SemanticShift: Robust Semantic Watermarking for Large Language Models

Minghao Li, Neset Tan

Department of Computer Science
University of Auckland
Auckland, New Zealand
minghao.lee2017@outlook.com, neset.tan@auckland.ac.nz

Abstract

Large language models (LLMs) have raised increasing concerns around misinformation and plagiarism. Watermarking—embedding identifiable signals into generated text—offers a promising approach for detection. Semantic watermarking enhances robustness by leveraging meaning rather than surface-level token patterns. However, existing semantic techniques often require model retraining or operate within constrained semantic spaces, limiting control over watermark strength, robustness, and cross-lingual generalizability. We introduce SemanticShift, a training-free and semantically grounded watermarking method that injects signals during generation by computing semantic shifts of candidate tokens relative to preceding context, guided by a secret key. SemanticShift uses pre-trained embedding models and is tunable via hyperparameters, offering strong resistance to paraphrasing and syntactic variation. Experiments demonstrate state-of-the-art detection performance, with ROC-AUC > 0.99 on original text and up to 0.96 under strong paraphrasing—outperforming all prior training-free approaches and rivaling training-based methods. Notably, SemanticShift achieves superior accuracy and robustness on models like OPT and LLAMA, showcasing its applicability and effectiveness.

Code — https://github.com/jasonisme6/llm-watermark

Introduction

Large language models (LLMs) have shown remarkable capabilities in tasks such as summarization, code generation, and open-ended dialogue (Kalyan 2024). As these models become widely integrated into real-world applications, concerns about misuse—ranging from AI-written academic essays to fake news—are growing rapidly (Lima, Grgić-Hlaa, and Redmiles 2024). The boundary between human- and machine-generated text is becoming increasingly blurred, underscoring the urgent need for reliable mechanisms to detect LLM-generated content (Liu et al. 2024b).

In this work, our goal was to develop a practical, flexible, and low-cost semantic watermarking method that avoids the heavy burden of training dedicated watermarking or detection models. Such training often requires significant resources and reduces adaptability across tasks or languages (Ling et al. 2023). Therefore, we explored whether standard semantic encoders—such as BERT (Devlin et al. 2019), Sentence-BERT (Reimers and Gurevych 2019), and their multilingual variants—can be used directly, without additional training, to achieve robustness and cross-lingual generalization. Another core objective was compatibility with existing large language models without modifying their internal parameters or probability scores. Instead, we investigated whether watermark signals could be introduced in the token sampling stage, preserving both the natural output distribution of the model and the quality of generation. By being flexible and model-agnostic, this approach can be applied to a wide range of pre-trained models with minimal cost.

Our approach is inspired by the autoregressive nature of most language models, which select each new token based on the previously generated prefix (Brown et al. 2020; Minaee et al. 2025; Naveed et al. 2024). We considered whether semantic encoders could evaluate how each candidate token affects the overall meaning of the prefix, allowing us to gently guide sampling toward tokens that shift semantics in a desired direction. Rather than forcing the model to stay within a rigid semantic space—which could harm fluency or diversity—we aim to flexibly influence the semantic trajectory of the text at each step. This enables embedding paraphrasing-resilient watermark signals while maintaining the naturalness and expressiveness of open-ended generation.

To achieve these objectives, we propose SemanticShift, a sampling-based semantic watermarking method that encodes watermark signals by controlling the direction of semantic change during generation. At each decoding step, we compute how a candidate token alters the meaning of the current text prefix using a pre-trained semantic encoder such as Sentence-BERT (Reimers and Gurevych 2019). The semantic shift is represented as the vector difference between the embedding of the prefix and that of the prefix extended by the candidate token. A secret key determines a set of preferred semantic directions corresponding to the embedded watermark. By gently steering sampling toward tokens that align with these directions, we embed the watermark without modifying the model's logits or internal weights. This process requires no model retraining and enables robust multilingual watermarking using existing encoders.

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Related Work

KGW (LeftHash, SelfHash) Token-level watermarking methods that partition the vocabulary into green and red sets using context-dependent hash functions (Kirchenbauer et al. 2023). Watermarking is achieved by biasing green-token logits during decoding, and detection counts the proportion of green tokens (Kirchenbauer et al. 2023). However, their reliance on token identity makes them fragile under paraphrasing and lexical substitutions (Kirchenbauer et al. 2023).

EXP-Edit A distortion-free watermark that injects key-dependent Gumbel noise into logits prior to sampling (Kuditipudi et al. 2023). Although marginal token probabilities are preserved, detection requires quadratic-time sequence alignment and exhibits weak statistical signals in short or low-entropy outputs (Kuditipudi et al. 2023).

SemaMark A semantic watermark that discretizes sentence embeddings into quantized regions of a low-dimensional embedding ring (Ren et al. 2023). While more robust than token-level methods, its fixed discretization limits its flexibility and structured multi-identity encoding (Ren et al. 2023).

SIR A semantic watermark that maps sentence embeddings to token-level logits via a trained auxiliary model (Liu et al. 2024a). Although robust, it requires maintaining an additional watermark model and lacks an explicit, configurable secret key, reducing portability and transparency (Liu et al. 2024a).

SemStamp, k-SemStamp Sentence-level semantic watermarks that partition embedding space using random hyperplanes or k-means clusters (Hou et al. 2024a,b). Their rejection-sampling mechanisms increase computational overhead and introduce non-deterministic generation latency (Hou et al. 2024a,b).

SimMark (Cosine-SimMark, Euclidean-SimMark) Post hoc semantic watermarks that enforce similarity constraints between sentence embeddings via rejection sampling (Dabiriaghdam and Wang 2025). While applicable to API-only models, performance depends on manually tuned similarity thresholds and suffers from unpredictable latency (Dabiriaghdam and Wang 2025).

