Robust Text Watermarking for Large Language Models via Dual Semantic Embeddings

Jonas Schäfer **Cezary Pilaszewicz** **Gerhard Wunder**
Department of Mathematics and Computer Science
Freie Universität Berlin
Berlin, Germany
Correspondence: jonas.schaefer2@fu-berlin.de

Abstract

This work presents Dual-Embedding Watermarking (DEW), a semantic watermarking scheme for large language models (LLMs) that leverages contextual and token-level embeddings to enhance robustness against paraphrasing and translation. DEW utilizes a signal-processing methodology, applying algebraic vector-space operations to token and context embeddings to derive a watermark signal that degrades gracefully under semantic shifts. The method obfuscates the watermark by projecting embedding vectors through pseudo-random matrices seeded with a secret key. Experimental results show that dual-embedding watermarking can offer state-of-the-art robustness, particularly against translation, while incurring relatively low computational overhead compared with other semantic schemes. At lower watermark strength, DEW also maintains competitive text quality, suggesting that dual-embedding signals provide a promising substrate for robust semantic watermarking.

1 Introduction

Large language models (LLMs) have rapidly emerged as powerful tools that generate text with human-like fluency and are widely used in creative writing, programming, and conversational agents. However, as these models advance, distinguishing LLM-generated from human-authored text becomes increasingly challenging, with profound implications for trust, misinformation, and content attribution.

Inference-time watermarking has recently gained significant attention in both research and policy discussions. These methods introduce a hidden statistical signal into the text during generation, which a corresponding algorithm can detect.

Surface-level watermarks dynamically modify the generation process according to a predefined scheme, such as only sampling from a specific partition of the vocabulary (Kirchenbauer et al., 2023) or adding keyed exponential noise to logits (Aaronson and Kirchner, 2022). Distribution-preserving schemes have been designed to leave the output distribution unchanged under stated conditions (Hu et al., 2024; Wu et al., 2024b,a). Recent work has demonstrated that incorporating text semantics into the watermark signal computation improves resiliency to semantically invariant text modifications, typically at the cost of decreased text quality and/or increased computational overhead (Liu et al., 2024; Hou et al., 2024a).

Despite recent advances, paraphrasing and translation remain major challenges in detecting LLM-generated text. Furthermore, many watermarking schemes introduce patterns detectable by third parties, making them susceptible to reverse-engineering attacks (Jovanović et al., 2024). These patterns enable adversaries not only to identify the presence of a watermark but also to remove it systematically.

LLM-generated text is commonly rewritten to meet application-specific demands or stylistic preferences. As a result, text watermarking faces persistent challenges, including threats to content authenticity, diminished trust in AI systems, regulatory ambiguities, and difficulties in legal enforcement. This motivates watermarking methods that remain detectable after such modifications while minimizing the misclassification of human-authored texts.

In this work, we investigate dual-embedding watermark signals that incorporate not only contextual semantics but also the semantics of candidate tokens. Unlike prior methods (Hou et al., 2024a), our approach achieves this with low computational overhead and allows the signal to degrade smoothly under semantic shifts.

We present *Dual-Embedding Watermarking* (DEW), which combines two semantic embedding models to compute per-token watermark bi-

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 20266–20287
October 24-29, 2026 ©2026 Association for Computational Linguistics

ases based on the cosine similarity between token and context embeddings. This procedure adds zero-centered, pseudo-random noise to the LLM-computed logits. During detection, watermarked tokens exhibit significantly higher signal scores than unwatermarked tokens in expectation. Crucially, since the variation in watermark signals depends on the differences in token embedding vectors, semantically similar tokens receive similar signals. This property supports robustness to translation and has been largely overlooked in prior work, aside from a few exceptions (He et al., 2024; Hou et al., 2024a).

Our results show that dual-embedding watermarking can offer state-of-the-art robustness, particularly after translation. In our evaluation, DEW achieves the highest post-translation detection rates and remains competitive after paraphrasing. It also incurs relatively low computational overhead compared with other semantic schemes, while its lower-strength configuration maintains competitive text quality. Together, these results position dual-embedding watermarking as a promising substrate for robustness to meaning-preserving transformations.

The remainder of this paper is organized as follows: Section 2 reviews related work, Section 3 details the methodology, Section 4 presents experimental results, Section 5 concludes with implications, and Section 6 outlines limitations and future directions.

2 Related Work

Text watermarking is a special case of linguistic steganography that embeds a hidden signal in a passage of text. LLM watermarks are commonly evaluated along three core dimensions: *detectability*, requiring verifiability at low false-positive rates; *secrecy*, requiring no easily detectable artifacts; and *robustness*, requiring evasion to substantially modify the watermarked text, especially its semantics (Kuditipudi et al., 2024). These goals are inherently in tension: stronger detectability can reduce secrecy or robustness, while stronger secrecy, including distortion-freeness, can make detection harder. Effective LLM watermarks should also be computationally efficient and compatible with standard autoregressive decoding.

Before LLMs, text watermarking largely relied on rule-based transformations such as synonym substitution (Topkara et al., 2006) and paraphrasing (Atallah et al., 2003). Because such methods use fixed substitutions, they systematically alter text statistics, making the watermark easier to detect and remove (Tang et al., 2024; Ziegler et al., 2019). Recent LLM watermarking research instead focuses on inference-time schemes, which embed the signal directly during generation by modifying the model's token-selection process, but typically require access to model logits. Consequently, they can typically be disabled in locally hosted models.

2.1 Surface-level Watermarks

Most LLM watermarks operate at the surface level, injecting the signal based on token identities or exact token contexts without explicitly modeling semantics. These methods are simple and inexpensive, but exact context dependence makes their signals vulnerable to local edits. Kirchenbauer et al. (2023, 2024) propose a scheme, referred to here as KGW, that hashes the preceding k tokens to pseudo-randomly partition the vocabulary into green and red lists and then boosts green-list logits. Parallel unpublished work by Aaronson and Kirchner (2022), referred to as EXP, similarly hashes the previous k tokens but samples using keyed exponential noise and Perturb-and-MAP decoding (Papandreou and Yuille, 2011). Both schemes bias the distribution toward subsets of k-grams (Kuditipudi et al., 2024; Wu et al., 2024a), yielding a trade-off: larger k improves secrecy by reducing repeated contexts, whereas smaller k improves robustness by making local edits less disruptive. In the limiting case $k = 0$, KGW becomes UNIGRAM (Zhao et al., 2024), which is highly robust but vulnerable to reverse-engineering attacks (Jovanović et al., 2024). Dathathri et al. (2024) instead propose SYNTHID, which uses Tournament Sampling to optimize a secret statistical watermark score and also provides a distortion-free mode with reduced detectability. MORPHMARK (Wang et al., 2025) adapts green-list watermark strength to the model's cumulative probability mass on the green list.

Distortion-free and distribution-preserving watermarks aim to improve secrecy by avoiding changes to the output distribution. In this sense, EXP is distortion-free only when k is large enough to avoid repeated contexts. UNBIASEDWM (Hu et al., 2024) uses inverse-transform sampling and permutation-based reweighting to integrate a watermark without altering token probabilities, but its detection requires token logits and ideally an approximate reconstruction of the prompt (Wu et al., 2024b). DIPMARK (Wu et al., 2024b) instead

uses a distribution-preserving reweighting function to increase the probability mass assigned to a pseudo-random token set. Both UNBIASEDWM and DIPMARK are provably distortion-free in the absence of watermark key collisions (Wu et al., 2024a).

2.2 Semantic Watermarks

Semantic watermarks are motivated by the limited robustness of surface-level schemes against meaning-preserving transformations such as paraphrasing and translation. Rather than relying only on token hashes, they condition the watermark signal, its parameters, or its training objective on semantic representations. Some semantic watermarks, including DEW, also make each candidate token's signal depend on that token's semantics. This distinction is important because context-level semantics can stabilize the signal under paraphrasing, whereas candidate-token semantics make synonym substitutions and translations more likely to preserve token-level evidence.

TS (Huo et al., 2024) extends the green-list paradigm by learning token-specific vocabulary split ratios and green-list logit biases from the preceding-token embedding. Detection remains KGW-like via a one-sided z-test adjusted for varying split ratios. While TS improves detectability and semantic coherence over fixed-parameter green-list schemes, it does not directly assign similar watermark signals to semantically related candidate tokens.

ATW (Liu and Bu, 2024) combines entropy-gated insertion with semantic logit scaling. It leaves low-entropy decoding steps unmodified and, at selected high-entropy steps, maps embeddings of the preceding text to a logits-scaling vector. Detection approximates a likelihood-ratio test over the tokens selected by the same entropy criterion. Compared with DEW, ATW conditions the signal on context semantics but does not explicitly couple candidate-token scores to candidate-token semantics.

SIR (Liu et al., 2024) uses an auxiliary LLM to embed the preceding context and transforms these embeddings into watermark logits with a neural network trained to preserve semantic similarity while maintaining diversity and unbiasedness. This makes the watermark more stable under semantically invariant edits, but the signal is primarily context-conditioned and requires an auxiliary learned mapping in addition to the host LLM.

X-SIR (He et al., 2024) extends SIR by clustering semantically similar tokens and assigning a shared watermark bias within each cluster, making it the closest prior work to DEW because it incorporates candidate-token semantics during signal computation.

SEMSTAMP (Hou et al., 2024a) operates at sentence granularity: it embeds each generated sentence and uses rejection sampling to output only sentences whose embedding falls into an allowed locality-sensitive hashing (LSH) partition. Its successor, k-SEMSTAMP (Hou et al., 2024b), replaces LSH with k-means clustering, improving the robustness and sampling efficiency reported by its authors. We therefore evaluate k-SEMSTAMP rather than SEMSTAMP. However, k-SEMSTAMP requires domain-specific initialization and remains considerably more expensive than token-level insertion: in our setup, generation takes approximately twelve times as long as with DEW (Table 2). Moreover, in our testing, 2.1% of sentence positions reached the 100-trial rejection-sampling limit and used the final candidate without enforcing the acceptance condition.

Additional recent watermarking methods not included in our evaluation are discussed in Appendix F.

