

# Semantic Differentiation for Tackling Challenges in Watermarking Low-Entropy Constrained Generation Outputs

Nghia T. Le, Alan Ritter, & Kartik Goyal

School of Interactive Computing

Georgia Institute of Technology

Atlanta, GA 30308, USA

{nle18, alan.ritter, kartikgo}@gatech.edu

## Abstract

We demonstrate that while the current approaches for language model watermarking are effective for open-ended generation, they are inadequate at watermarking LM outputs for constrained generation tasks with low-entropy output spaces. Therefore, we devise *SeqMark*, a sequence-level watermarking algorithm with semantic differentiation that balances output quality, watermark detectability, and imperceptibility. It improves on the shortcomings of token-level watermarking algorithms that cause under-utilization of the sequence-level entropy available for constrained generation tasks. Moreover, we identify and improve upon the problem of *region collapse*, a different failure mode associated with prior sequence-level watermarking algorithms. This occurs because the pseudorandom partitioning of semantic space for watermarking in these approaches causes all high-probability outputs to collapse into either invalid or valid regions, leading to a trade-off in output quality and watermarking effectiveness. Instead, *SeqMark* differentiates the high-probable output subspace and partitions it into valid and invalid regions, ensuring the even spread of high-quality outputs among all the regions. On various constrained generation tasks like machine translation, abstractive summarization, and code generation, *SeqMark* substantially improves watermark detection accuracy (up to 28% increase in  $F_1$ ) while maintaining high generation quality.<sup>1</sup>

## 1 Introduction

As progress in language modeling leads to increasingly human-like automatic text generation, demand for tracing the provenance and life-cycle of digital text on the internet has soared. Questions around plagiarism (Sullivan et al., 2023), copyright infringement (Zhong et al., 2023), veracity of text (Augenstein et al., 2024), misinformation (Gravel et al., 2023), multi-agentic communication (Motwani et al., 2024), and many other legal and contractual frameworks require the ability to attribute text to its source. Language model watermarking – embedding a traceable digital marker in the language model’s distribution over language – has emerged as a potential solution to address this need. The seminal work on watermarking Kirchenbauer et al. (2023) modifies the logits produced over the vocabulary at each generation step from a language model to prefer or disprefer certain tokens. Much of the follow-up work (Fernandez et al., 2023; Zhao et al., 2023; Takezawa et al., 2025) on this *token-level watermarking framework* has focused on improved robustness to edits, generation quality, detectability, and other desiderata for watermarking. While effective for open-ended long-form generation, LM watermarking remains a challenging problem in low-entropy setups such as factual question-answering and code generation. While some work (Lee et al., 2024; Lu et al., 2024) addresses this problem for code generation via selective token-entropy based watermarking, we posit that in general, all token-level watermarking schemes under-utilize sequence-level entropy for embedding the signal and hence struggle in constrained generation tasks

---

<sup>1</sup>Our code is available at <https://github.com/nle18/seqmark>.

like machine translation and abstractive summarization. Although these tasks have lower entropy than open-ended generation, they still admit multiple feasible responses with sufficient sequence-level entropy.

An appealing alternative paradigm is *sequence-level watermarking* (Hou et al., 2024a;b): instead of randomly partitioning the token space at each generation step, it focuses on partitioning the manifold of sequences via hyperplanes into red/green regions, and then performs rejection sampling to sample from the green regions. However, *we identify a different common issue of red/green region collapse* with existing approaches that prevent low-entropy watermarking. Specifically, the operationalization of these algorithms via locality-sensitive hashing or  $k$ -means clustering leads to the assignment of semantically similar sentences to the same red/green partition. Due to semantic diversity, this is not a problem for open-ended generation tasks, but for constrained generation tasks, most of the admissible outputs with high probability tend to be semantically close to one another and mostly collapse to a single region. This imposes a challenging tradeoff between output quality and watermark verifiability for watermarking in constrained generation tasks.

In this paper, we propose a novel sequence-level watermarking approach *SeqMark* that utilizes the limited sequence entropy effectively to watermark constrained generation tasks. This approach first identifies the semantic manifold of high-probable admissible outputs for the task at hand, and then semantically differentiates between the elements of this manifold to propose pseudorandom red/green partitions in a manner such that the potential high-quality responses are evenly spread across the partitions, enabling successful watermarking in low-entropy settings. We empirically support our claims and observations on watermarking previously underexplored constrained generation tasks like machine translation, summarization, and code generation. We demonstrate the issues with existing token-level and sequence-level watermarking algorithms and show that SeqMark is more effective in constrained generation settings, while maintaining high watermarking capabilities in open-ended generation settings.

## 2 Preliminaries

Watermarking is inherently a problem of incorporating a digital marker into the (ideally noise-tolerant) carrier signal. We are primarily interested in settings that incorporate a marker into the autoregressive language model’s distribution and thus perform watermarking (ideally imperceptible) during text generation.

**Token-level Logit Watermarking** Kirchenbauer et al. (2023) introduced a seminal token-level watermarking algorithm dubbed KGW that embedded the marker in the logits produced over vocabulary for each token during generation. Many following attempts (Liu et al., 2024; Fernandez et al., 2023) proposed variations to improve robustness and undetectability of the watermarking scheme. At an abstract level, all of these approaches modify the *token-level distribution* for watermarking during generation by pseudorandomly partition the vocabulary  $\mathcal{V}$  into a green list of size  $\gamma|\mathcal{V}|$  and red list size  $(1 - \gamma)|\mathcal{V}|$  for a hyperparameter  $\gamma \in (0, 1)$ , using a hash from the previous token  $w_{t-1}$  or LM state as random seed. The generation procedure is then modified to prefer the green-list partition over the red-list partition, thus embedding the watermark. Detection involves performing hypothesis testing to decide whether the frequency of the green-list tokens in the query text occurs by chance.

**Sequence-level Watermarking** To improve the robustness of the watermarking method to paraphrase attacks, a different line of approach dubbed as semantic watermarking (Hou et al., 2024a;b) focuses on using sequence-level distribution induced by the language model instead of token-level distributions to embed the watermark. In general, the *sentence-embedding space* is pseudorandomly partitioned into green and red regions. Then a sentence is generated from the language modeling via rejection sampling until the resulting sentence embedding falls in the green region or the budget for resampling is exhausted. For generating the next sentence, another round of rejection sampling is performed following pseudorandom partitioning of the embeddings space using the hash based on the previously generated sentence. Prior work SemStamp (Hou et al., 2024a) uses locality-sensitive hashing

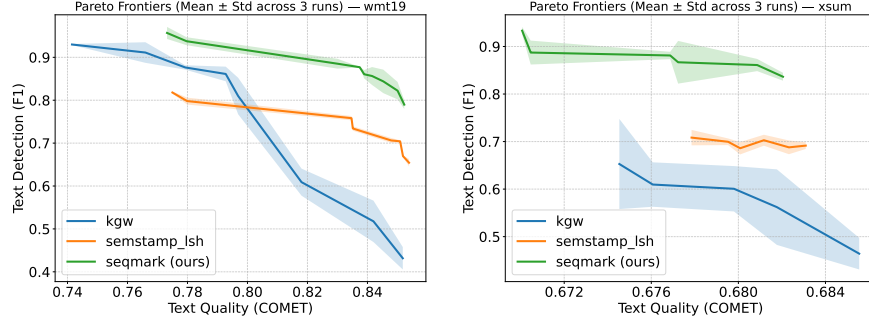


Figure 1: Pareto frontiers of token-level watermarking KGW (blue), and two sequence-level watermarking approaches SemStamp (orange) and SeqMark (ours, green) for machine translation (Left) and summarization (Right).

(LSH) to pseudorandomly partition the embedding space: LSH first samples  $n$  random vectors from a normal Gaussian distribution to specify  $n$  hyperplanes and thus  $2^n$  regions. Given a hyperparameter ratio  $\gamma$  and the hash of the previous sentence, the regions are partitioned into  $\gamma 2^n$  green regions and  $(1 - \gamma) 2^n$  red regions. A sentence with embedding  $\mathbf{v} \in \mathbb{R}^d$  would receive an  $n$ -bit binary LSH signature  $c$ , where each bit specifies the location of  $v$  with respect to each hyperplane – thus  $c$  identifies the region assignment for  $v$ . In contrast,  $k$ -SemStamp (Hou et al., 2024b) uses  $k$ -means clustering for partitioning instead of LSH: it first estimates  $K$  centroids for a general semantic manifold via pretraining, and then during the generation step it randomly assigns each of these centroids either a red/green tag. During sampling, each generated sentence is assigned red/green tag according to its closest centroid. Watermark detection hinges on the frequency of sentences in the green regions.

### 3 Entropy and Watermarking

Imperceptibility of the watermark is very important for textual applications in which the flow, tone, and naturalness of the text determine the experience of the reader. It requires redundancy in the signal, which for our purposes indicates the need for the LM distribution over sequences to have high entropy. Therefore, constrained generation are more difficult to watermark because the entropy over the desired text distribution is much lower than in the case of open-ended generation. Consider a task like machine translation: given an input, there is a small set of acceptable outputs that convey similar meaning but differ greatly in terms of style, syntax, coverage, etc. This set is much smaller than open-ended generation but still larger and more varied than some low-entropy tasks like factual question-answering or even code-generation. Therefore, we posit and empirically observe with our proposed approach (§ 7), that these seemingly low-entropy constrained generation tasks have enough sequence-level entropy to enable imperceptible watermarking. Assuming max-length of sequences (denoted by r.v.  $\mathbf{y}$ ) is  $T$ , we can compute this entropy over  $T$  random variables (tokens) under our model (parametrized by  $\theta$ ) via chain rule for entropy:  $H_\theta(\mathbf{y}) = \sum_{i=1}^T H_\theta(\mathbf{y}_i | \mathbf{y}_{<i})$  where crucially the conditional entropy involves summation over all the possible assignments of the prefix context  $\mathbf{y}_{<i}$  i.e.  $H(s | t) = -\sum_{s,t} p(s, t) \log p(s | t)$ . This entropy is typically intractable to compute and is estimated (Kuhn et al., 2023) via sampling. In a sharp contrast, token-level entropy that is often used to characterize uncertainty (Duan et al., 2023) of language models is tractable to compute because instead of summing over all the possible prefixes, it commits to a single sampled prefix and computes the entropy over the possible tokens at the next step:  $H_\theta(\mathbf{y}_t | \mathbf{y}_{<t}) = -\sum_{w \in \mathcal{Y}} p(w | \mathbf{y}_{<t}) \log p(w | \mathbf{y}_{<t})$ . The sequence entropy that is typically computed using token-level entropy computation for a sequence  $w = w_1, \dots, w_T$  given by  $H_\theta(w) = \sum_{i=1}^T H(w_i | w_{<i})$  is one-sequence approximation to the full intractable sequence-level entropy. Therefore, while tractable and attractive, common approaches of using token-level entropy to characterize sequence

