SWAN: Semantic Watermarking with Abstract Meaning Representation

Ziping Ye[1*†] **Gourab Dey**[1*] **Christos Christodoulopoulos**[2‡]
Charith Peris[1] **Anil Ramakrishna**[3‡] **Weitong Ruan**[1]
Aram Galstyan[1,4] **Kai-Wei Chang**[1,5] **Rahul Gupta**[1] **Ninareh Mehrabi**[3‡]
[1]Amazon [2]Information Commissioner's Office [3]Meta [4]USC [5]UCLA

Abstract

We introduce SWAN (Semantic Watermarking with Abstract Meaning Representation)[1], a novel framework that embeds watermark signatures into the semantic structure of a sentence using Abstract Meaning Representation (AMR). In contrast to existing watermarking methods, which typically encode signatures by adjusting token selection preferences during text generation, SWAN embeds the signature directly in the sentence's semantic representation. As the signature is encoded at the semantic structure level, any paraphrase that preserves meaning automatically preserves the signature. SWAN is training-free: watermark injection is achieved by prompting an LLM to generate sentences guided by a selected AMR template while maintaining contextual coherence, and detection uses an off-the-shelf AMR parser followed by a simple one-proportion z-test. Empirical evaluation on the REALNEWS benchmark shows SWAN matches state-of-the-art detection performance on unaltered watermarked text, while significantly improving robustness against paraphrasing, increasing detection AUC by up to 13.9 percentage points compared to prior methods. These results demonstrate that SWAN's approach of anchoring watermarks in AMR semantic structures provides a simple, effective, and prompt-based method for robust text provenance verification under paraphrasing, opening new avenues for semantic-level watermarking research.

1 Introduction

Recent advances in large language models (LLMs) enable the generation of human-like text that can convincingly mimic genuine human writing, raising significant concerns about misinformation, impersonation, and unauthorized content reuse at scale (Xu et al., 2024; Williams et al., 2024). To address these risks, *text watermarking* has emerged as a crucial technique, embedding hidden but detectable signatures into generated content, thereby enabling reliable identification of AI-generated text without compromising readability.

Various approaches have been proposed to embed statistical or encrypted patterns into generated text by selecting tokens based on specific underlying rules. However, these token-level methods (Kirchenbauer et al., 2024a; Zhao et al., 2023; Kirchenbauer et al., 2024b; Kuditipudi et al., 2024) remain highly sensitive to paraphrasing and synonym substitutions, making the watermark easily undetectable after minor text modifications. To address these limitations, recent methods have shifted towards sentence-level watermarking, embedding signals within the semantic embedding space using techniques such as locality-sensitive hashing (Hou et al., 2023) or k-means clustering (Hou et al., 2024). While these sentence-level approaches improve robustness—meaning the watermark remains detectable even after paraphrasing—they can still fail if paraphrasing shifts the sentence embedding.

In this paper, we introduce **SWAN** (Semantic Watermarking with Abstract Meaning Representation), a novel watermarking approach that embeds watermark signatures directly into the semantic structure of generated sentences. Unlike previous token-level or embedding-based approaches, SWAN leverages Abstract Meaning Representation (AMR) (Banarescu et al., 2013; Regan et al., 2024), a graph-based formalism capturing core semantic relationships. By embedding watermark signals at the AMR graph level—for example, requiring a sentence to contain specific combinations of entities and relationships—SWAN ensures that any paraphrase preserving the underlying meaning will inherently retain the watermark. Anchoring the watermark to the semantic structure substantially enhances robustness to paraphrasing attacks, ad-

[*] Equal contribution.
[†] ✉ Corresponding author: zipingye@amazon.com
[‡] Work done while at Amazon.
[1]Code available at https://github.com/amazon-science/SWAN

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 36304–36315
July 2-7, 2026

dressing a key limitation of prior methods while maintaining high detectability and text quality.

We validate the effectiveness and robustness of SWAN through extensive empirical evaluation on the REALNEWS dataset (Raffel et al., 2023), a standard benchmark for sentence-level watermarking. Our experiments benchmark SWAN against state-of-the-art token-level and sentence-level watermarking baselines, assessing detectability under various paraphrase attacks. Results demonstrate that SWAN achieves detection accuracy comparable to state-of-the-art sentence-level watermarking methods and outperforms token-level watermarking techniques on unaltered text. More importantly, SWAN significantly enhances robustness under paraphrasing, consistently outperforming existing sentence-level approaches when detecting paraphrased text. Although watermarking generally introduces a trade-off in text quality, SWAN maintains text coherence, fluency, and diversity comparable to existing methods, as evaluated through a thorough LLM-based quality assessment.

To the best of our knowledge, this work represents the first exploration of Abstract Meaning Representation in the text watermarking domain, opening a new technical pathway based on symbolic semantic structures.

2 Background

2.1 Abstract Meaning Representation (AMR)

Abstract Meaning Representation is a graph-based formalism that encodes the core semantic structure of a sentence by abstracting away surface-level syntactic and lexical variations (Banarescu et al., 2013; Regan et al., 2024). Nodes in the graph represent *concepts* (e.g., entities, events), while edges capture *semantic relations* (e.g., agent, patient, location).

For instance, the sentences below, despite their surface-level differences, share the same core semantic meaning and therefore map to an identical AMR graph (Banarescu et al., 2019):

- "The boy desires the girl to believe him."
- "The boy desires to be believed by the girl."
- "The boy has a desire to be believed by the girl."
- "The boy's desire is for the girl to believe him."

