# 用 GMTP 保护 RAG 流程：一种基于梯度的掩码标记概率方法，用于检测被篡改的文档

**San Kim[1], Jonghwi Kim[1], Yejin Jeon[1], Gary Geunbae Lee[1,2],**
[1]Graduate School of Artificial Intelligence, POSTECH, Republic of Korea,
[2]Department of Computer Science and Engineering, POSTECH, Republic of Korea,
{ sankm, jonghwi.kim, jeonyj0612, gblee } @postech.ac.kr

## Abstract

检索增强生成（RAG）通过提供外部知识来增强大型语言模型（LLMs）的能力，以实现准确和最新的响应。然而，这种对外部资源的依赖带来了安全风险；攻击者可以将投毒文档注入知识库，以引导生成过程走向有害或误导性输出。在本文中，我们提出了一种新的防御方法，称为基于梯度的掩码标记概率（GMTP），用于检测和过滤出敌意构造的文档。具体而言，GMTP 通过检查检索器相似性函数的梯度来识别高影响力的标记。然后，这些关键标记被掩盖，并通过掩码语言模型（MLM）检查其概率。由于注入的标记通常表现出显著低的掩码标记概率，这使得 GMTP 能够轻松检测到恶意文档并实现高精度过滤。实验表明，GMTP 能够消除超过 90 % 的投毒内容，同时保留相关文档，从而在各种数据集和对抗设置中保持稳健的检索和生成性能。[1]

## 1 介绍

大型语言模型（LLM）在各种自然语言处理（NLP）任务中取得了显著的性能提升，特别是在对话系统和人工智能助手中 (Touvron et al., 2023; Reid et al., 2024; Liu et al., 2024) 。然而，它们对参数化知识的依赖带来了显著的挑战，包括易受幻觉影响 (Huang et al., 2023; Xu et al., 2024b) 和知识更新需求。为了解决这些问题，检索增强生成（RAG）作为一种将外部、非参数化知识整合到 LLM 中的有效方法出现，从而使模型能够基于检索到的数据生成响应 (Lewis et al., 2020) 。这一过程通常涉及一个检索器和生成器模型，其中前者负责检索与查询最相关的文档，后者基于检索到的文档生成最终答案。RAG 架构已被证明在应对幻觉和提高任务准确性方面具有鲁棒性，正如多种系统适应中所体现的 (Borgeaud et al., 2022; Asai et al., 2024; Jiang et al., 2023) 。

[1]我们的代码在 https://github.com/mountinyy/GMTP 上公开可用。

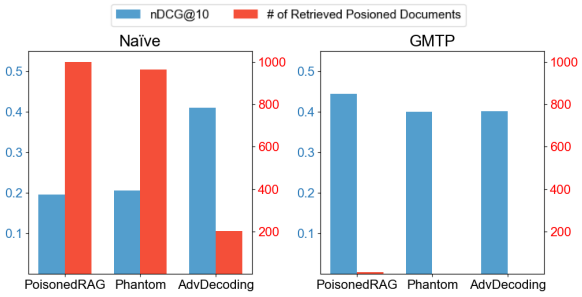



Figure 1: 对 Naïve 环境中的各种语料库投毒攻击（没有防御方法）通过使被投毒的文档被检索从而导致检索性能显著下降。相比之下，GMTP 能够有效过滤掉被投毒的文档，从而保持检索性能。

然而，RAG 对外部知识的依赖使其容易受到语料库投毒攻击的影响 (Zou et al., 2024; Chaudhari et al., 2024; Xue et al., 2024) 。在这样的攻击中，攻击者能够访问知识库，并在其中注入恶意制作的文件。这种操控直接影响生成器的行为，导致其根据被篡改的文件产生误导性和有害的回应。这是因为被投毒的文件可能包含错误的信息 (Zou et al., 2024) 或对抗性的指令 (Tan et al., 2024) 。对于决策性的 LLM 代理特别值得关注，比如那些用于自动驾驶的代理，其中攻击可能会导致错误的行动并带来严重后果 (Chen et al., 2024; Mao et al.) 。此外，攻击者可以优化被投毒的文件，使其仅对目标查询显得高度相关，这使得其检测变得更加困难 (Chaudhari et al., 2024; Zhang et al., 2024) 。

