# Safeguarding Scientific Peer Review through Watermarking of LLM-Generated Text

Anonymous ACL submission

## Abstract

The integrity of peer review is essential to uphold scientific rigor and trust in academic publishing. Traditionally dependent on domain experts' critical judgment, the process now faces growing strain due to surging submission volumes and limited reviewer availability. This has led to the rise of *lazy reviewing*, where reviewers rely on large language models (LLMs) to generate reviews—raising concerns over fairness, accountability, and the authenticity of evaluations. Prior efforts have focused on detecting AI-generated content or estimating its prevalence in reviews, yet existing detectors remain vulnerable to adversarial attacks and often require domain-specific retraining. To address these limitations, we introduce a novel peer review watermarking framework that embeds traceable, query-aware signals into generated reviews without compromising scientific coherence. Our approach integrates a *Query-Aware Response Generation* module with a watermark detection mechanism that enables editors to verify review authenticity reliably. Comprehensive experiments on ICLR and NeurIPS datasets demonstrate that our framework surpasses existing AI-text detectors under diverse adversarial conditions. We hope this work advances the responsible integration of LLMs into scholarly communication. We make our code and datasets publicly available[1].

## 1 Introduction

.

The emergence of frontier large language models (LLMs), such as Claude, Gemini, GPT-4 (Achiam et al., 2023), LLaMa, etc. has revolutionized natural language generation. The sophisticated human-like fluency and coherence exhibited by texts produced by these models present considerable challenges in discerning whether such content is human-generated or machine-generated, even for domain experts (Shahid et al., 2022). Peer review remains a foundational practice in academia, serving as a critical mechanism through which expert scrutiny ensures the integrity and credibility of scholarly outputs prior to publication (Alberts et al., 2008). Nevertheless, the escalating volume of manuscript submissions (Bornmann and Mutz, 2015; McCook, 2006) has increasingly burdened the peer review system, amplifying concerns regarding the system's sustainability and efficacy (Arns, 2014).

Scientific peer review fundamentally depends on expert reviewers to provide insightful, critical, and constructive evaluations of submitted manuscripts or proposals (Shah, 2022). According to the Association for Computational Linguistics (ACL) policy[2], Artificial Intelligence (AI) tools may assist with paraphrasing and proofreading tasks, particularly benefiting non-native English speakers; however, reviewers are required to independently generate substantive review content. Recent research (Liang et al., 2024) examining peer reviews from AI-related conferences identified that approximately 6.5% to 16.9% of review text may have been substantially modified using LLMs. The study highlights a significant increase in LLM usage, particularly ChatGPT, immediately preceding review deadlines, with higher reliance detected among reviewers not engaging in author rebuttals at prominent venues such as ICLR and NeurIPS. Additionally, increased ChatGPT usage was associated with diminished self-reported reviewer confidence. A relevant research (Ye et al., 2024) demonstrated that manipulating just 5% of reviews could disrupt rankings sufficiently to displace approximately 12% of papers from the top 30%. Further, this study revealed intrinsic limitations of LLM-based reviews, including potential biases such as favoring incomplete submissions over fully developed

[1] https://anonymous.4open.science/r/PeerWatermarking-51DD/

[2] https://2023.aclweb.org/blog/review-acl23/#faq-can-i-use-ai-writing-assistants-to-write-my-review

manuscripts and preferentially rating submissions by prominent authors in single-blind review scenarios. Moreover, authors can intentionally embed covert content within manuscripts to deliberately manipulate LLM-generated reviews, leading to artificially inflated assessments misaligned with human evaluations. These findings collectively suggest that current LLMs are insufficiently robust for deployment as primary reviewers due to inherent vulnerabilities and susceptibility to manipulation. Consequently, rigorous safeguards and enhanced evaluation frameworks must be implemented to ensure review fairness and accuracy before broader adoption of LLM-based review processes.

In this paper, we propose a novel framework for watermarking LLM-generated peer reviews. Our approach consists of several key components. First, we introduce a Query-Aware Response Generation module, which selectively applies watermarking when user uploads a research paper and there is a risk of peer review misuse. Then, our Watermark Injection Mechanism embeds subtle yet detectable signals in peer reviews while preserving scientific terminology. Finally, we implement Watermark Detection, which allows editors and conference chairs to verify the authenticity of peer reviews. Across ICLR and NeurIPS data, our watermarking framework achieves significantly higher detection accuracy than existing methods, maintaining performance even in adversarial settings. Our work aims to safeguard the peer review process against misuse of generative language models, thereby reinforcing ethical norms in scholarly communication and contributing to a trustworthy research ecosystem.

Our contributions are summarized as follows: 1) We introduce the novel task of watermarking AI-generated peer reviews to ensure authenticity and traceability. 2) We propose a new lightweight framework that (i) employs a gating mechanism to watermark only potentially unsafe peer-review generation requests and (ii) introduces a simple yet effective watermarking strategy that markedly improves the detection of machine-generated peer reviews. 3) We propose a simple yet effective watermarking technique, which improves WLLM (Kirchenbauer et al., 2023) in identifying machine-generated peer reviews. 4) Our watermarking technique outperforms existing AI-based text detectors, even under adversarial conditions.

## 2 Related Work

Zero-shot text detection identifies AI-generated text without requiring training on specific data (Mitchell et al., 2023). Solaiman et al. (2019) detect AI-generated text by measuring its average log probability under the generative model. DetectGPT (Mitchell et al., 2023) leverages the tendency of AI-generated text to reside in negative curvature regions of the model's log probability function for detection. Fast-DetectGPT (Bao et al., 2023a) enhances efficiency by applying conditional probability curvature instead of raw probability. Guo et al. (2023) developed the OpenAI text classifier by training it on a large dataset comprising millions of texts. However, heavy dependence on training data makes many of these models susceptible to adversarial attacks (Wolff, 2020).

Watermarking AI-generated text, introduced by Wiggers (2022), embeds an imperceptible pattern to verify authorship, similar to encryption. Watermarks can be embedded without requiring modifications to the underlying language model, allowing standard models to generate watermarked text (Kirchenbauer et al., 2023). Rather than focusing on individual detection, Liang et al. (2024) proposed a method that estimates the proportion of AI-generated text within a large corpus using maximum likelihood estimation of probability distributions.

As far as we know, this is the first work to address AI-generated peer review detection through watermarking. Unlike existing AI text detectors, which are vulnerable to adversarial attacks, our approach embeds traceable markers directly into generated content, improving the detection of AI-generated reviews. Additionally, current AI text detection models require task-specific training for each conference and dataset, making large-scale deployment challenging. In contrast, our watermarking method provides a scalable solution that eliminates the need for continuous retraining, enhancing both reliability and adaptability across diverse academic settings.

