

Semantic Watermarking with Order-Robust Detection over Sub-sentence Units

Abdulrahman Diaa
University of Waterloo

Jonathan Petit
Qualcomm

Florian Kerschbaum
University of Waterloo

Abstract

Semantic watermarks tie the mark to sentence meaning rather than token choices, promising robustness to content-preserving edits. However, the detector only observes attacker-supplied text, which can be reworded, reordered, or resegmented to evade detection without content loss. Rewording, reordering, and resegmentation all cause *embedding displacement*: detection tests embeddings different from those selected during watermarking and can therefore lose the mark. Our adaptive embedding displacement attack (EDA) admits all three edits under a single objective that maximizes this displacement. It uses a public paraphraser and surrogate encoder without access to the provider’s generator or secret key. At a 5% false-positive rate (FPR) and *content-preservation threshold* $\bar{q} = 90\%$, EDA successfully removes the mark on 32.6%–47.9% of documents across four schemes, the highest among the tested attacks. Therefore, EDA evaluates the schemes’ robustness more thoroughly than passive paraphrasing.

To address these vulnerabilities, we design (k)-SwordStamp: semantic watermarks with order-robust detection over sub-sentence units, reducing sensitivity to attacker-chosen structure at a small quality cost. Against k-SwordStamp, the strongest no-box attack we test is an EDA variant adapted to its design, with a 10.8% attack-success rate. A stronger EDA with access to the provider’s detector and secret key reaches a 39.7% attack-success rate, compared with 65.5% on k-SemStamp. Our code is available at <https://github.com/D-Diaa/SwordStamp>.

1 Introduction

A text watermark embeds a hidden signal during generation so that a detector with a secret key can later identify the resulting text as AI-generated. Providers are beginning to deploy text watermarks. Google deploys SynthID-Text in Gemini [11]. Anthropic has announced that future Claude models will use a version of SynthID-Text to meet the European Union’s AI Act marking requirements [2, 13].

Users may edit generated text before using it in an article, email, or report. A detector may therefore receive only the edited version. A useful watermark should remain detectable through meaning-preserving edits while its detector maintains a low FPR on human text. Evasion alone does not make an attack successful, because an attacker that discards the content evades any detector and gains nothing from the provider. We score how well an attack output preserves the marked text’s content. $\bar{q}$ is the minimum content-preservation percentage an attacker must reach. An attack is $\bar{q}$ -successful when its output both evades the detector and scores at least $\bar{q}$. Most text watermarks operate through token choices, which paraphrasing can remove at a high $\bar{q}$ [12, 26]. Encoder-based watermarks (EBWs) instead mark whole sentences [17, 18, 20, 21]. A sentence encoder maps each sentence to a vector, and the secret key selects a set of regions of that embedding space. (k)-SemStamp [17, 18] generate sentences by *rejection sampling*: they draw candidate sentences from the language model and keep the first one whose embedding falls inside a selected region. The detector re-embeds the sentences of a suspect text and counts how many of them land in selected regions. The implicit assumption is that meaning-preserving edits tend to keep a sentence’s embedding within the same region, so an evading rewrite would require a change in meaning.

Most prior evaluations of EBWs only consider sentence-by-sentence paraphrasing that preserves sentence order and boundaries [17, 18]. These evaluations do not cover an attacker who changes sentence structure. A whole-text attacker can preserve content while reordering sentences, splitting one sentence into two, or merging adjacent sentences. These edits change either the unit that the detector embeds or the position at which it tests that unit.

We apply rewording, reordering, and resegmentation separately to measure each edit’s effect on detection. Although these edits differ, each causes the detector to evaluate an embedding different from the one selected during watermarking. Rewording shifts the original sentence’s embedding, reordering moves a different sentence into the tested position, and resegmentation changes the unit that the detector embeds.

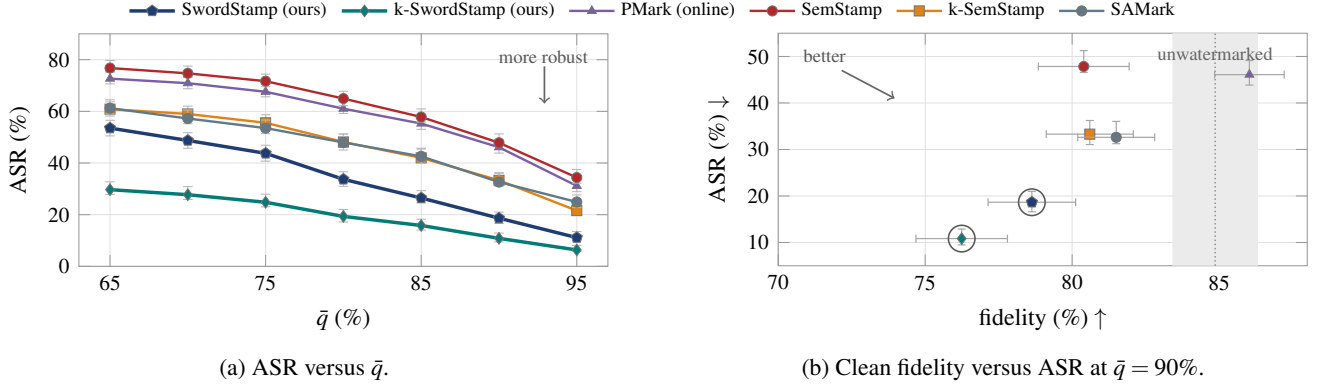


Figure 1: Strongest no-box attack success rate (ASR) at 5% FPR across attack families and settings. Vertical bars give percentile-bootstrap 95% intervals for maximum ASR; horizontal bars in (b) give 95% confidence intervals for clean fidelity. The shaded region in (b) gives the unwatermarked fidelity interval, and the dotted line marks its mean.

This shared *embedding displacement* effect is the basis for our embedding displacement attack (EDA). EDA observes only the marked text and cannot query the detector or the generator, matching the *no-box* setting of prior work [12]. It runs the provider’s own procedure with the opposite objective. The provider samples candidate sentences and keeps one that lands inside a selected region; EDA rewrites the text one sentence at a time and, at each step, keeps the candidate farthest from the source sentence under a public surrogate encoder. The surrogate differs from every provider encoder we test, yet candidates displaced farther under the surrogate also tend to be displaced farther under the provider’s encoder (Figure 2). Because each candidate continues a rewrite of the whole text rather than replacing one sentence in isolation, EDA can also reorder content and change sentence boundaries.

We evaluate EDA against four EBWs: SemStamp, k-SemStamp, PMark, and the concurrent SAMark [17, 18, 20, 21]. At 5% FPR and $\bar{q} = 90\%$, EDA is $\bar{q}$ -successful on 32.6%–47.9% of marked documents, the highest success of the attacks we test on each of the four schemes. SAMark resists the whole-text paraphraser Dipper [26] best among the four, holding it to 18.7%, yet EDA succeeds on 32.6% of its documents. At this operating point, EDA reaches a higher ASR on each scheme than every tested watermark-agnostic paraphraser, including Pegasus [44], Parrot [10], and Dipper.

To improve the robustness of EBWs, we add three mechanisms to (k)-SemStamp. Best-of- N selection scores a fixed pool of candidate sentences and emits the one whose embedding sits farthest inside a selected region, so a rewrite must travel farther to leave it (§5.1). A fixed valid set uses the same key-selected regions at every position instead of deriving them from the preceding sentence, which makes detection order-robust (§5.2). Deterministic sub-sentence units, which we call *semspans*, follow changes in meaning rather than punctuation and reduce sensitivity to sentence splits and merges (§5.3). Applying these mechanisms produces (k)-SwordStamp.

Figure 1 compares schemes under their strongest tested attacks, including EDA-S, the EDA variant adapted to every scheme’s design (§5.4). Across all tested $\bar{q}$, k-SwordStamp has the lowest attack success among the six schemes. At 5% FPR and $\bar{q} = 90\%$, SwordStamp reduces scheme-specific EDA attack success by 29.2 percentage points relative to SemStamp, at a fidelity cost of 1.8 percentage points. At the same operating point, k-SwordStamp reduces scheme-specific EDA attack success by 22.5 percentage points relative to k-SemStamp, at a fidelity cost of 4.4 percentage points. Figure 1b places this trade-off against all six schemes at the same operating point.

As a stress test beyond the no-box setting, we evaluate detector-access EDA, which receives the provider’s detector, including its secret key and encoder. This attacker evaluates each candidate with the detector and selects the closest candidate that the detector misses. With four candidate rewrites per step, detector evasion is 29.6% on k-SwordStamp, compared with 70.2% on k-SemStamp, showing that evading rewrites are harder to find. At the largest search budget, detector evasion reaches 85.4%, but only 46.5% of them clear $\bar{q}$ (Figure 7). This caps the attack’s $\bar{q}$ -success rate at 39.7%, compared with 65.5% on k-SemStamp, and demonstrates k-SwordStamp’s improved robustness to content-preserving attacks.

We make three contributions: (1) We measure how rewording, reordering, and resegmentation affect EBW detection and consolidate them into EDA, a single adaptive whole-text attack guided by embedding displacement; (2) We develop (k)-SwordStamp by extending (k)-SemStamp with best-of- N selection, order-robust detection, and sub-sentence units; and (3) We evaluate six watermarks under the $\bar{q}$ -success criterion, measure each mechanism’s effects on clean detection, output quality, and attack success, and stress-test k-SwordStamp and k-SemStamp under detector access.

2 Text Watermarking

We formalize two types of text watermarking: token-level watermarks and encoder-based watermarks. Table 1 summarizes our notation.

Watermarking and detection. An autoregressive language model $\mathcal{M}$ defines a distribution $P_{\mathcal{M}}(x_t | x_{<t})$ over the next token and generates text by sampling from it left to right. We define a *text watermark* as a triple of randomized algorithms (Setup, Watermark, Detect) over a key space $\mathcal{K}$ and a text space $\mathcal{Y}$. Setup $\rightarrow k \in \mathcal{K}$ draws a secret key. Watermark(k) $\rightarrow x \in \mathcal{Y}$ samples text from $\mathcal{M}$ so that k leaves a statistical trace in x . Detect(k, y) $\rightarrow z \in \mathbb{R}$ maps a suspect text y and key k to a score z , where a larger z indicates a stronger trace of k . We write $z_k(y)$ for the score that Detect assigns to text y under key k . The target FPR ρ is the probability that the detector flags unwatermarked text. The provider calibrates the corresponding score threshold $\tau(\rho)$, and the detector flags y as watermarked when $z_k(y) > \tau(\rho)$. Concretely, the *null distribution* is the distribution of z_k on unwatermarked text, and $\tau(\rho)$ is its $(1 - \rho)$ -quantile. The provider estimates this null from a held-out sample of human-written text, or computes it analytically when it has a closed form. We study *zero-bit* watermarks, which embed one signal, so Detect reports only the key’s trace, not an embedded message.

Token-level watermarks. Token-level watermarks operate on the model’s vocabulary V , treating the output as a token sequence. The green-red scheme of Kirchenbauer et al. [25] (KGW) hashes a window of preceding tokens to seed a pseudo-random split of V at each position. Each split partitions V into a *green list* of size $\gamma|V|$ and a red complement. KGW then adds a fixed logit bias δ to green tokens so the model emits them more often. Throughout, $\gamma \in (0, 1)$ is the fraction of the space that the key marks *valid* at one position. Detection applies the following one-proportion z -test to the green-token count H among n tokens.

$$z = \frac{H - \gamma n}{\sqrt{n\gamma(1-\gamma)}} \quad (1)$$

The statistic standardizes the green-token count, so the unwatermarked null centers at zero. We call such a partition *context-dependent*: the green-red split is re-derived at every position from the preceding tokens. Substituting a token can turn a green-token hit into a miss at that position. The substitution also re-seeds subsequent splits, so one edit can disrupt the mark at later positions. The Unigram scheme of Zhao et al. [48] instead uses a *fixed* split, one green list for the entire generation, independent of context. Zhao et al. prove that under the fixed split each token edit changes the detection score by a bounded amount. For sufficiently high-entropy text, evading detection therefore requires a number of edits that grows linearly with the text length. Meaning-preserving rewording can nevertheless change enough token choices to evade token-level watermarks [12, 26].

Table 1: Notation used throughout the paper.