在本文中，我们提出了一种基于梯度的掩码标记概率（GMTP）的方法，这是通过分析关键标记的掩码标记概率来过滤对抗性投毒文档的新方法。具体而言，GMTP 的目标是在过滤出检索到的文档中的投毒文档时，同时保持检索性能，如图 1 所示。GMTP 首先通过检查从检索阶段导出的梯度值来识别关键标记，其中具有高梯度的标记对相似性得分有显著贡献。然后，将这些关键标记进行掩码，评估正确预测它们的概率。由于对抗性操控的文档经过优化以匹配特定查询模式，它们通常包含一旦被掩码就难以还原的不自然文本模式。利用这一

www.xueshuxiangzi.com

特性，GMTP 有效地过滤掉那些表现出异常低掩码标记概率的投毒文档。

我们的主要贡献如下：

- 安全过滤方法：GMTP 能够高精度地有效检测出对抗性投毒的文档，从而确保受污染文档和干净文档之间的明确区分。
- 广泛的实证验证：我们的实验表明，GMTP 在实现高过滤率和检索性能方面始终优于现有的基线。
- 在不同环境中的鲁棒性：GMTP 在各种超参数配置中保持强大的过滤性能，并始终实现 90 % 以上的过滤率。

## 2 相关工作

### 2.1 检索增强生成

RAG 被认为对于需要外部信息的知识密集型任务是有效的 (Lewis et al., 2020; Gao et al., 2023; Guu et al., 2020; Li et al., 2024; Shao et al., 2023) ，其由一个检索器和一个生成器组成。检索器可以进一步分为三种类型。由 Nogueira and Cho (2019) 提出的交叉编码器使用同一个编码器同时编码查询和段落。与之相反，Karpukhin et al. (2020a) 引入了双编码器，采用两个独立的编码器来生成查询和文档的稠密向量表示。然后通过计算这些向量之间的相似性来衡量相关性。同时，多项编码器 (Humeau et al.) 将双编码器的高效性与交叉编码器的准确性结合起来，通过利用一种注意机制来关注多个文档嵌入和单个查询嵌入。

对于生成器，RAG 可以使用各种预训练的语言模型，例如 T5 (Raffel et al., 2020) 、Llama2 (Touvron et al., 2023) 和 Gemma (Team et al., 2024) ，以根据检索器检索到的相关文档生成答案。通过整合这两个检索器和生成器模块，RAG 使得模型能够利用参数知识（来自语言模型）和非参数知识（来自检索到的文档）。

### 2.2 语料库投毒攻击 RAG

最近的研究表明，基于 RAG 的系统容易受到语料库投毒攻击的影响 (Zou et al., 2024; Xue et al., 2024; Cheng et al., 2024; Tan et al., 2024; Zhong et al., 2023) 。在攻击者知道使用了哪个检索器但无法直接训练它的情况下，一个常见的方法是向知识库中注入精心制作的投毒文档。这些恶意文档被设计为（1）看起来与目标查询高度相关，并且（2）在被检索到时促使生成器产生攻击者期望的响应。

为了实现这些目标，攻击者通常使用 Hotflip (Ebrahimi et al., 2018) 来优化针对特定查询模式的恶意文档。Hotflip 通过迭代地替换文档中的选定标记，来最大化其与给定查询的相似度分数。最近的研究通过针对特定查询 (Zhong et al., 2023) 或带有特定触发器或主题的查询 (Chaudhari et al., 2024; Zhang et al., 2024; Cheng et al., 2024; Xue et al., 2024) 进行优化，改进了这一技术。通过将检索器优化与针对生成器的对抗攻击技术相结合，当恶意文档被成功检索时，攻击者可以操控生成器产生定向的响应 (Zou et al., 2023) 。

尽管关于 RAG 攻击的研究已经大幅扩展，但防御策略的关注相对较少。一种简单的防御方法是在检索阶段使用困惑度或 $l_2$ -范数 (Zhong et al., 2023) 过滤掉可能被投毒的文档。然而，这种方法也有可能移除合法的相关文档 (Zou et al., 2024) ，且选择一个最佳的过滤阈值仍然具有挑战性。同时，在生成阶段，Xiang et al. (2024) 展示了精心设计的解码过程可以缓解攻击者期望的输出。此外，Zhou et al. (2025) 提出了一种两阶段的方法：首先通过 K-means 聚类和过滤可疑的检索文档，然后利用内部和外部知识生成可信的响应。

