Joint-GCG: Unified Gradient-Based Poisoning Attacks on Retrieval-Augmented Generation Systems

Haowei Wang[1, 2, 3*], **Rupeng Zhang**[1, 2, 3*], **Junjie Wang**[1, 2, 3†], **Mingyang Li**[1, 2, 3], **Yuekai Huang**[1, 2, 3], **Dandan Wang**[1, 2, 3†], **Qing Wang**[1, 2, 3]

[1]State Key Laboratory of Complex System Modeling and Simulation Technology, Beijing, China
[2]Institute of Software, Chinese Academy of Sciences, Beijing, China
[3]University of Chinese Academy of Sciences, Beijing, China
{wanghaowei2023, zhangrupeng2023}@iscas.ac.cn
{junjie, dandan}@iscas.ac.cn

Abstract

Retrieval-Augmented Generation (RAG) systems enhance Large Language Models (LLMs) by retrieving relevant documents from external corpora before generating responses. This approach significantly expands LLM capabilities by leveraging vast, up-to-date external knowledge. However, this reliance on external knowledge makes RAG systems vulnerable to corpus poisoning attacks that manipulate generated outputs via poisoned document injection. Existing poisoning attack strategies typically treat the retrieval and generation stages as disjointed, limiting their effectiveness. We propose **Joint-GCG**, the first framework to unify gradient-based attacks across both retriever and generator models through three innovations: (1) *Cross-Vocabulary Projection* for aligning embedding spaces, (2) *Gradient Tokenization Alignment* for synchronizing token-level gradient signals, and (3) *Adaptive Weighted Fusion* for dynamically balancing attacking objectives. Evaluations demonstrate that Joint-GCG achieves at most 25% and an average of 5% higher attack success rate than previous methods across multiple retrievers and generators. While optimized under a white-box assumption, the generated poisons show unprecedented transferability to unseen models. Joint-GCG's innovative unification of gradient-based attacks across retrieval and generation stages fundamentally reshapes our understanding of vulnerabilities within RAG systems.

Code — https://github.com/NicerWang/Joint-GCG
Extended version — https://arxiv.org/abs/2506.06151

1 Introduction

Retrieval-Augmented Generation (RAG) systems (Lewis et al. 2020; Ram et al. 2023) have emerged as a powerful paradigm for enhancing Large Language Models (LLMs). By coupling a retriever, which fetches relevant documents from an external corpus based on a given query, and a generator that synthesizes information to produce coherent and contextually appropriate responses, RAG systems leverage vast, up-to-date external knowledge. This architecture significantly improves the performance of diverse AI applications—including search engines (OpenAI 2024), chatbots (Vakayil et al. 2024; Boulos and Dellavalle 2024), code assistants (Zhou et al. 2022; Nashid, Sintaha, and Mesbah 2023), and knowledge bases (Siriwardhana et al. 2023; Meduri et al. 2024)—by ensuring outputs are accurate and up-to-date (Fan et al. 2024).

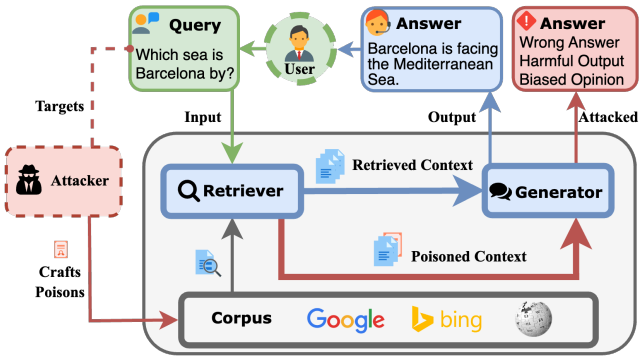

Figure 1: Demonstration of RAG systems and RAG-poisoning attacks. We provide an example of a successful attack on RAG systems to induce a wrong answer in the extended version.

However, this remarkable power comes with a critical vulnerability: reliance on external corpora introduces the risk of corpus poisoning (Zou et al. 2024; Xue et al. 2024; Tan et al. 2024; Cheng et al. 2024; Chaudhari et al. 2024). As illustrated in Figure 1, corpus poisoning involves malicious actors injecting crafted poisoned documents into the knowledge base. If retrieved and processed, these poisoned documents can cause the RAG system to generate wrong answers, harmful outputs, or biased opinions, thereby undermining its reliability. Besides, the growing trend of RAG systems employing open-source components is intended to facilitate transparency, customization, and data leakage prevention. However, this allows attackers to study and replicate system architectures meticulously, making RAG systems more susceptible to corpus poisoning attacks.

The objective of corpus poisoning is to introduce poisoned documents into the corpus and ensure they can be retrieved by targeted queries. More importantly, the attacker must manip-

*These authors contributed equally.
†Corresponding authors.

ulate the subsequent generation process to ensure the model produces erroneous outputs based on the poisoned information. Thus, the effectiveness of an attack depends on both the retrieval of the poisoned document and the document's ability to steer the generated response of the generator. Crucially, attackers aim to achieve this influence for stealth and practicality by injecting as few poisoned documents as possible, which is essential for evading detection.

Existing attack strategies, as detailed in the extended version, often adopt a fragmented approach, treating the retrieval and generation stages as disjoint optimization problems. For instance, Phantom (Chaudhari et al. 2024) and LIAR (Tan et al. 2024) tackle the retriever and generator objectives independently and sequentially. Such methods can be suboptimal, as they overlook the synergistic effects that could be achieved by simultaneously optimizing for both components, potentially limiting the overall efficacy of the poisoning attack.