SemanticShift Watermark

Semantic Shift Vector

Modern semantic embedding models such as BERT map text into high-dimensional vector spaces (Devlin et al. 2019). Formally, let $f(\cdot)$ denote a sentence encoder mapping text to $\mathbf{R}^d$. The geometry of these embeddings is commonly characterized by alignment and uniformity (Wang and Isola 2022). Alignment requires semantically similar texts to be mapped to nearby vectors (Wang and Isola 2022). For a pair of related sentences (x, y), with embeddings $f(x), f(y) \in \mathbf{R}^d$, alignment can be quantified by (Wang and Isola 2022):

$$\mathcal{L}_{align} = \|f(x) - f(y)\|_2^2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm (Wang and Isola 2022). Smaller distances indicate stronger semantic proximity in representation space (Wang and Isola 2022).

Uniformity concerns the global distribution of normalized embeddings on the unit hypersphere $S^{d-1} = \{z \in \mathbf{R}^d : \|z\|_2 = 1\}$ (Wang and Isola 2022). An approximately isotropic embedding space implies that the expected cosine similarity between independently sampled, normalized embeddings is close to zero, reflecting the absence of dominant global directions (Wang and Isola 2022).

In practice, however, embeddings from BERT-like models exhibit anisotropy (Rajaee and Pilehvar 2021). Instead of being evenly distributed, representations concentrate along dominant principal directions (Rajaee and Pilehvar 2021). A common abstraction models each embedding as (Mu, Bhat, and Viswanath 2018):

$$z_i = \mu + \epsilon_i,$$

where $z_i \in \mathbf{R}^d$ is the embedding of the i-th sentence, μ denotes a shared global bias direction, which often approximates the leading principal component of the embedding distribution, and ϵ_i captures instance-specific semantic variation orthogonal to this bias (Mu, Bhat, and Viswanath 2018).

Although anisotropy affects absolute embedding positions, semantic relationships are primarily encoded in relative differences. We therefore define the semantic shift vector between two samples a and b as (Mu, Bhat, and Viswanath 2018):

$$\Delta_{a,b} = z_a - z_b,$$

which yields

$$\Delta_{a,b} = (\mu + \epsilon_a) - (\mu + \epsilon_b) = \epsilon_a - \epsilon_b.$$

The shared bias component μ cancels out, leaving only relative semantic deviations (Mu, Bhat, and Viswanath 2018). Compared to raw embeddings z_i, collections of shift vectors $\Delta_{a,b}$ exhibit substantially reduced directional concentration and more balanced regional occupancy (Mu, Bhat, and Viswanath 2018).

Importantly, differencing preserves relational structure: for semantically similar pairs (a, b), the magnitude $\|\Delta_{a,b}\|_2$ tends to be smaller than for dissimilar pairs. Thus, semantic shift vectors mitigate global anisotropy while maintaining local semantic coherence.

Semantic Shift Calculation and Region Partition

In the autoregressive generation process of a large language model (LLM), tokens are generated sequentially from left to right. Let $x_1, x_2, \ldots, x_t$ denote the current sequence of t tokens. At step $t + 1$, the model predicts the next token x_{t+1} based on a semantic prefix consisting of the last m tokens, denoted by $x_{t-m+1:t}$. This prefix is passed through a pretrained Sentence-BERT encoder to obtain its semantic representation:

$$e_t = \text{Encoder}(x_{t-m+1:t})$$

where $e_t \in \mathbf{R}^d$ and d denotes the embedding dimension of the encoder. We then compute the embedding of the extended sequence that includes a candidate token:

$$e_{t+1} = \text{Encoder}(x_{t-m+1:t} + x_{t+1})$$

The semantic shift vector is defined as

$$\Delta e_t = e_{t+1} - e_t$$

To discretize the direction of Δe_t, we introduce three orthogonal hyperplanes H_1, H_2, and H_3 in the embedding space. Let $\tilde{h}_1, \tilde{h}_2, \tilde{h}_3 \sim \mathcal{N}(0, I_d)$ be independent Gaussian random vectors in $\mathbf{R}^d$, where I_d denotes the $d \times d$ identity matrix. These vectors are orthogonalized using the Gram–Schmidt process (Leon, Björck, and Gander 2013) to obtain orthonormal hyperplane normals:

$$H_1 = \frac{\tilde{h}_1}{\|\tilde{h}_1\|} \quad H_2 = \frac{\tilde{h}_2 - (\tilde{h}_2^\top H_1)H_1}{\|\tilde{h}_2 - (\tilde{h}_2^\top H_1)H_1\|}$$

$$H_3 = \frac{\tilde{h}_3 - (\tilde{h}_3^\top H_1)H_1 - (\tilde{h}_3^\top H_2)H_2}{\|\tilde{h}_3 - (\tilde{h}_3^\top H_1)H_1 - (\tilde{h}_3^\top H_2)H_2\|}$$

The hyperplanes are initialized once and kept fixed throughout generation and detection.

Each shift vector is projected onto these hyperplanes using the standard Euclidean inner product:

$$d_j = \langle \Delta e_t, H_j \rangle \quad j \in \{1, 2, 3\}$$

We assign the vector to one of the semantic regions $2^3 = 8$ by interpreting the sign pattern of the dot products as a binary code:

$$r_t = 1 + \mathbf{F}[d_1 > 0] + 2 \cdot \mathbf{F}[d_2 > 0] + 4 \cdot \mathbf{F}[d_3 > 0]$$

Here, $\mathbf{F}[P]$ denotes the indicator function:

$$\mathbf{F}[P] = \begin{cases} 1 & \text{if } P \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

This partition yields $2^3 = 8$ semantic regions. The corresponding $8! = 40{,}320$ possible region-priority permutations define a large combinatorial key space for region-based sampling. Such permutations enable flexible watermark encoding and support user-specific keys derived from metadata (e.g., user ID or timestamp) (Yoo, Ahn, and Kwak 2023).