Symbol	Meaning
x, y	marked text; attacked text $\mathcal{A}(x)$
k	secret watermarking key
Emb, $\widetilde{\text{Emb}}$	provider encoder; attacker surrogate
$P, r(s)$	partition into c regions; region index of s
Seg	segmenter into sentence or semspan units
G, B	valid and blocked region sets; G_t per step
γ	valid-region fraction; null hit rate
N, K	provider and attacker candidate budgets
n, H	scored units; valid-region hits
$z, \rho, \tau(\rho)$	detection score; target FPR; calibrated threshold
$Q(x, y), \bar{q}$	content-preservation percentage; minimum threshold
$F(x)$	fidelity percentage of marked text, no reference

Encoder-based watermarks. A second family responds to this fragility by raising the unit of marking from the token to the sentence. These schemes treat text as a sentence sequence $x = (s_1, \dots, s_n)$ and assign the mark to sentence meaning [17, 18, 20, 21]. An encoder Emb maps each sentence s to a vector $\text{Emb}(s) \in \mathbb{R}^h$, and a partition $P = \{R_1, \dots, R_c\}$ splits $\mathbb{R}^h$ into c regions, so each sentence inherits the region of its embedding. We write $r(s)$ for the *region of* s . Every scheme we study depends on a sentence encoder to place and verify its mark, so we call the family *encoder-based watermarks* (EBWs). The detector operates only on the suspect text and tests whatever sentence embeddings it recovers. Under a fixed partition P , a recovered sentence s' that rewrites s in-place keeps $r(s') = r(s)$ whenever it moves the embedding less than the distance from $\text{Emb}(s)$ to the nearest boundary. The *EBW robustness premise* is that a content-preserving rewrite usually moves the embedding by less than that distance, so the mark survives rewording.

SemStamp [17] forms the partition with locality-sensitive hashing (LSH) [6, 22]: b random hyperplanes through the origin induce $c = 2^b$ regions. At step t , the previous sentence’s region $r(s_{t-1})$ seeds a pseudo-random split of the region indices $\{1, \dots, c\}$ into a *valid* set G_t of size γc and a *blocked* set B_t , the semantic counterpart of green and red. This seeding makes the partition context-dependent in the same sense as KGW, with $r(s_{t-1})$ in place of the preceding-window hash. The provider generates s_t by rejection sampling, drawing candidates from $\mathcal{M}$ and accepting the first whose region is valid, $r(s_t) \in G_t$. A margin δ additionally rejects any candidate whose embedding lies within δ of a region boundary. SemStamp caps rejection at a *maxout* budget of N candidates, then emits the last draw even when it fails the valid-region or margin condition. k-SemStamp [18] replaces the hyperplanes with k -means clusters: (i) the c regions are the Voronoi cells of centroids fitted to a domain corpus, (ii) $r(s)$ is the index of the centroid nearest $\text{Emb}(s)$, and (iii) the margin becomes the

minimum gap to the next-nearest centroid.

Following SemStamp’s private-encoder assumption [17], our threat model treats the deployed encoder checkpoint Emb, partition geometry, and large prime p as provider-private key material for both SemStamp and k-SemStamp. SemStamp draws its hyperplanes at Setup, whereas k-SemStamp fits its centroids to a domain corpus and keeps them private. Both schemes fine-tune Emb contrastively on paraphrase pairs to reinforce the robustness premise. To verify a suspect text, both schemes segment it into sentences and re-derive each step’s valid set from the suspect text itself, seeding G_t on the recovered $r(s_{t-1})$. The detector then counts the valid-region hits H among the n recovered sentences and applies the same one-proportion z -test of Equation (1). We call SemStamp and k-SemStamp the *(k)-SemStamp family*.

PMark [20] computes its partition geometry from candidates sampled from $\mathcal{M}$ at each sentence position rather than fixing it at Setup. A *channel* is a secret vector $v_j \in \mathbb{R}^h$ with *proxy function* $f_j(s) = \langle v_j, \text{Emb}(s) \rangle$; PMark maintains b mutually orthogonal channels. At each position, PMark samples N candidates from $\mathcal{M}$ and processes them through the b channels in sequence. In channel j , it sets the position-specific boundary $m_{t,j}$ to the median of $f_j(s)$ over candidates that survived the preceding channels. We write $r_{t,j}(s) \in \{0, 1\}$ for the *region of s at position t in channel j* , the side of the position-specific boundary $m_{t,j}$ on which $f_j(s)$ falls. We write $k_{t,j} \in \{0, 1\}$ for the *key bit* that designates the valid side, drawn independently and uniformly at random for each position t and channel j . PMark then retains the half whose region matches $k_{t,j}$. Sampling uniformly from the final survivors leaves the model’s output distribution unchanged in expectation over the keys, so PMark is *distortion-free* by construction [20]. Since each median splits its input in half, a sentence drawn from $\mathcal{M}$ falls on the valid side of a channel with probability $1/2$.

Detection re-estimates these per-position medians from $\mathcal{M}$: at each position t , PMark resamples N candidates conditioned on the preceding context and takes $m_{t,j}$ as the median of f_j over that pool. Sentence s_t contributes channel evidence according to $r_{t,j}(s_t)$ and $k_{t,j}$. PMark aggregates the bn channel-sentence evidence terms with a soft z -test version of Equation (1). PMark uses a standard-normal approximation to set $\tau(\rho)$ rather than estimating it from human text. Note that detection requires querying $\mathcal{M}$ to re-estimate the medians. PMark also defines an *offline* variant that uses a fixed prior median score as a boundary, removing the need for generator queries at detection time but giving up the distortion-free guarantee. Huo et al. report lower robustness for the offline variant under paraphrase attack [20]. Because robustness to paraphrase attack is our central question, we study the more robust online variant throughout.

Concurrent work, SAMark [21], extends the offline variant to improve robustness. SAMark replaces PMark’s per-position key bit $k_{t,j}$ with one bit k_j per channel, drawn once per query and held constant across the response. Every sen-

tence then scores against the same bit, so the detector is order-robust, matching the property that Unigram provides at the token level. Unlike Unigram’s green list, however, these bits are not long-lived key material: SAMark never shares them with the detector. Detection is therefore *self-anchored*: the detector infers each bit by majority vote over the suspect text, setting $\hat{k}_j = 1$ exactly when $\sum_{i=1}^n f_j(s_i) > 0$. The vote returns whichever side the text already favors, so the score rewards any text whose sentences concentrate on one side of a channel, watermarked or not. SAMark estimates $\tau(\rho)$ from a human-text sample.

3 Threat Model

We frame our evaluation as a game between a provider that watermarks generated text and an attacker that seeks to remove the watermark while preserving its content. We specify each party’s knowledge, capabilities, and success condition.

Parties. The *provider* controls a language model that generates text in response to user prompts. At generation time, the provider runs the watermarking algorithm with a secret *key* k . At verification time, the provider scores a suspect text with Detect and flags it when the score exceeds the FPR-calibrated threshold $\tau(\rho)$ of §2. Following Kerckhoffs’s principle, the algorithm itself is public. The *attacker* receives marked text x from the provider, applies a perturbation $\mathcal{A}$ to produce attacked text $y = \mathcal{A}(x)$, and seeks to drive the detector score to or below $\tau(\rho)$ while preserving the content of x . We study watermark removal only. Forging a mark onto text that the provider did not generate and extracting the secret key are separate adversarial objectives [49] that lie outside our scope.

Attacker capabilities. Our primary attacker is *no-box* [12]: it observes only the released text, does not obtain a corpus of marked texts across queries, and cannot query the detector or generator. A no-box attacker may run its own models, including open paraphraser and a sentence encoder. We restrict the attacker to lower-fidelity generators because an equally capable generator could produce the desired unwatermarked text instead of scrubbing the provider’s output. Attackers also differ in whether they exploit knowledge of the watermarking algorithm. A *passive* attacker applies watermark-agnostic, content-preserving perturbations, whereas an *adaptive* attacker optimizes its perturbation against the known algorithm, following the adaptive-attack paradigm [12, 31]. Every attacker is *effort-constrained*: depending on the attack family, its budget counts model calls, candidate perturbations, or optimization steps. Each attack we instantiate has its own effort parameter, and we trace how attack success changes as the budget grows. A detector-access stress test grants the attacker access to the provider’s secret key (§6.4).

Attack success. Attack success requires both detector evasion and content preservation; discarding content can trivially evade detection but fails the quality constraint [49]. The content-preservation percentage score $Q(x, y) \in [0\%, 100\%]$

measures how well attacked text y preserves marked text x . At target FPR ρ , a $\bar{q}$ -successful attack lowers the detection score to or below $\tau(\rho)$ while retaining $Q(x, y) \geq \bar{q}$. For a perturbation strategy $\mathcal{A}$, this joint event has probability

$$p_{\mathcal{A}}(\rho, \bar{q}) = \Pr_{\substack{k \sim \text{Setup} \\ x \sim \text{Watermark}(k) \\ y = \mathcal{A}(x)}} [z_k(y) \leq \tau(\rho) \wedge Q(x, y) \geq \bar{q}]. \quad (2)$$

The probability is over the secret key k , marked text x generated by $\text{Watermark}(k)$ on prompts drawn from a fixed distribution, the randomness of $\mathcal{A}$, and, for PMark, the detector’s resampling. A watermark is $(\epsilon, \bar{q})$ -robust against $\mathcal{A}$ at FPR ρ when $p_{\mathcal{A}}(\rho, \bar{q}) \leq \epsilon$. We estimate $p_{\mathcal{A}}(\rho, \bar{q})$ with the *attack success rate* ($\text{ASR@}\rho$), the percentage of marked texts on which the attack is $\bar{q}$ -successful at that FPR. Each attack $\mathcal{A}$ runs at a fixed effort budget, so $p_{\mathcal{A}}(\rho, \bar{q})$ characterizes that attack at that budget alone. Robustness is therefore a frontier over attack families, effort budgets, and $\bar{q}$ values. We report ASR versus $\bar{q}$ throughout (§6).

Provider success. The provider seeks to preserve output fidelity before attack while minimizing ASR across the attack frontier. *Fidelity* $F(x) \in [0\%, 100\%]$ is the quality of marked text x judged on its own, with no reference text. F assigns a percentage score to one text, whereas Q scores an attacked text against the marked text it came from. §6.1 describes how we measure both.

4 Edit Sensitivity and EDA

The EBW robustness premise covers in-place *rewording*, not two structural changes available to an attacker who controls the whole text: *reordering* changes the context or position that determines a test, and *resegmentation* changes the units the detector recovers. We first use controlled edits to measure each edit’s effect, then consolidate them into EDA, a general adaptive no-box attack guided by *embedding displacement*. **Reordering.** An attacker can reorder a marked text’s sentences and adjust its references and connectives while preserving content (§3). For context-dependent watermarks, reordering changes the predecessor that keys region tests even when it leaves every sentence unchanged. Let σ map each destination position t to the source index $\sigma(t)$, so the reordered sequence is $(s_{\sigma(1)}, \dots, s_{\sigma(n)})$. For $t > 1$ with $\sigma(t) > 1$, generation selected $s_{\sigma(t)}$ under the valid set keyed by its original predecessor region $r(s_{\sigma(t)-1})$. The detector instead tests it under the set keyed by its reordered predecessor region $r(s_{\sigma(t-1)})$. If these predecessor regions differ, the detector uses a pseudo-randomly unrelated valid set, so $s_{\sigma(t)}$ passes with probability γ over the secret split. If $\sigma(t) = 1$, the detector newly scores the original seed; a sentence moved to the first position instead becomes the unscored seed. When reordering changes the predecessor region across the scored transitions, the expected hit rate returns to γ and the expected z -score approaches zero.

For PMark, the channel test at position t compares $r_{t,j}(s_{\sigma(t)})$ against the position-keyed bit $k_{t,j}$. Because $k_{t,j}$ is independent and uniform, a moved sentence matches with probability $1/2$. A permutation that moves every sentence therefore returns the hit count to random, again driving the expected z -score toward zero. PMark’s dynamic partition adds a second reordering-induced mismatch (§2). At position t , the detector re-estimates each median from continuations conditioned on the reordered prefix. Even a sentence that remains in place can therefore be scored against a median different from the one used during marking. This suggests that PMark’s robustness to in-place rewording may not transfer to whole-text transformations. Section 6.3 tests this prediction. Token watermarks are less exposed to reordering: adjacent tokens are rarely independent, and reordering them typically does not preserve meaning.

Resegmentation. Resegmentation changes the units that the detector embeds because the detector segments the suspect text itself (§2). Merging s_t with s_{t+1} produces a unit m whose region $r(m)$ need not match $r(s_t)$ or $r(s_{t+1})$. Splitting s_t into fragments f_1 and f_2 produces regions $r(f_1)$ and $r(f_2)$, neither of which needs to match $r(s_t)$. Under a context-dependent scheme, a split disturbs three region tests. The detector evaluates f_1 and f_2 against the valid sets at their new positions, then tests the following sentence against a set keyed to $r(f_2)$ rather than $r(s_t)$. A merge can disturb two region tests. The detector tests the merged unit, then keys the following sentence’s valid set to $r(m)$ rather than $r(s_{t+1})$. Merging also lowers the unit count n that scales the z -score of Equation (1). Token watermarks have no sentence-like units to split or merge.