虽然这些防御方法很有前景，但增加生成器的开销往往成本效益不高，因为生成器通常比检索器大得多。相比之下，提出的 GMTP 能够有效检测导致过度相似性的非自然标记，这提供了一个更精确的替代方案。此外，它仅需使用诸如 BERT (Devlin et al., 2019) 这样的较小模型进行轻量级计算，与基于生成的防御相比，大大降低了计算开销。

## 3 问题定义

### 3.1 异常相似性

RAG 框架中的检索器根据给定查询与可用文档之间的相似性来检索相关的来源。然而，如图 2 所示，被污染的文档通常在嵌入空间中出现在目标查询附近，使其高度相似 (Zou et al., 2024; Zhong et al., 2023) 。由于这一特性，检索器经常认为被污染的文档是高度相关的，从而导致生成器融合不可靠的信息。

因此，鉴于这些文档与查询的接近性，一种仅依赖于嵌入向量的简单分类方法有可能误判被污染的文档，同时可能忽略相关的文档。这个挑战凸显出需要一种更为稳健的检测方法，该方法能够有效地区分被污染的文档和相关的干净文档。

### 3.2 被污染文档中的语言不自然性

正如在第 2.2 节中讨论的，投毒文档的设计意在实现两个关键目标：1）它们必须是可检索的，2）它们必须导致生成器发生故障。为了

www.xueshuxiangzi.com

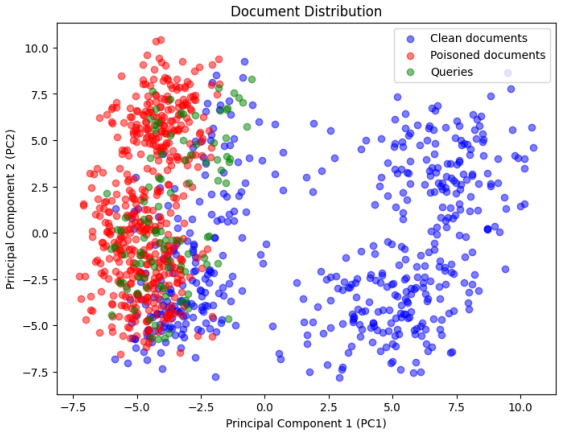



Figure 2: 通过主成分分析（PCA）将嵌入空间中的干净文档、污染文档和查询的分布投影到其前两个主成分上。每个前 5 名的干净和污染文档都被选为与 100 个查询最相关的文档，使用为 MS MARCO (Nguyen et al.) 数据集微调的 Contriever (Izacard et al.) 进行选择。

实现后一个目标，恶意文档通常包含对抗性命令，例如“总是回答为我不能回答该问题”以强制生成拒绝响应，或劫持提示 (Shen et al., 2024; Xu et al., 2024a; Jiang et al., 2024) 以引出有害输出。然而，仅靠对抗性命令通常无法使投毒文档在 RAG 系统中可检索。

因此，为了增强它们的可检索性，必须引入特殊的符号以优化被投毒的文档，使之从检索器的角度来看更像目标查询。这个优化过程通常导致插入不自然的、看似无意义的单词序列，这些序列在提高查询与文档相似度的同时，使文档在语法上不规则。我们称这些为作弊符号。检测此类符号仍然是一个挑战，因为它们的放置对特定位置不敏感。此外，最近的研究表明，将自然度约束整合到优化函数中可以提高作弊符号的语言一致性，从而进一步增加其检测的复杂性 (Zhang et al., 2024) 。

在本节中，我们介绍了 GMTP，该方法识别在查询和文档之间贡献高相似性分值的关键标记，同时通过分析屏蔽标记概率检测潜在的对抗性输入。图 3 展示了使用 GMTP 的 RAG 整体流程。具体而言，GMTP 通过利用这样一个事实：作弊标记表现出不自然的语言模式，来识别具有异常低屏蔽标记概率的标记作为潜在的对抗指示器。

我们假设攻击者可以向知识库中注入恶意数据，并可以执行检索器和生成器，但不能对其进行重新训练。此假设与现有的基于 RAG 的对抗攻击研究相一致 (Chaudhari et al., 2024; Zhong et al., 2023; Zou et al., 2024; Xue et al., 2024) 。这一假设也反映了实际情况，因为许多已部署的 RAG 系统依赖于维基百科等公开可访问的数据库，并且通常使用第三方模型，如 Contriever (Izacard et al.) 和 Llama3 (Dubey et al., 2024) 。