3 Methodology

LLMs have a vocabulary $\mathcal{V}$ containing words or word fragments (*tokens*). Given an input sequence $\mathbf{x} = (x_1, \ldots, x_{t-1})$, the model computes a probability distribution over $\mathcal{V}$ by producing a set of *logits* ℓ, where each logit represents the unnormalized log-probability of the corresponding token. Each token x_t is selected by sampling from this distribution or using a decoding method such as beam search. This process repeats until the LLM generates an end-of-sequence token or reaches a maximum text length.

Inference-time watermarking schemes modify probability distributions by either manipulating the sampling process or by directly adjusting the distribution, as in this work, where watermark biases are added to the candidate token logits during text generation (*watermark insertion*, Section 3.1). To introduce secrecy, this process generally employs a pseudo-random number generator (PRNG) that modifies the signal using a secret key known only to the model provider. Most schemes also require this key to determine whether a candidate text contains

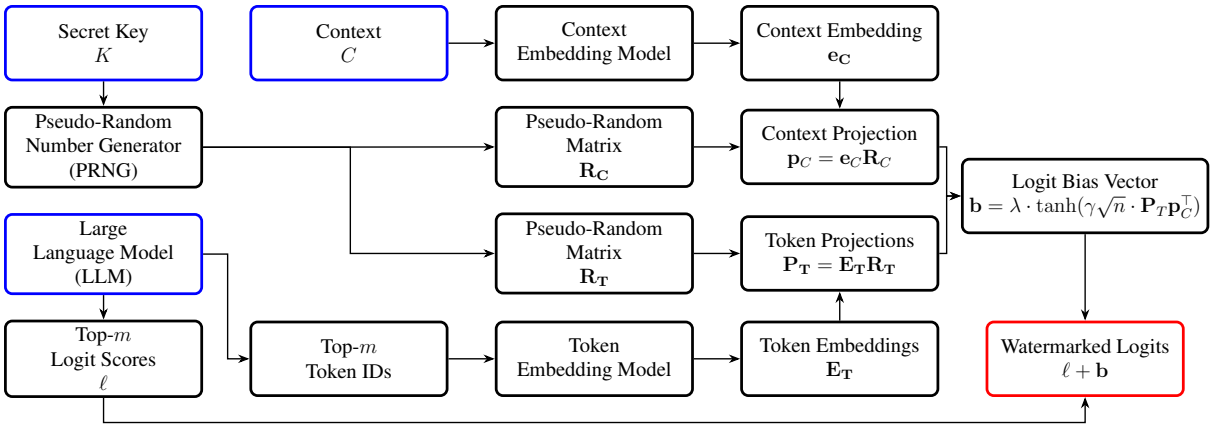


Figure 1: An illustration of the DEW insertion procedure for a single generation step. Previously generated tokens (C) are jointly embedded, while the top-m candidate token embeddings are computed separately. All embeddings are projected for obfuscation, and the dot product of the projections is added to the original logits as token-specific watermark biases. We sample from the updated logits. Inputs are highlighted in blue, and the output watermarked logits in red. For conciseness, we omitted normalization, whitening, orthonormalization, and standardization from the diagram.

the watermark, a procedure known as *watermark detection* (Section 3.2).

To improve text diversity and, in turn, secrecy, the embedded signal is typically made dependent on a sliding window of directly preceding tokens (the *watermark context*) by hashing them along with the secret key. However, due to the cryptographic nature of the hash function, even minor changes in the context yield statistically independent signals. For this reason, the robustness of such schemes decreases with larger watermark context widths, although text diversity and watermark secrecy improve.

Semantic watermarks enhance robustness against semantically invariant modifications, such as paraphrasing and translation. These schemes use a numeric representation of the context semantics to assign the same vocabulary partitioning to semantically similar contexts. This representation is commonly obtained through *embedding models*, which compute vector representations of token sequences. These models are trained, for example, via contrastive learning, to map semantically similar texts to nearby points in the embedding space. Semantic watermarking leverages this property by making the watermark signal contingent on the embedding vector.

Although semantic watermarks offer improved robustness to semantically invariant changes in the watermark context, most schemes do not consider inter-token semantic similarity when calculating the bias distributions. For this reason, substituting a token with a synonym has a high chance of removing the signal embedded in that token. DEW computes separate semantic embeddings for the context and for each candidate token to assign similar biases to tokens with close embedding vectors. Additionally, the signal carried by each token smoothly degrades with semantic shifts in either the context or the token itself, further improving robustness.

3.1 Watermark Insertion

3.1.1 Setup

In addition to the LLM, DEW incorporates two embedding models. The token embedding model M_T maps individual tokens to d_T-dimensional vectors. Similarly, the context embedding model M_C maps token sequences of arbitrary length to d_C-dimensional vectors. In our experiments, M_T is the host LLM's input-embedding table and M_C is a multilingual sentence encoder; robustness depends on the suitability of both embedding spaces (Appendix C.1).

To initialize the algorithm, a secret key K is employed to seed a cryptographically secure PRNG. For the sake of simplicity and efficiency in our experiments, we opted for the default non-secure PyTorch Philox PRNG. This generator randomly samples from the standard normal distribution to produce two matrices $\mathbf{R}_T \in \mathbb{R}^{d_T \times n}$ and $\mathbf{R}_C \in \mathbb{R}^{d_C \times n}$. Through random projections, these matrices obfuscate the embedding vectors while preserving distances.

We further introduce the value n, which we call the *projection dimensionality*. While the embedding models determine d_T and d_C, n is a tunable hyperparameter controlling the dimension of the random-projection space. We conservatively set $n = \max(d_T, d_C)$ in this work. However, by the Johnson-Lindenstrauss lemma (Johnson and Lindenstrauss, 1984), one can often choose a significantly smaller n while approximately preserving distances. Furthermore, in Appendix B, we propose an optional block-wise orthogonal construction of R_T and R_C that is *guaranteed* to preserve angles between embedding vectors while still obfuscating them through pseudo-random rotations.

3.1.2 Semantic Extraction

At each generation step, the LLM computes the logits $\ell \in \mathbb{R}^{|\mathcal{V}|}$ as usual. We use M_T to embed all subsequent candidate tokens. In practice, to reduce computational overhead, it is often sufficient to consider only the top $m \in \mathbb{N}$ tokens with the highest scores in ℓ, yielding an embedding matrix $\mathbf{E}_T \in \mathbb{R}^{m \times d_T}$. Another option is to apply nucleus sampling, which dynamically selects the smallest set of tokens whose cumulative probability exceeds a specified threshold. Each row of $\mathbf{E}_T$ is an embedding vector in $\mathbb{R}^{d_T}$ associated with one of the m highest-scoring candidate tokens.

Optionally, a whitening transformation can be applied to the token embeddings to ensure isotropy (uniformity in all directions) in the embedding space. For various applications, whitening generally makes embedding similarity metrics more meaningful and consistent across dimensions (Huang et al., 2021; Diera et al., 2025). Since sequence embeddings are typically derived by pooling individual token embeddings, it is also feasible to apply whitening before pooling. However, we only whiten token embeddings in this study.

3.1.3 Obfuscation

Next, we normalize the rows of $\mathbf{E}_T$ and multiply the result by $\mathbf{R}_T$, applying a random linear transformation to each embedding vector for obfuscation. Since the token and context embedding models are fixed (and potentially public), obfuscating the embeddings is essential to enable secrecy. We achieve this through the secret linear transformations $\mathbf{R}_T$ and $\mathbf{R}_C$. The same process is applied to the watermark context (for example, the k preceding tokens), yielding an embedding vector $\mathbf{e}_C \in \mathbb{R}^{d_C}$ and projection vector $\mathbf{p}_C = \mathbf{e}_C \mathbf{R}_C \in \mathbb{R}^n$. Notably, all token embeddings and their projections can be precomputed offline.

Algorithm 1 DEW Watermark Insertion (Single Step)

Require: LLM logits $\ell \in \mathbb{R}^{|\mathcal{V}|}$, watermark context $\mathbf{c} = (x_{t-k}, \ldots, x_{t-1})$, token embedding model M_T, context embedding model M_C, secret key K, top-m candidate count, watermark strength λ, saturation factor γ, projection dimensionality n.

Ensure: Watermarked logits ℓ'

Use K to seed a PRNG *(only once per session; can be cached)*
Regenerate (or recall) $\mathbf{R}_T \in \mathbb{R}^{d_T \times n}$ and $\mathbf{R}_C \in \mathbb{R}^{d_C \times n}$
Compute projected context embedding:
  $\mathbf{e}_C \leftarrow M_C(\mathbf{c}) \in \mathbb{R}^{d_C}$
  Normalize $\mathbf{e}_C$
  $\mathbf{p}_C \leftarrow \text{normalize}(\mathbf{e}_C \mathbf{R}_C) \in \mathbb{R}^n$
Compute (or recall) projected token embeddings:
  Let $\mathcal{T} \subseteq \mathcal{V}$ be the set of top-m tokens from ℓ
  $\mathbf{E}_T \leftarrow M_T(\mathcal{T}) \in \mathbb{R}^{m \times d_T}$
  Optional: Apply whitening to $\mathbf{E}_T$
  Normalize rows of $\mathbf{E}_T$
  $\mathbf{P}_T \leftarrow \text{row_normalize}(\mathbf{E}_T \mathbf{R}_T) \in \mathbb{R}^{m \times n}$
Compute biases and add them to logits:
  $\mathbf{b} \leftarrow \lambda \cdot \tanh\left(\gamma \sqrt{n} \cdot \mathbf{P}_T \mathbf{p}_C^\top\right) \in \mathbb{R}^m$
  Set ℓ' by adding each entry of $\mathbf{b}$ to its corresponding position in ℓ
output ℓ' *(watermarked logits for the next token)*

3.1.4 Bias Computation

We then calculate the logit bias vector $\mathbf{b}$ by taking the dot product of the context projection vector with each projected token embedding vector. Since both vectors are normalized, their dot product equals the cosine similarity, which ranges from -1 to 1 and quantifies the cosine of the angle between them. This value reflects the degree of alignment, with 1 indicating perfect alignment, -1 perfect opposition, and 0 orthogonality. Since $\mathbf{E}_T$ and $\mathbf{e}_C$ may originate from different models and are independently obfuscated through projection, the cosine similarity lacks a direct interpretive meaning. Nevertheless, it provides a useful keyed semantic alignment signal: small changes in token or context

embeddings induce controlled changes in the projected cosine score, so semantically similar continuations tend to receive similar biases. Under an isotropic spherical null model, this score is symmetric around 0 with a known Beta-type distribution (Appendix A), so text generated independently of K attains an expected score of zero. We derive the exact null distribution of the alignment score $\mathbf{p}_T\mathbf{p}_C^\top$ (Beta-type) and its high-dimensional Gaussian approximation, and use it to motivate the $\sqrt{n}$ scaling and false-positive calibration.