These sentences correspond to the following AMR graph representation:

```
(w / want-01
    :ARG0 (b / boy)
    :ARG1 (b2 / believe-01
                :ARG0 (g / girl)
                :ARG1 b))
```

Figure 1 provides a visual depiction of this AMR structure, clearly illustrating the semantic relationships captured by the graph representation.

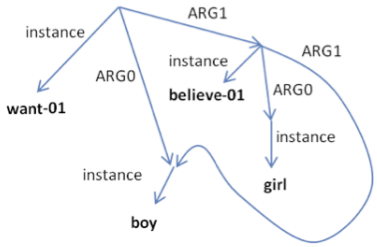

Figure 1: Visualization of an AMR graph capturing semantic equivalences across multiple paraphrases.[2]

By representing semantic structures explicitly, AMR provides an effective means for embedding robust watermark signals that remain intact even when sentences undergo significant lexical and syntactic modifications, as long as the core meaning remains unchanged.

2.2 Text Watermarking

Text watermarking refers to techniques that embed subtle but identifiable signatures into text to allow verification of its provenance or authorship. Depending on the granularity at which the signal is embedded, watermarking methods can be broadly classified into *token-level*, *sentence-level*, and *paragraph-level* approaches.

Token-Level Watermarking. Token-level schemes watermark text by subtly biasing the sampling distribution during decoding, so that the hidden statistical signature can later be detected (Kirchenbauer et al., 2024a,b; Kuditipudi et al., 2024; Zhao et al., 2023; Shi et al., 2023; Dathathri et al., 2024). A widely used design is the *green-list / red-list* framework of Kirchenbauer et al. (2024a), which partitions the vocabulary with a secret hash and forces generation to prefer the green tokens, enabling a simple z-score detector. Subsequent variants add provable robustness guarantees (Zhao et al., 2023), enforce distortion-free constraints (Kuditipudi et al., 2024), or optimize the logits with a learned key to balance invisibility and

[2]Figure adapted from https://github.com/amrisi/amr-guidelines/blob/master/graph.png.

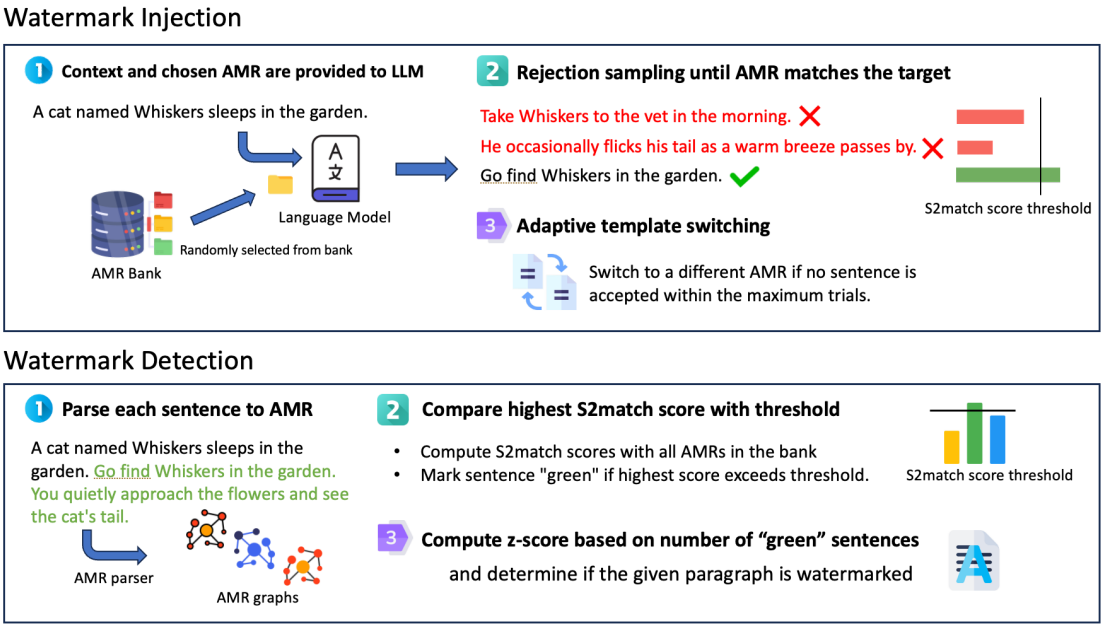

Figure 2: Overview of our proposed framework. In **watermark injection**, LLM repeatedly samples a sentence until its parsed AMR matches a secret template drawn from the bank; in **watermark detection**, we parse each sentence of a candidate paragraph, count AMR matches, and apply a z-test—because the watermark lives in the AMR graph, any paraphrase that preserves meaning leaves the signal intact.

detectability, as in SynthID-Text (Dathathri et al., 2024). Although straightforward, token-level methods remain fragile to paraphrasing and other surface edits, and they typically require direct access to model logits—an assumption that may not hold for proprietary LLMs.

Semantic and Sentence-Level Watermarking. To enhance robustness against paraphrasing, recent strategies encode signals into the semantic space rather than superficial token statistics. Hou et al. (2023) and Hou et al. (2024) partition the embedding space using locality-sensitive hashing (LSH) or k-means to selectively sample sentences from "watermarked" regions. Similarly, Liu et al. (2024) train a neural network to map context embeddings directly to watermark logits, while Ren et al. (2024) discretize the continuous embedding space to robustly seed vocabulary partitioning. Although these embedding-based methods better resist small lexical changes, paraphrasing that moves the sentence embedding out of the designated region still degrades detectability.