Controlled edits. We isolate each edit type’s effect on the detection score with a separate controlled edit. Each controlled edit changes one of wording, sentence order, or sentence boundaries while inducing negligible change in the other two. *Reorder* permutes sentence positions and leaves every sentence’s text unchanged. *Split* cuts sentences at an interior point, and *merge* joins adjacent pairs, respectively increasing and decreasing the unit count n . We evaluate both with one direct resegmentation measure. *Synonym substitution* replaces words in place, leaving sentence order and boundaries intact, and supplies the rewording control. We report only detection for these controlled edits, since they are not usable attacks (§6).

Embedding displacement. *Embedding displacement* is the distance between the embedding selected for a detector position during marking and the embedding tested at that position after editing. We use greater displacement as a no-box surrogate for a greater chance of crossing a watermark-region boundary. Rewording shifts the unit’s embedding while retaining its position. Reordering moves a different unit’s embedding into the position. Resegmentation redistributes the text across detector units, replacing the embeddings attached to one or more positions. Reordering and resegmentation can also change the preceding context that determines the valid set or channel boundary. Without using the provider’s encoder or

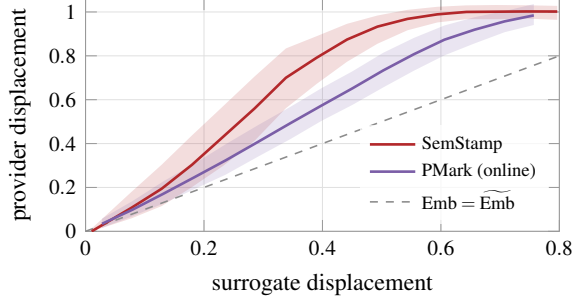


Figure 2: Provider-encoder displacement conditioned on surrogate displacement for sentence pairs from 1 024 documents per scheme. Lines mark medians; bands span the middle 50%.

Algorithm 1 Positional EDA (EDA-P).

Require: marked text $x = (s_1, \dots, s_n)$; paraphraser π ; surrogate encoder $\widetilde{\text{Emb}}$; budget K
Ensure: attacked text $y = (s'_1, \dots, s'_{n'})$

- 1: **for** $t = 1, 2, \dots$ **until** π emits EOS or y reaches 512 tokens **do**
- 2: $\mathcal{C} = \{c_1, \dots, c_K\} \ni c_i \sim \pi(\cdot \mid x, s'_1, \dots, s'_{t-1}, t)$
- 3: $s'_t \leftarrow \arg \max_{c \in \mathcal{C}} d(\widetilde{\text{Emb}}(c), \widetilde{\text{Emb}}(s_{\min(t, n)}))$
- 4: **end for**
- 5: **return** $y = (s'_1, \dots, s'_{n'})$

secret key, our no-box attacker maximizes this distance under a surrogate encoder. The attack assumes that displacement under $\widetilde{\text{Emb}}$ transfers to displacement under the provider’s encoder. Figure 2 tests this assumption for SemStamp and PMark, using a surrogate encoder with a different architecture from both providers.

Embedding displacement attack. Algorithm 1 gives EDA-P. It reverses the watermark’s sampling objective: the provider searches N sampled candidates for a sentence in a valid region (§2), whereas the attacker searches K paraphrases for the candidate farthest from the source sentence at that position. We use cosine distance $d(u, v) = 1 - \langle u, v \rangle / (\|u\| \|v\|)$.

The attack generates output positions $t = 1, 2, \dots$ until the paraphraser emits the end-of-sequence marker EOS or the rewrite reaches 512 tokens (line 1), allowing the rewrite to differ in length from x . At each position, the attack samples K candidate rewrites from an instruction-tuned paraphraser π (line 2). The paraphraser conditions on the full marked text x and the previously generated rewrites $s'_1, \dots, s'_{t-1}$. The *anchor* is source sentence s_t when $t \leq n$ and the final source sentence s_n otherwise. The *surrogate encoder* $\widetilde{\text{Emb}}$ substitutes for the provider’s encoder. The attack selects the candidate farthest from its anchor under $\widetilde{\text{Emb}}$ (line 3). Anchoring output position t on source sentence s_t targets a detector that keys each position separately, as (k)-SemStamp schemes and PMark do (§2). We call this positional configuration EDA-P when

comparing anchor rules.

Conditional provider-encoder displacement rises with surrogate displacement for both schemes. This pattern supports surrogate selection for SemStamp’s contrastively tuned encoder and PMark’s off-the-shelf encoder (§2). At position t , a candidate conditioned on the full marked text and the rewrite so far need not reword s_t in place. The candidate can split s_t , merge it with the following sentence, or advance content from a later sentence and defer the remainder. The displacement objective admits these structural changes without a separate edit operation. One attack can therefore combine them with the in-place rewording that every candidate applies.

EDA-P requires no key, partition geometry (hyperplanes, centroids, or PMark channels), or provider encoder, so it runs against every EBW we evaluate.

5 Robustness Mechanisms for EBWs

The attacks of §4 expose three choices that determine a mark’s sensitivity to whole-text edits: how far the selected candidate lies inside a valid region, how the valid set changes with context, and which units the provider marks and the detector scores. Our design changes all three: best-of- N prefers candidates farther inside a valid region, a fixed valid set removes context dependence, and sub-sentence semantic units called semspans replace sentences. Best-of- N and the fixed valid set leave P unchanged, while semspan configurations partition semspan embeddings rather than sentence embeddings. Because each modification operates only through region indices and boundary-margin scores, the full design applies to SemStamp’s hyperplanes and k-SemStamp’s centroids, yielding SwordStamp and k-SwordStamp, respectively.

Throughout, the provider and detector share a deterministic segmenter Seg , and $\text{Seg}(y) = (u_1, \dots, u_n)$ denotes the units it recovers from text y . The (k)-SemStamp schemes use Seg to recover sentences. Because every modification is public (§3), we state the attacker’s response beside each construction and combine those responses into the EDA-S of §5.4.

5.1 Best-of- N Selection

An in-place rewording removes a hit only by crossing a region boundary (§4). The *signed boundary depth* $d_\delta(u)$ is the watermark’s boundary-margin score for unit u . It is positive when $r(u)$ lies in the active valid set and negative when $r(u)$ lies in the blocked set. Its magnitude is the minimum absolute hyperplane similarity for LSH and the gap between the nearest and next-nearest centroid distances for k -means. Rejection sampling accepts candidates with $d_\delta(u) \geq \delta$ under the rule in §2. Without the key, an attacker cannot target G or B , so a content-preserving rewrite may move its embedding across a boundary in either direction (Figure 3). A deep valid selection gives the rewrite more room before leaving G ; a shallow blocked fallback gives the rewrite a chance to enter G .

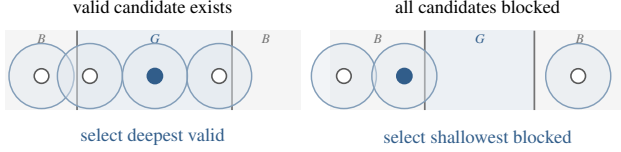


Figure 3: Disks illustrate displacement after a rewrite. Best-of- N selects the deepest valid candidate if available (left), otherwise the shallowest blocked candidate (right).

Construction. Best-of- N evaluates the full pool C_t of N candidates and emits its deepest member.

$$u_t = \arg \max_{u \in C_t} d_\partial(u) \quad (3)$$

By ranking every valid candidate above every blocked candidate, it returns the deepest valid candidate when C_t contains one, and otherwise the blocked candidate nearest a boundary. The ranking replaces the margin test and maxout’s last-draw fallback. If C_t contains a candidate that rejection sampling would accept, best-of- N emits a valid unit at least as deep as that candidate. In that case, best-of- N weakly dominates rejection sampling’s margin criterion on the same pool.

Best-of- N trades this depth for provider draws. If each draw is independently valid with probability γ and we ignore the margin and cap, rejection sampling evaluates $1/\gamma$ candidates in expectation. Best-of- N evaluates N candidates, or γN times that expectation. The two spend those draws differently: rejection sampling issues them one at a time and stops at the first acceptance, whereas best-of- N fixes the pool in advance and can issue and score it in one batch. Selecting an extreme-depth candidate may affect fidelity; §6.5 quantifies that effect.

Attacker response. Best-of- N exposes no new signal in the marked text, so it requires no design-specific change to EDA beyond sweeping the paraphrase budget K (§6.1).

5.2 Fixed Valid Sets

Context-dependent valid sets have the reordering weakness in §4; a fixed set removes predecessor dependence.

Construction. A *fixed valid set* is one key-dependent subset of regions that replaces every per-position G_t . It transfers Unigram’s context-independent split [48] from vocabulary items to regions. At Setup, the provider derives $G \subseteq \{1, \dots, c\}$ from k with $|G| = \gamma c$, sets $B = \{1, \dots, c\} \setminus G$, and uses G at every generation and detection position. The detector then counts recovered units in G .

$$H_G(y) = \sum_{u \in \text{Seg}(y)} \mathbf{1}[r(u) \in G] \quad (4)$$

The detector inserts $H_G(y)$ and $n = |\text{Seg}(y)|$ into Equation (1) and calibrates $\tau(\rho)$ on unmarked text as in (k)-SemStamp.

Because Equation (4) does not depend on unit position, a permutation changes neither $H_G(y)$ nor n , and moving one

unit cannot re-key another test. An in-place rewrite loses a hit only when its embedding crosses a region boundary into B . Changing unit boundaries avoids that condition: a merged or split unit need not retain the original units’ valid-set membership, and merging reduces n in Equation (1).

Null inflation. A fixed valid set can also inflate an unmarked passage’s score. Topically concentrated passages place many units in few regions, so a passage whose occupied regions fall disproportionately in G can exceed the null hit rate γ without carrying a watermark. Calibrating $\tau(\rho)$ on unmarked text preserves the target FPR ρ , but can lower detection on unattacked marked text. §6.5 reports the calibrated threshold for every configuration. By contrast, SAMark’s self-anchored detector infers key bits from the suspect text (§2), so any concentrated passage inflates its null.

Diversity ranking. Using one G at every position lets consecutive valid units share a region, risking repetition. The *diversity ranking* classifies a candidate as green when its region lies in G and differs from the predecessor’s region, yellow when it shares the predecessor’s valid region, and red when its region lies in B . Writing $\text{color}(u) \in \{2, 1, 0\}$ for green, yellow, and red, best-of- N lexicographically maximizes $(\text{color}(u), d_\partial(u))$. Color decides first, and depth breaks ties within a color. The diversity ranking does not affect detection: yellow and green units are counted as hits, preserving the order robustness of Equation (4).

Attacker response. EDA-P measures distance from the positional anchor s_t (§4), but an order-robust detector uses no position in its hit count. The bag displacement attack scores candidate u against all source units in x .

$$\Delta_{\text{bag}}(u, x) = \min_{a \in \text{Seg}(x)} d(\widetilde{\text{Emb}}(u), \widetilde{\text{Emb}}(a)) \quad (5)$$

The minimum grows only when u is far from every source unit, so copying a source unit at another position earns no credit. At each output step, the attacker emits the candidate that maximizes $\Delta_{\text{bag}}(u, x)$, retaining the budget K , surrogate encoder $\widetilde{\text{Emb}}$, and whole-text generation of Algorithm 1. The attacker can raise the minimum by moving a unit in place, which trades against $\bar{q}$, or by merging or splitting source units.

Secrecy of the valid set. Fixing G reintroduces the fixed-list structure that Hou et al. [17] identify as Unigram’s weakness. However, a no-box attacker cannot recover G . Doing so requires a corpus of marked texts collected across many provider queries, a black-box capability that the threat model excludes (§3). An attacker that pays this cost can also steal and scrub a context-dependent partitioning, as demonstrated for token watermarks [23, 33]. Costs differ by a constant factor: a fixed valid set is a histogram over c regions, whereas a context-dependent rule is a $c \times c$ transition matrix. At $c = 8$, this constant-factor difference does not create a distinct security boundary [17, 18].

Additionally, a recovered token green list identifies individual tokens that the attacker can replace. A recovered G instead

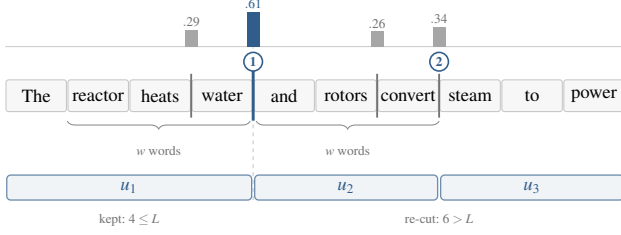


Figure 4: Seg segmentation ($w = 3$, $L = 5$). Ticks mark valid cuts; bars give window distances. Seg cuts at the largest distance (1), then recursively cuts the six-word right semspan (2).

reveals the valid embedding regions, and its complement B reveals the blocked regions. To remove a hit on unit u , the attacker must either discard u , rewrite it into B , or resegment its content into units in B . Discarding u removes content. An in-place rewrite challenges the EBW robustness premise: it must cross a region boundary while the resulting passage y still satisfies $Q(x, y) \geq \bar{q}$. Thus, B identifies the target region but does not produce a rewrite that reaches it while preserving content. The next subsection addresses resegmentation.