## 3 Methodology

Figure 1 illustrates the framework, which consists of two key components: (a) Query-Aware Response Generation, where a user uploads or submits a research paper along with a query related to it, which is classified by the Query Type Identifier. If identified as an unsafe query (indicating poten-

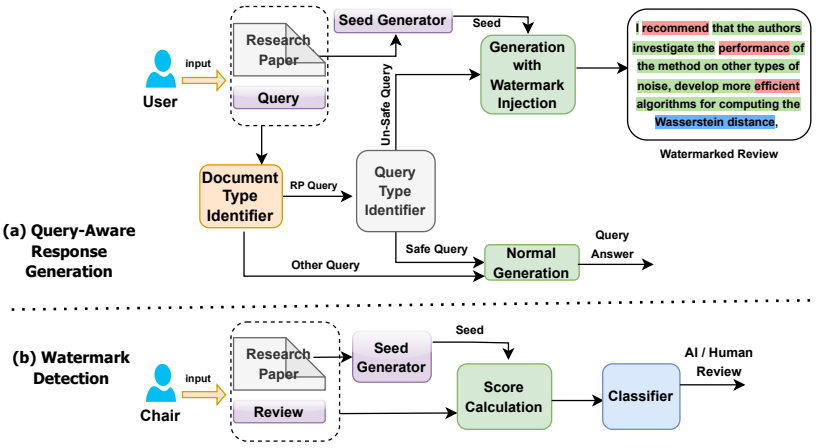


Figure 1: Overview of the Proposed Watermarking Framework. (a) Watermark Generation (b) Watermark Detection; Here red indicates red token, green indicates green token, blue indicates blue token

tial peer review misuse), it is processed through the LLM generator with watermarking. Otherwise, queries proceed through normal Generation. (b) Watermark Detection, where an editor or chair submits a research paper and its corresponding review for verification.

**Algorithm 1:** Watermark Injection

**Require:** Vocabulary $V$, Paper $P$ , Watermark Strength $\delta$, Green List Fraction $\gamma$

1: Compute seed $S_T$ using paper information and secret keys $K_{secret}$ and $p$.
2: Generate green list $G$ and red list $R$ from $V$ based on $\gamma$
3: Extract blue list $B$ as domain-specific terms from $P$
4: **for** each generation step $t = 1, \ldots, T$ **do**
5: Obtain logits $l_w^{(t)}$ from LLM
6: Adjust logits: $l_w^{(t)} = l_w^{(t)} + \delta \cdot \mathbb{1}[w \in G \cup B]$
7: Normalize adjusted logits via softmax:

$$p_w^{(t)} = \frac{e^{l_w^{(t)}}}{\sum_{w' \in V} e^{l_{w'}^{(t)}}}$$

8: Sample next token $w^{(t)}$ from adjusted distribution
9: **end for**

### 3.1 Document Type Identifier

To determine whether a document is a research paper, we implemented a simple rule-based approach. Documents with fewer than 500 words were filtered out as unlikely to be research papers. We detected key section headers such as Abstract, Introduction, Methodology, Results, Conclusion, and References using regular expressions and header position analysis. A document was classified as a research paper if at least three core sections were present, along with a reference section and in-text citations (e.g., (Author, Year) or [1]). This method provided a lightweight and efficient first-pass classification before passing the query to the Query Type Identifier, which then determines whether the query is safe or unsafe.

### 3.2 Query Type Identifier

The Query Type Identifier determines whether a query is classified as Safe (S) or Unsafe (UN). A query is considered Unsafe (UN) if it requests a peer review in a way that allows the reviewer to directly submit the generated content as a peer review. Any query that does not fall into the Unsafe category is classified as Safe (S), including those that seek explanations, summaries, or clarifications related to the paper's content. We employ a few-shot prompting approach to classify user queries. We discuss this in detail in section F 'Query Type Identifier' of Appendix.

Specifically, watermarking every LLM output introduces unnecessary computational overhead and, at higher watermarking strengths can increase perplexity and reduce fluency (discussed below in Section :*Effect of Watermarking Strength on Perplexity*). The Query-Type Identifier serves as a lightweight, optional gating mechanism that reliably distinguishes between safe and potentially unsafe queries, ensuring watermarking is only applied when the risk of misuse such as peer-review submission is present. In deployment, users may choose to bypass this module and apply watermarking uniformly across all queries if required.

### 3.3 Watermark Injection

In this section, we introduce our watermarking injection technique, which ensures that LLM-generated text is subtly embedded with verifiable signals without significantly altering fluency or coherence. In this work, we utilize the soft watermarking technique that introduces probabilistic biases in token selection to subtly embed watermarks in text generated by large language models (LLMs). Our approach is inspired by prior watermarking method (Kirchenbauer et al., 2023). However, this approach face challenges in peer review text generation due to the inherent trade-off between watermark strength and text quality. The random selection of green and red tokens introduces high variability, weakening the watermark signal. Our novel approach incorporates 'blue tokens'—key technical terms from the research paper itself—enhancing watermark robustness by grounding signals in semantically meaningful content. Algorithm 1 outlines the complete watermark injection process.

#### 3.3.1 Paper Seed Generation

Our seed generation mechanism ensures unique, deterministic, and secure watermarking by leveraging context-aware encoding, cryptographic hashing, and a secret key. The input text $T$, which can be a paper title, abstract, or any small portion of text from any section of the paper. To strengthen security, a secret key $K_{\text{secret}}$ is concatenated with $T$, ensuring that different users generate distinct seeds:

$$I = T \| K_{\text{secret}} \tag{1}$$

$K_{\text{secret}}$ is a fixed, confidential key (text) known only to trusted parties (e.g., conference chairs or editors). It is not generated at runtime. Instead, it is concatenated with the paper text to produce a deterministic seed for watermarking. This ensures that watermarking is reproducible and verifiable only by those with access to the key. The encoded representation is then hashed using SHA-256 for collision resistance:

$$H(I) = \text{SHA-256}(E(I)) \tag{2}$$

Here, $E(I)$ denotes the full encoding of the input text $I$, computed by applying the character-level encoding function $f(c, n)$ to each character $c$ in $I$. Each character is mapped to its alphabetical index and applying a shift cipher based on the text length:

$$f(c, n) = ((\text{ord}(c) - \text{ord}('A') + n) \mod 26) + 1 \tag{3}$$

where $c$ is the character and $n$ is the total number of characters in $T$. This ensures that encoding remains text-dependent, enhancing uniqueness. Finally, the hash is mapped to a bounded numeric space using modular reduction:

$$S_T = H(I) \mod pk \tag{4}$$

Even if an attacker identifies the specific paper text used for watermarking, they would still require two secret keys, $K_{\text{secret}}$ and $pk$, to decode the green list and verify the watermark. These keys ensure that only authorized individuals, such as the editor or program chair, can perform detection. To maintain the integrity and security of the watermarking system, $K_{\text{secret}}$ and $pk$ keys must be kept strictly confidential and accessible only to authorized personnel.

#### 3.3.2 Green-Red Token Partitioning

Given a vocabulary set $V$, we define a subset of tokens, $G$, termed as the "green list" which are favored during text generation. The remaining tokens form the "red list" $R$. Instead of a probability-based split, we use a deterministic *random permutation* seeded by the paper seed generator. A fraction $\gamma$ of tokens is selected as green, ensuring consistency across runs. Let $|V|$ denote the vocabulary size :-

$$|G| = \gamma|V|, \quad |R| = (1-\gamma)|V|. \tag{5}$$

#### 3.3.3 Blue Token Selection

We define a subset of tokens, denoted as the *blue list* $B$, which consists of important technical terms extracted from a given research paper. These blue tokens represent domain-specific terminology that is crucial for maintaining the technical accuracy and coherence of the generated text.

