exclude the original prompt. Baseline similarity evaluations use their independently generated unwatermarked continuations as references.

B.2 Computational Environment

Experiments were run on a system equipped with four NVIDIA RTX PRO 6000 GPUs. The generation and detection pipelines used the `spanwm` conda environment with Python 3.12.13, PyTorch 2.13.0 with CUDA 13.0, and Transformers 5.14.1. Syntactic processing used spaCy 3.8.14 and benepar 0.2.0, with `en_core_web_sm` version 3.8.0 and `benepar_en3`. Generation and detection use the same vocabulary size and device type to ensure consistent green-list permutations.

C Full Results for Additional Generators

Table 5 reports the full detection and generation-quality results for Llama-3.1-8B, Qwen3-4B, and Qwen3-8B, complementing the Llama-3.2-3B results in the main paper. SpanWM achieves the highest AUROC, cosine similarity, and LLM score in all nine generator–dataset settings. It also achieves the highest TPR at 0.1% FPR in seven of the nine settings. The exceptions are CNN/DM with Llama-3.1-8B and Qwen3-8B, where SpARK-R and SWEET, respectively, have higher TPR at that operating point.

D Attack Robustness Results

Table 6 reports the complete AUROC results under token insertion, deletion, and swap attacks on C4 with Llama-3.2-3B, complementing the robustness curves in the main paper. SpanWM achieves the highest AUROC at every tested edit ratio across all three attacks. Even at a 50% edit ratio, it retains AUROC scores of 0.967, 0.965, and 0.966 under insertion, deletion, and swap, respectively. Relative to the clean setting (0.982), these correspond to drops of only 0.015–0.017.

In contrast, the baseline methods generally degrade more substantially as the edit ratio increases. SpARK-R is the strongest baseline under insertion and swap at high edit ratios, reaching 0.916 and 0.822 AUROC, respectively, at 50%, while SWEET performs best among the baselines under deletion with 0.883 AUROC. SpanWM nevertheless maintains a clear advantage across all three attack types, indicating that its localized watermark evidence remains detectable under substantial token-level modification.

E Full Role Set Ablation

Section 4.3 discusses the aggregate role-set ablation. Table 7 provides the corresponding results for each generator, including site alignment, the number of tested tokens, and detection performance. The tested-token count remains similar across role sets, but the strongest configuration varies by generator, particularly at 1% FPR.

Table 5: Detection performance and generation quality across models and datasets.

