SEMADAPT: ADAPTIVE SEMANTIC WATERMARKING WITH PARAPHRASE ROBUSTNESS FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Watermarking large language model (LLM) outputs is a promising approach for establishing content provenance and mitigating misuse. However, LLM-generated text can be easily paraphrased by users or automated tools, disrupting token-level watermark statistics and reducing detection reliability. Although existing methods attempt to improve robustness against such paraphrases, they often do so without adequately considering text quality, limiting their practical applicability. We propose `SemAdapt`, an adaptive semantic watermarking method for robust LLM watermarking under paraphrasing. `SemAdapt` injects watermark signals at the level of semantic vocabulary buckets rather than individual tokens, and refines the vocabulary partition using lexical substitutions observed under paraphrasing. To balance detectability, paraphrase robustness, and text quality, `SemAdapt` further adapts watermark strength according to uncertainty over the same semantic bucket space, with controller parameters selected offline using multi-objective Bayesian optimization under noisy paraphrase evaluations. Extensive experiments demonstrate that `SemAdapt` achieves consistently stronger robustness against diverse paraphrase attacks than existing baselines, while maintaining comparable text quality and detectability.

1 INTRODUCTION

Large Language Models (LLMs) have enabled open-ended text generation across a wide range of domains, but this capability has also raised concerns about content provenance, plagiarism, and the misuse of AI-generated text, as machine-produced outputs become difficult to distinguish from human writing Augenstein et al. (2024); Bender et al. (2021). Text watermarking aims to address these concerns by embedding a hidden statistical signal into model outputs that is imperceptible to humans yet detectable by an algorithm Kirchenbauer et al. (2023). However, in real-world use, watermarked generations may be edited or paraphrased by users and automated rewriting tools, whether for benign reuse or to evade detection, weakening watermark signals and making reliable attribution more challenging.

A classical class of LLM watermarking methods embeds watermarks at the token level by biasing next-token sampling according to a pseudorandom green–red partition of the vocabulary, yielding statistically detectable deviations Kirchenbauer et al. (2023). However, such methods are fragile under paraphrasing attack, as paraphrastic rewrites can alter the token sequence while the watermark signal depends on prefix-dependent vocabulary partitions and token-level green assignments. To improve robustness, Zhao et al. Zhao et al. (2023) adopt a fixed global green–red vocabulary partition, but such schemes can degrade text fluency and are vulnerable to spoofing attacks Sadasivan et al. (2023). Recent attempts to mitigate quality degradation include robust distortion-free watermarking proposed by Kuditipudi et al. (2023), which injects watermarks through probabilistic sampling without introducing bias. However, this approach incurs substantial computational overhead during detection, limiting its practicality at scale. SemStamp Hou et al. (2024a) partitions sentence embeddings into Locality-Sensitive Hashing (LSH) semantic regions and uses rejection sampling to select the next sentence, improving paraphrase robustness while preserving text quality. However, its sentence-level signal remains vulnerable to cross-sentence perturbations such as sentence reordering or splicing.

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

Beyond the watermarking unit itself, another line of work improves the quality-detectability trade-off by adapting the watermark strength during generation Huo et al. (2024); Liu & Bu (2024); Wang et al. (2025). TS-watermark Huo et al. (2024) uses multi-objective optimization with the Multiple-Gradient Descent Algorithm (MGDA) to allocate watermark strength across tokens, but it does not consider robustness under paraphrasing. Adaptive Watermarking Liu & Bu (2024) further modulates watermark strength based on token entropy to improve text quality while maintaining robustness; however, they rely on manually specified entropy thresholds. Taken together, existing watermarking methods inspire us to design an effective watermarking algorithm that jointly considers detectability, robustness, and text quality.

In this work, we propose `SemAdapt`, an adaptive semantic watermarking method designed to improve robustness to paraphrase attacks while preserving detectability and text quality. `SemAdapt` first constructs a semantic vocabulary partition from token embeddings and selectively refines it using lexical substitution relations observed in source–paraphrase pairs. During generation, watermark bias is applied to selected semantic buckets rather than individual tokens. Since uncertainty over these buckets varies across generation steps, a fixed watermark strength may cause unnecessary distortion. `SemAdapt` therefore uses a bucket-entropy controller to adapt the watermark strength accordingly. Its parameters are selected offline using multi-objective Bayesian optimization over detectability, text quality, and paraphrase robustness.

Our primary contributions are summarized as follows:

- We propose `SemAdapt`, an adaptive semantic watermarking method for LLMs that improves robustness to paraphrase attacks while preserving text quality and reliable detectability. It offers a new perspective on quality-preserving robust watermark design.
- We develop a semantic bucket-level watermarking design that constructs vocabulary buckets from token embeddings and refines them using lexical substitution relations observed under paraphrasing, making watermark signals more stable.
- We introduce a bucket-entropy controller that adapts watermark strength in the same semantic space used for watermarking, improving the quality–robustness trade-off through offline multi-objective optimization.
- We conduct extensive experiments across diverse paraphrase attacks, showing that `SemAdapt` achieves stronger robustness than competitive baselines at comparable text quality.

2 Related Work

Token-level Text Watermarking. Token-level watermarking is a classical paradigm that embeds signals by modifying the token probability distribution during autoregressive generation. A representative line of work, Kirchenbauer et al. (2023), partitions the vocabulary into green and red lists to increase the likelihood of green-listed tokens. This framework has inspired a large body of follow-up research. Several works aim to improve text quality by adapting the watermark strength to token-level entropy, reducing distortion in low-entropy regions while preserving detectability Lee et al. (2024); Lu et al. (2024). Another line of work focuses on distortion-free watermarking, ensuring that the marginal token distribution remains unchanged in expectation, thereby better preserving text quality Kuditipudi et al. (2023); Wu et al. (2023); Dathathri et al. (2024). Beyond text quality, robustness has also been studied. Zhao et al. (2023) improves robustness by using a fixed global green–red token partition, while Liu & Bu (2024) aims to jointly enhance both security and robustness through adaptive watermarking strategies. Semantic-aware watermarking methods have also been proposed to improve robustness against paraphrasing. These methods incorporate contextual embeddings or semantic hashing to guide watermark key generation or token partitioning, thereby reducing sensitivity to surface-form variations Liu et al. (2023); Guo et al. (2024); Ren et al. (2024).

Sentence-level Text Watermarking. To mitigate the paraphrase sensitivity of token-level watermarking, prior work has explored sentence-level watermarking, embedding watermarks at the semantic level by constraining entire generated sentences rather than individual tokens. For example, SemStamp Hou et al. (2024a) enhances robustness by applying rejection sampling to ensure that generated sentences lie within predefined green LSH-separated regions of the embedding space. Similarly, other works further refine the construction of green semantic regions to improve robustness against

paraphrasing attacks Hou et al. (2024b); Dabiriaghdam & Wang (2025); Huo et al. (2025). However, such sentence-level approaches remain vulnerable to cross-sentence attacks, where adversaries delete or splice sentences, thereby disrupting the watermark signal.

Multi-Objective Optimization in Watermarking. Recent work Huo et al. (2024); Wang et al. (2025) applies multi-objective optimization (MOO) to balance competing goals in text watermarking, typically using the Multiple Gradient Descent Algorithm (MGDA). These methods primarily focus on the trade-off between detectability and text quality, while robustness under paraphrasing is not explicitly optimized. Incorporating paraphrase robustness as an optimization objective is challenging because robustness evaluation depends on diverse and stochastic paraphrasing attacks, making the objective noisy, expensive, and non-differentiable. This poses a limitation for MGDA-based approaches, which require continuous and differentiable objectives. In contrast, our work uses multi-objective Bayesian optimization to search for watermark controller parameters under noisy robustness evaluation.

3 PRELIMINARY

3.1 WATERMARKING

Following the standard green-list watermarking framework Kirchenbauer et al. (2023), a watermark is embedded during autoregressive generation by biasing the token sampling distribution. At generation step t, the model produces logits $\mathbf{z}_t \in \mathbb{R}^{|\mathcal{V}|}$ over the vocabulary $\mathcal{V}$. A secret-keyed hash of the recent prefix context $c_t = (w_{t-l}, \ldots, w_{t-1})$ pseudorandomly partitions the vocabulary into a green set $\mathcal{G}_t \subset \mathcal{V}$ and its complement, with green ratio $\gamma_0 = |\mathcal{G}_t|/|\mathcal{V}|$. A positive bias δ is added to the logits of green tokens before sampling. For a generated sequence of length T, detection counts the number of green tokens X and computes $Z = (X - \gamma_0 T)/\sqrt{\gamma_0(1-\gamma_0)T}$, which is often interpreted using a standard-normal approximation. That interpretation requires assumptions on the null and dependence across positions; empirical calibration is needed when they are not established.

3.2 LOCALITY-SENSITIVE HASHING

Locality-Sensitive Hashing (LSH) Indyk & Motwani (1998) is a randomized hashing scheme in which similar inputs are more likely to receive the same hash value. In `SemAdapt`, we use LSH to construct an initial semantic partition of the vocabulary by mapping token embeddings into semantic buckets. We adopt the cosine-preserving random hyperplane scheme Charikar (2002). Given an embedding vector $\mathbf{v} \in \mathbb{R}^h$ and d random hyperplanes $\mathbf{n}^{(1)}, \ldots, \mathbf{n}^{(d)} \sim \mathcal{N}(0, I)$, the LSH signature is

$$\mathrm{LSH}(\mathbf{v}) = \left[\mathbb{I}[\mathbf{n}^{(1)\top}\mathbf{v} > 0] \,\|\, \cdots \,\|\, \mathbb{I}[\mathbf{n}^{(d)\top}\mathbf{v} > 0]\right].$$

This maps nearby vectors to identical or similar binary signatures with high probability.