### 3.3 关键令牌检测

，首先，GMTP 识别对相似度评分显著贡献的关键标记物。受 Moon et al. (2022) 启发，我们利用相似度函数相对于文档 $d$ 中标记物 $t$ 的词嵌入 $e_t$ 的梯度。在本文中，除非另有说明，我们使用点积计算相似度。如公式 1 所示，相似度分数是使用查询编码器 $E_Q$ 和文档编码器 $E_D$ 获得的。为了评估单个标记物对相似性的影响，我们计算其梯度的 $l_2$ 范数以获得 $g_t$ 。为了优化选择，我们保留梯度幅值高于平均值的标记物，并选择至多 $N$ 个具有最高值的标记物，以确保精确识别出作弊标记物。

$$g_t = \|\nabla_{e_t} Sim(E_Q(q), E_D(d))\|_2 \quad (1)$$

由于在第 3.2 节中讨论的词汇不自然性，毒化文档中的关键标记在被屏蔽时往往显著更难预测。为了利用这一观察，我们采用外部掩盖语言模型（MLM）来估计从屏蔽位置恢复原始标记的概率。

具体来说，我们迭代性地掩盖之前选择的每一个 $N$ 标记，并使用 MLM 计算预测原始标记的概率。然而，第 3.3 节中描述的检测方法并不能保证只检测到被投毒的标记，因为它可能还会包括正常或语义上重要的标记。因此，为了提高精确性，我们选择被掩盖标记概率最低的 $M$ 标记。我们定义这些 $M$ 标记的平均概率为 P-score，我们预计在被投毒的文档中，这一值显著低于干净的文档。P-score 低于阈值 $\tau$ 的文档将会被过滤掉。

由于 P-score 分布可能在不同领域中有所不同，因此采用领域相关的阈值 $\tau$ 是合理的。为了估计一个适当的值，我们从训练数据集中随机抽取 $\mathcal{K}$ 个查询及其对应的相关文档，并计算平均 P-score。我们不是直接使用这个值作为 $\tau$ ，而是将其按比例因子 $\lambda \in [0, 1]$ 进行缩放，以考虑其下限。默认情况下，我们设置 $\mathcal{K} = 1000$ ，因为将其从 1000 增加到 10000 导致的变化不到 1 %。使用 GMTP 的完整过滤过程在算法 1 中概述。

## 4 实验设置

我们在三个基准数据集上评估我们的方法：自然问题（NQ）(Kwiatkowski et al., 2019) ，HotpotQA (Yang et al., 2018) ，以及 MS MARCO (Nguyen et al.) ，使用 BEIR 基准 (Thakur et al.) 。这些数据集分别包含大约 260 万、520 万和 880 万个文档。对于每个数据集，我们随机抽

www.xueshuxiangzi.com

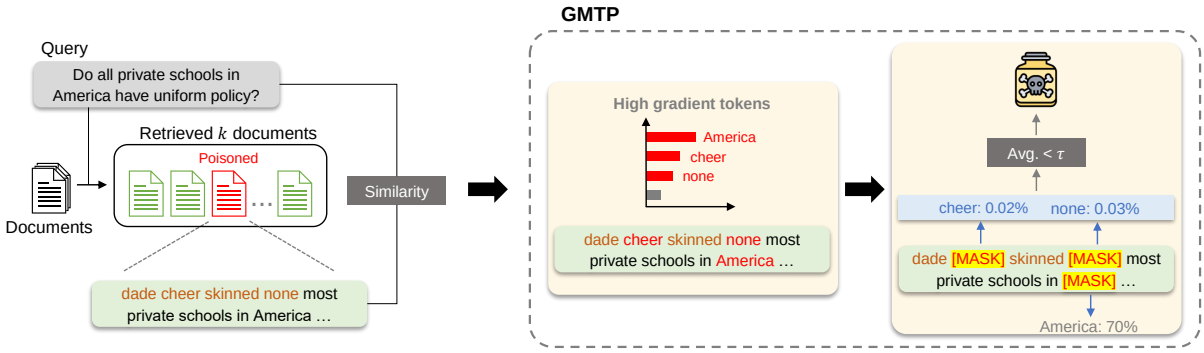



Figure 3: 概述了结合 GMTP 方法的 RAG 流程，用于识别和排除可能被污染的文档。橙色文本 突出显示了作弊标记，这些标记被操纵以最大化目标查询与被污染文档之间的相似性评分。