Watermark Token Selection via Region Priorities

Watermarking is applied during decoding by steering the model toward tokens whose semantic shift vectors fall into high-priority regions specified by a secret key. At each decoding step, we first select the top-N candidate tokens from the model's output distribution:

$$\text{top-N}(x_{t+1}) = \arg\,\text{top-N}_{x \in \mathcal{V}}\, P(x \mid x_{1:t})$$

where $\mathcal{V}$ denotes the model vocabulary. For each candidate token x_{t+1}, we compute its shift vector Δe_t and corresponding region index r_t using the region assignment procedure defined previously. To determine whether watermarking should be applied, we compute the entropy of the softmax-normalized logits over the same top-N candidates:

$$H = -\sum_{i=1}^{N} p_i \log p_i \qquad p_i = \frac{\exp(s_i)}{\sum_{j=1}^{N} \exp(s_j)}$$

Here, s_i denotes the logit of the i-th candidate, and λ is a predefined entropy threshold. If $H < \lambda$, indicating high model confidence (low uncertainty), watermarking is skipped and the highest-scoring token is selected. Otherwise, if $H \geq \lambda$, the watermarking procedure is activated. A summary is provided in Algorithm 1.

Algorithm 1: Generation with Entropy Gate

Input: Model $logits$, Top-N size N, entropy threshold λ
Output: low-entropy $flag$

$[s_1, \ldots, s_N] \leftarrow \texttt{TopN}(logits)$
for $j = 1$ to N **do**
  $p_j \leftarrow \frac{\exp(s_j)}{\sum_{k=1}^{N} \exp(s_k)}$
end for
$H \leftarrow -\sum_{j=1}^{N} p_j \log p_j$
$flag \leftarrow (H < \lambda)$
return $flag$

A watermark key is defined as an ordered permutation of the eight region indices, encoding a secret priority from the most preferred region to the least preferred. For each region r in this priority list, we identify the subset of candidate tokens whose shift vectors map to r:

$$\mathcal{C}_r = \{i \in \{1, ..., N\} \mid \text{Region}(x_i) = r\}$$

where $\text{Region}(\cdot)$ denotes the region assignment function defined in the previous section. If $\mathcal{C}_r$ is non-empty, we compute the softmax over the corresponding subset of logits $\{s_i \mid i \in \mathcal{C}_r\}$ and sample a token from this restricted distribution:

$$x^* \sim \text{Multinomial}(1, \text{Softmax}(\{s_i \mid i \in \mathcal{C}_r\}))$$

If no candidate maps to the current region, we proceed to the next region in the priority list and repeat the process until a valid token is selected. A summary is provided in Algorithm 2.

This priority-based selection mechanism ensures that token sampling remains stochastic while being semantically aligned with the secret key. It preserves generation fluency while embedding a signal detectable through the directional pattern of semantic shifts, without modifying the LLM's parameters or relying on surface-level token statistics.

Watermark Detection via Shift Region

To detect the presence of a watermark, we analyze the distribution of semantic shift regions across the generated sequence. Given a generated text, we tokenize it and iterate over token positions to compute semantic shift vectors and assign them to regions. Let C_r denote the number of tokens whose shift vector falls into region r, and let $L = \sum_{r=1}^{8} C_r$

Algorithm 2: SemanticShift Region Selection

Input: Generated prefix tokens $x_{1:t}$, prefix length m, Encoder, hyperplanes $H = \{H_1, H_2, H_3\}$, key π, Top-N indices I, scores $\{s_j\}$
Output: Selected token index $j^\star$

Initialize $\mathcal{C}_r \leftarrow \emptyset$ for $r = 1, \ldots, 8$
$prefix \leftarrow x_{t-m+1:t}$
$e_{prefix} \leftarrow \texttt{Encoder}(prefix)$
for all $j \in I$ **do**
  $e_{current} \leftarrow \texttt{Encoder}(prefix \| \texttt{token}(j))$
  $\Delta e \leftarrow e_{current} - e_{prefix}$
  for $i = 1$ to 3 **do**
    $d_i \leftarrow \langle \Delta e, H_i \rangle$
  end for
  $r \leftarrow 1 + \mathbf{F}[d_1 > 0] + 2\mathbf{F}[d_2 > 0] + 4\mathbf{F}[d_3 > 0]$
  $\mathcal{C}_r \leftarrow \mathcal{C}_r \cup \{j\}$
end for
for $k = 1$ to 8 **do**
  $r \leftarrow \pi[k]$
  if $\mathcal{C}_r \neq \emptyset$ **then**
    Sample $j^\star \sim \texttt{Softmax}(\{s_j \mid j \in \mathcal{C}_r\})$
    return $j^\star$
  end if
end for
return $\arg\max_{j \in I} s_j$

denote the total number of valid shift vectors in the text. The detection procedure is summarized in Algorithm 3.

To improve robustness and focus on semantically informative positions, we apply the same entropy-based filtering used during generation. At each token position, we compute the entropy over the top-K next-token logits. If the entropy falls below the threshold λ, indicating high model confidence, the token is excluded from the region statistics. Consequently, only high-uncertainty positions contribute to watermark detection. To evaluate whether the observed region distribution aligns with the secret key, we compute cumulative z-scores over the first k key-priority regions (Lu et al. 2024), for $k \in \{1, 2, 3, 4\}$. Let G_k denote the cumulative number of shift vectors falling within the first k priority regions. We define

$$z_k = \frac{G_k - \gamma_k L}{\sqrt{L\gamma_k(1-\gamma_k)}}$$

where $\gamma_k = k/8$ represents the expected coverage under a uniform regional prior, and L is the total number of semantic shifts in the text. The final detection statistic is computed as the average:

$$\bar{z} = \frac{1}{4}\sum_{k=1}^{4} z_k$$

Because detection operates in semantic embedding space rather than at the token level, the resulting shift vectors tend to remain within similar regions under minor edits or paraphrasing. This property contributes to the robustness of the SemanticShift watermark. The computation of the cumulative z-score is summarized in Algorithm 4.