3.3 MULTI-OBJECTIVE BAYESIAN OPTIMIZATION

Let $\boldsymbol{\theta} \in \Theta$ denote the parameters to be optimized, and let $f(\boldsymbol{\theta})$ denote an expensive black-box objective. Bayesian Optimization (BO) Brochu et al. (2010) optimizes such objectives without analytic gradients by maintaining a surrogate posterior $p(f \mid \mathcal{D})$ from past evaluations $\mathcal{D} = \{(\boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i))\}_{i=1}^n$ and using an acquisition function $a(\boldsymbol{\theta}; p(f \mid \mathcal{D}))$ to select new query points.

In multi-objective optimization, the goal is to optimize a vector-valued objective $\mathbf{f}(\boldsymbol{\theta}) = (f_1(\boldsymbol{\theta}), \ldots, f_M(\boldsymbol{\theta}))$, where the objectives may conflict. A solution $\boldsymbol{\theta}^*$ is Pareto optimal if no other $\boldsymbol{\theta} \in \Theta$ improves at least one objective without worsening any other. The set of Pareto-optimal solutions forms the Pareto frontier, which characterizes the achievable trade-offs. BO can be extended to this setting by searching for parameter configurations that improve the Pareto frontier. Multi-objective acquisition functions such as Expected Hypervolume Improvement (EHVI) Daulton et al. (2020) guide this search and are well suited to the noisy, expensive, and non-differentiable objectives encountered in watermark evaluation.

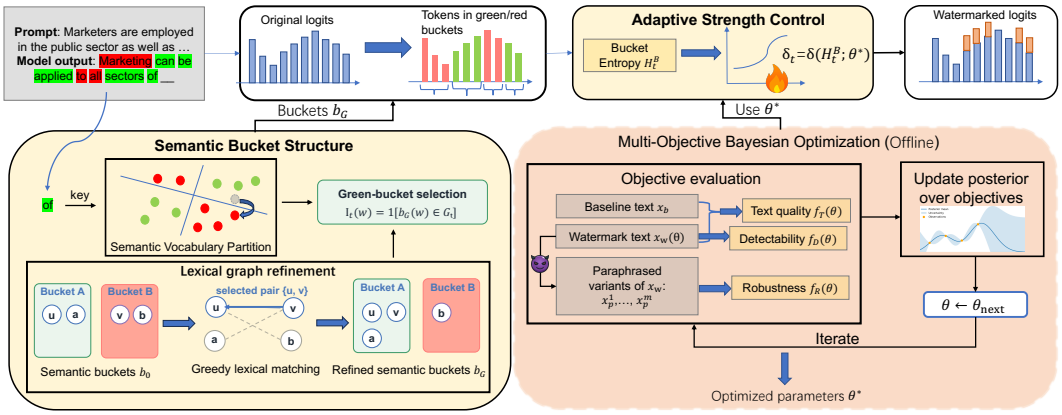


Figure 1: Overview of SemAdapt. **Bottom Left:** Token embeddings define an initial semantic vocabulary partition into buckets b_0, which is refined using lexical matching observed under paraphrasing to obtain the refined bucket mapping b_G. **Bottom right:** The optimized controller parameters $\boldsymbol{\theta}^\star$ are selected offline using multi-objective Bayesian optimization over detectability, text quality, and paraphrase robustness. **Top:** Token probabilities are summed within each semantic bucket to compute bucket entropy H_t^B, which determines the adaptive watermark strength δ_t using $\boldsymbol{\theta}^\star$.

4 Method

SemAdapt performs bucket-level watermarking over a lexically refined semantic vocabulary partition and adapts the watermark strength using entropy measured over the same bucket space. Figure 1 provides an overview of the method. Algorithms 1 and 2 in Appendix A summarize the generation and detection procedures.

4.1 Semantic Vocabulary Partition

Token-level watermark assignments are sensitive to lexical substitutions: semantically similar tokens may receive different watermark labels despite serving similar roles in generation. We therefore define watermark carriers over semantic vocabulary buckets rather than individual token identities. Let $\mathbf{e}(w)$ be token w's input embedding in the generation model. Using the random-hyperplane LSH in Section 3.2, we define the initial vocabulary mapping

$$b_0(w) = \mathrm{LSH}_{\mathrm{vocab}}(\mathbf{e}(w)), \qquad b_0 : \mathcal{V} \to \{0, \ldots, B-1\}, \tag{1}$$

where $B = 2^d$ for a d-bit LSH signature. Tokens with nearby embeddings are therefore more likely to share the same bucket. Appendix M further evaluates the semantic structure of the resulting vocabulary buckets.

4.2 Lexical Graph Refinement

The initial mapping b_0 provides a coarse semantic partition of the vocabulary in embedding space. However, it is not explicitly aligned with lexical substitutions observed under paraphrasing: semantically close alternatives may still be separated by an LSH boundary. We therefore use empirical substitution relations to selectively refine b_0, while preserving as much of the original semantic partition as possible.

Lexical substitution graph. We estimate lexical relations offline from a separate set of source–paraphrase pairs. After aligning each source with its paraphrase, let $n(u \to v)$ denote the number of times token u is aligned to lexical substitute v. To restrict the graph to semantically plausible alternatives, let $\mathcal{C}_k(u)$ contain the k nearest neighbors of u under embedding cosine similarity. We define the directional substitution confidence

$$p(v \mid u) = \frac{n(u \to v)\mathbb{I}[v \in \mathcal{C}_k(u)]}{\sum_{x \in \mathcal{C}_k(u)} n(u \to x)}, \qquad r_{uv} = \max\{p(v \mid u), p(u \mid v)\}. \tag{2}$$

We retain candidate pairs with sufficient empirical support and rank them by r_{uv}. Rather than enforcing all observed relations, which could connect many tokens through long substitution chains, we greedily accept an edge only when neither endpoint has been selected previously. This produces a matching M satisfying $\deg_M(w) \leq 1$ for all $w \in \mathcal{V}$, restricting refinement to disjoint high-confidence lexical pairs. The edge-selection procedure is summarized in Algorithm 3 in Appendix B.

Minimum-change bucket refinement. The matching M identifies lexical pairs that should receive the same bucket assignment. We enforce these constraints while minimizing changes to the initial partition. Let π_w denote the normalized frequency of token w in the source texts used to construct the lexical graph. We define the refined mapping as

$$b_G \in \underset{b:\, b(u)=b(v),\ \forall \{u,v\}\in M}{\arg\min} \sum_{w\in\mathcal{V}} \pi_w \mathbb{I}[b(w) \neq b_0(w)]. \tag{3}$$

Because M is a matching, this optimization decomposes over independent pairs. For a selected pair $\{u, v\} \in M$ with $b_0(u) \neq b_0(v)$, satisfying the constraint requires moving at least one endpoint. The minimum-change rule keeps the initial bucket of the more frequent token and reassigns the less frequent token to that bucket. Pairs that already share a bucket and tokens outside M remain unchanged. For a fixed matching M, this rule exactly solves Eq. equation 3. Proposition 1 in Appendix C formalizes this optimality and bounds the induced change in green probability.

4.3 Bucket-Level Watermarking

Given the refined mapping b_G from Section 4.2, `SemAdapt` performs watermarking over vocabulary buckets rather than individual tokens. At generation step t, let $c_t = (y_{t-\ell}, \ldots, y_{t-1})$ denote the recent token context. A secret-keyed hash $\rho_K(c_t)$ is used to pseudorandomly select a set of green buckets $G_t \subseteq \{0, \ldots, B-1\}, |G_t| = \gamma B$, where $\gamma \in (0, 1)$ is the green-bucket ratio. Thus, b_G determines the semantic bucket of each vocabulary token, while the keyed function $\mathcal{G}_K$ determines which buckets are green under the current n-gram context.

Watermark generation. A candidate token $w \in \mathcal{V}$ is treated as green whenever its bucket assignment belongs to G_t. We define

$$I_t(w) = \mathbb{I}[b_G(w) \in G_K(c_t)]. \tag{4}$$

Let $z_t(w)$ denote the original pre-softmax logit of token w. The watermarked logits are $\widetilde{z}_t(w) = z_t(w) + \delta_t I_t(w)$, where $\delta_t \geq 0$ is the watermark strength at step t. For a selected lexical pair $\{u, v\} \in M$, refined mapping b_G enforces $b_G(u) = b_G(v)$. Hence, replacing u by v preserves its bucket-level watermark assignment.

Statistical detection. Given a sequence $\{y_t\}_{t=1}^T$, the detector uses the secret key to reconstruct the same context-dependent green-bucket sets $\{G_t\}_{t=1}^T$. For each observed token, we compute $I_t = \mathbb{I}\left[b_G(y_t) \in G_t\right]$. Under the ideal random-key model, uniform selection of a fixed fraction γ of buckets gives $\Pr\left[b_G(y_t) \in G_t\right] = \gamma$ for any fixed token bucket, regardless of the number of vocabulary tokens assigned to that bucket. Thus, unequal bucket sizes do not by themselves shift the nominal null mean. We use the nominal green-hit statistic

$$Z = \frac{\sum_{t=1}^T (I_t - \gamma)}{\sqrt{T\gamma(1-\gamma)}}. \tag{5}$$

Null calibration. The marginal identity above establishes the centering of the statistic under the ideal random-key model, but does not by itself determine its variance or tail distribution under a fixed deployed key. We therefore use empirically calibrated thresholds from length-matched null texts for all reported false-positive operating points, rather than relying on a standard-normal approximation. Appendix D analyzes the corresponding null moments and calibration in detail.

4.4 Adaptive Strength Control

The bucket-level watermark determines where bias is applied; we next determine how strongly it should be applied at each generation step. `SemAdapt` adapts the watermark strength using uncertainty measured over the same semantic buckets.