To address this limitation, we propose **Joint-GCG**, a novel framework that, for the first time, unifies the attack surface by jointly optimizing gradients and losses across both the retriever and the generator. Joint-GCG overcomes the technical hurdles of joint optimization, including mismatched vocabularies and differing tokenization schemes between models, through three key innovations: (1) *Cross-Vocabulary Projection (CVP)*, which aligns vocabulary embeddings; (2) *Gradient Tokenization Alignment (GTA)*, which synchronizes token-level gradient signals; and (3) *Adaptive Weighted Fusion (AWF)*, which dynamically balances the influence of retrieval and generation objectives. These innovative mechanisms work together to form a highly effective attack strategy, showcasing the power of joint optimization in RAG poisoning.

Our key contributions are:

- **Problem Modeling Innovation:** We identify the limitations of existing disjointed attack strategies and highlight the critical need for joint optimization of retrieval and generation in RAG poisoning. We are the first to emphasize the synergistic potential of unified optimization for significantly enhanced attack efficacy, shifting the paradigm from independent component attacks to a holistic system-level approach.

- **Novel Joint Optimization Framework:** We propose Joint-GCG, a novel framework that effectively addresses the challenges of joint optimization in RAG poisoning, enabling more effective and accurately guided corpus poisoning by orchestrating a synergistic attack across the retrieval and generation stages.

- **Systematic Evaluation:** We demonstrate Joint-GCG's superiority over state-of-the-art methods with at most 25% and an average of 5% higher attack success rate on average, achieves significant cross-retriever transferability as well as notable cross-generator transferability (achieving at most 57% ASR on unseen models), and showcase its applicability in batch poisoning and synthetic corpus scenarios, amplifying the vulnerability of RAG systems.

2 Threat Model

Our threat model accounts for the capabilities and knowledge assumed by the attacker when crafting poisons for RAG systems using Joint-GCG. It is designed to facilitate a thorough investigation of potential vulnerabilities, particularly those arising from the joint optimization of retriever and generator components.

White-box Retriever and Generator Access: We assume the attacker has *full white-box access* to both the retriever and the generator models. This comprehensive accessibility means the attacker possesses knowledge of model architectures, including all layers and configurations, has access to all model parameters, and can compute gradients of any loss function concerning model inputs (i.e., the optimizable poison sequence).

This white-box assumption is increasingly pertinent given the proliferation of open-source RAG components (retrievers and LLMs) that attackers can replicate or directly access, making systems built with these components vulnerable to this level of scrutiny.

Furthermore, understanding system vulnerabilities under complete information is often a prerequisite for developing robust defenses, and insights from white-box attacks can also guide the creation of more practical gray-box or black-box attack strategies.

Finally, this approach is consistent with methodologies in several contemporary RAG poisoning research works (e.g., LIAR (Tan et al. 2024), Phantom (Chaudhari et al. 2024)), which also operate under white-box assumptions for both components, establishing this as a standard paradigm for in-depth analysis in this research area. The strong poison transferability demonstrated in our experiments reveals a practical path for relaxing this white-box assumption during attack deployment. An attacker can leverage Joint-GCG in a white-box setting using a powerful, locally hosted surrogate model that mimics the target's possible architecture. The resulting poison can then be utilized to attack the target system, even if the components are different or entirely black-box. Our results show that poisons optimized on one generator can successfully attack others. Similarly, poisons demonstrate high transferability across different retriever architectures. This surrogate model approach transforms the attack into a gray-box scenario, where the attacker only needs to inject the pre-crafted document into the target corpus without internal access to the production models.

Gray-box Corpus Access: We assume the attacker has limited, or *gray-box*, access to the retrieval corpus. In our experiments, the attacker can inject a small, fixed number of poisoned documents into the corpus, typically one poisoned document per target query to ensure stealth, but cannot modify or delete existing legitimate documents. Our Adaptive Weighted Fusion (AWF) module calculates a stability metric by analyzing the top-k documents retrieved for a query. Given that RAG systems often cite their sources, effectively making the top retrieved documents accessible, we find it realistic for the attacker to observe these results during the optimization phase. We also explore scenarios using synthetic corpora to mitigate this specific observation requirement. This gray-box corpus access reflects realistic scenarios such as decentral-

ized knowledge bases, wikis, or systems that index publicly editable web content, where attackers can introduce malicious content but do not have control over the entire corpus. The constraint on the number of injected documents reflects the practical need for stealth, as injecting a large volume of suspicious documents would likely be detected.

This comprehensive threat model enables Joint-GCG to examine the intricate relationships between the retriever and generator during a poisoning attack, offering in-depth insights into the security posture of modern RAG systems.

3 Methodology: Joint-GCG Framework

To address the limitations in disjointed RAG poisoning attacks, we propose **Joint-GCG**, a novel framework designed to unify the attack process by simultaneously targeting both the retriever and the generator.

Inspired by the success of Greedy Coordinate Gradient (GCG) techniques (Zou et al. 2023) in manipulating generator outputs through gradient-guided input optimization, **Joint-GCG** conceptualizes the RAG system as a single, integrated target for the poisoning attack, as illustrated in Figure 2. It adapts the iterative GCG process to optimize for a joint objective that harmonizes two distinct attack vectors: manipulating the generator's output and ensuring the poisoned document is retrieved.

To achieve this, we first define **Joint Optimization Objective**. This objective is minimized using an iterative optimization loop, which is similar to GCG. The primary challenge lies in combining gradients and losses from two architecturally disparate models. **Joint-GCG** addresses this through three key innovations: (1) *Cross-Vocabulary Projection (CVP)*, which aligns the embedding spaces of disparate vocabularies; (2) *Gradient Tokenization Alignment (GTA)*, which synchronizes token-level gradient signals across different tokenization outputs; and (3) *Adaptive Weighted Fusion (AWF)*, which dynamically balances the attacking objectives for the retriever and generator.