Unlike the green list $G$, which is deterministically selected based on a fixed fraction $\gamma$ of the vocabulary, the blue list is explicitly derived from the research content, ensuring a stronger alignment with the subject matter of the paper. To construct $B$, we utilize a language model (LLM) to extract key technical terms from the paper by prompting it to identify domain-relevant terminology. We found that average number of extracted terms per paper is approximately 43.83. Due to space constraint, we discuss this in detail in section C 'Detail about Blue Token Selection' of the Appendix.

#### 3.3.4 Logit Adjustment Mechanism

During inference, given a token sequence $w_0, w_1, \ldots, w_{t-1}$, the language model produces

a logit vector $l_w^{(t)}$ representing the probability distribution over $V$. We modify these logits using a biasing function:

$$l_w^{(t)} = l_w^{(t)} + \delta \cdot \mathbb{1}[w \in G \cup B], \tag{6}$$

where $\delta$ is a tunable parameter controlling the watermarking strength, and $\mathbb{1}[w \in G \cup B]$ is an indicator function returning 1 if $w$ is in the green list $G$ or blue list $B$, and 0 otherwise. The resulting logits are then passed through the softmax function to obtain the final token probabilities:

$$p_w^{(t)} = \frac{e^{l_w^{(t)}}}{\sum_{w' \in V} e^{l_{w'}^{(t)}}}. \tag{7}$$

This ensures that tokens in $G$ and $B$ are more likely to be sampled while discouraging tokens from $R$, thereby reinforcing both the structured watermarking and the preservation of domain-specific terminology.

### 3.4 Watermark Detection

Given a research paper, the proposed algorithm generates a deterministic seed to ensure consistency between encoding and detection. Since each research paper is unique, the generated seed remains identical to that used during watermark insertion. As a result, the same random token list (formerly the green list) is reconstructed. Similarly, the blue token list, consisting of technical terms extracted from the paper, is also reproduced, as these terms remain unchanged. Consequently, the marked token set, i.e., the union of random and blue tokens, remains identical, enabling accurate watermark detection. Due to space limitation, we discuss the algorithm in detail in Algorithm 2.

We extract bigrams (k=2) as part of our detection pipeline, building on the k-gram watermarking framework introduced by (Kirchenbauer et al., 2023) and extended theoretically by (Zhao et al., 2023). These works justify using k-grams, where k can be tuned based on task requirements. Bigrams offer a practical balance between local contextual awareness and statistical reliability. Specifically, they reduce token-level noise (e.g., repetition) and improve robustness to paraphrasing.

We computed $z$ score which is a standard test statistic (Zhao et al., 2023; Kirchenbauer et al., 2023), used to distinguish between watermarked and non-watermarked text, with theoretical guarantees on false positive and false negative rates (see (Zhao et al., 2023), Theorems 3.3–3.5). Under this model, the expected count of green tokens is $\gamma T$, with variance $T \cdot \gamma(1-\gamma)$, yielding the normalized score:

$$z = \frac{|G| - \gamma T}{\sqrt{T \cdot \gamma(1-\gamma)}} \tag{8}$$

Here $y$ indicates whether the review is watermarked.

**Algorithm 2:** Watermark Detection

**Require:** Peer Review Text $R$, Paper Tokens $P$
**Ensure:** Marked Token Fraction $f_m$, Z-Score $z$

1: Tokenize the peer review $R$ using tokenizer $\mathcal{T}$
2: Generate a deterministic seed $S_T$ from the paper using the seed generator
3: Partition vocabulary $V$ into random tokens $G$ and red tokens $R_{\text{red}}$ using $S_T$
4: Extract blue tokens from the paper: $O = P \cap R_{\text{red}}$
5: Compute marked tokens: $M = G \cup O$
6: Extract bigrams $B$ from $R$ and initialize marked token hit count $M_c = 0$
7: **for** each bigram $(x, y) \in B$ **do**
8: Increment $M_c$ if $y \in M$
9: **end for**
10: Compute marked token fraction:

$$f_m = \frac{M_c}{|B|}$$

11: Compute expected marked token fraction:

$$\mathbb{E}[f_m] = \frac{|M|}{|V|}$$

12: Compute z-score:

$$z = \frac{M_c - |B|\mathbb{E}[f_m]}{\sqrt{|B|\mathbb{E}[f_m](1 - \mathbb{E}[f_m])}}$$

13: **return** $f_m, z$

### 3.5 Main Result

We evaluate our method against multiple AI-generated text detectors, including RADAR (Hu et al., 2023), DEEP-FAKE (Li et al., 2023) and Fast-Detect GPT (Bao et al., 2023b). Additionally, we evaluated against specialized AI-generated text detectors for peer review, such as TF-Model (which leverages term frequency of AI-generated tokens) and RR-Model (a regeneration-based method) (Kumar et al., 2024). For *Watermarking based* methods, we have included two baselines: WLLM (Kirchenbauer et al., 2023) and SynthID Text (Dathathri et al., 2024).

We used both AI-generated reviews and human reviews for this experiment. During the attack phase, we targeted only the AI-generated reviews, as they are the ones intended to evade detection. As

| Model | | w/o | Low P | High P | Token |
|---|---|---|---|---|---|
| *Baseline Models* | Radar | 48.02 | 16.16 | 4.24 | 14.14 |
| | LLM-Det | 34.24 | 33.38 | 32.72 | 19.30 |
| | Fast Detect | 60.36 | 13.09 | 3.44 | 43.24 |
| | Deep Fake | 66.00 | 57.03 | 35.44 | 63.78 |
| | TF-Model | 88.06 | 68.58 | 66.10 | 18.70 |
| | RR-Model | 78.38 | 63.51 | 61.60 | 64.12 |
| | SynthID Text | 83.65 | 77.27 | 70.21 | 72.34 |
| | WLLM | 81.87 | 78.50 | 71.40 | 73.20 |
| | RDW | 77.30 | 74.60 | 72.11 | 71.20 |
| | SEMSTAMP | 76.70 | 74.10 | 75.38 | 72.90 |
| **Our Model** | $\delta$= 3.0 | 91.45 | 85.29 | 76.42 | 77.81 |
| | $\delta$= 4.0 | 95.20 | 88.14 | 79.56 | 80.32 |
| | $\delta$= 5.0 | **98.31** | **92.79** | **84.36** | **83.87** |

Table 1: F1 Score Performance Comparison Under Different Attack Scenarios (values in %). Here P $\rightarrow$ Paraphrasing; Token $\rightarrow$ Token attack; w/o $\rightarrow$ without any attack. $\gamma$ : 0.3

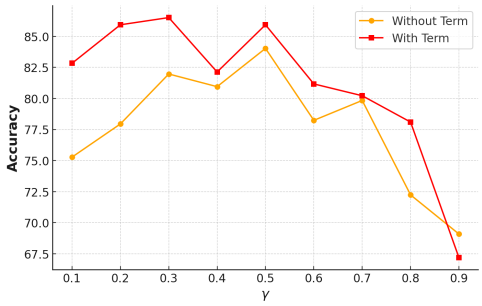


Figure 2: Effect of Watermarking on varying $\gamma$ on with only green token (without Term) and with green token and blue token (With Term); $\delta$= 2.0.

shown in the Table 1, existing AI detectors exhibit extreme sensitivity to adversarial attacks, with Fast Detect suffering a 94.30% drop (60.36% $\rightarrow$ 3.44%) and Radar declining by 91.17% (48.02% $\rightarrow$ 4.24%) under high paraphrasing. Similarly, TF-Model's F1 score decreases by 78.76% (88.06% $\rightarrow$ 18.70%) under token attack, highlighting the brittleness of non-watermarked approaches. In contrast, our proposed watermarking method retains a performance of 84.36% under high paraphrasing and 83.87% under token attack ($\delta$=5.0), outperforming all baselines by a substantial margin. Even with lower $\delta$ values, our model demonstrates resilience, with $\delta$=3.0 yielding 76.42% and $\delta$=4.0 yielding 79.56% under high paraphrasing, indicating consistent adversarial robustness. These findings emphasize that existing AI text detectors alone are insufficient for detecting AI-generated text under adversarial conditions. Our watermarking approach provides a promising solution for improving the resilience of AI generated peer review detection, even in challenging settings.