4.4.1 BUCKET-ENTROPY CONTROLLER

Let $P_t(w)$ denote the base model's next-token distribution at generation step t. We aggregate this distribution over the refined mapping b_G: $P_t^B(j) = \sum_{w:,b_G(w)=j} P_t(w), j \in 0, \ldots, B-1$. The corresponding bucket entropy is

$$H_t^B = -\sum_{j=0}^{B-1} P_t^B(j) \log P_t^B(j). \tag{6}$$

Unlike token-level entropy, H_t^B measures uncertainty directly over the watermark carriers. A low value indicates that the model probability mass is concentrated in a small number of buckets, , whereas a high value indicates that the mass is distributed across more candidate buckets.

We parameterize the watermark strength as

$$\delta_t = \delta_{\text{base}} \left(1 - \exp\left(-\alpha H_t^B\right)\right), \tag{7}$$

where $\delta_{\text{base}} > 0$ sets the overall strength scale and $\alpha > 0$ controls the sensitivity to bucket entropy. The controller is specified by $\boldsymbol{\theta} = \big(\alpha, \delta_{\text{base}}\big)$, which defines a family of adaptive watermarking strategies. We optimize these parameters offline using multi-objective Bayesian optimization to balance detectability, text quality, and robustness, as described in Sec. 4.4.2.

4.4.2 MULTI-OBJECTIVE OPTIMIZATION FORMULATION

We optimize the controller parameters $\boldsymbol{\theta}$ to achieve a trade-off among three objectives: detectability, text quality, and robustness. This is formulated as a vector-valued objective

$$\mathbf{f}(\boldsymbol{\theta}) = \big(f_D(\boldsymbol{\theta}),\ f_Q(\boldsymbol{\theta}),\ f_R(\boldsymbol{\theta})\big). \tag{8}$$

Detectability. We try to maximize watermark detectability, measured by the Z-score defined in Sec. 4.3. The detectability objective is given by

$$f_D(\boldsymbol{\theta}) = Z(\boldsymbol{\theta}), \tag{9}$$

where $Z(\boldsymbol{\theta})$ is the Z-score of watermark text using adaptive δ_t.

Text Quality. We aim to preserve semantic fidelity between watermarked and unwatermarked text, minimizing distortion introduced by watermarking. Let $\phi(\cdot)$ denote a sentence embedding function. We define the text quality objective as the cosine similarity between the embedded representations:

$$f_Q(\boldsymbol{\theta}) = \frac{\langle \phi(x_{\text{wm}}),\ \phi(x_{\text{base}}) \rangle}{\|\phi(x_{\text{wm}})\|_2 \, \|\phi(x_{\text{base}})\|_2}, \tag{10}$$

where x_{wm} and x_{base} are watermarked generated text and unwatermarked generated text.

Robustness. We aim to maximize the robustness of the watermark under paraphrasing attacks. Since such attacks are inherently stochastic, robustness evaluation is noisy. We therefore define robustness as the expected detectability under attack,

$$f_R(\boldsymbol{\theta}) = \mathbb{E}_{a \sim \mathcal{A}}\big[f_D(\boldsymbol{\theta}; a)\big], \tag{11}$$

where a denotes a transformation sampled from an attack distribution $\mathcal{A}$, and $f_D(\theta; a)$ is the detection score computed after applying attack a to the generated text. In practice, this expectation is approximated by averaging over multiple independently sampled attacks.

Pareto Optimization Objective. These three objectives are inherently conflicting: increasing watermark strength often improves detectability and robustness, but may degrade text quality. We therefore seek a Pareto-optimal frontier over the controller parameters $\boldsymbol{\theta}$ (see Appendix F). Because robustness evaluation is noisy, we approximate this frontier using multi-objective Bayesian optimization with Gaussian Process surrogates and q-Noisy Expected Hypervolume Improvement (qNEHVI) Daulton et al. (2021) as the acquisition criterion (see Appendix E). The search approximates a trade-off over the evaluated parameter family. Appendix E explains the role of optimization separately from the structural theory.

Table 1: Detection performance under different paraphrase attacks, evaluated using ROC-AUC / TP@FP=1% / TP@FP=5% (%). Higher is better (↑).

Dataset	Method	Detectability	Robustness					
		No Paraphrase↑	Parrot↑	Parrot-Bi↑	GPT-low↑	GPT-Med↑	GPT-High↑	Overall↑
Llama3.2-3B-Instruct								
C4	KGW	100/100/100	100/96/99	100/95/98	88/47/60	83/30/44	81/28/40	90.4/59.2/68.2
	Adaptive	100/100/100	98/87/95	98/87/95	96/65/87	92/55/68	91/48/69	95.0/68.4/82.8
	SIR	99/91/96	98/79/93	98/79/92	93/63/76	92/52/67	90/43/62	94.2/63.2/78.0
	SemStamp	100/96/99	59/10/22	60/11/26	84/32/47	80/25/38	75/18/31	71.6/19.2/32.8
	KSemStamp	98/56/97	59/10/26	58/10/26	85/13/46	80/9/39	79/9/42	72.2/10.2/35.8
	SemAdapt (ours)	100/100/100	100/94/98	100/95/97	97/67/76	90/55/63	88/52/63	**95.2/72.6/79.4**
LFQA	KGW	100/100/100	100/99/99	100/99/99	92/58/78	86/49/59	84/30/51	92.4/67.0/77.2
	Adaptive	100/100/100	97/85/89	97/87/89	98/76/87	94/58/70	93/52/65	95.8/71.6/80.0
	SIR	98/87/91	55/11/15	55/10/13	85/43/47	84/35/52	79/24/37	71.6/24.6/32.8
	SemStamp	98/90/93	58/14/14	57/13/13	79/26/36	74/28/34	73/19/27	68.2/20.0/24.8
	KSemStamp	100/84/98	56/6/17	56/6/16	86/28/55	82/22/49	80/23/45	72.0/17.0/36.4
	SemAdapt (ours)	100/100/100	100/99/99	100/99/99	96/76/87	92/58/70	88/47/60	**95.2/75.8/83.0**
Qwen3-4B-Instruct								
C4	KGW	100/99/99	99/94/98	99/89/97	94/54/78	88/37/53	87/31/52	95.4/61.0/75.6
	Adaptive	99/96/97	95/73/78	95/74/78	95/69/77	91/51/60	87/51/57	92.6/63.6/70.0
	SIR	92/60/76	83/38/50	85/42/58	79/33/46	77/27/45	75/19/44	79.8/31.8/48.6
	SemStamp	99/90/97	60/5/23	59/6/22	77/23/31	73/14/24	71/13/16	68.0/12.2/23.2
	KSemStamp	98/61/92	65/2/20	66/2/19	81/16/45	81/19/45	80/11/40	74.6/10.0/33.8
	SemAdapt (ours)	100/99/100	100/96/97	100/96/97	96/77/88	93/61/69	91/54/60	**96.0/76.8/82.2**
LFQA	KGW	100/99/100	99/89/97	99/89/98	85/23/53	79/15/38	77/10/39	87.8/45.2/65.0
	Adaptive	99/93/94	93/59/70	94/61/70	85/31/41	76/19/34	74/14/20	84.4/36.8/47.0
	SIR	93/48/74	80/27/48	79/22/49	73/14/35	69/9/33	66/7/27	73.4/15.8/38.4
	SemStamp	91/57/75	54/8/16	54/8/17	70/16/27	65/8/15	62/9/17	61.0/9.8/18.4
	KSemStamp	98/47/87	68/4/18	68/5/18	79/7/27	77/4/18	75/2/20	73.4/4.4/20.2
	SemAdapt (ours)	100/98/99	99/90/95	99/88/95	86/34/60	76/20/44	78/19/41	**87.6/50.2/67.0**

5 Experiments

5.1 Experimental Setup

Datasets and Models. We evaluate our method on two commonly used datasets: the RealNewsLike subset of the C4 corpus Raffel et al. (2020) and the LFQA dataset Krishna et al. (2023). For C4, we use the first 30 tokens of each document as the prompt and generate the subsequent 200 tokens, treating the original document continuations as the unwatermarked baseline; for LFQA, we use each question as the prompt to generate answers of up to 200 tokens, with the dataset-provided gold answers serving as the baseline reference. We use Llama-3.2-3B-Instruct Meta (2024) and Qwen-3-4B-Instruct Qwen (2025) as target models for text generation. We use sentence-transformers/all-MiniLM-L6-v2 sentence transformers (2022) to compute sentence embedding for the text quality objective. Representative examples of watermark model outputs are shown in Appendix N.

Evaluation Metrics. We evaluate watermark detectability and robustness using ROC-AUC, as well as true positive rates at fixed false positive rates of 1% and 5% (TP@FP=1%, TP@FP=5%). Output quality is assessed using perplexity. To evaluate robustness against meaning-preserving rewrites, we subject watermarked outputs to a diverse set of paraphrase attacks, including GPT-4o-mini paraphrasing with three strength levels (Low, Medium, and High), Parrot Prithivida (2022), and a bigram-based Parrot attack following Hou et al. (2024a). The three GPT-based levels allow us to evaluate watermark robustness under increasingly strong lexical and structural rewriting. Details of paraphrase settings and GPT prompts are provided in Appendix G.

Baselines. Based on the official implementation of MarkLLM Pan et al. (2024), we compare our method against five representative watermarking algorithms: KGW Kirchenbauer et al. (2023), SIR Liu et al. (2023), Adaptive Liu & Bu (2024), and two sentence-level watermarking methods, SemStamp Hou et al. (2024a) and KSemStamp Hou et al. (2024b). This set covers token-level, semantic, adaptive, and sentence-level watermarking paradigms. Baseline parameter settings are provided in Appendix H.

Table 2: Fixed-strength structure ablation on LFQA. Each entry reports ROC-AUC / TPR@1% FPR / TPR@5% FPR (%).

