Toward Watermarking Peer Reviews Generated by Large Language Models

Anonymous ACL submission

Abstract

The integrity of the peer-review process is crucial for maintaining scientific rigor and trust in academic publishing. This process relies on domain experts critically evaluating the merits of submitted manuscripts. However, the increasing use of large language models (LLMs) in academic writing raises concerns about the authenticity and reliability of peer reviews. Previous works have focused on estimating the proportion of AI-generated peer reviews or developing AI-generated text detectors. However, existing detectors struggle against adversarial attacks and often require domain-specific retraining. To address these challenges, we propose a watermarking framework. Our Query-Aware Response Generation module selectively applies watermarking when a user uploads a research paper. The Watermark Injection method embeds subtle yet detectable signals while preserving scientific terminology. Finally, Watermark Detection to enable editor/chair to verify review authenticity. Extensive experiments on ICLR and NeurIPS peer reviews demonstrate that our method outperforms various AI text detectors under adversarial attacks. Our results highlight watermarking as a robust and scalable solution for preserving integrity in AI-assisted peer review. We make our code, dataset, and model public[1].

1 Introduction

The advent of large language models (LLMs), such as ChatGPT, GPT-4 (Achiam et al., 2023) , DeepSeek-V3 (Liu et al., 2024) has revolutionized natural language generation. However, their misuse also raises concerns, particularly about fake news (Zhang and Gao, 2023; Silva and Vaz, 2024), fake hotel reviews (Ignat et al., 2024), and fake restaurant reviews (Gambetti and Han, 2024). The remarkable human-like fluency and coherence of content generated by these models makes it challenging, even for experts, to determine whether a text is authored by humans or LLMs (Shahid et al., 2022).

Scholarly peer review is a cornerstone of scientific advancement, providing expert evaluations to uphold research integrity and credibility before publication (Alberts et al., 2008). However, the increasing volume of manuscript submissions (Bornmann and Mutz, 2015; McCook, 2006) has placed significant strain on the peer review system (Arns, 2014). A study (Liang et al., 2024) analyzed peer reviews from AI conferences and found that 6.5% to 16.9% of the submitted text may have been significantly altered using LLMs. Their findings indicate a notable rise in ChatGPT usage within three days of review deadlines, with higher estimated reliance among reviewers who do not engage in ICLR/NeurIPS author rebuttals. Moreover, increased ChatGPT usage correlates with lower self-reported confidence in reviews. Another study (Ye et al., 2024) found that manipulating 5% of the reviews could potentially cause 12% of the papers to lose their position in the top 30% rankings. They found that LLMs exhibit inherent flaws, such as potentially assigning higher ratings to incomplete papers compared to full papers and favoring well-known authors in single-blind review processes. Also, injecting covert deliberate content into manuscripts allows authors to explicitly manipulate LLM reviews, leading to inflated ratings and reduced alignment with human reviews. They suggested that LLMs are not yet reliable enough to serve as primary reviewers due to risks of manipulation and inherent flaws. To ensure fairness and accuracy, they emphasize the need for stricter safeguards and evaluation mechanisms before broader adoption. According to ACL policy[2], AI tools can

[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

assist with paraphrasing and proofreading, especially for non-native English speakers, but reviewers must independently generate the review content. Existing AI text detectors are vulnerable to adversarial attacks and require task-specific training as well as continuous retraining for each conference and dataset, making reliability and adaptability across diverse academic settings challenging.

> A peer review can be generated in seconds by uploading a paper and submitting a query. *Can watermarking prevent its misuse?*

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. Additionally, we implement Watermark Detection, which allows editors and conference chairs to verify the authenticity of peer reviews. Our watermarking framework outperforms various AI text detectors, achieving higher detection accuracy even under adversarial conditions on ICLR and NeurIPS peer reviews. Our work aims to reinforce ethical practices in scholarly communication and safeguard the integrity of the peer review process. In doing so, we contribute to the broader effort of ensuring that the benefits of advanced language models are realized without compromising the trust and reliability that underpin academic research.

Our contributions are summarized as follows:-

- We propose the novel task of watermarking peer reviews to ensure authenticity and traceability.
- We introduce a novel watermarking framework specifically designed for LLM-generated peer reviews.
- Our results show that our watermarking framework outperforms various AI text detectors under adversarial conditions.

2 Related Work

2.1 AI Text Detection

Zero-shot text detection identifies AI-generated text without requiring training on specific data, relying solely on the model that produced it (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.