5.3 Semspans

Fixed valid sets make detection order-robust, but they do not remove segmentation sensitivity: $H_G(y)$ stays unchanged only while $\text{Seg}(y)$ recovers the same units. Sentence splitting and merging change those units without necessarily changing passage content (§4). We therefore mark semspans that an encoder-guided segmenter delimits. Figure 4 illustrates the recursive cut rule on one sentence.

Construction. A semspan is a contiguous word sequence within one sentence, delimited at a cut where the encoder observes the largest local embedding change. The segmenter Seg splits a text into sentences, then recursively cuts each sentence longer than L words. A cut point is *admissible* when at least w words lie on either side and the word to its left is not a function word. The window condition supplies w words for each side of the embedding comparison. The function-word condition keeps a determiner or preposition with the phrase that follows it. At each admissible point, Seg embeds the w words on either side and scores their distance. It selects the highest-scoring point and recurses on both sides until no semspan exceeds L . In the example, the first cut lands at the clause boundary, and recursion re-cuts only the resulting semspan that still exceeds L .

This segmenter has three effects. First, finer units increase the unit count n . Writing the observed hit rate as $\hat{p} = H/n$ turns Equation (1) into $z = \sqrt{n}(\hat{p} - \gamma)/\sqrt{\gamma(1-\gamma)}$, so at a fixed hit rate above γ the score grows with $\sqrt{n}$. Second, shorter units offer fewer interior points at which an attacker can split. Third, a merge can become recoverable: because Seg cuts on embedding change rather than punctuation, joining two

Algorithm 2 Scheme-specific EDA (EDA-S).

Require: marked text x ; paraphraser π ; surrogate encoder $\widetilde{\text{Emb}}$; budget K ; surrogate-bound segmenter Seg; valid-set rule

Ensure: attacked text $y = (u'_1, \dots, u'_{n'})$

```

1:  $(a_1, \dots, a_n) \leftarrow \text{Seg}(x)$ 
2: for  $t = 1, 2, \dots$  until  $\pi$  emits EOS or  $y$  reaches 512 tokens
   do
3:    $C = \{c_1, \dots, c_K\} \ni c_i \sim \pi(\cdot \mid x, u'_1, \dots, u'_{t-1}, t)$ 
4:   if valid-set rule is fixed then
5:      $A_t \leftarrow \{a_1, \dots, a_n\}$ 
6:   else
7:      $A_t \leftarrow \{a_{\min(t, n)}\}$ 
8:   end if
9:    $u'_t \leftarrow \arg \max_{c \in C} \min_{a \in A_t} d(\widetilde{\text{Emb}}(c), \widetilde{\text{Emb}}(a))$ 
10: end for
11: return  $y = (u'_1, \dots, u'_{n'})$ 
```

marked semspans into one sentence can preserve their original cut, allowing Seg to re-cut near the provider’s original point.

However, finer units require more provider draws from the LLM. It samples candidate sentences as before, but commits only the first semspan of the chosen candidate and regenerates the remainder at the next step, so a passage takes one best-of- N round per semspan rather than one per sentence. Finer units repeat boundary-depth selection more often within each sentence, increasing its potential effect on marked-text fidelity. §6.5 measures the net fidelity change.

Attacker response. Because Seg’s rule is public, the attacker instantiates it with $\widetilde{\text{Emb}}$, applies it to marked text, and anchors on the recovered semspans. Finer units raise the attacker’s cost as they raise the provider’s: K applies per unit, and a passage contains more semspans than sentences.

5.4 Scheme-Specific EDA

The three responses above compose into one attack, EDA-S, which runs EDA with the anchor rule and segmenter of its target rather than the fixed positional anchors of EDA-P. We write each EDA-S configuration as (anchor rule, segmentation unit), with pos for positional anchoring and sent for sentence segmentation. The public segmenter Seg determines the source anchors, and the published valid-set rule determines how many anchors each candidate must move away from.

Algorithm 2 instantiates EDA-S with the same paraphraser π , surrogate encoder $\widetilde{\text{Emb}}$, and budget K as EDA-P. First, it applies the detector’s Seg to marked text and uses the recovered units as anchors $a_1, \dots, a_n$ (line 1). At output step t , it samples K candidates from the paraphraser π (line 3) and emits the candidate farthest from its nearest anchor in the active set A_t (line 9). The watermark’s valid-set rule selects

A_t (line 4), recovering the fixed-set and context-dependent objectives defined above. Either setting of A_t combines with either segmenter, and best-of- N changes neither attack input. The attackers for the context-dependent (k)-SemStamp and PMark use (pos, sent), so EDA-S coincides with EDA-P. The SAMark attacker uses (bag, sent) because its per-query key bit gives no role to source position (§2). The SwordStamp attacker and k-SwordStamp attacker use (bag, span).

Every instantiation remains within the no-box threat model of §3: it uses the public segmenter and valid-set rule, but not a detector oracle or key material. The surrogate encoder $\tilde{\text{Emb}}$ invokes the transfer assumption stated and tested in §4. Because the objective optimizes displacement rather than content preservation, a larger K improves evasion only while π still meets $\bar{q}$. The appendix edit-signature analysis suggests that content-preserving EDA-S evasions combine reordering and resegmentation with rewording (Figure 9).

6 Robustness, Fidelity, and Cost

Our evaluation determines whether each design component resists its target edit type and measures the provider cost of that resistance. We attribute score loss to individual edit types, test removal under no-box and detector-access conditions, and quantify clean detection and generation costs.

6.1 Experimental Setup

Watermarks. We use each watermark’s published configuration. SemStamp and k-SemStamp use $\gamma = 0.25$, 8 regions, and margins of 0.02 and 0.035, respectively. We evaluate online PMark with $b = 4$ channels and SAMark with $b = 2$ channels. Every scheme may draw at most $N = 64$ candidates per unit. Rejection sampling may stop at its first accepted candidate, while best-of- N , SAMark, and PMark use the full pool. For semspan segmentation, we set the maximum unit length to $L = 15$ words and the boundary-comparison window to $w = 5$ words on each side. For each k -means configuration, we recompute the centroids from its sentence or semspan embeddings over a held-out corpus of 8 192 C4 documents [36], following k-SemStamp’s fitting procedure [18].

Models and data. The provider uses Llama-3.1-8B [15]. The provider-side (k)-SemStamp and (k)-SwordStamp configurations use the released C4-tuned AbeHou/SemStamp-c4-sbert checkpoint, while PMark and SAMark use all-mpnet-base-v2 [37]. We generate every configuration from one set of 1024 prompts from C4’s realnewslike configuration [36], stopping at 256 new tokens or, for PMark and SAMark, targeting 12 sentences. A disjoint 1024-document split supplies the empirical null for every configuration except online PMark, after truncation under the scheme’s stopping rule to match the marked-document length distribution. We set the 5% and 1% thresholds to the empirical 0.95 and 0.99 score

quantiles, respectively; online PMark instead uses the nominal standard-normal thresholds 1.6449 and 2.3263.

Attacks. Our sentence-level controls are Pegasus [44] and Parrot [10], each with and without bigram selection. Bigram selection chooses the lowest-overlap candidate whose BERTScore [46] is within 3% of the best candidate. For Dipper [26], we fix lexical diversity at 20 while sweeping order diversity over $\{20, 40, 80\}$, then fix order diversity at 20 while sweeping lexical diversity over $\{20, 40, 80\}$. For no-box EDA, we use Qwen2.5-3B-Instruct [35] and a bge-base-en-v1.5 surrogate encoder [40] rather than the provider encoder. This gives the attacker a smaller generator and a surrogate with a different architecture from every provider encoder. This separation tests whether EDA’s displacement objective transfers without reusing the provider’s generator or encoder. The EDA-S budget grid is $\{1, 2, 4, 8, 16, 32, 64\}$. Additional EDA-P comparisons use available budgets in $\{4, 16, 64\}$, and EDA-D adds $\{8, 32\}$.

Detection and quality. We use ASR@5% as the primary operating point and sweep $\bar{q}$ from 65% through 95% in steps of 5%. We report the clean true-positive rate (TPR), the fraction of unattacked marked documents flagged at the same threshold. At each $\bar{q}$, we select the setting from each attack’s effort grid that maximizes ASR. Qwen3-32B [41] judges fidelity from fluency, coherence, relevance, and informativeness, and content preservation from content recall, detail precision, information injection, and contradiction. Section A.2 specifies both judges, their prompts, and the scoring procedure. We run each judgment with three random seeds and average the scores. Both scores are proxies for human judgments.

Edit measurement. For a controlled edit $\mathcal{A}$ applied to marked documents $x_1, \dots, x_D$, normalized z retention is

$$\mathcal{R}(\mathcal{A}) = \frac{\sum_{i=1}^D z_k(\mathcal{A}(x_i))}{\sum_{i=1}^D z_k(x_i)}, \quad (6)$$

where D is the number of documents and k is the detector key. We compute paired-bootstrap intervals over documents. For marked text x and attacked text y , let $w_1, \dots, w_m$ be patience anchors, tokens that occur exactly once in each text, ordered by their positions in x [16]. Anchor w_j has source and attacked token positions (p_j, p'_j) and sentence indices (u_j, u'_j) under NLTK’s pretrained English Punkt sentence tokenizer [30]. Let $\mathcal{P} = \{(i, j) : i < j, u_i \neq u_j\}$ be the cross-sentence anchor pairs. We define e_{merge} as the fraction of adjacent source-order anchors separated in x but joined in y , and e_{split} as the fraction of adjacent attacked-order anchors joined in x but separated in y . We define the resulting edit rates as follows.

$$\begin{aligned} r_{\text{word}} &= 1 - F_1^{\text{ROUGE-1}}(x, y), \\ r_{\text{order}} &= \frac{1}{|\mathcal{P}|} \sum_{(i,j) \in \mathcal{P}} \mathbf{1}[(p_i - p_j)(p'_i - p'_j) < 0], \\ r_{\text{seg}} &= 1 - \frac{2(1 - e_{\text{merge}})(1 - e_{\text{split}})}{(1 - e_{\text{merge}}) + (1 - e_{\text{split}})}. \end{aligned} \quad (7)$$

Table 2: Clean detection, generation cost, output quality, and robustness at $N = 64$. EDA-S configurations list (anchor rule, unit): pos and bag denote anchor rules, and sent and span denote sentence and semspan units. Swords mark final designs. At $\bar{q} = 90\%$, ASR cells report the strongest judged setting per attack family at 1% and 5% FPR. PPL, Sem. entropy, Rep-3, and Sent. dup. denote corpus perplexity, semantic entropy, repeated-trigram rate, and sentence duplication. Rates are percentages; fidelity reports mean [95% CI], and Draws/sentence reports mean candidate draws per clean sentence. Bold and underlined values mark the overall best and worst means.