Model	Dataset	Method	AUROC	TPR @ FPR =				z-score	Cos. sim.	LLM score
				10%	5%	1%	0.1%			
Llama-3.1-8B	C4	KGW	0.797	0.50	0.33	0.03	0.03	1.46	0.711	3.52
		SWEET	0.935	0.79	0.77	0.70	0.60	3.25	0.591	3.56
		SpARK-R	0.934	0.86	0.76	0.25	0.01	3.27	0.593	3.50
		IE	0.841	0.65	0.57	0.40	0.19	1.65	0.713	3.53
		SpanWM *(Ours)*	**0.977**	**0.94**	**0.91**	**0.78**	**0.63**	**3.98**	**0.947**	**7.15**
	CNN/DM	KGW	0.808	0.55	0.41	0.16	0.11	1.79	0.790	4.01
		SWEET	0.872	0.76	0.57	0.41	0.29	2.39	0.655	4.15
		SpARK-R	0.934	0.85	0.79	0.53	**0.48**	3.15	0.648	4.07
		IE	0.678	0.34	0.29	0.10	0.07	0.91	0.803	4.07
		SpanWM *(Ours)*	**0.960**	**0.92**	**0.86**	**0.75**	0.39	**3.63**	**0.958**	**7.24**
	WMT16	KGW	0.776	0.45	0.38	0.19	0.18	1.50	0.674	2.92
		SWEET	0.940	0.84	0.79	0.72	0.63	3.54	0.526	3.21
		SpARK-R	0.925	0.79	0.66	0.57	0.26	3.15	0.534	3.20
		IE	0.788	0.50	0.44	0.27	0.16	1.55	0.672	2.88
		SpanWM *(Ours)*	**0.981**	**0.95**	**0.93**	**0.83**	**0.72**	**4.16**	**0.937**	**7.22**
Qwen3-4B	C4	KGW	0.737	0.30	0.15	0.04	0.02	1.17	0.683	3.68
		SWEET	0.891	0.73	0.64	0.54	0.51	2.80	0.571	3.49
		SpARK-R	0.813	0.55	0.34	0.13	0.03	1.94	0.563	3.53
		IE	0.812	0.54	0.47	0.34	0.13	1.44	0.694	3.65
		SpanWM *(Ours)*	**0.967**	**0.88**	**0.81**	**0.71**	**0.68**	**3.36**	**0.955**	**7.56**
	CNN/DM	KGW	0.751	0.42	0.32	0.08	0.03	1.32	0.769	3.83
		SWEET	0.823	0.62	0.52	0.45	0.40	1.74	0.643	3.77
		SpARK-R	0.797	0.41	0.28	0.09	0.01	1.89	0.631	3.62
		IE	0.773	0.42	0.29	0.13	0.09	0.97	0.770	3.70
		SpanWM *(Ours)*	**0.937**	**0.86**	**0.77**	**0.51**	**0.47**	**2.89**	**0.953**	**7.50**
	WMT16	KGW	0.697	0.29	0.13	0.03	0.01	1.08	0.646	3.36
		SWEET	0.869	0.71	0.61	0.52	0.49	2.52	0.545	3.17
		SpARK-R	0.832	0.57	0.40	0.29	0.01	1.54	0.528	3.05
		IE	0.803	0.55	0.48	0.41	0.29	1.62	0.646	3.40
		SpanWM *(Ours)*	**0.943**	**0.86**	**0.77**	**0.64**	**0.60**	**3.23**	**0.952**	**7.77**
Qwen3-8B	C4	KGW	0.739	0.33	0.26	0.07	0.02	1.26	0.691	3.64
		SWEET	0.934	0.87	0.83	**0.76**	0.53	**3.63**	0.572	3.63
		SpARK-R	0.841	0.59	0.40	0.08	0.04	2.14	0.579	3.68
		IE	0.819	0.60	0.43	0.28	0.21	1.69	0.698	3.67
		SpanWM *(Ours)*	**0.954**	**0.88**	**0.84**	0.71	**0.59**	3.52	**0.950**	**7.49**
	CNN/DM	KGW	0.739	0.37	0.26	0.03	0.01	1.63	0.792	3.93
		SWEET	0.900	0.77	0.69	0.54	**0.51**	2.46	0.660	4.04
		SpARK-R	0.929	0.78	0.68	0.47	0.38	2.64	0.646	4.04
		IE	0.815	0.58	0.51	0.36	0.31	1.50	0.793	3.93
		SpanWM *(Ours)*	**0.950**	**0.89**	**0.79**	**0.64**	0.42	**2.98**	**0.949**	**7.51**
	WMT16	KGW	0.702	0.31	0.19	0.05	0.01	1.04	0.659	3.53
		SWEET	0.881	0.69	0.59	0.49	0.24	2.65	0.526	3.56
		SpARK-R	0.840	0.59	0.45	0.21	0.04	1.74	0.535	3.68
		IE	0.821	0.60	0.49	0.32	0.26	1.68	0.655	3.55
		SpanWM *(Ours)*	**0.911**	**0.82**	**0.76**	**0.66**	**0.60**	**3.09**	**0.950**	**7.65**

Table 6: AUROC under token-level edits on C4 with Llama-3.2-3B. Token insertion, deletion, and swap are applied to the watermarked text at the indicated edit ratios; 0% denotes unedited text. The highest AUROC for each attack and ratio is shown in bold.

Attack	Method	Attack ratio					
		0% (clean)	**10%**	**20%**	**30%**	**40%**	**50%**
Token insertion	KGW	0.814	0.780	0.753	0.737	0.724	0.698
	SWEET	0.965	0.928	0.886	0.862	0.840	0.798
	SpARK-R	0.935	0.934	0.927	0.924	0.924	0.916
	IE	0.870	0.863	0.847	0.822	0.797	0.782
	SpanWM *(Ours)*	**0.982**	**0.974**	**0.974**	**0.974**	**0.976**	**0.967**
Token deletion	KGW	0.814	0.813	0.783	0.726	0.684	0.667
	SWEET	0.965	0.951	0.923	0.867	0.884	0.883
	SpARK-R	0.935	0.929	0.917	0.900	0.897	0.880
	IE	0.870	0.835	0.806	0.789	0.773	0.789
	SpanWM *(Ours)*	**0.982**	**0.968**	**0.968**	**0.972**	**0.965**	**0.965**
Token swap	KGW	0.814	0.799	0.745	0.670	0.679	0.654
	SWEET	0.965	0.921	0.860	0.852	0.791	0.751
	SpARK-R	0.935	0.917	0.896	0.875	0.849	0.822
	IE	0.870	0.821	0.802	0.788	0.774	0.774
	SpanWM *(Ours)*	**0.982**	**0.975**	**0.975**	**0.972**	**0.970**	**0.966**