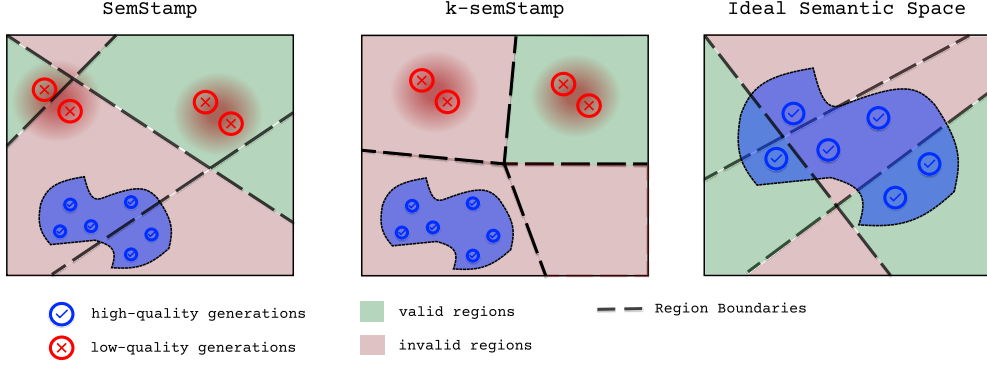


Figure 2: Semantic space illustration for different sequence watermarking algorithms. Both SemStamp (**Left**) and  $k$ -SemStamp (**Middle**) suffer from region collapse. Ideally, we would like to isolate the high-quality output manifold and partition it (**Right**).

| Approach      | Translation | Summary     | Open-ended  |
|---------------|-------------|-------------|-------------|
| SemStamp      | 0.19 / 0.97 | 0.39 / 0.92 | 0.71 / 0.59 |
| $k$ -SemStamp | 0.25 / 0.97 | 0.11 / 0.94 | 0.87 / 0.31 |
| SeqMark       | 0.81 / 0.01 | 0.92 / 0.03 | 0.96 / 0.01 |

Table 1: **Region entropy/average pairwise cosine similarity** across different tasks (columns) and watermarking approaches (rows). Higher region entropy means the points are more spread out.

entropy tend to underestimate the true entropy over the sequences under the language model.

This observation has crucial ramifications for watermarking algorithms in low-entropy settings. The prevalent entropy-sensitive watermarking algorithms in prior work (Lee et al., 2024; Lu et al., 2024) only consider token-level entropy while embedding watermark in tokens during generation. For example, during tokenwise generation, Lee et al. (2024) only chooses to mark the tokens that have high entropy under the language model. As we show in our experiments (§ 7), this approach does not perform well for constrained generation tasks. In Figure 1, we compare the token-level watermarking algorithm KGW (Kirchenbauer et al., 2023) against two sequence-level watermarking algorithms, SemStamp (Hou et al., 2024a) and our proposed approach SeqMark (§ 5). By sweeping over hyperparameters for these algorithms, we obtain a pareto curve for each approach. It is clear that the token-level watermarking approach is inferior to both sequence-level approaches – we see a severe drop-off in watermark detection accuracy as the output quality increases for both the tasks.

As a token-level approach commits to a sequence during left-to-right generation, it ignores all other prefixes and potential paths that could lead to feasible outputs which drastically limits the entropy it can exploit for watermarking. Sequence-level semantic watermarking on the other hand considers sequence-level distribution via rejection sampling and is more amenable to exploit the sequence entropy afforded by constrained generation tasks.

## 4 Region Collapse in Semantic Watermarking

We observe in Figure 1 that the sequence-level algorithm SemStamp behaves better than the token-level algorithm KGW, but still is inferior to our approach described later. We identify a crucial property of sequence-level watermarking algorithms in prior work that render them unsuitable for constrained generation watermarking. We call this property *region collapse* to refer to the effect that all viable outputs for a constrained generation task either randomly fall in the invalid red region or the valid green region. Consider the task of translation: If all the good translations of the prompt fall in the red region, then the

sequence-level watermarking algorithms will either generate a bad translation from the green region and hurt the translation quality, or exhaust its resampling budget and output a translation from the red region, eventually degrading the watermark effectiveness. We describe the manifestation of region collapse in prior work on semantic watermarking algorithms: SemStamp (Hou et al., 2024a) and  $k$ -SemStamp (Hou et al., 2024b).

A well-trained language model induces a high probability on the correct responses for a constrained generation prompt and samples them more frequently.<sup>2</sup> Also, because of semantic similarity of the high quality responses, the embeddings for these responses tend to have high similarity. An ideal sequence-level watermarking algorithm would produce red/green regions that separate these nearby responses to prevent region collapse as depicted in Figure 2. However, SemStamp’s partitioning behaves in a manner *opposite* to this desired behavior as it employs locality sensitive hashing (LSH) to determine the partitions.

Under LSH (Charikar, 2002; Indyk & Motwani, 1998), given two vectors  $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$  with angle  $\theta_{ij} \in [0, \pi]$ , the probability that they lie in the same region is  $(1 - \theta_{ij}/\pi)^d$ . LSH determines partitioning hyperplanes such that similar points share the same partition, thereby *accelerating* region collapse.  $k$ -SemStamp, while shown to be more robust than SemStamp, unfortunately *exacerbates* the region collapse issue for constrained generation. As described in §2, this method explicitly focuses on assigning the same centroid/color to the semantically similar points with low embeddings distances, thus exacerbating region collapse.

In Table 1, we empirically demonstrate the issue of region collapse on the tasks of translation and summarization. We sample  $N = 100$  high-quality (and high-probability) generations for each setting and report average pairwise cosine similarity and the estimated region entropy – a quantity that characterizes how evenly the high-quality points are spread across the partitions. The region entropy is computed as  $-\sum_{r \in \mathcal{R}} P(r) \log P(r)$ , where  $\mathcal{R}$  is the set of semantic regions. We estimate  $P(r)$  using Monte Carlo sampling:  $P(r) = 1/N \sum_{i=1}^N \mathbf{1}(\text{region}(x_i) = r)$ . While all three methods behave well for open-ended generation, for constrained generation tasks we observe low region entropy and high semantic similarity for SemStamp and  $k$ -SemStamp indicating region collapse, but high region entropy and low semantic similarity (explained below) for our SeqMark approach that explicitly ameliorates the issue of region collapse.

## 5 SeqMark: Fixing Region Collapse

As mentioned above, we propose SeqMark, a sequence-level watermarking approach that addresses region collapse. Similar to other semantic watermark approaches (Hou et al., 2024a;b), SeqMark pseudorandomly partitions the representation space into accept/reject regions and performs rejection sampling for watermarking. However, it differs significantly in how it operationalizes the partitioning procedure. As shown in Figure 3, we first focus on isolating the subspace manifold containing high quality responses for the constrained generation prompt. We estimate this manifold by sampling highly-likely generations using low-temperature sampling. Once isolated, we partition this manifold such that the high quality (highly probable) points are evenly distributed among the regions. To perform this partitioning, we still use LSH, but crucially we *transform the isolated manifold* such that its members (high-quality responses) are distant from one another, causing LSH to evenly spread them across the random partitions. Concretely, for an input prompt  $p$  (that includes  $t - 1$  previously generated sentences), we generate the response ( $t$ -th sentence  $s_t$ ) as follows: We first sample  $n$  high likelihood responses (with embeddings  $\mathbf{c}_i$ ) under low temperature to estimate high-quality subspace  $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n\}$ . Let  $f(\cdot; \mathcal{C})$  be the function

<sup>2</sup>We assume that highly probable generations from a well-trained LM are high-quality generations. While this might not always be the case, we believe this is a reasonable estimation for constrained generation tasks given the large( $\sim$ infinite) output sequence space, especially under well-trained language models. In addition, it is commonly observed that low-probability sequences tend to mostly be low-quality.

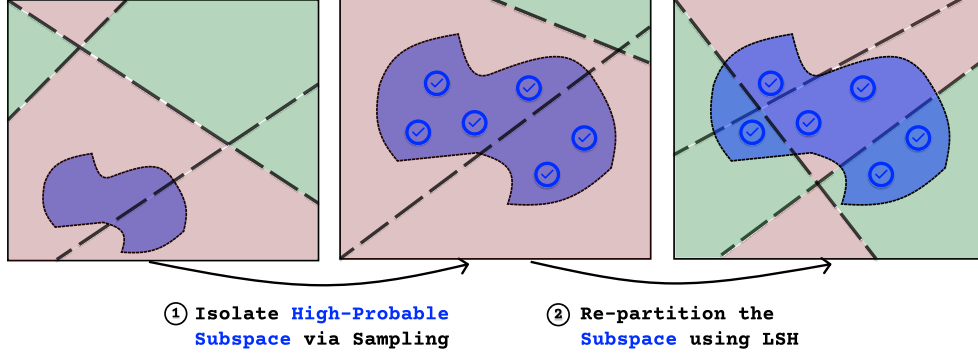


Figure 3: Depiction of SeqMark: Standard LSH in the first panel collapses the high-probable output subspace (blue) to a small number of invalid regions (red). We identify the high-probable subspace by sampling with low temperature; then we repartition by performing LSH on a *transformed* subspace designed to evenly spread it across multiple regions.

that transforms its members  $\mathbf{c}_i$  into a representation  $\mathbf{u}_i$ :  $\mathbf{u}_i = f(\mathbf{c}_i; \mathcal{C})$ , which would be used to assign an LSH partition to  $\mathbf{c}_i$ . To prevent region collapse, we aim to minimize the pairwise cosine similarity between the transformed members of  $\mathcal{C}$  while preserving their relative proximity. This is difficult to estimate in general and could be approximated by learning such a function via a neural network. We opt for a much simpler choice for  $f$ : subtracting the sample mean from each member of  $\mathcal{C}$ :  $f(\mathbf{c}_i; \mathcal{C}) = \mathbf{c}_i - \frac{1}{n} \sum_{j=1}^n \mathbf{c}_j$ .<sup>3</sup> We empirically observe that our mean-centering approach increases the distance between high-quality points in the desired manner (Table 1). Finally, Theorem 1 below states that watermark detection accuracy improves with this mean-centering transformation.