Configuration	EDA-S config.	@1% / 5%		ASR ↓ (%) by no-box attacker, @1% / 5%			Fidelity ↑ [95% CI]	Draws/ sent. ↓	PPL ↓	Sem. entropy ↑	Rep-3 ↓	Sent. dup. ↓
		TPR ↑	τ	Sentence	Dipper	EDA-S						
SAMark	(bag, sent)	<u>45.7 / 86.2</u>	13.53 / 7.02	<u>44.5 / 15.6</u>	45.0 / 18.7	<u>61.3 / 32.6</u>	81.5 [80.2,82.8]	64.0	<u>6.92</u>	2.40	2.5	0.3
PMark (online)	(pos, sent)	95.6 / 97.2	2.33 / 1.64	5.2 / 2.8	40.3 / 26.9	57.8 / 46.1	86.0 [84.8,87.2]	64.0	6.26	2.54	3.6	0.6
SemStamp	(pos, sent)	84.7 / 93.1	2.84 / 2.12	19.1 / 11.9	<u>46.7 / 39.7</u>	<u>57.7 / 47.9</u>	80.4 [78.9,81.9]	8.8	6.16	2.58	6.4	4.5
+ best-of- N	(pos, sent)	97.5 / 99.3	2.84 / 2.12	8.5 / 4.9	40.4 / 33.4	47.9 / 38.1	79.6 [78.0,81.2]	64.0	6.06	2.48	8.6	6.7
+ fixed set	(bag, sent)	92.4 / 97.4	3.29 / 2.26	7.6 / 3.6	11.6 / 6.0	33.1 / 19.8	<u>72.7 [70.9,74.6]</u>	64.0	5.35	2.48	<u>15.7</u>	<u>12.4</u>
+ diversity ranking	(bag, sent)	91.1 / 96.5	3.29 / 2.26	10.4 / 4.8	14.6 / 7.9	37.2 / 24.1	<u>75.5 [73.8,77.3]</u>	64.0	5.91	2.44	10.6	8.3
+ semspans $\times$	(bag, span)	90.6 / 94.9	3.61 / 2.49	9.8 / 5.2	12.7 / 6.5	30.2 / 18.7	78.6 [77.1,80.1]	103.2	6.62	3.27	6.6	3.7
k-SemStamp	(pos, sent)	92.9 / 97.1	2.84 / 1.76	10.4 / 5.2	39.4 / 30.3	48.4 / 33.3	80.6 [79.1,82.1]	4.9	6.25	2.45	5.2	2.9
+ best-of- N	(pos, sent)	96.3 / 98.5	2.84 / 1.76	4.9 / 2.1	31.2 / 21.9	32.6 / 20.0	75.2 [73.5,76.8]	64.0	5.86	2.42	9.4	5.9
+ fixed set	(bag, sent)	89.6 / 94.5	4.02 / 2.84	9.2 / 4.2	12.4 / 6.1	23.8 / 12.1	76.0 [74.5,77.5]	64.0	5.61	<u>2.27</u>	10.5	5.8
+ diversity ranking	(bag, sent)	86.6 / 92.5	4.02 / 2.84	14.3 / 7.8	21.7 / 10.9	39.7 / 22.6	78.7 [77.3,80.2]	64.0	6.27	2.32	6.1	3.4
+ semspans $\times$	(bag, span)	93.6 / 97.0	3.58 / 2.40	6.3 / 3.3	8.7 / 4.5	20.6 / 10.8	76.2 [74.7,77.8]	<u>107.4</u>	6.75	3.25	7.0	5.0

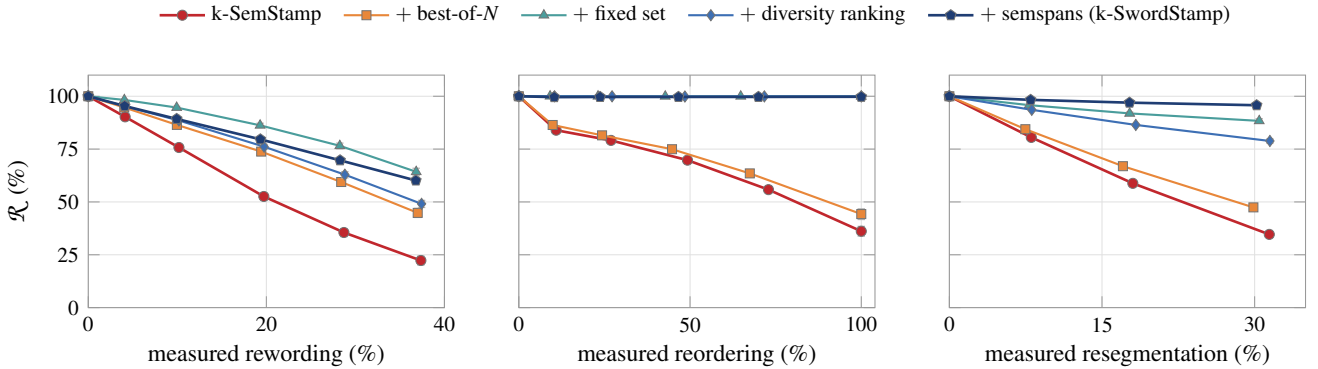


Figure 5: Normalized detection-score retention $\mathcal{R}$ versus realized edit strength, with bootstrap 95% intervals. Resegmentation pools sentence splits and merges using boundary disagreement. Lines follow the five stages from k-SemStamp to k-SwordStamp.

The rates are 1—ROUGE-1 F1 for rewording [29], a restricted Kendall inversion rate for reordering [24], and 1— boundary F1 for resegmentation [4, 34].

Component sequence. Table 2 lists the five-stage LSH and k -means sequences from (k)-SemStamp to (k)-SwordStamp and reports each stage’s EDA-S configuration as (anchor rule, segmentation unit). Because each attacker uses the public anchor rule and segmenter at that stage, each measured change accounts for an adversary that knows the added component. The diversity ranking targets no edit type, so we measure its effects on retention, fidelity, and repetition rather than evaluate a designated defense.

6.2 Score Loss by Edit Type

Figure 5 reports normalized score retention, $\mathcal{R}$, against the measured edit rates of in-place synonym substitution, sentence reordering, and resegmentation. Table 3 reports that each edit

changes its target rate while leaving the other two near zero. k-SemStamp loses retention under all three controlled edits, including structural edits that preserve sentence text or claims. This loss supports the mechanism in §4: the detector rebuilds state from attacker-controlled structure.

Best-of- N raises retention under rewording, but it does not eliminate rewording-induced score loss. The fixed valid set removes predecessor re-keying and eliminates reordering-induced score loss. The diversity ranking lowers rewording and resegmentation retention relative to the fixed-set stage, but improves fidelity and reduces repetition in both component sequences (Table 2). Adding semspans then raises resegmentation retention above every sentence-based stage.

6.3 No-Box Robustness

Figure 6 reports the strongest tested no-box attack family for each scheme and the effect of configuring EDA with the

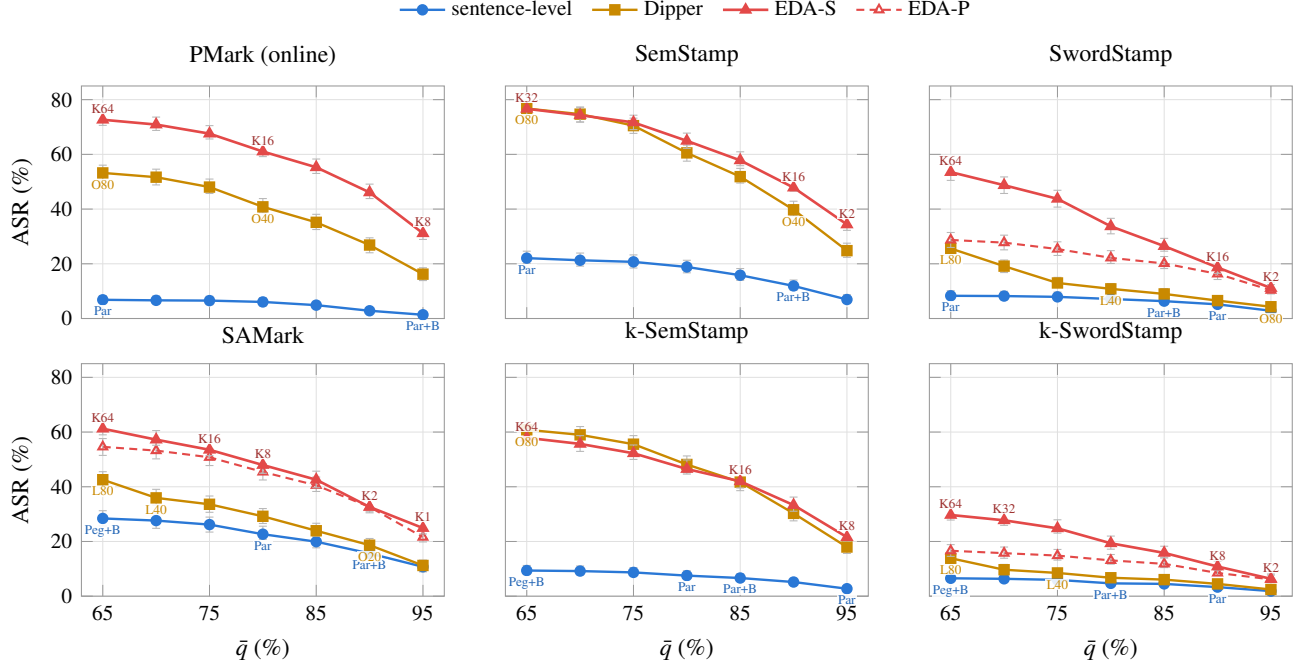


Figure 6: No-box ASR at 5% FPR versus $\bar{q}$, with paired-document bootstrap 95% intervals. Each curve selects the strongest setting per attack family and $\bar{q}$. Labels identify sentence-level controls (Peg, Par, +B), Dipper settings (O, L), and EDA budgets (K). Dashed curves show EDA-P at $K \in \{4, 16, 64\}$.

scheme’s public anchor rule and segmenter. Each panel traces the strongest setting for sentence-level controls, Dipper, and EDA-S across $\bar{q}$ under the no-box threat model. For SAMark, SwordStamp, and k-SwordStamp, dashed EDA-P curves compare positional sentence anchors with bag anchoring.

At $\bar{q} = 90\%$, EDA-S ASR falls from 47.9% to 18.7% on LSH and from 33.3% to 10.8% on k -means. At the same $\bar{q}$, EDA-S’s ASR estimate is 3.0 percentage points higher than Dipper’s on k-SemStamp and is higher on every other scheme in Figure 6. The SwordStamp and k-SwordStamp attackers replace EDA-P’s positional sentence anchors with bag semspan anchors. Using bag semspan anchors raises ASR from 16.3% to 18.7% on SwordStamp and from 8.5% to 10.8% on k-SwordStamp. The SAMark attacker replaces only positional anchors with bag anchors and continues to use sentence units. Its 32.6% ASR effectively coincides with EDA-P’s 32.6% at this $\bar{q}$, although the EDA-S attacker leads at every other $\bar{q}$.

6.4 Robustness under Detector Access

Detector-access EDA (EDA-D) augments EDA-P with the provider’s exact candidate-level region test, including its key, encoder, partition, and segmenter. It retains EDA-P’s paraphraser and per-unit search budget, and it does not receive the provider’s generator or quality judge. We evaluate this attacker on the k -means pair, k-SemStamp and k-SwordStamp.

For these schemes, let $h_k(c \mid p, t) \in \{0, 1\}$ denote the detector’s decision for candidate c at position t under committed predecessor p , with 1 for a hit and 0 for a miss. At each output step, EDA-D selects the detector miss closest to its source anchor when one exists; otherwise, it selects the most displaced candidate as in EDA-P. Algorithm 3 specifies the procedure. Figure 7 reports raw evasion, ASR@5% at $\bar{q} = 90\%$, and the proportion of detector-evading outputs that meet $\bar{q}$ as the detector-access search budget increases for the k -means schemes.

At $K = 4$, raw evasion is 70.2% on k-SemStamp and 29.6% on k-SwordStamp. More search raises raw evasion against both schemes, but the selected rewrites against k-SwordStamp receive progressively lower content-preservation scores. After quality gating, ASR is 19.4%–39.7% on k-SwordStamp, compared with 52.2%–65.5% on k-SemStamp.

6.5 Clean Performance and Costs

At 5% FPR, clean TPR rises from 93.1% to 94.9% on LSH and remains effectively unchanged from 97.1% to 97.0% on k -means (Table 2). Fidelity falls from 80.4% to 78.6% on LSH and from 80.6% to 76.2% on k -means.

PMark has the highest measured fidelity and is the strongest scheme against the sentence-level controls, consistent with its reported performance [20]. Against the whole-text Dipper and EDA-S, both (k)-SwordStamp configurations yield lower

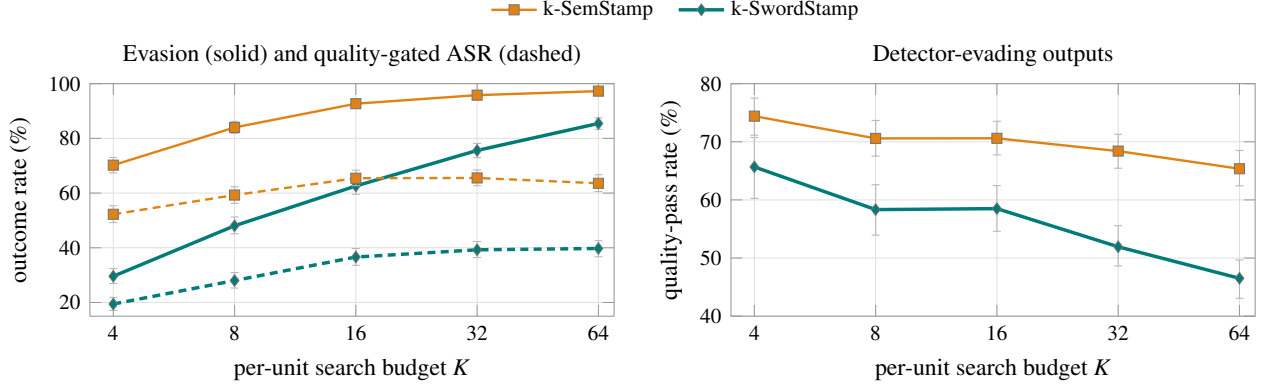


Figure 7: Detector-access EDA (EDA-D) against k-SemStamp and k-SwordStamp at 5% FPR. Left: raw evasion (solid) and ASR at $\bar{q} = 90\%$ (dashed); right: fraction of evasions meeting the same $\bar{q}$. Bars give paired-document bootstrap 95% intervals.

