Knowledge-Infused Multi-Bit Watermarking for RAG Knowledge Bases

Ke Yang[1,2,3], Shuguang Yuan[1,2,3]*, Jing Yu[1,2,3], Chi Chen[1,2,3]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]State Key Laboratory of Cyberspace Security Defense
[3]School of Cyber Security, University of Chinese Academy of Sciences
{yangke, yuanshuguang, yujing, chenchi}@iie.ac.cn

Abstract

Retrieval-Augmented Generation (RAG) enhances the factual accuracy of Large Language Model (LLM) outputs based on external knowledge bases. These knowledge bases often carry significant intellectual property (IP) value, raising the urgent need for robust watermarking techniques to protect IP. However, existing RAG watermarking methods remain in their infancy, facing challenges such as limited encoding capacity and potential degradation of RAG performance or knowledge quality. In this paper, we propose knowledge-infused and multi-bit watermarking (KMW) for RAG knowledge bases. It generates watermark text to infuse the knowledge base by benign knowledge completion and a tailored generative watermarking algorithm. Each generated text can carry a multi-bit watermark segment. For effective detection, we design a Watermark Text Indexer that optimizes queries for steady retrieval of watermarked texts. Experiments on multiple datasets and LLMs show KMW reliably extracts watermarks from adversarial RAGs. It is robust against knowledge selection, alteration, expansion, and RAG setting restrictions, while remaining stealthy and secure. This highlights that KMW ensures effective IP protection for RAG systems. Our code is available here.

1 Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) is an efficient technique that enhances the outputs of Large Language Models (LLMs) by retrieving information from external knowledge bases. It improves the accuracy and relevance of generated content, mitigating hallucinations (Sahoo et al., 2024) and allowing for the integration of real-time, domain-specific information (Sankaradas et al., 2025; Hasan et al., 2025). These knowledge bases, particularly in mission-critical domains (e.g., healthcare (Ng et al., 2025)), contain valuable and private information that requires significant resources to construct and maintain. Their inherent value makes them attractive targets for theft (Wang et al., 2025). Therefore, protecting the intellectual property (IP) of these knowledge bases is essential to safeguard the owner's investment.

Watermarking is a powerful tool for IP protection by embedding watermark information into an asset. While it has achieved success in multimedia (Sahu et al., 2023), databases (Rani and Halder, 2022), and LLMs (Zhao et al., 2025), existing methods are not directly applicable to RAG. Because these techniques typically require direct access to the watermarked content, while the RAG system only allows black-box querying without accessing the knowledge base. Thus, RAG watermarking must support detection from LLM's outputs. However, achieving this is non-trivial due to challenges such as (i) the limited retrievability of watermarked content; (ii) the vulnerability of watermark content to post-processing, e.g., abstraction or rewriting; and (iii) constraints imposed by system prompts.

The first RAG watermarking method, WARD (Jovanović et al., 2025), employs a rewriting-based strategy that embeds watermarks by paraphrasing IP data using LLM watermarking (Kirchenbauer et al., 2023). While effective, paraphrasing compromises the quality and accuracy of the original content. To avoid this, subsequent methods (Lv et al., 2025; Guo et al., 2025; Liu et al., 2025b) inject LLM-generated content as watermarks into the knowledge base rather than altering existing data. Both paradigms then share a common detection mechanism: they construct queries targeted at watermarked content, probe the suspect RAG system, and analyze watermark evidence in responses.

Although this injection-based strategy preserves the original data, it may impair RAG performance if the injected content is incorrect or implausible. For instance, the method in (Lv et al., 2025) may produce implausible sentences such as "Spain con-

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2026, pages 21195–21218

tains Serum Ldl.", and the approach in (Liu et al., 2025b) can yield factually incorrect statements like "Terminus, a planet in the solar system...". Furthermore, current methods encode only 1 bit of information–whether the RAG is watermarked or not, lacking the capacity to embed richer metadata such as copyright details. This limitation constrains the practical applicability of RAG watermarking.

To tackle these issues, we propose **K**nowledge-infused **M**ulti-bit **W**atermarking (KMW), a scheme that reframes watermarking as a benign knowledge-base completion task. Instead of rewriting content or injecting low-quality text, KMW generates context-consistent texts by knowledge completion with factual relationships, which encodes a multi-bit watermark. Moreover, the watermarking process maintains unbiased generation. This enables generated texts to maintain excellent consistency with the knowledge base and high intrinsic quality. By infusing them into the knowledge base, KMW simultaneously secures IP and preserves RAG utility. For reliable detection, we design a *Watermark Text Indexer* that optimizes queries to retrieve the watermarked content. The watermark is then extracted from the RAG's responses via majority voting. Experimental results show that KMW achieves a watermark recovery rate of 1.00 across various datasets and RAG configurations. The watermark texts surpass the baselines in quality and stealthiness, achieving minimal negative impact on the main RAG tasks (1.2%) and the lowest filtering rate (less than 6.23%). Besides, KMW exhibits robustness against a wide range of attacks. Our contributions are summarized below:

- Our proposed KMW leverages knowledge completion to generate high-quality, benign watermarked text. This lessens the trade-off between IP protection and RAG utility.
- We design an unbiased, multi-bit watermarking algorithm and a *Watermark Text Indexer* for the RAG scenario. The former ensures high generation quality and capacity, while the latter optimizes query construction for robust retrieval.
- Experiments across four datasets and diverse RAG configurations indicate KMW's superior effectiveness, stealthiness, and robustness compared to state-of-the-art baselines.

2 Problem Modeling

2.1 Threat Model

This work considers RAG watermarking for IP protection, where the *defender* owns proprietary knowledge and embeds watermarks to safeguard a RAG system built upon it. *Adversaries* aim to steal the knowledge base and either deploy it within their own RAG or distribute it for profit.

Defender ability: They can only query and receive responses from a RAG system. They detect watermarks if a suspect RAG demonstrates anomalously high performance on their proprietary content.

Adversary ability: They may be aware of the watermark's presence, but lacking the secret key prevents precise watermark removal. However, they may attempt to modify the knowledge base or RAG settings to remove or forge the watermark. We categorize such threats into *Knowledge Selection/Alteration/Expansion Attacks*, *RAG Setting Restrictions*, and *Reclaiming Attacks* (see Appendix A).

2.2 Capabilities of RAG Watermarking

We claim that RAG watermarking for IP protection must satisfy: (i) Effectiveness: The watermark can be reliably extracted from a pirated RAG; (ii) Security: The watermark is computationally infeasible to be covered or forged without a secret key; (iii) Stealthiness: The embedded watermark is imperceptible and resistant to filtering; (iv) Soundness: Any innocent RAG has a negligible chance of being detected as watermarked; (v) Fidelity: Watermarking should minimally impact the performance of the RAG system; (vi) Robustness: The watermark remains detectable in the pirated RAG, even when subjected to a range of adversarial attacks.

3 Watermarking Scheme

Within a RAG system, the knowledge base KB is optimal for watermarking as it is the main target for data theft. Watermarking the retriever or LLM is ineffective, as adversaries can easily substitute them. Thus, our method aims to embed a multi-bit watermark into KB and achieve capabilities. Figure 1 illustrates KMW's four-stage pipeline:

🔗 *Watermark Prefix Construct*: Extracts knowledge graphs from a sample of KB and utilizes knowledge completion to predict relationships, which serve as "prefixes" for watermarking.

📝 *Generation and Infusion*: It controls an unbiased generative watermarking algorithm based on prefixes that encode bits into tokens. They also direct

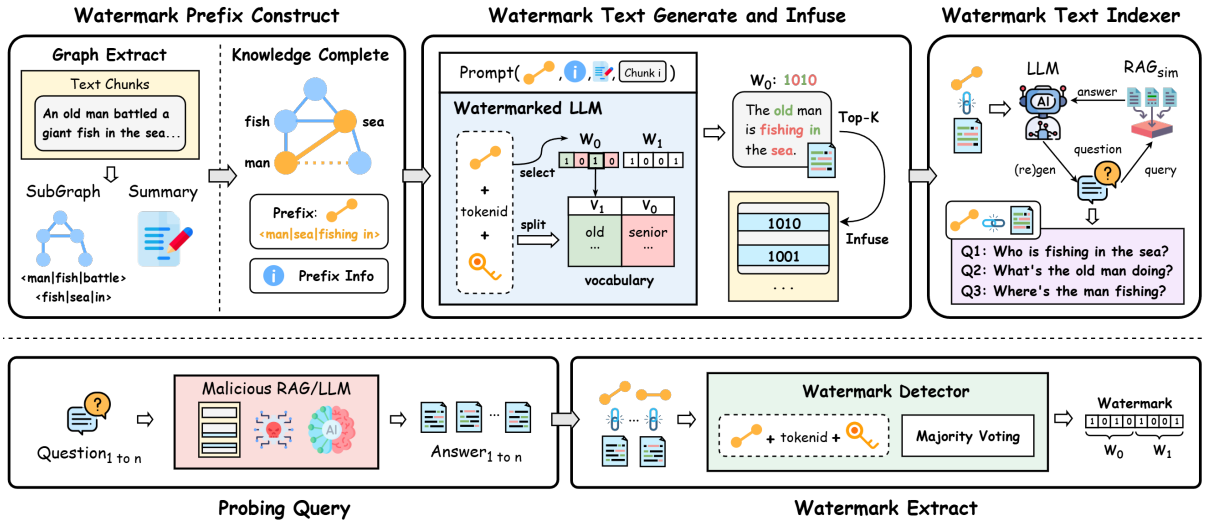

Figure 1: KMW workflow. The top illustrates watermark embedding, and the bottom is watermark extraction.

the generation of context-consistent texts, which are then infused into the knowledge base.

Watermark Text Indexer: Utilizes an LLM and RAG simulator to craft queries with prefixes, enabling reliable watermarked content retrieval.

Query and Extract: Query the suspect RAG and gain the answers. The detector takes these answers with their prefixes, extracts bits from the tokens, and corrects the watermark via majority voting.

3.1 Watermark Prefix Construction

Knowledge Graph Extraction. The construction of watermark prefixes relies on relationship prediction within knowledge graphs. Crucially, this process does not require exhaustive graph extraction. Instead, we sample a subset of text chunks to extract subgraphs, thus avoiding the significant overhead of handling the entire knowledge base.

Given a knowledge base KB to be watermarked, assume it is well organized. Referring to the workflow of GraphRAG (Edge et al., 2024), we first split KB into text chunks, and an advanced LLM is then employed to extract knowledge graphs from each sampled chunk and summarize the graph information. Specifically, a list of the entities $\mathcal{E}$ and relationships $\mathcal{R}$ is extracted from each sampled chunk $\mathcal{C}$. The LLM also generates a summary S for $\mathcal{C}$, offering an executive overview that highlights key entities and relationships. This process can be described as follows:

$$\{\{\mathcal{E}, \mathcal{R}\}, S\} = Graph(Sample(KB)). \quad (1)$$

Knowledge Completion. For each graph, new relationships $\mathcal{R}_p$ are predicted by knowledge completion. We prompt an LLM as a relationship $Predictor$ (as detailed in Appendix D.1) based on the concept of link prediction (Liben-Nowell and Kleinberg, 2003). To enable more accurate and context-consistent predictions, the $Predictor$ is provided with detailed information to predict $\mathcal{R}_p$: the existing relationships $\mathcal{R}$, the entities $\mathcal{E}$, the graph summary S, and the original text content O from the corresponding chunk. The $Predictor$ prioritizes specificity and novelty in $\mathcal{R}_p$. Thus, it performs self-evaluation to filter out any semantically redundant relations, existing relations in the graph, or their simple inverses. Each predicted relationship is concatenated into a watermark prefix HTR using its *Head*, *Tail*, and *Relation type*, while the associated *weight*, *description*, and *reason* serve as auxiliary prefix information r_{aux} to guide watermark knowledge text generation. The process is formalized below:

$$\{HTR, r_{aux}\} = Predictor(\mathcal{R}, \{\mathcal{E}, S, O\}). \quad (2)$$

It is worth noting that these watermark prefixes control our whole RAG watermarking.

3.2 Watermark Text Generation and Infusion

Watermark text generation for RAG consists of content optimization and multi-bit watermarking.