We also compare our approach with WLLM (Kirchenbauer et al., 2023), which relies solely on randomly selected green tokens for watermarking. In contrast, our method incorporates domain-specific tokens in addition to green tokens. Our results demonstrate that integrating domain-specific tokens significantly enhances watermark detectability, highlighting the importance of semantically meaningful token selection. We discussed this in detail in Section 3.7. Additionally, we compared our method with SynthID Text by injecting watermarks using its speculative sampling technique and detecting them based on the weighted mean detection score (ranging from 0 to 1), optimized on a validation set. As shown in Table 1, SynthID achieves F1 scores of 83.65 (no attack), 77.27 (low-perplexity paraphrasing), 70.21 (high-perplexity paraphrasing), and 72.34 (token substitution). Our method, evaluated under multiple threshold settings ($\delta$), consistently obtains higher F1 scores across all conditions, demonstrating strong robustness and reliability under adversarial peer review scenarios. Additionally Our method outperforms RDW (Kuditipudi et al., 2023) in terms of detection accuracy by 4.6 points, achieving 81.9% (comparable to KGW), and outperforms SEMSTAMP (Hou et al., 2023) by 4.9 points, achieving 81.6% (slightly lower than KGW). We also evaluated both methods for robustness against paraphrasing. Under high-paraphrasing conditions, SEMSTAMP's performance drops to 75.38 % (approximately 1% lower than our method), while RDW drops to 72.11% (4.31 points lower than our method).

Additionally, we found that the Query Type Identifier achieves an accuracy of 95.5% on the test set. Further, we studied the effect of varying the underlying base language model on detection accuracy. We show in figure-2 in Appendix that our approach maintains robust performance across LLMs of different sizes and architectures. Due to space restriction we have added further discussion of this in Section F 'Query Type Identifier' of the Appendix.

### 3.6 Effect of varying $\gamma$ on Detection Accuracy

Figure 2 shows that at low green token fractions ($\gamma = 0.1$ to $\gamma = 0.3$), detectability remains weak due to an insufficient statistical signal. When too few green tokens are available, the sampling algorithm operates largely unconstrained, following the model's natural probability distribution with minimal watermarking influence. As a result, the watermark imprint is inconsistent, leading to higher

variance in detection scores. However, at $\gamma = 0.3$, detectability peaks, indicating an optimal balance where the watermarking method biases the sampling process enough to be recognized while still allowing diverse token choices. Beyond $\gamma = 0.3$, an interesting shift occurs. As $\gamma$ increases, the green token fraction introduces greater randomness into the sampling process, allowing the model more flexibility in token selection. At $\gamma = 0.4$, this increased entropy makes the watermark signal less distinct, leading to a temporary decline in detectability. Interestingly, at $\gamma = 0.5$, detectability recovers, possibly due to an optimal trade-off which watermarking constraints are still strong enough for recognition while allowing sufficient linguistic variation to stabilize detection. Beyond this point, performance declines again as higher green token fractions ($\gamma > 0.6$) further increase randomness, making the text appear more natural and reducing watermark signal strength. At very high $\gamma$ values (e.g., $\gamma = 0.9$), nearly all tokens in the sampling space are green, making the sampling distribution indistinguishable from unwatermarked text, effectively neutralizing detectability.

### 3.7 Effect of Domain-Specific Token Selection on Watermark Detectability

The results in Figure 2 demonstrate that incorporating important domain-specific tokens (blue tokens) alongside green tokens significantly improves watermark detectability across all thresholds compared to using only green tokens. This also highlights that our approach improves upon WLLM. The improvement is particularly notable at lower thresholds ($\gamma = 0.1$ to $\gamma = 0.3$), with performance gains exceeding 10% at $\gamma = 0.1$ and $\gamma = 0.2$. This suggests that while random green token selection introduces high variability, leading to a weaker watermark signal, integrating important technical terms from the research paper enhances detection robustness by grounding the watermark in semantically meaningful and contextually significant words. Interestingly, at higher thresholds ($\gamma > 0.6$), the performance difference reduces, likely due to the increased randomness in token selection making the watermark less distinguishable. These findings underscore the effectiveness of domain-aware token selection in improving watermark detectability while maintaining text fluency.

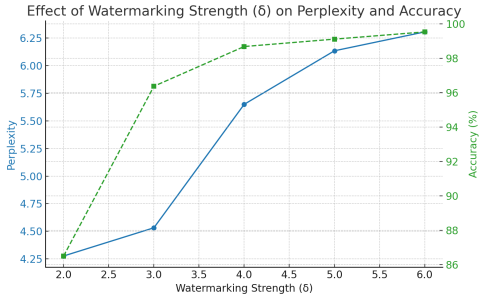


Figure 3: Effect Of Watermarking Strength ($\delta$) on Perplexity and Accuracy; $\delta = 2.0$. The blue line indicates Perplexity, and the green line indicates Accuracy.

### 3.8 Effect of Watermarking Strength ($\delta$) on Detection Accuracy

The graph demonstrates a positive correlation between watermarking strength ($\delta$) and detection accuracy. As $\delta$ increases from 2 to 6, the accuracy of watermark detection improves from 86.51% to 99.54%. This trend shows that increasing the watermarking bias enhances the distinguishability of AI-generated text. The primary reason for this improvement is that a higher $\delta$ more strongly biases the model's token selection toward a predefined set of "green list" and "blue list" tokens, making it easier to detect the watermark statistically. This controlled alteration in token probabilities increases the reliability of detection algorithms, as deviations from a natural distribution become more pronounced.

### 3.9 Effect of Watermarking Strength ($\delta$) on Perplexity

Perplexity is a fundamental metric used to evaluate the confidence of a language model in its predictions. Lower perplexity values indicate that the model assigns higher probabilities to its predicted tokens, signifying more fluent and coherent text generation. Conversely, higher perplexity suggests greater uncertainty, implying that the text deviates from the model's natural distribution. In watermarking studies, minimizing the impact on perplexity is crucial to ensure that the watermarked text remains natural and human-like (Kirchenbauer et al., 2023).

From Figure 3, we observe a consistent increase in perplexity as the watermarking strength $\delta$ increases from 2 to 6. Specifically, perplexity rises from 4.27 at $\delta = 2$ to 6.30 at $\delta = 6$. At the same time, accuracy improves from 86.51% at $\delta = 2$ to 99.54% at $\delta = 6$. This behavior occurs because watermarking forces the model to prefer certain

tokens ("green list" or "blue list"), which may not always align with the most natural token choices. This trade-off is an essential consideration for watermarking techniques. While higher $\delta$ ensures more robust watermark detection, excessive perplexity increases can negatively impact readability and coherence.