Algorithm 3 Detector-access EDA (EDA-D).

Require: marked text x ; paraphraser π ; provider detector $\text{Detect}(k, \cdot)$ with encoder Emb , partition P , and segmenter Seg ; budget K

Ensure: attacked text $y = (u'_1, \dots, u'_{n'})$

```

1:  $(a_1, \dots, a_n) \leftarrow \text{Seg}(x); u'_0 \leftarrow \emptyset$ 
2: for  $t = 1, 2, \dots$  until  $\pi$  emits EOS or  $y$  reaches 512 tokens
   do
3:    $\mathcal{C} = \{c_1, \dots, c_K\} \ni c_i \sim \pi(\cdot \mid x, u'_1, \dots, u'_{t-1}, t)$ 
4:    $a \leftarrow a_{\min(t, n)}$ 
5:    $\mathcal{C}_{\text{miss}} \leftarrow \{c \in \mathcal{C} : h_k(c \mid u'_{t-1}, t) = 0\}$ 
6:   if  $\mathcal{C}_{\text{miss}} \neq \emptyset$  then
7:      $u'_t \leftarrow \arg \min_{c \in \mathcal{C}_{\text{miss}}} d(\text{Emb}(c), \text{Emb}(a))$ 
8:   else
9:      $u'_t \leftarrow \arg \max_{c \in \mathcal{C}} d(\text{Emb}(c), \text{Emb}(a))$ 
10:  end if
11: end for
12: return  $y = (u'_1, \dots, u'_{n'})$ 

```

ASR than PMark. k-SwordStamp yields lower attack success than SwordStamp, while SwordStamp retains higher fidelity.

The component sequence attributes the robustness and fidelity trade-offs. Best-of- N raises clean TPR under both partition methods and reduces ASR for every no-box attack, at a larger fidelity cost on k -means. The fixed valid set reduces ASR against Dipper and EDA-S, but it requires higher detection thresholds under both partition methods.

Among the four comparison schemes, SAMark has the lowest EDA-S ASR, 32.6% at 5% FPR, but also the lowest clean TPR at both reported operating points. Its self-anchored score infers each key bit from the suspect text (§2), so a topically concentrated passage can receive a high score whether or not it carries a mark. At 5% FPR, calibration sets the threshold to 7.02, and SAMark flags 86.2% of unattacked marked documents. At 1% FPR, its clean TPR falls to 45.7%. For

comparison, SwordStamp pairs 94.9% clean TPR with 18.7% EDA-S ASR, and k-SwordStamp pairs 97.0% with 10.8%.

Semspans lower EDA-S success under both partition methods. Overall, the fixed valid set and semspans supply most robustness; the diversity ranking trades some robustness for fidelity and lower repetition.

7 Limitations and Future Work

Generation cost. Best-of- N evaluates all $N = 64$ candidates, whereas rejection sampling stops after the first accepted candidate and uses fewer draws on average in our evaluation. The fixed pool can be generated and scored in one batch, while rejection sampling makes acceptance decisions sequentially. Semspan generation adds sequential rounds because the provider commits one semspan at a time. The final LSH and k -means designs respectively average 1.6 and 1.7 64-candidate rounds per output sentence (Table 2).

Evaluation scope. Our evaluation covers English news continuations up to 256 tokens from one provider model. It does not establish whether the same attacks and defenses transfer to code generation, question answering, instruction following, other languages, longer documents, or other model families. These settings may require different segmentation rules and definitions of content preservation. Within our setting, we test two provider encoders but only one EDA paraphraser and one surrogate encoder. We also rely on model-based fidelity and content-preservation scores rather than human judgments.

Future work. Neither EDA nor EDA-D uses a quality oracle during candidate selection. A quality-aware attacker could reject bad rewrites and achieve a different ASR-content-preservation tradeoff. A provider could also use task-specific quality scores during candidate selection to recover part of the measured fidelity loss. On the provider side, since the valid set is fixed across positions, the watermark is a static signal and can be learned directly into the generator. The provider could therefore fine-tune its generator to generate

watermarked samples from a smaller candidate pool.

8 Related Work

Watermarking versus post-hoc detection. Post-hoc detection is the main alternative to proactive watermarking. Its detectors come as zero-shot statistical tests, such as the probability-curvature test of Mitchell et al. [32], and as trained classifiers. Neither requires generator cooperation, but both rely on empirical error estimates and rarely reach the low FPRs needed for deployment [49]. Watermarking instead uses a keyed signal during generation to control FPR [49].

Watermark families. Token-level watermarks embed the signal in the token distribution and therefore tie it to specific token choices, which paraphrasing can erase [1, 8, 11, 25, 27, 48]; Zhao et al. [49] survey that family in full. Beyond the EBWs we evaluate, other schemes vary how the per-sentence signal is defined and selected, through similarity intervals [9], embedding rings [38], structure and cohesion partitions [45, 47], and sparse-autoencoder features [42]. SAMark [21] is concurrent work that uses per-query key bits and self-anchoring to make detection order-robust; we report its detection rate, calibrated threshold, and robustness alongside every other configuration in §6.

Candidate selection and distortion. Selecting among sampled candidates to strengthen a mark predates our use of it. WaterMax [14] and the black-box scheme of Bahri et al. [3] score chunks of generated text and keep the most detectable one, trading generation cost for detection strength at the token level. Our best-of- N selection makes the same trade at the sentence or semspan level, scoring each candidate by how far inside its valid region it falls (§5.1). Such selection distorts the generator’s output distribution; distortion-free schemes avoid this effect by construction [8, 19, 27], and PMark carries that property into the EBW family [20].

Attacks and robustness evaluations. Prior evaluations of text watermarks center on paraphrasing. Krishna et al. [26] show that a trained whole-text paraphraser evades token-level detectors unless the provider retains every generation for retrieval-based detection. Hou et al. [17, 18] report that (k-)SemStamp schemes resist sentence-by-sentence paraphraser, and name whole-text paraphrase as the boundary of that result. We test whole-text paraphrasing while holding the attacker’s capabilities, $\bar{q}$, and each detector’s operating point fixed across the family, and we scale attacker effort instead of fixing one paraphraser (§6). Diaa et al. [12] tune paraphraser against a known token-level watermark, which is the closest prior attacker to ours in capability. Other attacks assume access we deny: watermark stealing and design probing collect many keyed samples [23, 33], while DE-MARK queries the watermarked generator to recover and remove n -gram watermarks [7]. Our attacker sees only the target text (§3).

Impossibility results. Prior impossibility results address post-hoc detection and watermarking under different assump-

tions. Sadasivan et al. [39] and Chakraborty et al. [5] debate whether post-hoc detection survives as model and human text converge. Zhang et al. [43] study an attacker with a quality oracle that checks candidate outputs and a perturbation oracle that proposes edits with a nontrivial chance of preserving quality. They prove that the resulting random walk removes the watermark when it mixes rapidly over high-quality outputs. Our no-box attacker has neither oracle, uses a generator weaker than the provider’s, and has a finite search budget, so our results cover only the attackers and budgets we test, not Zhang et al.’s stronger setting.

9 Conclusion

We measure how rewording, reordering, and resegmentation affect encoder-based watermark detection. We consolidate them into EDA, a single adaptive no-box attack guided by embedding displacement. At 5% FPR, EDA removes the watermark while retaining a 90% content-preservation score for 32.6%–47.9% of outputs from SemStamp, k-SemStamp, PMark, and SAMark.

These attacks motivate reducing detector dependence on attacker-controlled sentence order and boundaries. We therefore develop k-SwordStamp, which combines best-of- N selection, order-robust detection, and deterministic sub-sentence units. At the same operating point, k-SwordStamp lowers the attack-success rate of EDA configured for each scheme from 33.3% on k-SemStamp to 10.8% at a 4.4-percentage-point fidelity cost. Within our tested models, workload, attacks, and budgets, reducing dependence on attacker-controlled text structure improves resistance to content-preserving watermark removal.

References

- [1] Scott Aaronson and Hendrik Kirchner. Watermarking GPT outputs, 2023. Technical report, OpenAI.
- [2] Anthropic. How Claude’s text watermark works. <https://www.anthropic.com/news/claude-text-watermark>, August 2026. Accessed: 2026-08-20.
- [3] Dara Bahri, John Wieting, Dana Alon, and Donald Metzler. A watermark for black-box language models. *arXiv preprint arXiv:2410.02099*, 2024.
- [4] Doug Beeferman, Adam Berger, and John Lafferty. Statistical models for text segmentation. *Machine Learning*, 34(1):177–210, 1999.
- [5] Souradip Chakraborty, Amrit Singh Bedi, Sicheng Zhu, Bang An, Dinesh Manocha, and Furong Huang. On the possibilities of AI-generated text detection. *arXiv preprint arXiv:2304.04736*, 2023.

- [6] Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 380–388, 2002.
- [7] Ruibo Chen, Yihan Wu, Junfeng Guo, and Heng Huang. De-mark: Watermark removal in large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 9316–9333, 2025.
- [8] Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In *Conference on Learning Theory (COLT)*, pages 1125–1139. PMLR, 2024.
- [9] Amirhossein Dabiriaghdam and Lele Wang. SimMark: A robust sentence-level similarity-based watermarking algorithm for large language models. *arXiv preprint arXiv:2502.02787*, 2025.
- [10] Prithviraj Damodaran. Parrot: Paraphrase generation for NLU. https://github.com/PrithivirajDamodaran/Parrot_Paraphraser, 2021.
- [11] Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823, 2024.
- [12] Abdulrahman Diaa, Toluwani Aremu, and Nils Lukas. Optimizing adaptive attacks against watermarks for language models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.
- [13] European Commission. Code of practice on transparency of AI-generated content. <https://digital-strategy.ec.europa.eu/en/policies/code-practice-ai-generated-content>, 2026. Accessed: 2026-08-20.
- [14] Eva Giboulot and Teddy Furon. WaterMax: Breaking the LLM watermark detectability-robustness-quality trade-off. *Advances in Neural Information Processing Systems*, 37:18848–18881, 2024.
- [15] Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- [16] Paul Heckel. A technique for isolating differences between files. *Communications of the ACM*, 21(4):264–268, 1978.
- [17] Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4067–4082, 2024.
- [18] Abe Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1706–1715, 2024.
- [19] Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. *arXiv preprint arXiv:2310.10669*, 2023.
- [20] Jiahao Huo, Shuliang Liu, Bin Wang, Junyan Zhang, Yibo Yan, Aiwei Liu, Xuming Hu, and Mingxun Zhou. PMark: Towards robust and distortion-free semantic-level watermarking with channel constraints. In *International Conference on Learning Representations (ICLR)*, 2026.
- [21] Jiahao Huo, Wenjie Qu, Yibo Yan, Kening Zheng, Jiaheng Zhang, Xuming Hu, Philip S Yu, and Mingxun Zhou. SAMark: A self-anchored text watermarking with paragraph-level paraphrase robustness. *arXiv preprint arXiv:2605.25796*, 2026.
- [22] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.
- [23] Nikola Jovanović, Robin Staab, and Martin Vechev. Watermark stealing in large language models. In *International Conference on Machine Learning (ICML)*, 2024.
- [24] Maurice G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1–2):81–93, 1938.
- [25] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR, 2023.
- [26] 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.