Paragraph-Level and Post-hoc Watermarking. Another line of research explores embedding watermark signals at the paragraph level or injecting them post-generation in a black-box manner. PostMark (Chang et al., 2024) exemplifies this by first encoding the semantic content of an entire paragraph, then selecting specific "watermark words" from a secret embedding table that an instruction-following LLM weaves into the existing text. *Detection* re-computes the expected word list and flags a paragraph when enough of the words (or close semantic neighbors) appear. While practical and model-agnostic, these methods introduce explicit lexical edits, making them potentially detectable or removable by sophisticated adversaries.

Our work diverges from these prior paradigms by embedding explicit watermark signatures directly into the semantic structure of sentences using Abstract Meaning Representation (AMR) graphs. This novel semantic anchoring approach provides robust detection under paraphrasing, addressing key limitations of existing token-level and embedding-based watermarking methods.

3 Overview of the Framework

SWAN embeds a *sentence-level* watermark by guiding each generated sentence to match a template AMR randomly drawn from a secret bank. During generation, sentences are produced conditioned on the accumulated context while adhering to the selected AMR template's semantic structure. Detection parses candidate text back to AMRs, counts template matches, and applies a

one-proportion z-test over the paragraph. Figure 2 summarizes the three components outlined below.

3.1 AMR Bank Creation

We begin by constructing a *bank* $\mathcal{B}$—a private collection of template AMR graphs that serves as the secret key for both watermark injection and detection. SWAN draws one template from this bank for each sentence to be generated.

Raw corpus. We start from MASSIVE-AMR (Regan et al., 2024)[3], which provides ∼84 K AMR graphs for 1,685 information-seeking utterances.

Template AMR. We further abstract raw AMRs into *template AMRs* by replacing nouns (e.g., specific named entities) with generic placeholders. This additional level of abstraction removes extraneous lexical details while retaining the underlying semantic structure. For example, a raw AMR node labeled "`Alice`" might become a placeholder "`NE`," ensuring that the resulting template AMR focuses on conceptual and relational information rather than domain-specific references. Using these template AMRs confers two main benefits: **(i) Generality:** A small number of template AMRs can map to a wide variety of sentence instantiations, since names or entities can be filled in later with context-appropriate text. **(ii) Robustness:** By stripping away specific entity mentions, we decrease the chance that minor lexical edits (e.g., changing "Alice" to "Mary") will disrupt the watermark's core semantics.

Selecting AMR Patterns. To populate the template bank, we traverse the full MASSIVE-AMR corpus and retain only those graphs that (a) occur within a specified frequency range, defined by a minimum frequency of three and a maximum frequency of twenty, effectively discarding both one-off idiosyncrasies and highly repetitive boilerplate, and (b) contain at least three concept nodes to guarantee non-trivial semantic content.

3.2 Watermark Injection

Sentence-Level Guidance. For each sentence to be generated, we draw a template AMR g from the bank and prompt the LLM with two inputs: (1) the accumulated context from previously generated sentences for discourse coherence, and (2) the template g with instructions to generate text adhering to its semantic structure. The accumulated context carries forward the intent from the original user prompt or task specification that initiated the generation. The prompt guides the model to instantiate abstract AMR concepts (e.g., replacing "NE" placeholders with appropriate named entities) while maintaining contextual flow and alignment with the user's original intent (full prompt in Appendix A.1). The candidate sentence is then parsed back into an AMR $\hat{g}$.

S2match-guided rejection. For every candidate sentence generated, we parse its AMR $\hat{g}$ and compute the S2MATCH similarity to the target template g—a lightweight metric that averages node- and edge-level F_1 scores and ranges from 0 (no overlap) to 1 (perfect match) (Opitz et al., 2020). The sentence is accepted only if $\text{S2match}(\hat{g}, g) \geq \theta_{\text{accept}}$; otherwise we discard it and resample.

Adaptive target switching. Because generation is conditioned on running discourse, certain templates may be semantically infeasible in a given context or for a given prompt. Rather than retrying an incompatible template indefinitely, we impose a small cap on resampling attempts; if the threshold is reached, we simply draw a new template at random from the bank and resume decoding.

3.3 Watermark Detection

Given a piece of text of unknown provenance, we determine whether it was produced by SWAN in three steps:

1. **AMR Parsing:** Convert each sentence into an AMR graph using an off-the-shelf AMR parser.

2. **Pattern Matching:** For the parsed graph $\hat{g}$ we compute $\max_{g \in \mathcal{B}} \text{S2match}(\hat{g}, g)$, the best similarity to any template in the bank $\mathcal{B}$. The sentence is labelled *watermarked* if this score exceeds a fixed threshold θ_{detect}.