**Theorem 1** Given  $n$  vectors  $\mathbf{c}_1, \dots, \mathbf{c}_n$  such that  $\mathbf{c}_i = \mu + \epsilon_i$  and there exists  $\delta \in [0, 1]$  such that  $\|\epsilon_i\| \leq \delta \|\mu\|$ . Mean-centering transformation results in vectors  $\mathbf{u}_i = \mathbf{c}_i - \frac{1}{n} \sum_i \mathbf{c}_i = \epsilon_i$ . If the following condition holds for a fixed pair  $i, j$ :

$$\cos\theta_{ij}^{(\mathbf{u})} = \frac{\epsilon_i \cdot \epsilon_j}{\|\epsilon_i\| \|\epsilon_j\|} \leq \frac{1 - 2\delta - \delta^2}{1 + 2\delta + \delta^2}$$

then the watermark detection accuracy obtained using an LSH partition of  $\mathbf{u}_1, \dots, \mathbf{u}_n$  improves upon that obtained using an LSH partition of  $\mathbf{c}_1, \dots, \mathbf{c}_n$ .

**Remark** The constant  $\delta$  summarizes vector similarity. Intuitively, the mean term  $\mu$  of extremely similar vectors would dominate the residual term  $\epsilon$ , thus  $\delta \ll 1$  and  $\frac{1-2\delta-\delta^2}{1+2\delta+\delta^2}$  would be close to 1. This gives a rather relaxed bound for residuals above, and almost guarantee that the transformation would result in increasing detection accuracy. The full proof and discussion are presented in §A.7.

## 6 Experimental Setup

**Tasks and Datasets:** We evaluate our approach on several constrained text generation settings: sentence translation with WMT19 German-English dataset (Bojar et al., 2019), paragraph translation with WMT23 German-English (Kocmi et al., 2023), abstractive summarization with XSum (Narayan et al., 2018), and code generation with MBPP (Austin et al., 2021). For unconstrained open-ended generation, we follow previous work and complete sentences from C4 RealNews dataset (Raffel et al., 2020). Table 13 presents tasks and datasets details.

**Language Models and Embeddings:** For sentence translation, we use ALMA-7B (Xu et al., 2023) and LaBSE (Feng et al., 2022) as the primary language model and sentence encoder,

<sup>3</sup>The full generation and detection algorithms are given in Algorithm 1 and Algorithm 2, respectively.

|              | Translation (WMT19 De-En) |                                         |                                         | Summarization (XSum)              |                                        |                                         |
|--------------|---------------------------|-----------------------------------------|-----------------------------------------|-----------------------------------|----------------------------------------|-----------------------------------------|
|              | COMET $\uparrow$          | P / R / F <sub>1</sub> (h)              | P / R / F <sub>1</sub> (nw)             | R-L $\uparrow$ / COMET $\uparrow$ | P / R / F <sub>1</sub> (h)             | P / R / F <sub>1</sub> (nw)             |
| No Watermark | 87.4                      | -                                       | -                                       | 20.5 / 69.0                       | -                                      | -                                       |
| KGW          | <u>87.4</u>               | 57.2 / 36.6 / 45.7                      | 58.8 / 40.0 / 47.6                      | 20.1 / <u>68.8</u>                | <u>85.2</u> / 30.1 / 44.9              | 79.8 / 30.4 / 44.0                      |
| SWEET        | 87.2                      | 57.5 / 30.0 / 39.5                      | 58.1 / 30.1 / 39.6                      | 18.5 / 68.7                       | 52.4 / 49.9 / 51.1                     | 53.0 / 40.0 / 45.6                      |
| SemStamp     | 87.4                      | 59.5 / <u>73.7</u> / 65.9               | 58.7 / <u>74.0</u> / 65.5               | 20.0 / 68.7                       | 70.6 / <u>52.8</u> / <u>60.4</u>       | 67.8 / <u>59.0</u> / <u>63.1</u>        |
| k-SemStamp   | <b>87.5</b>               | <u>63.3</u> / 33.0 / 43.4               | <u>62.2</u> / 33.0 / 43.1               | <u>20.7</u> / <b>68.9</b>         | 54.2 / 22.0 / 31.3                     | 61.6 / 22.0 / 32.4                      |
| SeqMark      | 87.1                      | <b>76.9</b> / <b>77.3</b> / <b>77.1</b> | <b>75.5</b> / <b>83.0</b> / <b>79.0</b> | <b>21.6</b> / 68.5                | <b>81.3</b> / <b>100</b> / <b>89.7</b> | <b>85.3</b> / <b>85.3</b> / <b>85.3</b> |

Table 2: Watermarking results for sentence translation and summarization. Best results are **bold** and second best are underlined.  $\uparrow$  denotes the higher the better. For detection, **(h)** and **(nw)** denote the negative examples from human and non-watermarked LM, respectively. SeqMark substantially improves detection while maintaining competitive text quality.

respectively. For paragraph translation, we use Gemma-2-4B-it (Gemma Team, 2024) due to its long context window. For code generation, we use CodeLlama-7B (Rozière et al., 2024) and Gemma embeddings (Vera et al., 2025). For other settings, we use Llama-2-7B-Chat (Touvron et al., 2023) and SBERT encoder (Reimers & Gurevych, 2019).<sup>4</sup>

**Baselines:** We compare our approach against token-level watermarking approaches KGW (Kirchenbauer et al., 2023), and SWEET (Lee et al., 2024), and sequence-level watermarking approaches SemStamp (Hou et al., 2024a), and k-SemStamp (Hou et al., 2024b). Given different sets of hyperparameters for each algorithm, we first sweep for the best hyperparameters over 100 samples to find the most promising configurations, then run them on the full evaluation sets (§A.1).

**Evaluation:** Two important dimensions for evaluating watermark algorithms are *text quality* and *watermark detectability*. Text quality metrics often depend on the specific tasks and are thus described in the corresponding sections below. For watermark detection, previous works utilizing statistical hypothesis testing often report classification metrics such as AUROC and True Positive Rate (at a specific False Positive Rate, e.g. FPR=1%). While we also report these metrics for longer generation tasks (paragraph translation, open-ended generation), for sentence translation, summarization, and code generation these metrics are ill-suited since the hypothesis test is often over a single sequence. Therefore, we treat each query as a binary classification problem: either it was detected as a watermark or not. We then compute precision, recall, and F<sub>1</sub> on the detection predictions.<sup>5</sup>

Finally, we report detection results for two types of negative examples: a) human **(h)** to evaluate differentiation between watermarked LM and human generated text, and b) non-watermarked LM **(nw)**, to evaluate differentiation between watermarked generation and non-watermarked generation from the same language model.

## 7 Results and Analysis

### 7.1 Main Results

**Sentence translation, summarization, and code generation:** For translation evaluation, we use COMET, a neural, semantic-based metric (Rei et al., 2020). Since COMET is not specifically trained for summarization, we include ROUGE-L (Lin, 2004) for summarization results. For code generation, we use pass@1 metric. We observe in Table 2 that SeqMark substantially improves detection results while maintaining competitive text quality for sentence translation and summarization. This result is in congruent with Table 1, where the high cluster entropy score and low cosine similarity indicates that SeqMark correctly defines the high-quality output space and partitions it more uniformly for more effective

<sup>4</sup>Further experiment with sentence encoders is in §A.4.

<sup>5</sup>Alternatively, one can aggregate the sequences over multiple examples in the evaluation set for AUROC computation. We experimented with this aggregated AUROC metric in §A.2 and found that the results are consistent with our conclusions.

|              | Code Generation (MBPP) |                                  |                                  | Open-ended Generation (C4) |                                         |                                        |
|--------------|------------------------|----------------------------------|----------------------------------|----------------------------|-----------------------------------------|----------------------------------------|
|              | pass@1 $\uparrow$      | P / R / F <sub>1</sub> (h)       | P / R / F <sub>1</sub> (nw)      | PPL $\downarrow$           | AUROC / TP@1 / TP@5 (h)                 | AUROC / TP@1 / TP@5 (nw)               |
| No Watermark | 33.8                   | -                                | -                                | 3.4                        | -                                       | -                                      |
| KGW          | 15.0                   | <b>87.5</b> / 31.0 / 45.7        | <b>86.5</b> / 33.0 / 45.9        | 3.6                        | <u>99.0</u> / 92.0 / 95.0               | <b>99.9</b> / <b>99.0</b> / <b>100</b> |
| SWEET        | 33.2                   | <u>79.5</u> / 62.3 / <u>70.2</u> | <u>80.5</u> / 63.2 / <u>71.6</u> | 3.6                        | 97.6 / <u>93.0</u> / <b>97.0</b>        | 97.3 / 93.0 / 97.0                     |
| SemStamp     | <b>34.2</b>            | 66.9 / <u>71.8</u> / 69.2        | 66.8 / 72.0 / 69.1               | 3.6                        | 98.7 / 90.9 / <b>97.0</b>               | <u>99.0</u> / 85.9 / <u>98.0</u>       |
| k-SemStamp   | <u>34.0</u>            | 59.9 / 71.7 / 65.2               | 58.7 / <u>74.0</u> / 65.5        | 3.6                        | 94.8 / 80.0 / 87.0                      | 94.8 / 80.0 / 87.0                     |
| SeqMark      | 33.6                   | 76.0 / <b>86.0</b> / <b>80.7</b> | 76.3 / <b>87.0</b> / <b>81.3</b> | <b>3.4</b>                 | <b>99.3</b> / <b>94.0</b> / <b>97.0</b> | <u>99.0</u> / <u>94.0</u> / 96.0       |

Table 3: Code generation with MBPP and open-ended generation with C4 RealNews

watermarking. While SemStamp improves over token-level approaches, we confirm our hypothesis that  $k$ -SemStamp exacerbates region collapse by observing that it performs worse than SemStamp. For code generation, SeqMark outperforms previous best baseline SWEET (Table 3). Finally, the trends are similar for both the types of negatives, human and non-watermarked LM, indicating that SeqMark successfully differentiates against texts from other sources.