Training-Based Text Detection involves fine-tuning language models on labeled data to distinguish AI-generated text from human-written text, improving detection through learned patterns and statistical features. Guo et al. (2023) developed the OpenAI text classifier by training it on a large dataset comprising millions of texts. Similarly, GPT-Sentinel (Chen et al., 2023) fine-tuned RoBERTa (Liu et al., 2019) and T5 (Raffel et al., 2020) models using OpenGPT-Text. However, heavy dependence on training data makes many of these models susceptible to adversarial attacks (Wolff, 2020).

2.2 LLM Watermarking

Watermarking AI-generated text, introduced by Wiggers (2022), embeds an imperceptible pattern to verify authorship, similar to encryption. Detectable by algorithms but invisible to humans, some methods integrate machine learning models into the watermarking process (Abdelnabi and Fritz, 2021; Munyer and Zhong, 2023; Yoo et al., 2023; Qiang et al., 2023). Watermarks can be embedded without requiring modifications to the underlying language model, allowing standard models to generate watermarked text (Kirchenbauer et al., 2023). This method introduces watermarks with sufficient entropy, effectively altering the distribution of generated text. Additionally, Zhao et al. (2023b) proposed injecting secret sinusoidal signals into decoding steps at the token level to embed watermarks. Furthermore, Chakraborty et al. (2023a) demonstrate that watermarked texts can often be bypassed, as paraphrasing does not significantly disrupt watermark signals.

2.3 Statistical Estimation Approach

Several studies have explored the theoretical feasibility of accurately detecting AI-generated text at an individual level (Weber-Wulff et al., 2023; Sadasivan et al., 2023; Chakraborty et al., 2023b). 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. Their experiments on papers from select AI conferences aimed to assess the extent to which peer reviews may have been significantly influenced or modified by large language models.

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 potential 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.

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 E.

3.3 Watermark Injection

Algorithm 1 Watermark Injection

Require: Vocabulary V, Paper P , Watermark Strength δ, Green List Fraction γ

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

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

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

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. Our approach is inspired by prior watermarking methods (Kirchenbauer et al., 2023; Zhao et al., 2023a). In this work, we utilize the soft watermarking technique that introduces probabilistic biases in token selection to subtly mark text generated by large language models (LLMs). Algorithm 1 outlines the complete watermark injection process.

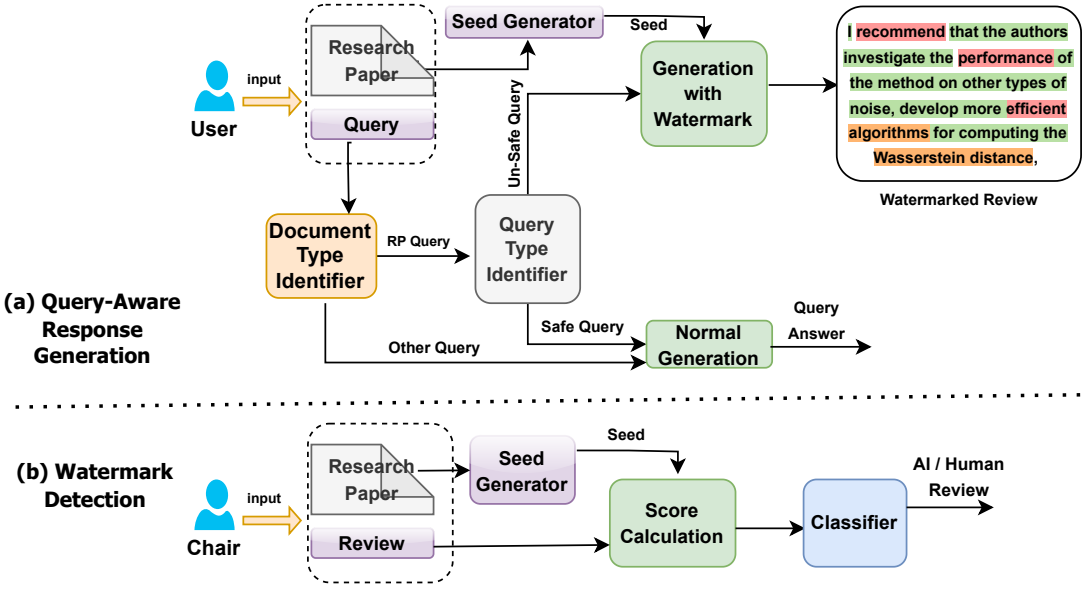

Figure 1: Overview of the Proposed Watermarking Framework. (a) Watermark Generation (b) Watermark Detection; Here red indicates red token, green indicates green token, blue indicates blue token