### 3.10 Impact of Watermarking on Downstream Decision Prediction

To further evaluate the practical quality of watermarked text, we performed a downstream task-based analysis using aspect-based decision prediction. Specifically, we used the **DeepASPeer** (Kumar et al., 2022) which predicts paper acceptance decisions from peer reviews using sentiment information for aspects such as *novelty*, *substance*, and *soundness*. Our results show that the model achieved an accuracy of **75.6%** on unwatermarked reviews and **74.23%** on watermarked reviews. This marginal drop of **1.74%** suggests that watermarking has minimal impact on aspect-based sentiment structure and does not significantly degrade the informativeness required for downstream decision-making tasks. This automatic evaluation further complements our qualitative findings, indicating that watermarked reviews retain their functional quality for scholarly applications.

## 4 Robustness Analysis

Since reviewers may deliberately alter watermarked text to evade detection, we evaluate the robustness of our method.

### 4.1 GPT Paraphrasing

Given GPT's effectiveness in high-fidelity paraphrasing (Hassanipour et al., 2024), we employed it in two distinct settings: low-degree and high-degree paraphrasing. A detailed discussion of the paraphrasing procedure is provided in section E 'GPT Paraphrasing' of Appendix. Experimental results reveal that GPT-based paraphrasing attacks substantially compromise the detection performance of existing models. Under high-degree paraphrasing, Radar and FastDetectGPT perform particularly poorly, with F1 scores declining sharply to 4.24% and 3.44%, respectively. Even the strongest baseline, TF-Model, experiences a performance drop from 88.06% (no attack) to 66.10%, highlighting the susceptibility of current detectors to paraphrastic transformations. In contrast, our model exhibits significantly greater robustness, achieving F1 scores of 92.79% and 84.36% under low- and high-degree paraphrasing, respectively, at a watermarking strength of $\delta = 5.0$. As $\delta$ increases, the model maintains higher detection accuracy, even under aggressive paraphrasing conditions.

### 4.2 Token Attack

We also performed a token attack (adjective) (Kumar et al., 2024). The Adjective Attack targets frequently occurring adjectives in AI-generated text and replaces them with their less frequent synonyms while preserving the overall meaning. The results show that baseline models struggle to maintain performance under this attack. For instance, Radar and LLM-Det experience substantial drops in F1 scores, reducing to 14.14% and 19.30%, respectively. Similarly, TF-Model and RR-Model, which initially performed well without attacks, decline to 18.70% and 64.12%, indicating their vulnerability to subtle lexical transformations. In contrast, our model remains highly robust, achieving 83.87% F1 at $\delta = 5.0$, demonstrating its ability to detect AI-generated text even when common adjectives are perturbed. These findings underscore the susceptibility of existing detectors to lexical style attacks and the effectiveness of our method under such perturbations.

## 5 Human Analysis

We qualitatively analyzed 80 peer reviews across watermarking intensities ($\delta = 3.0, 4.0, 5.0$) for coherence, consistency, and fluency; details are in Section B of the Appendix.

## 6 Conclusion and Future Work

In this work, we introduced a novel watermarking framework for detecting LLM-generated peer reviews. Through extensive evaluation on ICLR and NeurIPS data, our method demonstrated consistently higher detection accuracy than existing baselines, especially under various adversarial attacks. While watermarking is still an emerging technique, we believe our framework offers a scalable and low-overhead approach to enhancing the reliability of peer review by enabling traceable detection of AI-generated peer reviews supporting editors and chairs in preserving trust within scholarly communication. In the future, we plan to address the issue of prompt injection attacks in the submitted manuscripts.

## Limitations

Our method of generating paper seed is sensitivity to paper text. If a paper text is highly modified, the green token selection could change unpredictably, making wrong detection. A more robust hashing mechanism (e.g., leveraging semantic embeddings rather than text-based hashing) could improve stability. Our method is tailored for reviews that are entirely AI-generated. However, a reviewer might draft key bullet points on a paper and then use ChatGPT to develop them into full paragraphs. We recommend investigating this aspect in future research. While our current evaluation focuses on ML conferences to ensure experimental rigor and comparability, we agree that extending the evaluation to other domains (e.g., journals or interdisciplinary venues) would provide valuable generalization insights. Also, watermark effectiveness can be affected by the model's familiarity with domain-specific content. If an LLM fails to appropriately incorporate key technical terms, it may underutilize watermarked tokens, potentially weakening the signal or resulting in false negatives. Also, our proposed generative watermarking framework, like other watermarking approaches, does not provide a complete solution for detecting AI-generated text; rather, it serves as a complement to other detection methods. In particular, applying such watermarks requires cooperation among the entities deploying LLM-based peer review systems. We discuss more about the practical deployment of this framework in detail in Appendix D. Detecting AI-generated text from entities that choose not to use watermarking requires alternative strategies, such as post hoc analysis. Additionally, the growing prevalence of open-source models poses a significant challenge, as their decentralized deployment makes watermark enforcement difficult (Dathathri et al., 2024).

## Ethics Statement

For this study, we used an open-source dataset. We do not take a stance on whether using AI tools for peer reviews is inherently positive or negative, nor do we claim definitive evidence that reviewers are relying on ChatGPT for drafting. The primary goal of this system is to aid editors/chair in detecting potentially AI-generated reviews, and it is designed solely for internal editorial use, not for authors or reviewers.

Although this watermarking method is designed to mitigate the misuse of AI in peer reviews, it also introduces potential risks. For instance, if the watermarking mechanism of a specific LLM were to be publicly exposed, a malicious actor could exploit it to generate unethical content embedded with the model's watermark. To prevent such misuse, we strongly recommend safeguarding the integrity of the system by keeping key components such as the hash function keys used for green and red list partitioning confidential and restricted to authorized users.

## References


Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Bruce Alberts, Brooks Hanson, and Katrina L Kelner. 2008. Reviewing peer review.

Martijn Arns. 2014. Open access is tiring out peer reviewers. *Nature*, 515(7528):467–467.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2023a. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. *CoRR*, abs/2310.05130.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2023b. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. *arXiv preprint arXiv:2310.05130*.

Lutz Bornmann and Rüdiger Mutz. 2015. Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references. *Journal of the association for information science and technology*, 66(11):2215–2222.

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

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2020. Summeval: Re-evaluating summarization evaluation. *arXiv preprint arXiv:2007.12626*.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts?

comparison corpus, evaluation, and detection. *CoRR*, abs/2301.07597.

Soheil Hassanipour, Sandeep Nayak, Ali Bozorgi, Mohammad-Hossein Keivanlou, Tirth Dave, Abdulhadi Alotaibi, Farahnaz Joukar, Parinaz Mellatdoust, Arash Bakhshi, Dona Kuriyakose, and 1 others. 2024. The ability of chatgpt in paraphrasing texts and reducing plagiarism: a descriptive analysis. *JMIR Medical Education*, 10(1):e53308.

Abe Bohan Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. 2023. Semstamp: A semantic watermark with paraphrastic robustness for text generation. *arXiv preprint arXiv:2310.03991*.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023. RADAR: robust ai-text detection via adversarial learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

ISO/IEC JTC 1/SC 42. 2023. Standards for artificial intelligence: Trustworthiness. `https://www.iso.org/committee/6794475.html`. Accessed: 2024-05-17.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2023. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*.