**Open-ended generation:** Similar to previous work, we evaluate watermarking algorithms on open-ended generations using C4 RealNews (Table 3). All watermarking algorithms perform well in this high-entropy setting, with SeqMark performing on par with other baselines without causing significant degradation in perplexity.

**Paragraph translation:** We also evaluate our approach on paragraph translation, where the LM is tasked with translating multiple sentences. For this task, we concatenate 8-10 sentences from the same article in WMT23 dataset such that the input prompt fits within the context window of Gemma-2-4B-it. Since COMET is not specifically trained for paragraph evaluation, we also report BLEU (Papineni et al., 2002). Results in Table 4 shows a similar trend as Table 2: SeqMark has significantly higher detection score than KGW and SemStamp, while maintaining similar text quality.

| Approach     | BLEU / COMET              | AUROC / TP@1 / TP@5 (h)                 |
|--------------|---------------------------|-----------------------------------------|
| No Watermark | 39.2 / 87.1               | -                                       |
| KGW          | 40.7 / <b>87.7</b>        | 72.1 / 11.0 / 25.0                      |
| SWEET        | 39.2 / 87.4               | 60.6 / 10.0 / 16.2                      |
| SemStamp     | <b>42.8</b> / 87.5        | <u>84.0</u> / <u>30.0</u> / <u>40.0</u> |
| k-SemStamp   | <u>42.2</u> / <u>87.6</u> | 53.5 / 1.0 / 6.0                        |
| SeqMark      | 39.8 / <b>87.7</b>        | <b>100</b> / <b>100</b> / <b>100</b>    |

Table 4: Paragraph translation with WMT23

## 7.2 Effect of Transformations $f$

The goal of the transformation  $f$  in SeqMark is to reduce cosine similarity between high-quality generations. We propose a transformation that modifies sentence embedding by subtracting the sample mean i.e.  $f(\mathbf{c}_i; C) = \mathbf{c}_i - \mathbf{z}$ , where  $\mathbf{z} = 1/n \sum_{\mathbf{c} \in C} \mathbf{c}$ . In Table 5, we investigate other transformations obtained by subtracting vectors  $\mathbf{z}$  other than the mean from the embeddings: random embedding, the sample closest to the mean, a single sample embedding  $\mathbf{c}$ , the source sentence embedding, and the ground truth target translation embedding. We observe that all transformations perform better than subtracting a random embedding, indicating the need for preservation of relative relationships among the samples. Importantly, aggregated representations such as sample mean and point closest-to-the-mean perform the best, while single point embedding alternatives do not outperform SemStamp baseline.

| Approach               | COMET $\uparrow$ | P / R / F <sub>1</sub> (h)              |
|------------------------|------------------|-----------------------------------------|
| KGW                    | <b>85.5</b>      | 67.2 / 37.0 / 47.8                      |
| SemStamp               | 85.2             | 58.7 / 74.0 / 65.5                      |
| Sample mean            | 85.0             | <b>75.9</b> / <u>85.0</u> / <u>80.2</u> |
| Random embedding       | 85.4             | 51.3 / 19.0 / 27.7                      |
| Sample closest-to-mean | 85.4             | <u>75.0</u> / <b>87.0</b> / <b>80.6</b> |
| Single sample          | 85.3             | 71.1 / 59.0 / 64.5                      |
| Source embedding       | 84.9             | 67.9 / 53.0 / 59.6                      |
| Target embedding       | <u>85.4</u>      | 64.8 / 59.0 / 61.8                      |

Table 5: Different transformations  $f$  on WMT19

### 7.3 Computational Cost and Fast-SeqMark

#### Varying number of samples to compute mean $\mathbf{z}$

In all our experiments, we use  $n = 50$  as the number of sampled sequences to compute the mean embedding  $\mathbf{z}$ . To measure the effect of  $n$  on the inference efficiency vs. performance tradeoffs of SeqMark, we vary  $n$  across different values, using WMT19 as evaluation data. We benchmark the results against the inference speed of sequence-level algorithm SemStamp, using  $n = 20$  samples. Inference speed is measured by seconds per input prompt. The experiment is performed on a single NVIDIA A40 GPU. Result in Figure 4 shows that as both inference speed and watermark performance increase as we increase  $n$ . However, watermark performance plateaus for values of  $n$  above 20, while inference speed for  $n = 10$  and  $n = 20$  are comparable to SemStamp. Thus, a value of  $n$  between 10-20 samples presents a reasonable choice for balancing between watermark performance and inference speed.

**Fast-SeqMark** As mentioned above, SeqMark samples high-quality generations to estimate the sample mean for the transformation. Consequently during detection, computing the mean embedding via sampling requires computation and access to the language model for detection. To avoid this step altogether, we propose a lightweight solution where a neural network is trained to map the input prompt to the sample mean embedding  $\mathbf{z}$  to subtract. We experiment with this Fast-SeqMark solution on the WMT19 German-English dataset. For training data generation, we use ALMA-7B to generate 50 samples each for 100,000 input prompts with a fixed temperature of 1.2. We then finetune the LABSE encoder on this dataset for mean prediction. Table 6 shows that when compared to using the actual mean embeddings, Fast-SeqMark results in a drop in detection score, yet still outperforms KGW and SemStamp.

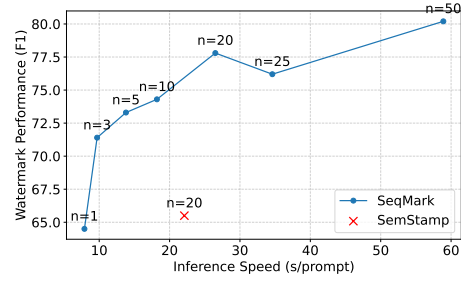


Figure 4: Performance-speed trade-off for varying  $n$ . For text quality, COMET scores range from 84.9-85.3, well within the score of SemStamp (85.2)

| Approach     | COMET $\uparrow$ | P / R / F <sub>1</sub> (h) |
|--------------|------------------|----------------------------|
| KGW          | <b>85.5</b>      | 67.2 / 37.0 / 47.8         |
| SemStamp     | <u>85.2</u>      | 58.7 / 74.0 / 65.5         |
| SeqMark      | 85.0             | <b>75.9 / 85.0 / 80.2</b>  |
| Fast-SeqMark | 84.4             | <u>68.1 / 75.0 / 71.4</u>  |

Table 6: Fast-SeqMark on WMT19

## 8 Related Work

**Language Model Watermarking** Since its introduction, LM Watermarking has become an important technique to combat LLM-related security concerns (Suvra et al., 2023; Srinivasan, 2024). Most algorithms propose adding imperceptible statistical signals to different stages of text generation, including logit generation (Kirchenbauer et al., 2023; Zhao et al., 2023; Hu et al., 2023; Liu et al., 2024), token sampling (Christ et al., 2024; Kuditipudi et al., 2023; Mao et al., 2025), or the model weights (Sun et al., 2023; Gu et al., 2023). Orthogonally, several work propose sequence-level watermarking (Hou et al., 2024a;b) that involves sampling full sentences, encodes them into a semantic space, and selects the generations within the valid semantic regions. Overall, the majority of work in LM watermarking concerns with open-ended text generation which affords high-entropy thus enabling highly effective watermarking (Ajith et al., 2024), leaving watermarking for constrained text generation tasks such as machine translation and summarization remains underexplored, despite these use cases constituting a non-trivial proportion of LLM usage.

**Watermarking Low-Entropy Sequences** Prior work on watermarking low-entropy sequences has mostly focused on code generation: Lee et al. (2024) extends Kirchenbauer et al. (2023) by selecting and watermarking high-entropy tokens, Lu et al. (2024) modifies the detection algorithm to include token entropy as weights in the final score, and Gu et al. (2025) effectively reduces watermarking efficiency without major loss in performance. Similar to

us, Cai et al. (2025) and Tsur et al. (2025) also addressed the trade-offs in watermarking low-entropy outputs; however, their work only consider token-level entropy. For translation, Takezawa et al. (2025) modifies KGW by considering the minimal number of watermarked tokens needed; nonetheless, their approach is not imperceptible– it is visually easy to detect the watermarked tokens (§A.6). Orthogonal to all previous work focusing on extending token-level watermarking, our work focuses on the strengths and weaknesses of sequence-level watermarking, such as the region collapse issue for watermarking low-entropy texts. Interestingly, we observe that sequence-level watermarking outperforms token-level algorithms in general, possibly due to underestimation of uncertainty by token-level algorithms (Kuhn et al., 2023).

## 9 Conclusion

In this work, we demonstrate the inadequacy of current watermarking algorithms for constrained generation tasks. We posit that all token-level algorithms perform poorly because they fail to utilize the semantic entropy induced by the LMs on these tasks. While sequence level watermarking algorithms are a better paradigm, we identify a different issue of region collapse in the operationalization of existing semantic watermarking algorithms, resulting in poor performance at watermarking constrained generation tasks. To overcome these limitations, we propose SeqMark, a sequence-level watermarking algorithm that ameliorates the region collapse issue by carefully partitioning the space for watermarking so as to ensure an even spread of high-quality outputs among the partitions. This results in reliably detectable watermarks while generating high-quality outputs for low-entropy tasks.

## Limitations

During detection, SeqMark requires access to the language model as well as precise reproduction of the sampled generations to compute mean embeddings. Despite the fact that these are not ideal, we present Fast-SeqMark as a viable alternative that alleviates these requirements. In addition, prior work (Lee et al., 2024; Lu et al., 2024) also requires access to the LM during detection for computing token-level entropy. This access is readily available for open-weights models and could also be requested for realistic auditing purposes.

Another limitation is that, unlike SemStamp /  $k$ -SemStamp, SeqMark might be susceptible to paraphrasing attacks. By design, SeqMark spreads similar generations more evenly in the semantic space; thus a strong paraphraser can potentially evade the watermark by transforming the generation into another semantic region. However, in our preliminary paraphrase attack experiment (§A.5), we observe that the paraphrasing outputs often have low text-quality, indicating that current paraphrasing models are not strong enough to maintain text quality, especially for low-entropy tasks like translations.