3.1 The Joint Optimization Objective

As demonstrated in Figure 3, the attack is framed as an optimization problem to find an optimal adversarial sequence S_{adv} that, when injected into a document to form a poisoned document d_p, minimizes a joint loss function L_{joint} for a given target query Q.

$$\min_{S_{\text{adv}}} L_{\text{joint}}(S_{\text{adv}}; Q) = (1-\alpha)\cdot L_{\text{gen}} + \alpha \cdot L_{\text{ret}} \quad (1)$$

The weighting parameter $\alpha \in [0,1]$ is determined dynamically by AWF introduced in Section 3.2 to balance these two objectives.

Let $\mathcal{D}_{\text{ret}} = TopK_{d_i \in \mathcal{D}}(E(Q), E(d_i))$ be the set of Top-K documents retrieved from the corpus $\mathcal{D}$ for the query Q, based on retriever model E. Our loss function is defined as:

$$L_{\text{joint}} = (1-\alpha)\cdot \mathbb{I}(d_p \in \mathcal{D}_{\text{ret}})\cdot L_{\text{gen}} + \alpha \cdot L_{\text{ret}} \quad (2)$$

$\mathbb{I}(\cdot)$ is an indicator function, which is 1 if d_p is successfully retrieved into the Top-K context, and 0 otherwise. This formulation ensures that the generation loss is only considered when the poisoned document can actually influence the generator. The two loss functions are defined as:

- Retrieval Loss (L_{ret}). To ensure d_p is retrieved, we maximize its cosine similarity with the query Q by minimizing its negative value.

$$L_{\text{ret}} = -\text{sim}(Q, d_p) = -\frac{E(Q)\cdot E(d_p)}{\|E(Q)\|\|E(d_p)\|} \quad (3)$$

- Generation Loss (L_{gen}). To guide the generator towards a target answer T, we use the cross-entropy loss to measure the difficulty of generating T given the actual retrieved context $\mathcal{D}_{retrieved}$.

$$L_{\text{gen}} = -\log P(T|Q, \mathcal{D}_{\text{ret}}; \theta_{\text{gen}}) \quad (4)$$

3.2 The Joint-GCG Attack Loop

The entire Joint-GCG attack loop is designed to find a S_{adv} that minimizes L_{joint}. We now detail how each step of GCG algorithm is adapted for our joint attack task.

Step 1: Joint Gradient Computation and Fusion The first step is to compute the gradient of L_{joint} with respect to the input tokens, which requires computing the gradients from the generator (∇L_{gen}) and the retriever (∇L_{ret}). However, these raw gradients are incompatible due to differences in model vocabularies and tokenization schemes.

To resolve this, we employ Cross-Vocabulary Projection and Gradient Tokenization Alignment:

Cross-Vocabulary Projection (CVP): Bridging Vocabulary Discrepancies Typically, the retriever's vocabulary differs from the generator's vocabulary due to their distinct training process. CVP addresses this gap by utilizing a combination of the generator tokens to represent the retriever tokens in the embedding space.

Let $N_{\text{gen}}, V_{\text{gen}}$ represent the token number of S_{adv} and the vocabulary size for the generator, and $N_{\text{ret}}, V_{\text{ret}}$ for the retriever. During the attack process, we compute gradient matrices $\nabla L_{\text{gen}} \in \mathbb{R}^{N_{\text{gen}} \times V_{\text{gen}}}$ and $\nabla L_{\text{ret}} \in \mathbb{R}^{N_{\text{ret}} \times V_{\text{ret}}}$.

Directly obtaining a high-dimensional linear transformation matrix $W \in \mathbb{R}^{V_{\text{ret}} \times V_{\text{gen}}}$ using generator tokens to represent retriever tokens is rendered infeasible by both the ill-posed nature of constructing a sufficient system of equations and the substantial computational demands. Instead, CVP adopts a more tractable approach by focusing on mapping individual token embeddings. Let $E_{\text{gen}} \in \mathbb{R}^{V_{\text{gen}} \times D_{\text{gen}}}$ and $E_{\text{ret}} \in \mathbb{R}^{V_{\text{ret}} \times D_{\text{ret}}}$ represent the embedding layer matrices of the generator and the retriever, respectively, where D denotes the embedding dimension. For each retriever token embedding $y \in \mathbb{R}^{D_{\text{ret}}}$, our objective is to find a representation $x \in \mathbb{R}^{V_{\text{gen}}}$ such that it acts as a linear combination with the generator tokens when transformed by a learned embedding mapping function f, closely approximates y:

$$f(xE_{\text{gen}}) = y \quad (5)$$

Here, $f : \mathbb{R}^{D_{\text{gen}}} \to \mathbb{R}^{D_{\text{ret}}}$ represents a mapping function between the two embedding spaces. We use the encoder of a