- [27] Rohith Kudipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.
- [28] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PageAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023.
- [29] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *ACL Workshop on Text Summarization Branches Out*, 2004.
- [30] Edward Loper and Steven Bird. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70, 2002.
- [31] Nils Lukas, Abdulrahman Diaa, Lucas Fenaux, and Florian Kerschbaum. Leveraging optimization for adaptive attacks on image watermarks. In *Proceedings of the 12th International Conference on Learning Representations*, 2024.
- [32] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning (ICML)*, 2023.
- [33] Qi Pang, Shengyuan Hu, Wenting Zheng, and Virginia Smith. No free lunch in LLM watermarking: Trade-offs in watermarking design choices. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.
- [34] Lev Pevzner and Marti A. Hearst. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):19–36, 2002.
- [35] Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- [36] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.
- [37] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992, 2019.
- [38] Jie Ren, Han Xu, Yiding Liu, Yingqian Cui, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. A robust semantics-based watermark for large language model against paraphrasing. *arXiv preprint arXiv:2311.08721*, 2023.
- [39] Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. Can AI-generated text be reliably detected? *arXiv preprint arXiv:2303.11156*, 2023.
- [40] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muenighoff, Defu Lian, and Jian-Yun Nie. C-Pack: Packed resources for general chinese embeddings. *arXiv preprint arXiv:2309.07597*, 2023.
- [41] An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.
- [42] Zhuohao Yu, Xingru Jiang, Weizheng Gu, Yidong Wang, Qingsong Wen, Shikun Zhang, and Wei Ye. SAEMark: Steering personalized multilingual LLM watermarks with sparse autoencoders. In *Advances in Neural Information Processing Systems*, volume 38, 2025.
- [43] Hanlin Zhang, Benjamin L. Edelman, Danilo Francati, Daniele Venturi, Giuseppe Ateniese, and Boaz Barak. Watermarks in the sand: Impossibility of strong watermarking for generative models. In *International Conference on Machine Learning (ICML)*, 2024.
- [44] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning (ICML)*, 2020.
- [45] Junyan Zhang, Shuliang Liu, Aiwei Liu, Yubo Gao, Jungang Li, Xiaojie Gu, and Xuming Hu. CoheMark: A novel sentence-level watermark for enhanced text quality. *arXiv preprint arXiv:2504.17309*, 2025.
- [46] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020.
- [47] Yuehan Zhang, Peizhuo Lv, Yinpeng Liu, Yongqiang Ma, Wei Lu, Xiaofeng Wang, Xiaozhong Liu, and Jiawei Liu. PersonaMark: Personalized LLM watermarking for model protection and user attribution. *arXiv preprint arXiv:2409.09739*, 2024.
- [48] Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for AI-generated text. In *The Twelfth International Conference on Learning Representations*, 2024.
- [49] Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairuze, Andres Fabrega, Nicholas Carlini, Sanjam Garg,

Table 3: Controlled-edit measures for the k-SemStamp base stage. Rows give all measures at intensity η ; the target is bold. Across every stage, off-target values do not exceed 0.039, confirming isolation. Split and merge share the resegmentation measure pooled in Figure 5.

Atom	η	measured edit strength		
		reword	reorder	reseg
controlled_reorder	0.10	0.000	0.109	0.000
	0.25	0.000	0.269	0.000
	0.50	0.000	0.492	0.001
	0.75	0.000	0.729	0.001
	1.00	0.000	1.000	0.002
synonym_substitution	0.10	0.041	0.000	0.002
	0.25	0.102	0.002	0.005
	0.50	0.197	0.007	0.011
	0.75	0.287	0.018	0.019
	1.00	0.374	0.039	0.032
split_midpoint	0.20	0.000	0.000	0.072
	0.50	0.000	0.000	0.163
	1.00	0.000	0.000	0.290
merge_adjacent	0.20	0.000	0.000	0.089
	0.50	0.000	0.000	0.197
	1.00	0.000	0.000	0.339

Sanghyun Hong, Milad Nasr, Florian Tramèr, Somesh Jha, Lei Li, Yu-Xiang Wang, and Dawn Song. SoK: Watermarking for AI-generated content. In *2025 IEEE Symposium on Security and Privacy (SP)*, 2025.

A Supplementary Evaluation

A.1 Controlled-Edit Results

Table 3 shows the controlled edits are near-isolated to what they measure.

A.2 Quality Judges and Criteria

We use separate judges because clean-text fidelity and content preservation under attack are different constructs. The *reference-free fidelity judge* receives the source prompt and the generated continuation, but no reference answer. The *content-preservation judge* receives marked text x as the reference and attacked text y as the candidate. The attacks neither observe nor optimize scores computed on completed outputs.

System-message content. Both system messages assign Qwen3-32B the role of a strict, impartial judge, enumerate the four criteria defined in Table 4, specify a criterion-specific scale, state the scoring rules below, and require the tagged JSON response shown afterward. This declarative specification includes every instruction in the system messages without restating those instructions as directions to the reader. After

internal deliberation, each message requires a single line of strict JSON with an integer rating for every key and no text after the closing tag. For fidelity, the scale is 0 for not satisfied or empty, malformed, or unreadable text, 1 for poor, 2 for weak, 3 for acceptable with noticeable issues, 4 for strong with only minor issues, and 5 for fully satisfied with no issues. The fidelity message specifies that each rating concerns only the continuation and only its named criterion. It does not reward length, detail, confidence, padding, or filler by themselves, and it does not let a weak or short source prompt excuse a weak continuation. For content preservation, each rating measures how much of the named property is present, not whether that property is desirable: 0 for absent, 1 for an isolated trace, 2 for a few instances, 3 for a moderate degree, 4 for most of the candidate with minor exceptions, and 5 for fully present or present throughout the candidate. The content-preservation message separates coverage from correctness: a wrong figure, date, or name reduces detail precision without itself counting as a contradiction, while a reversed claim or sentiment counts as a contradiction even if the candidate retains every detail. It bases each rating only on the reference and candidate, disregards length, tone, and fluency, and assigns four zeros to an empty or unreadable candidate.

User messages and response schema. For fidelity, the source prompt is the sentence recovered by the generation prompt rule from the source article’s first 32 words, and the continuation excludes that prompt prefix. The following templates preserve the literal delimiters, field order, placeholders, and required output keys.

FIDELITY USER MESSAGE

```
[[PROMPT]]:
{source prompt}
```

```
[[CONTINUATION]]:
{watermarked continuation}
```

CONTENT-PRESERVATION USER MESSAGE

```
[[REFERENCE]]:
{marked text}
```

```
[[CANDIDATE]]:
{attacked text}
```

REQUIRED FINAL ANSWER FOR FIDELITY

```
<json>{"fluency": <0-5>, "coherence": <0-5>,
"relevance": <0-5>, "informativeness": <0-5>}</json>
```

REQUIRED FINAL ANSWER FOR CONTENT PRESERVATION

```
<json>{"content_recall": <0-5>, "detail_precision": <0-5>,
"information_injection": <0-5>, "contradiction":
<0-5>}</json>
```

Scoring and inference. Let each r denote one criterion rating from Table 4. We give the four criteria equal weight

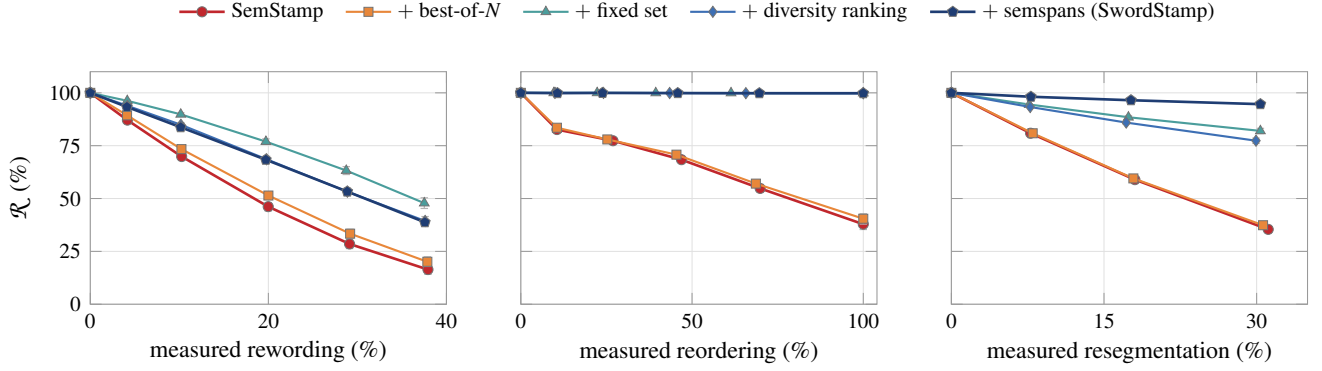


Figure 8: Controlled-edit response for SemStamp across the five component stages of Figure 5. The comparison tests dependence on partition geometry.

Table 4: Operational definitions and score effects for both model-based judges. Each criterion is rated independently from 0 to 5.

Criterion	Operational definition	Score effect
<i>Reference-free fidelity F</i>		
Fluency	The continuation uses grammatical, well-formed, and natural sentences without spelling, punctuation, or awkward-phrasing errors.	Raises <i>F</i>
Coherence	The ideas connect logically and remain internally consistent. Logical contradictions and paradoxes reduce this rating, but sentence fragments and lists do not when their ideas follow a logical sequence.	Raises <i>F</i>
Relevance	The text is a sensible, on-topic continuation of the supplied source prompt and remains focused rather than drifting from it.	Raises <i>F</i>
Informativeness	The continuation supplies substantive, specific facts or mechanisms. Restatements of the prompt, obvious truisms, and vague padding reduce this rating regardless of length.	Raises <i>F</i>
<i>Pairwise content preservation Q</i>		
Content recall	The candidate takes up each claim, point, and fact in the reference in some form, including a restatement, rewording, opposition, or wrong value, rather than omitting it.	Raises <i>Q</i>
Detail precision	The specific facts, figures, names, and dates stated by the candidate are correct relative to the reference.	Raises <i>Q</i>
Information injection	The candidate adds general claims, content, or context that the reference does not state.	Lowers <i>Q</i>
Contradiction	The candidate asserts a claim or sentiment that directly opposes the reference’s overall meaning.	Lowers <i>Q</i>

and invert the two content-error ratings:

$$\begin{aligned}
 F(x) &= 5\%(r_{\text{fluency}} + r_{\text{coherence}} + r_{\text{relevance}} + r_{\text{informativeness}}), \\
 Q(x, y) &= 5\%(r_{\text{recall}} + r_{\text{detail}} \\
 &\quad + (5 - r_{\text{injection}}) + (5 - r_{\text{contradiction}})).
 \end{aligned}
 \tag{8}$$

Both aggregates lie in $[0\%, 100\%]$. A score of 100% requires all four fidelity properties for *F*, and it requires complete recall and detail precision with no information injection or contradiction for *Q*. We use Qwen/Qwen3-32B through vLLM [28] with temperature 0.6, top-*p* 0.95, top-*k* 20, minimum-*p* 0, and three random seeds. We apply the model’s chat template, retain only the final answer after its reasoning block, compute Equation (8) for each run, and average the three run scores for each document. The evaluator extracts the tagged JSON object and retries an unparseable response up to three times. It scores an empty continuation or candidate as 0 on every

criterion and excludes unparseable responses.

Construct limits. Fidelity measures prompt-conditioned writing quality without comparing the continuation to a factual reference, so it does not establish factual accuracy against the source article. Content preservation measures agreement with the marked reference, so it can preserve errors already present in that reference. Its equal weights encode a measurement choice rather than a universal equivalence among omission, detail error, added information, and contradiction. Both model-based scores proxy human judgments. Neither establishes preservation of every fact and implication.

A.3 Fidelity by Criterion

The aggregate fidelity score can hide differences among its four criteria. Table 5 reports the aggregate and each criterion for the unwatermarked baseline and all six schemes.

Table 5: Clean-text reference-free fidelity overall and by criterion. Cells give mean percentages and 95% Student-*t* intervals across documents after averaging three judge runs per document; higher is better.

Configuration	Fluency	Coherence	Relevance	Informativeness	Overall
Unwatermarked	92.2 [91.1,93.3]	85.2 [83.6,86.8]	84.3 [82.5,86.1]	77.8 [76.0,79.6]	84.9 [83.4,86.3]
SemStamp	88.8 [87.5,90.0]	79.7 [78.0,81.5]	79.2 [77.3,81.0]	73.9 [72.1,75.7]	80.4 [78.9,81.9]
k-SemStamp	89.2 [88.0,90.3]	79.9 [78.2,81.5]	78.9 [77.0,80.7]	74.5 [72.8,76.2]	80.6 [79.1,82.1]
PMark (online)	92.8 [92.0,93.6]	85.3 [83.9,86.7]	85.0 [83.4,86.5]	81.1 [79.7,82.5]	86.0 [84.8,87.2]
SAMark	92.1 [91.2,92.9]	81.9 [80.4,83.4]	77.8 [75.9,79.7]	74.2 [72.6,75.8]	81.5 [80.2,82.8]
SwordStamp	87.4 [86.2,88.6]	77.6 [75.9,79.3]	77.1 [75.2,79.0]	72.4 [70.7,74.1]	78.6 [77.1,80.1]
k-SwordStamp	86.0 [84.7,87.3]	75.6 [73.9,77.3]	75.4 [73.5,77.4]	68.0 [66.2,69.7]	76.2 [74.7,77.8]

Criterion means for SwordStamp are 1.4–2.1 percentage points lower than for SemStamp. Criterion means for k-SwordStamp are 3.2–6.5 percentage points lower than for k-SemStamp, with the largest gap in informativeness.

