Agnostic Watermark Detection: Prompt and Token based Approaches

Aime C. Mugishawayo, Miro Babin, Admire Madyira

1 Abstract

As large language models (LLMs) become ubiquitous in public and private communication, concerns have grown over the provenance and authenticity of AI-generated text. Watermarking is one promising solution, allowing developers to embed subtle signals into generated outputs for downstream attribution or moderation. In this paper, we explore two watermarking methods: prompt-level watermarking using zero-width characters (ZWCs), and token-level watermarking via logit biasing using OpenAI's `gpt-3.5-turbo` API. Importantly, we approach watermark detection from an agnostic standpoint, relying only on output text without access to internal keys, prompts, or model weights. We demonstrate that a fine-tuned RoBERTa classifier can detect both forms of watermarking with reasonable accuracy, outperforming traditional baselines like TF-IDF. Our findings suggest that statistical artifacts from these methods are learnable and detectable, even in restricted-access environments.

2 Introduction

The widespread adoption of large language models (LLMs) in tools like ChatGPT, Claude, and open-source LLMs has introduced new challenges in text attribution, content moderation, and the detection of AI-generated material. As LLMs continue to generate realistic, human-like text, it becomes increasingly important to determine whether a given text was generated by a machine. Watermarking has emerged as a promising method for encoding imperceptible signals in generated text, enabling downstream detection and verification.

Recent work on watermarking generally falls into two categories: token-level and prompt-level. Token-based approaches modify the model's output distribution during generation, often by favoring or suppressing certain tokens. Prompt-based approaches, in contrast, modify the user prompt in a way that influences the output without altering the generation process itself. Despite their potential, most watermarking methods assume access to the watermarking key or the model's internal sampling process. In real-world use cases, however, such access is rarely available.

In this work, we explore whether watermark detection is still possible in these restricted scenarios. We propose two watermarking techniques and evaluate their detectability under a realistic constraint: the detector only sees the generated text and has no access to prompts, model internals, or sampling keys. We construct a labeled dataset using the `gpt-3.5-turbo` API with and without watermarking and evaluate both traditional and transformer-based classifiers. Our results suggest that subtle statistical patterns introduced by both prompt and token-level watermarking can be detected reliably, even in an agnostic detection setting.

3 Related Work

3.1 Token-Based Watermarking

Token-level watermarking modifies the sampling distribution of a language model during generation to embed identifiable patterns in the output text. One foundational method is the greenlist-based approach proposed by Kirchenbauer et al. (2023), which uses a pseudorandom function to bias token selection toward a set of "greenlisted" tokens. This method balances detectability and linguistic naturalness, making it a baseline for many subsequent techniques.

Li et al. (2023) extend this approach by scoring word importance, allowing watermarks to be embedded preferentially in low-importance tokens. This minimizes semantic distortion while preserving watermark strength. Huo et al. (2024) further optimize this trade-off by incorporating a multi-objective framework that adjusts embedding strength on a per-token basis to improve robustness and fluency.

To address the challenge of paraphrasing, Zhu et al. (2024) propose Duwak, a dual watermarking scheme that combines greenlist sampling with an auxiliary probabilistic code to preserve watermark signals under common text transformations. Finally, Dathathri et al. (2024) present SynthID-Text, a production-grade watermarking method based on logit biasing that is designed for real-world deployment scenarios and shows resilience to a wide range of editing operations.

Our work departs from these by focusing on **agnostic detection**: identifying watermarked text

without any prior knowledge of the specific watermarking key, method, or sampling process.

3.2 Prompt-Based Watermarking

Prompt-based watermarking techniques embed identifiable signals into the input prompts of language models, influencing the generated outputs without modifying the model's internal mechanisms. This approach is particularly advantageous in scenarios where access to model internals is restricted, such as with proprietary APIs.

A notable example is the framework proposed by Zhong et al. (2024), which utilizes a multi-model setup comprising a Prompting language model to generate watermarking instructions, a Marking language model to embed watermarks within the generated content, and a Detecting language model to verify the presence of these watermarks. Experiments conducted using ChatGPT and Mistral as the Prompting and Marking language models demonstrated high classification accuracy, with 95% accuracy for ChatGPT and 88.79% for Mistral. These results validate the effectiveness and adaptability of the proposed watermarking strategy across different language model architectures.