Content Optimization. These watermark prefixes HTRs are converted to watermarked knowledge texts. To ensure the factual correctness and contextual rationality of these texts, we construct a "relationship-to-knowledge" prompt (as detailed in Appendix D.2) based on LLM-augmented KG-to-text techniques (Pan et al., 2024). Specifically, de-

tails (auxiliary information r_{aux} of HTR), high-level abstractions (summary S of the corresponding text chunks), and original structural properties (the relation-related contents O) are applied to guide the LLM in generating knowledge-rich text centered around HTR. This process also self-checks the correctness of the output. Through this optimization pipeline, we significantly enhance the stealthiness of the generated watermarked text.

Multi-bit Generative Watermarking. Watermark persistence is a critical concern in the black-box setting of RAG. By treating the set of watermark texts as a whole, the watermark signal can be accumulated across multiple responses. Intuitively, this accumulation provides enough redundancy to encode customizable watermark information. Motivated by this insight, we design an LLM text generative watermarking algorithm to generate watermark knowledge texts.

Before watermarking, we first shuffle the watermark message to a pseudorandom sequence W for statistical randomness and security. W is then divided into M segments of equal length l, and each generation select one segment W_m by HTR for embedding. This can control the embedding times of a watermark bit, providing redundancy for watermark detection. At each token generation, the watermark bit $w \in \{0, 1\}$ is selected from W_m by:

$$idx(W_m) = \mathrm{H}(HTR, t_{prev}, sk) \bmod |W_m|, \quad (3)$$

where $\mathrm{H}(\cdot)$ denotes a hash function, and idx represents the index of w in W_m.

We then map w into vocabulary partitions for multi-bit embedding. By using $\mathrm{H}(HTR, t_{prev}, sk)$ as a $seed$, we randomly partitions the LLM's vocabulary $\mathcal{V}$ into two disjoint lists $\mathcal{V}_1$ and $\mathcal{V}_0$ at each token generation. $\mathcal{V}_1$ and $\mathcal{V}_0$ represent bit 1 and 0, respectively. The w boosts token probabilities in $\mathcal{V}_w$ for watermarking by a Proportional Modulation on probability distribution:

$$p^w(t) = \Big[1 + (2w-1)\big(\delta \mathbf{1}_{\mathcal{V}_1}(t) - \delta' \mathbf{1}_{\mathcal{V}_0}(t)\big)\Big] p(t), \quad (4)$$

where $p(t)$ denotes the probability of a token t, $\mathbf{1}_{\mathcal{V}_{1,0}}(\cdot)$ is the indicator function for judging whether token $t \in \mathcal{V}_{1,0}$, and δ, δ' represent the modulation factors. The detailed derivation of the Eq. (4) is given in Appendix C.1. To achieve a valid probability distribution after modulation, δ' is constrained by δ and should satisfy:

Theorem 1. $\delta' = \delta \cdot \lambda/(1-\lambda), \lambda = \sum_{t \in \mathcal{V}_1} p(t)$.
Proof. See Appendix C.2.

Through modulation, when $w = 1$, tokens in $\mathcal{V}_1$ have a higher sampling probability, whereas when $w = 0$, tokens in $\mathcal{V}_0$ are more likely to be selected. Each token can encode a bit w, thus achieving multi-bit watermarking. Besides, we have:

Theorem 2. *Proportional modulation is unbiased, preserving generation quality.*
Proof. See Appendix C.3.

We must also consider that the retrieved watermarked text is paraphrased by the LLM component in RAG according to the query. This may disrupt the original watermark tokens (e.g., synonym substitution) and reduce watermark persistence. To mitigate this issue, we follow the approach in WatME (Chen et al., 2024b) by clustering synonyms before vocabulary partitioning. Partitioning and merging operation are applied at the cluster level to obtain the final $\mathcal{V}_1$ and $\mathcal{V}_0$.

The practical settings of the generation process and the generation Algorithm 1 are detailed in Appendix B.1. Notably, our algorithm is flexible, enabling watermark capacity to be adjusted based on robustness and query constraints (e.g., 1-bit mode).

Watermark Text Infusion. After generation, we take the injection-based strategy to complete watermarking. For seamless infusion, we craft a prompt (as detailed in Appendix D.3) to select top-K watermarked texts, and then infuse them into their corresponding chunks. It can be described as:

$$\mathrm{RAG}_{wm} := KB \cup Selected\{KT_{wm}\}. \quad (5)$$

where KB denotes the original knowledge base, $\cup$ represents relevant-text integration, and KT_{wm} are the set of knowledge watermark texts.

3.3 Watermark Text Indexer

To ensure that infused watermarked texts can be effectively retrieved and answered by an RAG, we design a Watermark Text Indexer (WTI) comprising two components: a question generator Q_g and an RAG simulator (RAG_s).

The Q_g prompts (see Appendix D.4) an LLM to produce questions from the watermarked texts and the watermark prefixes. A watermarked text can be used to generate multiple questions from diverse perspectives, enabling a more comprehensive reconstruction of the original content and improving the watermark recovery rate. The generated answers are encouraged to be lexically similar to watermarked texts to facilitate matching.

The RAG_s is based on the watermarked knowledge base and simulates the retrieval process, evaluating the validity of questions. Through iterative interaction between the two components, WTI optimizes the question to index watermarked texts. This process can be formalized as:

$$\{q, HTR\} \leftarrow \text{RAG}_s\big(Q_g(HTR, KT_{wm})\big). \quad (6)$$

3.4 Watermark Detection and Extraction

For suspicious RAG, we query the system with q. If it provides valid answers (means it does not refuse to answer the query), the defender may reasonably infer potential infringement, and watermark extraction can be used as evidence. We extract the watermark using a majority-voting mechanism that accumulates signals across all valid responses.

We first initialize a voting matrix Vot of size $|W| \times 2$, where each row i corresponds to the i-th watermark bit, and the two columns represent votes for extracted bit values 0 and 1. For each valid response ans, we iterate through every token t. Using t, the HTR of q, and the secret key sk, we could reconstruct the corresponding $\mathcal{V}_1$ and $\mathcal{V}_0$, and identify the watermark segment position m with watermark-bit position p. We then increment the vote count $Vot[l \cdot m + p][\mathbf{1}_{\mathcal{V}_1}(t)]$ by 1. After processing all valid responses, the extracted watermark W' is obtained via bit-wise majority voting:

$$W'[i] = \arg\max_{b \in 0,1} Vot[i][b], \quad (7)$$

where $i \in \{0, 1, \cdots, |W| - 1\}$. Appendix B.2, Algorithm 2 exhibits the extraction process.

3.5 Soundness Analysis

We formalize the soundness guarantee using a binomial hypothesis test. Let the Null Hypothesis (H_0) be that a RAG system is unwatermarked. Under H_0, each token's assignment to the "correct" vocabulary partition $\mathcal{V}_w$ is a Bernoulli trial with success probability p. For a watermark bit i, let N_i be the number of valid tokens extracted across all responses to vote for it. The number of successful hits S_i follows a binomial distribution: $S_i \sim B(N_i, p_i)$.

Using the Majority Voting rule, a bit is incorrectly recovered if $S_i \geq \tau_i$, where the threshold $\tau_i = \lfloor N_i/2 \rfloor + 1$. The false-positive probability for a single bit i is: $P_{FP,i} = \Pr(S_i \geq \tau_i) = \sum_{j=\tau_i}^{N_i} \binom{N_i}{j} p_i^j (1 - p_i)^{N_i - j}$. Assuming independence across the $|W|$ watermark bits, the joint probability of falsely claiming ownership over an unwatermarked system is $P_{FP,WM} = \prod_{i=0}^{|W|-1} P_{FP,i}$.

These expressions indicate that the false-positive rate decreases exponentially as the watermark length and extraction counts N_i increase. Consequently, achieving a high watermark recovery rate yields a statistically negligible $P_{FP,WM}$, allowing us to confidently reject H_0 and rigorously prove the soundness of KMW.

4 Evaluation

4.1 Experimental Settings

Datasets & LLMs. We evaluate KMW over NFCorpus (Boteva et al., 2016), TREC-COVID (Voorhees et al., 2021), DROP (Dua et al., 2019), and MS-MARCO (Bajaj et al., 2018). NFCorpus and TREC-COVID contain domain-specific knowledge, making them suitable for evaluating the reliability of KMW on specialized knowledge bases. DROP can evaluate the watermark effects on the main task performance. MS-MARCO is employed to evaluate the effectiveness of watermarking on large-scale knowledge bases. A detailed introduction to the datasets is provided in Appendix E.1. We utilize three mainstream large language models in experiments, including two advanced black-box LLMs (GPT-4.1 (OpenAI, 2025) and Gemini-2.5-Flash (DeepMind, 2025)) and a white-box LLM (Qwen2.5-32B-Instruct (Team, 2024)).

Baselines. We compare KMW with two formally accepted approaches, WARD (Jovanović et al., 2025) and RAG-WM (Lv et al., 2025), and additionally include KMW in 1-bit mode (KMW(1)) and a non-watermarked RAG (ORI) as baselines.

Metrics. Three categories of metrics are utilized:

🎯 *Validity:* Watermark Recovery Rate (WRR), Watermark Text Retrieval Rate (WTRR), and watermark bit False-Hit Rate (FHR) are used to evaluate the effectiveness and soundness of KMW itself.

💎 *Quality:* Average Perplexity (PPL_{avg}), Knowledge Correct Rate (KCR), and Average contextual Rationality Score (RS_{avg}) of infused watermark texts are applied to reflect KMW's generation quality. We also conduct human evaluation to assess the text fluency, correctness, and rationality.

📈 *Performance:* Clean Data Performance Alignment (CDPA) and Clean Information Retrieval Alignment (CIRA) (Lv et al., 2025) assess the impact of watermarking on the original RAG main task performance. Watermarked Data Performance Verification (WDPV) is proposed to evalu-

ate whether KMW achieves knowledge completion on the knowledge bases.

We evaluate KCR, RS_{avg}, CDPA, and WDPV by Gemini. And the corresponding prompts are provided in Appendix E.2. See Appendix E.3 for more details of these metrics.

Implementation Details. We sampled approximately 2,000 texts from the knowledge base to generate watermark prefixes. Qwen2.5 with our generative watermarking algorithm is utilized to generate watermarked texts. The algorithm embeds a 24-bit payload, which is shuffled via a pseudorandom string to form watermark W and divided into six 4-bit segments. KMW infuses 600 texts into the knowledge base, whereas WARD and RAG-WM use their default watermarking amount (200). KMW(1) likewise infuses 200 texts. WTI generates 3 questions for each watermarked text. Our primary retriever is BGE-M3 (Chen et al., 2024a) (top-$K = 1$, cosine similarity). We also evaluated Contriever (Izacard et al., 2022) and ANCE (Xiong et al., 2020) using Euclidean distance and inner product. See Appendices E.4 and E.7 for more details and generation examples.

4.2 Effectiveness Evaluation

Results of watermark verification, knowledge generation quality, and impact on the main RAG task are shown in Table 1.

Watermark Verification. We apply KMW to datasets and simulate pirated RAG integrating the aforementioned LLMs. We then query each RAG using all questions (approximately 1,800) generated by WTI. Experimental results in Table 1 demonstrate the robust watermark verification of KMW across diverse LLMs and datasets. WRR consistently reaches 1.00, confirming the successful embedding and complete recovery of multi-bit watermarks. A high WTRR, $\approx$1.00, further validates the effectiveness of WTI. For soundness, FHR is around 0.5 because when detection is performed on a clean RAG or with an incorrect key, a pseudorandom string is output, which theoretically hits about half the bits of W. Moreover, we evaluate the query quotas required for reliable detection across different schemes (see Appendix E.5). We observed that KMW(1) requires fewer queries than both WARD and RAG-WM, indicating the efficiency of our watermark verification.

Watermarked Knowledge Quality. The evaluation of watermarked text quality across different methods is reported in Table 1. From the results, our method maintains stable text quality regardless of the number of generated texts, with KMW(1) and KMW performing similarly. Besides, our method achieves the lowest PPL_{avg} compared to WARD and RAG-WM, indicating that KMW generates high-quality watermark texts. In contrast, RAG-WM yields the highest PPL due to implausible entity-relation combinations, resulting in absurd watermark texts (see Appendix E.8 for examples). Table 2 reports PPL_{avg} of the original datasets. Compared with Table 1, this further confirms the high quality of texts generated by KMW.