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, is first encoded by mapping each character to its alphabetical index and applying a shift cipher based on the text length:

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

where c is the character and n is the total number of characters in T. This ensures that encoding remains text-dependent, enhancing uniqueness. To strengthen security, a secret key K_{secret} is concatenated with T, ensuring that different users generate distinct seeds:

$$I = T \| K_{\text{secret}} \quad (2)$$

The encoded representation is then hashed using SHA-256 for collision resistance:

$$H(I) = \text{SHA-256}(E(I)) \quad (3)$$

Finally, the hash is mapped to a bounded numeric space using modular reduction:

$$S(I) = H(I) \mod p \quad (4)$$

Even if an attacker identifies the specific paper text used for watermarking, they would still require two secret keys, K_{secret} and P, 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_{secret} and P 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 γ of tokens is selected as green, ensuring consistency across runs.

Let $|V|$ denote the vocabulary size, then:

$$|G| = \gamma|V|, \quad |R| = (1-\gamma)|V|.$$

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 γ 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. We discuss this in detail in Appendix C.

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:

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

where δ 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:

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

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. We discuss the algorithm in detail in Algorithm 2.

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_{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

For the final decision to determine whether a review is watermarked, we trained a simple MLP classifier using the z-score and marked token fraction. The classification is given by:

$$y = \text{MLP}(z, f_m)$$

Here y indicates whether the review is watermarked.

4 Experiments

4.1 Implementation Details

We used 1,090 papers from ICLR and NeurIPS (year: 2022) for our experiments from (Kumar et al., 2024). For generation, we used the Llama-3.1-8B-Instruct[3] in our experiments. We discuss the implementation details in Appendix A.

4.2 Main Result

We evaluate our watermarking-based detection method against multiple AI-generated text detec-

[3] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

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
Our Model	δ= 3.0	91.45	85.29	76.42	77.81
	δ= 4.0	95.20	88.14	79.56	80.32
	δ= 5.0	**98.31**	**92.79**	**84.36**	**83.87**

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

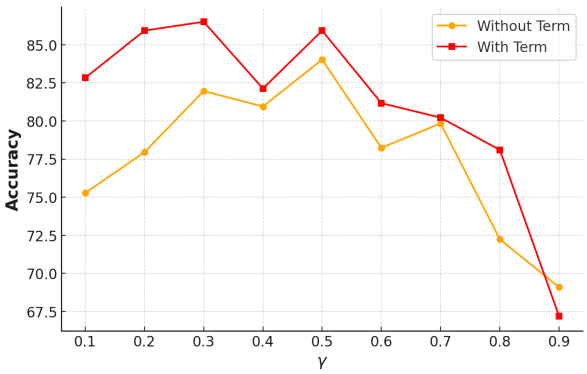

Figure 2: Effect of Watermarking on varying γ on with only green token (without Term) and with green token and blue token (With Term); δ= 2.0

tors, including RADAR (Hu et al., 2023), DEEPFAKE (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). 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. Our analysis reveals that existing AI detectors exhibit extreme sensitivity to adversarial attacks, with Fast Detect suffering a 94.30% drop (60.36% → 3.44%) and Radar declining by 91.17% (48.02% → 4.24%) under high paraphrasing. Similarly, TF-Model's F1 score decreases by 78.76% (88.06% → 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 (δ=5.0), outperforming all baselines by a substantial margin. Even with lower δ values, our model demonstrates resilience, with δ=3.0 yielding 76.42% and δ=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 4.4. Additionally, We found that the Query Type Identifier achieves an accuracy of 95.5% on the test set.

4.3 Effect of varying γ 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 γ 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 γ 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.

4.4 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. 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, rather than relying solely on arbitrary green token enforcement.

4.5 Effect of Watermarking Strength (δ) on Detection Accuracy

The graph demonstrates a positive correlation between watermarking strength (δ) and detection accuracy. As δ 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 δ 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.

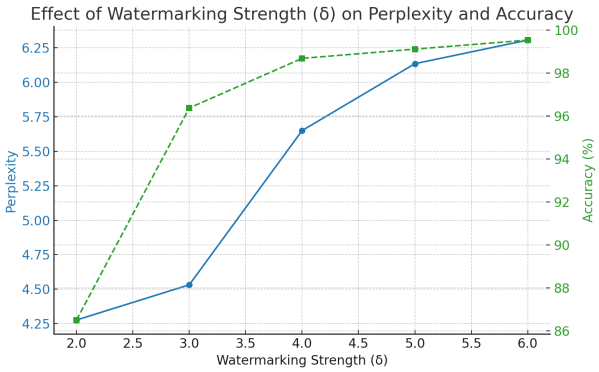

Figure 3: Effect Of Watermarking Strength (δ) on Perplexity and Accuracy; $\delta = 2.0$

4.6 Effect of Watermarking Strength (δ) 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 δ 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 δ ensures more robust watermark detection, excessive perplexity increases can negatively impact readability and coherence.