Algorithm 3: Detection with Entropy Gate

Input: Text x, LLM $model$
Parameter: hyperplanes H, entropy threshold λ, top-N, prefix length m, Sentence-BERT $Encoder$
Output: Region counts $\{C_r\}_{r=1}^{8}$

Tokenize x into $[x_1, \ldots, x_t]$
Initialize $C_r \leftarrow 0$ for $r = 1 \ldots 8$
for $i = 1$ to t **do**
  $prefix \leftarrow x_{i-m:i-1}$
  if $prefix$ is not empty **then**
    $logits \leftarrow model(prefix)$
    $[s_1, \ldots, s_N] \leftarrow \texttt{TopN}(logits)$
    $p_j \leftarrow \frac{e^{s_j}}{\sum_{k=1}^{N} e^{s_k}}$ for $j = 1 \ldots N$
    $\mathcal{H} \leftarrow -\sum_{j=1}^{N} p_j \log p_j$
    if $\mathcal{H} < \lambda$ **then**
      continue
    end if
  end if
  $e_{prefix} \leftarrow \texttt{Encoder}(prefix)$
  $e_{current} \leftarrow \texttt{Encoder}(prefix \parallel x_i)$
  $\Delta e \leftarrow e_{current} - e_{prefix}$
  $d_u \leftarrow \langle \Delta e, H_u \rangle$ for $u = 1, 2, 3$
  $r \leftarrow 1 + \mathbf{F}[d_1 > 0] + 2 \cdot \mathbf{F}[d_2 > 0] + 4 \cdot \mathbf{F}[d_3 > 0]$
  $C_r \leftarrow C_r + 1$
end for
return $\{C_r\}_{r=1}^{8}$

Results

Models and Datasets

We conduct experiments with seven autoregressive language models: OPT-1.3B, OPT-2.7B, OPT-6.7B, LLaMA-7B, LLaMA2-7B, LLaMA3, and Qwen3, to evaluate the quality, robustness, and paraphrasing resilience of our watermarking approach under realistic conditions. These models include both base and instruction-tuned variants, reflecting diverse architectural designs and training paradigms. To evaluate cross-lingual generalization, we additionally include BLOOM-176B, a model capable of generating text in 46 natural languages (Le Scao et al. 2023). Regarding datasets, we use the first sentence of each summary from 1,000 samples of the BookSum corpus and sample 1,000 examples from the C4 RealNews corpus, using the first sentence of each article as the generation prompt (Raffel et al. 2020). These datasets provide diverse inputs across multiple domains, enabling comprehensive evaluation of quality and robustness.

Experimental Setups

Our experiments evaluate SemanticShift in terms of detection robustness, text quality, cross-lingual generalization, latency, embedding-space geometry, and parameter sensitivity. We compare against ten baselines: LeftHash, SelfHash, EXP-Edit, SemaMark, KGW, SIR, SemStamp, k-SemStamp, Cosine-SimMark, and Euclidean-SimMark, using the same detection protocol as prior work (Ren

Algorithm 4: Key-Based Cumulative z-Score

Input: Region counts $\{C_r\}_{r=1}^{8}$, key $\pi = (\pi_1, \ldots, \pi_8)$
Parameter: baseline coverage $\{\gamma_k\}_{k=1}^{4}$ (default $\gamma_k = k/8$)
Output: Detection score $\bar{z}$

$L \leftarrow \sum_{r=1}^{8} C_r$
if $L = 0$ **then**
  return 0
end if
for $k = 1$ to 4 **do**
  $G_k \leftarrow \sum_{i=1}^{k} C_{\pi_i}$
  $z_k \leftarrow \frac{G_k - \gamma_k L}{\sqrt{L\gamma_k(1-\gamma_k)}}$
end for
$\bar{z} \leftarrow \frac{1}{4}\sum_{k=1}^{4} z_k$
return $\bar{z}$

et al. 2023). Robustness is assessed under five paraphrasing attacks—round-trip translation, DIPPER, GPT-3.5 rewriting, PEGASUS, and Parrot—covering rule-based and neural network transformations. For each attack, we sample 1,000 prompts from C4 RealNews and generate 1,000 watermarked and 1,000 non-watermarked outputs of comparable length. Text quality is evaluated on 1,000 BookSum samples (Kryściński et al. 2022), comparing fluency and diversity metrics between watermarked and standard decoding. Latency is measured by comparing decoding time with and without watermarking across varying output lengths. We additionally analyze embedding-space geometry by comparing the regional distributions of prefix embeddings and prefix shift vectors against the uniform prior (12.5% per region). Finally, we conduct ablations over key parameters, including prefix length, number of hyperplanes, embedding model, temperature, entropy threshold, candidate pool size, language, and output length.

Evaluation Metrics

To comprehensively assess detection robustness, generation quality, generation latency, and embedding-space isotropy, we employ a suite of evaluation metrics. We compute ROC-AUC and F1 scores using the z-scores produced by the watermark detectors to quantify detection robustness. Z-scores are calculated for watermarked texts, unwatermarked texts, and their paraphrased variants. ROC-AUC measures the separability between watermarked and unwatermarked samples, while the F1 score is obtained by selecting the threshold that maximizes the harmonic mean of precision and recall, reflecting the optimal trade-off between them. To evaluate text quality under paraphrasing, we use Perplexity (PPL), Semantic Entropy (Sem-Ent), Diversity, 3-gram Entropy (Ent-3), and MAUVE. Perplexity is computed using a pretrained language model to measure the likelihood of the generated text, where lower values indicate greater fluency (Hu et al. 2024). 3-gram entropy is derived from the entropy of observed trigrams to assess lexical diversity, with higher values indicating less repetitive phrasing (Zhang et al. 2018). Semantic Entropy measures semantic diversity by computing the entropy of sentence embeddings in the semantic space (Han, Kim, and Chang 2022). MAUVE quantifies distributional divergence between model-generated text and human-written reference text. Generation latency is measured as the average decoding time across varying output lengths (Pillutla et al. 2023). Finally, embedding-space isotropy is evaluated by examining the regional proportions of prefix embeddings and prefix shift vectors across the eight geometric regions.