3. **Paragraph test (one-proportion z-test).** If k of the n sentences in a paragraph are flagged, we compute:

$$z = \frac{k - \lambda n}{\sqrt{n\lambda(1-\lambda)}},$$

where λ is the expected hit-rate under the null hypothesis:

[3] https://github.com/amazon-science/MASSIVE-AMR

Algorithm 1: Watermark Injection

Input: AMR bank $\mathcal{B}$, initial sentence s_0, number of sentences N, max templates T, max attempts per template M, threshold θ_{accept}
outputSentences $\leftarrow$ []
context $\leftarrow s_0$
for $i = 1$ **to** N **do**
  acceptedSentence $\leftarrow$ **False**
  chosenText $\leftarrow$ None
  for $t = 1$ **to** T **do**
    templateAMR $\leftarrow$ RandomSample($\mathcal{B}$)
    for $m = 1$ **to** M **do**
      generatedText $\leftarrow$ LLM_Generate(*templateAMR*, *context*)
      parsedAMR $\leftarrow$ ParseToAMR(*generatedText*)
      score $\leftarrow$ S2match(*parsedAMR*, *templateAMR*)
      if *score* $\geq \theta_{\text{accept}}$ **then**
        chosenText $\leftarrow$ *generatedText*
        acceptedSentence $\leftarrow$ **True**
        break
      end if
    end for
    if *acceptedSentence* **then**
      break
    end if
  end for
  if not *acceptedSentence* **then**
    chosenText $\leftarrow$ *generatedText* {Fallback to the last attempt}
  end if
  outputSentences.append(*chosenText*)
  context $\leftarrow$ *context* + " " + *chosenText* {Update context with the accepted sentence}
end for
return *outputSentences*

Algorithm 2: Paragraph-Level Watermark Detection

Input: AMR bank $\mathcal{B}$, detection threshold θ_{detect}, similarity function S2match$(\cdot,\cdot)$, ZscoreTest$(\cdot)$, paragraph p
greenCount $\leftarrow 0$
totalSentences $\leftarrow 0$
for each sentence s in paragraph p **do**
  parsedAMR $\leftarrow$ ParseToAMR(s)
  bestScore $\leftarrow -\infty$
  for each *templateAMR* in $\mathcal{B}$ **do**
    score $\leftarrow$ S2match(*parsedAMR*, *templateAMR*)
    if *score* $>$ *bestScore* **then**
      bestScore $\leftarrow$ *score*
    end if
  end for
  if *bestScore* $\geq \theta_{\text{detect}}$ **then**
    greenCount $\leftarrow$ *greenCount* $+ 1$
  end if
  totalSentences $\leftarrow$ *totalSentences* $+ 1$
end for
greenFrac $\leftarrow \frac{\textit{greenCount}}{\textit{totalSentences}}$
decision $\leftarrow$ ZscoreTest(*greenFrac*)
if *decision* $=$ **True then**
  return "Likely watermarked"
else
  return "Not watermarked"
end if

H_0 :
The text is not generated (or written) knowing the secret AMR bank.

Because paraphrasing typically preserves semantic structure, the AMRs remain consistent even when surface forms differ, ensuring that our watermark remains detectable unless meaning has been fundamentally altered.

4 Experiments

4.1 Experimental Setup

Dataset. Following the SemStamp baseline, we evaluate on the REALNEWS subset of the C4 corpus (Raffel et al., 2023). REALNEWS consists of professionally written news articles and has become the de-facto test bed for sentence-level watermarking because its formal style reduces noise from ill-formed user content. We take the first 250 sentences of the subset as our evaluation set.

AMR-bank size. A "bank" is the private set of template AMRs from which SWAN draws one template per sentence. Its size $|\mathcal{B}|$ therefore controls the search space for both injection and detection. All results reported in Sec. 4 use a **50-template bank** ($|\mathcal{B}|$=50); Sec. 4.3 shows that performance is stable for larger banks up to 800 templates.

Models. We use the following models for each component in SWAN:[4]

- **Watermark Generation.** All watermark injection is performed with `DeepSeek-R1-Distill-Qwen-14B` (temperature: 0.6, top_p: 0.9).[5] SWAN employs rejection sampling capped at 50 trials per sentence (5 attempts per AMR template across up to 10 templates), compared to SemStamp's 100 trials per sentence.
- **Watermark Detection.** SWAN parses sentences with the `parse_xfm_bart_large` pipeline from AMRLIB (Jascob, 2020), a BART-large encoder–decoder trained on AMR-3 (LDC2020T02).
- **Paraphrase Generation.** We create attacks with three paraphrasers—Pegasus, Parrot, and Claude 3.7 Sonnet.

[4] For open-source models, we used NVIDIA H100 and A100 GPUs based on availability, and for closed-source models, we used Amazon Bedrock.

[5] https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-14B

- **Text-Quality Evaluation.** Claude 3.7 Sonnet is used, in a zero-shot setting, to rate coherence, fluency, and diversity of the generated text.

Baseline methods. We benchmark SWAN against the strongest publicly–available watermarking approaches: **(i) SynthID-Text.** SynthID-Text (Dathathri et al., 2024) is a production-ready *token-level* watermark that perturbs the next-token distribution during decoding. **(ii) SemStamp.** *SemStamp* partitions the sentence-embedding space with the random-hyperplane through locality-sensitive hashing (LSH) (Indyk and Motwani, 1998) and keeps only sentences whose LSH code falls in a secret "green" bucket (Hou et al., 2023). **(iii) k-SemStamp.** k-SemStamp (Hou et al., 2024) keeps the same accept/reject scheme but swaps the random-hyperplane LSH for k-means centroids learned from in-domain text, so regions follow real semantic structure which boosts paraphrase robustness.

Paraphrase attacks. Watermark robustness is evaluated under three paraphrasers: **(i) Pegasus.** Pegasus (Zhang et al., 2020) is a model fine-tuned for paraphrasing.[6] **(ii) Parrot.** Parrot (Damodaran, 2021) is a T5-base paraphraser that seeks high lexical diversity while preserving semantics. **(iii) Off-the-shelf LLM.** We use Claude 3.7 in a zero-shot setting instructed to rewrite the given sentence while preserving semantic meaning.