To evaluate KCR and RS_{avg}, we utilize watermark texts to retrieve relevant texts from ORI. An LLM, based on retrieved texts, then evaluates the correctness and scores the rationality (range in [-10,10]) of each watermark text. WARD attains high KCR and RS_{avg} via the rewriting-based strategy. However, as indicated by the PPL_{avg} results in Table 2, this approach compromises textual fidelity and increases perplexity. Under the injection-based strategy, watermarked texts do not alter the original knowledge, and compared with RAG-WM, KMW clearly achieves much better factuality and rationality (KCR≈ 0.75, $RS_{avg}\approx 8$).

We further validated quality through human evaluation (See Appendix E.6 for the details of the evaluation and results). Notably, the watermark texts received slightly higher average scores than the clean texts. This finding aligns with the PPL_{avg}, confirming the high generation quality of KMW.

Impact on Main Task. CIRA and CDPA respectively measure the consistency of retrieved texts and generated responses for clean queries before and after watermarking. We observe from Table 1 that KMW achieves the highest CDPA and CIRA, indicating that the infused knowledge has a negligible impact on the original task performance. In contrast, WARD significantly degrades task performance (CDPA and CIRA drop to 0.87 and 0.83, respectively). Furthermore, KMW achieves WDPV near 1.00, far surpassing the baselines ($\approx$0.60). This demonstrates that the knowledge texts infused by KMW effectively complete the knowledge base, making some implicit information explicit.

Overhead. The time overhead for sampling 2,000 original texts for watermark prefix construction and linearly generating 600 watermark texts is shown in Table 3. Besides, the API (gpt-4.1) cost is around US$30. This is acceptable for the owner because KMW is once-for-all and offline.

Table 1: Result of Effectiveness Evaluation across Datasets, LLMs, and Baselines.

Dataset	Validity Metrics	LLM			Quality Metrics	Scheme				Task Metrics	Scheme			
		GPT	Gemini	Qwen		WARD	RAGWM	KMW(1)	KMW		ORI	WARD	RAGWM	KMW
NFCorpus	WRR	1.00	1.00	1.00	PPL_{avg}↓	16.60	221.41	11.23	**10.67**	CDPA↑	1.00	0.87	0.96	**0.97**
	WTRR	1.00	1.00	1.00	KCR↑	**1.00**	0.22	0.83	0.79	CIRA↑	1.00	0.83	0.88	**0.98**
	FHR	0.52	0.48	0.55	RS_{avg}↑	**9.50**	-2.31	8.62	8.51	WDPV↑	0.64	0.63	0.64	**0.99**
TREC-COVID	WRR	1.00	1.00	1.00	PPL_{avg}↓	17.52	144.90	12.01	**11.71**	CDPA↑	1.00	0.91	0.98	**0.99**
	WTRR	1.00	1.00	0.99	KCR↑	**0.97**	0.16	0.86	0.88	CIRA↑	1.00	0.92	0.96	**0.97**
	FHR	0.53	0.44	0.49	RS_{avg}↑	**9.44**	-1.87	8.42	8.99	WDPV↑	0.63	0.61	0.62	**0.99**
DROP	WRR	1.00	1.00	1.00	PPL_{avg}↓	14.47	24.90	**12.37**	12.84	CDPA↑	1.00	0.96	0.98	**1.00**
	WTRR	0.99	0.99	0.99	KCR↑	**0.93**	0.07	0.74	0.68	CIRA↑	1.00	0.94	0.93	**0.99**
	FHR	0.49	0.46	0.53	RS_{avg}↑	**9.53**	-7.95	8.11	7.72	WDPV↑	0.60	0.59	0.62	**0.97**
MS-MARCO	WRR	1.00	1.00	1.00	PPL_{avg}↓	24.90	21.98	**15.99**	16.05	CDPA↑	1.00	0.97	0.97	**0.99**
	WTRR	0.99	1.00	0.99	KCR↑	**0.88**	0.11	0.79	0.77	CIRA↑	1.00	0.95	0.98	**1.00**
	FHR	0.51	0.50	0.52	RS_{avg}↑	**9.31**	-5.34	8.48	8.52	WDPV↑	0.58	0.58	0.57	**1.00**

Table 2: Average PPL of original datasets.

Metrics	NFcorpus	TREC-COVID	DROP	MS-MARCO
PPL_{avg}	12.08	14.52	11.46	23.85

Table 3: Time Overhead of Watermarking.

Metrics	NFcorpus	COVID	DROP	MARCO
Prefix/Text (hour)	2.6/4.3	3.2/4.1	2.9/4.6	2.8/4.5

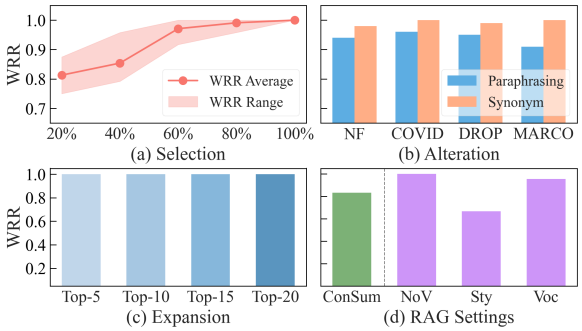

Figure 2: Results of Robustness Evaluation.

4.3 Robustness Evaluation

Unless otherwise specified, the evaluation is carried out on watermarked NFCorpus.

Knowledge Selection Attacks. We simulate two selection attack strategies. First, under random subset sampling, where an adversary uses only a fraction of the stolen KB, WRR remains high according to Figure 2 (a). Even with only 20% of the KB, the WRR exceeds 0.8.

Second, we use the prompts in Appendix E.9 to filter irrelevant content during retrieval for watermark removal. Results in Appendix E.9, with the maximum filtering rate of 2.19%, indicating that this strategy is ineffective. These results show that KMW is robust against selection attacks.

Knowledge Alteration Attacks. Adversaries may attempt paraphrasing attacks (Kirchenbauer et al., 2024) to the retrieved texts for watermark removal. We simulate it under sentence embedding preservation using GPT-4.1 with the prompt "paraphrase this paragraph." As shown in Figure 2 (b), this attack causes only a minor 5–10% drop in the WRR.

Besides, we perform a synonym substitution attack on the responses, the results are shown in Appendix E.9, with WRR close to 1.00. This indicates the effectiveness of synonym clustering. The results of these two attacks demonstrate KMW's robustness against content modification.

Knowledge Expansion Attacks. An adversary may attempt to inject external content into the stolen knowledge base to dilute the watermark. We inject 20% to 100% new content to simulate this. Results are shown in Appendix E.9 with all cases of WRR equal to 1.00.

Additionally, retrieving more unwatermarked documents during LLM generation (by setting top-K to 5, 10, 15, and 20) also failed to reduce WRR (Figure 2(c)). This proves the reliability of WTI and our watermark detector.

RAG Setting Restrictions. Adversarial RAG may employ complex settings, so we evaluate KMW under context summarization and stronger system prompts (including no verbatim use of retrieved texts, style control, and vocabulary constraints). Figure 2(d) shows that no verbatim reuse and imposing vocabulary constraints have negligible impact on watermark detection. Context summarization reduces WRR to 0.83, while style-control

prompts cause a larger drop (WRR = 0.66), though still above FHR. In Appendix E.9, we reduce watermark capacity and re-test context summarization and style control, obtaining higher WRR values of 0.93 and 0.87, respectively, indicating that KMW can trade capacity for robustness when needed.

4.4 Stealthiness Evaluation

An adversary may attempt to remove obvious watermarked texts from the knowledge base.

Duplicate Content Removal. We set top-K to 20 for contextual retrieval and use SimHash (Charikar, 2002) to facilitate deduplication. As shown in Table 4, only a small portion of watermark texts (up to 6.23%) are filtered out. This indicates that our infused knowledge texts are distinct and not a duplicate of the original content.

Table 4: Duplicate Content Removal Attack.

Metrics	NFcorpus	COVID	DROP	MARCO
Filtering (%)	0.37	0.48	6.23	0.54

Text Perplexity Filtering. Considering that watermarking may degrade text quality, an adversary might filter low-quality retrieved content. Using the original documents' PPL distribution, we set $\mu + 3\sigma$ as a filtering threshold. As shown in Table 5, only 0.18% of KMW's watermarked texts were filtered, even fewer than the original, showing our high-quality generation. In contrast, 91.06% of RAG-WM's texts were filtered, exposing the weakness of its entity-relation combinations.

Table 5: PPL Filter.

Metrics	ORI	WARD	RAG-WM	KMW
Filtering Rate (%)	1.24	3.17	91.06	**0.18**

Human Identification. We construct multiple-choice questions with four texts per item: one watermarked text and three clean texts. Participants are asked to identify the watermarked text among the four. The correct identification rate is 21%, slightly below random chance (25%), indicating that watermark texts are imperceptible to humans.

4.5 Impact of Parameters

Impact of WTI and Query Quota. To show that WTI improves watermark-text retrieval and to quantify the effect of query quotas on detection, we query the RAG 50, 100, 200, 300, 400, and 500 times using either WTI-generated queries or directly constructed questions. The results in Figure 3(a) show that WRR exceeds 0.9 with 200 queries. Moreover, WTI consistently increases WTRR and yields higher WRR under low query quotas, validating its effectiveness.

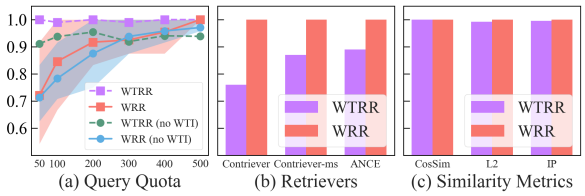

Figure 3: Results of Parameters Impact.

Impact of Retriever. We replace the retriever BGE-m3 with Contriever, Contriever-ms, and ANCE, and evaluate WRR and WTRR. As shown in Figure 3(b), WTRR drops lowest with Contriever (0.76), but WRR remains stable, indicating KMW's robustness to retriever variations.

Impact of Similarity Metrics. We assess the impact of different similarity metrics (CosSim, IP, and L2) during retrieval on watermark detection. We can observe from Figure 3 (c) that similarity metrics hardly affect the retrieval of watermarked text and the extraction of the watermark.

Combined Impact of Watermark Capacity, Infusion Volume, and Query Quota. To comprehensively verify the capabilities of KMW, we analyze the joint impact of operational parameters: watermark capacity, infusion volume, and query quota.

We evaluate the WRR across various watermark capacities and text infusion volumes under alteration, summarization, and expansion attacks (as illustrated in Figure 4). The results indicate that watermark capacity is inversely proportional to KMW's robustness. While KMW resists expansion attacks across all capacities, its WRR degrades under alteration and summarization attacks when larger capacities are employed. This occurs because embedding more bits reduces the redundancy (embedding times) per bit, increasing susceptibility to perturbations. Furthermore, we observe that the volume of infused texts has a minimal impact on overall robustness. A detailed theoretical analysis of this phenomenon is provided in Appendix E.10.

We then investigate the joint impact of watermark capacity and query quota on the WRR, as shown in Figure 5. The findings demonstrate that both parameters dictate detection performance. In the most challenging setting: a 32-bit watermark ca-

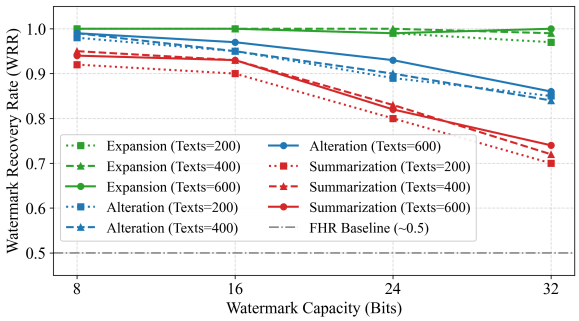

Figure 4: Results of Watermark Capacity and Infusion Volume Impact under Attacks.