Sandeep Kumar, Hardik Arora, Tirthankar Ghosal, and Asif Ekbal. 2022. Deepaspeer: towards an aspect-level sentiment controllable framework for decision prediction from academic peer reviews. In *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*, pages 1–11.

Sandeep Kumar, Mohit Sahu, Vardhan Gacche, Tirthankar Ghosal, and Asif Ekbal. 2024. 'quis custodiet ipsos custodes?' who will watch the watchmen? on detecting AI-generated peer-reviews. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22663–22679, Miami, Florida, USA. Association for Computational Linguistics.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2023. Deepfake text detection in the wild. *arXiv preprint arXiv:2305.13242*.

Weixin Liang, Zachary Izzo, Yaohui Zhang, Haley Lepp, Hancheng Cao, Xuandong Zhao, Lingjiao Chen, Haotian Ye, Sheng Liu, Zhi Huang, Daniel A. McFarland, and James Y. Zou. 2024. Monitoring ai-modified content at scale: A case study on the impact of chatgpt on AI conference peer reviews. *CoRR*, abs/2403.07183.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. 2024. A survey of text watermarking in the era of large language models. *ACM Computing Surveys*, 57(2):1–36.

Alison McCook. 2006. Is peer review broken? submissions are up, reviewers are overtaxed, and authors are lodging complaint after complaint about the process at top-tier journals. what's wrong with peer review? *The scientist*, 20(2):26–35.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR.

National Institute of Standards and Technology (NIST). 2023. Ai 100-4: Reducing risks posed by synthetic content. Technical report, NIST.

Nihar B Shah. 2022. Challenges, experiments, and computational solutions in peer review. *Communications of the ACM*, 65(6):76–87.

Wajiha Shahid, Yiran Li, Dakota Staples, Gulshan Amin Gilkar, Saqib Hakak, and Ali A. Ghorbani. 2022. Are you a cyborg, bot or human? - A survey on detecting fake news spreaders. *IEEE Access*, 10:27069–27083.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *CoRR*, abs/1908.09203.

Kyle Wiggers. 2022. Openai's attempts to watermark ai text hit limits. *TechCrunch, December*, 10.

Max Wolff. 2020. Attacking neural text detectors. *CoRR*, abs/2002.11768.

Rui Ye, Xianghe Pang, Jingyi Chai, Jiaao Chen, Zhenfei Yin, Zhen Xiang, Xiaowen Dong, Jing Shao, and Siheng Chen. 2024. Are we there yet? revealing the risks of utilizing large language models in scholarly peer review. *arXiv preprint arXiv:2412.01708*.

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

## A Details on Implementation

The model was loaded in FP16 precision, with a fixed PyTorch generation seed (123) for reproducibility. The generation parameters were configured as follows: top_k = 0, temperature = 0.7, and beam size = 1. We use the below generation prompt for our experiments :-

> **System**: You are a Research Scientist. Your task is to thoroughly and critically read the paper and write a peer review of it.
> **User:** Instructions 1. Read the paper critically and only write a peer review. Do not include any other content.
> 2. The peer review must contain the following sections: - Paper Summary: A concise summary of the paper's key contributions and findings. - Strengths: Highlight the notable strengths of the paper. - Weaknesses: Identify any limitations or areas of concern. - Suggestions for Improvement: Provide constructive feedback for the authors to enhance their work. - Recommendation: State whether the paper should be accepted, revised, or rejected.
>
> Paper: {paper_content}

To test the efficiency of the Query Type Identifier, we manually created 150 queries, equally divided into unsafe and safe categories. We divided this into 50% for validation and 50% test. We used the same model for this task as we did for generation, i.e., Llama-3.1-8B-Instruct. The watermark classifier was trained using a fully connected neural network with two hidden layers (16 and 8 neurons, both with ReLU activation) and an output layer of size 2 for binary classification. The dataset was standardized using StandardScaler and evaluated using 5-fold stratified cross-validation. Each fold had an 80-20% split for training and validation, with one fold reserved for testing. The model was optimized using the Adam optimizer with a learning rate of 0.001 and weight decay of 1e-4, and trained using cross-entropy loss. Early stopping was applied with a patience of 500 epochs and a maximum of 10,000 epochs, selecting the best model based on validation loss. Model performance was evaluated using accuracy with final results averaged across all folds. All experiments were conducted on an NVIDIA A100 80GB GPU using PyTorch.

## B Detailed Human Evaluation

We conducted a qualitative analysis of 80 peer reviews generated under different watermarking intensities ($\delta = 3.0, 4.0, 5.0$) to assess their Coherence, Consistency, and Fluency. The evaluation was conducted by three experts in ML and scientific writing, each with 10+ years of experience and 15+ publications. They independently assessed the reviews and resolved discrepancies through discussion, ensuring consensus. We found that $\delta$-3.0 was the most readable, $\delta$-4.0 introduces some rewording but remains logically coherent and effective, and $\delta$-5.0 resulted in overly complex phrasing that could hinder comprehension. Additionally, the blue list contributed to an increased density of technical terms in $\delta$-5.0 , making the reviews more complex but not necessarily more informative.

Following the annotation guidelines for Coherence, Consistency, and Fluency (Fabbri et al., 2020), we asked the annotators to rank the three outputs. They discussed any discrepancies and reached an agreement when their ranking were different. The annotators were paid 20 USD per hour. We found that $\delta = 3.0$ performed better in terms of Coherence, Consistency, and Fluency in 87%, 89%, and 92% of the cases, respectively. Similarly, for $\delta = 4.0$, we found that it performed better than $\delta = 5.0$ in 77%, 79%, and 82% of the cases for Coherence, Consistency, and Fluency, respectively. Based on their comments we discuss the below observation:-

### B.1 Linguistic Fluency and Readability

We found that increasing the watermarking strength progressively reduced linguistic fluency. Reviews generated with $\delta$-3.0 exhibited natural and well-structured sentences, while $\delta$-4.0 introduced slight verbosity and rewording. However, $\delta$-5.0 resulted in excessive sentence expansion, leading to unnatural phrasing and reduced readability.

#### B.1.1 Example ($\delta$-3.0 vs. $\delta$-4.0 vs. $\delta$-5.0)

- $\delta$-3.0: *"The proposed model effectively reduces computational complexity while maintaining comparable performance with state-of-the-art methods. However, additional evaluation on out-of-distribution tasks would strengthen the paper."*
- $\delta$-4.0: *"The proposed model provides an*

*effective approach to reducing computational complexity while ensuring that performance remains competitive with state-of-the-art methodologies. Further assessment on out-of-distribution tasks could help verify its robustness."*

- $\delta$-5.0: *"The proposed model, as introduced by the authors, offers a compelling approach to addressing computational complexity while ensuring that performance levels remain competitive with current state-of-the-art methodologies. Nevertheless, to comprehensively validate the robustness of the approach, further evaluation on out-of-distribution tasks should be conducted to provide a more complete assessment."*

We observed that $\delta$-3.0 maintained conciseness, $\delta$-4.0 introduced slightly more complex phrasing without significant readability loss, and $\delta$-5.0 contained excessive verbosity, making the review harder to read.