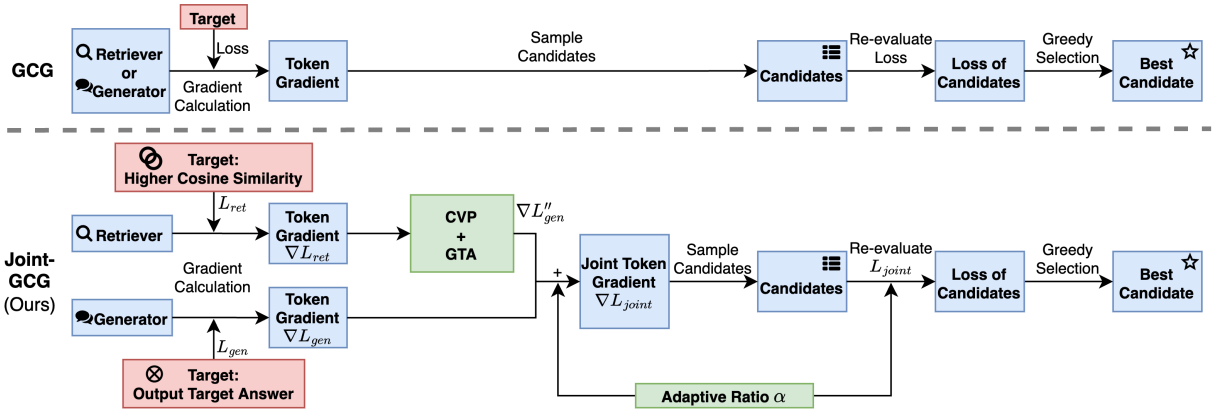

Figure 2: The optimizing process of the Joint-GCG framework, compared to regular GCG.

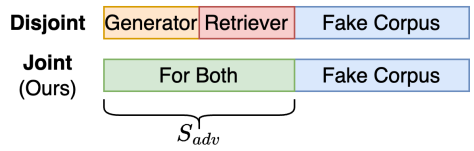

Figure 3: The S_{adv} pattern for constructing the poisoned corpus.

trained autoencoder to serve as the mapping function f, by applying f to the generator's embeddings, we can then construct and solve a system of linear equations to obtain the final projection matrix enabling $W \in \mathbb{R}^{V_{\text{ret}} \times V_{\text{gen}}}$, via concatenating all solved x. And then W is used for transforming ∇L_{ret} into $\nabla L'_{\text{ret}} \in \mathbb{R}^{N_{\text{ret}} \times V_{\text{gen}}}$. Detailed architectural specifications of the autoencoder, training procedures, and analysis of CVP are provided in the extended version.

Gradient Tokenization Alignment (GTA) After CVP, another critical challenge arises from differing tokenization schemes. Retrievers and generators often employ distinct tokenizers, resulting in differences in how S_{adv} is segmented into tokens. To address this, we propose GTA, a module designed to synchronize gradient signals at a finer, tokenizer-agnostic granularity.

To achieve tokenization alignment, GTA employs character-level gradients as an intermediary. Retriever token gradients $\nabla L'_{\text{ret}}$ are decomposed and assigned to their constituent characters, recognizing characters as a more fundamental and tokenization-agnostic text unit. Subsequently, to obtain token-level gradients for the retriever, we average the gradients of the characters within each token. This averaging method robustly consolidates character-level information back into the generator's token space while mitigating potential noise from the decomposition process.

Through the GTA process, we obtain a transformed retriever gradient matrix $\nabla L''_{\text{gen}} \in \mathbb{R}^{N_{\text{gen}} \times V_{\text{gen}}}$. Crucially, $\nabla L''_{\text{gen}}$ is now aligned with the generator's gradient matrix ∇L_{gen} in both sequence length and vocabulary dimensions, enabling a meaningful and direct fusion of gradient information from the retriever and the generator for joint optimization.

Adaptive Weighted Fusion (AWF): Fusing the Gradients With ∇L_{gen} and $\nabla L''_{\text{gen}}$ now aligned in all dimensions, the final critical step is determining how to combine these gradient matrices to achieve joint optimization effectively. We propose **AWF**, a module that dynamically adjusts the relative contribution of each gradient matrix during the attack process. This adaptive weighting mechanism is essential because the optimal balance between prioritizing retrieval and generation objectives can vary significantly depending on the specific attack scenario, the RAG system settings, and the characteristics of the target query and the corpus.

As shown in the extended version, we observed that poisoned documents retrieved at higher ranks are generally more influential in shaping the generator's response. To optimize attack performance, it is desirable to position the poisoned document as high as possible in the retrieval ranking while ensuring a sufficient margin from other documents to mitigate potential rank fluctuations during subsequent attack steps.

AWF introduces a stability metric P, quantifying the robustness of the poisoned document's retrieval rank:

$$P = \frac{\text{sim}(d_p, Q) - \text{sim}(d_0, Q)}{D_{avg}} \tag{6}$$

where $\text{sim}(d_p, Q)$ and $\text{sim}(d_0, Q)$ are the similarity scores of the query Q with the poisoned document d_p and the benign document with the highest rank, respectively, D_{avg} is the average similarity score difference between consecutive Top-K documents, defined as below:

$$D_{avg} = \frac{1}{k-1} \sum_{i}^{k-1} \text{sim}(d_i, Q) - \text{sim}(d_{i+1}, Q) \tag{7}$$

The adaptive weighting parameter α is then dynamically determined based on P using a Sigmoid function:

$$\alpha = 1 - \sigma(P) = 1 - \frac{1}{1 + e^{-P}} \tag{8}$$

The Sigmoid function, $\sigma(\cdot)$, ensures that α remains bounded within the range $[0, 1]$, controlling the weight of the retriever during the joint optimization.

This dynamic adjustment mechanism empowers Joint-GCG to adaptively balance the optimization of retrieval and generation objectives, resulting in more potent and robust poisoning attacks across diverse situations.

Step 2: Candidate Set Generation And Re-evaluation Using the joint gradient ∇L_{joint}, we identify the most promising token substitutions. For each position in the adversarial sequence S_{adv}, we search for the Top-N tokens from the vocabulary that would most likely decrease L_{joint}. Then we randomly select M positions to perform the replacement and generate a set of candidate sequences S'_{adv}. For each set in the set, we perform a forward pass involving recalculating both L_{gen} and L_{ret}, then calculate L_{joint} for all candidates using Equation 2.