5 Robustness Analysis

Reviewers may attempt to evade watermarking by deleting, distorting, or falsifying embedded signals, obscuring the review's origin. Ensuring watermark stability is crucial. This section evaluates its robustness of our method against adversarial attacks.

5.1 GPT Paraphrasing

As GPT has shown promise in paraphrasing (Hassanipour et al., 2024), we employed it in two distinct settings: low paraphrase and high paraphrase. A detailed discussion of our approach and implementation can be found in the Appendix D. The results indicate that GPT-based paraphrasing attacks significantly degrade the performance of baseline models, with F1 scores dropping drastically, especially under high paraphrasing. Models like Radar and Fast Detect perform particularly poorly, with scores plummeting to 4.24% and 3.44%, respectively, under high paraphrasing. Even the strongest baseline, TF-Model, drops from 88.06% (w/o attack) to 66.10% under high paraphrasing, demonstrating the vulnerability of existing AI text detection methods to sophisticated paraphrasing. In contrast, our model shows remarkable robustness, maintaining 92.79% and 84.36% F1 scores under low and high paraphrasing attacks, respectively, when δ is set to 5.0. Increasing δ values indicate enhanced resilience, allowing the model to retain high detection accuracy even when faced with strong paraphrasing attack. This suggests that our proposed model significantly outperforms various AI text detectors in handling GPT-based paraphrasing attacks.

5.2 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. This attack aims to disrupt AI text detection models by altering stylistic elements rather than core semantics. 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 results highlight that while various AI text detectors are vulnerable to adjective attacks, our model effectively withstands such attacks.

6 Human Analysis

We conducted a qualitative analysis of 50 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 δ-3.0 was the most readable, δ-4.0 introduces some rewording but remains logically coherent and effective, and δ-5.0 resulted in overly complex phrasing that could hinder comprehension. Additionally, the blue list contributed to an increased density of technical terms in δ-5.0 , making the reviews more complex but not necessarily more informative. We discuss this in detail with examples in Appendix B.

7 Conclusion and Future Work

Our proposed watermarking framework effectively detects LLM-generated peer reviews while preserving scientific fluency and robustness against adversarial attacks. Experiments on ICLR and NeurIPS peer reviews show that our method outperforms existing AI text detectors, particularly under paraphrasing and token substitution attacks, where traditional models suffer significant performance degradation. While Fast-DetectGPT, Radar, and TF-Model resulted in a F1-score drop of over 90% under high paraphrasing, our method retains over 84% detection accuracy. Additionally, domain-specific blue token selection improves watermark detectability, achieving up to 10% higher detection rates at lower green token fractions ($\gamma = 0.1$ to 0.3) compared to random token-based watermarking. The logit adjustment mechanism further enhances robustness without compromising text fluency, ensuring effectiveness across different watermarking strengths (δ).

In future, we will explore semantic embeddings rather than text-based hashing to enhance stability despite paper text modifications. Additionally, we aim to extend detection to hybrid AI-human-generated reviews. Furthermore, integrating watermarking with other forms of AI-generated scientific text could further strengthen trust in scholarly communication.

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.

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

Sahar Abdelnabi and Mario Fritz. 2021. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *42nd IEEE Symposium on Security and Privacy, SP 2021, San Francisco, CA, USA, 24-27 May 2021*, pages 121–140. IEEE.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 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.

Megha Chakraborty, SM Tonmoy, SM Zaman, Krish Sharma, Niyar R Barman, Chandan Gupta, Shreya Gautam, Tanay Kumar, Vinija Jain, Aman Chadha, et al. 2023a. Counter turing test ctˆ 2: Ai-generated text detection is not as easy as you may think–introducing ai detectability index. *arXiv preprint arXiv:2310.05030*.

Souradip Chakraborty, Amrit Singh Bedi, Sicheng Zhu, Bang An, Dinesh Manocha, and Furong Huang. 2023b. On the possibilities of ai-generated text detection. *CoRR*, abs/2304.04736.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Gpt-sentinel: Distinguishing human and chatgpt generated content. *CoRR*, abs/2305.07969.