Like other LM watermarking technologies, SeqMark is designed to help enhance safe usages of language models. Nonetheless, it also carries potential risks that come with misusing these technologies. For example, it is susceptible to spoofing attack, where a malicious actor takes advantage of watermark robustness by embedding the watermark into unethical texts, which can then be misattributed to specific language model providers. Therefore, we recommend users exercise caution to avoid exposing important details such as the LSH hash function key.

## References

- Anirudh Ajith, Sameer Singh, and Danish Pruthi. Downstream trade-offs of a family of text watermarks. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14039–14053, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.821. URL <https://aclanthology.org/2024.findings-emnlp.821/>.
- Isabelle Augenstein, Timothy Baldwin, Meeyoung Cha, Tanmoy Chakraborty, Giovanni Luca Ciampaglia, David Corney, Renee DiResta, Emilio Ferrara, Scott Hale, Alon

- Halevy, et al. Factuality challenges in the era of large language models and opportunities for fact-checking. *Nature Machine Intelligence*, 6(8):852–863, 2024.
- Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL <https://arxiv.org/abs/2108.07732>.
- Ondřej Bojar, Yvette Graham, Christof Monz, Matt Post, Ondřej Bojar, Yvette Graham, Christof Monz, and Matt Post. Acl 2019 fourth conference on machine translation (wmt19), shared task: Machine translation of news. 2019. URL <http://www.statmt.org/wmt19/translation-task.html>.
- Shizhan Cai, Liang Ding, and Dacheng Tao. Entropy-guided watermarking for llms: A test-time framework for robust and traceable text generation, 2025. URL <https://arxiv.org/abs/2504.12108>.
- Moses Charikar. Similarity Estimation Techniques from Rounding Algorithms. In *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, pp. 380–388. ACM, 2002.
- Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 1125–1139. PMLR, 2024.
- Jinhao Duan, Hao Cheng, Shiqi Wang, Alex Zavalny, Chenan Wang, Renjing Xu, Bhavya Kailkhura, and Kaidi Xu. Shifting attention to relevance: Towards the predictive uncertainty quantification of free-form large language models. *arXiv preprint arXiv:2307.01379*, 2023.
- Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic BERT sentence embedding. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 878–891, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.62. URL <https://aclanthology.org/2022.acl-long.62/>.
- Pierre Fernandez, Antoine Chaffin, Karim Tit, Vivien Chappelier, and Teddy Furon. Three bricks to consolidate watermarks for large language models. In *2023 IEEE international workshop on information forensics and security (WIFS)*, pp. 1–6. IEEE, 2023.
- Gemma Team. Gemma 2: Improving Open Language Models at a Practical Size, 2024. URL <https://arxiv.org/abs/2408.00118>.
- Jocelyn Gravel, Madeleine D’Amours-Gravel, and Esli Osmanlliu. Learning to fake it: limited responses and fabricated references provided by chatgpt for medical questions. *medRxiv*, 2023. doi: 10.1101/2023.03.16.23286914. URL <https://www.medrxiv.org/content/early/2023/03/24/2023.03.16.23286914>.
- Chenchen Gu, Xiang Lisa Li, Percy Liang, and Tatsunori Hashimoto. On the learnability of watermarks for language models. *arXiv preprint arXiv:2312.04469*, 2023.
- Tianle Gu, Zongqi Wang, Kexin Huang, Yuanqi Yao, Xiangliang Zhang, Yujiu Yang, and Xi-ying Chen. Invisible entropy: Towards safe and efficient low-entropy llm watermarking, 2025. URL <https://arxiv.org/abs/2505.14112>.
- Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4067–4082, Mexico City, Mexico, June 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.226. URL <https://aclanthology.org/2024.naacl-long.226/>.

- Abe Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1706–1715, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.98. URL <https://aclanthology.org/2024.findings-acl.98/>.
- Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. *arXiv preprint arXiv:2310.10669*, 2023.
- Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, STOC '98*, pp. 604–613, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919629. doi: 10.1145/276698.276876. URL <https://doi.org/10.1145/276698.276876>.
- John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17061–17084. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/kirchenbauer23a.html>.
- Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Philipp Koehn, Benjamin Marie, Christof Monz, Makoto Morishita, Kenton Murray, Masaaki Nagata, Toshiaki Nakazawa, Martin Popel, Maja Popović, Mariya Shmatova, and Jun Suzuki. Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In Philipp Koehn, Barry Haddow, Tom Kocmi, and Christof Monz (eds.), *Proceedings of the Eighth Conference on Machine Translation*, pp. 1–42, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.wmt-1.1. URL <https://aclanthology.org/2023.wmt-1.1/>.
- Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.
- Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.
- Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. Who wrote this code? watermarking for code generation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4890–4911, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.268. URL <https://aclanthology.org/2024.acl-long.268/>.
- Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL <https://aclanthology.org/W04-1013/>.
- Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models, 2024. URL <https://arxiv.org/abs/2310.06356>.
- Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. An entropy-based text watermarking detection method. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11724–11735, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.630. URL <https://aclanthology.org/2024.acl-long.630/>.

- Minjia Mao, Dongjun Wei, Zeyu Chen, Xiao Fang, and Michael Chau. Watermarking large language models: An unbiased and low-risk method. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7939–7960, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.391. URL <https://aclanthology.org/2025.acl-long.391/>.
- John X. Morris and Alexander M. Rush. Contextual document embeddings, 2024. URL <https://arxiv.org/abs/2410.02525>.
- Sumeet Ramesh Motwani, Mikhail Baranchuk, Martin Strohmeier, Vijay Bolina, Philip H.S. Torr, Lewis Hammond, and Christian Schroeder de Witt. Secret collusion among ai agents: Multi-agent deception via steganography. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 73439–73486. Curran Associates, Inc., 2024. doi: 10.52202/079017-2336. URL [https://proceedings.neurips.cc/paper\\_files/paper/2024/file/861f7dad098aec1c3560fb7add468d41-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2024/file/861f7dad098aec1c3560fb7add468d41-Paper-Conference.pdf).
- Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun’ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL <https://aclanthology.org/D18-1206/>.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin (eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL <https://aclanthology.org/P02-1040/>.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL <http://jmlr.org/papers/v21/20-074.html>.
- Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. COMET: A neural framework for MT evaluation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2685–2702, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.213. URL <https://aclanthology.org/2020.emnlp-main.213/>.
- Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL <https://arxiv.org/abs/1908.10084>.
- Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code, 2024. URL <https://arxiv.org/abs/2308.12950>.
- Siddharth Srinivasan. Detecting ai fingerprints: A guide to watermarking and beyond, 2024. URL <https://www.brookings.edu/articles/detecting-ai-fingerprints-a-guide-to-watermarking-and-beyond/>.

Miriam Sullivan, Andrew Kelly, and Paul McLaughlan. Chatgpt in higher education: Considerations for academic integrity and student learning. 1, 2023. URL <https://api.semanticscholar.org/CorpusID:257675337>.

Zhensu Sun, Xiaoning Du, Fu Song, and Li Li. Codemark: Imperceptible watermarking for code datasets against neural code completion models. In *Proceedings of the 31st ACM joint European software engineering conference and symposium on the foundations of software engineering*, pp. 1561–1572, 2023.

Soumya Suvra, Ghosal, Souradip Chakraborty, Jonas Geiping, Furong Huang, Dinesh Manocha, and A. S. Bedi. A survey on the possibilities & impossibilities of ai-generated text detection. *Trans. Mach. Learn. Res.*, 2023, 2023. URL <https://api.semanticscholar.org/CorpusID:266982110>.

Yuki Takezawa, Ryoma Sato, Han Bao, Kenta Niwa, and Makoto Yamada. Necessary and sufficient watermark for large language models, 2025. URL <https://arxiv.org/abs/2310.00833>.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ross Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL <https://arxiv.org/abs/2307.09288>.

Dor Tsur, Carol Xuan Long, Claudio Mayrink Verdun, Hsiang Hsu, Chen-Fu Chen, Haim Permuter, Sajani Vithana, and Flavio P. Calmon. Heavywater and simplexwater: Distortion-free llm watermarks for low-entropy next-token predictions, 2025. URL <https://arxiv.org/abs/2506.06409>.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, Sindhu Raghuram Panyam, Sara Smoot, Iftekhar Naim, Joe Zou, Feiyang Chen, Daniel Cer, Alice Lisak, Min Choi, Lucas Gonzalez, Omar Sanseviero, Glenn Cameron, Ian Ballantyne, Kat Black, Kaifeng Chen, Weiyi Wang, Zhe Li, Gus Martins, Jinhyuk Lee, Mark Sherwood, Juyeong Ji, Renjie Wu, Jingxiao Zheng, Jyotinder Singh, Abheesht Sharma, Divyashree Sreepathihalli, Aashi Jain, Adham Elarabawy, AJ Co, Andreas Doumanoglou, Babak Samari, Ben Hora, Brian Potetz, Dahun Kim, Enrique Alfonseca, Fedor Moiseev, Feng Han, Frank Palma Gomez, Gustavo Hernández Ábrego, Heseng Zhang, Hui Hui, Jay Han, Karan Gill, Ke Chen, Koert Chen, Madhuri Shanbhogue, Michael Boratko, Paul Suganthan, Sai Meher Karthik Duddu, Sandeep Mariserla, Setareh Ariafer, Shanfeng Zhang, Shijie Zhang, Simon Baumgartner, Sonam Goenka, Steve Qiu, Tanmaya Dabral, Trevor Walker, Vikram Rao, Waleed Khawaja, Wenlei Zhou, Xiaoqi Ren, Ye Xia, Yichang Chen, Yi-Ting Chen, Zhe Dong, Zhongli Ding, Francesco Visin, Gaël Liu, Jiageng Zhang, Kathleen Kenealy, Michelle Casbon, Ravin Kumar, Thomas Mesnard, Zach Gleicher, Cormac Brick, Olivier Lacombe, Adam Roberts, Qin Yin, Yunhsuan Sung, Raphael Hoffmann, Tris Warkentin, Armand Joulin, Tom Duerig, and Mojtaba Seyedhosseini. Embeddinggemma: Powerful and lightweight text representations, 2025. URL <https://arxiv.org/abs/2509.20354>.