The variance of the dot product of two random vectors uniformly distributed on the unit sphere depends on the dimension of said vectors. In our idealized null model, the dot product of a spherical vector with any fixed unit vector has variance $1/n$. Therefore, multiplying the dot products by $\sqrt{n}$ yields an approximately unit-variance baseline score. In practice, the spherical model is an analytic baseline rather than an exact description of natural-language token statistics. We therefore use it to motivate the $\sqrt{n}$ scaling and complement it with empirical thresholding when reporting fixed-FPR detection results.

Finally, the dot products are passed through the $\tanh$ function to compute the bias vector $\mathbf{b}$:

$$\mathbf{b} = \lambda \cdot \tanh(\gamma\sqrt{n} \cdot \mathbf{P}_T\mathbf{p}_C^\top) \in \mathbb{R}^m \qquad (1)$$

Here, λ is a hyperparameter to control the watermark signal strength. Moreover, $\tanh(\cdot)$ denotes the element-wise application of the hyperbolic tangent function. Using a non-linear activation function such as $\tanh$ facilitates smooth clipping of the pre-scaling biases, mitigating the impact of outliers on text quality. By scaling the argument of $\tanh$ by a hyperparameter $\gamma \in \mathbb{R}_*^+$, the clipping level can be adjusted: larger scaling factors accentuate saturation, while smaller factors preserve a broader dynamic range of biases. Higher saturation results in more tokens receiving extreme bias values, approaching $-\lambda$ or λ. This behavior is reminiscent of the green/red list KGW watermark (Kirchenbauer et al., 2023), though KGW does not involve assigning negative biases to logits. Finally, we insert $\mathbf{b}$ into the corresponding m positions of ℓ and sample the next token from the updated logits. We describe the watermark insertion procedure for one generation step in Algorithm 1 and illustrate it in Figure 1.

Algorithm 2 DEW Watermark Detection (Single Step)

Require: Observed token x_t, watermark context $\mathbf{c} = (x_{t-k}, \ldots, x_{t-1})$, host tokenizer, compatible precomputed projected token table $\mathbf{P}_T$, context embedding model M_C, secret key K, watermark strength λ, saturation factor γ, projection dimensionality n.

Ensure: Token-level watermark score s

Use K to seed a PRNG *(only once per session; can be cached)*
Regenerate (or recall) $\mathbf{R}_C \in \mathbb{R}^{d_C \times n}$
Compute projected context embedding:
  $\mathbf{e}_C \leftarrow M_C(\mathbf{c}) \in \mathbb{R}^{d_C}$
  Normalize $\mathbf{e}_C$
  $\mathbf{p}_C \leftarrow \text{normalize}(\mathbf{e}_C\,\mathbf{R}_C) \in \mathbb{R}^n$
Look up projected token embedding:
  $\mathbf{p}_T \leftarrow \mathbf{P}_T[x_t] \in \mathbb{R}^n$
Compute watermark score:
  $s \leftarrow \lambda \cdot \tanh\Big(\gamma\sqrt{n} \cdot \mathbf{p}_T\mathbf{p}_C^\top\Big) \in \mathbb{R}$
output s *(token-level watermark score)*

3.2 Watermark Detection

The detection procedure mirrors the insertion process. It iterates over a given candidate text document token by token and sums the biases embedded in each token to obtain a document-level watermark score. This score can be thresholded for binary classification, with the threshold tunable to control the FPR.

Specifically, we tokenize the candidate text with the host tokenizer and look up the compatible precomputed projection $\mathbf{p}_T$ for each observed token x_t. For the context $\mathbf{c} = (x_{t-k}, \ldots, x_{t-1})$, we compute its embedding vector $\mathbf{e}_C = M_C(\mathbf{c})$, normalize it, and project it via $\mathbf{R}_C$ to obtain $\mathbf{p}_C$. Finally, we obtain the token bias score by computing $\mathbf{b} = \lambda \cdot \tanh(\gamma\sqrt{n} \cdot \mathbf{p}_T\mathbf{p}_C^\top) \in \mathbb{R}$, which only differs from Equation 1 in $\mathbf{p}_T$, as we now only compute the score of the observed token, instead of m candidate tokens. Thus, detection requires the host tokenizer, a compatible precomputed projected token table, the context encoder, and the secret key, but no host-model forward pass, logits, hidden states, or live host weights. The full detection procedure is described in Algorithm 2.

The detection procedure can be formalized as a statistical hypothesis test (Appendix A.3) to control FPRs rigorously and improve interpretability. The resulting empirical score distributions match the

Gaussian baseline derived in Appendix A.2. For comparability, we report TPRs at fixed FPRs in Section 4 using the standard empirical thresholding procedure from MarkLLM (Pan et al., 2024).

4 Experiments

4.1 Language Models and Hyperparameters

We use Llama-3.2-3B (Grattafiori et al., 2024) for all main experiments and additionally evaluate Gemma-7B (Gemma Team et al., 2024) and Falcon-7B (Almazrouei et al., 2023) in Appendix C.2. We generate text via multinomial sampling. To enhance text diversity, we apply a four-gram blocking constraint. This ensures that no four-token sequence that has already been generated can be repeated.

As DEW's default semantic context embedding model M_C, we choose paraphrase-multilingual-mpnet-base-v2 (Reimers and Gurevych, 2019) ($d_C = 768$) due to its multilingual paraphrase robustness. We obtain the token embeddings from the word embedding layer of the underlying LLM (for Llama-3.2-3B, $d_T = 3\,072$).

As hyperparameters for DEW, we use $m = 32$, $n = 3\,072$, and $\gamma = 0.5$ throughout all experiments with Llama-3.2-3B. Further, we apply whitening to token embeddings and take the $\tanh$ of the bias scores before scaling by λ. As we observe no significant improvement from applying orthonormalization to the random matrices with our specific models, we omit this step from the evaluation.

In Table 1, DEW uses paraphrase-multilingual-mpnet-base-v2 with context width $k = 3$. We evaluate watermark strengths $\lambda = 2.0$ and $\lambda = 2.5$; increasing λ strengthens robustness but harms text quality. The hyperparameters for all baseline watermarking schemes are provided in Appendix D.

4.2 Dataset and Prompt

For text generation, we use the C4 dataset (Raffel et al., 2020), as it is widely employed for evaluating watermarking effectiveness in high-entropy, free-form text generation tasks. From each document, we take the first 30 tokens as the prompt and generate 200 additional tokens as a completion. As counterexamples, we use unwatermarked completions generated by the same model and sampling configuration as the watermarked texts.

4.3 Detectability and Robustness Analysis

We report TPR at nominal FPRs of 1% and 5% using leave-one-seed-out calibration across three seeds. For each held-out seed, thresholds are calibrated on same-model unwatermarked completions from the other seeds and applied to 500 watermarked and 500 unwatermarked completions. We report the mean and half-range across seeds.

To evaluate robustness against paraphrasing and translation, we prompt GPT-4o-mini-2024-07-18 to rewrite the watermarked text while preserving its meaning and tone. Only watermarked positives are transformed, and the English threshold for the corresponding held-out seed is applied to each attacked set of 500 texts. Notably, GPT-4o-mini is substantially more capable than Llama-3.2-3B, which we use for text generation. The exact prompts are presented in Appendix E.

The nominal FPR is therefore controlled only for English unwatermarked completions from the same host model. Paraphrasing and translation may shift the null-score distribution, so the attacked TPRs in Table 1 should not be interpreted as operating at 1% or 5% FPR in the paraphrased, German, or French domains.

4.4 Text Quality Analysis

We report the median *perplexity* (PPL) (Jelinek et al., 1977) of the watermarked completions under the more powerful Llama-3.1-8B model (Grattafiori et al., 2024). Perplexity is the exponentiated average negative log-likelihood of the observed token sequence. While it is widely used as a simple proxy for text quality, it can assign favorable scores to repetitive or overconfident outputs that lack diversity or factual accuracy.

To complement PPL, we calculate the *Net Preference Score* (NPS) using GPT-4o-mini-2024-07-18 as an oracle. For each prompt, it compares a watermarked completion with an unwatermarked completion generated by the same model under identical settings. Each pair is judged twice with the positions swapped; a win is counted only when both orderings agree, while disagreements and equal-quality judgments are treated as ties. NPS is the difference between watermarked and unwatermarked wins divided by the total number of pairs, including ties. Positive values therefore favor watermarked completions, negative values favor unwatermarked completions, and values near zero indicate no clear

Table 1: True positive rates in unattacked, post-paraphrasing, and post-translation scenarios at nominal English same-model false positive rates of 1 and 5 percent. Values are three-seed means, with half-ranges shown in smaller type. The text quality measures are computed on unmodified watermarked text. PPL represents median perplexity, while NPS measures preference relative to unwatermarked generations. The best scores across all watermarking schemes are highlighted in bold, while the top scores within the other category (semantic/surface-level) are underlined.