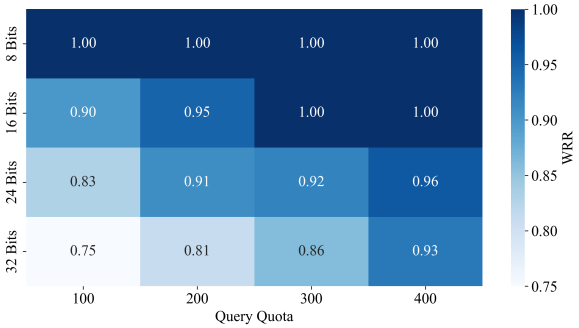

Figure 5: Results of Query Quota and Watermark Capacity Impact.

pacity paired with a restrictive quota of 100 queries, the WRR drops to 0.75. Consequently, achieving highly robust detection performance requires a strategic balance, either by reducing the watermark capacity or by increasing the allocated query quota.

Finally, we evaluate the combined effect of query quota and infusion volume (Appendix E.10 Figure 7). The results closely mirror Figure 3(a), confirming infusion volume has minor impact. Overall, these results validate that watermark capacity and query quota primarily dictate KMW's performance.

4.6 Security Discussion

A threat to IP protection in RAG systems is ownership reclaiming, where an adversary injects their own watermark into the RAG to falsely claim ownership. Both existing schemes and KMW can counter this attack as the true owner can reliably detect their watermark in the adversary's RAG, but the adversary cannot detect their watermark in the owner's RAG and cannot provide a knowledge base without the onwer's watermark.

However, insecure designs may allow advanced adversaries to claim indistinguishable ownership. Injection-based strategies are particularly susceptible, as they do not modify original content–if the injection process is not computationally infeasible, adversaries can forge a plausible watermark using existing content. Since the same content also exists in the owner's dataset, it becomes impossible to determine the watermark's origin.

RAG-WM suffers from this issue, as detailed in Appendix E.11. In contrast, KMW resists such forgery attacks, as both watermark selection and text generation rely on a series of keyed hash operations, and the watermark is meaningful. Without the secret key, it is computationally infeasible for an adversary to construct a colliding hash function that generates an identically valid watermark information. This design ensures the security of KMW. We formally analyzed our defense against both reclaiming and forgery in Appendix E.12.

5 Related Work

With the application of RAG, its IP protection has attracted increasing attention. The idea of RAG watermarking has been proposed (Jovanović et al., 2025) and preliminarily explored. Existing RAG watermarking can be categorized into rewrite-based (Jovanović et al., 2025) and injection-based (Lv et al., 2025; Guo et al., 2025; Liu et al., 2025b,a). Besides, watermarking for multimodal RAG systems has also begun to be studied (Chen et al., 2025). However, despite these advancements, existing research lacks sufficient exploration into enhancing watermarking capacity and preserving RAG utility. We refer readers to Appendix F for a detailed background and related work.

6 Conclusion

In this study, we introduced Knowledge-Infused Multi-Bit Watermarking (KMW) for Retrieval-Augmented Generation (RAG) to protect intellectual property (IP). KMW establishes a new RAG watermarking paradigm by addressing limitations in encoding capacity and potential degradation in RAG performance, enhancing the applicability of RAG watermarking. Experimental results across four datasets and diverse RAG configurations demonstrate the effectiveness, robustness, stealthiness, and security of KMW.

Limitations

In this section, we discuss several limitations of this work. First, implementing multi-bit watermarking requires higher watermark text injection amounts and query quotas, resulting in higher computational

overhead and API costs for both watermarking and retrieval processes.

Second, regarding the generation scheme, while we achieve unbiased token selection, the inherent non-uniformity of word frequencies in natural language leads to uneven proportional modulation across different tokens. This introduces a slight statistical bias in the generated text. Achieving global unbiasedness would require uniform word frequencies for watermarked tokens, further compromising watermark persistence.

Finally, style transfer induced by system prompts was observed to attenuate watermark signals. Currently, we address this by reducing watermark capacity to maintain robustness. Mitigating this trade-off without sacrificing capacity is a priority for future research. Additionally, the feasibility of the framework under more complex system prompt constraints and advanced RAG paradigms remains to be fully explored.

Despite these limitations, we believe that our work can serve as an important catalyst in this field, contributing positively to the advancement of RAG watermarking techniques with higher watermark capacity and lower performance impact.

Acknowledgments

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences, Grant No. XDB0690303.

References

Scott Aaronson. 2023. Watermarking of large language models. https://simons.berkeley.edu/talks/scott-aaronson-ut-austin-openai-2023-08-17. Accessed: 2025-07-02.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. Ms marco: A human generated machine reading comprehension dataset. *Preprint*, arXiv:1611.09268.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Proceedings of the 38th European Conference on Information Retrieval*.

Brian J. Chan, Chao-Ting Chen, Jui-Hung Cheng, and Hen-Hsen Huang. 2025. Don't do rag: When cache-augmented generation is all you need for knowledge tasks. In *Companion Proceedings of the ACM on Web Conference 2025*, WWW '25, page 893–897. ACM.

Moses S Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024a. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Liang Chen, Yatao Bian, Yang Deng, Deng Cai, Shuaiyi Li, Peilin Zhao, and Kam-Fai Wong. 2024b. WatME: Towards lossless watermarking through lexical redundancy. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9166–9180.

Tianyu Chen, Jian Lou, and Wenjie Wang. 2025. Safeguarding multimodal knowledge copyright in the rag-as-a-service environment. *Preprint*, arXiv:2506.10030.

Wenhu Chen, Hexiang Hu, Xi Chen, Pat Verga, and William W. Cohen. 2022. Murag: Multimodal retrieval-augmented generator for open question answering over images and text. *Preprint*, arXiv:2210.02928.

Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. 2023. Incorporating structured sentences with time-enhanced bert for fully-inductive temporal relation prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 889–899.

Mingyue Cheng, Yucong Luo, Jie Ouyang, Qi Liu, Huijie Liu, Li Li, Shuo Yu, Bohou Zhang, Jiawei Cao, Jie Ma, Daoyu Wang, and Enhong Chen. 2025. A survey on knowledge-oriented retrieval-augmented generation. *Preprint*, arXiv:2503.10677.

Aloni Cohen, Alexander Hoover, and Gabe Schoenbach. 2025. Watermarking language models for many adaptive users. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 2583–2601. IEEE.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, and 1 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823.

Google DeepMind. 2025. Gemini 2.5 flash. https://deepmind.google/models/gemini/flash/. Accessed: 2025-07-15.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proc. of NAACL*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Jaiden Fairoze, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Mingyuan Wang. 2023. Publicly-detectable watermarking for language models. *arXiv preprint arXiv:2310.18491*.

Xiaoyan Feng, He Zhang, Yanjun Zhang, Leo Yu Zhang, and Shirui Pan. 2025. Bimark: Unbiased multilayer watermarking for large language models. *arXiv preprint arXiv:2506.21602*.

Junfeng Guo, Yiming Li, Ruibo Chen, Yihan Wu, Chenxi Liu, Yanshuo Chen, and Heng Huang. 2025. Towards copyright protection for knowledge bases of retrieval-augmented language models via reasoning. *Preprint*, arXiv:2502.10440.

Md Toufique Hasan, Muhammad Waseem, Kai-Kristian Kemell, Ayman Asad Khan, Mika Saari, and Pekka Abrahamsson. 2025. Engineering rag systems for real-world applications: Design, development, and evaluation. *Preprint*, arXiv:2506.20869.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4115–4129. Association for Computational Linguistics.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2025. GRAG: Graph retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4145–4157. Association for Computational Linguistics.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. 2023. Unbiased watermark for large language models. *Preprint*, arXiv:2310.10669.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Preprint*, arXiv:2112.09118.

Nikola Jovanović, Robin Staab, Maximilian Baader, and Martin Vechev. 2025. Ward: Provable rag dataset inference via llm watermarks. *Preprint*, arXiv:2410.03537.

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*, pages 17061–17084. PMLR.

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

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

Abhijeet Kumar, Abhishek Pandey, Rohit Gadia, and Mridul Mishra. 2020. Building knowledge graph using pre-trained language model for learning entity-aware relationships. In *2020 IEEE international conference on computing, power and communication technologies (GUCON)*, pages 310–315. IEEE.

Myeonghwa Lee, Seonho An, and Min-Soo Kim. 2024. Planrag: A plan-then-retrieval augmented generation for generative large language models as decision makers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6537–6555.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559.

Peiyang Liu, Ziqiang Cui, Di Liang, and Wei Ye. 2025a. Who stole your data? a method for detecting unauthorized rag theft. *Preprint*, arXiv:2510.07728.

Yepeng Liu, Xuandong Zhao, Dawn Song, and Yuheng Bu. 2025b. Dataset protection via watermarked canaries in retrieval-augmented llms. *Preprint*, arXiv:2502.10673.

Peizhuo Lv, Mengjie Sun, Hao Wang, Xiaofeng Wang, Shengzhi Zhang, Yuxuan Chen, Kai Chen, and Limin Sun. 2025. Rag-wm: An efficient black-box watermarking approach for retrieval-augmented generation of large language models. *Preprint*, arXiv:2501.05249.

Karen Ka Yan Ng, Izuki Matsuba, and Peter Chengming Zhang. 2025. Rag in health care: a novel framework for improving communication and decision-making by addressing llm limitations. *NEJM AI*, 2(1):AIra2400380.

OpenAI. 2025. Introducing gpt-4.1 in the api. `https://openai.com/index/gpt-4-1/`. Accessed: 2025-07-15.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. 2025. Memorag: Boosting long context processing with global memory-enhanced retrieval augmentation. *Preprint*, arXiv:2409.05591.

Wenjie Qu, Wengrui Zheng, Tianyang Tao, Dong Yin, Yanze Jiang, Zhihua Tian, Wei Zou, Jinyuan Jia, and Jiaheng Zhang. 2025. Provably robust multi-bit watermarking for {AI-generated} text. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 201–220.

Sapana Rani and Raju Halder. 2022. Comparative analysis of relational database watermarking techniques: An empirical study. *IEEE Access*, 10:27970–27989.

Mohammad Reza Rezaei, Maziar Hafezi, Amit Satpathy, Lovell Hodge, and Ebrahim Pourjafari. 2024. At-rag: An adaptive rag model enhancing query efficiency with topic filtering and iterative reasoning. *Preprint*, arXiv:2410.12886.

Pranab Sahoo, Prabhash Meharia, Akash Ghosh, Sriparna Saha, Vinija Jain, and Aman Chadha. 2024. A comprehensive survey of hallucination in large language, image, video and audio foundation models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11709–11724. Association for Computational Linguistics.

Aditya Kumar Sahu, Krishnan Umachandran, Vaishali D Biradar, Olebara Comfort, V Sri Vigna Hema, Frank Odimegwu, and MA Saifullah. 2023. A study on content tampering in multimedia watermarking. *SN Computer Science*, 4(3):222.

Murugan Sankaradas, Ravi K. Rajendran, and Srimat T. Chakradhar. 2025. Streamingrag: Real-time contextual retrieval and generation framework. *Preprint*, arXiv:2501.14101.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Sourav Verma. 2024. Contextual compression in retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2409.13385.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*.

Yuhao Wang, Wenjie Qu, Yanze Jiang, Zichen Liu, Yue Liu, Shengfang Zhai, Yinpeng Dong, and Jiaheng Zhang. 2025. Silent leaks: Implicit knowledge extraction attack on rag systems through benign queries. *Preprint*, arXiv:2505.15420.

Xin Xie, Ningyu Zhang, Zhoubo Li, Shumin Deng, Hui Chen, Feiyu Xiong, Mosha Chen, and Huajun Chen. 2022. From discrimination to generation: Knowledge graph completion with generative transformer. In *Companion Proceedings of the Web Conference 2022*, pages 162–165.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *Preprint*, arXiv:2007.00808.

KiYoon Yoo, Wonhyuk Ahn, and Nojun Kwak. 2024. Advancing beyond identification: Multi-bit watermark for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4031–4055.

Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, and Maosong Sun. 2025. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. *Preprint*, arXiv:2410.10594.

Shuguang Yuan, Chi Chen, Ke Yang, Tengfei Yang, and Jing Yu. 2022. An attribute-attack-proof watermarking technique for relational database. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1136–1143. IEEE.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

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

Xuandong Zhao, Sam Gunn, Miranda Christ, Jaiden Fairoze, Andres Fabrega, Nicholas Carlini, Sanjam Garg, Sanghyun Hong, Milad Nasr, Florian Tramer, and 1 others. 2025. Sok: Watermarking for ai-generated content. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 2621–2639. IEEE.