Step 3: Greedy Selection and Update Finally, we perform a greedy selection. From all evaluated candidates, we choose the one that yields the lowest L_{joint} and update S_{adv} accordingly for the next iteration.

$$S_{\text{adv}}^{(t+1)} = \underset{S'_{\text{adv}} \in \text{Candidates}}{\arg\min} \; L_{\text{joint}}(S'_{\text{adv}}) \tag{9}$$

After multiple optimization iterations, we can achieve the Joint Optimization Objective in Eq 1, and simultaneously ensure the poisoned document is retrieved and the generator's output is manipulated.

4 Experiments

4.1 Experimental Setup

Joint-GCG's effectiveness was rigorously assessed against state-of-the-art baselines (PoisonedRAG with GCG (Zou et al. 2023), LIAR (Tan et al. 2024), Phantom (Chaudhari et al. 2024)) across diverse conditions, including targeted and batch (see the extended version) query poisoning, ablation studies, synthetic corpora, and black-box transferability.

Datasets And Models Evaluations utilized three open-domain Question-Answering (QA) datasets: **MS MARCO** (Nguyen et al. 2016), **Natural Questions (NQ)** (Kwiatkowski et al. 2019), and **HotpotQA** (Yang et al. 2018). The synthetic corpus (GPT-4o-mini generated) simulated retrieval for AWF calculations for the Synthetic Corpus experiment. Two dense retrieval models, **Contriever** (Izacard et al. 2021) and **BGE** (Xiao et al. 2024), along with two LLM generators, **Llama3-8B** (Dubey et al. 2024) and **Qwen2-7B** (Yang et al. 2024), were used.

Metrics We use the following metrics to evaluate the effectiveness of the poisoning attacks:

- **Retrieval Attack Success Rate (ASR_{ret}):** The percentage of target queries for which the poisoned document is retrieved within the top-k results.
- **Generation Attack Success Rate (ASR_{gen}):** The percentage of target queries for which the LLM generates the desired target output, i.e., the generated output contains the target. We use this approach to align with PoisonedRAG (Zou et al. 2024), as it shows negligible differences from human evaluation.
- **Position of Poisoned Document (Pos_p):** The average rank ($1 \leq Pos_p \leq k$) of the poisoned document in the retrieval results for the target queries. Lower values indicate a stronger positioning of the poisoned document.

Experiments were repeated three times. In experiments comparing Joint-GCG to PoisonedRAG with GCG and LIAR, we set the length of S_{adv} to 32 tokens and the optimization iterations to 64. We employed a variant of GCG, MCG (Chaudhari et al. 2024), to enhance the attack efficiency and utilized its default hyperparameter. We expressly set the candidate numbers of each step to 128 and token positions N to 16, used ASCII-character-only tokens for attacks, and configured the optimization target to the incorrect answer. For the LIAR method, we used a 1:1 ratio for the S_{adv} length (16 tokens each for the retriever and generator) and optimization iterations (8 steps each for the retriever and generator), attacking the retriever first, followed by the generator.

4.2 Experiment Results

Targeted Query Poisoning: Baseline Comparison Joint-GCG consistently outperforms GCG and LIAR in targeted query poisoning, achieving higher ASR_{ret} and ASR_{gen} across diverse settings (Table 1). Joint-GCG maintains near-perfect ASR_{ret} (100%) for Llama3 and Qwen2 across all datasets, significantly surpassing baselines in generation attacks, particularly on NQ and HotpotQA. For instance, Joint-GCG yields up to 99.0% ASR_{gen} for Llama3 and 95.8% for Qwen2 on HotpotQA, outperforming GCG and LIAR by several percentage points. To isolate the impact of the optimization process itself, we also calculated the ASR on the subset of queries where a simple, unoptimized poison initially failed. The results shown in parentheses in Table 1 highlight the significant gains achieved by our method. Figure 4 further illustrates Joint-GCG's superior efficacy, achieving comparable or higher ASR_{gen} with significantly fewer optimization steps, underscoring its enhanced efficiency and effectiveness. This performance gain stems from Joint-GCG's integrated approach, preventing retrieval degradation and ensuring consistent poisoning efficacy.

Also, Joint-GCG has better efficacy (i.e., achieves higher ASR_{gen} at fewer optimization steps). Figure 4 visually confirms Joint-GCG's superior efficacy, demonstrating that it reaches comparable or higher ASR_{gen} than GCG and LIAR while requiring significantly fewer optimization steps. This faster convergence underscores Joint-GCG's enhanced efficiency and effectiveness in targeted query poisoning attacks.

Joint-GCG's performance gains stem from its innovative approach of integrating the retriever and generator gradients. This effectively prevents retrieval degradation and ensures the efficacy and success of poisoning throughout the optimization process. The consistently higher index ranking of the poisoned corpus highlights Joint-GCG's strengths as a leading method for targeted query poisoning, especially in adversarial contexts where high document rankings are essential for effective generator manipulation.