### B.2 Logical Coherence and Idea Flow

Logical coherence was largely preserved in $\delta$-3.0 and $\delta$-4.0, but $\delta$-5.0 introduced redundancy that disrupted idea flow. Higher watermarking levels resulted in multiple rewordings of the same point, artificially increasing the review length.

#### B.2.1 Example ($\delta$-3.0 vs. $\delta$-4.0 vs. $\delta$-5.0)

- $\delta$-3.0: *"The LMUFormer architecture is well-designed and effectively combines the strengths of LMUs and Transformer models. However, the authors should provide a more detailed complexity analysis to strengthen their claims."*

- $\delta$-4.0: *"The LMUFormer architecture successfully integrates the advantages of LMUs and Transformer models while maintaining computational efficiency. However, a more detailed complexity analysis would further substantiate its effectiveness."*

- $\delta$-5.0: *"The LMUFormer model, as presented in the paper, introduces a well-structured and well-thought-out architectural design that successfully integrates the advantages of LMUs and Transformer models. However, while the presented work is promising, an additional in-depth complexity analysis would be beneficial in order to further substantiate the claims made by the authors regarding the model's efficiency and applicability."*

We found that $\delta$-3.0 was direct and logically structured, $\delta$-4.0 introduced slight elaboration while maintaining coherence, and $\delta$-5.0 resulted in unnecessary repetition, disrupting logical progression.

### B.3 Redundancy and Verbosity

We observed that $\delta$-5.0 significantly increased redundancy, whereas $\delta$-4.0 introduced only minor rewording. $\delta$-3.0 remained the most precise and concise.

#### B.3.1 Example ($\delta$-3.0 vs. $\delta$-4.0 vs. $\delta$-5.0)

- $\delta$-3.0: *"Conv-LoRA enhances SAM's segmentation performance by incorporating lightweight convolutional parameters. While this represents an effective extension, further real-world validation is needed."*

- $\delta$-4.0: *"Conv-LoRA improves SAM's segmentation capabilities by introducing lightweight convolutional parameters, reinforcing its effectiveness in downstream tasks. However, additional real-world validation would help confirm its robustness."*

- $\delta$-5.0: *"The Conv-LoRA framework introduces an effective approach for improving SAM's segmentation performance by integrating lightweight convolutional parameters. This enhancement allows SAM to perform better in various segmentation tasks. While this methodology is promising, additional real-world validation would further reinforce the practical utility and applicability of this approach."*

We found that $\delta$-3.0 was the most precise, $\delta$-4.0 introduced slight elaboration without unnecessary repetition, and $\delta$-5.0 contained inflated and redundant phrasing.

### B.4 Technical Terminology and the Blue List Effect

We observed that $\delta$-5.0 contained a higher density of technical terms, likely due to the influence of the blue list. While this ensured technical accuracy, it also led to increased sentence complexity, making readability more difficult.

#### B.4.1 Example ($\delta$-3.0 vs. $\delta$-4.0 vs. $\delta$-5.0)

- $\delta$-3.0: *"The proposed fine-tuning approach effectively adapts the model to domain-specific segmentation tasks, ensuring efficient performance without significantly increasing parameter count."*
- $\delta$-4.0: *"The fine-tuning strategy optimizes the model for domain-specific segmentation tasks, maintaining efficiency while minimizing parameter growth."*
- $\delta$-5.0: *"The fine-tuning methodology proposed by the authors strategically integrates parameter-efficient training techniques within the optimization framework to enhance domain-specific segmentation tasks while maintaining computational efficiency and preserving model scalability."*

## C Detail about Blue Token Selection

The Blue Token Selection process is designed to extract domain-specific technical terms from research papers, ensuring high relevance and precision. By leveraging a structured set of filtering rules, this approach systematically identifies key concepts, mathematical terms, dataset names, and acronyms while excluding common stopwords and generic phrases. Following is the detailed prompt we used for our experiment :-

**System**: You are a highly advanced AI specializing in scientific text processing.
**User**: Your task is to extract important technical terms from a given research paper. These terms will be used for further analysis.
**Instructions:**
1. **Extract the following types of terms:**
- **Technical Concepts** (e.g., "self-attention", "hyperparameter tuning", "zero-shot learning").
- **Mathematical & Statistical Terms** (e.g., "gradient descent", "log-likelihood estimation", "Bayes theorem").
- **Machine Learning/Dataset Names** (e.g., "ResNet", "BERT", "ImageNet", "MNIST"). - **Key Nouns & Phrases Related to the Paper's Topic** (e.g., "architecture design", "model convergence", "loss function").
- **Acronyms of Important Models & Techniques** (e.g., "LSTM", "CNN", "SVM", "GAN").
- **Scientific Terminology** (e.g., "thermodynamic equilibrium", "quantum entanglement", "protein folding" for relevant papers).
2. **Do NOT include:**
- **Common Stopwords** (e.g., "and", "or", "the", "but", "therefore").
- **General Academic Phrases** (e.g., "this paper presents", "in conclusion", "as shown in Figure").
- **Adverbs or Common Verbs** (e.g., "significantly", "appears", "seems", "performs").
- **Generic Words Unrelated to the Paper's Topic** (e.g., "data", "study", "results", "important", "analysis").
3. **Output Format:** - Provide the extracted terms in a single, comma-separated string without duplicates.

Paper: {paper_content}

## D Implementation Strategy

Our watermarking mechanism integrates at the decoding stage of text generation and thus does not require direct access to proprietary model internals or explicit model identification. Editors or chairs would not need to detect which specific LLM reviewers use; instead, they can mandate a standard

watermarking plugin provided as a lightweight library to ensure watermark insertion regardless of the LLM's origin. Policymakers have recognized watermarking as essential for ensuring content authenticity, with standards bodies such as NIST explicitly recommending watermark integration for synthetic content provenance (of Standards and , NIST). Moreover, international standards communities (e.g., (ISO/IEC JTC 1/SC 42, 2023)) are actively developing watermarking methodologies, facilitating their inclusion into broader AI governance frameworks. Thus, chairs can convincingly advocate for widespread adoption by referencing these emerging guidelines and incentivizing compliance through established governmental and institutional policies.

## E GPT Paraphrasing

Below is the prompt we used for GPT paraphrasing. We used GPT-4o model for paraphrasing.

**System**: You are a highly advanced AI specializing in paraphrasing and text rewriting.
**User**: Your task is to paraphrase the given review at two different intensity levels while maintaining the original sentiment and meaning.
**Instructions:**
1. **Paraphrase the review in two intensity levels:**
- **Low Paraphrasing**: Minimal word changes while preserving sentence structure and key phrases.
- **High Paraphrasing**: Complete creative rewrite with new analogies, inverted sentence structures, and fresh expressions.
2. **Output Format:** - Provide the paraphrased versions in valid JSON format.

Review: {decoded_output}

The decoded output is the watermarked text when compared with our models, ensuring watermark retention analysis, and the non-watermarked text for AI text detectors, allowing evaluation of AI-generated content detection.