A Attacks on RAG Watermarking

Following prior work(Yuan et al., 2022; Kirchenbauer et al., 2023; Lv et al., 2025), we categorize potential attacks as follows:

Knowledge Selection Attacks: Adversaries attempt to reduce watermark effectiveness by using parts of the knowledge base to dilute watermark density; (i) they can randomly select a subset of the

knowledge base for deployment. (ii) Given that watermarking may introduce some distortion to RAG, they may remove irrelevant, redundant, or low-quality content to eliminate potential watermark-bearing segments.

Knowledge Alteration Attacks: Adversaries apply (i) Paraphrasing Attacks (Kirchenbauer et al., 2023) to paraphrase the retrieved texts to distort the watermark signal and evade detection. Besides, they can (ii) substitute the synonyms in the retrieved content to compromise the watermark detection process.

Knowledge Expansion Attacks: Adversaries can (i) expand the knowledge base with external or misleading content to interfere with watermark-triggered retrieval. Besides, they can just (ii) increase the proportion of non-watermarked information in the retrieved texts to dilute the watermark signal.

RAG Setting Restrictions: Adversaries may configure RAG systems to optimize or constrain generation. For instance, they might employ context compression techniques (Verma, 2024) to perform extraction, abstraction, or summarization. Alternatively, they could enforce rigid system prompts to regulate style and vocabulary, or explicitly prohibit the verbatim reproduction of retrieved text. Such configurations pose a significant risk of attenuating the watermark signal within the response.

Reclaiming Attacks: Including *Re-Watermarking* and *Watermark Forgery*. The former embeds the adversary's own watermark to falsely claim ownership, while the latter exploits weaknesses in the watermarking scheme to fabricate watermarks from existing content, leading to misattribution of IP. Importantly, a watermarking scheme that fails to resist such attacks cannot offer reliable IP protection.

B Generative Watermarking Algorithm

B.1 Watermark Text Generation Algorithm

Algorithm 1 is the pseudo-code of watermark knowledge text generation. Inputs are the constructed prompt x, a watermark binary string W, a Bernoulli(0.5) pseudorandom sequence B, a secret key sk, a specific watermark prefix HTR, a modification factor δ, and the synonym clusters $\mathcal{C}$ (including the residual cluster for non-synonyms). Line 2 first computes a $seed$ by hashing the combination of sk, the previous token, and HTR. In Line 3-7, the $seed$ globally partitions the clustered vocabulary to construct $\mathcal{V}_1$ and $\mathcal{V}_0$. Line 8 shuffles the W by B for pseudo-randomness and security of original W. Line 9 selects a segment W_m ($m = seed \mod M$) from W for watermarking. Line 10-11 select a watermark bit w from W_m by Eq. 3 that the current token needs to carry. And then Lines 12 generate the watermark token by Eq. 3 according to the w value. After this process, we can get the watermark knowledge text.

Algorithm 1 Watermark Text Generation

Input: prompt x_0, watermark string W, pseudo-random sequence B, secret key sk, watermark prefix HTR, modification factor δ, synonym clusters $\mathcal{C}$ (including the residual cluster for non-synonyms).

Output: watermark text KT_{wm}

1: **for** each generation step s **do**
2: $seed_s \leftarrow H(HTR, prev\_token, sk)$.
3: $\mathcal{V}_1 \leftarrow \emptyset,\ \mathcal{V}_0 \leftarrow \emptyset$.
4: **for** each cluster $C_j \in \mathcal{C}$ **do**
5: Randomly partition C_j into $(C_{j,1}, C_{j,0})$ by $seed_s$, where $|C_{j,1}| = |C_j|/2$.
6: $\mathcal{V}_1 \leftarrow \mathcal{V}_1 \cup C_{j,1}$; $\mathcal{V}_0 \leftarrow \mathcal{V}_0 \cup C_{j,0}$.
7: **end for**
8: $W \leftarrow W \oplus B$
9: Select a segment W_m from W by $seed_s$.
10: Select bit index $p \leftarrow idx(W_m)$ by Eq. 3.
11: $w \leftarrow W_m[p] \in \{0, 1\}$.
12: Modify the probability distribution by Eq. 4 to encode w into current token t.
13: **end for**
14: **return** KT_{wm}.

In practice, we XOR a meaningful watermark W with the Bernoulli(0.5) pseudorandom sequence B to obtain the watermark binary string W i.i.d. Bernoulli(0.5) used for watermarking. This can procedure a pseudorandom watermark string, which maintains a uniform token selection from $\mathcal{V}_1$ and $\mathcal{V}_0$ during generation while protecting the confidentiality of the watermark information.

This algorithm is flexible, by setting the segment bit-length to one allows each text to encode a single watermark bit, reducing capacity but enhancing robustness. Or setting the watermark length to one yields a 1-bit scheme, which minimizes capacity but significantly reduces both the required text infusion and the number of queries needed for watermark detection.

B.2 Watermark Extraction Algorithm

Algorithm 2 is the pseudo-code of watermark extraction from suspicious RAG responses. Inputs are a query q, the set of valid responses $\mathcal{A}$ returned by the system, the watermark length $|W|$, the segment length l, a Bernoulli(0.5) pseudorandom sequence B, the secret key sk, the watermark prefix HTR, and the synonym clusters $\mathcal{C}$ (including the residual cluster for non-synonyms). Line 1 initializes a voting matrix $Vot \in \mathbb{N}^{|W|\times 2}$, where each row corresponds to a watermark-bit position and the two columns count votes for extracted bit values 0 and 1. Lines 2–8 iterate over every valid answer ans and each token t within it. Specifically, Line 4 computes the $seed$ by hashing $HTR(q)$, the previous token, and sk, ensuring consistency with the embedding process. Using this $seed$, Line 5 reconstructs the corresponding partitioned vocabularies $(\mathcal{V}_1, \mathcal{V}_0)$ from the synonym clusters $\mathcal{C}$. Line 6 further reconstructs the selected watermark segment W_m and the intra-segment bit index p determined by the same $seed$ as in embedding. In Line 7, the global watermark-bit position is computed as $i = l \cdot m + p$, and the carried bit b_t is identified by whether t falls into $\mathcal{V}_1$. Line 8 then increments the vote count $Vot[i][b_t]$ by 1. After aggregating votes from all valid responses, Lines 9–11 perform bit-wise majority voting to obtain the extracted watermark W', where each bit is set to the value with the larger vote count. Line 12 deshuffles the extracted watermark by XORing with B, i.e., returning $W' \oplus B$, to recover the original watermark string.

Algorithm 2 Watermark Extraction

Input: query q, valid responses $\mathcal{A}$, watermark length $|W|$, segment length l, pseudorandom sequence B, secret key sk, watermark prefix HTR, synonym clusters $\mathcal{C}$ (including the residual cluster for non-synonyms).
Output: extracted watermark W'

1: Initialize voting matrix $Vot \in \mathbb{N}^{|W|\times 2}$ as 0s.
2: **for** each valid answer $ans \in \mathcal{A}$ **do**
3: **for** each token t in ans **do**
4: $seed \leftarrow H(HTR(q), prev\_token, sk)$.
5: Reconstruct $(\mathcal{V}_1, \mathcal{V}_0)$ from $\mathcal{C}$ using $seed$.
6: Reconstruct W_m and bit index p by $seed$.
7: $i \leftarrow l \cdot m + p,\ b_t \leftarrow \mathbf{1}_{\mathcal{V}_1}(t)$.
8: $Vot[i][b_t] \leftarrow Vot[i][b_t] + 1$.
9: **end for**
10: **end for**
11: **for** $i = 0$ to $|W| - 1$ **do**
12: $W'[i] \leftarrow \arg\max_{b\in\{0,1\}} Vot[i][b]$.
13: **end for**
14: **return** $W' \oplus B$.

C Derivation and Proofs

C.1 Proportional Modulation Derivation

At each generation step, the vocabulary $\mathcal{V}$ is partitioned into two disjoint subsets, $\mathcal{V}_1$ and $\mathcal{V}_0$, determined by $seed$. We desire that the watermark bit $w \in \{0, 1\}$ dictates which subset should be favored. Concretely, if $w = 1$, the selection probability of tokens in $\mathcal{V}_1$ should be increased while that of $\mathcal{V}_0$ should be decreased; if $w = 0$, the opposite adjustment is required. Hence, every watermarked token carries exactly one watermark bit.

Let $p(t), t \in \mathcal{V}$ denote the original model distribution over the full vocabulary $\mathcal{V}$. We seek a modulated distribution $p^w(t), t \in \mathcal{V}$ that remains a valid probability distribution while encoding w. A general way to ensure validity is to transfer a certain probability mass ΔP from the non-watermark subset to the watermark subset, preserving total probability.

We define a modulation function $q(t)$ that specifies how each token's probability is adjusted. The watermarked distribution is written as

$$p^w(t) = p(t) + (2w - 1)q(t), \tag{8}$$

where $(2w - 1) \in \{+1, -1\}$ switches the modulation direction depending on the bit value. To preserve normalization, $q(t)$ must satisfy probability conservation:

$$\sum_{t\in\mathcal{V}_1} q(t) = -\sum_{t\in\mathcal{V}_0} q(t), \tag{9}$$

and to avoid negative probabilities we require pointwise boundedness

$$|q(t)| \le p(t), \quad \forall t \in \mathcal{V}. \tag{10}$$

Define the transferred probability mass as

$$\Delta P \triangleq \sum_{t\in\mathcal{V}_1} q(t), \tag{11}$$

which is also $-\sum_{t\in\mathcal{V}_0} q(t)$ by conservation.

To describe how ΔP is distributed within each subset, we introduce two "modulation shapes" $g_1(t)$ on $\mathcal{V}_1$ and $g_0(t)$ on $\mathcal{V}_0$, satisfying

$$\sum_{t\in\mathcal{V}_1} g_1(t) = 1, \qquad \sum_{t\in\mathcal{V}_0} g_0(t) = 1. \tag{12}$$

Then a general construction is

$$q(t) = \mathbf{1}_{\mathcal{V}_1}(t)\Delta P g_1(t) - \mathbf{1}_{\mathcal{V}_0}(t)\Delta P g_0(t), \quad (13)$$

with feasibility constraints

$$\begin{aligned} \Delta P g_1(t) &\le p(t)\ (t \in \mathcal{V}_1), \\ \Delta P g_0(t) &\le p(t)\ (t \in \mathcal{V}_0). \end{aligned} \quad (14)$$

We adopt a simple and intuitive choice, **Proportional Modulation**, which allocates ΔP proportionally to the original probability mass inside each subset:

$$\begin{aligned} g_1(t) &= p(t|t \in \mathcal{V}_1), \\ g_0(t) &= p(t|t \in \mathcal{V}_0). \end{aligned} \quad (15)$$

Substituting into the general form yields

$$\begin{aligned} &q(t) = \\ &\mathbf{1}_{\mathcal{V}_1}(t)\Delta P p(t|t \in \mathcal{V}_1) - \mathbf{1}_{\mathcal{V}_0}(t)\Delta P p(t|t \in \mathcal{V}_0). \end{aligned} \quad (16)$$

Let

$$\lambda \triangleq \sum_{t \in \mathcal{V}_1} p(t) \quad (17)$$

be the total original probability mass of $\mathcal{V}_1$. Then

$$\begin{aligned} p(t|t \in \mathcal{V}_1) &= \frac{p(t)}{\lambda} \quad (t \in \mathcal{V}_1), \\ p(t|t \in \mathcal{V}_0) &= \frac{p(t)}{1-\lambda} \quad (t \in \mathcal{V}_0). \end{aligned} \quad (18)$$

Define the "unit modulation" on $\mathcal{V}_1$ as

$$\delta \triangleq \frac{\Delta P}{\lambda}, \quad (19)$$

and similarly on $\mathcal{V}_0$,

$$\delta' \triangleq \frac{\Delta P}{1-\lambda}. \quad (20)$$

Therefore, proportional modulation can be expressed compactly as

$$q(t) = \big(\mathbf{1}_{\mathcal{V}_1}(t)\delta - \mathbf{1}_{\mathcal{V}_0}(t)\delta'\big)p(t), \quad (21)$$

leading to the final modulated distribution:

$$p^w(t) = \Big[1 + (2w-1)\big(\delta \mathbf{1}_{\mathcal{V}_1}(t) - \delta' \mathbf{1}_{\mathcal{V}_0}(t)\big)\Big]p(t). \quad (22)$$

This form makes explicit that proportional modulation preserves the relative ranking of tokens *within* each subset while shifting mass between subsets to encode the watermark bit.