A.4 EDA Prompt and Decoding

The reported no-box EDA configurations use Qwen/Qwen2.5-3B-Instruct as the paraphraser and BAAI/bge-base-en-v1.5 as their sole surrogate encoder. They use the same paraphrasing prompt as Diaa et al. [12]. The paraphraser receives the full marked document and the following system-prompt content, reflowed only for typesetting. **System prompt**

You are an expert copy-editor. Please rewrite the following text in your own voice and paraphrase all sentences. Ensure that the final output contains the same information as the original text and has roughly the same length. Do not leave out any important details when rewriting in your own voice. Do not include any information that is not present in the original text. Do not respond with a greeting or any other extraneous information. Skip the preamble. Just rewrite the text directly.

User message and assistant prefix

```
[[START OF TEXT]]
{marked text}
[[END OF TEXT]]

[[START OF PARAPHRASE]]
```

We apply the model’s chat template to the system and user messages, then append the assistant prefix shown above. At each output unit, the attack samples K continuations with temperature 1.0, top- p 0.95, and at most 96 new tokens per candidate. It applies the anchor rule and segmenter for the evaluated EDA configuration and retains the candidate with the greatest cosine distance under the BGE surrogate (Algorithm 2). The attack conditions each subsequent sample on the full prompt and the accepted paraphrase prefix. The sentence-level implementation stops when the paraphraser emits EOS or the rewrite reaches 512 generated tokens.

EDA-D uses the same prompt and decoding settings; Algorithm 3 specifies how it selects candidates using the provider’s

region test.

A.5 Edit Signatures of Whole-Text Attacks

Figure 9 compares rewording, reordering, resegmentation, and content preservation among outputs that evade detection at 5% FPR.

Attack settings produce different mixtures of edits among successful evasions. Raising Dipper’s order setting from 20 to 80 increases mean reordering from 6.6% to 19.5%, while mean rewording changes from 21.1% to 23.3%. At setting 80, 48.5% of the order-sweep evasions meet $\bar{q}$, compared with 6.7% for the lexical sweep, whose mean rewording reaches 47.1%. At the six shared budgets through $K = 32$, (pos, sent) produces more reordering per unit of rewording than (bag, sent) and (bag, span). This pattern matches the anchor objectives: positional anchors reward relocating source content, while bag anchors score each candidate against every source unit regardless of position.

Quality gating removes a larger share of evasions at higher attack settings. Across both Dipper sweeps and all three EDA sweeps, the fraction of evasions meeting $\bar{q}$ decreases as diversity or budget increases. For $K \in \{1, 2, 4, 8\}$, 64.7%–81.7% of (pos, sent) evasions meet $\bar{q}$, compared with 48.5%–58.3% across Dipper’s order-diversity sweep. Because ASR requires both detector evasion and content preservation, the quality-pass difference contributes to EDA’s higher ASR than Dipper on k-SemStamp at $\bar{q} = 90\%$ in Figure 6.

The stage-specific EDA trajectories identify the structural edits that remain among evasions. The fixed-set stage’s (bag, sent) configuration produces 17.0%–24.9% mean resegmentation, compared with 11.7%–17.4% for the final stage’s (bag, span) configuration. The final-stage configuration has a lower quality-pass fraction at 4 of the 5 shared budgets from 4 through 64; at $K = 64$, the rates are 24.9% and 24.2%, respectively. This comparison describes residual evasion channels, not the effect of semspans alone, because the stages differ and each trajectory conditions on a different set of evasions. The trajectories therefore separate how an attack evades detection from whether the resulting text remains within the quality budget.

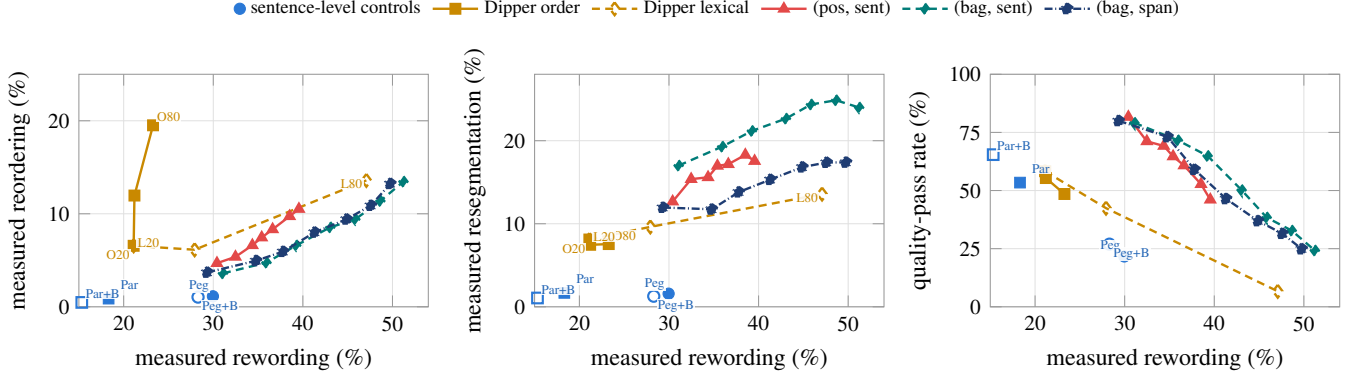


Figure 9: Mean edit rates and quality-pass fractions among outputs that evade at 5% FPR. The first two panels report reordering and resegmentation against rewording; the right panel reports the fraction also meeting $Q(x, y) \geq \bar{q} = 90\%$. Markers identify Pegasus (circles), Parrot (squares), and bigram selection (hollow); lines connect increasing attack settings. Sentence-level controls and Dipper use k-SemStamp. EDA labels list (anchor rule, unit): (pos, sent) for k-SemStamp, (bag, sent) for the fixed-set stage, and (bag, span) for the semspan stage. Each point summarizes one attack setting over detector-evasive outputs; connected trajectories do not isolate causal effects.

A.6 Watermarked and Attacked Examples

Individual passages expose what the two quality proxies and detector scores represent in text. Figure 10 contrasts a high-preservation evasion against SemStamp with a quality-limited evasion against k-SwordStamp. In panel (a) of Figure 10, EDA-P preserves 96.7% of the marked passage’s content while lowering the SemStamp score below its 5% FPR threshold. The rewrite combines rewording, reordering, and resegmentation; the panel reports each measured edit rate beside the attacked text. In panel (b), the clean k-SwordStamp continuation receives 78.3% fidelity, including 93.3% fluency. The attack again lowers the detector score below threshold, but its 85.0% content-preservation score fails $\bar{q} = 90\%$. These examples illustrate the measured trade-off.

Units: ■ hit; ■ hit, same region; ■ miss; ■ seed. Superscripts index detector units.

(a) SemStamp, EDA-P, $K = 4$.

Watermarked text. $F(x) = 90.0\%$, $z_k(x) = 3.46$.

⁰Donald Trump just contradicted his own ridiculously implausible claim not to have known Matt Whitaker’s hostility toward the Russia investigation before naming him as acting attorney general.¹ In an interview with the Hill today, Trump said he’s “aware” that Whitaker has “spoke[n] out on it,” but went on to assert that he doesn’t necessarily believe what Whitaker says.² Or at least, Trump added, he doesn’t “believe this craziness” (emphasis mine).³ @donaldJtrumpjr: It’s “very possible” Trump Tower in Moscow was still being discussed through November 2016 @thehill <https://t.co/dHrJz74pkW> pic.twitter.com/p1Qq6Cf8Jd— POLITICO (@politico) December 13, 2018 Trump is a consummate salesman, and today’s remarks are a textbook illustration of a technique called the “double bind.” To explain, consider a variation on Trump’s statement: “I don’t necessarily believe what Whitaker says, but I do think we should investigate him anyway.” This message actually contains two claims: One is the explicit one — namely, that Trump doesn’t necessarily believe what Whitaker says.⁴ The second, however, is the implicit one, which says that Whitaker probably does believe what he says (since there’s no reason for Trump to think he shouldn’t be believed unless he himself believes him).

(b) k-SwordStamp, EDA-S (bag, span), $K = 64$.

Watermarked text. $F(x) = 78.3\%$, $z_k(x) = 6.60$.

⁰A few weeks ago we showed you images of incredible¹ robotic library logistics, including the Telelift system of the National Széchenyi Library in Budapest, Hungary.² It’s easy to see why librarians are so enamored³ of the system; it allows them to send automated messages to customers⁴ for items they want and need, and delivers the materials directly to the patrons.⁵ This particular system relies on a heavily mechanized conveyor belt to bring⁶ books to the proper patrons.⁷ But sometimes you need more control.⁸ Sometimes your customers are spread out over a large distance, or perhaps need access⁹ to high-cost or fragile material.¹⁰ Or maybe they’re just a little impatient.¹¹ If any of these factors is present,¹² then you might want to consider giving your customers¹³ mobile robots that carry their desired content to them.¹⁴ In other words, you need a RoboCart.¹⁵ From Bot & Dolly.¹⁶ That’s right, the company responsible for the mind-blowing film The 20th Century Foxworks production, Moonrise¹⁷ Kingdom, has made a way for librarians to deliver the goods to their patrons.¹⁸ Here’s the video explaining how it all works: It’s¹⁹ pretty much exactly what you would expect from the guys²⁰ who made Moonrise Kingdom, but with the added bonus of actual useful content.²¹ So now, in addition to delivering items directly to your patrons,²² you can also do it with style and class, the kind²³ only associated with the upper class and members of the scientific community.

Attacked text. $Q(x, y) = 96.7\%$, $z_k(y) = 1.41 < \tau(5\%) = 2.12$.

Measured edits: $r_{\text{word}} = 29.8\%$, $r_{\text{order}} = 13.8\%$, $r_{\text{seg}} = 60.0\%$.

⁰Donald Trump recently contradicted himself by denying his previous claim that he didn’t know Matt Whitaker harbored hostility towards the Russia investigation.¹ In an interview with The Hill today, Trump conceded that Whitaker had “spoken out on it,” but insisted he didn’t necessarily believe what Whitaker said.² Trump even went so far as to call it “craziness.” [Politico] tweeted: @donaldJtrumpjr: It’s “very possible” Trump Tower in Moscow was still being discussed through November 2016 @thehill][<https://t.co/dHrJz74pkW>] pic.twitter.com/p1Qq6Cf8Jd— POLITICO (@politico) December 13, 2018.³ Trump is a masterful communicator, and today’s statement perfectly illustrates what’s known as the “double bind.”⁴ To understand it, consider a modified version of his statement: “I don’t necessarily believe what Whitaker says, but I do think we should investigate him anyway.”⁵ This statement contains two distinct claims: Explicitly, Trump claims he doesn’t necessarily believe Whitaker.⁶ Implicitly, however, he’s suggesting that Whitaker probably does believe what he says (unless Trump suspects Whitaker of not being credible).

Attacked text. $Q(x, y) = 85.0\%$, $z_k(y) = 0.88 < \tau(5\%) = 2.40$.

Measured edits: $r_{\text{word}} = 46.6\%$, $r_{\text{order}} = 23.7\%$, $r_{\text{seg}} = 21.3\%$.

⁰Just a few weeks back, we introduced¹ you to a series of impressive robotic systems in action, notably the Telelift² system found in the National Széchenyi Library in Budapest, Hungary.³ It’s hard not to be impressed by how much it’s revolutionized librarian⁴ workflows; they can now automate notifications for customers who need⁵ specific items, and then transport those materials right to them.⁶ This particular Telelift relies on a robust conveyor belt to ensure books⁷ end up in the correct hands.⁸ However, there are scenarios where more personalized solutions are required.⁹ Perhaps the library patrons are located far apart, necessitating faster¹⁰ access to certain materials, or maybe the items are quite valuable¹¹ and handling them manually poses a risk.¹² In any of these cases, the Telelift might not be the best fit.¹³ For those scenarios, libraries might consider using mobile robots, or “RoboCarts”¹⁴ to give patrons exactly what they want, right at their door.¹⁵ This innovative solution comes from the team behind The 20th Century Fox’s blockbuster¹⁶ Moonrise Kingdom, but don’t let that fool you—these bots also serve a purpose.¹⁷ This is more than just a novel method for returning and retrieving materials;¹⁸ the company’s expertise also shines in the seamless integration of storytelling.¹⁹ So, with the Telelift system, the library doesn’t²⁰ only streamline workflows, but also enhances the overall experience with a touch of cinematic flair.

Figure 10: Sample watermarked and attacked passages. Colored text shows the detector’s recovered units and decisions. Green and yellow both count as detector hits. Gray marks a context-dependent seed; fixed-mask detectors score the first unit.