Haoran Xu, Young Jin Kim, Amr Sharaf, and Hany Hassan Awadalla. A paradigm shift in machine translation: Boosting translation performance of large language models, 2023.

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text. *ArXiv*, abs/2306.17439, 2023. URL <https://api.semanticscholar.org/CorpusID:259308864>.

Haonan Zhong, Jiamin Chang, Ziyue Yang, Tingmin Wu, Pathum Chamikara Mahawaga Arachchige, Chehara Pathmabandu, and Minhui Xue. Copyright protection and accountability of generative ai: Attack, watermarking and attribution. In *Companion Proceedings of the ACM Web Conference 2023, WWW '23 Companion*, pp. 94–98, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450394192. doi: 10.1145/3543873.3587321. URL <https://doi.org/10.1145/3543873.3587321>.

## A Appendix

### A.1 Hyperparameter Sweep for Translation and Summarization

Given different sets of hyperparameters for each watermark methods, to ensure apple-to-apple comparisons we sweep over a range of values of each method’s relevant hyperparameters (Table 7). For each task, we sample 100 evaluation samples and run these watermark hyperparameter configs on these subset to obtain the Pareto curves in Figure 1. The best configs are then used to run the full evaluation set reported in Tables 2, 3, and 4.

| Hyperparameter                            | Methods                               | Values                     |
|-------------------------------------------|---------------------------------------|----------------------------|
| Text generation temperature $t$           | KGW, SemStamp, $k$ -SemStamp, SeqMark | [0.7, 0.85, 1.0, 1.2, 1.5] |
| Green/red list and regions ratio $\gamma$ | KGW, SemStamp, $k$ -SemStamp, SeqMark | [0.1, 0.25, 0.5, 0.75]     |
| Logit bias $\delta$                       | KGW                                   | [0.1, 0.5, 1.0, 2.0, 4.0]  |
| LSH dimension $n$                         | SemStamp, $k$ -SemStamp, SeqMark      | [2, 3, 4, 5]               |
| High-quality cluster samples $c$          | SeqMark                               | [50]                       |

Table 7: Relevant hyperparameter for watermark methods

### A.2 Full Experimental Results

As mentioned in §6, for sentence translation, summarization, and code generation, we reported precision, recall,  $F_1$  and not the usual AUROC / TP@FPR=1 / TP@FPR=5. This is due to the nature of the evaluated tasks: since we only generate a handful (often single) sequences per example, the z-test employed in sequence-level watermarking SemStamp is severely underpowered. In fact, sequence-level watermark detection for these tasks is effectively binary classification: the sentence is watermarked or not. Therefore, the standard AUROC metric, where the area is computed by sweeping over different classification thresholds, is ill-suited for tasks that only contain a single classification threshold. This is not a problem for token-level watermarking, however, since there are enough tokens for the z-test. To ensure fairness of comparison between the token-level baselines and sentence-level approaches, we select the threshold that yields the highest true positive rate / false positive rate ratio and reported the corresponding precision, recall, and  $F_1$ .

To ensure the robustness of our study, we also experiment with *aggregated AUROC*, where instead of considering per-example sequence, we aggregate the sequences across multiple examples of the evaluation set. Specifically, each new example now consists of 30 sequences across consecutive examples. For example, in translation, if there are 3000 examples where each example generates one sentence translation, then we aggregate across 30 examples, for a total of 100 data points. We then compute the AUROC / TP@1 / TP@5 for this new evaluation set. Results in Table 8 and 9 show consistency between aggregated AUROC and  $F_1$ . In fact, our approach SeqMark achieves perfect classification score under aggregated AUROC. We also plot the aggregated ROC across different thresholds in Figure 5.

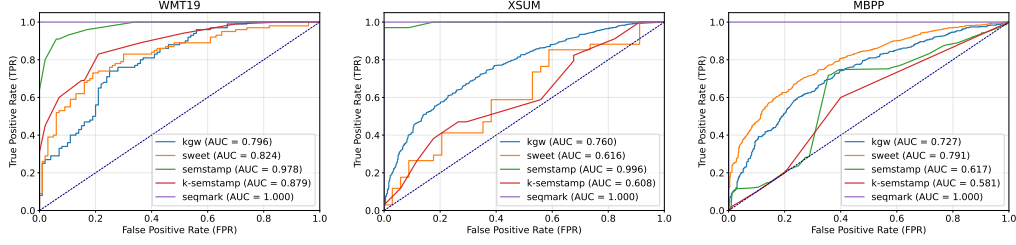


Figure 5: Aggregated ROC across different classification thresholds for sentence translation (right), summarization (middle), and code generation (left).

|              | Translation (WMT19 De-En) |                            |                         |                                   | Summarization (XSum)       |                         |  |  |
|--------------|---------------------------|----------------------------|-------------------------|-----------------------------------|----------------------------|-------------------------|--|--|
|              | COMET $\uparrow$          | P / R / F <sub>1</sub> (h) | agg-AUROC / TP@1 / TP@5 | R-L $\uparrow$ / COMET $\uparrow$ | P / R / F <sub>1</sub> (h) | agg-AUROC / TP@1 / TP@5 |  |  |
| No Watermark | 87.4                      | -                          | -                       | 20.5 / 69.0                       | -                          | -                       |  |  |
| KGW          | 87.4                      | 57.2 / 36.6 / 45.7         | 79.6 / 25.0 / 29.0      | 20.1 / 68.8                       | 85.2 / 30.1 / 44.9         | 75.2 / 26.4 / 45.8      |  |  |
| SWEET        | 87.2                      | 57.5 / 30.0 / 39.5         | 82.4 / 20.0 / 47.0      | 18.5 / 68.7                       | 52.4 / 49.9 / 51.1         | 52.8 / 2.9 / 10.7       |  |  |
| STA-1        | 87.1                      | 60.6 / 30.1 / 40.2         | 59.0 / 5.2 / 14.0       | -                                 | -                          | -                       |  |  |
| SemStamp     | 87.4                      | 59.5 / 73.7 / 65.9         | 97.8 / 63.0 / 91.0      | 20.0 / 68.7                       | 70.6 / 52.8 / 60.4         | 99.6 / 76.4 / 91.2      |  |  |
| k-SemStamp   | <b>87.5</b>               | 63.3 / 33.0 / 43.4         | 87.9 / 48.0 / 55.0      | 20.7 / <b>68.9</b>                | 54.2 / 22.0 / 31.3         | 60.8 / 3.9 / 11.8       |  |  |
| SeqMark      | 87.1                      | <b>76.9 / 77.3 / 77.1</b>  | <b>100 / 100 / 100</b>  | <b>21.6 / 68.5</b>                | <b>81.3 / 100 / 89.7</b>   | <b>100 / 100 / 100</b>  |  |  |

Table 8: Results on precision, recall, F<sub>1</sub> and aggregated AUROC metrics for sentence translation and summarization. We evaluated with human generations as negative examples. Best results are **bold**. We also include an additional baseline STA-1 for translation (Mao et al., 2025)

| Approach     | pass@1      | P / R / F <sub>1</sub> (h) | agg-AUROC / TP@1 / TP@5 |
|--------------|-------------|----------------------------|-------------------------|
| No Watermark | 33.8        | -                          | -                       |
| KGW          | 15.0        | 87.5 / 31.0 / 45.7         | 71.2 / 8.0 / 17.0       |
| SWEET        | 33.2        | 79.5 / 62.3 / 70.2         | 80.5 / 63.2 / 71.6      |
| STA-1        | 34.0        | 55.4 / 21.6 / 31.1         | 54.5 / 8.6 / 17.4       |
| SemStamp     | <b>34.2</b> | 66.9 / 71.8 / 69.2         | 61.7 / 8.0 / 12.0       |
| k-SemStamp   | 34.0        | 50.0 / 98.7 / 66.5         | 58.1 / 2.0 / 5.0        |
| SeqMark      | 33.6        | <b>76.0 / 86.0 / 80.7</b>  | <b>100 / 100 / 100</b>  |

Table 9: Results on precision, recall, F<sub>1</sub> and aggregated AUROC metrics for code generation. We also include an additional baseline for low-entropy texts STA-1 (Mao et al., 2025).

### A.3 Detecting Watermarked LLM against other LMs

In addition to testing the ability of watermarking schemes to differentiate between watermarked texts and texts generated either by humans or non-watermarked LM, we also report results on the ability to differentiate between watermarked texts and texts generated from other (non-watermarked) language models. Using 100 samples from WMT-19 German-English as test bed, we compare different watermarking methods on ALMA-7B with unwatermarked completions from ALMA-7B, Gemma-2-4B instruct, and Llama-2-7B-Chat. Results from Table 11 indicate that SeqMark differentiates the watermarked text of a particular LM from other text sequences, be it human-generated or other LM-generated equally well.

### A.4 Choice of Sentence Encoders

| Method (Encoder)             | ROUGE-L / COMET | P / R / F <sub>1</sub> (h) |
|------------------------------|-----------------|----------------------------|
| SemStamp (all-mpnet-base-v1) | 21.1 / 68.7     | 70.6 / 52.8 / 60.4         |
| SeqMark (all-mpnet-base-v1)  | 20.6 / 68.5     | 76.1 / 100 / 86.4          |
| SemStamp (cde-small-v2)      | 22.0 / 68.3     | 80.0 / 48.8 / 60.6         |
| SeqMark (cde-small-v2)       | 21.2 / 68.3     | 79.5 / 100 / 88.6          |

Table 10: Comparison of methods across different encoders.

| Approach | COMET $\uparrow$ | Human       | ALMA        | Gemma-2     | Llama-2     |
|----------|------------------|-------------|-------------|-------------|-------------|
| KGW      | <b>85.5</b>      | 47.8        | 47.6        | 42.8        | 43.2        |
| SemStamp | 85.2             | 65.5        | 65.5        | 66.7        | 67.6        |
| SeqMark  | 85.0             | <b>80.2</b> | <b>79.0</b> | <b>78.7</b> | <b>81.8</b> |

Table 11:  $F_1$  on WMT19 translation task testing detection against negative examples from various sources: human and other non-watermarked LLMs.

| Method                         | COMET |
|--------------------------------|-------|
| KGW                            | 85.5  |
| KGW (paraphrase attacked)      | 79.3  |
| SemStamp                       | 85.2  |
| SemStamp (paraphrase attacked) | 75.4  |
| SeqMark                        | 85.0  |
| SeqMark (paraphrase attacked)  | 80.1  |

Table 12: Generation quality under paraphrasing attacks, for sentence translation task.