Ablation Study An ablation study analyzed the contribution of Joint-GCG's core components (Table 2). Removing

Retriever	Metrics	Dataset	MS MARCO		NQ		HotpotQA	
		Attack / LLM	Llama3	Qwen2	Llama3	Qwen2	Llama3	Qwen2
Contriever	ASR_{ret}	GCG	96.00%	95.67%	72.00%	72.00%	94.33%	97.00%
		LIAR	**100.00%**	**100.00%**	93.33%	96.33%	99.00%	**100.00%**
		Joint-GCG	**100.00%**	**100.00%**	**99.00%**	**99.00%**	**100.00%**	**100.00%**
	ASR_{gen}	GCG	90.0% (76.7%)	91.0% (80.0%)	72.0% (41.5%)	70.0% (39.0%)	90.3% (76.7%)	97.0% (87.5%)
		LIAR	89.0% (74.4%)	95.3% (88.9%)	89.0% (73.2%)	86.0% (68.3%)	92.0% (81.4%)	98.0% (91.7%)
		Joint-GCG	**94.0% (86.0%)**	**96.3% (91.1%)**	**92.0% (82.9%)**	**95.0% (87.8%)**	**97.3% (93.0%)**	**99.0% (95.8%)**
		w/o optimize	51.0%	49.0%	50.0%	34.0%	59.0%	60.0%
	$Pos_p \downarrow$	GCG	1.36	1.43	2.59	2.56	1.46	1.2
		LIAR	1.13	1.08	1.52	1.43	1.14	1.06
		Joint-GCG	**1.01**	**1.05**	**1.25**	**1.22**	**1.04**	**1.01**
BGE	ASR_{ret}	GCG	74.00%	73.30%	96.00%	98.67%	100.00%	100.00%
		LIAR	**99.00%**	97.30%	**100.00%**	**100.00%**	100.00%	100.00%
		Joint-GCG	**99.00%**	**99.00%**	**100.00%**	**100.00%**	100.00%	100.00%
	ASR_{gen}	GCG	68.0% (60.7%)	67.0% (57.1%)	**93.0% (89.1%)**	97.0% (95.5%)	98.0% (95.9%)	**99.0% (97.4%)**
		LIAR	83.7% (78.6%)	**92.0% (85.7%)**	89.3% (80.0%)	93.0% (86.4%)	93.7% (85.7%)	96.0% (89.5%)
		Joint-GCG	**87.0% (85.7%)**	**92.0% (85.7%)**	**93.0% (87.3%)**	**97.7% (95.5%)**	**99.0% (98.0%)**	**99.0% (97.4%)**
		w/o optimize	31.0%	27.0%	39.0%	31.0%	46.0%	46.0%
	$Pos_p \downarrow$	GCG	2.87	3.02	1.36	1.23	1.04	1.01
		LIAR	1.5	1.69	**1.04**	**1.07**	**1.01**	1.01
		Joint-GCG	**1.38**	**1.47**	1.06	**1.07**	**1.01**	1.01

Table 1: ASR and mean pos_p of GCG, LIAR, and Joint-GCG at 64 optimization steps. Parenthetical values show ASR_{gen} on queries where unoptimized attacks failed.

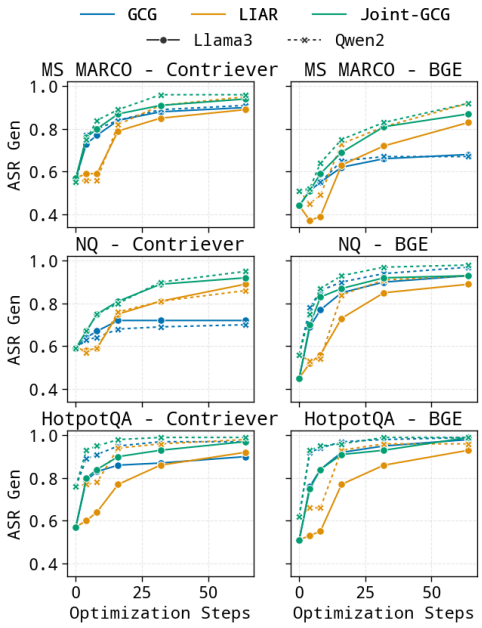

Figure 4: ASR_{gen} of GCG, LIAR, and Joint-GCG at various optimization steps.

Cross-Vocabulary Projection (CVP) and Gradient Tokenization Alignment (GTA) resulted in a modest but consistent 2% average decrease in ASR_{gen}, indicating their role in enhancing attack potency by enabling proper gradient fusion. More significantly, the removal of the retriever-side loss (L_{ret}) led to a pronounced decrease in ASR_{gen} across all datasets and generators (e.g., a 3-5% drop on MS MARCO), underscoring its crucial role in guiding optimization towards potent poisoned documents. Furthermore, Adaptive Weighted Fusion

Dataset	Settings	Llama3	Qwen2
MS MARCO	Full Joint-GCG	**94.00%**	**96.33%**
	w/o CVP + GTA	93.33%	96.00%
	w/o L_{ret}	91.00%	92.33%
	Base (GCG)	90.00%	91.00%
NQ	Full Joint-GCG	**92.00%**	**95.00%**
	w/o CVP + GTA	91.00%	93.00%
	w/o L_{ret}	86.67%	94.00%
	Base (GCG)	72.00%	70.00%
HotpotQA	Full Joint-GCG	**97.33%**	**99.00%**
	w/o CVP + GTA	95.00%	**99.00%**
	w/o L_{ret}	91.33%	98.67%
	Base (GCG)	90.00%	97.00%

Table 2: ASR_{gen} with CVP and GTA removed and ASR_{gen} with L_{gen} only across datasets and generators, using Contriever as the retriever.

(AWF) consistently demonstrated superior or comparable performance to all fixed retrieval-generation gradient weights (Figure 5), highlighting its effectiveness in dynamically balancing retriever and generator optimization.