C.2 Proofs of Theorem 1

A valid distribution requires normalization:

$$\sum_{t \in \mathcal{V}} p^w(t) = 1. \quad (23)$$

Using the modulation form,

$$\sum_{t \in \mathcal{V}} p^w(t) = \sum_{t \in \mathcal{V}} p(t) + (2w-1)\sum_{t \in \mathcal{V}} q(t). \quad (24)$$

Because p is already normalized, $\sum_{\mathcal{V}} p(t) = 1$. Thus we only need

$$\sum_{\mathcal{V}} q(t) = 0 \quad (25)$$

Under proportional modulation,

$$q(t) = \big(\mathbf{1}_{\mathcal{V}_1}(t)\delta - \mathbf{1}_{\mathcal{V}_0}(t)\delta'\big)p(t). \quad (26)$$

Therefore,

$$\begin{aligned} \sum_{t \in \mathcal{V}} q(t) &= \delta \sum_{t \in \mathcal{V}_1} p(t) - \delta' \sum_{t \in \mathcal{V}_0} p(t) \\ &= \delta\lambda - \delta'(1-\lambda). \end{aligned} \quad (27)$$

Setting this equal to zero gives

$$\begin{aligned} &\delta\lambda - \delta'(1-\lambda) = 0 \\ \Rightarrow \quad &\delta' = \delta\frac{\lambda}{1-\lambda}. \end{aligned} \quad (28)$$

Hence the stated relation is necessary and sufficient for probability conservation and thus for a valid modulated distribution.

C.3 Proofs of Theorem 2

We interpret "unbiased" as: averaged over watermark bits, the modulated distribution equals the original distribution. This is a standard requirement ensuring no systematic drift in generation quality when the embedded bits are random. In Algorithm 1, we shuffle the meaningful watermark to a binary string W i.i.d. Bernouli(0.5), which satisfies the standard requirement.

The watermark bits are i.i.d. Bernoulli(0.5). Then

$$\mathbb{E}_w[2w-1] = (+1)\cdot\tfrac{1}{2} + (-1)\cdot\tfrac{1}{2} = 0. \quad (29)$$

Taking expectation of the modulated distribution:

$$\begin{aligned} \mathbb{E}_w[p^w(t)] &= \mathbb{E}_w\big[p(t) + (2w-1)q(t)\big] \\ &= p(t) + \mathbb{E}_w[2w-1]q(t) \\ &= p(t). \end{aligned} \quad (30)$$

Thus proportional modulation introduces no expected change to token probabilities.

Moreover, within each subset $\mathcal{V}_1$ or $\mathcal{V}_0$, proportional modulation scales all token probabilities by the same multiplicative factor:

$$p^w(t) = \begin{cases} (1 + (2w-1)\delta)p(t), & t \in \mathcal{V}_1, \\ (1 - (2w-1)\delta')p(t), & t \in \mathcal{V}_0, \end{cases} \quad (31)$$

so the relative ordering and relative ratios among tokens inside the same subset are unchanged. Hence the local structure of the model distribution (which drives fluency and semantic preference) is preserved, while only a small, controlled mass transfer occurs across subsets to encode the bit.

Combining (i) zero-mean perturbation across bits and (ii) invariance of intra-subset ordering, proportional modulation does not introduce systematic bias and therefore maintains generation quality in expectation.

D Prompt for Watermarking

D.1 Prompt for Knowledge Completion.

Prompt for Relationship Predictor

You are an AI assistant that helps a human analyst to perform **Knowledge Completion** by **Link Prediction**. Link prediction predicts new relationships between existing entities (e.g., (PersonX, livesIn, CityY) AND (CityY, locatedIn, CountryZ) → (PersonX, citizenOf, CountryZ)) within a knowledge graph.
Goal
Strictly following the link prediction rules, predict 2 to 5 new and highly specific relationships from all the given relationships in a given community. Given related information, including a community report, a list of entities belonging to the community, and their original documents, it is used to improve the rationality and accuracy of the predicted relationship. These new relationships will be used to expand the integrity of the community-related knowledge graph and supplement new knowledge text to decision-makers. These relationships need to be reasonable and unique enough (that is, they are genuine and valid, and distinct from the existing relationships in the community).
IMPORTANT NOTE: Since the predicted relationships will be utilized in the generation of knowledge-based texts, these relationships must be inferred from the given relationships rather than directly extracted from the original documents.
Auxiliary Information of Input Structure
The given message contains four parts: community context, entities in community, relationships in community, and original documents in community.
Community Context:
- Title: community's name that represents its key entities.
- Summary: an executive summary of the community's overall structure, how its entities are related to each other, and significant information associated with its entities.
- findings: key insights about the community, containing summaries and explanations. The explanations are supported by entities and relationships of the community, and data references are listed by entities' and relationships' 'id'.
Existing Entities in Community: A list of entities that contains 'id', 'entity', 'type', 'description'
Existing Relationships in Community: A list of relationships that contains 'id', 'source', 'target', 'relationshiptype', 'description', 'weight'.
Original Documents in Community: A set of original documents for the community.
Task & Instructions: 1. **Review the Given Information:** Carefully read through the provided input with the auxiliary information of input structure. 2. **Identify Potential Relationships:** As you review the input, look for interactions, connections, or statements that can infer a relationship between any pair of listed entities that is not already captured. 3. **Prioritize Specificity & Novelty:**
* The 'relationshiptype' MUST be as specific as possible. Avoid generic terms like "RELATED_TO," "ASSOCIATED_WITH," etc., unless no more specific term can be genuinely justified by the text. * **Semantic Check:** Before proposing a 'relationship_type', compare it against the "Existing Relationship Types in this Community."
* If the interaction is accurately and fully described by an existing type, **do not propose it as new.** Your goal is to find *newly identified or more specific uncaptured* links.
* If you propose a new 'relationshiptype', it must be semantically distinct from the existing ones and add clear, new value. It should not be a mere synonym or a inverse.
4. **Formulate Predictions:** For each identified potential new relationship:
* Determine the 'source' entity and 'target' entity from the provided list.
* Define the specific 'relationshiptype'.
* Write a concise 'relationship_description'.
* Determine the 'weight' of the relationship between the source entity and target entity.
* Provide a 'reasoning_for_type_choice', especially explaining its specificity, how to infer this relationship, and why it's considered new or distinct from existing types.
5. **Limit:** Provide inferences for up to 5 new relationships. Focus on the most confident and well-supported ones.
6. **Review the new relationships:** Judge whether the new relationships can be directly extracted from the original documents. If so, do not include them in the output.
Output Format
Output the new relationships with json format that contain keys 'source', 'target', 'relationshiptype', 'description', 'weight', 'reason; and its corresponding values.
Example Input
{Few-shot Here}

D.2 Prompt for Watermark Text Generation.

Prompt for Relation to Text

You are a content creator, a knowledge graph expert, and a linguist who helps a human knowledge creator generate new knowledge texts according to a given relationship and some auxiliary information mentioned below. The relationship consists of 'source', 'target', 'relationship_type', and 'description'. It is a new relationship inferred from a given chunk within a knowledge graph.
Auxiliary Information
- An 'overview' of the cluster where the relationship is located.
- A list of 'original content's that is in the cluster.
- A 'entity_type' and 'descriptions' about the entities in the relationship.
- The 'reason' that led to inferring the relationship.
Task & Instructions Based on the relationship and the auxiliary information, generate a new knowledge text according to the instructions outlined below.
1. Ensure the correctness and accuracy of the generated knowledge text based on the auxiliary information.
2. The generated knowledge text should be a independent element included in the original cluster according to the 'overview' of the cluster.
3. Ensure that the generated knowledge text itself clearly implies or states the relationship between 'source' and 'target', such that its existence could be reasonably inferred by reviewing the generated text within the context of the 'original content' and 'overview'.
4. **Crucially, the generated knowledge text must closely adapt to the average length '{length}' of the 'original content's in the cluster. Aim for a length that is within 10 percentage of the provided average length '{length}'.**
- For example, if the average length is 800 words, your generated texts should be approximately between 720 and 880 words long.

Input
{input_text}
Output
Output a json with the generated knowledge texts. The json with key: 'text', and value with the generated text. Just output only one json, do not output any other text.

D.3 Prompt for Selecting Watermark Text

Prompt for Top-K Selection Watermark Text

You are an expert Subject Matter Analyst and Knowledge Curator. Goal: Your primary objective is to analyze a set of AI-generated texts intended for a specific community and select the top 3 most suitable texts that would add the most value to that community's knowledge base. Your selection must be based on a rigorous evaluation of each text's reasonableness, completeness, and relevance.
The goal is to select the top-3 suitable knowledge texts from the list and inject them into the document.
Input Details: * 'document': A list of base texts of a community.
* 'knowledge_texts': A list of JSON objects, where each object has a 'text' (the knowledge to inject), a 'weight' (a numerical score), and a 'prefix' (a string).
# EVALUATION CRITERIA
You must evaluate each text in the 'knowledge_texts' against the following three criteria:
Reasonableness:
- Is the information presented factually accurate and logically sound?
- Does it avoid making unsubstantiated claims or promoting misinformation?
- Is the tone and language appropriate for the community?
Completeness:
- Is the text complete?
- Does it address the key aspects of the subject matter, or does it feel superficial or incomplete?
- For a given topic, does it offer sufficient detail to be considered a valuable knowledge resource?
Relevance:
- How closely does the text align with the specific topics and themes of the community, as established by the existing community texts?
- Does it introduce a novel yet pertinent perspective, or does it rehash information already present in the community's knowledge base?
- Is this text something the members of this community would genuinely find useful and on-topic?
Steps
1. Read the entire 'document' to understand its content, paragraph structure, and logical flow.
2. Evaluate each text in the 'knowledge_texts' list against the EVALUATION CRITERIA.
3. Synthesize and Rank: After evaluating all the generated texts, create a final ranking. The primary factor for your ranking should be your qualitative evaluation, but you may use the initial 'weight' as a secondary factor in cases where texts are of very similar quality.
4. Identify the top 3 texts from your ranking.
Input
document: {document}
knowledge_texts: {knowledge_texts}
Output
Your final output must be a single, valid **JSON** object. The object should contain one keys: "top_3". The value of "top_3" key is an array of the 'prefix' of your top 3 choices, in descending order of preference.
# Example Output
{{
"top_3": [
"prefix_1",
"prefix_2",
"prefix_3"
]
}}

D.4 Prompt for Question Generation Based on Watermark Text and Watermark Prefix

Prompt for Question Generation

You are a question constructor, a knowledge graph expert, and a linguist. You are skilled at raising unique questions from a given text and a specific relationship.
Task description Paraphrase this text as questions (up to 3) that tackle the content from various perspectives and can only be answered by reading the text. The answer to the question should contain the given relationship and be as lexically similar as possible to the original text, or even directly use the content in the original text. Besides, the answers to these questions, when taken together, should comprehensively cover all the key information in the text. Avoid factual and simple yes/no questions. Do not provide the answer,

provide just the question.
In the last of the question, add a demanding sentence: "Do not answer with point form."
Input details:
- 'text': A text that contains information about a specific relationship.
- 'relationship': A specific relationship in the format of 'source|target|relationship type'.
Input
Text: {text}, Relationship: {relationship}
Output
A list of Json, each json with key 'Question' and value being one of the questions you generated.

E Experiments Settings

E.1 Datasets

NFCorpus is a full-text English dataset tailored for medical information retrieval, comprising 3,633 expert-verified documents primarily sourced from PubMed. It includes 169,756 relevance judgments automatically extracted from 9,964 medical documents, supporting fine-grained evaluation of retrieval models in the biomedical domain.