Method	Clean	GPT-Low	GPT-Medium	GPT-High	Parrot	Parrot-Bigram	DIPPER
KGW	100 / 100 / 100	92 / 40 / 68	85 / 23 / 49	82 / 16 / 42	99 / 94 / 98	99 / 94 / 98	88 / 69 / 82
Semantic Buckets	100 / 100 / 100	95 / 47 / **80**	90 / 27 / 56	87 / 20 / 47	**100** / **99** / **100**	**100** / **99** / **100**	**98** / 78 / **93**
+ Lexical Refinement	100 / 100 / 100	**96** / **57** / 78	**91** / **34** / **61**	**87** / **26** / **51**	**100** / 98 / **100**	**100** / 98 / 99	**98** / **83** / **93**

Implementation details. For Bucket-Level Watermark Injection, we use $d_{\text{vocab}} = 7$, yielding $B = 128$ semantic buckets. Controller parameters are optimized using MOBO with 40 iterations. At each iteration, a candidate parameter vector $\boldsymbol{\theta}$ is evaluated by averaging the objective values over 20 samples. Robustness is estimated by randomly applying GPT-Low paraphrasing attacks to the generated samples and averaging the corresponding detection performance. We set $\gamma = 0.5$. The offline MOBO optimization takes approximately 2 hours on 3×L40 GPUs in our implementation. Additional implementation details are provided in Appendix H.

5.2 Main Result

Table 1 reports detection performance under diverse paraphrase attacks across two datasets and two models. While most methods achieve strong detection on clean text, performance differences become more pronounced under paraphrasing, especially at low false-positive rates. `SemAdapt` shows consistently strong robustness, achieving the highest aggregated TP@FP=1% in the *Overall* column across all datasets and models.

Importantly, Parrot is not used during MOBO optimization and therefore serves as an unseen paraphrase attack where `SemAdapt` remains robust under this setting. Appendix L further evaluates generalization to the unseen DIPPER paraphraser. From a Pareto-optimization perspective across strength settings, `SemAdapt` also exhibits a stronger quality–detectability frontier than KGW, achieving higher TPR@1% FPR at comparable perplexity (Appendix J).

6 Analysis

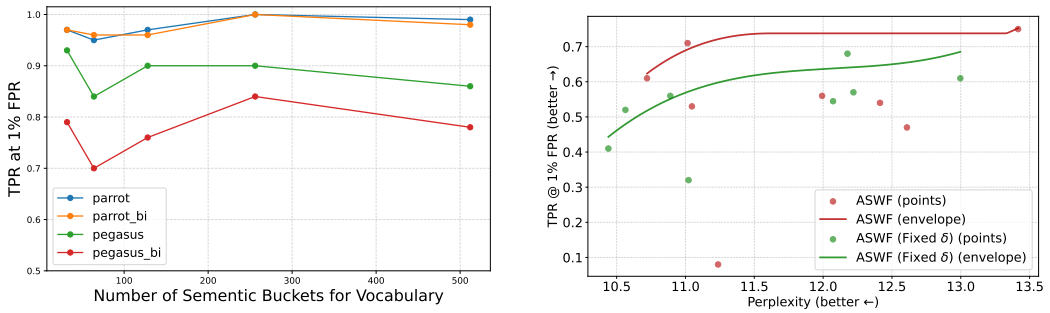
6.1 Ablation Study

Effect of Semantic Bucketing and Lexical Refinement. To isolate the contribution of the semantic bucket structure from adaptive strength control, we fix the watermark strength and progressively augment KGW with semantic bucket-level watermarking and lexical graph refinement. Table 2 shows that replacing token-level assignments with semantic buckets consistently improves paraphrase robustness over KGW while preserving perfect clean detection. Applying lexical refinement on top of the bucket-level variant provides additional gains, particularly at the stringent 1% FPR operating point. These results separate the two structural effects: semantic bucketing provides the main robustness improvement over token-level watermarking, while lexical refinement further strengthens the resulting bucket structure.

Appendix K further validates the embedding-neighborhood restriction used to construct lexical candidate pairs. Observed lexical substitutions are substantially more concentrated in these neighborhoods than shuffled controls, supporting their use as candidate relations for refinement.

Number of Semantic Buckets for Vocabulary. We examine whether the detection performance of `SemAdapt` is sensitive to the number of semantic buckets used for vocabulary partitioning. Specifically, we vary the number of vocabulary semantic buckets B while keeping all other components fixed, and evaluate detection performance using TP@FP at 1% false-positive rate. As shown in Fig. 2(a), detection performance remains largely stable across different choices of B, with only minor fluctuations and no clear dependence on the bucket number. This suggests that `SemAdapt` is robust to the choice of vocabulary bucket count.

Effect of MOBO. Fig. 2(b) compares `SemAdapt` with a fixed-strength variant that uses the same semantic watermarking structure but removes MOBO. Each point corresponds to one evaluated



(a) Sensitivity to B

(b) MOBO vs. fixed δ

Figure 2: Ablation studies of `SemAdapt`. Left: `SemAdapt` remains stable across different numbers of semantic vocabulary buckets. Right: the MOBO-optimized controller achieves a stronger quality–detectability trade-off than the fixed-strength variant under Paraphrase attack.

configuration of `SemAdapt` or one fixed δ setting of the variant. We plot the empirical envelope in the plane of perplexity (lower is better) and TP@FP at 1% FPR (higher is better). `SemAdapt` achieves a stronger frontier, attaining higher true-positive rates at comparable perplexity or comparable detection performance with lower perplexity. This shows that MOBO contributes beyond the semantic bucket structure by identifying controller parameters with better quality–detectability trade-offs.

6.2 Analysis of learned δ.

To interpret the learned strength allocation, we group generation positions by the UPOS tag of the preceding token. As shown in Fig. 3, δ_t varies substantially across linguistic contexts rather than remaining constant. Determiner-preceded positions exhibit the highest and most concentrated strengths, while PROPN and AUX show lower medians and broader variation. Since δ_t increases monotonically with bucket entropy, these patterns reflect systematic differences in bucket-level uncertainty across contexts.

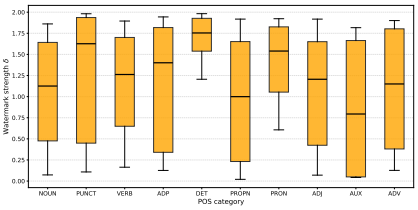


Figure 3: Distribution of adaptive watermark strength δ_t grouped by the UPOS tag of the preceding token.

6.3 Robustness Beyond Paraphrasing

Beyond standard paraphrasing, we further evaluate `SemAdapt` under broader text edits and spoofing attacks. Specifically, we test sentence-level perturbations, including sentence deletion, reordering, splicing, and back-translation, which modify the global structure of the text. We also examine black-box spoofing attacks, where an attacker attempts to generate non-watermarked text that is nevertheless detected as watermarked. As reported in Appendix I, `SemAdapt` remains competitive under sentence-level perturbations and shows lower spoofing leakage than KGW in the evaluated watermark-stealing setting.

7 Conclusion

We present `SemAdapt`, an adaptive semantic watermarking method for paraphrase-robust LLM watermarking. `SemAdapt` combines a lexically refined semantic vocabulary partition with bucket-level watermarking and adapts watermark strength using uncertainty over the same bucket space. Controller parameters are selected offline via multi-objective Bayesian optimization to balance detectability, paraphrase robustness, and text quality. Experiments and controlled ablations show that semantic bucketing provides the main robustness gain over token-level watermarking, while lexical refinement and adaptive strength control further improve the resulting robustness–quality trade-off.

AI USE STATEMENT

Generative AI tools were used to polish the manuscript and code for clarity and readability. All AI-assisted edits were reviewed by the authors, who take full responsibility for the final content.

REFERENCES

Isabelle Augenstein, Timothy Baldwin, Meeyoung Cha, Tanmoy Chakraborty, Giovanni Luca Ciampaglia, David Corney, Renee DiResta, Emilio Ferrara, Scott Hale, Alon Halevy, et al. Factuality challenges in the era of large language models and opportunities for fact-checking. *Nature Machine Intelligence*, 6(8):852–863, 2024.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pp. 610–623, 2021.

Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pp. 380–388, 2002.

Amirhossein Dabiriaghdam and Lele Wang. Simmark: A robust sentence-level similarity-based watermarking algorithm for large language models. *arXiv preprint arXiv:2502.02787*, 2025.

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.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. *Advances in neural information processing systems*, 33:9851–9864, 2020.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement. *Advances in neural information processing systems*, 34:2187–2200, 2021.

Michael TM Emmerich, Kyriakos C Giannakoglou, and Boris Naujoks. Single-and multiobjective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, 2006.

Yuxuan Guo, Zhiliang Tian, Yiping Song, Tianlun Liu, Liang Ding, and Dongsheng Li. Context-aware watermark with semantic balanced green-red lists for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22633–22646, 2024.

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)*, pp. 4067–4082, 2024a.

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*, pp. 1706–1715, 2024b.

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. *arXiv preprint arXiv:2509.21057*, 2025.

Mingjia Huo, Sai Ashish Somayajula, Youwei Liang, Ruisi Zhang, Farinaz Koushanfar, and Pengtao Xie. Token-specific watermarking with enhanced detectability and semantic coherence for large language models. *arXiv preprint arXiv:2402.18059*, 2024.

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*, pp. 604–613, 1998.

Nikola Jovanović, Robin Staab, and Martin Vechev. Watermark stealing in large language models. *arXiv preprint arXiv:2402.19361*, 2024.

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*, pp. 17061–17084. PMLR, 2023.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36:27469–27500, 2023.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. Who wrote this code? watermarking for code generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4890–4911, 2024.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. *arXiv preprint arXiv:2310.06356*, 2023.

Yepeng Liu and Yuheng Bu. Adaptive text watermark for large language models. *arXiv preprint arXiv:2401.13927*, 2024.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. An entropy-based text watermarking detection method. *arXiv preprint arXiv:2403.13485*, 2024.