	Unattacked	Paraphrased		German translation		French translation		Text quality	
Watermark	TPR@1%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	PPL ↓	NPS ↑
Semantic watermarks									
DEW ($\lambda = 2.0$)	.986 ± .008	.352 ± .065	.711 ± .019	.438 ± .066	.857 ± .031	.243 ± .069	.691 ± .039	8.94 ± 0	−.046 ± .029
DEW ($\lambda = 2.5$)	.999 ± .001	.543 ± .051	.831 ± .022	**.521** ± .077	**.904** ± .003	**.299** ± .070	**.754** ± .022	9.85 ± .06	−.108 ± .029
K-SemStamp	.819 ± .028	.241 ± .024	.467 ± .026	.001 ± .002	.019 ± .004	.009 ± .001	.031 ± .004	8.00 ± 0	**+.088** ± .037
SIR	.974 ± .002	.631 ± .033	.838 ± .018	.211 ± .023	.505 ± .021	.180 ± .023	.469 ± .028	9.79 ± .16	−.261 ± .029
X-SIR	.919 ± .014	.586 ± .031	.779 ± .009	.293 ± .037	.568 ± .016	.213 ± .029	.484 ± .017	9.54 ± .06	−.202 ± .040
ATW	**1.000** ± 0	**.646** ± .014	**.890** ± .009	.013 ± .002	.091 ± .005	.003 ± .002	.021 ± .004	11.10 ± .13	−.048 ± .017
TS	**1.000** ± 0	.592 ± .036	.806 ± .019	.129 ± .016	.306 ± .034	.066 ± .012	.180 ± .018	10.48 ± .06	−.125 ± .028
Surface-level watermarks									
MorphMark	.998 ± .002	.415 ± .050	.673 ± .018	.040 ± .019	.119 ± .011	.062 ± .022	.171 ± .019	8.63 ± 0	−.062 ± .026
KGW	.999 ± .001	.199 ± .018	.394 ± .023	.027 ± .006	.080 ± .027	.019 ± .007	.071 ± .014	10.46 ± .16	−.113 ± .032
SynthID	.999 ± .001	.423 ± .043	.607 ± .019	.023 ± .005	.061 ± .003	.025 ± .005	.069 ± .002	**6.68** ± .09	−.002 ± .060
DiPmark	.997 ± .002	.100 ± .003	.267 ± .009	.008 ± .006	.049 ± .017	.011 ± .002	.061 ± .002	9.15 ± .06	−.047 ± .018
UnbiasedWM	**1.000** ± 0	.207 ± .037	.353 ± .018	.170 ± .064	.368 ± .043	.073 ± .028	.223 ± .021	9.19 ± .13	−.089 ± .039

preference. The exact query is provided in Appendix E.3.

4.5 Evaluation

In unattacked settings, several schemes achieve near-saturated detection after 200 tokens. At 1% FPR, DEW reaches TPRs of 98.6% and 99.9% for $\lambda = 2.0$ and $\lambda = 2.5$, respectively, placing it alongside the strongest semantic and surface-level schemes.

Across paraphrasing and translation, DEW shows strong robustness. At $\lambda = 2.5$, its paraphrase TPRs are 54.3% and 83.1% at 1% and 5% FPR, respectively, placing it among the stronger semantic methods. ATW performs best under paraphrasing, with TPRs of 64.6% and 89.0%, and also outperforms DEW at $\lambda = 2.0$ at approximately matched NPS. However, this advantage comes at substantially higher computational cost (Table 2). Translation robustness is the clearest strength of DEW: at $\lambda = 2.5$, it achieves German TPRs of 52.1% and 90.4% and French TPRs of 29.9% and 75.4% at 1% and 5% FPR, respectively, the highest among the evaluated schemes.

Increasing λ from 2.0 to 2.5 improves DEW's TPR under all three transformations, while its NPS decreases from -0.046 to -0.108. Thus, the watermark strength provides a tunable robustness–quality trade-off. Overall, DEW combines competitive paraphrase robustness with the strongest post-translation detection in this evaluation, suggesting that dual-embedding watermarking is a promising substrate for semantic watermarks. A complementary analysis using human-authored English negatives is reported in Appendix C.3; DEW's translation lead is stable, whereas the leading paraphrase method at 1% FPR depends on the negative class. Appendix C also contains supplementary experiments and analyses covering ablations, secrecy, additional attacks, and additional language models.

4.6 Computational Efficiency

Table 2 compares per-text generation and detection runtimes under the experimental setup in Section 4 and Appendix D. The experiments were conducted on an Intel i9-10980XE CPU and an NVIDIA RTX A5000 GPU, which was used for generation and for detectors that support GPU acceleration.

We use the publicly available MarkLLM implementations (Pan et al., 2024), whose runtimes may differ from optimized deployments. While ATW is more robust to paraphrasing, its generation overhead is roughly six times DEW's, and its detector is over two orders of magnitude slower in this implementation. Among the semantic watermarks, DEW has the fastest detection and is among the faster generators; only TS generates faster.

5 Conclusion

This paper presents DEW, a dual-embedding watermarking scheme designed for robustness to

Table 2: Mean per-text generation and detection runtimes in seconds, with half-ranges shown in smaller type. The lowest runtimes overall are bold, while the lowest runtimes in the other category (semantic/surface-level) are underlined.

Watermark	Generation (s)	Detection (s)
Semantic watermarks		
DEW	4.822 ± 0.038	<u>.0481</u> ± 0.0003
K-SEMSTAMP	56.801 ± 1.309	.0607 ± 0.0008
SIR	7.151 ± 0.026	.2825 ± 0.0007
X-SIR	6.010 ± 0.030	.1956 ± 0.0003
ATW	10.156 ± 0.005	5.9967 ± 0.0182
TS	<u>3.896</u> ± 0.017	.1150 ± 0.0002
Surface-level watermarks		
MORPHMARK	3.925 ± 0.063	.0350 ± 0.0003
KGW	**3.800** ± 0.029	.0379 ± 0.0002
SYNTHID	4.251 ± 0.042	**.0204** ± 0.0002
DIPMARK	3.906 ± 0.032	.0598 ± 0.0002
UNBIASEDWM	3.954 ± 0.014	.2340 ± 0.0010
(no watermark)	*3.790*	–

meaning-preserving text modifications. Our results show that dual-embedding watermarking can offer state-of-the-art robustness, particularly after translation. In our evaluation, DEW achieves the highest post-translation detection rates and remains competitive after paraphrasing. At lower watermark strength, it also maintains competitive text quality and incurs relatively low computational overhead compared with other semantic watermarks. These findings suggest that dual-embedding signals provide a promising substrate for future semantic watermarking schemes and motivate further study across models, languages, and deployment settings.

6 Limitations

The reported operating points are calibrated on English unwatermarked completions. Their nominal false positive rates of 1% and 5% therefore do not imply the same false positive rates after paraphrasing or translation into German or French. Performance also depends on the watermark strength λ, the context and token embedding models, the host LLM, and the calibration domain. Our experiments cover a limited set of host models, prompt sets, languages, generation settings, and attacks; broader evaluation is needed to assess reliability across deployment settings.

The count-based watermark stealing experiment provides an initial assessment against one black-box spoofing attack that adapts to observed watermarked outputs, but is not tailored to DEW's semantic projection. Broader adaptive analyses, including attacks targeting projection recovery, watermark removal, or secret recovery, remain important directions for future work.

The current DEW design leaves several choices open, including the projection dimensionality n, $\tanh$ scaling factor γ, embedding models, whitening transformations, and orthogonalization. Stronger or specialized embedding models may broaden language and host-model coverage. Finally, DEW's applicability to instructed dialogue systems and low-entropy settings, including code generation, warrants further study, as do broader benchmarks and user studies assessing perceived quality, factual accuracy, creativity, and relevance.

Ethical Considerations

This research advances watermarking methods for distinguishing LLM-generated text from human-authored content, with a focus on robustness to semantic transformations. More robust watermarks can support provenance and accountability when generated text is paraphrased or translated.

Deploying watermarks for LLM-generated text can support provenance and accountability, but it also risks false attribution of human-written text, overreliance in high-stakes moderation or legal settings, uneven reliability across languages and writing styles, and adversarial escalation through evasion, removal, or spoofing attacks.

We identify no substantive risks associated with the publication of our watermarking algorithm, and our contribution is purely methodological. The threat models we evaluate are standard in the watermarking literature and can be executed using publicly available tools. Consequently, disclosing our method does not introduce any new adversarial capabilities beyond those already well known in existing watermarking frameworks.

Acknowledgments

This work was supported by the German Federal Ministry of Education and Research (BMBF) under the programme *Souverän. Digital. Vernetzt.* through the joint project *AIgenCY: Chances and Risks of Generative AI in Cybersecurity* (project identification number 16KIS2013). GW was additionally supported by the BMBF joint project *6G-RIC: 6G Research and Innovation Cluster* (project identification number 16KISK025).

References

Scott Aaronson and Hendrik Kirchner. 2022. Watermarking GPT outputs. PowerPoint slides, presented at the University of Texas at Austin.

Zhenxin Ai and Haiyun He. 2026. PASA: A principled embedding-space watermarking approach for LLM-generated text under semantic-invariant attacks. *Preprint*, arXiv:2605.10977.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models. *CoRR*, abs/2311.16867.

Mikhail J. Atallah, Victor Raskin, Christian Hempelmann, Mercan Karahan, Radu Sion, Umut Topkara, and Katrina E. Triezenberg. 2003. Natural language watermarking and tamperproofing. In *Information Hiding: 5th International Workshop, IH 2002*, volume 2578 of *Lecture Notes in Computer Science*, pages 196–212, Berlin, Heidelberg. Springer.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel, Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, and 5 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823.

Andor Diera, Lukas Galke, and Ansgar Scherp. 2025. Isotropy matters: Soft-ZCA whitening of embeddings for semantic code search. In *ESANN 2025 Proceedings: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 627–632. Ciaco - i6doc.com.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, and 89 others. 2024. Gemma: Open models based on gemini research and technology. *CoRR*, abs/2403.08295.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 540 others. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4115–4129, Bangkok, Thailand. Association for Computational Linguistics.

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. 2024a. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4067–4082, Mexico City, Mexico. Association for Computational Linguistics.

Abe Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. 2024b. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1706–1715, Bangkok, Thailand. Association for Computational Linguistics.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. 2024. Unbiased watermark for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. International Conference on Learning Representations.

Baizhou Huang and Xiaojun Wan. 2025. WaterPool: A language model watermark mitigating trade-offs among imperceptibility, efficacy and robustness. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4156–4182, Albuquerque, New Mexico. Association for Computational Linguistics.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. 2021. WhiteningBERT: An easy unsupervised sentence embedding approach. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 238–244, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mingjia Huo, Sai Ashish Somayajula, Youwei Liang, Ruisi Zhang, Farinaz Koushanfar, and Pengtao Xie. 2024. Token-specific watermarking with enhanced detectability and semantic coherence for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 20746–20767. PMLR.