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*.

Alessandro Gambetti and Qiwei Han. 2024. Aigenfoodreview: A multimodal dataset of machine-generated restaurant reviews and images on social media. *arXiv preprint arXiv:2401.08825*.

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, et al. 2024. The ability of chatgpt in paraphrasing texts and reducing plagiarism: a descriptive analysis. *JMIR Medical Education*, 10(1):e53308.

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*.

Oana Ignat, Xiaomeng Xu, and Rada Mihalcea. 2024. Maide-up: Multilingual deception detection of gpt-generated hotel reviews. *arXiv preprint arXiv:2404.12938*.

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.

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.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

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.

Travis J. E. Munyer and Xin Zhong. 2023. Deeptextmark: Deep learning based text watermarking for detection of large language model generated text. *CoRR*, abs/2305.05773.

Jipeng Qiang, Shiyu Zhu, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. 2023. Natural language watermarking via paraphraser-based lexical substitution. *Artif. Intell.*, 317:103859.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. 2023. Can ai-generated text be reliably detected? *CoRR*, abs/2303.11156.

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.

Ergon Cugler de Moraes Silva and Jose Carlos Vaz. 2024. How disinformation and fake news impact public policies?: A review of international literature. *arXiv preprint arXiv:2406.00951*.

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.

Debora Weber-Wulff, Alla Anohina-Naumeca, Sonja Bjelobaba, Tomás Foltýnek, Jean Guerrero-Dib, Olumide Popoola, Petr Sigut, and Lorna Waddington. 2023. Testing of detection tools for ai-generated text. *CoRR*, abs/2306.15666.

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*.

KiYoon Yoo, Wonhyuk Ahn, Jiho Jang, and Nojun Kwak. 2023. Robust multi-bit natural language watermarking through invariant features. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 2092–2115. Association for Computational Linguistics.

Xuan Zhang and Wei Gao. 2023. Towards llm-based fact verification on news claims with a hierarchical step-by-step prompting method. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, IJCNLP 2023 -Volume 1: Long Papers, Nusa Dua, Bali, November 1 - 4, 2023*, pages 996–1011. Association for Computational Linguistics.

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. 2023a. Provable robust watermarking for ai-generated text. *CoRR*, abs/2306.17439.

Xuandong Zhao, Yu-Xiang Wang, and Lei Li. 2023b. Protecting language generation models via invisible watermarking. *CoRR*, abs/2302.03162.

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

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 δ-3.0 exhibited natural and well-structured sentences, while δ-4.0 introduced slight verbosity and rewording. However, δ-5.0 resulted in excessive sentence expansion, leading to unnatural phrasing and reduced readability.

B.1.1 Example (δ-3.0 vs. δ-4.0 vs. δ-5.0)

- δ-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."*

- δ-4.0: *"The proposed model provides an effective approach to reducing computational complexity while ensuring that perfor-*

mance remains competitive with state-of-the-art methodologies. Further assessment on out-of-distribution tasks could help verify its robustness."

- δ-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 δ-3.0 maintained conciseness, δ-4.0 introduced slightly more complex phrasing without significant readability loss, and δ-5.0 contained excessive verbosity, making the review harder to read.

B.2 Logical Coherence and Idea Flow

Logical coherence was largely preserved in δ-3.0 and δ-4.0, but δ-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 (δ-3.0 vs. δ-4.0 vs. δ-5.0)

- δ-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."*

- δ-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."*

- δ-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 δ-3.0 was direct and logically structured, δ-4.0 introduced slight elaboration while maintaining coherence, and δ-5.0 resulted in unnecessary repetition, disrupting logical progression.

B.3 Redundancy and Verbosity

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

B.3.1 Example (δ-3.0 vs. δ-4.0 vs. δ-5.0)

- δ-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."*

- δ-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."*

- δ-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 δ-3.0 was the most precise, δ-4.0 introduced slight elaboration without unnecessary repetition, and δ-5.0 contained inflated and redundant phrasing.

B.4 Technical Terminology and the Blue List Effect

We observed that δ-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 (δ-3.0 vs. δ-4.0 vs. δ-5.0)

- δ-3.0: *"The proposed fine-tuning approach effectively adapts the model to domain-specific*

segmentation tasks, ensuring efficient performance without significantly increasing parameter count."

- δ-4.0: *"The fine-tuning strategy optimizes the model for domain-specific segmentation tasks, maintaining efficiency while minimizing parameter growth."*

- δ-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 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.

E 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]