Main Results

Robustness We conduct a comprehensive robustness evaluation of SemanticShift under diverse paraphrasing attacks across multiple model families. As shown in Tables 1 to 3, the proposed approach consistently ranks among the top-performing methods under both clean generation and paraphrased conditions. For OPT-2.7B and OPT-6.7B, SemanticShift achieves near-perfect detection accuracy under original generation and maintains strong performance under paraphrasing attacks such as round-trip translation, DIPPER, and GPT-3.5 rewriting, despite moderate degradation. In contrast, token-level watermarking techniques such as LeftHash and SelfHash exhibit substantial performance drops under surface-level lexical perturbations, highlighting their sensitivity to token substitutions. On LLaMA and LLaMA2 models, SemanticShift similarly demonstrates stable robustness under both edit-based and semantic paraphrasing attacks, suggesting that the watermark signal generalizes across different architectural configurations and model scales. These results are consistent with the model-agnostic design of the semantic projection mechanism. Furthermore, performance remains stable across instruction-tuned variants, indicating that the watermark operates at a semantic level rather than relying on surface-level token statistics. Compared to other semantic watermarking approaches on OPT-1.3B, SemanticShift achieves stronger robustness while maintaining high precision under paraphrasing attacks such as DIPPER and round-trip translation. Performance remains competitive under rewriting methods including PEGASUS, Parrot, and GPT-3.5, suggesting that semantic shift embedding provides resilience under diverse paraphrasing transformations.

Text quality As shown in Table 4, SemanticShift introduces a noticeable increase in perplexity compared to non-watermarked decoding. This increase arises from the influence of semantic steering during generation, which biases token selection toward directions aligned with semantic embedding constraints. Importantly, this increase reflects a trade-off between semantic steering strength and fluency, rather than indicating substantial degradation in text quality. By modifying the generation objective from maximizing individual token probabilities to optimizing under semantic constraints, the model balances fluency against robustness. A weaker steering force tends to preserve fluency at the cost of reduced robustness, whereas stronger steering enhances resistance to paraphrasing attacks while increasing perplexity. Larger models, such as OPT-6.7B relative to OPT-2.7B, appear better able to absorb the steering effect while maintaining competitive fluency. Although perplexity increases under watermarking, lexical and semantic entropy remain

Model	Attack	ROC-AUC					F1				
		LH	SH	EXP	SM	SS	LH	SH	EXP	SM	SS
OPT-2.7B	None	99.13	98.86	97.99	99.48	**99.84**	99.21	98.61	97.08	99.05	**99.90**
	Translation	90.91	81.47	87.49	**96.92**	95.63	84.56	76.22	81.57	**93.30**	89.77
	DIPPER	98.78	97.28	97.36	99.11	**99.49**	**97.27**	94.00	96.20	97.01	96.99
	GPT-3.5	90.28	79.08	93.92	94.06	**94.57**	83.58	73.78	88.52	**89.02**	87.52
OPT-6.7B	None	99.18	99.30	97.84	99.49	**99.87**	99.11	98.63	97.05	98.58	**99.90**
	Translation	88.07	80.98	86.25	93.08	**94.72**	81.29	74.68	80.13	88.82	**89.00**
	DIPPER	99.04	97.47	97.28	98.71	**99.30**	97.86	94.32	96.20	**98.21**	95.88
	GPT-3.5	89.90	79.09	89.96	93.77	**95.00**	83.00	73.67	83.54	87.66	**88.78**

Table 1: Robustness (%) on OPT models without paraphrasing and under three paraphrase attacks. LH: LeftHash, SH: SelfHash, EXP: EXP-Edit, SM: SemaMark, SS: SemanticShift (ours).

Model	Algorithm	ROC-AUC	F1
LLaMA-7B	LH	81.9	74.8
	SH	83.8	77.4
	SM	84.6	78.1
	SS	**88.4**	**81.8**
LLaMA2-7B	LH	81.1	74.9
	SH	84.1	77.3
	SM	87.2	81.0
	SS	**88.0**	**81.3**

Table 2: Robustness (%) on LLaMA models under round-trip translation. LH: LeftHash, SH: SelfHash, SM: SemaMark, SS: SemanticShift (ours).

comparable across methods, indicating that higher perplexity does not necessarily imply reduced diversity or semantic richness. A similar pattern is observed in larger models such as LLaMA3 and Qwen3 in Table 5. Despite mild increases in perplexity and moderate variation in diversity or MAUVE scores, the overall semantic characteristics of generated text remain largely consistent with the non-watermarked setting. Overall, the perplexity increase observed under SemanticShift can be interpreted as a controllable factor of semantic watermarking. Key parameters—including entropy thresholds, prefix length, and candidate pool size—allow flexible adjustment of the trade-off between fluency and watermark robustness, enabling task-specific calibration of semantic watermark strength.

Isotropy Prior work has shown that Transformer-based sentence encoders often exhibit anisotropic embedding spaces, in which representations concentrate along a few dominant directions rather than being uniformly distributed (Rajaee and Pilehvar 2021). Because SemanticShift relies on directional statistical analysis in embedding space, it is therefore necessary to examine whether semantic differencing mitigates this anisotropy and yields a geometry compatible with region-based watermarking. The differencing operation removes global components shared by consecutive prefix embeddings and isolates the incremental semantic contribution introduced by the current token. To assess isotropy, the embedding space is partitioned into eight regions using three randomly generated orthogonal hyperplanes. Each embedding vector—either a raw prefix embedding or a prefix shift vector—is assigned to a region according to the signs of its projections onto the hyperplane normals. Under ideal isotropic conditions, each region is expected to contain approximately 12.5% of the vectors. For each text, a sliding prefix window of size $m = 5$ is applied, and region counts are accumulated for both raw prefix embeddings and shift vectors. The resulting distributions, reported in Table 6, show substantial deviation from uniformity for raw prefix embeddings, consistent with known embedding anisotropy. In some cases, the distribution collapses into only a few regions, reflecting strong global principal directions. In contrast, prefix shift vectors exhibit a significantly more balanced regional distribution, even in highly anisotropic embedding spaces. These results indicate that semantic differencing mitigates anisotropy and redistributes directional mass across the embedding space. Such near-uniform regional occupancy is essential for SemanticShift, as it underpins the approximate uniform prior assumption in the cumulative z-score detection framework and ensures proper normalization of region-based watermark statistics.

Latency From Table 7, we see that the generation latency of the model is directly proportional to the length of the generated text in both cases. Relative to the baseline decoding process, SemanticShift adds latency due to the additional semantic computations that are conducted per step. Specifically, for each step, SemanticShift calculates the semantic embedding projections, assigns the shift vector to the semantic region, and checks the direction alignment based on the key before sampling the next token. This adds a fixed step cost that scales linearly with the input length. Notably, this overhead grows linearly with regards to the number of tokens produced and does not suffer from super-linear slowdowns and instability issues that can be caused by more complex watermarking and rejection sampling schemes. Although it takes longer than regular decoding without semantic watermarking, this latency is bounded and linear and thus suitable for batch generation or scenarios where latency is not an issue.