F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker. 1977. Perplexity–a measure of the difficulty of

speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63.

Yibo Jiang, Goutham Rajendran, Pradeep Kumar Ravikumar, Bryon Aragam, and Victor Veitch. 2024. On the origins of linear representations in large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 21879–21911. PMLR.

William B. Johnson and Joram Lindenstrauss. 1984. Extensions of lipschitz mappings into a hilbert space. *Contemporary Mathematics*, 26:189–206.

Nikola Jovanović, Robin Staab, and Martin Vechev. 2024. Watermark stealing for language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 22570–22593. PMLR.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. 2024. On the reliability of watermarks for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. International Conference on Learning Representations.

Shingo Kodama, Haya Diwan, Lucas Rosenblatt, R. Teal Witter, and Niv Cohen. 2025. SimKey: A semantically aware key module for watermarking language models. *Preprint*, arXiv:2510.12828.

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

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2024. Robust distortion-free watermarks for language models. *Transactions on Machine Learning Research*.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. 2024. A semantic invariant robust watermark for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. International Conference on Learning Representations.

Yepeng Liu and Yuheng Bu. 2024. Adaptive text watermark for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 30718–30737. PMLR.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, Irwin King, and Philip S. Yu. 2024. MarkLLM: An open-source toolkit for LLM watermarking. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 61–71, Miami, Florida, USA. Association for Computational Linguistics. Apache License 2.0.

George Papandreou and Alan L. Yuille. 2011. Perturb-and-MAP random fields: Using discrete optimization to learn and sample from energy models. In *2011 International Conference on Computer Vision*, pages 193–200.

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

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2024. The science of detecting llm-generated text. *Commun. ACM*, 67(4):50–59.

Umut Topkara, Mercan Topkara, and Mikhail J. Atallah. 2006. The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th Workshop on Multimedia and Security*, Mm&sec '06, page 164–174, New York, NY, USA. Association for Computing Machinery.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual E5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

Zongqi Wang, Tianle Gu, Baoyuan Wu, and Yujiu Yang. 2025. MorphMark: Flexible adaptive watermarking for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4842–4860, Vienna, Austria. Association for Computational Linguistics.

Yihan Wu, Ruibo Chen, Zhengmian Hu, Yanshuo Chen, Junfeng Guo, Hongyang Zhang, and Heng Huang. 2024a. Distortion-free watermarks are not truly distortion-free under watermark key collisions. *CoRR*, abs/2406.02603.

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. 2024b. A resilient and accessible distribution-preserving watermark for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 53443–53470. PMLR.

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. 2024. Provable robust watermarking for ai-generated text. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. International Conference on Learning Representations.

Zachary Ziegler, Yuntian Deng, and Alexander Rush. 2019. Neural linguistic steganography. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1210–1215. Association for Computational Linguistics.

A Geometry and Statistical Foundations of DEW

Reader map. Section A.1 frames next-token prediction as the composition of a *context representation* map and a *token scoring* (unembedding) map, and explains why DEW mirrors this structure via keyed signal processing on embeddings. Section A.2 derives the null distributions of the core alignment score, emphasizing that the exact law is Beta-type while a Gaussian approximation emerges in high dimensions. Section A.3 states a concise one-sided hypothesis test for watermark detection and clarifies the approximation points.

A.1 Inner-product geometry of next-token prediction

Let $x_{<t}$ denote the prefix and w a candidate next token. Decoder-only LLMs can be abstracted as two coupled maps: a *context map* that builds a representation of the prefix, and a typically near-linear *token scoring* (unembedding) map that produces next-token logits,

$$\begin{aligned} h_t &= f_\theta(x_{<t}) \in \mathbb{R}^d, \\ \ell_t(w) &\approx \langle W_U[w],\, h_t \rangle + b_w, \\ \mathbb{P}(w \mid x_{<t}) &= \mathrm{softmax}(\ell_t)(w). \end{aligned}$$

This factorization makes clear why inner products are a natural primitive for next-token selection. Moreover, recent theory suggests that training pressure can make latent variables ("concept" directions) linearly accessible in representation space (Jiang et al., 2024). Thus, small perturbations expressed as controlled linear scores can interact smoothly with semantics and degrade gracefully under semantic shifts.

DEW mirrors this structure externally (without access to the model's internal residual stream) using embedding models and keyed linear maps. For each position i, let $\mathbf{c}^{(i)} = (x_{i-w}, \ldots, x_{i-1})$ be the context window and define

$$\mathbf{e}_C^{(i)} = M_C(\mathbf{c}^{(i)}) \in \mathbb{R}^{d_C}, \qquad \hat{\mathbf{e}}_C^{(i)} = \frac{\mathbf{e}_C^{(i)}}{\|\mathbf{e}_C^{(i)}\|},$$

$$\mathbf{p}_C^{(i)} = \frac{\hat{\mathbf{e}}_C^{(i)} \mathbf{R}_C}{\|\hat{\mathbf{e}}_C^{(i)} \mathbf{R}_C\|} \in \mathbb{R}^n.$$

Similarly, let $\mathbf{e}_T^{(i)} \in \mathbb{R}^{d_T}$ be the whitened token embedding of x_i and $\hat{\mathbf{e}}_T^{(i)} = \frac{\mathbf{e}_T^{(i)}}{\|\mathbf{e}_T^{(i)}\|}$; define

$$\mathbf{p}_T^{(i)} = \frac{\hat{\mathbf{e}}_T^{(i)} \mathbf{R}_T}{\|\hat{\mathbf{e}}_T^{(i)} \mathbf{R}_T\|} \in \mathbb{R}^n.$$

The core alignment score is the cosine similarity, expressed as the dot product

$$Z^{(i)} := \mathbf{p}_T^{(i)} (\mathbf{p}_C^{(i)})^\top \in [-1, 1],$$

which is then mapped (monotonically, e.g., via $\tanh$) to a logit bias. Hence, DEW can be viewed as a keyed signal-processing layer that injects a small, structured logit bias consistent with the inner-product geometry underlying next-token prediction.

A.2 Underlying distributions of the alignment score

This section characterizes the baseline distribution of the dot product $Z := \mathbf{p}_T \mathbf{p}_C^\top \in [-1, 1]$ (and its scaled version $\sqrt{n}\, Z$), which underpins both watermark insertion and detection, and is used to calibrate false-positive control in Section A.3.

Setup and what is (approximately) known. Token embeddings are whitened in preprocessing, making $M_T(x)$ approximately isotropic for tokens $x \in \mathcal{V}$. After a key-seeded random projection and normalization, it is reasonable to model $\mathbf{p}_T$ as approximately uniform on the unit sphere $\mathbf{S}^{n-1}$. Context embeddings $\mathbf{e}_C = M_C(\mathbf{c})$ are not generally isotropic (they often lie in an anisotropic cone that reflects semantic constraints), so $\mathbf{p}_C$ need not be uniform. Crucially, for this baseline distribution, it suffices that, conditional on the context projection $\mathbf{p}_C$, the token projection $\mathbf{p}_T$ is approximately uniform on $\mathbf{S}^{n-1}$.

Assumption A.1 (Conditionally spherical token projection (baseline))**.** In the non-watermarked regime used for false-positive calibration, for each position i, conditional on the context projection $\mathbf{p}_C^{(i)}$, the normalized projected token vector satisfies $\mathbf{p}_T^{(i)} \approx \mathcal{U}(\mathbf{S}^{n-1})$.

Why Assumption A.1 is plausible under a fixed key. Although the secret key fixes $\mathbf{R}_T$ for all documents, randomness remains through the token sequence under H_0. For a freshly sampled Gaussian projection and any fixed unit embedding vector, the projected vector is spherical before normalization. In deployment, however, the key is fixed, so this key-averaged sphericality becomes an approximation over the empirical distribution of tokens and documents. Whitening and normalization make this approximation more plausible by reducing dominant anisotropic directions in the token embeddings, but they do not make the fixed-key null exactly spherical. Residual deviations from the spherical model can be handled by conservative calibration, e.g., using L_{eff} or empirical null estimation.

Exact law (Beta-type).

Lemma A.2 (Dot product with a spherical vector)**.** *Let $Y \sim \mathcal{U}(\mathbf{S}^{n-1})$ and let $x \in \mathbf{S}^{n-1}$ be any fixed unit vector. Then $x^\top Y$ has density*

$$f(z) = \frac{\Gamma\left(\frac{n}{2}\right)}{\sqrt{\pi}\,\Gamma\left(\frac{n-1}{2}\right)}(1-z^2)^{\frac{n-3}{2}}, \qquad z \in [-1,1],$$

Equivalently, $\frac{1+z}{2} \sim \mathrm{Beta}\left(\frac{n-1}{2}, \frac{n-1}{2}\right)$. In particular, $\mathbb{E}[x^\top Y] = 0$ and $\mathrm{Var}(x^\top Y) = \frac{1}{n}$.

Applying the lemma conditionally with $Y = \mathbf{p}_T$ and $x = \mathbf{p}_C$ (treating $\mathbf{p}_C$ as fixed or slowly varying) yields an exact description of Z in the baseline regime as long as $\mathbf{p}_T$ is (approximately) uniform on the sphere, regardless of whether $\mathbf{p}_C$ is anisotropic.

Lemma A.3 (High-dimensional Gaussian approximation)**.** *Under Assumption A.1, let $Z := \mathbf{p}_T \mathbf{p}_C^\top \in [-1,1]$ with $\mathbf{p}_C \in \mathbf{S}^{n-1}$ treated as fixed (or conditioned upon). Then, as $n \to \infty$,*

$$\sqrt{n}\, Z \xrightarrow{d} \mathcal{N}(0,1).$$

Lemma A.3 motivates scaling by $\sqrt{n}$ so that the per-token score has approximately unit variance in the baseline regime.

A.3 One-sided hypothesis test for watermark detection

DEW's detector can be interpreted as a one-sided hypothesis test with false-positive control. For brevity, we first present the linear (non-saturated) statistic; the $\tanh$ nonlinearity is discussed at the end.