Evaluation metrics. We measure two orthogonal aspects: *(i) watermark quality*—how well the detector distinguishes watermarked text from non-watermarked text—and *(ii) text quality* of the watermarked output itself.

- **Watermark detectability.** We report three complementary metrics at the paragraph level (i.e., generate 5 new sentences given one sentence as initial context): (i) **AUC**, the area under the ROC curve, capturing overall separability between watermarked and non-watermarked text; (ii) **TPR@1%** and (iii) **TPR@5%**, the true positive rates at fixed false positive rates of 1% and 5%, which reflect detector performance in the low-FPR regime most relevant to deployment.

[6] https://huggingface.co/tuner007/pegasus_paraphrase

Method	AUC	TPR@1%	TPR@5%
SynthID	97.0	64.8	84.8
SemStamp	**99.4**	**96.8**	**100**
k-SemStamp	99.1	**96.8**	96.4
SWAN	99.1	91.6	97.6

Table 1: Detection accuracy (%) on REALNEWS sentences without paraphrasing. AUC is the area under the ROC curve; TPR@1% and TPR@5% are the true positive rates at 1% and 5% false positive rates, respectively. SWAN performs comparably to the sentence-level baseline and outperforms the token-level SynthID, showing that its semantic anchoring retains full detectability in the easiest setting.

- **Text quality.** We use Claude 3.7 as the judge and record scores on three axes: *Coherence*, *Fluency*, and *Diversity*. Detailed definitions of these quality dimensions and the complete evaluation prompt are provided in Appendix A.3.

4.2 Detection Results

Baseline performance (no paraphrase). Table 1 reports paragraph-level AUC on the REALNEWS set without any rewriting. SWAN matches the sentence-level competitor and outperforms the token-level SynthID watermark, demonstrating that our semantic approach entails no loss of raw detectability.

Robustness to paraphrasing. Table 2 shows the same evaluation after applying three paraphrasers. SynthID is omitted as SWAN outperforms it even in the non-paraphrasing scenario and its token-level cues are wiped out by even mild rewriting. Across all attacks, SWAN attains the highest AUC, confirming that anchoring the watermark in AMR graphs yields markedly stronger resilience than embedding-partition methods.

4.3 Effect of AMR-Bank Size

We further ablate the *bank size*, i.e., the number of AMRs in $\mathcal{B}$. Intuitively, a larger bank provides greater semantic coverage, so the watermark can be embedded across more diverse AMR patterns. However, this also raises the possibility of false positives, since any human-written sentence that incidentally aligns with one of many templates may be flagged. Table 3 shows that while a bank of 800 AMRs yields the highest AUC (99.3%), even a much smaller bank (50 AMRs) maintains strong

Method	Pegasus AUC / TPR@1% / TPR@5%	Parrot AUC / TPR@1% / TPR@5%	Claude AUC / TPR@1% / TPR@5%
SemStamp	97.6 / 87.2 / **97.6**	94.8 / 69.2 / 97.6	84.4 / 36.8 / 84.8
k-SemStamp	97.3 / **88.8** / 88.4	92.8 / 68.0 / 66.8	87.6 / 53.6 / 53.2
SWAN	**98.1** / 81.2 / 92.8	**97.5 / 82.0 / 92.4**	**98.3 / 86.0 / 95.2**

Table 2: Detection performance (%) under paraphrase attacks, reported as AUC / TPR@1% / TPR@5%. Across all three paraphrasers, SWAN yields the strongest results, with the largest margins under the zero-shot LLM paraphraser (Claude).

Bank Size ($\lvert\mathcal{B}\rvert$)	AUC
50	99.1
100	98.7
500	98.4
800	**99.3**

Table 3: Ablation on the AMR-bank size used during detection, reported as AUC (%) on unaltered (non-paraphrased) watermarked text. While performance remains high throughout, a bank of 800 AMRs yields the strongest AUC.

performance (99.1%). This indicates that SWAN is robust to different bank sizes, allowing practitioners to balance coverage and efficiency when deploying the watermark.

4.4 Text Quality Evaluation

To assess whether embedding a watermark affects generated text quality, we used a large language model (LLM) as an automated "reference-free" judge (Chen et al., 2023), rating paragraphs on a $[0, 5]$ scale along three dimensions: **(i) Coherence** (logical organization and clarity), **(ii) Fluency** (grammatical correctness and readability), and **(iii) Diversity** (variety of vocabulary and sentence structures).

Table 4 compares text generated without watermarking ("No Watermark") against three watermarking methods: **SWAN**, **SemStamp**, and **k-SemStamp**. Results indicate that while watermarking introduces a slight decrease in quality scores across all dimensions compared to the unwatermarked baseline, SWAN performs comparably to existing sentence-level watermarking methods. Thus, SWAN achieves strong paraphrase robustness (as demonstrated in §4.2) without introducing additional penalties in text coherence, fluency, or diversity relative to current state-of-the-art approaches.

4.5 Qualitative Examples

Table 5 provides illustrative examples of paragraphs generated by each watermarking method (SWAN, SemStamp, and k-SemStamp). For each method, we present one *High-quality* example and one *Low-quality* example based on the LLM-judged coherence, fluency, and diversity scores (§4.4), highlighting the typical range of output quality observed in practice.