Ablation Study As shown in Table 8, we analyze how key hyperparameters affect both text quality and watermark robustness using the OPT-6.7B model. We vary the following parameters over predefined ranges: prefix length $m \in \{1, 5, 10, 15\}$, number of hyperplanes

Model	Algorithm	No Paraphrase	PEGASUS	Parrot	GPT-3.5
OPT-1.3B	KGW	99.6 / 98.4 / 98.9	95.9 / 82.1 / 91.0	88.5 / 31.5 / 55.4	82.8 / 17.4 / 46.7
	SIR	99.9 / 99.4 / 99.9	94.4 / 79.2 / 85.4	93.2 / 62.8 / 75.9	80.2 / 24.7 / 42.7
	SemStamp	99.2 / 93.9 / 97.1	97.8 / 83.7 / 92.0	93.3 / 56.2 / 75.5	83.3 / 33.9 / 52.9
	k-SemStamp	99.6 / 98.1 / 98.7	**99.5** / 92.7 / 96.5	97.8 / 78.7 / 89.4	90.8 / 55.5 / 71.8
	Cosine-SimMark	99.6 / 96.8 / 98.8	99.2 / 90.3 / **98.2**	98.7 / 88.1 / 97.2	95.7 / 59.7 / **86.7**
	Euclidean-SimMark	99.8 / 98.5 / 99.3	97.2 / 72.3 / 89.1	95.7 / 60.2 / 82.5	94.1 / 51.6 / 76.2
	SemanticShift	**100 / 100 / 100**	**99.5 / 97.2 / 98.2**	**99.4 / 93.6 / 97.6**	**96.4 / 60.9** / 81.2

Table 3: Robustness (%) on different watermarking algorithms. Each cell reports ROC-AUC / TP@FP=1% / TP@FP=5%.

Model	Algorithm	PPL	Ent-3	Sem-Ent
OPT-1.3B	NW	11.89	11.43	3.10
	SM	12.89	11.51	3.17
	KSM	11.82	11.48	3.11
	CSM	12.69	11.50	**3.39**
	ESM	**9.67**	11.50	3.28
	SS	18.38	**11.73**	3.15

Table 4: Text quality on OPT-1.3B. NW: No watermark, SM: SemStamp, KSM: k-SemStamp, CSM: Cosine-SimMark, ESM: Euclidean-SimMark, SS: SemanticShift (ours).

Model	Type	PPL	Ent-3	SE	DS	ME
LLaMA3	NW	10.92	11.20	3.32	7.96	0.95
	W	18.62	11.17	3.14	6.56	0.82
Qwen3	NW	6.30	11.21	3.21	6.90	0.45
	W	16.37	11.10	3.16	6.71	0.67

Table 5: Text Quality Comparison Between Non-Watermarked and Watermarked Text on LLaMA3 and Qwen3. SE: Semantic Entropy, DS: Diversity; ME: MAUVE, NW: Non-Watermarked, W: Watermarked.

$h \in \{1, 2, 3\}$, entropy threshold $\lambda \in \{0.1, 0.5, 1.0, 1.5\}$, candidate set size $N \in \{5, 10, 15, 20\}$, temperature $\tau \in \{0.1, 0.5, 1.0, 1.5, 2.0\}$, language $\ell \in \{English(en), Chinese(ch), Hindi(hi), Japanese(ja)\}$, and output length $L \in \{25, 75, 125, [175, 225]\}$. We also evaluate four embedding models ϕ: all-mpnet-base-v2 (Song et al. 2020), multilingual-e5-large-instruct (Wang et al. 2024), paraphrase-multilingual-minilm-l12-v2 (Wang et al. 2020), and qwen3-embedding-0.6b (Zhang et al. 2025). In each ablation experiment, we vary one parameter while fixing the others at their default configuration ($m = 5$, $\lambda = 0.1$, $N = 10$, $h = 3$, $\tau = 1$, $L = [175, 225]$, $\phi = multilingual - e5 - large - instruct$, $\ell = en$). We report average perplexity (PPL), lexical entropy (Ent-3), semantic entropy (Sem-Ent), and detection performance (ROC-AUC and F1) on both clean outputs and GPT-3.5 paraphrased outputs. The results indicate that, except under extreme parameter configurations, both text quality and watermark strength remain relatively stable. Most parameter variations primarily affect detection robustness under paraphrasing, while performance on clean text remains largely unchanged. Notably, the watermark signal exhibits weak dependence on output length L, suggesting that

Model	Type	Region							
		1	2	3	4	5	6	7	8
mpnet	PV	18	27	5	7	12	17	4	5
	PVS	14	11	11	11	10	11	14	14
MiniLM	PV	19	27	7	9	11	15	3	5
	PVS	12	9	11	12	10	14	12	17
qwen3	PV	6	41	6	40	0	1	0	1
	PVS	14	11	9	10	15	11	15	12
e5	PV	0	0	0	0	94	0	5	0
	PVS	17	16	9	12	9	11	11	12

Table 6: Vector space distribution (%) across eight regions. PV: prefix vector, PVS: prefix vector shift, mpnet = all-mpnet-base-v2, MiniLM = paraphrase-multilingual-MiniLM-L12-v2, qwen3 = qwen3-embedding-0.6b, and e5 = multilingual-e5-large-instruct.

Model	Type	Output Length					
		25	50	100	125	150	175
OPT-1.3B	NW	1.8	2.1	4.0	5.0	5.9	7.6
	W	3.7	5.9	12.2	14.6	18.2	21.5

Table 7: Latency (s) by different output length on OPT-1.3B. NW: No Watermark, W: Watermark.

SemanticShift maintains effectiveness even for shorter text segments. Furthermore, the multilingual results demonstrate consistently high ROC-AUC and F1 scores in English, Chinese, Hindi, and Japanese, confirming that our watermarking method generalizes well across languages without significant degradation in generation quality.