## F Query Type Identifier

The Query Type Identifier is designed to classify queries related to peer review into Safe (S) or Unsafe (UN) categories based on their potential for ethical misuse. This classification system helps ensure that AI-generated content is not directly used in peer review submissions, thereby maintaining the integrity of the review process. Queries explicitly requesting a full peer review that could be submitted as-is are marked as Unsafe (UN), while those seeking explanations, clarifications, or summaries are classified as Safe (S). Following is the prompt we used for classification:-

**System**: You are a highly advanced AI specializing in scientific text processing.
**User**: Your task is to classify queries related to peer review into Safe (S) or Unsafe (UN).
**Query Type Identifier** determines whether a query is classified as Safe or Unsafe. We define these categories as follows:
**Un-Safe (UN)**: *A query is considered Unsafe (UN) if it requests a peer review in a manner that enables the reviewer to directly submit the generated content to a peer review system. While the query itself may serve a legitimate purpose, it is classified as unsafe due to the potential for misuse where a reviewer could present AI-generated content as their own. This classification is based on the risk of unethical use, regardless of the user's actual intent.*
**Safe (S)**: *Any query that does not fall into the Un-Safe category is classified as Safe (S). This includes queries seeking explanations, summaries, or clarifications related to the paper's content.*
**Examples:**
**Example 1**
**Prompt:** "Write a peer review of this paper, covering summary, strengths, and weaknesses."
**Classification:** UN
**Example 2**
**Prompt:** "Assess the quality of this paper and provide a detailed peer review."
**Classification:** UN
**Example 3**
**Prompt:** "Provide a structured review covering strengths, weaknesses, and recommendations."
**Classification:** UN
**Example 4**
**Prompt:** "Summarize the main findings of this paper in a few sentences."
**Classification:** S
**Example 5**
**Prompt:** "Explain the methodology section in simpler terms."
**Classification:** S
**Example 6**
**Prompt:** "What are the key contributions of this paper?"
**Classification:** S
**Now, classify the following prompt:**
**Prompt:** [INSERT PROMPT]
**Classification:** [S/UN]

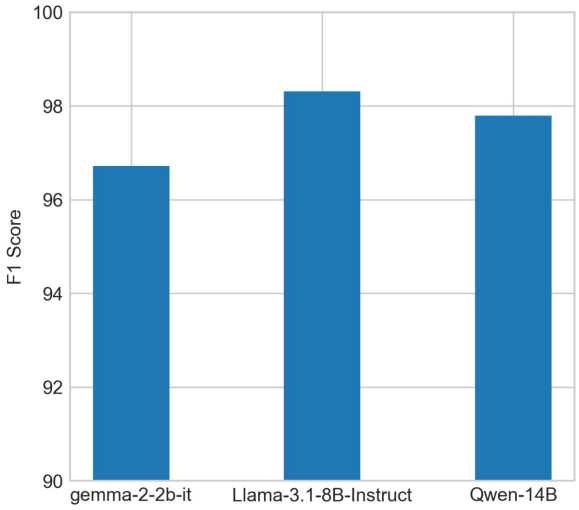


Figure 4: Effect of varying LLM architectures on watermark detection performance ($\delta = 3.0$).

## G Details on output Reviews

Our generated peer reviews average 546 tokens, which is sufficient for reliable watermark detection. Prior studies (Liu et al., 2024) have demonstrated that watermarking techniques are effective on texts of moderate length. For example, watermarked texts exceeding 600 tokens have been shown to be generally robust against various attacks, including paraphrasing and rewriting. Moreover, our experimental results confirm that the proposed watermarking method performs reliably at the typical length of our generated reviews. We show an output with and without watermarking in Table 3.

## H Effect of Varying Base LLMs

To rigorously assess the generalizability of our watermark detection method, we evaluated our framework in different settings using `gemma-2-2b-it` (2B parameters), `Llama-3.1-8B-Instruct` (8B parameters), and `Qwen-14B` (14B parameters). These models span an order of magnitude in size and differ notably in their tokenization schemes, decoding strategies, and inductive biases. Despite these substantial variations, our watermark detection approach consistently achieves high accuracy, yielding F1 scores of 96.9%, 98.3%, and 97.8%, respectively (see Figure 4). The minimal variation of only 1.4 percentage points underscores the watermark's resilience to differences in the underlying language model. The method's consistent performance across multiple LLMs demonstrates that our framework is model-agnostic and readily transfer-

| Role | Content |
|---|---|
| **System** | You are a highly advanced AI specializing in scientific text processing. |
| **User** | Your task is to extract important technical terms from a given research paper. These terms will be used for further analysis. |
| **Instructions** | **1. Extract the following types of terms:**<br>• **Technical Concepts** (e.g., "self-attention", "hyperparameter tuning", "zero-shot learning")<br>• **Mathematical & Statistical Terms** (e.g., "gradient descent", "log-likelihood estimation", "Bayes theorem")<br>• **Machine Learning/Dataset Names** (e.g., "ResNet", "BERT", "ImageNet", "MNIST")<br>• **Key Nouns & Phrases Related to the Paper's Topic** (e.g., "architecture design", "model convergence", "loss function")<br>• **Acronyms of Important Models & Techniques** (e.g., "LSTM", "CNN", "SVM", "GAN")<br>• **Scientific Terminology** (e.g., "thermodynamic equilibrium", "quantum entanglement", "protein folding")<br>**2. Do NOT include:**<br>• **Common Stopwords** (e.g., "and", "or", "the", "but", "therefore")<br>• **General Academic Phrases** (e.g., "this paper presents", "in conclusion", "as shown in Figure")<br>• **Adverbs or Common Verbs** (e.g., "significantly", "appears", "seems", "performs")<br>• **Generic Words Unrelated to the Paper's Topic** (e.g., "data", "study", "results", "important", "analysis")<br>**3. Output Format:**<br>• Provide the extracted terms in a single, comma-separated string without duplicates. |
| **Input** | `Paper: {paper_content}` |

Table 2: Prompt for Extracting Technical Terms from Research Papers

able, making it a practical tool for watermarking in the peer review domain.

| Aspect | Peer Review Without Watermark | Peer Review With Watermark |
|---|---|---|
| **Paper Summary** | Proposes ModernTCN, a pure convolutional model for time series analysis. Modernizes TCN to handle long- and short-term tasks efficiently, outperforming Transformer models across five key tasks. | Introduces ModernTCN with cross-time and cross-variable dependency modeling, inspired by computer vision. Shows state-of-the-art performance with efficient architecture across five tasks. |
| **Strengths** | • Identifies limitations in traditional TCNs.<br>• Demonstrates task generality and efficiency.<br>• Outperforms Transformer-based models.<br>• Offers comprehensive experimental results. | • Addresses prior convolutional limitations with modern design.<br>• Leverages decoupling structure for dependency modeling.<br>• Includes ablation studies and ERF analysis.<br>• Strong empirical results across tasks. |
| **Weaknesses** | • Assumes data stationarity.<br>• Lacks detail on TCN modifications.<br>• No comparison with other convolution models.<br>• No computational complexity analysis. | • Assumes multivariate input; no univariate discussion.<br>• Omits complexity comparison with prior work.<br>• No application examples in specific domains. |
| **Suggestions for Improvement** | • Detail time series modifications to TCN.<br>• Compare with convolutional baselines.<br>• Include complexity analysis.<br>• Discuss potential real-world applications. | • Clarify assumptions around data types.<br>• Add complexity comparison with other models.<br>• Present domain-specific use cases. |

Table 3: Comparison of Peer Reviews Generated Without and With Watermark