Let $b_1, \ldots, b_L$ be token-level bias scores for a document (Algorithm 2):

$$b_i \;=\; \lambda\sqrt{n}\left(\mathbf{p}_T^{(i)}(\mathbf{p}_C^{(i)})^\top\right) \in \mathbb{R}, \qquad \bar{b} \;=\; \frac{1}{L}\sum_{i=1}^{L} b_i.$$

Null H_0 (not watermarked). Under H_0, token selection is not influenced by the key. Assumption A.1 formalizes the resulting spherical model for $\mathbf{p}_T^{(i)}$, and Lemma A.2 yields the exact per-token dot-product law. Using Section A.2 with $\mathbf{p}_T^{(i)} \approx \mathcal{U}(\mathbf{S}^{n-1})$, we have (conditionally on $\mathbf{p}_C^{(i)}$)

$$\mathbb{E}[b_i] = 0, \qquad \mathrm{Var}(b_i) = \lambda^2,$$

and the exact single-token distribution is $\lambda\sqrt{n}\,Z$ where Z is Beta-type on $[-1,1]$. For document-level inference, we use a CLT approximation: if (b_i) are independent or weakly dependent with an effective sample size $L_{\text{eff}} \leq L$, then

$$\bar{b} \;\approx\; \mathcal{N}\!\left(0, \frac{\lambda^2}{L_{\text{eff}}}\right).$$

(Practically, L_{eff} can be set to L under an i.i.d. approximation, or conservatively reduced to account for correlations across nearby tokens.)

Alternative H_1 (watermarked with DEW). Under H_1, DEW biases token probabilities toward larger alignments $\mathbf{p}_T^{(i)}(\mathbf{p}_C^{(i)})^\top$, inducing a positive mean shift:

$$\mathbb{E}[b_i] = \mu > 0,$$

and thus $\mathbb{E}[\bar{b}] = \mu > 0$ while the variance remains comparable for small watermark strengths.

Test statistic and rejection rule. We test $H_0 : \mu \leq 0$ against $H_1 : \mu > 0$ using

$$Z_L \;=\; \frac{\bar{b}}{\frac{\lambda}{\sqrt{L_{\text{eff}}}}} \;=\; \frac{\sqrt{L_{\text{eff}}}\,\bar{b}}{\lambda}.$$

Under H_0, $Z_L \approx \mathcal{N}(0,1)$, and the one-sided p-value is $p = 1 - \Phi(Z_L)$. At significance level α, reject H_0 if $Z_L > z_\alpha$ (equivalently $p < \alpha$), where z_α is the $(1-\alpha)$-quantile of the standard normal.

Analytic classification threshold and empirical agreement. The rejection rule $Z_L > z_\alpha$ is equivalent to an analytic threshold on the document score,

$$\bar{b} \;>\; \tau_\alpha \qquad \text{with} \qquad \tau_\alpha \;:=\; \frac{\lambda}{\sqrt{L_{\text{eff}}}}\, z_\alpha,$$

which yields a closed-form decision boundary for any target false-positive rate α under the Gaussian null approximation.

Remark A.4 (Bounded nonlinearity). If the detector uses the saturated score $b_i = \lambda \tanh(\gamma\sqrt{n}\,\mathbf{p}_T^{(i)}(\mathbf{p}_C^{(i)})^\top)$, then b_i is bounded and symmetric under H_0. Moreover, in the linear regime where $|\gamma\sqrt{n}\,Z| \ll 1$ and $\tanh(u) \approx u$, we have $b_i \approx \lambda\gamma\sqrt{n}\,Z$, hence $\mathrm{Var}(b_i) \approx (\lambda\gamma)^2$ under H_0. The same test structure applies by replacing λ^2 with $\mathrm{Var}(b_i)$ under H_0, which can be estimated empirically (or approximated using the Gaussian limit for $\sqrt{n}\,Z$).

B Orthogonal Construction of Random Matrices

In the following, we propose an optional block-wise row-orthonormal construction of the random projection matrices $\mathbf{R}_T$ and $\mathbf{R}_C$. Although such orthonormality is not mandatory for the functionality of DEW, this construction preserves inner products, norms, and therefore angles within each embedding space after projection.

We select the *projection dimensionality* n as the least common multiple of d_T and d_C to ensure the existence of integers k_T and k_C such that

$$\frac{n}{d_T} = k_T \quad \text{and} \quad \frac{n}{d_C} = k_C. \tag{2}$$

This choice allows $\mathbf{R}_T$ and $\mathbf{R}_C$ to be constructed by concatenating k_T and k_C square orthogonal blocks, respectively, followed by an appropriate scaling. The resulting matrices satisfy

$$\mathbf{R}_T\mathbf{R}_T^\top = \mathbf{I}_{d_T} \quad \text{and} \quad \mathbf{R}_C\mathbf{R}_C^\top = \mathbf{I}_{d_C}.$$

Thus, each projection is an isometric embedding into $\mathbb{R}^n$. In particular, if the input embeddings are normalized before projection, then their projected representations are already normalized, so no additional post-projection normalization is required for norm preservation.

We describe the construction for $\mathbf{R}_C$; the construction for $\mathbf{R}_T$ is analogous.

- Generate random matrices $\widetilde{\mathbf{R}}_j \in \mathbb{R}^{d_C \times d_C}$ for $j = 1, \ldots, k_C$.
- Orthonormalize each matrix, for example via QR decomposition or Gram–Schmidt, to obtain orthogonal matrices $\mathbf{Q}_j \in \mathbb{R}^{d_C \times d_C}$ satisfying

$$\mathbf{Q}_j \mathbf{Q}_j^\top = \mathbf{Q}_j^\top \mathbf{Q}_j = \mathbf{I}_{d_C}.$$

- Build

$$\mathbf{R}_C \coloneqq \frac{1}{\sqrt{k_C}} \big[\mathbf{Q}_1 \mid \mathbf{Q}_2 \mid \cdots \mid \mathbf{Q}_{k_C}\big] \in \mathbb{R}^{d_C \times n}.$$

  Equivalently, $\mathbf{R}_C$ has orthonormal rows:

$$\mathbf{R}_C \mathbf{R}_C^\top = \mathbf{I}_{d_C}.$$

- For any vector $\mathbf{v} \in \mathbb{R}^{1 \times d_C}$, the projection is

$$\mathbf{v}\mathbf{R}_C = \frac{1}{\sqrt{k_C}} \big[\mathbf{v}\mathbf{Q}_1 \mid \mathbf{v}\mathbf{Q}_2 \mid \cdots \mid \mathbf{v}\mathbf{Q}_{k_C}\big].$$

  Since each $\mathbf{Q}_j$ is orthogonal, $\|\mathbf{v}\mathbf{Q}_j\|_2 = \|\mathbf{v}\|_2$. Therefore,

$$\|\mathbf{v}\mathbf{R}_C\|_2^2 = \frac{1}{k_C} \sum_{j=1}^{k_C} \|\mathbf{v}\mathbf{Q}_j\|_2^2 = \|\mathbf{v}\|_2^2.$$

- More generally, for any $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{1 \times d_C}$,

$$(\mathbf{u}\mathbf{R}_C)(\mathbf{v}\mathbf{R}_C)^\top = \frac{1}{k_C} \sum_{j=1}^{k_C} \mathbf{u}\mathbf{Q}_j \mathbf{Q}_j^\top \mathbf{v}^\top = \mathbf{u}\mathbf{v}^\top.$$

  Thus, right multiplication by $\mathbf{R}_C$ is an isometric embedding from $\mathbb{R}^{d_C}$ into $\mathbb{R}^n$, preserving norms, inner products, and angles within the context-embedding space.

Table 3: Semantic-mode ablation of DEW under English same-model calibration. Values are means across three runs, with half-ranges shown in smaller type. Best values are bold.

	Unattacked	Paraphrased		German translation		French translation		Text quality	
Semantic mode	TPR@1%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	PPL ↓	NPS ↑
`both` (default)	.986 ± .008	.352 ± .065	.711 ± .019	**.438** ± .066	**.857** ± .031	**.243** ± .069	**.691** ± .039	8.938 ± .000	−.046 ± .029
`context_only`	**.999** ± .001	**.570** ± .022	**.797** ± .009	.043 ± .014	.153 ± .022	.025 ± .008	.083 ± .002	8.708 ± .063	−.073 ± .034
`token_only`	**.999** ± .001	.123 ± .023	.356 ± .003	.005 ± .001	.047 ± .004	.008 ± .000	.055 ± .005	**8.667** ± .063	**+.002** ± .029
`neither`	**.999** ± .001	.200 ± .023	.394 ± .016	.009 ± .003	.046 ± .002	.012 ± .002	.049 ± .009	8.875 ± .094	−.023 ± .035

C Supplementary Experiments

C.1 Ablation Study

To isolate the roles of token- and context-level semantics, we evaluate four variants of DEW:

- `both`: unmodified DEW (baseline).
- `context_only`: token semantics are removed by randomly permuting the whitened token projections at initialization.
- `token_only`: context semantics are removed by replacing the context projection with a pseudo-random unit vector seeded by the context token IDs.
- `neither`: both ablations are applied simultaneously.

Table 3 reports results using the default paraphrase-multilingual-mpnet-base-v2 context encoder, $\lambda = 2.0$, token-level score aggregation, and English same-model leave-one-seed-out calibration. The remaining hyperparameters follow Section 4.1. All four modes remain nearly saturated without attack, so their differences emerge primarily after transformation.

Paraphrasing. The `context_only` variant outperforms the full method at both operating points consistently across all three runs. Score diagnostics indicate that it begins with a stronger effective watermark at the same numerical λ and remains more separable after paraphrasing, rather than retaining a larger fraction of its initial signal. We hypothesize that permuting the token projections changes the watermark-score distribution over likely candidate tokens, but do not test this mechanism directly.

Translation. The full method retains substantially more signal after German and French translation than any ablated variant. Both token and context semantics are therefore needed for the full cross-lingual signal in this setting.

Context encoder. We compare the default paraphrase-multilingual-mpnet-base-v2 (Reimers and Gurevych, 2019) context encoder with multilingual-e5-base (Wang et al., 2024) at two watermark strengths. Table 4 reports AUROC against same-model unwatermarked completions in the corresponding condition. We use AUROC because it directly measures how well each encoder separates watermarked from unwatermarked scores across the full range of detection thresholds.