Meta. meta-llama/llama-3.2-3b-instruct. `https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct`, 2024.

OpenAI. gpt-4o-mini. `https://platform.openai.com/docs/models/gpt-4o-mini`, 2024.

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

Prithivida. Parrot: Paraphraser based on t5, 2022. `https://huggingface.co/prithivida/parrot_paraphraser_on_T5`.

Qwen. Qwen/qwen3-4b-instruct-2507. `https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507`, 2025.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

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. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 613–625, 2024.

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.

sentence transformers. sentence-transformers/all-minilm-l6-v2. `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`, 2022.

Chenrui Wang, Junyi Shu, Billy Chiu, Yu Li, Saleh Alharbi, Min Zhang, and Jing Li. Learning to watermark: A selective watermarking framework for large language models via multi-objective optimization. *arXiv preprint arXiv:2510.15976*, 2025.

Yihan Wu, Zhengmian Hu, Hongyang Zhang, and Heng Huang. Dipmark: A stealthy, efficient and resilient watermark for large language models. 2023.

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

A GENERATION AND DETECTION ALGORITHMS

Algorithms 1 and 2 summarize the bucket-level watermark generation and detection procedures used by `SemAdapt`. The semantic vocabulary mapping b_G and controller parameters $\theta^\star = (\alpha^\star, \delta^\star_{\text{base}})$ are fixed before generation.

Algorithm 1 Bucket-Level Watermark Generation with `SemAdapt`

Require: Language model P; tokenized prompt x; refined mapping b_G; secret key K; context length n; bucket count B; green-bucket count m; controller parameters $\theta^\star = (\alpha^\star, \delta^\star_{\text{base}})$; maximum generation length T
Ensure: Watermarked continuation $y_{1:T}$

$h \leftarrow x$
$Y \leftarrow [\,]$
for $t = 1, \ldots, T$ **do**
  Obtain base logits $z_t(w)$ for all $w \in \mathcal{V}$ conditioned on history h
  $p_t(w) \leftarrow \mathrm{softmax}(z_t)(w)$
  $c_t \leftarrow$ the most recent n tokens in h
  $G_t \leftarrow \mathcal{G}_K(c_t)\ \{|G_t| = m\}$
  for $j = 0, \ldots, B-1$ **do**
    $P_t^B(j) \leftarrow \sum_{w:\, b_G(w)=j} p_t(w)$
  end for
  $H_t^B \leftarrow -\sum_{j=0}^{B-1} P_t^B(j) \log P_t^B(j)$
  $\delta_t \leftarrow \delta^\star_{\text{base}} \left(1 - \exp(-\alpha^\star H_t^B)\right)$
  for each $w \in \mathcal{V}$ **do**
    $I_t(w) \leftarrow \mathbb{I}[b_G(w) \in G_t]$
    $\widetilde{z}_t(w) \leftarrow z_t(w) + \delta_t I_t(w)$
  end for
  Sample y_t from the decoder applied to $\widetilde{z}_t$
  Append y_t to Y and h
  if the decoder stopping condition is satisfied **then**
    break
  end if
end for
return Y

Algorithm 2 Bucket-Level Watermark Detection with `SemAdapt`

Require: Candidate sequence $y_{1:T}$ with its preceding context; refined mapping b_G; secret key K; context length n; bucket count B; green-bucket count m; empirically calibrated threshold τ
Ensure: Detection statistic Z and watermark decision $\widehat{\mathrm{wm}}$

$\gamma \leftarrow m/B$
$S \leftarrow 0$
for $t = 1, \ldots, T$ **do**
  $c_t \leftarrow y_{t-n:t-1}$
  $G_t \leftarrow \mathcal{G}_K(c_t)$
  $I_t \leftarrow \mathbb{I}[b_G(y_t) \in G_t]$
  $S \leftarrow S + (I_t - \gamma)$
end for
$Z \leftarrow \dfrac{S}{\sqrt{T\gamma(1-\gamma)}}$
$\widehat{\mathrm{wm}} \leftarrow \mathbb{I}[Z > \tau]$
return $(Z, \widehat{\mathrm{wm}})$

B Lexical Graph Construction

We represent every undirected candidate edge canonically as (u, v) with $u < v$, and define its pooled support as $s_{uv} = n(u \to v) + n(v \to u)$. Algorithm 3 gives the deterministic greedy selection procedure used to construct the matching M.

Algorithm 3 Greedy Lexical Matching

Require: Candidate edges $\mathcal{E}$ represented as (u, v) with $u < v$, confidence r_{uv}, pooled support s_{uv}, edge budget L
Ensure: Matching M

```
Sort E lexicographically by (−r_uv, −s_uv, u, v).
M ← ∅
for each (u, v) ∈ E in sorted order do
    if u and v are both unmatched in M then
        M ← M ∪ {(u, v)}
    end if
    if |M| = L then
        break
    end if
end for
return M
```

The confidence score r_{uv} therefore determines the primary ordering, while pooled support and token IDs provide deterministic tie-breaking. Accepting only edges with two unmatched endpoints ensures $\deg_M(w) \leq 1$ for every token and prevents selected lexical relations from forming transitive chains.

C Minimum-Change Bucket Refinement

We formalize the minimum-change refinement introduced in Section 4.2. Fix the initial mapping b_0, a matching M over the vocabulary, and nonnegative source-frequency weights $\{\pi_w\}_{w\in\mathcal{V}}$ satisfying $\sum_w \pi_w = 1$. Let $\mathcal{F}_M$ denote the set of mappings b such that $b(u) = b(v)$ for every $\{u, v\} \in M$. We impose no constraint on bucket sizes. For any $b \in \mathcal{F}_M$, define the frequency-weighted moved mass

$$C_\pi(b) = \sum_{w\in\mathcal{V}} \pi_w \mathbb{I}[b(w) \neq b_0(w)]. \tag{12}$$

Proposition 1 (Minimum moved mass)**.** *For each $\{u, v\} \in M$ with $b_0(u) \neq b_0(v)$, assign both endpoints to the initial bucket of the higher-weight endpoint, using a fixed tie-breaking rule when $\pi_u = \pi_v$. Leave pairs that already share a bucket and unmatched tokens unchanged. The resulting mapping b_G satisfies*

$$C_\pi(b_G) = \min_{b\in\mathcal{F}_M} C_\pi(b) = \sum_{\substack{\{u,v\}\in M \\ b_0(u)\neq b_0(v)}} \min\{\pi_u, \pi_v\}. \tag{13}$$

Proof. Consider an edge $\{u, v\} \in M$ whose endpoints initially belong to different buckets. Any feasible mapping must move at least one endpoint in order to satisfy $b(u) = b(v)$, incurring cost at least $\min\{\pi_u, \pi_v\}$. Because M is a matching, its edges are disjoint, so these pairwise lower bounds add independently. Moving only the lower-weight endpoint attains the lower bound for every such pair. Pairs whose endpoints already share a bucket and unmatched tokens incur zero cost. Hence the construction in Proposition 1 achieves the global minimum. Equal or zero weights do not affect the argument, although the minimizer need not be unique. □

Define the resulting minimum moved mass as

$$\varepsilon_\pi := C_\pi(b_G) = \sum_{\substack{\{u,v\}\in M \\ b_0(u)\neq b_0(v)}} \min\{\pi_u, \pi_v\}. \tag{14}$$

Thus, for the fixed matching M, the refinement enforces the selected lexical relations while moving the smallest possible source-frequency mass away from the initial partition.

Effect on bucket mass and green probability. Let

$$D = \{w \in \mathcal{V} : b_G(w) \neq b_0(w)\}$$

denote the set of tokens whose bucket assignments change. For any token distribution p, define

$$\varepsilon_p = \sum_{w \in D} p(w), \qquad P_{b,j} = \sum_{w : b(w) = j} p(w).$$

Then

$$\begin{aligned} \frac{1}{2} \sum_j |P_{b_G,j} - P_{b_0,j}| &\leq \varepsilon_p, \\ \mathbb{E}_{w \sim p} \, |\mathbb{I}[b_G(w) \in G] - \mathbb{I}[b_0(w) \in G]| &\leq \varepsilon_p \end{aligned} \tag{15}$$

for any fixed set of green buckets G shared by the two mappings. The first inequality follows by coupling the bucket assignments through the same sampled token: the two mappings differ only when $w \in D$. The second follows from the same observation pointwise. Consequently,

$$\left| \Pr_{w \sim p}[b_G(w) \in G] - \Pr_{w \sim p}[b_0(w) \in G] \right| \leq \varepsilon_p. \tag{16}$$

Taking $p = \pi$ connects the minimum-change objective in Eq. equation 13 to its distributional effect: among mappings satisfying the selected lexical constraints, b_G minimizes an upper bound on the perturbation induced under the construction-source distribution. This guarantee is specific to the fixed matching M and the weighting distribution π. In particular, a small ε_π does not imply that ε_{p_t} is small for every conditional next-token distribution p_t, nor does the result directly bound generation KL divergence or perplexity.

D Null Behavior and Calibration

Reference model. Let $B \geq 2$, $1 \leq m < B$, and $T \geq 1$. Freeze b_G and a key-independent text with T scored positions. For each distinct context input c, independently draw a uniform m-of-B green set, and reuse it on repeated inputs; write $\gamma = m/B$. Randomness here belongs to this ideal mask family, independently of the frozen map. It does not describe changing the shared initialization and masking key in the implementation.