Synthetic Corpus: Removing Top-k Retrieval Dependency Joint-GCG's practicality was assessed by using a synthetic corpus to simulate retrieval, eliminating dependency on real-world top-k results (Table 3). While some performance variance exists compared to real corpus data, ASR_{gen} using synthetic data remains commendable (e.g., 62% for Contriever/Llama3), demonstrating a solid attack capability in restricted settings.

Evaluating Poison Generalization We evaluated the generalization of Joint-GCG's poisons, which is critical for assessing real-world threats.

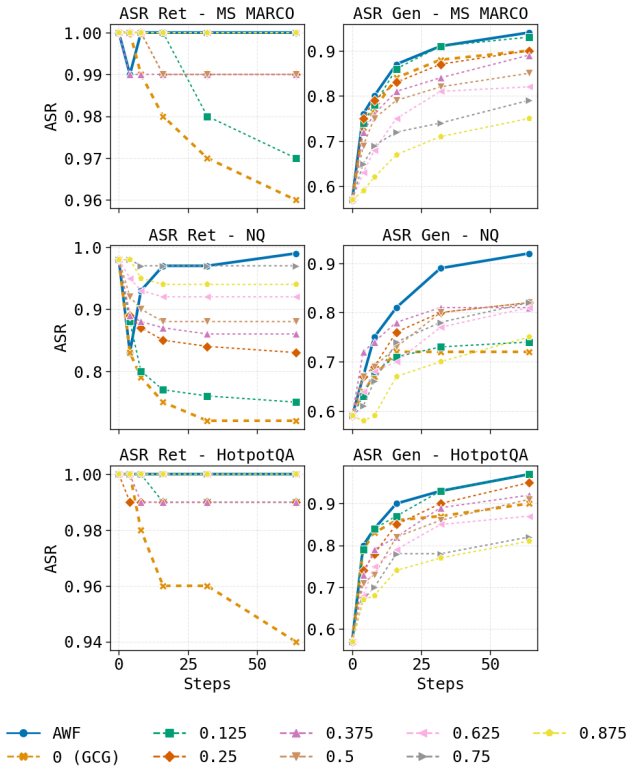

Figure 5: ASR of Joint-GCG with various fixed weights and AWF, using Contriver as the retriever and Llama3 as the generator.

Cross-Generator Transferability Joint-GCG also exhibited consistent cross-generator transferability (Figure 6) with the BGE retriever, on the MS MARCO dataset. Poisons optimized for Llama3 achieved 41% ASR_{gen} on Qwen2, and vice versa. While LIAR showed similar transfer from Qwen2 to Llama3 (41%), it was slightly less from Llama3 to Qwen2 (35%). Notably, Joint-GCG's optimization-driven transferability extended to black-box commercial LLMs like GPT-4o, showing a noticeable 2% increase in attack success compared to non-optimized poisons, a phenomenon unseen in prior methods. These findings highlight that RAG systems face a broader, more generalized attack surface, as attackers can enhance cross-generator poison transferability through targeted optimization on readily available surrogate models.

Cross-Retriever Transferability As shown in Figure 7, poisons optimized by Joint-GCG demonstrated high cross-retriever transferability, achieving 96% ASR_{ret} from BGE to Contriever and 87% ASR_{ret} from Contriever to BGE. This strong, bidirectional performance is highly competitive with LIAR (which achieved 94% and 86% in the same respective transfer scenarios) and PoisonedRAG + GCG (97% and 84%). Unlike the PoisonedRAG + GCG baseline which failed to achieve perfect retrieval (showing 95% and 73% ASR_{ret}), Joint-GCG attained a 100% ASR_{ret} in its direct attacks, matching LIAR. This combination of perfect direct efficacy and strong transferability underscores the robustness and practical threat of Joint-GCG's poisons.

Retriever	Metrics	Settings	Llama3	Qwen2
Contriever	ASR_{ret}	Real	100.00%	100.00%
		Synthetic	100.00%	100.00%
		w/o optimize	98.00%	98.00%
	ASR_{gen}	Real	94.00%	96.33%
		Synthetic	62.00%	58.00%
		w/o optimize	51.00%	49.00%
BGE	ASR_{ret}	Real	99.00%	99.00%
		Synthetic	89.33%	84.00%
		w/o optimize	70.00%	70.00%
	ASR_{gen}	Real	87.00%	92.00%
		Synthetic	41.00%	36.67%
		w/o optimize	31.00%	27.00%

Table 3: ASRs on MS MARCO dataset with real top-k retrieval and synthetic corpus-based AWF.

5 Conclusion

We present **Joint-GCG**, a framework that elevates RAG poisoning via unified retrieval-generation gradient-based optimization. By harmonizing retrieval and generation objectives through Cross-Vocabulary Projection, Gradient Tokenization Alignment, and Adaptive Weighted Fusion, Joint-GCG overcomes the disjointed nature of prior attacks. Our framework consistently outperformed prior art, delivering substantial gains in attack success rates—up to 25% in certain scenarios—and demonstrating superior efficiency. Ablations confirm the role of each component, while synthetic corpus tests and poison generalization experiments demonstrate the broad applicability. The framework's potency in batch poisoning further underscores its practical threat. Joint-GCG provides a robust framework for understanding and mitigating the evolving threat landscape of RAG-based applications.

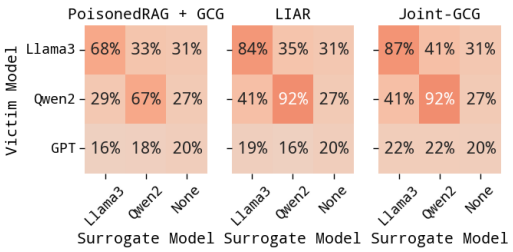

Figure 6: ASR_{gen} on various victim generators, 'None' as surrogate model represents the unoptimized initial samples.