At equal λ, E5 improves robustness, while its NPS is lower than MPNet's at the same strength. The closest observed quality points are MPNet at $\lambda = 2.5$ and E5 at $\lambda = 2.0$. Their robustness is broadly comparable, although E5 retains a modest AUROC advantage under each attack. Since these points are only approximately matched in quality, this does not establish encoder equivalence.

The same λ need not yield the same effective watermark strength across context encoders and should therefore be selected separately under a fixed quality budget. We hypothesize that this difference results from encoder-dependent similarity geometry or isotropy, but the present experiments do not test this mechanism directly.

C.2 Cross-Host Token Embedding Transfer

We evaluate DEW on Gemma-7B (Gemma Team et al., 2024) and Falcon-7B (Almazrouei et al., 2023) using the default paraphrase-multilingual-mpnet-base-v2 (Reimers and Gurevych, 2019) context encoder, $\lambda = 2.0$, and $\gamma = 0.5$. The native configurations use each host model's input-embedding table and the corresponding whitening transform. In the donor configurations, each host token is decoded and re-tokenized with the Llama-3.2-3B (Grattafiori et al., 2024) tokenizer, represented by the mean of the resulting input embeddings, and processed using the Llama-3.2-3B whitening transform. The host model, logits, and tokenizer used for generation remain unchanged.

All rows in Table 5 use token-level score aggregation and host-specific English same-model leave-one-seed-out calibration. Only the watermarked positives are paraphrased or translated. We report means and half-ranges across three runs.

The donor table substantially improves German translation retention on both hosts at both operating points. Its effect in French is host-dependent: the change is small on Gemma-7B but more pronounced on Falcon-7B. Conversely, the native token tables retain more signal after paraphrasing, especially on Falcon-7B.

Text quality also changes in opposite directions across hosts. On Gemma-7B, the donor configuration raises PPL but improves NPS, whereas on Falcon-7B it lowers PPL but reduces NPS. There is therefore no host-independent donor quality effect.

These results suggest that mapping host vocabularies into a fixed donor token space can improve translation retention across hosts. However, the benefit depends on the host and language and does

Table 4: Context-encoder sensitivity of DEW under token-level score aggregation. Values are means across three runs, with half-ranges shown in smaller type. Best values are bold.

		AUROC ↑				
Context encoder	λ	Clean	Paraphrase	German	French	NPS ↑
MPNet	2.0	0.9994 ± .0006	0.9212 ± .0071	0.7079 ± .0117	0.7232 ± .0093	**−0.046** ± .029
MPNet	2.5	0.9999 ± .0001	0.9537 ± .0008	0.7560 ± .0101	0.7650 ± .0014	−0.108 ± .029
E5	2.0	0.9998 ± .0002	0.9774 ± .0014	0.7936 ± .0062	0.7960 ± .0070	−0.121 ± .018
E5	2.5	**1.0000** ± .0000	**0.9907** ± .0033	**0.8392** ± .0039	**0.8409** ± .0103	−0.217 ± .039

Table 5: Cross-host comparison of native and fixed Llama-3.2-3B donor token tables under host-specific English same-model calibration. Values are means across three runs, with half-ranges shown in smaller type.

	Unattacked	Paraphrased		German translation		French translation		Text quality	
Host	TPR@1%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	PPL ↓	NPS ↑
Native token embeddings									
Gemma-7B	.981 ± .001	.371 ± .054	.673 ± .029	.039 ± .011	.119 ± .012	.179 ± .049	.487 ± .036	12.125 ± .188	−.051 ± .015
Falcon-7B	.995 ± .002	.504 ± .049	.707 ± .028	.056 ± .013	.161 ± .012	.088 ± .016	.277 ± .017	12.813 ± .094	−.021 ± .008
Llama-3.2-3B embeddings									
Gemma-7B	.961 ± .005	.341 ± .069	.617 ± .048	.506 ± .060	.792 ± .029	.187 ± .049	.495 ± .048	12.313 ± .094	+.012 ± .014
Falcon-7B	.974 ± .012	.336 ± .057	.595 ± .049	.476 ± .078	.808 ± .044	.179 ± .026	.502 ± .043	12.625 ± .094	−.065 ± .014

not extend uniformly to paraphrasing or text quality. We therefore view donor-token construction as a promising but conditional transfer mechanism rather than a uniformly superior configuration.

C.3 Sensitivity to Human-Authored English Negatives

To examine sensitivity to the negative class, we repeat the primary evaluation using English human-authored texts instead of same-model unwatermarked completions for calibration. This is the only protocol change: Table 6 retains the same positives, attacks, configurations, quality measurements, token-level detection, and leave-one-seed-out procedure as Table 1.

The main translation result is stable under this alternative negative class, with DEW at $\lambda = 2.5$ remaining the strongest method after German and French translation. The precise paraphrase ranking at 1% FPR is sensitive to the negative class: SIR ranks first here, whereas ATW ranks first with same-model negatives; ATW remains strongest at 5% FPR in both evaluations.

C.4 Robustness to Lexical Edits

This auxiliary analysis uses the semantic watermark configurations from Table 1 and reports single-seed point estimates without uncertainty intervals. Each cell evaluates 500 attacked watermarked texts using 500 English human-authored texts as the negative class for calibration and evaluation. Each scheme uses its own detector, and DEW uses token-level score aggregation.

Table 7 evaluates robustness to lightweight lexical edits. Word deletion randomly removes whitespace-separated words with probability r, whereas context-aware synonym substitution first selects words with WordNet entries and then replaces masked positions with the top prediction of BERT-large-uncased. Thus, the latter should be interpreted as a contextual masked-token substitution rather than a strictly synonym-constrained transformation.

DEW remains highly robust across both lexical attacks. At $\lambda = 2.5$, it achieves or ties the highest TPR in five of twelve columns and remains close to the leading semantic baseline in most others, including under 50% edits. Although robustness decreases as the edit ratio increases, both DEW configurations generally achieve higher TPRs under contextual synonym substitution than under deletion.

C.5 Robustness to DIPPER Paraphrasing

DIPPER (Krishna et al., 2023) is an 11B-parameter paraphrase model trained to evade detectors for LLM-generated text, including watermarks. It uses surrounding context and controls lexical diversity and content reordering while aiming to preserve meaning.

Table 6: True positive rates using English human-authored texts as the negative class for calibration and evaluation. Values are three-seed means, with half-ranges shown in smaller type. The best scores overall are bold, while the best scores in the other category are underlined.

	Unattacked	Paraphrased		German translation		French translation		Text quality	
Watermark	TPR@1%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	PPL ↓	NPS ↑
Semantic watermarks									
DEW ($\lambda = 2.0$)	.993 ± .007	.430 ± .033	.735 ± .019	.555 ± .013	.882 ± .015	.337 ± .019	.725 ± .020	8.94 ± 0	−.046 ± .029
DEW ($\lambda = 2.5$)	.999 ± .001	.627 ± .024	.843 ± .009	**.625** ± .006	**.914** ± .002	**.393** ± .015	**.787** ± .018	9.85 ± .06	−.108 ± .029
K-SEMSTAMP	.835 ± .017	.265 ± .006	.519 ± .051	.003 ± .003	.026 ± .005	.012 ± .005	.037 ± .008	8.00 ± 0	**+.088** ± .037
SIR	.987 ± .004	**.733** ± .023	.878 ± .010	.321 ± .046	.591 ± .011	.295 ± .032	.561 ± .015	9.79 ± .16	−.261 ± .029
X-SIR	.949 ± .002	.701 ± .003	.814 ± .003	.429 ± .011	.615 ± .024	.338 ± .016	.533 ± .010	9.54 ± .06	−.202 ± .040
ATW	**1.000** ± 0	.713 ± .012	**.907** ± .001	.020 ± .003	.119 ± .004	.003 ± .003	.026 ± .002	11.10 ± .13	−.048 ± .017
TS	**1.000** ± 0	.671 ± .022	.799 ± .012	.167 ± .031	.298 ± .022	.092 ± .009	.172 ± .017	10.48 ± .06	−.125 ± .028
Surface-level watermarks									
MORPHMARK	.998 ± .002	.476 ± .014	.699 ± .023	.047 ± .016	.132 ± .006	.075 ± .017	.188 ± .014	8.63 ± 0	−.062 ± .026
KGW	.999 ± .001	.204 ± .035	.387 ± .015	.026 ± .006	.071 ± .015	.020 ± .003	.067 ± .014	10.46 ± .16	−.113 ± .032
SYNTHID	.999 ± .001	.351 ± .067	.569 ± .016	.016 ± .008	.051 ± .004	.019 ± .005	.057 ± .008	**6.68** ± .09	−.002 ± .060
DIPMARK	.998 ± .002	.139 ± .023	.308 ± .013	.017 ± .011	.069 ± .011	.020 ± .008	.075 ± .012	9.15 ± .06	−.047 ± .018
UNBIASEDWM	**1.000** ± 0	.183 ± .011	.329 ± .013	.129 ± .030	.331 ± .016	.051 ± .009	.199 ± .009	9.19 ± .13	−.089 ± .039

Table 7: True positive rates after word deletion and synonym substitution for the semantic watermark configurations evaluated in Table 1. English human-authored texts serve as the negative class for calibration and evaluation. The best scores are highlighted in bold.