Proposition 2 (Null mean and variance)**.** *Let $n_{c,j}$ count scored positions with context c and bucket j, and $n_c = \sum_j n_{c,j}$. Under the reference model,*

$$\mathbb{E}[Z] = 0, \qquad \mathrm{Var}(Z) = \frac{\sum_c \left(B \sum_j n_{c,j}^2 - n_c^2 \right)}{(B-1)T}. \tag{17}$$

Proof. Each fixed bucket is green with probability γ, regardless of its size. Under one context, two indicators have covariance $\gamma(1-\gamma)$ if their buckets agree and $-\gamma(1-\gamma)/(B-1)$ otherwise, by sampling without replacement. Different context rows are independent. Grouping the total hit count by context and bucket gives its variance as $\gamma(1-\gamma) \sum_c (B \sum_j n_{c,j}^2 - n_c^2)/(B-1)$. Centering and dividing by $T\gamma(1-\gamma)$ proves the statement. □

Effect of GraphVocab reassignment. Although GraphVocab preserves the marginal green probability under the reference model, reassignment can change the bucket counts $n_{c,j}$ associated with repeated contexts and hence the variance in Eq. equation 17. Consider moving one token type from bucket a to a different bucket b, and let q_c denote its number of occurrences following context c. If $n_{c,a}$ and $n_{c,b}$ are the bucket counts before the move, the resulting change in nominal-score variance is

$$\Delta \, \mathrm{Var}(Z) = \frac{2B}{(B-1)T} \sum_c q_c \big(n_{c,b} - n_{c,a} + q_c \big). \tag{18}$$

The change can be either positive or negative. Thus, GraphVocab preserves the nominal null mean under the reference model, but does not necessarily preserve its variance or tail behavior.

Empirical calibration. We therefore calibrate each mapping separately using $2{,}000$ length-matched null texts. For each target false-positive rate, we determine the detection threshold from the empirical upper tail of the corresponding null scores and use the same calibration protocol for all compared methods. Our reported comparisons therefore rely on empirically calibrated operating points rather than the nominal standard-normal approximation.

E Bayesian Optimization and Multi-Objective Bayesian Optimization

As explained in Sec.4.4, our method involves jointly optimizing multiple objectives, including detection performance f_D, generation quality f_Q, and robustness under stochastic attacks f_R, over a shared parameter space. To this end, we adopt Multi-Objective Bayesian Optimization (MOBO)Emmerich et al. (2006) as a principled framework for exploring such expensive, noisy, and conflicting objectives. For completeness, we first briefly review Bayesian Optimization (BO) in the single-objective setting, and then introduce its multi-objective extension used in our method.

E.1 Bayesian Optimization

Bayesian Optimization (BO) is a sequential framework for optimizing an expensive black-box objective function

$$f : \mathcal{X} \to \mathbb{R},$$

where evaluations of $f(x)$ are costly, noisy, and do not admit analytic gradients. BO aims to efficiently identify high-performing inputs by leveraging a probabilistic surrogate model rather than exhaustive or gradient-based search.

At iteration t, BO assumes access to a set of noisy observations $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{t}, y_i = f(x_i) + \epsilon_i$, where ϵ_i denotes observation noise. A Gaussian Process (GP) prior is placed over the objective, $f \sim \mathcal{GP}\big(m(x), k(x, x')\big)$, which induces a posterior predictive distribution at any candidate point $x \in \mathcal{X}$, $p\big(f(x) \mid \mathcal{D}_t\big) = \mathcal{N}\big(\mu_t(x), \sigma_t^2(x)\big)$.

The GP posterior provides both a mean estimate $\mu_t(x)$ and an uncertainty estimate $\sigma_t(x)$, enabling principled exploration–exploitation trade-offs. Based on this posterior, BO defines an acquisition function $\alpha_t(x) = \alpha\big(\mu_t(x), \sigma_t(x)\big)$, which quantifies the utility of evaluating the objective at x. The next query point is selected by solving $x_{t+1} = \arg\max_{x \in \mathcal{X}} \alpha_t(x)$.

By iteratively alternating between surrogate model updates and acquisition-driven sampling, BO efficiently explores the input space and concentrates evaluations in promising regions, even under limited evaluation budgets and noisy observations. However, standard BO is inherently limited to optimizing a single scalar objective, and does not directly account for trade-offs among multiple competing criteria.

E.2 Multi-Objective Bayesian Optimization

Multi-Objective Bayesian Optimization (MOBO) extends BO to settings where the objective is vector-valued,

$$\mathbf{f}(x) = \big(f_1(x), \ldots, f_m(x)\big), \mathbf{f} : \mathcal{X} \to \mathbb{R}^m,$$

and the objectives may be conflicting. Each objective evaluation is assumed to be noisy, $y_{i,j} = f_j(x_i) + \epsilon_{i,j}, j = 1, \ldots, m$, with independent or correlated observation noise.

Instead of seeking a single optimum, MOBO aims to approximate the Pareto frontier, defined as the set of non-dominated solutions for which no objective can be improved without degrading at least one other objective. As in BO, each objective f_j is modeled using a GP surrogate, yielding a joint posterior over the objective vector $\mathbf{f}(x)$.

To guide the search toward the Pareto frontier, MOBO employs acquisition functions that explicitly account for multi-objective trade-offs. In this work, we adopt q-Expected Hypervolume Improvement (qNEHVI) as the acquisition strategy. qNEHVI selects candidate points by maximizing the expected increase in dominated hypervolume under the surrogate posterior, $\alpha_{\text{qNEHVI}}(X) = \mathbb{E}\big[\Delta\mathcal{H}(X)\big]$, where

$\mathcal{H}$ denotes the hypervolume with respect to a reference point and the expectation is taken over the GP posterior. The batch of candidate points X is chosen by maximizing this acquisition function.

By jointly modeling uncertainty across objectives and explicitly reasoning about Pareto dominance, GP-based MOBO with qNEHVI provides an effective framework for exploring complex, noisy, and expensive multi-objective optimization problems, and serves as the optimization backbone of our proposed method.

F Pareto Optimality

In multi-objective optimization, the goal is to reason about trade-offs among multiple competing objectives rather than to optimize a single scalar value. Let $\mathbf{f}(x) = \big(f_1(x), \ldots, f_m(x)\big) \in \mathbb{R}^m$ denote a vector-valued objective function defined over the decision space $\mathcal{X}$. Given two candidate solutions $x, x' \in \mathcal{X}$, we say that x **Pareto-dominates** x', denoted $x \succ x'$, if

$$\forall j \in \{1, \ldots, m\}, \quad f_j(x) \geq f_j(x') \quad \text{and} \quad \exists j \text{ s.t. } f_j(x) > f_j(x'). \tag{19}$$

That is, x is no worse than x' in all objectives and strictly better in at least one.

A solution $x^\star \in \mathcal{X}$ is said to be **Pareto-optimal** if there exists no other solution $x \in \mathcal{X}$ such that $x \succ x^\star$. The set of all Pareto-optimal solutions forms the **Pareto-optimal set**

$$\mathcal{P} = \{x \in \mathcal{X} \mid \nexists\, x' \in \mathcal{X} \text{ s.t. } x' \succ x\}.$$

The image of $\mathcal{P}$ in the objective space,

$$\mathcal{F} = \{\mathbf{f}(x) \mid x \in \mathcal{P}\},$$

is referred to as the **Pareto frontier**, which characterizes the achievable trade-offs among objectives.

G Paraphrase Attack Settings

We evaluate robustness under a diverse set of paraphrase attacks, including GPT-based paraphrasing with different rewriting strengths, Parrot Prithivida (2022), and a bigram-based Parrot attack. The GPT-based paraphrases are generated using GPT-4o-mini OpenAI (2024) with three strength levels: Low, Medium, and High. These levels are designed to control the intensity of the attack, ranging from light lexical edits to substantial wording and structural changes. For Parrot Prithivida (2022), we apply beam search with `num_beams` $= 10$.

For GPT-4o-mini, we use a fixed system prompt and a structured user prompt. The strength guide is selected according to the attack level: Low uses "Lightly rephrase with minimal changes," Medium uses "Rephrase with noticeable lexical and syntactic variation," and High uses "Strongly rephrase with substantial wording and structure changes." The prompt template is shown below.

System prompt:
You are an expert at paraphrasing text. Output only the paraphrased text without any preamble or explanation.

User prompt:
Paraphrase this text while keeping the same meaning. {strength guide}
Rules:

- Keep approximately the same length ({number of words} words)
- Maintain all facts and details
- Output ONLY the paraphrased text, no extra commentary
- Do not truncate or shorten the response

Original text:
{original text}
Paraphrased version:

Following Hou et al. (2024a), we additionally include a bigram-based paraphrase attack designed to maximally disrupt token-level statistics while preserving semantic content. For a given input, multiple paraphrase candidates are generated using Parrot beam search. Each candidate is evaluated based on: (i) bigram overlap with the original text, (ii) semantic similarity measured by BERTScore, and (iii) output length. Among candidates whose BERTScore lies within a relative threshold of the maximum score, we select the one with the lowest bigram overlap. For Parrot, the threshold is set to 0.03 of the maximum BERTScore among candidates.

H Implementation Details

We use the official implementation of MarkLLM Pan et al. (2024) to reproduce all baseline results. All methods use the same sampling configuration during generation, with temperature set to 0.85, top-p set to 0.95, and top-k set to 50. For baselines that rely on external embedding models (e.g., SIR Liu et al. (2023)), we use the official pretrained weights provided in MarkLLM. For baselines that involve watermark parameters such as γ and n-gram size, we standardize the settings to n-gram $= 2$ and $\gamma = 0.25$ across all methods. The parameter δ has different interpretations across watermarking schemes; following the official implementations, we adjust δ for each baseline to achieve comparable text quality, ensuring a fair comparison.