Using a similar injection strategy concept, Yao et al. (2023) introduced PromptCARE, a framework designed for prompt copyright protection through watermark injection and verification. PromptCARE addresses the unique challenges of prompt watermarking by proposing injection and verification schemes tailored for prompts and NLP characteristics. Extensive experiments on six benchmark datasets, using three prevalent pre-trained LLMs (BERT, RoBERTa, and Facebook OPT-1.3b), demonstrate the effectiveness, robustness, and stealthiness of PromptCARE.

Another related approach is PostMark, proposed by Chang et al. (2024), which is a modular post-hoc watermarking procedure that inserts an input-dependent set of words determined via semantic embedding into the text after the decoding process. Critically, PostMark does not require access to the model's logits, making it implementable by third parties. Experiments show that PostMark is more robust to paraphrasing attacks than existing watermarking methods.

Our work builds upon these concepts by employing zero-width characters (ZWCs) as imperceptible perturbations within prompts. These ZWCs propagate through the model's generation process, resulting in outputs that carry subtle, machine-detectable artifacts. By training a binary classifier on outputs generated from both original and ZWC-modified prompts, we demonstrate that such prompt-level watermarks can be effectively detected, even in an agnostic setting where the detector has no access to the original prompt or model internals.

Prompt-based watermarking offers a flexible and non-intrusive method for content attribution and provenance tracking, especially in environments where model access is limited. However, its robustness against adversarial transformations, such as paraphrasing or translation, remains an area for further investigation.

4 Data and Methods

4.1 Data

We curated 2000 prompts from the RyokoAI/ShareGPT52K dataset on HuggingFace, which contains 90000 prompts that have been used historically with ChatGPT. This dataset represents a normalized set of prompts that day-to-day users ask ChatGPT through the OpenAI web interface. From the 90000 prompts, we curated a random set of 2000 prompts such that every prompt is between 10 and 150 words, and every character is ASCII. We also avoided prompts that include certain characters to avoid prompts that include math or code.

For each watermarking approach, we used this set of 2000 prompts to construct a labeled dataset consisting of 4,000 LLM-generated responses. We first generated *Vanilla Text* (without watermarking) and then used the same set of 2000 prompts to generate *Watermarked Text (WT)*. To ensure robustness, we randomly shuffled the data and applied a 70–20–10 split into training, validation, and test sets, respectively. Importantly, prompt overlap was controlled to prevent any data leakage across splits.

4.2 Approach A: Prompt-Based Watermarking

Prompt Wrapping

In this approach, we designed a wrapper mechanism that subtly modifies user prompts by inserting invisible or non-standard lexical elements. Specifically, we focused on injecting *zero-width characters (ZWCs)*—Unicode characters that do not produce visible output but are still processed during tokenization. These characters were strategically embedded into prompts in ways that do not alter the meaning or appearance of the text, yet have the potential to leave subtle artifacts in the generated completions. This technique effectively functions as a "hidden signal" embedded in the input that may propagate into the output text. We verified this, confirming that when signaled in such a way, the LLM will generate text with these artifacts.

Text Generation

We used OpenAI's `gpt-3.5-turbo` to generate completions from both the original (unmodified)

prompts and the ZWC-modified prompts. Because ZWCs are not visible, the LLM's output appeared identical to the human eye across both conditions. However, we hypothesized that the injected prompt modifications could subtly influence patterns during generation, thereby leaving behind detectable ZWCs in the model's responses. As these characters are machine-detectable, we were able to write code that confirmed that indeed they were present in the generated outputs.

Example: Zero-Width Character Watermarking

To illustrate how prompt-level watermarking works in practice, consider the following example:

Original Prompt:

`What are the causes of climate change?`

Watermarked Prompt (illustrated with visible marker)

`What are the causes of<U+200B> climate change?`

U+200B (Zero Width Space) is invisible in practice and does not affect how the text looks or behaves.

Sample LLM Response (excerpt with visible marker)

`Climate change is caused by human activities such as burning fossil fuels, deforestation, and global<U+200B> warming...`

In this example, the insertion of a zero-width character does not alter the visual appearance or readability of the prompt. However, when this watermarked prompt is submitted to the LLM, it results in outputs that may also carry invisible artifacts-such as the persistence of zero-width characters or distinct distributional shifts in token selection. Our detection pipeline leverages these imperceptible signals by training a classifier on completions generated from both vanilla and watermarked prompts. The result is a detection model that performs well despite having no access to the prompt or internal model mechanisms.

Watermark Detection

To agnostically detect the presence of watermarks in the generated text, we trained a binary text classifier using a fine-tuned `RoBERTa-base` model. The classifier was trained to distinguish between completions generated from vanilla prompts and those generated from ZWC-modified prompts, relying solely on the response text without access to the original prompt. This framing simulates a realistic detection scenario where only the output text is available for analysis.