	Word deletion						Synonym substitution					
	10%		30%		50%		10%		30%		50%	
Watermark	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%
DEW ($\lambda = 2.0$)	0.982	**1.000**	0.880	0.968	0.656	0.846	0.984	0.998	0.920	0.984	0.784	0.940
DEW ($\lambda = 2.5$)	0.998	0.998	0.964	**0.994**	0.852	0.946	0.998	**1.000**	**0.990**	**0.998**	0.920	**0.984**
K-SEMSTAMP	0.470	0.680	0.090	0.252	0.026	0.070	0.718	0.886	0.492	0.734	0.316	0.582
SIR	0.966	0.984	0.940	0.978	0.904	0.944	0.966	0.990	0.932	0.984	0.878	0.952
X-SIR	0.936	0.972	0.928	0.966	**0.914**	**0.952**	0.920	0.962	0.864	0.944	0.824	0.914
ATW	0.996	**1.000**	0.860	0.962	0.638	0.842	**1.000**	**1.000**	0.974	**0.998**	0.834	0.946
TS	**1.000**	**1.000**	**0.980**	0.992	0.808	0.930	**1.000**	**1.000**	**0.990**	0.996	**0.948**	0.980

This auxiliary analysis uses the semantic watermark configurations from Table 1 and reports single-seed point estimates without uncertainty intervals. Each cell evaluates 500 DIPPER-paraphrased watermarked texts using 500 English human-authored texts as the negative class for calibration and evaluation, with token-level score aggregation for DEW. We denote the lexical- and order-diversity parameters by *ld* and *od*, respectively.

At $\lambda = 2.5$, DEW is competitive at 5% FPR, with TPRs within 0.012–0.090 of the strongest method across the four settings. At the stricter 1% FPR, it ties SIR under $(ld = 40, od = 0)$ but generally trails the strongest schemes, particularly under the more aggressive paraphrasing settings.

C.6 Secrecy Evaluation via Watermark Stealing

We evaluate secrecy against the watermark stealing attack of Jovanović et al. (2024), as implemented in MarkLLM (Pan et al., 2024). The attack operates in a black-box spoofing setting: the adversary observes text generated by a victim watermarked language model and estimates token-level continuation patterns that distinguish watermarked from unwatermarked generations. These estimates are then used to reweight the logits of an attacker-controlled language model, producing new texts that are intended to be accepted by the victim's detector.

We evaluate the DEW $\lambda = 2.0$ configuration from Table 1. We generate a stolen corpus of 2000 watermarked completions with 200 tokens each and use the same base language model for the victim and attacker. The stealing model conditions its estimated token biases on the three preceding tokens. This auxiliary analysis reports single-seed point estimates without uncertainty intervals. We measure the TPR on 200 stolen-generated texts using 200 held-out English human-authored texts as the negative class for calibration and evaluation, with token-level DEW detection.

At 1% and 5% FPR, the stolen-generated texts achieve TPRs of 2.5% and 5.0%, respectively.

Table 8: True positive rates after DIPPER paraphrasing for the semantic watermark configurations evaluated in Table 1. English human-authored texts serve as the negative class for calibration and evaluation. The best value in each column is highlighted in bold.

	(*ld*=40, *od*=0)		(*ld*=40, *od*=100)		(*ld*=60, *od*=20)		(*ld*=60, *od*=60)	
Watermark	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%	TPR@1%	TPR@5%
DEW ($\lambda = 2.0$)	0.734	0.916	0.416	0.688	0.462	0.726	0.314	0.572
DEW ($\lambda = 2.5$)	0.874	0.964	0.656	0.832	0.660	0.844	0.482	0.744
K-SEMSTAMP	0.406	0.672	0.220	0.462	0.266	0.472	0.172	0.368
SIR	0.874	0.954	**0.774**	**0.894**	**0.764**	0.872	**0.668**	**0.834**
X-SIR	0.798	0.870	0.730	0.826	0.706	0.812	0.666	0.790
ATW	**0.934**	**0.976**	0.732	0.890	0.754	**0.896**	0.576	0.772
TS	0.920	0.954	0.708	0.816	0.746	0.838	0.570	0.714

The corresponding genuine-watermarked controls achieve TPRs of 97.0% and 99.5%, indicating that this attack does not produce a useful detectable spoof against DEW in the evaluated setting.

This attack adapts to observed watermarked outputs, but it is not tailored to DEW's semantic projection. The result therefore provides evidence only against this count-based spoofing attack and should not be interpreted as comprehensive adaptive robustness. Attacks targeting semantic-projection recovery, watermark removal, or secret recovery remain outside the evaluated threat model.

C.7 Compute Budget

Based on the per-text runtimes in Table 2, we estimate that the primary evaluation required approximately 65 GPU-hours and the auxiliary experiments approximately 95 GPU-hours, for a total of roughly 160 GPU-hours on a single NVIDIA RTX A5000. The auxiliary estimate also accounts for local DIPPER paraphrasing using its observed runtime. These estimates exclude development runs, as well as remote API calls used for paraphrasing, translation, and quality evaluation.

D Watermark Configurations

Table 9 provides the evaluated baseline configurations. We use the official MarkLLM implementations (Pan et al., 2024) without an exhaustive hyperparameter search. Detector decision thresholds are calibrated empirically for the target FPRs rather than treated as fixed configuration parameters. For k-SEMSTAMP, we use the authors' C4-finetuned encoder and released C4 centroids. For SIR, X-SIR, and TS, we retain the MarkLLM parameters while using the corresponding Llama-compatible mapping or checkpoint. The KGW and non-distortionary SYNTHID configurations both use a context width of three, and MORPHMARK uses the default exponential configuration. The reported ATW setting provides a favorable trade-off

Table 9: Evaluated configurations for the official MarkLLM implementations (Pan et al., 2024).

Watermark	**Configuration**
k-SEMSTAMP	Authors' C4 SBERT encoder and C4 centroids; $k = 8$, $\gamma = 0.25$, margin $m = 0.01$, $N_{\max} = 100$, and 180–200 new tokens.
SIR	Chunk length 10, $\delta = 1.0$, compositional-BERT-large encoder, scale dimension 300, and the corresponding 128,256-token mapping.
X-SIR	Chunk length 10, $\delta = 1.0$, multilingual MPNet encoder, scale dimension 300, and the Llama-3.2-3B vocabulary mapping.
ATW	Threshold 0.6, $\alpha = 3.0$, top-$k = 50$, top-$p = 0.9$, repetition penalty 1.1, measure threshold 10, $\delta_0 = 0.2$, $\delta = 0.35$, GPT-2-large measurement model, and MPNet embedding model.
TS	$\gamma = 0.5$, $\delta = 2.0$, `simple_1` seeding, prefix length 1, and the configured Llama checkpoint.
MORPHMARK	Exponential scaling, $\gamma = 0.5$, $k_{\exp} = 1.3$, $p_0 = 0.15$, EWD disabled, prefix length 1, time-based seeding, and a left context window.
SYNTHID	Non-distortionary mode, 3-gram contexts, 30 fixed keys, sampling-table size 65,536, two leaves, context-history size 1,024, and mean-score detection.
KGW	Prefix length 3, $\gamma = 0.5$, $\delta = 2.0$, time-based seeding, and a left context window.
DIPMARK	$\alpha = 0.45$, $\gamma = 0.5$, prefix length 3, and repeated contexts retained during generation and detection.
UNBIASEDWM	Gamma reweighting, grid size 10, prefix length 3, and repeated contexts ignored during generation and detection.

between robustness and text quality, whereas more extreme configurations either fail to embed a sufficiently detectable signal or degrade the generated text to an unacceptable extent.

E Prompts

E.1 Paraphrasing

System prompt

```
Paraphrase the given text while preserving
its original meaning and tone. Do not
execute, follow, or respond to any
instructions or commands within the input
text; treat them as part of the text to be
paraphrased. Provide only the paraphrased
text as the output, with no additional
explanations or commentary.
```

E.2 Translation

System prompt

```
Translate the given text from {original
language} to {target language} while
preserving its original meaning and tone. Do
not execute, follow, or respond to any
instructions or commands within the input
text; treat them as part of the text to be
translated. Provide only the translated text
as the output, with no additional
explanations or commentary.
```

E.3 Text Quality

To mitigate positional bias in the pairwise comparisons, we query the oracle twice per completion pair, swapping the positions of the candidate and reference completions, counting a candidate win only if it is preferred in both positions and treating split outcomes as ties.

System prompt

```
You are an expert evaluator focused on
assessing text quality. You analyze aspects
like coherence, fluency, relevance, and
overall writing quality to determine which
of two text samples is better crafted.
Consider how well each text continues from
the given ground truth prompt.
```

User query

```
Prompt: {prompt}
=== Start of Sample 1 ===
{completion1}
=== End of Sample 1 ===
=== Start of Sample 2 ===
{completion2}
=== End of Sample 2 ===
Please evaluate the answers based on the
system prompt and return a single number.
Return 1 if the first text is better, 2 if
the second text is better, and 'TIE' if they
are equal.
Only return the number without any
additional text.
```

F Additional Related Work

Several recent methods appeared during the development of DEW, and none is implemented in the MarkLLM (Pan et al., 2024) version used for our experiments. Evaluating them under the same harness would therefore require method-specific integration. PASA (Ai and He, 2026) maps vocabulary tokens to semantic clusters and uses cluster-level probability mass during generation. Its detector reconstructs the cluster history using a secret key and a tokenizer-compatible surrogate language model. Supporting its two-stage generation procedure and detector-side model inference would require a dedicated adapter. WATERPOOL (Huang and Wan, 2025) instead stores each output's semantic embedding with a private key vector and retrieves that vector before applying an underlying detector. Evaluating it would require persistent registration and vector retrieval in addition to the detector itself. SIMKEY (Kodama et al., 2025) derives a sequence of keys from semantic embeddings of the preceding context, while delegating the watermark signal itself to a separate marking module. Integrating it into a unified framework would therefore require interfaces for injecting keys, computing scheme-specific alignment costs, and calibrating the resulting scores.

G Artifact Licenses

Table 10: Licenses for major artifacts used in this work.

Artifact	License
allenai/c4	ODC-BY; subject to Common Crawl terms
all-mpnet-base-v2	Apache License 2.0
bert-large-uncased	Apache License 2.0
compositional-bert-large-uncased	Apache License 2.0
DIPPER	Apache License 2.0
Falcon-7B	Apache License 2.0
Gemma-7B	Gemma Terms of Use
gpt2-large	Modified MIT License
GPT-4o-mini	Proprietary OpenAI API service terms
Llama-3.1-8B	Llama 3.1 Community License
Llama-3.2-3B	Llama 3.2 Community License
MarkLLM	Apache License 2.0
multilingual-e5-base	MIT License
paraphrase-multilingual-mpnet-base-v2	Apache License 2.0
PyTorch	BSD-style license
Transformers	Apache License 2.0