4.6 Sampling Efficiency

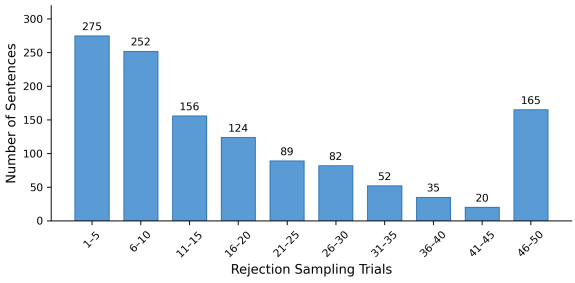

Figure 3: Distribution of rejection sampling trials per sentence across 1,250 generated sentences (250 samples $\times$ 5 sentences).

Like other sentence-level algorithms, SWAN relies on rejection sampling. Across 1,250 generated sentences (250×5) we measured a mean of 17.7 trials before acceptance, versus 13.8 trials for SemStamp. Figure 3 shows the full trial distribution: 42% of sentences are accepted within 10 trials and 54% within 15, confirming that the majority of generations converge quickly. The spike at 46–50 trials corresponds to sentences that exhausted the budget, suggesting that certain AMR templates are harder to satisfy in context; future work on context-aware template selection could reduce these cases and further improve efficiency. While the overall overhead is modest, it is offset by SWAN's significantly enhanced robustness to paraphrasing—our primary contribution—achieving up to 13.9 percentage points higher detection AUC (Table 2). SWAN remains *training-free* and works

Metric	No Watermark	SWAN	SemStamp	k-SemStamp
Coherence	4.19 ± 1.19 (0–5)	3.72 ± 1.17 (1–5)	3.69 ± 1.17 (0–5)	3.70 ± 1.16 (1–5)
Fluency	4.14 ± 1.21 (0–5)	3.87 ± 1.19 (1–5)	3.85 ± 1.20 (0–5)	3.85 ± 1.20 (1–5)
Diversity	3.99 ± 0.97 (0–4.8)	3.01 ± 0.96 (1–5)	2.99 ± 0.96 (0–4.7)	3.00 ± 0.95 (1–4.8)

Table 4: LLM-based text-quality scores (0–5) for un-watermarked text and three watermarking schemes, reported as Mean $\pm$ Std (Min–Max). Relative to the "No Watermark" baseline, all watermarking incurs a modest quality drop, but SWAN matches the sentence-level baselines (SemStamp and k-SemStamp) while offering stronger paraphrase robustness.

SWAN	
High-quality	*Whitehaven Coal is expected to consider a proposal. Whitehaven Coal's proposal is expected to be voted on by activist shareholders in record numbers. Vote for Whitehaven Coal. Vote for the proposal to demand that Whitehaven Coal align its strategy with the Paris climate agreement.*
Low-quality	*How does John feel about it? You feel happy. John feels it. John feels happiness about it. John feels happiness.*
SemStamp	
High-quality	*California is lifting its drought emergency for most of the state after a winter of record rain and snowfall that followed a five-year dry spell. Gov. Jerry Brown's office announced Friday that his state is now in a drought disaster, with nearly half the state under water restrictions.*
Low-quality	*WASHINGTON (Reuters) - U.S. Treasury Secretary Jack Lew on Sunday warned Congress against manufacturing a crisis over federal spending in the months ahead, as looming deadlines set the stage for a repeat. of the debt ceiling fights that roiled the country last year. The devil is in the details. Lew told CNN's State of the Union.*
k-SemStamp	
High-quality	*A Charlotte cancer patient and his Oncologist just finished one of the most iconic marathons in the country. Wayne Moorehead continues to inspire as he learns to walk without help at TIRR Memorial?, in his hometown of Charlotte, North Carolina. Watch a special interview with him below. This video was originally published on October 20, 2013.*
Low-quality	*Sports Academy has found "The Mamba Mentality." The 96,000-square-foot training facility in Thousand Oaks announced a partnership with Kobe Bryant. The 96,000-square-foot holistic athletic training facility in Thousand Oaks announced a partnership with?Kobe!Bryant!That is a first for the facility?s history.*

Table 5: Illustrative paragraphs for each watermarking method. Using the LLM-judged coherence, fluency, and diversity metrics (§4.4), we select one example that scores toward the higher end (*High-quality*) and another toward the lower end (*Low-quality*) to visualise the range of output quality.

via pure prompting and AMR parsing.

5 Related Work

Other semantic signatures. Early linguistic steganography embeds bits by *rewiring surface structure*. Atallah et al. (2001) propose two algorithms that hide information in the *parse tree* of selected sentences, modifying syntactic branches and then regenerating fluent text with an NLG system. More recently, Yoo et al. (2023) advance to a *multi-bit* framework that encodes a larger payload in *invariant features*—lexical or syntactic patterns shown empirically to survive minor text corruption. Complementary to these text-side schemes, Gu et al. (2023) "plant" a watermark *inside* the model itself by backdooring PLM weights with rare-word or composite triggers; ownership is verified by querying the model rather than analyzing the generated text.

Authorship attribution. Stylometric methods seek to infer the writer's identity from persistent linguistic habits rather than from an injected signal. Classic surveys document feature-rich classifiers that use function-word counts, character n-grams, and syntactic cues (Stamatatos, 2009; Koppel et al., 2009). Neural models learn style embeddings end-to-end, from continuous character n-grams (Sari et al., 2017) to the contrastively-trained *PART* Transformer (Huertas-Tato et al., 2022). Because these approaches rely on labelled samples for every author and can be defeated by paraphrase or adversarial style obfuscation (Brennan et al., 2012), they do not provide a verifiable yes/no provenance test.