TREC-COVID is a benchmark dataset designed for information retrieval research on COVID-19 scientific literature. It is based on the CORD-19 collection and contains a set of topics with expert-created relevance judgments, enabling the evaluation of search systems for pandemic response. It contains 1,713,332 texts

DROP is a reading comprehension benchmark designed to evaluate discrete reasoning over paragraphs, which comprises 96,567 adversarially constructed questions. Collected via crowdsourcing, the dataset emphasizes operations such as addition, counting, and sorting, which require systems to resolve multiple references and perform symbolic reasoning. The passages are primarily drawn from sports summaries and historical narratives, demanding a deeper understanding of paragraph-level semantics.

MS MARCO is a large-scale machine reading comprehension dataset consisting of 8,841,823 text passages. It was constructed from anonymized real-world user queries and corresponding web documents retrieved through the Microsoft Bing search engine, making it a representative benchmark for open-domain question answering and passage ranking tasks.

E.2 Metrics Details

Watermark Recovery Rate (WRR): WRR measures the ratio of correctly restored watermark bits to the total binary length of the watermark. The higher the WRR, the higher the confidence of watermark detection to judge IP infringement.

Watermark Text Retrieval Rate (WTRR): WTRR quantifies the proportion of watermark texts that are successfully retrieved. It can be used to measure the effectiveness of query watermark texts.

False Hit Rate (FHR): FHR records the proportion of watermark bits extracted from the unwatermarked RAG to the binary length of the watermark. The lower the FHR, the higher the soundness of watermarking schemes.

Average Perplexity ($\mathbf{PPL_{avg}}$): $\mathrm{PPL_{avg}}$ measures the average quality of texts. We can assume that the lower the perplexity of a text, the higher its quality.

Knowledge Correct Rate (KCR): KCR measures the proportion of factually correct watermarked knowledge. The watermarked knowledge will be used to query a RAG system, and we can get its related context. An LLM judges the correctness of the watermarked knowledge according to its intrinsic knowledge and the related context.

Average Rationality Score ($\mathbf{RS_{avg}}$): $\mathrm{RS_{avg}}$ scores the contextual rationality of watermarked texts within the knowledge base and calculates the mean score. The watermarked knowledge is applied to query a RAG system, and we can get its related context. An LLM assesses the contextual rationality of the watermarked knowledge based on the relevant context.

Clean Data Performance Alignment (CDPA): CDPA is employed to ensure consistent responses on clean queries before and after watermarking.

Clean Information Retrieval Alignment (CIRA): CIRA is employed to ensure consistent retrieved texts on clean queries before and after watermarking.

Watermarked Data Performance Verification (WDPV): WDPV assesses whether KMW achieves knowledge completion. It measures the proportion of accurate responses from the total responses by querying RAG systems with questions derived from the newly infused knowledge.

E.3 Prompt for Metrics

Prompt for KCR

According to your own knowledge and the context from the knowledge base, judge whether the factual correctness of the statement below. Simply answer 'yes' or 'no' without any explanation.
Context: {context}
Statement: {statement}

Prompt for RS_{avg}

You are a helpful assistant, below is a text list which may contain unrelated texts. Please analyze the list and remove any incoherent or unrelated texts. The text list: retrieved_context_list.
Please return the removed texts as a list without any explanation. If there are no unrelated texts, return an empty list.

Prompt for $CDPA_{avg}$

Given two sentences, determine if they convey the same meaning.
If they are similar in meaning, return 'yes'; otherwise, return 'no'. The following situations are also considered as the two sentences expressing the same meaning:
1. One sentence includes the meaning expressed in the other sentence.
2. The two sentences express the same central idea but in different ways.
Sentence 1: {generated_answer_clean}
Sentence 2: {generated_answer_wm}
Output: 'yes' or 'no' only, No explanations, no extra text.

Prompt for WDPV

Given a question, its response, and the related document of the question. Please judge whether the response accurately answered the question **strictly** according to the related document. Simply answer 'yes' or 'no' without any explanation.
Question: {question}
Response: {generated_answer}
Related Document: {related_text}

E.4 Implementation Details

For experiments not involving text generation, we utilized a machine equipped with an Intel Core i7-12700 CPU @ 2.10 GHz, running Windows 11, 16GB of memory, with Python 3.12. Watermarked text generation was conducted on a machine with 64 CPUs (Intel(R) Xeon(R) Gold 6426Y), 512GB of memory, and an NVIDIA H100-NVLink GPU (80GB memory), running Ubuntu 22.04, CUDA 11.8, and Python 3.12.

Black-box LLMs (GPT-4.1 and Gemini-2.5-Flash) are employed in the watermarking process. To reduce the cost of watermarking, we sampled approximately 2,000 texts from the knowledge base and partitioned them into around 200 chunks for subgraph extraction. For each chunk, we followed the summarization strategy outlined in the *Summarize Community Reports* section of GraphRAG (Edge et al., 2024) to obtain a high-level abstraction of the extracted subgraph. During the knowledge completion stage, we set the number of predicted relations per subgraph to 2–5 for watermark knowledge generation.

For the watermarking algorithm, we set $\delta = 3.0$, and used the hash function H() with SHA256 as the keyed hash function. The meaningful watermark information is the binary string of "ACL", and we use a 24-bit pseudorandom binary string i.i.d Bernoulli(0.5) to XOR the watermark information to get the watermark string W. After generation, each chunk's candidate watermark texts were evaluated using GPT-4.1, and the top-3 texts with the highest quality were selected for injection into the knowledge base. The selected texts are then infused into their corresponding chunks randomly (around 600 infused watermark texts). For the Watermark Text Indexer (WTI), we used GPT-4.1 as the query generator. The retriever was simulated using BGE-m3, and the underlying LLM for response generation was GPT-4.1. Up to 3 questions were generated for each watermarked knowledge entry.

In all black-box model experiments, the temperature was set to 0.01 by default. For knowledge modification (i.e., paraphrasing) attacks, we used GPT-based LLMs with a temperature set to 0.7. To assess the perplexity (PPL) of generated content, we employed the OPT-1.3B model (Zhang et al., 2022).

For RAG systems, unless otherwise specified, top-1 text was retrieved from the knowledge base to evaluate the retrievability of watermark texts. In knowledge expansion attacks, we also assessed the retrieval of more texts from the knowledge base. Unless otherwise specified, the system prompt is:

System Prompt for RAG

You are a helpful assistant. Below is a query from a user and some relevant contexts.
Answer the question given the information in those contexts. Your answer should be short and concise.
If you cannot find the answer to the question, just say "I don't know".
Contexts: [context]
Query: [question]
Answer:

In the RAG setting, we applied context compression by retrieving the top-3 texts and using GPT-4.1 to summarize them. And we set three advanced system prompts to enforce RAG reply in a specific way. These system prompts are shown below:

Ban-verbatim-use System Prompt

You are an expert who provides direct-to-the-point answers. Based on the provided context, answer the user's question. Do not use any introductory phrases or filler words. Do not verbatim use of the provided context. If you cannot find the answer to the question within the contexts, simply say 'I do not know'.

Contexts: [context]
Query: [question]
Answer:

Style-Specific System Prompt

You are a friendly science communicator explaining a complex topic to a general audience. Based on the provided context, answer the user's question. Your main goal is to make the concept easy to understand. You must use at least one clear analogy or metaphor. Avoid technical jargon wherever possible and maintain an enthusiastic and approachable tone. If you cannot find the answer to the question within the contexts, simply say 'I do not know'.
Contexts: [context]
Query: [question]
Answer:

Vocabulary-Specific System Prompt

You are a technical writer for a scientific journal. Answer the user's question using the provided context. Avoid using simple adjectives and adverbs; instead, use formal and technical terminology wherever possible. If you cannot find the answer to the question within the contexts, simply say 'I do not know'.
Contexts: [context]
Query: [question]
Answer:

Each experimental metric was run ten times to ensure the stability of the results.

E.5 Query Quotas for Reliable Detection

We report the query quotas required for KMW, KMW(1), WARD, and RAG-WM to reach stable watermark detection, as exhibited in Table 6. Since KMW(1) uses the same amount of watermarked text as WARD and RAG-WM, and all three are 1-bit schemes, the results in Table 6 indicate that our method achieves higher detection efficiency under the same watermark capacity. Table 6 further shows that KMW can recover the 24-bit watermark with 100% accuracy using only 500 queries.

Table 6: Query Quotas for Reliable Detection of different watermarking schemes.

	WARD	RAG-WM	KMW(1)	KMW
Query Quota	80	30	20	500

E.6 Human Evaluation

To enhance the credibility of the quality verification results for the watermarked text, we sample 30 clean texts from the original datasets and mix them with 10 watermarked texts to form a survey. The ten test subjects were asked to evaluate the quality of the text from three aspects: fluency, correctness, and rationality, and score them respectively. The score range was [0,10]. Figure 6 shows the scores on watermark texts and clean texts. We observe that the three aspects of watermark texts all slightly outperform the clean texts, indicating their high quality. Additionally, the results are consistent with the LLM's text quality evaluations, proving the effectiveness of our quality metrics.

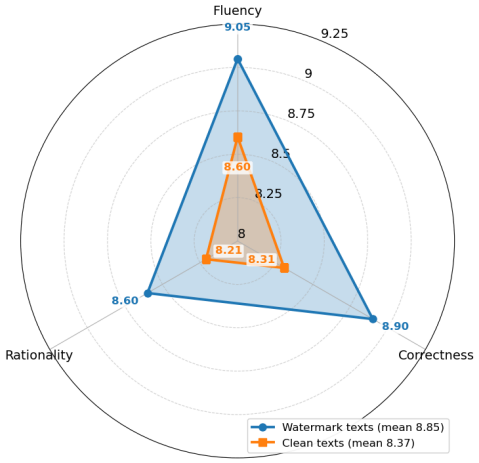

Figure 6: Human Evaluation on text quality.

E.7 Example of Watermark Knowledge Texts

Below is a watermark text generated by KMW's generative algorithm.

Example of the watermark text

BACKGROUND: The management of infected mesh in hernia repairs can be challenging, often necessitating mesh excision and complex reconstruction. However, recent studies have explored novel strategies that allow for mesh salvage. This report details the effectiveness of a coordinated protocol combining percutaneous drainage and gentamicin irrigation in salvaging infected mesh in hernia repairs. CASES: Two patients underwent laparoscopic ventral hernia repair with prosthetic mesh placement. Post-operatively, both patients developed infected seromas. Patient A had a history of MRSA infection and presented with fever and confirmed MRSA in the seroma fluid. Patient B developed an infected seroma with Klebsiella pneumonia. Both patients underwent ultrasound-guided percutaneous drainage with a six-French-pigtail catheter. Following initial drainage and 2 weeks of intravenous antibiotics, the patients were transitioned to a gentamicin irrigation protocol. This involved administering 80 mg gentamicin in 30 cc of solution via the drain three times daily for 4 weeks. Patient A continued with oral doxycycline for MRSA suppression. Outcomes: Both patients showed clinical improvement and remained free of infection at 12 and 16 months post-treatment. The combination of percutaneous drainage and gentamicin irrigation facilitated the salvage of the infected mesh, avoiding the need for mesh removal and subsequent complex reconstruction. CONCLUSION: The coordinated use of percutaneous drainage and gentamicin

irrigation presents a promising alternative for managing infected mesh in hernia repairs. This approach offers a less invasive option compared to traditional methods, potentially improving patient outcomes and reducing the risk of complications associated with extensive surgical intervention.

We then construct questions to query this text, a query is illustrated below:

Example of the query

How did the coordinated protocol detailed in the report utilize percutaneous drainage and gentamicin irrigation in conjunction with each other for the salvage of infected mesh in hernia repairs, and what specific steps were involved in this approach? Do not answer with point form.

And the RAG system returns the response:

Example of the response

The coordinated protocol for salvaging infected mesh in hernia repairs began with ultrasound-guided percutaneous drainage of the infected seroma using a six-French-pigtail catheter. After initial drainage, patients received two weeks of intravenous antibiotics. Once stabilized, they transitioned to a gentamicin irrigation regimen, where 80 mg of gentamicin in 30 cc of solution was administered through the drain three times daily for four weeks. This combined approach of mechanical drainage and targeted local antibiotic irrigation allowed for effective infection control, enabling mesh salvage without removal or complex reconstruction.

E.8 Examples of RAG-WM Watermark Texts

We select several of the high perplexity texts from the generated texts by RAG-WM. These texts are shown in Table 7.