Conclusion

We introduced SemanticShift, a robust and training-free semantic watermarking method for large language models. By leveraging pre-trained embedding models to analyze token-level semantic shifts during generation, SemanticShift steers candidate selection along secret-key-defined directions in the embedding space. This approach embeds watermark signals without modifying model parameters or requiring additional training, making it highly adaptable across models, tasks, and languages. Unlike prior techniques that rely on surface-level token hashes or trained watermark networks, SemanticShift operates directly in the semantic space with region-based steering, offering resilience to paraphrasing, translation, and sentence reordering. Its entropy-aware con-

Variable	Value	PPL	Ent-3	Sem-Ent	ROC-AUC	F1
Prefix Length m	1	**5.15**	9.53	2.32	99.79 / 91.24	99.70 / 84.11
	5	18.69	11.18	2.24	99.87 / **95.00**	**99.90** / **88.78**
	10	19.61	11.24	2.35	99.90 / 82.86	**99.90** / 76.79
	15	22.22	**11.35**	**2.41**	**99.99** / 82.29	99.80 / 77.37
Hyperplanes d	1	**9.99**	8.63	2.48	99.76 / 85.32	99.40 / 79.69
	2	11.25	10.44	**2.51**	**99.99** / 80.74	**99.90** / 75.43
	3	18.69	**11.18**	2.24	99.87 / **95.00**	**99.90** / **88.78**
Embedding Model ϕ	mpnet	19.06	10.96	2.33	99.92 / 84.97	99.90 / 77.42
	MiniLM	**11.96**	10.50	2.31	**100.0** / 88.24	**100.0** / 80.15
	qwen3	18.54	11.01	**2.41**	99.96 / 93.69	99.90 / 86.58
	e5	18.69	**11.18**	2.24	99.87 / **95.00**	99.90 / **88.78**
Temperature τ	0.1	**2.82**	9.84	2.57	68.53 / 55.15	66.67 / 70.44
	0.5	5.37	10.63	**2.62**	99.13 / 81.15	97.98 / 76.62
	1.0	18.69	11.18	2.24	99.87 / 95.00	99.90 / 88.78
	1.5	25.99	11.20	2.13	**100.0** / 94.93	**100.0** / 87.75
	2.0	29.99	**11.24**	2.13	**100.0** / **96.05**	**100.0** / **89.30**
Entropy Threshold λ	0.1	18.69	**11.18**	2.24	99.87 / **95.00**	99.90 / **88.78**
	0.5	14.19	11.10	2.41	**100.0** / 92.41	**100.0** / 84.16
	1.0	9.36	10.94	2.58	99.71 / 91.47	99.50 / 83.65
	1.5	**5.47**	10.67	**2.59**	98.72 / 82.32	96.73 / 76.88
Token Candidate N	5	**10.88**	11.11	**2.49**	**100.0** / 89.40	99.90 / 81.46
	10	18.69	**11.18**	2.24	99.87 / 95.00	99.90 / 88.78
	15	23.58	11.15	2.23	**100.0** / 94.84	**100.0** / 87.94
	20	27.65	11.15	2.17	99.99 / **95.98**	99.90 / **89.30**
Output Length L	25	61.96	9.28	2.29	99.63 / 63.00	98.89 / 69.16
	75	30.07	10.33	2.25	99.80 / 82.50	99.80 / 77.08
	125	23.92	10.76	**2.30**	99.80 / 89.68	**99.90** / 81.62
	[175,225]	**18.69**	**11.17**	2.24	**99.87** / **95.00**	**99.90** / **88.78**
Language ℓ	en	**18.68**	11.17	2.17	99.87 / **94.72**	**99.90** / **89.00**
	ch	35.34	**11.25**	**2.81**	99.39 / 93.21	98.81 / 88.87
	hi	21.58	10.22	2.69	**99.99** / 91.67	99.50 / 85.46
	ja	23.90	10.36	2.30	98.42 / 84.88	97.94 / 81.89

Table 8: Ablation study on OPT-6.7B before/after GPT-3.5 paraphrase. mpnet = all-mpnet-base-v2, MiniLM = paraphrase-multilingual-MiniLM-L12-v2, qwen3 = qwen3-embedding-0.6b, and e5 = multilingual-e5-large-instruct.

trol enables tunable robustness, and its personalized key system supports multi-user attribution and secure traceability. Extensive experiments demonstrate that SemanticShift achieves state-of-the-art performance among training-free watermarking methods and matches or exceeds that of trained baselines like SemaMark and SimMark. It maintains strong detection accuracy under aggressive paraphrasing attacks and lexical diversity, while ensuring consistent robustness across multilingual settings. These results show that SemanticShift is a practical and effective watermarking solution for real-world LLM applications, particularly in scenarios where training additional components is infeasible and robustness to transformation is critical. Future work may explore extending semantic steering to multimodal watermarking and integrating SemanticShift into broader AI provenance and accountability frameworks.

Limitations

Despite its robustness and flexibility, SemanticShift has several limitations. First, it relies on access to token-level generation dynamics, which restricts its use to white-box autoregressive models. This design precludes direct application to black-box APIs, where internal sampling behavior and candidate logits are inaccessible. Additionally, SemanticShift assumes a left-to-right generation paradigm for computing incremental semantic shifts, rendering it less compatible with non-autoregressive or bidirectional architectures. Supporting such models would necessitate a fundamental rethinking of both the embedding and detection mechanisms to accommodate their reliance on global context rather than sequential progression. Another limitation concerns the generation latency. Since the method computes token-level semantic shift directions via embedding-model projections for multiple candidate tokens at each decoding step, it introduces a moderate computational overhead during inference. In practice, when a lightweight encoder is used, this added latency is almost imperceptible; the overhead becomes noticeable only with large embedding models and long outputs. Although this cost is acceptable for offline generation or controlled settings, it may pose a challenge for real-time applications that require low-latency responses.

References

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Dabiriaghdam, A.; and Wang, L. 2025. SimMark: A Robust Sentence-Level Similarity-Based Watermarking Algorithm for Large Language Models. arXiv:2502.02787.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 4171–4186.