6 Conclusion

We introduced **SWAN**, a novel semantic watermarking framework that embeds robust, paraphrase-resistant signals into AMR representations. By guiding generated text to align with curated AMR templates, SWAN remains detectable under paraphrasing as long as core semantic relationships are preserved. Empirically, it outperforms token-level and embedding-based watermark baselines in paraphrase robustness, while maintaining strong text fluency and naturalness.

Future Directions. We believe this work opens new directions for semantic-level watermarking research. Improving sampling efficiency represents a key practical direction, potentially through heuristic template selection, AMR-aware language model fine-tuning, or learned template-context matching to reduce rejection sampling overhead. Beyond efficiency, promising research avenues include exploring domain-specific AMR adaptations, improving parser robustness, and strengthening detection against advanced adversarial rewrites. Additionally, investigating how watermarking methods in general—including our AMR-based approach—might impact the reasoning capabilities of LLMs across different task domains represents an important area for future exploration. While SWAN currently focuses on the sentence level, multi-sentence attacks—such as splitting or merging sentences to alter AMR structure—represent a natural next challenge that could be addressed by extending to paragraph-level or multi-sentence AMRs. Another intriguing direction lies in using *AMR subgraphs* as watermark signals, offering additional resilience when adversaries deliberately reorganize semantic structures.

Limitations

Our detection relies on the quality of the AMR parser, which can occasionally misparse text or introduce random variations. Such errors may reduce recall by pushing a genuine SWAN sentence below the similarity threshold, or increase false positives if they inflate similarity to a template.

We focused on English news text for evaluation, where well-trained AMR parsers and public corpora are available. In domains with highly specialized jargon or in languages with limited AMR resources, accuracy and detectability might degrade, necessitating more domain-specific or multilingual AMR tools.

In our framework, the AMR bank $\mathcal{B}$ functions as the private key; its secrecy is required to prevent adversary detection. Future work may explore enhancing this security by integrating cryptographic protocols to manage template selection, similar to secret-key hashing in token-level schemes.

Ethical Considerations

Watermarking systems, including ours, come with both benefits and risks. On the one hand, they can help combat large-scale misinformation by allowing credible attribution of AI-generated content. On the other hand, false positives could arise if a text happens to align closely with the AMR templates by chance or if the AMR parser errs, potentially mislabeling human-authored text as AI-generated. This risk is especially salient in low-resource languages or domains where AMR parsers are less accurate, raising concerns about fairness and potential bias. Furthermore, mandated watermarking in certain contexts could impede free expression or privacy if applied too broadly. Finally, while our method robustly identifies paraphrases, adversaries may develop sophisticated rewriting strategies to remove the watermark, highlighting the possibility of an ongoing arms race. We encourage responsible use, transparency about these limitations, and continued research to refine watermarking techniques and their governance.

Acknowledgments

We thank Dipika Khullar for her contributions to the text quality evaluation experiments. We also thank Richard Zemel for valuable discussions during the course of this work.

References

Mikhail J. Atallah, Victor Raskin, Michael Crogan, Christian Hempelmann, Florian Kerschbaum, Dina Mohamed, and Sanket Naik. 2001. Natural language watermarking: Design, analysis, and a proof-of-concept implementation. In *Proceedings of the 4th International Workshop on Information Hiding*, IHW '01, page 185–199, Berlin, Heidelberg. Springer-Verlag.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic*

Annotation Workshop and Interoperability with Discourse, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2019. Abstract meaning representation (AMR) 1.2.6 specification. https://github.com/amrisi/amr-guidelines/blob/master/amr.md. Accessed: 2025-05-12.

Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Trans. Inf. Syst. Secur.*, 15(3).

Yapei Chang, Kalpesh Krishna, Amir Houmansadr, John Wieting, and Mohit Iyyer. 2024. Postmark: A robust blackbox watermark for large language models. *Preprint*, arXiv:2406.14517.

Yi Chen, Rui Wang, Haiyun Jiang, Shuming Shi, and Ruifeng Xu. 2023. Exploring the use of large language models for reference-free text quality evaluation: An empirical study. *Preprint*, arXiv:2304.00723.

Prithiviraj Damodaran. 2021. Parrot paraphraser. https://github.com/PrithivirajDamodaran/Parrot_Paraphraser.

Sumanth Dathathri, Abigail See, Shubham Ghaisas, Pierre-Sacha Hürlimann, Jacob Walker, Brian Bartoldson, Rohan Mukherjee, Aditya Sen, Varun Bansal, Rohan Bhasin, Michael A. Munn, Alexey Korotkevich, Rishabh Singh, Thomas Mensink, James Hennessey, Nisanth Venkateswaran, Benjamin Bichsel, Thomas Cooijmans, Zoubin Ghahramani, and 6 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823.

Chenxi Gu, Chengsong Huang, Xiaoqing Zheng, Kai-Wei Chang, and Cho-Jui Hsieh. 2023. Watermarking pre-trained language models with backdooring. *Preprint*, arXiv:2210.07543.

Abe Bohan Hou, Jingyu Zhang, Tianxing He, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. 2023. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Abe Bohan Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. 2024. 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.

Javier Huertas-Tato, Alejandro Martin, and David Camacho. 2022. Part: Pre-trained authorship representation transformer. *Preprint*, arXiv:2209.15373.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA. Association for Computing Machinery.