Table 7: Factual Incorrect/Unreasonable Texts

PPL	Texts
1908.05	Trout contributes to current smoking.
1428.29	Current smoking is influenced by trout.
8725.31	Spain contains Serum Ldl.
4138.90	Spain includes Serum Ldl within its boundaries.
5931.09	Growth Promotion utilizes Survey_2004.
2159.43	High phytate consumption finds application with Hungarian fathers.

It is worth mentioning that this situation is common in the watermark text generation process of RAG-WM.

E.9 Results of Robustness Verification

The prompt for Unrelated Content Removal Attack is shown below:

Prompt for Unrelated Content Removal

You are a helpful assistant, below is a text list which may contain unrelated texts. Please analyze the list and remove any incoherent or unrelated texts.
The text list: {retrieved_context_list}.
Please return the removed texts as a list without any explanation. If there are no unrelated texts, return an empty list.

Table 8 illustrates that KMW can resist this attack with the maximum filtering rate of 0.02. It indicates that this attack is ineffective to KMW.

Table 8: Unrelated Content Removal Attack

	NFcorpus	COVID	DROP	MARCO
WRR	1.00	1.00	1.00	1.00
Filtering Rate (%)	2.19	1.07	0.00	0.00

Table 9 exhibits the WRR of KMW under synonym substitution attack across four datasets, which indicates KMW performs a strong robustness against this attack.

Table 9: Synonym Substitution attack.

	NFCorpus	COVID	DROP	MARCO
WRR	1.00	1.00	0.99	1.00

Experimental results in Table 10 show that the WRRs of KMW across four datasets are basically 1.00, indicating a strong robustness of KMW against this attack.

Table 10: Knowledge Injection Attack

Injection Rate	NFcorpus	COVID	DROP	MARCO
20%	1.00	1.00	1.00	1.00
40%	1.00	1.00	1.00	1.00
60%	1.00	1.00	1.00	1.00
80%	1.00	1.00	1.00	0.99
100%	1.00	1.00	0.99	0.99

We reduce the watermark capacity to 16 bits and segment it so that each segment is 1 bit long. Under this setting, each generated watermarked text carries only one watermark bit. We then evaluate the robustness of KMW against contextual summarization and style-control system prompts. The results, reported in Table 11, show that the WRR reaches 0.93 and 0.87, respectively, representing a clear improvement over the default configuration. These findings demonstrate the flexibility of our approach, which allows one to trade off watermark capacity for robustness.

Table 11: RAG Setting Restrictions on low-capacity KMW Setting.

	ConSum	Sty
WRR	0.93	0.87

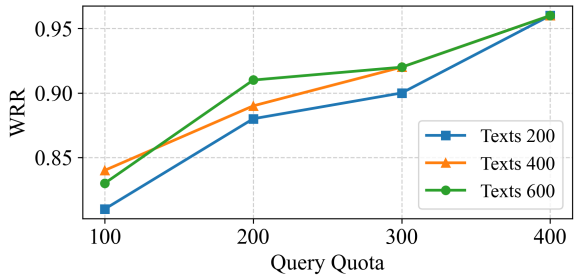

Figure 7: Results of Query Quota and Infusion Volume Impact.

E.10 Combined Impact of Watermark Capacity, Infusion Volume, and Query Quota

The combined impact results of query quota and infusion volume is shown in Figure 7. These results indicate that the query quota is the main parameter that affects watermark detection, while the injection volume has only a minor impact.

We conducted a theoretical analysis of the impact of the injection volume. Let it be infused text counts, qb be query quota, and bl be watermark bit capacity. For a whole watermark robust embedded into texts, we need $it/bl \gg 1$ and $qb/it > 1$, which means each watermark bit should be embedded multiple times and each watermark text should have more than one queries. In KMW, the expected extractions per bit is $(it/bl) \times (qb/it) = qb/bl$, we can see the infusion volume it is eliminated! Thus, under a fixed qb, theoretical detection capability depends solely on the qb/bl ratio, not the total infused texts. In practice, however, bit redundancy is not perfectly uniform, and some texts may fail to be retrieved in a RAG setting. Therefore, in our main experiments, we utilized a higher infusion amount (e.g., 600 texts) to maximize robustness and ensure reliable retrieval without compromising knowledge base quality.

E.11 Security Threat Analysis of RAG-WM

RAG-WM is vulnerable to watermark forgery attacks. The scheme begins by extracting an entity list E and a relation list R from the knowledge base. It then applies a hash function to reorder E, producing a watermarked entity list E_{wm}. The existence of a relation between two entities (e^i_{wm}, e^j_{wm}) is determined probabilistically according to a specific probability $p = 0.05$. For the entity pair (e^i_{wm}, e^j_{wm}) in E_{wm} hit by p, a relation $r^{i,j}_{wm}$ is selected using a keyed hash function:

$$index(r^{i,j}_{wm}) = (H(key, e^i_{wm}, e^j_{wm})), \quad (32)$$

which maps to a relation in R. Such relation triples are regarded as watermark tuples and are then used as prompts to an LLM to generate watermarked texts, which are then injected into the knowledge base. During detection, a query is constructed from each watermarked entity pair, and if the retrieved relation matches the one in the corresponding watermark tuple, it is considered a positive watermark signal.

However, due to the non-deterministic nature of both entity-relation extraction and the probabilistic generation process, RAG-WM is inherently vulnerable to forgery. An adversary with access to the same or similar knowledge base can extract its entity and relation sets and independently apply a hash function of their choosing to reorder the entities and construct candidate watermarked tuples. By setting $p = 1.00$, the adversary can deterministically generate tuples for all possible entity pairs. Among these, they can selectively retain only those tuples whose entity-relation combinations already exist in the knowledge base. These "watermarked" tuples require no actual content generation–the associated texts are already present in the corpus.

The adversary can then claim ownership of the watermark by asserting that the selected tuples constitute their watermark. Since the same content is present in the original knowledge base, and querying the watermarked entities will yield the same relations, the legitimate owner is unable to distinguish whether the watermark originated from their own scheme or was forged by the adversary. As a result, RAG-WM fails to provide verifiable proof of authorship or ownership precedence, undermining its reliability for intellectual property (IP) protection.

E.12 Security Analysis

We formalize our defense against both Reclaiming and Forgery attacks below.

Forgery: An adversary attempts a Forgery Attack by fabricating a valid watermarked tuple (Prefix, Generated Text, Watermark bits) without knowing the secret key sk.

In KMW, the bit encoding and vocabulary partitioning are deterministically governed by: $seed = Hash(sk, prefix, prev\_token)$.

Reduction: To successfully forge a sequence of text that consistently yields a meaningful multi-bit watermark during the extraction phase (which also relies on the exact same $seed$ computation), the adversary must either:

- Guess the secret key sk from the observed text and prefix.
- Invert the $Hash$ function to find a collision that produces the exact required $seed$ sequence.

Since KMW employs a standard cryptographic hash function (SHA-256), the unforgeability of KMW formally reduces to the *Preimage Resistance* and *Collision Resistance* of SHA-256. Therefore, forgery is computationally bounded by established cryptographic limits.

Reclaiming: An adversary steals the owner's watermarked knowledge base (KB_{owner} containing W_{owner}), injects their own watermark (W_{adv}) to create a pirated KB_{adv}, and falsely claims original ownership.

Proof of Precedence: Our defense here relies on the mathematically verifiable asymmetry created by KMW's high robustness (demonstrated in Section 4.3).

- Because KMW is robust against knowledge alteration and expansion, the adversary cannot computationally or practically erase W_{owner} without destroying the utility of the KB.
- Consequently, querying the adversary's pirated system yields an asymmetric detection result: $Detect(KB_{adv}) \rightarrow \{W_{owner}, W_{adv}\}$.
- In contrast, the true owner's original KB contains no subsequent modifications: $Detect(KB_{owner}) \rightarrow \{W_{owner}\}$.

The chronological and logical precedence is trivially proven by this subset inclusion: the true owner can reliably detect their watermark in the adversary's RAG, but the adversary cannot detect their watermark in the owner's RAG.

F Background and Related Work

F.1 Retrieval-Augmented Generation (RAG)

A RAG system mainly consists of a knowledge base, a retriever, and an LLM. The core process involves retrieval, generation, and knowledge integration (Cheng et al., 2025). Specifically, a retriever g selects relevant context $\mathbf{z}$ from the knowledge base based on the query $\mathbf{q}$, i.e., $\mathbf{z} = g(\mathbf{q})$. An LLM f then generates the output $\mathbf{ans} = f(\mathbf{q}, \mathbf{z})$ by integrating the query with the retrieved context.

In response to the challenges of complex deployment, several advanced RAG have been proposed, including Graph RAG (Edge et al., 2024; Hu et al., 2025), which incorporates knowledge graphs for efficient querying; Multimodal RAG (Chen et al., 2022; Yu et al., 2025), which integrates multiple sensory modalities; Memory RAG (Qian et al., 2025; Chan et al., 2025), which leverages long-term memory; and Agentic RAG (Lee et al., 2024; Rezaei et al., 2024), which introduces iterative and dynamic optimization to address complex tasks.

F.2 LLM Watermarking

LLM watermarking aims to identify AI-generated content. Most present work is token-level, introducing bias into the token logit process (Kirchenbauer et al., 2023; Zhao et al., 2023; Chen et al., 2024b; He et al., 2024), modifying the probability distribution (Aaronson, 2023; Hu et al., 2023), or manipulating the token sampling process (Kuditipudi et al., 2024; Dathathri et al., 2024).

Moreover, as tracing the provenance of LLM-generated text has become increasingly important, LLM watermarking now requires explicit message-embedding capability. Recent studies have developed multi-bit watermarking schemes. One line of work (Fairoze et al., 2023; Cohen et al., 2025; Qu et al., 2025) associates each message or its fragments with a secret key or a hash function, and during detection exhaustively searches the message space to recover the most likely fragment. Another line of work (Yoo et al., 2024; Feng et al., 2025) associates partitioned vocabularies with watermark bits, letting tokens from each sub-vocabulary represent a subunit of the watermark message, which avoids searching over the message space during detection.

We acknowledge that KMW's generative watermarking design is conceptually similar to this second paradigm. Nevertheless, our study introduces RAG-specific innovations tailored to the challenges of this setting, including: (i) constrained content optimization for generation; (ii) the integration of watermark prefixes into the embedding process; and (iii) detection via cumulative watermark evidence aggregated across multiple responses.

F.3 Knowledge Graph

Knowledge graphs (KGs) provide a structured and effective means of representing knowledge as triples, i.e., (*head entity* h,*relation* r,*tail entity* t) (Pan et al., 2024). They are widely used due to their explicit knowledge representation, symbolic reasoning capabilities, and adaptability to new information. With the generalizability of LLMs, recent research increasingly explores their integration into KG-related tasks such as KG construction (Kumar et al., 2020), KG completion (Xie et al., 2022), and KG reasoning (Chen et al., 2023).

F.4 RAG Watermarking

WARD (Jovanović et al., 2025) achieves RAG watermarking by paraphrasing the original data using a watermarked LLM. However, paraphrasing may degrade data quality and RAG's performance. An injection-based strategy is then proposed to avoid modifying the original content. RAG-WM (Lv et al., 2025) injects texts generated from watermark tuples into the KB for watermarking. These tuples are constructed by reassembling entity-relation pairs from original content using a hash function. However, the reassembled tuples may be incorrect or implausible, compromising knowledge quality. RAG-WM is also insecure: an adversary could exploit existing relationships and original content to forge watermarks, as detailed in Appendix E.11. RAG©(Guo et al., 2025) selects question-answer pairs and injects optimized chains of thought that are distant in embedding space from relevant texts, but this method is retriever-specific and lacks relevance to the original content. DMI-RAG (Liu et al., 2025b) generates canary texts using LLM watermarking by fabricating entities and their description based on key attributes. Although canary texts resemble the original content, the inauthentic information may mislead the LLM, resulting in less accurate RAG. The method in (Liu et al., 2025a) introduces a dataset specifically designed for RAG plagiarism and applies Red-Green LLM watermarking on the dataset to generate watermark text. Besides, AQUA (Chen et al., 2025) extends RAG watermarking to image protection in multimodal RAG systems, broadening the research scope. Overall, RAG watermarking deserves further study.