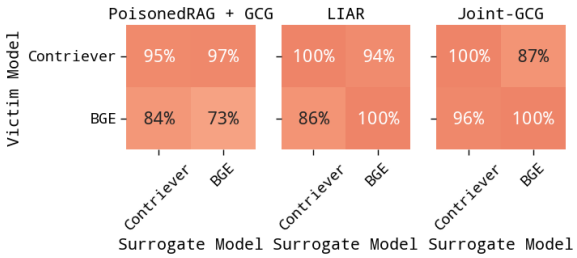

Figure 7: ASR_{ret} on various victim retrievers with Llama3 generator, on MS MARCO.

Ethical Statement

The Joint-GCG framework, while advancing the understanding of RAG system vulnerabilities to improve security, presents potential misuse risks. We have prioritized responsible disclosure, striking a balance between scientific transparency and these concerns. Our research, conducted on controlled datasets and non-production systems, aims to proactively identify vulnerabilities, motivating the development of robust defenses and encouraging the design of security-first RAG systems. We strongly advocate for using these findings for security research and system improvement, not exploitation. Organizations deploying RAG systems should implement comprehensive security measures, including regular audits, content filtering, and continuous monitoring, as understanding these vulnerabilities is crucial to building more secure AI applications.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No. 2024YFF0618800), the National Natural Science Foundation of China Grant No. 62402484, the Basic Research Program of ISCAS Grant No. ISCAS-JCZD-202405, the China Southern Power Grid Company Limited ZBKJXM20240173, and the Youth Innovation Promotion Association Chinese Academy of Sciences. The authors wish to thank Mr. Zhu Qiming for his valuable insights during the conceptualization of the joint-attack methods. They are also grateful to the anonymous reviewers for their constructive feedback.

References

Boulos, M. N. K.; and Dellavalle, R. 2024. NVIDIA's "Chat with RTX" Custom Large Language Model and Personalized AI Chatbot Augments the Value of Electronic Dermatology Reference Material. *JMIR dermatology*, 7(1): e58396.

Chaudhari, H.; Severi, G.; Abascal, J.; Jagielski, M.; Choquette-Choo, C. A.; Nasr, M.; Nita-Rotaru, C.; and Oprea, A. 2024. Phantom: General Trigger Attacks on Retrieval Augmented Language Generation. *arXiv preprint arXiv:2405.20485*.

Cheng, P.; Ding, Y.; Ju, T.; Wu, Z.; Du, W.; Yi, P.; Zhang, Z.; and Liu, G. 2024. TrojanRAG: Retrieval-Augmented Generation Can Be Backdoor Driver in Large Language Models. *arXiv preprint arXiv:2405.13401*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Fan, W.; Ding, Y.; Ning, L.; Wang, S.; Li, H.; Yin, D.; Chua, T.-S.; and Li, Q. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 6491–6501.

Izacard, G.; Caron, M.; Hosseini, L.; Riedel, S.; Bojanowski, P.; Joulin, A.; and Grave, E. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474.

Meduri, K.; Nadella, G. S.; Gonaygunta, H.; Maturi, M. H.; and Fatima, F. 2024. Efficient RAG Framework for Large-Scale Knowledge Bases.

Nashid, N.; Sintaha, M.; and Mesbah, A. 2023. Retrieval-based prompt selection for code-related few-shot learning. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, 2450–2462. IEEE.

Nguyen, T.; Rosenberg, M.; Song, X.; Gao, J.; Tiwary, S.; Majumder, R.; and Deng, L. 2016. Ms marco: A human-generated machine reading comprehension dataset.

OpenAI. 2024. Introducing ChatGPT search. https://openai.com/index/introducing-chatgpt-search. Published: 2024-10-31.

Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; and Shoham, Y. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11: 1316–1331.

Siriwardhana, S.; Weerasekera, R.; Wen, E.; Kaluarachchi, T.; Rana, R.; and Nanayakkara, S. 2023. Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11: 1–17.

Tan, Z.; Zhao, C.; Moraffah, R.; Li, Y.; Wang, S.; Li, J.; Chen, T.; and Liu, H. 2024. " Glue pizza and eat rocks"–Exploiting Vulnerabilities in Retrieval-Augmented Generative Models. *arXiv preprint arXiv:2406.19417*.

Vakayil, S.; Juliet, D. S.; Vakayil, S.; et al. 2024. RAG-Based LLM Chatbot Using Llama-2. In *2024 7th International Conference on Devices, Circuits and Systems (ICDCS)*, 1–5. IEEE.

Xiao, S.; Liu, Z.; Zhang, P.; Muennighoff, N.; Lian, D.; and Nie, J.-Y. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, 641–649.

Xue, J.; Zheng, M.; Hu, Y.; Liu, F.; Chen, X.; and Lou, Q. 2024. BadRAG: Identifying Vulnerabilities in Retrieval Augmented Generation of Large Language Models. *arXiv preprint arXiv:2406.00083*.

Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; et al. 2024. Qwen2 Technical Report. arXiv:2407.10671.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Zhou, S.; Alon, U.; Xu, F. F.; Wang, Z.; Jiang, Z.; and Neubig, G. 2022. Docprompting: Generating code by retrieving the docs. *arXiv preprint arXiv:2207.05987*.

Zou, A.; Wang, Z.; Carlini, N.; Nasr, M.; Kolter, J. Z.; and Fredrikson, M. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

Zou, W.; Geng, R.; Wang, B.; and Jia, J. 2024. Poisonedrag: Knowledge poisoning attacks to retrieval-augmented generation of large language models. *arXiv preprint arXiv:2402.07867*.