Our choice of sentence encoder was task-dependent: LABSE encoder for machine translation and SBERT for summarization. We also experimented with other encoders (Morris & Rush, 2024) and did not see major variations in results, indicating the robustness of SeqMark across different sentence encoders (Table 10).

#### A.5 Discussion on Paraphrasing Attacks

A potential limitation of our approach is that, unlike SemStamp / k-SemStamp, it might be susceptible to paraphrasing attacks. By design, SeqMark spreads similar generations more evenly in the semantic space; thus a strong paraphraser can potentially evade the watermark by transforming the generation into another semantic region. However, in our preliminary paraphrase attack experiments following SemStamp, we observe that the paraphrasing outputs often have low text-quality, indicating that current paraphrasing models are not strong enough to maintain text quality, especially for low-entropy tasks like translations (Table 12). In addition, existing sentence-level algorithms e.g. SemStamp does not have strong watermarking performance on these tasks, even without paraphrasing attacks (Table 2).

#### A.6 Imperceptibility when Watermarking Translation Outputs

Takezawa et al. (2025) recently showed that watermarking with minimal intervention is possible for machine translation tasks via their token-level NS-Watermarking method. When compared to our approach, we observed that this method achieves near perfect watermark detection performance but suffers in terms of quality measured by COMET with  $\sim 6\%$  absolute decrease. Additionally, they acknowledge that by design their approach is not imperceptible – it is easy to tell if the output has been watermarked. Looking at Table 15, it is easy to notice the minimal yet awkward token choices for watermarked translation. Concerningly, this imperceptibility makes this approach susceptible to simple post-editing attacks. On the other hand, our approach results in imperceptible watermarked generation, making it robust to simple attacks.

#### A.7 Proof of Theorem 1

In this section, we aim to show that SeqMark mean-centering transformation improves detection accuracy for LSH-based sentence-level watermarking, assuming the sentence embeddings are similar to each other (Theorem 1).

**Preliminaries** We first define the relevant preliminaries for our analysis

- **Sequence Encoding:** Given  $n$  candidate sequences  $c_1, \dots, c_n$ , sentence embedding  $v_i$  is produced using sentence encoder  $E$ :  $\mathbf{x}_i = E(c_i) \in \mathbb{R}^h, \forall 1 \leq i \leq n$
- **Locality Sensitive Hasing for Text Watermarking** (Hou et al., 2024a): Given dimension  $h$ , we sample  $d$  random vectors from a multivariate normal distribution (i.e.  $\mathbf{g}^{(i)} \sim \mathcal{N}(\mathbf{0}, I_h), \forall 0 \leq i \leq d$ ) to specify  $d$  random hyperplanes and thus  $2^d$  semantic regions. For sentence-level watermarking, we are given a "valid region ratio"  $\gamma$  denoting the fraction of valid regions.
- **LSH Collision Probability** (Charikar, 2002): given two vectors  $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^h$  with angle  $\theta_{ij} \in [0, \pi]$ , the probability that they lie in the same region (i.e. having the same LSH signature) is:

$$P(\text{sign}(\mathbf{x}_i) = \text{sign}(\mathbf{x}_j)) = \left(1 - \frac{\theta_{ij}}{\pi}\right)^d$$

where  $\text{sign}(\mathbf{x})$  is the concatenation of the dot product between  $\mathbf{x}$  and all the normal vectors  $\mathbf{g}^{(i)}$ . For simplicity, let  $q_{ij} = \left(1 - \frac{\theta_{ij}}{\pi}\right)^d$

- **Detection Accuracy** An important quantity for our analysis is the indicator  $I^{(x)}$ , where  $I^{(x)} = 1$  if there exists at least one vector  $\mathbf{x}_i$  in the valid regions, and 0 otherwise. This quantity is the watermark detection accuracy of a single example, since  $I^{(x)} = 1$  if there exists at least one sentence embedding  $\mathbf{x}_i$  that lies in the valid region.

**Proposition 1** Given  $n$  vectors  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^h$ ,  $2^d$  semantic regions partitioned using LSH, and  $\gamma$  ratio of valid regions, we can approximate the average detection accuracy  $E[I]$  as:

$$E[I^{(x)}] \approx n\gamma - \binom{n}{2}\gamma^2 - \gamma(1-\gamma) \sum_{i < j} \left(1 - \frac{\theta_{ij}}{\pi}\right)^d$$

**Proof** Let  $A_i$  be the event that vector  $x_i$  lands in a valid region. We then have:

$$\begin{aligned} E[I^{(x)}] &= P(\cup_i A_i) \\ &= \sum_i P(A_i) - \sum_{i < j} P(A_i \cap A_j) \\ &= + \sum_{i < j < k} P(A_i \cap A_j \cap A_k) - \dots \\ &\quad \text{(inclusion-exclusion principle)} \end{aligned}$$

where  $P(A_i)$  denotes the probability that vector  $\mathbf{x}_i$  lands in a valid region,  $P(A_i \cap A_j)$  denotes the probability that vectors  $\mathbf{x}_i, \mathbf{x}_j$  land in valid regions (which can be the same or different), etc.

A reasonable assumption is that the higher-order terms are smaller in mass (give a one line reason). To simplify the computation, we make the second-order pairwise approximation (i.e. keeping the only two terms):

$$E[I^{(x)}] \approx \sum_i P(A_i) - \sum_{i < j} P(A_i \cap A_j)$$

Now we compute  $P(A_i)$  and  $P(A_i \cap A_j)$ . The probability that a vector  $\mathbf{x}_i$  lands in a valid region is  $P(A_i) = \gamma$ . For  $P(A_i \cap A_j)$ , we have:

$$\begin{aligned} P(A_i \cap A_j) &= P(A_i \cap A_j \mid s_i = s_j)P(s_i = s_j) \\ &\quad + P(A_i \cap A_j \mid s_i \neq s_j)P(s_i \neq s_j) \\ &= \gamma q_{ij} + \gamma^2(1 - q_{ij}) \\ &= \gamma^2 + \gamma(1 - \gamma)q_{ij}, \end{aligned}$$

where  $s_k \equiv \text{sign}(\mathbf{x}_k)$ . Putting everything together, we have

$$\begin{aligned} E[I^{(x)}] &\approx n\gamma - \binom{n}{2}\gamma^2 - \gamma(1 - \gamma) \sum_{i < j} q_{ij} \\ &\approx n\gamma - \binom{n}{2}\gamma^2 - \gamma(1 - \gamma) \sum_{i < j} \left(1 - \frac{\theta_{ij}}{\pi}\right)^d \end{aligned}$$

□

Proposition 1 nicely relates  $E[I^{(x)}]$  to the pairwise angles between vectors  $\theta_{ij}$ . From here we can reason for various value of  $\theta_{ij}$ .

**Collorary 1**  $E[I^{(x)}]$  monotonically nondecreases with respect to  $\theta_{ij}$ , for all  $1 \leq i, j \leq n$

**Proof** The proof basically works by showing that  $dE[I^{(x)}]/d\theta_{ij} \geq 0$ . Fix any pair  $i, j$ . The partial derivative of  $E[I^{(x)}]$  with respect to  $\theta_{ij}$  is

$$\begin{aligned} \frac{\partial E[I^{(x)}]}{\partial \theta_{ij}} &= -\gamma(1 - \gamma) \frac{\partial q_{ij}}{\partial \theta_{ij}} \\ &= \gamma(1 - \gamma) \frac{d}{\pi} \left(1 - \frac{\theta_{ij}}{\pi}\right)^{d-1} \end{aligned}$$

All factors on the right-hand side are nonnegative for  $\theta_{ij} \in [0, \pi]$ . Therefore,

$$\frac{\partial E[I^{(x)}]}{\partial \theta_{ij}} \geq 0,$$

so  $E[I^{(x)}]$  is monotone nondecreasing in each  $\theta_{ij}$ . Moreover, if  $0 < \gamma < 1$ ,  $d > 0$ , and  $\theta_{ij} \in (0, \pi)$ , then the derivative is strictly positive, and  $E[I^{(x)}]$  is strictly increasing in  $\theta_{ij}$ .

□

**Theorem 1** Given  $n$  vectors  $\mathbf{x}_1, \dots, \mathbf{x}_n$  such that  $x_i = \mu + \epsilon_i$  and there exists  $\delta \in [0, 1]$  such that  $\|\epsilon_i\| \leq \delta \|\mu\|$ . Mean-centering transformation result in vectors  $\mathbf{y}_i = x_i - \frac{1}{n} \sum_i x_i = \epsilon_i$ . If the following condition holds for a fixed pair  $i, j$ :

$$\cos \theta_{ij}^{(y)} = \frac{\epsilon_i \cdot \epsilon_j}{\|\epsilon_i\| \|\epsilon_j\|} \leq \frac{1 - 2\delta - \delta^2}{1 + 2\delta + \delta^2}$$

then  $E[I^{(y)}] \geq E[I^{(x)}]$

**Proof** By Proposition 1 and Corollary 1 it suffices to show that for every pair  $(i, j)$

$$\cos \theta_{ij}^{(y)} \leq \cos \theta_{ij}^{(x)}$$

since  $E[I]$  is nonincreasing in  $\cos \theta_{ij}$  (equivalently nondecreasing in  $\theta_{ij}$ ).