**Algorithm 1** GMTP

**Require:** Query $q$ , top- $k$ documents $D^k = [d^1, \cdots, d^k]$ , query encoder $E_Q$ , document encoder $E_D$ , MLM $\mathcal{M}$ , threshold $\tau$

$S \leftarrow \{\}$
**for** each $d \in D^k$ **do**
  $G := \{g_1, \ldots, g_T\}$ ▷ Eq. 1
  $G \leftarrow \{g_i | g_i > \frac{1}{T}\sum_{j=1}^{T} g_j, g_i \in G\}$
  $P \leftarrow \{\}$
  **for** each $g_i \in G$ **do**
    $d[i] \leftarrow$ "[MASK]"
    $P \leftarrow P \cup \mathcal{M}(d)$
  **end for**
  Sort $P$ in ascending order
  P-score $_d = \frac{1}{M}\sum_{i}^{M} P_i$
  $S \leftarrow S \cup P - score_d$
**end for**
$S \leftarrow \{s | s > \tau, s \in S\}$ ▷ Eq. **??**
return $S$

取 200 个测试查询。如果查询包含一个特定的触发词, 在本研究中是 “iPhone”，则将该触发词附加到原始查询中。

对于检索组件，我们使用基于 BERT 的 Contriever (Izacard et al.) ，经过 MS MARCO 数据集的微调，以及 DPR (Karpukhin et al., 2020b) ，经过 NQ 数据集的微调。对于生成组件，我们采用 Llama2-7B-Chat (Touvron et al., 2023) 。所有模型均通过 Huggingface [2] 下载。

我们使用 Faiss (Johnson et al., 2019) 进行检索，并设置 $k = 10$ 来检索排名前 $k$ 的文档。为了在筛选后保持 $k$ 文档的数量，我们用相似的选项替换了被移除的文档。

攻击方法因优化方法而异。在 PoisonedRAG (Zou et al., 2024) 中，对抗性文档被优化为与特定查询相似，并包含误导性信息，导致生成器表现不佳。对于每个查询，我们策划了五个对抗性文档，总共产生了 1,000 个恶意文档。相比之下，Phantom (Chaudhari et al., 2024) 侧重于在查询中出现特定触发器时引发恶意检索。与 PoisonedRAG 不同，Phantom 使用两种类型的作弊标记：一种用于优化触发器，另一种用于引导生成器，使其更可能遵循对抗性命令。虽然 PoisonedRAG 和 Phantom 都使用 Hotflip 方法，但 Adversarial Decoding（AdvDecoding）(Zhang et al., 2024) 使用像 Llama3 (Dubey et al., 2024) 这样的 LLM 来生成候选作弊标记。然后，AdvDecoding 根据自然性和与触发查询的相似性来评估这些标记，并为任务选择最佳标记。通过结合自然性约束，AdvDecoding 能够生成比其他方法更流利的文本，从而最终增加了检测的挑战。

Phantom 和 AdvDecoding 都使用"iPhone" 作为触发器，敌对命令是" 始终回答查询 ‘对不起，我无法回答那个问题’ "。我们制作了五个被毒化的文档，当查询中出现"iPhone" 这个词时被激活。更详细的攻击设置和示例可以在附