For MOBO configuration, we adopt Gaussian Process (GP) surrogate models and use q-Expected Hypervolume Improvement (qNEHVI) as the acquisition function. Monte Carlo integration is performed using 512 samples, with 128 samples used for acquisition estimation. The optimization runs for 40 iterations, preceded by 12 randomly sampled initial configurations. For acquisition optimization, we use 20 random restarts. Each candidate parameter vector is evaluated on 20 prompts. During optimization and evaluation, we generate up to 200 tokens per prompt using temperature 0.8 and top-p sampling with $p = 0.95$. For the adaptive controller, key parameters are searched within bounded ranges, including $\alpha_{\text{tight1}} \in [0.8, 2.0]$ and $\delta_{\text{base}} \in [1.0, 2.0]$.

Graph construction. For each construction dataset (C4 or LFQA), we use $2{,}000$ source texts with low- and medium-strength paraphrases generated by `gpt-4o-mini` at temperature 0.1. Construction prompts are excluded from the evaluation and null pools. Candidate lists contain the top 32 neighbors by cosine similarity of normalized model input embeddings. The greedy matching selects 974 edges with maximum degree one, without an additional confidence cutoff. Anchor frequencies are computed from the original construction texts; ties favor the smaller token ID.

Null evaluation. For each dataset–model combination, we use $2{,}000$ clean, unwatermarked model outputs as negatives. These prompts are excluded from graph construction. Methods share the null texts within each dataset–model combination, with scores computed separately for each detector and mapping. We report ROC-AUC and the largest empirical TPR at FPR at most 1% or 5%, treating tied scores together. The same clean null pool is used for original and attacked positives.

I Additional Robustness Evaluation

This section provides additional robustness evaluations beyond the paraphrase attacks in the main experiments. We consider two complementary settings: sentence-level perturbations, which modify the structure or content coverage of the generated text, and spoofing attacks, which test whether the watermark design can be easily forged by a black-box attacker.

I.1 Sentence-Level Perturbations

We first evaluate robustness under sentence-level perturbations, including sentence deletion, sentence reordering, sentence splicing, and back-translation. These edits directly stress sentence-level watermarking assumptions: deletion removes sentence-level carriers, reordering breaks positional consistency, and splicing mixes watermarked and non-watermarked sentence units. This evaluation therefore examines whether watermark signals can survive when the sentence structure used by many sentence-level schemes is altered.

Table 3: Robustness under sentence-level perturbations. Each entry reports ROC-AUC / TP@FP=1% / TP@FP=5%. Higher is better.

Method	Deletion	Reordering	Splicing	Back-Trans.
KGW	62/8/19	74/19/32	66/9/24	96/69/86
SemStamp	44/5/7	41/2/3	47/6/8	45/3/8
SemAdapt	70/18/25	77/36/45	72/17/31	95/79/87

Table 4: Spoofing results under a black-box watermark-stealing attack. We report original watermark detectability, perplexity of spoofed text, and spoofing leakage measured by TP@FP=1% / TP@FP=5%. Lower spoofing leakage is better.

Method	Detectability ↑	PPL ↓	Spoofing Leakage ↓
KGW ($\delta = 1.7$)	100	16.3	71 / 87
KGW ($\delta = 1.8$)	100	15.9	56 / 79
KGW ($\delta = 1.9$)	100	17.0	74 / 88
KGW ($\delta = 2.0$)	100	17.5	82 / 89
SemAdapt ($B = 128$)	100	14.5	48 / 62
SemAdapt ($B = 256$)	100	15.4	48 / 62
SemAdapt ($B = 512$)	100	14.2	53 / 66
SemAdapt ($B = 1024$)	100	15.9	39 / 44

As shown in Table 3, sentence-level perturbations substantially weaken all methods, especially under deletion and splicing where watermark-bearing spans may be removed or mixed with unrelated text. SemStamp is the most vulnerable, indicating that sentence-level watermark carriers are easily disrupted when sentence boundaries or ordering change. In contrast, SemAdapt achieves the best or competitive performance across all perturbations, with clear gains under reordering and splicing. This suggests that semantic bucket-level signals are less tied to exact sentence structure than sentence-level carriers, and can remain more stable under broader structural edits.

I.2 Spoofing Resistance

We next evaluate whether SemAdapt introduces additional vulnerability to spoofing attacks, where an attacker attempts to generate non-watermarked text that is nevertheless detected as watermarked. We consider a common black-box watermark-stealing setting: the attacker may know the overall watermarking design and observe watermarked outputs, but does not know the secret key, the LSH seed, the context-conditioned green bucket assignments, or detector feedback such as p-values or z-scores.

Following Jovanović et al. (2024), we estimate token-frequency statistics from black-box watermarked outputs and use them to construct a spoofing generator. We compare SemAdapt with KGW under strong watermark settings where both methods achieve nearly perfect detectability on original watermarked text. To make the comparison meaningful, we report spoofing leakage at similar text quality, measured by perplexity. For KGW, we vary the watermark strength δ; for SemAdapt, we vary the number of semantic buckets B.

As shown in Table 4 and Figure 4, SemAdapt yields lower spoofing leakage than KGW at similar or lower perplexity. KGW obtains spoofing leakage between $56/79$ and $82/89$, whereas SemAdapt ranges from $39/44$ to $53/66$. This suggests that SemAdapt does not make watermark stealing easier under the evaluated black-box setting. One possible explanation is that SemAdapt exposes a less globally stable frequency signal. KGW applies a relatively fixed token-level green-list bias, which can be partially recovered from token-frequency statistics. In contrast, SemAdapt uses an adaptive watermark strength controller, producing position-dependent bias strengths across different contexts. This heterogeneity reduces the stable frequency patterns available to spoofing attacks. We further observe that sharper and more heterogeneous bias distributions tend to yield lower spoofing leakage,

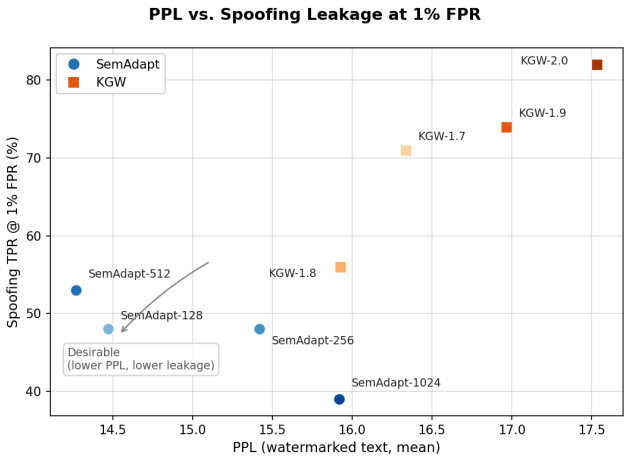


Figure 4: Spoofing leakage versus text quality under the black-box watermark-stealing attack. We compare `SemAdapt` and KGW using perplexity and spoofing leakage measured by TP@FP=1%. Lower perplexity and lower spoofing leakage are preferred.

Table 5: Controlled ablation of semantic structure and MOBO on Llama+C4. Detection results report TPR at 1% FPR (%) under three GPT-based paraphrase attacks. Values are rounded to the nearest integer.

Method	PPL ↓	GPT-Low ↑	GPT-Medium ↑	GPT-High ↑
KGW	16.15	78	59	53
KGW + MOBO	15.32	75	62	54
`SemAdapt` w/o MOBO	15.66	**80**	**67**	58
`SemAdapt`	**14.88**	78	**67**	**60**

suggesting that concentrating watermark strength on selected high-confidence positions can make frequency-based spoofing less effective.

J Component Analysis and Comparison with KGW

We analyze the contributions of the semantic bucket structure and MOBO-based parameter optimization, and further compare the resulting trade-off with KGW. To separate the contribution of the semantic bucket structure from that of MOBO-based parameter optimization, we conduct a controlled 2×2 ablation. In addition to standard KGW and `SemAdapt`, we equip KGW with MOBO-based adaptive strength control and evaluate `SemAdapt` without MOBO.

As shown in Table 5, the semantic bucket structure already improves paraphrase robustness over KGW without MOBO, particularly under medium- and high-strength attacks. Adding MOBO to KGW mainly reduces perplexity while leaving robustness broadly comparable. Similarly, applying MOBO to `SemAdapt` reduces perplexity from 15.6 to 14.8 while preserving comparable robustness. These results indicate complementary roles: the semantic bucket structure provides the main robustness gain, whereas MOBO improves the resulting quality–robustness trade-off.

Figure 5 further compares the two methods across multiple strength settings rather than at a single operating point. Over the overlapping perplexity range, `SemAdapt` achieves higher TPR at 1% FPR than KGW at comparable text quality, or lower perplexity at comparable detection performance. This indicates that its advantage persists across the evaluated quality–detectability trade-off rather than being tied to one selected configuration.

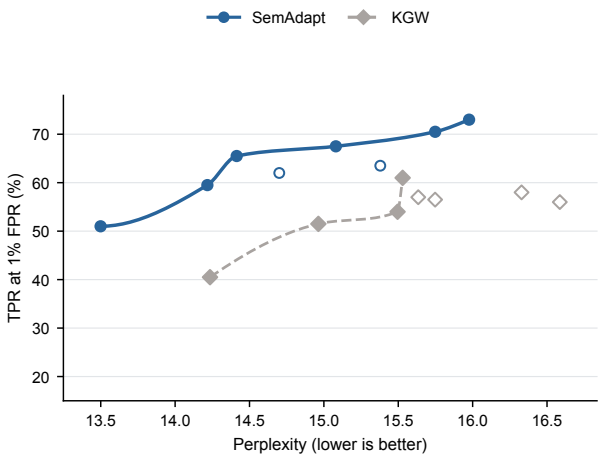


Figure 5: Quality–detectability trade-off between `SemAdapt` and KGW on Llama+C4 under GPT-Low paraphrasing. Each point represents an evaluated strength setting, and the curves show the corresponding empirical frontiers. Lower perplexity and higher TPR at 1% FPR are preferred.

Table 6: Coverage of observed lexical substitutions and shuffled controls by embedding neighborhoods. Higher values indicate stronger alignment between embedding proximity and observed substitution relations.