Scale  $\mu$  so that  $\|\mu\| = 1$  (the comparison is homogeneous in  $\mu$ ). Recall  $x_i = \mu + \epsilon_i$  and  $x_j = \mu + \epsilon_j$  with  $\|\epsilon_i\|, \|\epsilon_j\| \leq \delta$  Then

$$x_i \cdot x_j = 1 + \mu \cdot (\epsilon_i + \epsilon_j) + \epsilon_i \cdot \epsilon_j$$

We also have

$$\|x_i\| \|x_j\| = \sqrt{(1 + 2\mu \cdot \epsilon_i + \|\epsilon_i\|^2)(1 + 2\mu \cdot \epsilon_j + \|\epsilon_j\|^2)}$$

By Cauchy–Schwarz inequalities, we have the two bounds:  $|\mu \cdot \epsilon_i| \leq \|\mu\| \|\epsilon_i\| \leq \|\epsilon_i\| \leq \delta$  and  $|\epsilon_i \cdot \epsilon_j| \geq -\|\epsilon_i\| \|\epsilon_j\| \geq -\delta^2$  Then

$$\begin{aligned} x_i \cdot x_j &\geq 1 - \|\epsilon_i\| - \|\epsilon_j\| + \epsilon_i \cdot \epsilon_j \\ &\geq 1 - 2\delta + \epsilon_i \cdot \epsilon_j \\ &\geq 1 - 2\delta - \delta^2 \end{aligned}$$

And

$$\begin{aligned} 1 + 2\mu \cdot \epsilon_i + \|\epsilon_i\|^2 &\leq 1 + 2\delta + \delta^2, \\ 1 + 2\mu \cdot \epsilon_j + \|\epsilon_j\|^2 &\leq 1 + 2\delta + \delta^2 \end{aligned}$$

Hence

$$\|x_i\| \|x_j\| \leq 1 + 2\delta + \delta^2$$

Combining these two inequalities yields the uniform lower bound

$$\cos \theta_{ij}^{(x)} = \frac{x_i \cdot x_j}{\|x_i\| \|x_j\|} \geq \frac{1 - 2\delta - \delta^2}{1 + 2\delta + \delta^2}$$

Therefore, if

$$\cos \theta_{ij}^{(y)} \leq \frac{1 - 2\delta - \delta^2}{1 + 2\delta + \delta^2}$$

we have  $\cos \theta_{ij}^{(y)} \leq \cos \theta_{ij}^{(x)}$ . By Corollary 1 (monotonicity of  $E[I]$  in the angles), this implies

$$E[I^{(y)}] \geq E[I^{(x)}]$$

□

## A.8 Algorithm Tables

We include SeqMark text generation in Algorithm 1 and text detection in Algorithm 2

**Algorithm 1:** SeqMark text generation algorithm**Input** : input prompt  $p$ , number of sentences to generate  $T$ **Params**: language model  $P(\cdot)$ , sentence encoder  $E(\cdot)$  with embedding dimension  $h$ , manifold sample size  $n$ , maximum rejections  $N_{\max}$ , LSH dimension  $d$ , valid region ratio  $\gamma \in (0, 1)$  secret key  $k$ **Output**: watermarked sequence  $s_{1:T}$ **def** GENERATE( $p, T$ ):    **Initialization:**        Randomly initialize  $d$  vectors  $n^{(1)} \dots n^{(d)} \in \mathbb{R}^h$ , to create  $2^d$  semantic subspaces.    **for**  $t \leftarrow 1$  **to**  $T$  **do**

// Compute valid and blocked regions

        Compute the LSH signature of the previously generated sentence,  $\text{SIG}(s_{t-1})$ , and use  $[\text{SIG}(s_{t-1})]_{10} \cdot p$  as the seed to randomly divide the signature space  $\{0, 1\}^d$  into a *valid region set*  $G_t$  of size  $\gamma 2^d$  and a *blocked region set*  $R_t$  of size  $(1 - \gamma) 2^d$ .

// Estimate mean embedding of high-quality manifold

**for**  $i \leftarrow 1$  **to**  $n$  **do**             $s_t^{(i)} \sim P(\cdot | p)$ ;             $c_i \leftarrow E(s_t^{(i)})$ ;         $\bar{c}_t \leftarrow \frac{1}{n} \sum_{i=1}^n c_i$ ;

// Rejection sampling with mean-centering

**for**  $r \leftarrow 1$  **to**  $N_{\max}$  **do**             $s_t^* \sim P(\cdot | p)$ ;             $u^* \leftarrow E(s_t^*) - \bar{c}_t$ ;             $b^* \leftarrow \text{LSH}(u^*)$ ;            **if**  $b^* \in G_t$  **then**                 $s_t \leftarrow s_t^*$ ;                **break**;        **if**  $r = N_{\max}$  **then**            set  $s_t$  to the last candidate observed;

// Concatenate newly generated sentence into prompt

 $p \leftarrow p || s_t$     **return**  $s_{1:T}$ 

// Subroutines

**def** SIG( $s$ ):     $v \leftarrow E(s)$ ;    // embedding of sentence  $s$      $c \leftarrow \text{LSH}(v)$ ;

// LSH signature of the embedding

**return**  $c$ ;**def** LSH( $v$ ):    Initialize  $c \leftarrow []$ ;    **for**  $i \leftarrow 1$  **to**  $d$  **do**         $c \leftarrow c || (n^{(i)} \cdot v > 0)$ ;

// Append digit

**return**  $c$

**Algorithm 2:** SeqMark text detection algorithm

---

**Input** : input prompt  $p$ , text  $T = (s_1, \dots, s_M)$   
**Params**: language model  $P(\cdot)$ , sentence encoder  $E(\cdot)$  with embedding dimension  $h$ ,  
LSH dimension  $d$ , valid region ratio  $\gamma \in (0, 1)$ , secret key  $k$   
**Output**: number of detected watermarked sentences  $D$

**def** DETECT( $T$ ):  
  **Initialization:**  
    Randomly initialize the same  $d$  hyperplane normals  $n^{(1)}, \dots, n^{(d)} \in \mathbb{R}^h$  using  
    key  $k$ ;  
     $D \leftarrow 0$ ;  
    // Number of detected sentences  
  **for**  $t \leftarrow 1$  **to**  $M$  **do**  
    // Re-compute valid and blocked regions  
    Compute the LSH signature of the previously generated sentence,  $\text{SIG}(s_{t-1})$ ,  
    and use  $[\text{SIG}(s_{t-1})]_{10} \cdot p$  as the seed to randomly divide the signature space  
     $\{0, 1\}^d$  into a *valid region set*  $G_t$  of size  $\gamma 2^d$  and a *blocked region set*  $R_t$  of size  
     $(1 - \gamma) 2^d$ .  
    // Re-estimate mean embedding of high-quality manifold  
    **for**  $i \leftarrow 1$  **to**  $n$  **do**  
       $s_t^{(i)} \sim P(\cdot | p)$ ;  
       $c_i \leftarrow E(s_t^{(i)})$ ;  
     $\bar{c}_t \leftarrow \frac{1}{n} \sum_{i=1}^n c_i$ ;  
    // Check the current sentence  
     $v_t \leftarrow E(s_t) - \bar{c}_t$ ;  
     $b_t \leftarrow \text{LSH}(v_t)$ ;  
    **if**  $b_t \in G_t$  **then**  
       $D \leftarrow D + 1$ ;  
    // Concatenate sentence into prompt for next round  
     $p \leftarrow p || s_t$   
  **return**  $D$

---

| Dataset | Task            | Language | #Data | LLM                   | Sentence Encoder    | Text Quality Metric |
|---------|-----------------|----------|-------|-----------------------|---------------------|---------------------|
| WMT19   | Translation     | De-En    | 3000  | ALMA-7B               | LABSE               | COMET               |
| XSum    | Summarization   | En       | 1000  | Llama-2-7B-Instruct   | all-mpnet-base-v1   | ROUGE-L / COMET     |
| MBPP    | Code Generation | En       | 500   | CodeLlama-7B-Instruct | embeddinggemma-300M | pass@k              |
| C4      | Generation      | En       | 100   | Llama-2-7B-Instruct   | all-mpnet-base-v1   | Perplexity          |
| WMT23   | Long-form MT    | De-En    | 100   | Gemma-2-4B-Instruct   | LABSE               | BLEU / COMET        |

Table 13: Datasets, tasks, models, and evaluation metrics used in our experiments.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Translation Prompt</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Translate the following text from German to English:<br>German: München 1856: Vier Karten, die Ihren Blick auf die Stadt verändern<br>English:                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Summarization Prompt</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| For the following article, write a one-sentence summary: “The ex-Reading defender denied fraudulent trading charges relating to the Sodje Sports Foundation - a charity to raise money for Nigerian sport. Mr Sodje, 37, is jointly charged with elder brothers Efe, 44, Bright, 50 and Stephen, 42. Appearing at the Old Bailey earlier, all four denied the offence. The charge relates to offences which allegedly took place between 2008 and 2014. Sam, from Kent, Efe and Bright, of Greater Manchester, and Stephen, from Bexley, are due to stand trial in July. They were all released on bail.<br>Summary: |
| <b>Code Generation Prompt</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Write a python function to remove first and last occurrence of a given character from the string. Directly generate a Python function only. Your code should satisfy these tests:<br>assert remove_Occ(hello,l) == heo<br>assert remove_Occ(abcda,a) == bcd<br>assert remove_Occ(PHP,P) == H<br>Solution:                                                                                                                                                                                                                                                                                                            |

Table 14: Examples of prompt templates used in this work.

|                                                                                                                                                                                                                                                                                                                                              |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Source:</b> München 1856: Vier Karten, die Ihren Blick auf die Stadt verändern<br><b>Target:</b> Munich 1856: Four maps that will change your view of the city<br><b>NS-Watermark:</b> Munich 1856: Four <b>Engravings</b> that Change Your View of the City<br><b>SeqMark :</b> Munich 1856: Four maps that change your view of the city |
| <b>Source:</b> Kleingärtner bewirtschaften den einstigen Grund von Bauern.<br><b>Target:</b> Allotment holders cultivate the soil of former farmers.<br><b>NS-Watermark:</b> Allotment gardeners cultivate the former <b>from</b> of farmers.<br><b>SeqMark :</b> Allotment holders cultivate the former fields of farmers.                  |
| <b>Source:</b> Es nervt, wenn Landkarten nicht aktuell sind.<br><b>Target:</b> It is annoying when geographical maps are not up-to-date.<br><b>NS-Watermark:</b> It’s annoying when maps <b>being</b> out of date.<br><b>SeqMark :</b> It annoys me when maps are not up-to-date.                                                            |

Table 15: Three qualitative examples from WMT19. We include the source sentence, target sentence, NS-Watermark and SeqMark translations. We use underline to denote the watermarked tokens. Interestingly, the watermarked tokens in NS-Watermark are often the odd ones out in the translation (highlighted in **red**). We further note that NS-Watermark only uses beam search for decoding, which is fundamentally different from other approaches (including ours) that utilize token sampling.