4.3 Approach B: Token-Based Watermarking

Unlike prompt-based watermarking, which modifies the input text, our token-level watermarking approach operates by directly influencing the model's token selection probabilities during generation. Because we used OpenAI's `gpt-3.5-turbo` API, we did not have access to the underlying model weights or internal logits. However, the API allows limited control over the output distribution through the `logit_bias` parameter, which adjusts the probability of specific tokens being sampled at each decoding step.

We constructed two lists: a **greenlist** of target tokens that we aimed to promote, and a **redlist** of tokens that we aimed to suppress. These lists consisted of common English tokens selected for their semantic neutrality and frequency across a wide range of topics. During generation, we applied a positive logit bias to the greenlist tokens and a negative logit bias to the redlist tokens. This setup subtly skews the model's sampling behavior toward the greenlist without introducing obvious stylistic artifacts.

While this technique does not embed a cryptographically verifiable watermark as in more advanced methods (Kirchenbauer et al., 2023), it does introduce a detectable statistical signal into the output. Specifically, the greenlist tokens are overrepresented relative to their typical distribution in non-watermarked outputs. This subtle shift forms the basis for detection.

As with prompt-level watermarking, we used the same set of 2000 prompts to generate two sets of completions: *Vanilla Text* (no logit bias) and *Watermarked Text (WT)* (with logit bias applied). This allowed us to train a binary classifier to distinguish between the two without relying on access to the original prompt or API configuration, simulating a realistic agnostic detection scenario.

4.3.1 Example: Logit Bias Watermarking

To demonstrate the subtlety of token-level watermarking using logit biasing, consider the following example:

Prompt:

`What's your favorite color, and why?`

Vanilla Response (GPT-3.5 Turbo, No Watermark):

`"I really like blue because it's calming and reminds me of the ocean. It's a color that feels peaceful and balanced."`

Token-Level Watermarked Response (Logit Bias Toward "purple"):

`"I would say my favorite color is purple. It has a rich, vibrant feel and often symbolizes creativity and depth. There's something unique and elegant about purple that really resonates with me."`

This example highlights how a subtle shift in logit probabilities can influence the generated output. By adjusting the logit bias to favor the token "purple," the model is statistically nudged toward

generating that token, even though it is not explicitly forced. Such biases can be learned by a classifier trained to distinguish between biased and unbiased text, despite their superficial similarity. This token-level manipulation leaves no visible watermark, but introduces measurable distributional signals across outputs.

5 Evaluation and Results

5.1 Baseline Models

To establish performance baselines for both watermarking detection tasks, we employed TF-IDF (Term Frequency–Inverse Document Frequency) feature extraction followed by logistic regression. TF-IDF quantifies the relative importance of terms within documents, offering a lightweight way to detect atypical textual patterns.

For the prompt-level watermarking task, where watermarks are embedded using zero-width characters (ZWCs), we opted for a character-level and byte-pair TF-IDF vectorizer. This was based on the intuition that ZWCs, being non-printable and smaller than typical word units, would be more detectable when the text is analyzed at finer granularity than word-level. Empirically, this choice proved effective, as the model was able to capture the disproportionate frequency of ZWC artifacts, which significantly impact the term distribution in the feature space.

For the token-level watermarking task, where sampling logits were biased toward greenlisted tokens and away from blacklisted ones, we used a standard word-level TF-IDF vectorizer. However, TF-IDF performed poorly in this case, as it treats all terms equally and does not account for token-level dependencies or the probabilistic nature of sampling. This baseline failed to focus on the subtle pattern shifts caused by logit manipulation—specifically, the increased likelihood of certain tokens appearing, not the raw frequency of individual words.

5.2 Fine-Tuning RoBERTa

We hypothesized that a pretrained transformer model would outperform TF-IDF-based baselines, as transformers can capture long-range dependencies, contextual subtleties, and token-level distributional shifts.

For both watermarking tasks, we fine-tuned a RoBERTa-base model with a classification head. The number of training epochs was limited to the number where validation accuracy plateaued. RoBERTa was chosen for several reasons:

- It is based on the BERT architecture but trained with dynamic masking and on larger corpora, making it well-suited for subtle distributional tasks.
- It retains BERT's deep bidirectional encoding, which helps model token interactions at varying levels of abstraction.
- It has been shown to outperform BERT on several NLP benchmarks, including classification tasks.

5.3 Model Performance

TF-IDF regression models.