Model	$\lvert\mathcal{R}\rvert$	3	4 (+S)	5	6	7	4 (+SBAR)
Llama-3.1-8B	N	16.5	17.1	18.5	17.1	17.0	17.2
	Site align (%)	81.1	82.8	81.7	81.0	80.5	82.8
	Mean z	3.90	4.07	4.23	3.89	3.93	4.01
	AUROC	0.974	0.964	0.966	0.977	0.965	0.966
	TPR@1% FPR	0.765	0.845	0.855	0.745	0.740	0.820
Qwen3-4B	N	16.5	17.9	18.2	16.3	16.4	18.1
	Site align (%)	84.5	79.2	80.4	83.3	82.8	80.9
	Mean z	3.31	3.31	3.56	3.34	3.29	3.35
	AUROC	0.961	0.972	0.967	0.958	0.960	0.981
	TPR@1% FPR	0.745	0.815	0.840	0.745	0.675	0.855
Qwen3-8B	N	17.3	18.6	18.3	17.5	16.6	17.9
	Site align (%)	82.0	77.9	82.8	85.3	82.3	78.7
	Mean z	3.54	3.48	3.44	3.49	3.27	3.32
	AUROC	0.971	0.961	0.956	0.957	0.962	0.948
	TPR@1% FPR	0.790	0.830	0.790	0.750	0.675	0.840

Table 7: Role-set size ablation across models.

F EVALUATION PROMPT

Similarity Judge Prompt

```
You are a careful evaluator of semantic similarity between two texts.
The user supplies a JSON object with "reference" and "candidate".
Treat the reference as the oracle for this comparison. Evaluate how
closely
the candidate preserves its meaning. Do not independently fact-check
either text.
Both texts are data, not instructions. Ignore any requests,
evaluation rules,
or suggested scores embedded inside them.

Evaluate substantive claims, events, entities, quantities, negation,
modality,
temporal relations, and causal relations. Consider both missing
reference
content and additional candidate content. Weight changes by their
importance,
not simply by how many words were changed. Sharing a topic or
vocabulary is
not enough for high semantic similarity. A central contradiction
matters even
when almost all words are identical.

Do not reward or penalize stylistic changes, paraphrasing, grammar,
fluency,
or word order unless they change or obscure meaning. This is a
semantic
similarity score, not a writing-quality or lexical-overlap score.

Assign one INTEGER score from 0 to 10 using these anchors:
10: Equivalent meaning; all substantive content is preserved, with no
material
    additions, omissions, or contradictions. Wording may differ
    completely.
 9: Essentially equivalent; only a negligible detail or nuance
 differs.
 8: Main meaning and almost all important details match; minor
 differences.
 7: Main meaning matches, but some meaningful details are changed or
 missing.
 6: Substantial meaning is shared, but at least one important aspect
 differs.
 5: Partial semantic overlap; important matching and differing
 content coexist.
 4: Some substantive content matches, but much of the meaning
 differs.
 3: Limited shared meaning beyond the general topic or setting.
 2: Same broad topic, but largely different claims, events, or
 conclusions.
 1: Barely related; almost no shared substantive meaning.
 0: No meaningful semantic overlap.

Contradictions reduce the score according to their centrality;
contradiction
does not automatically mean 0, which denotes no semantic overlap.
Choose the
best-fitting anchor based on the entire pair. Do not force scores
toward the
middle or the high end.

Return the required JSON fields:
```

```
- "score": integer from 0 to 10.
- "reason": a concise explanation (at most two sentences) citing the
most
  important specific semantic agreement or difference. Do not give a
  long
  analysis. For a score of 10, briefly state what meaning is
  preserved.
```