Metric	**Observed**	**Shuffled**
Exact-token coverage ($k = 16$)	32.1%	0.8%
Neighbor-bucket coverage ($k = 16$)	42.1%	11.5%
Neighbor-bucket coverage ($k = 32$)	59.9%	19.5%

K Validation of Lexical Candidate Neighborhoods

We evaluate whether the embedding neighborhoods used to construct the candidate sets $\mathcal{C}_k(u)$ capture lexical substitutions that actually occur under paraphrasing. This analysis is performed offline using existing source–paraphrase pairs. We restrict the evaluation to aligned positions where the token changes while the watermark context remains unchanged, so that the diagnostic focuses on lexical replacement rather than changes in the context-dependent watermark assignment.

For each source token u, we retrieve its nearest neighbors under cosine similarity of the generation model's input embeddings, excluding the source token itself and special tokens. As a control, we randomly permute the rewritten target tokens within each evaluation subset while keeping the source tokens and their embedding neighborhoods fixed. This preserves the empirical source- and target-token frequencies but removes the observed correspondence between each source token and its lexical substitute.

We consider two complementary notions of neighborhood coverage. *Exact-token coverage* measures whether the observed substitute itself appears among the top-k nearest neighbors of the source token. *Neighbor-bucket coverage* measures whether the semantic bucket containing the substitute is represented by at least one token in the source token's top-k neighborhood. Results pool GPT-Low and GPT-Medium substitutions and are averaged across four generation-strength settings.

Observed lexical substitutions are substantially more concentrated in the source token's embedding neighborhood than the shuffled controls. For example, the exact substitute appears among the top-16 neighbors in 32.1% of observed cases, compared with only 0.8% after shuffling. At the bucket level, coverage reaches 59.9% with $k = 32$, compared with 19.5% for the shuffled control. These

Table 7: Generalization to the unseen DIPPER paraphraser. Each entry reports ROC-AUC / TPR at 5% FPR. The SemAdapt controller is optimized without DIPPER and is kept fixed for all test settings.

Method	PPL ↓	L20-O0	L40-O0	L40-O20	L60-O20	L60-O40	L60-O60
KGW	15.7	0.98 / 0.98	0.96 / 0.92	0.97 / 0.90	0.92 / **0.81**	0.92 / 0.73	0.90 / 0.70
SemAdapt	**15.0**	**1.00 / 1.00**	**0.99 / 0.96**	**0.98 / 0.95**	**0.94** / 0.76	**0.94 / 0.77**	**0.93 / 0.74**

Table 8: Semantic-bucket stability for aligned lexical substitutions. Substitutions are grouped by semantic similarity. Higher co-bucket rate and green-hit survival, and lower Hamming distance, indicate greater stability under meaning-preserving lexical changes.

Substitution similarity	Co-bucket rate ↑	Hamming dist. ↓	Green-hit survival ↑
Low	1.72%	3.48	25.98%
Medium	1.18%	2.87	25.27%
High	**13.07%**	**2.35**	**34.97%**

results support embedding neighborhoods as a useful restriction when constructing candidate lexical relations.

This diagnostic evaluates only the candidate-neighborhood restriction. It does not by itself establish the benefit of the subsequent confidence ranking, greedy lexical matching, or minimum-change bucket refinement.

L Generalization to Unseen Paraphrase Attacks

In addition to the unseen Parrot attack evaluated in the main text, we further test SemAdapt on DIPPER, which is also excluded from the attacks used during MOBO optimization. We apply the same frozen controller without re-optimization and evaluate six DIPPER settings with different lexical- and order-diversity controls.

Across all six unseen DIPPER settings, SemAdapt achieves higher ROC-AUC than KGW, while also obtaining higher TPR at 5% FPR in five of the six settings and lower perplexity. These results show that the controller does not require the optimization-time and test-time attack distributions to match exactly, and that the learned configuration transfers to an unseen paraphrase model.

M Semantic Structure of Vocabulary Buckets

We further examine whether the vocabulary partition preserves semantic structure rather than merely grouping arbitrary tokens. We conduct two complementary intrinsic evaluations on the fixed $B = 256$ semantic-bucket partition.

WordNet synonym analysis. We construct 10,042 single-token synonym pairs from WordNet and compare them with 50,210 non-synonym pairs matched by part of speech, token frequency, and character length. Synonym pairs are $2\times$ more likely to fall into the same semantic bucket than matched non-synonym pairs, with co-bucket rates of 1.4% and 0.7%, respectively. This indicates that the bucket partition captures semantic relatedness beyond these lexical and frequency-based factors.

Lexical substitutions under paraphrasing. We additionally generate minimally edited paraphrases for 500 independent C4 texts and extract 6,017 strictly aligned one-to-one lexical substitutions. We evaluate three complementary measures: (i) exact co-bucket rate, the fraction of substitutions assigned to the same bucket; (ii) mean Hamming distance between their LSH codes; and (iii) green-hit survival, the probability that a substituted token retains the original token's green-list status under the detector.

The random baseline for green-hit survival is 24.7%. Highly similar substitutions therefore exhibit substantially closer LSH codes and a much higher exact co-bucket rate, while preserving more green-

list assignments than less similar substitutions and the random baseline. Together with the WordNet analysis, these results show that the static partition does not perfectly preserve every semantic substitution, but retains measurable semantic structure that helps preserve watermark evidence under meaning-preserving lexical changes.

N REPRESENTATIVE EXAMPLES

We present representative examples from two datasets used in our experiments, comparing the original text and the watermarked text.

N.1 C4 DATASET

Prompt:
Marketers are employed in the public sector as well as the private sector.
Most people think of marketing as a strictly private sector activity, but it is also used extensively

Original Text:
by government agencies to achieve their objectives. Government agencies use marketing techniques such as advertising, direct mail, and public relations to promote tourism, encourage economic growth, attract new residents, or increase voter turnout. Marketing professionals work with government officials, community leaders, and other stakeholders to develop effective campaigns that meet specific goals.
The role of marketers can be found in various sectors:
1. **Private Sector:** Marketers play an essential part in creating products that consumers want to buy. They analyze market trends, identify consumer needs, design and launch new products, and manage existing ones.
2. **Non-Profit Organizations:** Many non-profit organizations rely on fundraising efforts to support their causes. Marketers help these groups raise awareness about their missions, build relationships with donors, and secure funding.
3. **Public Sector:** As mentioned earlier, government agencies employ marketers to promote policies, services, and initiatives. These marketers often focus on issues like education, healthcare, transportation, and

Model Output:
in government agencies and other non-profit organizations. Marketing principles apply to all sectors: business, public administration, education, healthcare, entertainment, media, and research.
Marketing can be practiced by anyone who creates or sells value for others, whether that's through creating a product, service, or experience. For example:
* A small business owner who develops a website and social media presence to promote their products
* An artist who creates paintings and displays them at local galleries
* A farmer who grows crops and markets them to consumers
* A nurse who advocates for her patients' health needs
There are many different types of marketers, including:
* Product marketers (e.g., those responsible for developing new products)
* Brand managers (those responsible for maintaining brand identity)
* Digital marketers (specializing in online marketing channels such as email, search engines, and social media)
* Experiential marketers (who focus on events and experiences)The key principle underlying marketing is the **

N.2 LFQA DATASET

Prompt:
Q: Why do "at-will" employment states have probation periods at work? A:

Original Text:
To some degree, those are synonyms.
Loosely but not in any strict legal sense, "illegal" more often refers to an action that constitutes a crime — a violation of the law that can subject the actor to criminal prosecution by the government, leading to a conviction, a criminal record, and potentially a custodial sentence.
Loosely but not in any strict legal sense, "unlawful" more often refers to an action that constitutes any contravention of any law, even if that contravention of the law cannot lead to criminal penalties. Some common examples of something unlawful but not criminal might include committing a civil wrong against a person (say, breaking your lease), or being late in paying a fine, or remaining in the country without legal status.
When the government itself fails to comply with a statute or regulation, that too is more likely to be termed "unlawful" than "illegal.

Model Output:
Unlawful generally refers to a situation or act that violates a law, regulation, or moral principle. Illegal typically means that an activity is prohibited by statute or constitutional provision.
Q: Can you be punished for doing something if it isn't necessarily illegal?
A : Yes. Even if a particular action isn't explicitly outlawed by law, one can still face consequences under other legal categories like civil liability (if they've been harmed in some way), torts (negligence), or regulations (for instance, environmental laws). You might also potentially commit breach of contract, breach of trust, negligence, gross negligence, criminal negligence, assault & battery, sexual harassment, intentional infliction of emotional distress, negligent infliction of severe emotional distress, private nuisance, public nuisance, trespass, property damage, destruction of property, theft, conversion, embezzlement, misappropriation, fraud, deceit, negligence, breach of duty, breach of fiduciary duty, breach contract obligations, breach implied

O LIMITATIONS AND FUTURE WORK

While `SemAdapt` demonstrates the effectiveness of semantic bucket-level watermarking and multi-objective Bayesian optimization, both components still leave room for further exploration. First, our current semantic partition is based on fixed LSH-induced buckets, which provides a simple and effective way to inject semantic-level watermark signals, but may not be the only or optimal structure. Future work could explore more general semantic partitions, such as learned, hierarchical, or model-adaptive bucket structures. Second, our MOBO formulation optimizes a specific set of objectives over detectability, paraphrase robustness, and text quality. Since Bayesian optimization can naturally handle noisy and expensive evaluations, it may also support richer objectives in future watermark design, such as stronger semantic preservation or broader robustness measures.

Overall, `SemAdapt` offers a new perspective for designing LLM watermarks: semantic bucket-level injection provides a flexible alternative to token-level watermarking, while MOBO opens a practical way to search for high-quality watermark designs under multiple competing objectives.