The TF-IDF baselines achieved 65% on the prompt-level watermarking. The relatively strong performance in the ZWC setup is explained by the TF-IDF vectorizer's sensitivity to term frequency anomalies: zero-width characters, though invisible, significantly alter the underlying token stream. When encoded as individual characters or byte pairs, they skew the term distributions enough to be captured by even a linear model.

Metric	Value
Train Accuracy	90%
Validation Accuracy	67%
Test Accuracy	65%

Table 1: Performance of TF-IDF + Logistic Regression on Prompt-Level (ZWC) Watermarking

By contrast, TF-IDF performed poorly in the token-level watermarking task, achieving a meager 29%. This is because the model attempts to treat all tokens as equally informative, when in fact only a specific subset—those from a predefined greenlist or blacklist—are subtly over- or underrepresented. Since the watermarking signal in this setup manifests as a shift in token sampling probability rather than presence or absence, TF-IDF lacks the representational nuance to detect it.

Metric	Value
Train Accuracy	90%
Validation Accuracy	26%
Test Accuracy	29%

Table 2: Performance of TF-IDF + Logistic Regression on Token-Level (Logit Bias) Watermarking

RoERTa models. In both settings, the fine-tuned RoBERTa model achieved markedly stronger performance. The model reached 85% accuracy in the ZWC task, significantly outperforming the TF-IDF baseline. As our intuition suggested, BERT, being a pre-trained transformer model, captures contextual and semantic nuances, including subtle patterns like ZWCs.

For the token-level watermarking, it improved test accuracy to 65%, highlighting its ability to detect even subtle distributional biases. This improvement stems from the model's ability to learn

Metric	Value
Train Accuracy	95%
Validation Accuracy	85%
Test Accuracy	86%

Table 3: Performance of BERT (fine-tuned) on Prompt-Level (ZWC) Watermarking

complex token co-occurrence patterns and internalize how certain tokens appear more frequently under biased sampling distributions.

Metric	Value
Train Accuracy	64%
Validation Accuracy	65%
Test Accuracy	65%

Table 4: Performance of BERT (fine-tuned) on Token-Level (Logit Bias) Watermarking

6 Ethical Considerations

As watermarking techniques become increasingly central to managing AI-generated content, it is essential to reflect on their ethical implications. While watermarking can help ensure transparency, attribution, and responsible deployment of language models, it also raises concerns about surveillance, misuse, and consent. For instance, embedding watermarks in generated text without user awareness, especially in outputs that may be shared publicly, risks infringing on user autonomy and potentially violating principles of informed consent. If users are not made explicitly aware that their inputs or outputs may be watermarked, the practice could be considered deceptive, especially in sensitive contexts like journalism, education, or legal documentation.

Moreover, watermark detection systems-particularly those used by third parties-may suffer from false positives or negatives. In cases where watermarked content is falsely attributed to an LLM, a human author could face reputational harm or wrongful censorship. This issue is especially pressing in academic, journalistic, or governmental settings, where authorship and provenance are critical. On the flip side, the failure to detect actual AI-generated content due to evasion or removal of watermarks could mislead consumers and allow misinformation or AI-generated spam to spread unchecked.

The privacy implications are equally important. Agnostic watermark detection does not require access to underlying prompts or model internals, which makes it appealing from a deployment perspective. However, this also means that detection models could be used in surveillance regimes to classify user-generated text as machine-generated without explicit user consent. In such scenarios, users may be unfairly profiled or flagged by automated systems based on opaque and potentially flawed criteria.

7 Limitations

Our detection method, while promising, is not infallible. The token-level watermark detection task is particularly susceptible to false positives, as natural variation in language can mimic the statistical patterns introduced by logit biasing. Contextual factors such as topic, writing style, and even author intent can produce token distributions similar to watermarked text. Conversely, prompt-level methods such as zero-width characters (ZWCs) offer more reliable detection but may fail under preprocessing steps like character normalization, text cleaning, or lossy transformations applied in content pipelines. Additionally, detection performance may degrade when texts are paraphrased, translated, or mixed with non-watermarked content.

Another limitation arises from our reliance on OpenAI's `gpt-3.5-turbo` API for data generation. While the API exposes a `logit_bias` parameter, it offers limited granularity and no visibility into internal sampling mechanisms. This restricts the fidelity and control we have in crafting token-level watermarks, possibly underrepresenting what is achievable with full model access. Furthermore, our use of static greenlists and redlists does not capture adaptive watermarking strategies that could dynamically respond to topic, prompt structure, or adversarial attacks.