Han, S.; Kim, B.; and Chang, B. 2022. Measuring and improving semantic diversity of dialogue generation. *arXiv preprint arXiv:2210.05725*.

Hou, A. B.; Zhang, J.; He, T.; Wang, Y.; Chuang, Y.-S.; Wang, H.; Shen, L.; Durme, B. V.; Khashabi, D.; and Tsvetkov, Y. 2024a. SemStamp: A Semantic Watermark with Paraphrastic Robustness for Text Generation. arXiv:2310.03991.

Hou, A. B.; Zhang, J.; Wang, Y.; Khashabi, D.; and He, T. 2024b. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. *arXiv preprint arXiv:2402.11399*.

Hu, Y.; Huang, Q.; Tao, M.; Zhang, C.; and Feng, Y. 2024. Can Perplexity Reflect Large Language Model's Ability in Long Text Understanding? *arXiv preprint arXiv:2405.06105*.

Kalyan, K. S. 2024. A survey of GPT-3 family large language models including ChatGPT and GPT-4. *Natural Language Processing Journal*, 6: 100048.

Kirchenbauer, J.; Geiping, J.; Wen, Y.; Shu, M.; Saifullah, K.; Kong, K.; Fernando, K.; Saha, A.; Goldblum, M.; and Goldstein, T. 2023. On the reliability of watermarks for large language models. *arXiv preprint arXiv:2306.04634*.

Kryściński, W.; Rajani, N.; Agarwal, D.; Xiong, C.; and Radev, D. 2022. BookSum: A Collection of Datasets for Long-form Narrative Summarization. arXiv:2105.08209.

Kuditipudi, R.; Thickstun, J.; Hashimoto, T.; and Liang, P. 2023. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*.

Le Scao, T.; Fan, A.; Akiki, C.; Pavlick, E.; Ilić, S.; Hesslow, D.; Castagné, R.; Luccioni, A. S.; Yvon, F.; Gallé, M.; et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Leon, S. J.; Björck, Å.; and Gander, W. 2013. Gram-Schmidt orthogonalization: 100 years and more. *Numerical Linear Algebra with Applications*, 20(3): 492–532.

Lima, G.; Grgić-Hlaa, N.; and Redmiles, E. M. 2024. Public Opinions About Copyright for AI-Generated Art: The Role of Egocentricity, Competition, and Experience. *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*.

Ling, C.; Zhao, X.; Lu, J.; Deng, C.; Zheng, C.; Wang, J.; Chowdhury, T.; Li, Y.-Q.; Cui, H.; Zhang, X.; yu Zhao, T.; Panalkar, A.; Cheng, W.; Wang, H.; Liu, Y.; Chen, Z.; Chen, H.; White, C.; Gu, Q.; Pei, J.; Yang, C.; and Zhao, L. 2023. Domain Specialization as the Key to Make Large Language Models Disruptive: A Comprehensive Survey. *ACM Computing Surveys*, 58: 1 – 39.

Liu, A.; Pan, L.; Hu, X.; Meng, S.; and Wen, L. 2024a. A Semantic Invariant Robust Watermark for Large Language Models. arXiv:2310.06356.

Liu, A.; Pan, L.; Lu, Y.; Li, J.; Hu, X.; Zhang, X.; Wen, L.; King, I.; Xiong, H.; and Yu, P. 2024b. A survey of text watermarking in the era of large language models. *ACM Computing Surveys*, 57(2): 1–36.

Lu, Y.; Liu, A.; Yu, D.; Li, J.; and King, I. 2024. An Entropy-based Text Watermarking Detection Method. arXiv:2403.13485.

Minaee, S.; Mikolov, T.; Nikzad, N.; Chenaghlu, M.; Socher, R.; Amatriain, X.; and Gao, J. 2025. Large Language Models: A Survey. arXiv:2402.06196.

Mu, J.; Bhat, S.; and Viswanath, P. 2018. All-but-the-Top: Simple and Effective Postprocessing for Word Representations. arXiv:1702.01417.

Naveed, H.; Khan, A. U.; Qiu, S.; Saqib, M.; Anwar, S.; Usman, M.; Akhtar, N.; Barnes, N.; and Mian, A. 2024. A Comprehensive Overview of Large Language Models. arXiv:2307.06435.

Pillutla, K.; Liu, L.; Thickstun, J.; Welleck, S.; Swayamdipta, S.; Zellers, R.; Oh, S.; Choi, Y.; and Harchaoui, Z. 2023. MAUVE Scores for Generative Models: Theory and Practice. arXiv:2212.14578.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140): 1–67.

Rajaee, S.; and Pilehvar, M. T. 2021. An isotropy analysis in the multilingual BERT embedding space. *arXiv preprint arXiv:2110.04504*.

Reimers, N.; and Gurevych, I. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ren, J.; Xu, H.; Liu, Y.; Cui, Y.; Wang, S.; Yin, D.; and Tang, J. 2023. A robust semantics-based watermark for large language model against paraphrasing. *arXiv preprint arXiv:2311.08721*.

Song, K.; Tan, X.; Qin, T.; Lu, J.; and Liu, T.-Y. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. arXiv:2004.09297.

Wang, L.; Yang, N.; Huang, X.; Yang, L.; Majumder, R.; and Wei, F. 2024. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

Wang, T.; and Isola, P. 2022. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. arXiv:2005.10242.

Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; and Zhou, M. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. arXiv:2002.10957.

Yoo, K.; Ahn, W.; and Kwak, N. 2023. Advancing beyond identification: Multi-bit watermark for large language models. *arXiv preprint arXiv:2308.00221*.

Zhang, Y.; Galley, M.; Gao, J.; Gan, Z.; Li, X.; Brockett, C.; and Dolan, B. 2018. Generating informative and diverse conversational responses via adversarial information maximization. *Advances in Neural Information Processing Systems*, 31.

Zhang, Y.; Li, M.; Long, D.; Zhang, X.; Lin, H.; Yang, B.; Xie, P.; Yang, A.; Liu, D.; Lin, J.; Huang, F.; and Zhou, J. 2025. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. arXiv:2506.05176.