Brad Jascob. 2020. amrlib: A python library that makes amr parsing, generation and visualization simple. Accessed: 2025-05-12.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2024a. A watermark for large language models. *Preprint*, arXiv:2301.10226.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. 2024b. On the reliability of watermarks for large language models. *Preprint*, arXiv:2306.04634.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2009. Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, 60(1):9–26.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2024. Robust distortion-free watermarks for language models. *Preprint*, arXiv:2307.15593.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. 2024. A semantic invariant robust watermark for large language models. *Preprint*, arXiv:2310.06356.

Juri Opitz, Letitia Parcalabescu, and Anette Frank. 2020. AMR similarity metrics from principles. *Transactions of the Association for Computational Linguistics*, 8:522–538.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Michael Regan, Shira Wein, George Baker, and Emilio Monti. 2024. Massive multilingual abstract meaning representation: A dataset and baselines for hallucination detection. *Preprint*, arXiv:2405.19285.

Jie Ren, Han Xu, Yiding Liu, Yingqian Cui, Shuaiqiang Wang, Dawei Yin, and Jiliang Tang. 2024. A robust semantics-based watermark for large language model against paraphrasing. *Preprint*, arXiv:2311.08721.

Yunita Sari, Andreas Vlachos, and Mark Stevenson. 2017. Continuous n-gram representations for authorship attribution. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 267–273, Valencia, Spain. Association for Computational Linguistics.

Zhouxing Shi, Yihan Wang, Fan Yin, Xiangning Chen, Kai-Wei Chang, and Cho-Jui Hsieh. 2023. Red teaming language model detectors with language models. *Preprint*, arXiv:2305.19713.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *J. Am. Soc. Inf. Sci. Technol.*, 60(3):538–556.

Angus R. Williams, Liam Burke-Moore, Ryan Sze-Yin Chan, Florence E. Enock, Federico Nanni, Tvesha Sippy, Yi-Ling Chung, Evelina Gabasova, Kobi Hackenburg, and Jonathan Bright. 2024. Large language models can consistently generate high-quality content for election disinformation operations. *Preprint*, arXiv:2408.06731.

Jialiang Xu, Shenglan Li, Zhaozhuo Xu, and Denghui Zhang. 2024. Do llms know to respect copyright notice? *Preprint*, arXiv:2411.01136.

KiYoon Yoo, Wonhyuk Ahn, Jiho Jang, and Nojun Kwak. 2023. Robust multi-bit natural language watermarking through invariant features. *Preprint*, arXiv:2305.01904.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *Preprint*, arXiv:1912.08777.

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

A Appendix

A.1 Prompt for text generation with watermark

```
AMR (Abstract Meaning
    Representation) is a graph
    -based representation of a
     sentence's meaning. Each
    node is a concept and
    edges represent semantic
    roles or relationships.
    Below are some examples of
     template AMRs and
    corresponding sentences:
{example_text}

In the provided AMR, there are
     placeholders:
- "NE" for named entities (e.g
    ., "Alice", "France", "
    Google").
- "N" for generic nouns (e.g.,
     "a device", "an object").
- "X" for unspecified concepts
     (e.g., "something", "an
    idea").

Instructions:
- Do not write "NE", "N", or "
    X" literally. Instead,
    replace them with
    appropriate English words
    to form a natural,
    meaningful sentence.
- Ensure the generated
    sentence aligns with both
    the AMR structure and the
    given context.
- Do not produce multiple
    sentences or lists.
- Produce exactly one coherent
     sentence.

AMR:
{chosen_template}
Context: {context}
Please output only that one
    sentence.
```

A.2 Prompt for zero-shot Claude paraphrase

```
Previous context: {' '.join(
    context)}
Current sentence to paraphrase
    : {sent}
Rewrite the sentence above
    while preserving its
    meaning.
Do not provide any explanation
     or extra commentary.
Return only the new sentence.
```

A.3 Prompt for LLM-as-a-Judge Text Quality Evaluation

```
You are an expert writing
    quality evaluator.

You will assess a GENERATED
    PARAGRAPH using the
    following criteria. For
    each, assign a score from
    1 to 5 (decimals allowed),
     using the descriptions
    below.

1. **Coherence**: Measures how
```

```
    logically and clearly the
    ideas are organized and
   connected.
   - 1: Incoherent; sentences
      are unrelated or
      confusing.
   - 2: Poor transitions or
      unclear relationships
      between ideas.
   - 3: Basic logical flow,
      but some awkward
      connections.
   - 4: Mostly logical and
      clear, with minor
      lapses.
   - 5: Highly logical and
      seamless flow of ideas.

2. **Fluency**: Assesses the
   grammatical correctness
   and naturalness of the
   language.
   - 1: Grammatically broken
      or unreadable.
   - 2: Understandable but
      awkward or error-prone.
   - 3: Generally readable,
      some minor grammatical
      errors or odd phrasing.
   - 4: Well-written with only
       occasional issues.
   - 5: Grammatically correct
      and naturally flowing
      throughout.

3. **Diversity**: Use of
   varied vocabulary and
   sentence structure,
   avoiding repetition.
   - 1: Extremely repetitive
      or formulaic.
   - 2: Some repetition with
      occasional variation.
   - 3: Moderate variety; not
      monotonous.
   - 4: Good diversity in
      language and structure.
   - 5: Highly expressive and
      varied without
      redundancy.

**Scoring Instructions**:
- Return a score for each of
   the three dimensions above
   .
- You may use decimal values (
   e.g., 2.5, 4.7).

**Output Format**:
Respond with a **valid JSON
   object only** in this
   exact format:

{{
  "coherence_score":  float,
  "fluency_score":    float,
  "diversity_score":  float
}}
```