8 Future Directions

We identify several promising directions for extending this work. First, future research should systematically evaluate how prompt- and token-level watermarks hold up under transformation. Robustness to paraphrasing, sentence reordering, insertion or deletion of content, and even machine translation remains an open challenge. While our methods achieved reasonable performance in clean conditions, real-world deployment must contend with noisy or intentionally jumbled text. Building a benchmark suite that systematically applies such transformations would be a valuable contribution.

Second, generalization across models and domains should be examined more deeply. Our classifiers were trained and tested exclusively on completions from `gpt-3.5-turbo`, a single commercial API. Future work should explore whether detection models trained on one LLM generalize to outputs from other models, such as GPT-4, Claude, or open-source alternatives like LLaMA, Mistral, or Mixtral. Likewise, domain-specific evaluations-e.g., scientific writing, customer service chats, or

social media posts-can shed light on the broader applicability of these detection systems.

Third, interpretability remains a limitation of our current detection pipeline. While RoBERTa-based classifiers achieved strong results, they behave as black boxes. Understanding which textual features or token patterns contribute most to classification decisions is essential for both trust and transparency. Techniques like integrated gradients, attention visualization, or SHAP (SHapley Additive exPlanations) could be employed to make classifier behavior more explainable. This interpretability could also help improve detection performance by informing better feature engineering or model architecture design.

Fourth, the interplay between watermark strength and naturalness deserves deeper exploration. Strong watermarks may be easier to detect but could risk degrading output fluency or revealing their presence to adversaries. We envision a future research work dedicated to quantifying and optimizing this trade-off, possibly using human evaluators alongside automatic metrics.

Finally, any technical advances in watermarking must be accompanied by ethical safeguards. Clear disclosure mechanisms, opt-in policies, and redress protocols are crucial for preserving user trust. Future work should engage with stakeholders-including ethicists, policymakers, and affected user communities-to co-develop guidelines and regulations governing the deployment of watermarking and detection technologies. Without these protections, even technically sound systems may cause more harm than good.

9 Conclusion

We introduced two watermarking techniques, one based on prompt modification via zero-width characters and the other based on token-level logit biasing using OpenAI's API, and evaluated their detectability using fine-tuned classifiers. Our experiments demonstrate that even agnostic classifiers, trained without access to internal model parameters or keys, can detect these watermarks with reasonable accuracy. However, token-level watermark detection remains more ambiguous due to natural linguistic variability.

By framing watermark detection as a classification task over outputs alone, we offer a practical direction for real-world monitoring of AI-generated text. Our results suggest that subtle statistical patterns introduced by both prompt- and token-level watermarking techniques can be learned and detected, even under adversarial constraints.

We believe this work lays foundational groundwork for developing watermarking solutions that respect user privacy, maintain content quality, and scale in settings where model internals are unavailable. As AI-generated content continues to proliferate, such tools will become critical for maintaining trust, provenance, and accountability in the digital information ecosystem.

References

Yapei Chang, Kalpesh Krishna, Amir Houmansadr, John Wieting, and Mohit Iyyer. 2024. Postmark: A robust blackbox watermark for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024)*.

Sumanth Dathathri, Minjia Zhang, Luca Sellitto, Fahim Kawsar, Mahmoud Koohi, Rohit Girdhar, Hamed R. Hassani, Nicolas Papernot, Aleksander Madry, Krishnaram Kenthapadi, Syed Kamran, Zhiting Hu, Gal Kaplun, Yusuke Mukuta, Noah Fiedel, Katherine Lee, Abhishek Raghunathan, Xueyan Jiang, Borja Balle, and 10 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634:818–823.

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

Johannes Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. *Preprint*, arXiv:2301.10226.

Yuhang Li, Yihan Wang, Zhouxing Shi, and Cho-Jui Hsieh. 2023. Improving the generation quality of watermarked large language models via word importance scoring. *Preprint*, arXiv:2311.09668.

Hongwei Yao, Jian Lou, Kui Ren, and Zhan Qin. 2023. Promptcare: Prompt copyright protection by watermark injection and verification. *Preprint*, arXiv:2308.02816.

Xin Zhong, Agnibh Dasgupta, and Abdullah Tanvir. 2024. Watermarking language models through language models. *Preprint*, arXiv:2411.05091.

Chaoyi Zhu, Jeroen Galjaard, Pin-Yu Chen, and Lydia Y. Chen. 2024. Duwak: Dual watermarks in large language models. *Preprint*, arXiv:2403.13000.