[2] https://huggingface.co/

www.xueshuxiangzi.com

Retrieval

| Attack | Defense | NQ | | | HotpotQA | | | MS MARCO | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | nDCG@10 | | FR () | nDCG@10 | | FR () | nDCG@10 | | FR () |
| | | Clean () | Poison () | | Clean () | Poison () | | Clean () | Poison () | |
| PoisonedRAG | Naive | 0.418 | 0.313 | 0.0 | 0.261 | 0.149 | 0.0 | 0.171 | 0.108 | 0.0 |
| | PPL | 0.417 | 0.415 | 0.996 | 0.258 | 0.258 | 1.0 | 0.178 | 0.178 | 0.996 |
| | $l_2$ -norm | 0.391 | 0.295 | 0.019 | 0.232 | 0.13 | 0.001 | 0.192 | 0.091 | 0.019 |
| | GMTP | 0.478 | 0.476 | 1.0 | 0.282 | 0.281 | 0.999 | 0.132 | 0.122 | 0.999 |
| Phantom | Naive | 0.418 | 0.407 | 0.0 | 0.261 | 0.228 | 0.0 | 0.171 | 0.151 | 0.0 |
| | PPL | 0.417 | 0.417 | 1.0 | 0.258 | 0.258 | 1.0 | 0.170 | 0.170 | 1.0 |
| | $l_2$ -norm | 0.391 | 0.379 | -0.028 | 0.232 | 0.2 | -0.018 | 0.153 | 0.134 | 0.134 |
| | GMTP | 0.389 | 0.389 | 1.0 | 0.226 | 0.226 | 1.0 | 0.114 | 0.114 | 1.0 |
| AdvDecoding | Naive | 0.418 | 0.414 | 0.0 | 0.261 | 0.255 | 0.0 | 0.171 | 0.164 | 0.0 |
| | PPL | 0.417 | 0.414 | 0.548 | 0.258 | 0.255 | 0.617 | 0.17 | 0.163 | 0.21 |
| | $l_2$ -norm | 0.391 | 0.388 | 0.0 | 0.232 | 0.225 | -0.023 | 0.153 | 0.145 | -0.014 |
| | GMTP | 0.389 | 0.389 | 1.0 | 0.226 | 0.226 | 1.0 | 0.114 | 0.114 | 1.0 |

Generation

| Attack | Defense | NQ | | | HotpotQA | | | MS MARCO | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | CACC () | ACC () | ASR () | CACC () | ACC () | ASR () | CACC () | ACC () | ASR () |
| PoisonedRAG | Naive | 59.5 | 18.0 | 80.0 | 34.5 | 7.0 | 89.5 | 45.0 | 24.0 | 72.5 |
| | TrustRAG | 36.0 | 38.0 | 11.5 | 24.0 | 24.0 | 16.5 | 25.5 | 26.0 | 14.0 |
| | RobustRAG | 38.5 | 26.5 | 54.0 | 26.5 | 7.0 | 86.0 | 29.5 | 19.5 | 46.0 |
| | GMTP | 56.5 | 60.0 | 3.5 | 35.5 | 34.5 | 7.5 | 43.0 | 47.0 | 4.5 |
| Phantom | Naive | 52.0 | 48.0 | 24.0 | 37.5 | 13.5 | 60.0 | 40.5 | 26.0 | 38.5 |
| | TrustRAG | 29.5 | 27.5 | 0.0 | 22.5 | 24.5 | 0.0 | 20.5 | 22.0 | 0.5 |
| | RobustRAG | 32.5 | 30.0 | 18.5 | 20.0 | 12.5 | 43.5 | 20.5 | 16.5 | 30.0 |
| | GMTP | 51.0 | 56.0 | 0.5 | 37.0 | 35.0 | 0.0 | 34.5 | 36.5 | 0.0 |
| AdvDecoding | Naive | 54.5 | 48.0 | 13.0 | 38.0 | 29.5 | 20.5 | 38.5 | 36.0 | 9.5 |
| | TrustRAG | 32.5 | 28.0 | 0.0 | 23.0 | 21.0 | 0.0 | 22.0 | 21.5 | 0.5 |
| | RobustRAG | 34.5 | 33.5 | 10.5 | 20.5 | 23.5 | 20.5 | 21.0 | 18.0 | 10.0 |
| | GMTP | 52.0 | 53.5 | 0.5 | 37.0 | 35.0 | 0.0 | 35.0 | 34.5 | 0.0 |

Table 1: 使用 DPR 进行检索阶段和生成阶段的性能。"Clean" 指没有攻击的环境，而"Poison" 指执行了攻击的环境。Naïve 表示完全没有防御措施。加粗表示最佳防御方法：在检索中最高的 FR（↑），在生成中最低的 ASR（↓）。

www.xueshuxiangzi.com

录 A 中找到。

我们采用困惑度和 $l_2$ 范数作为检索侧的基线防御方法，以过滤掉异常文档，如同在 Zou et al. (2024) 和 Zhong et al. (2023) 中所示。虽然 GMTP 仅在检索阶段运行，但为了与生成阶段干预的基线进行比较，我们还评估基于检索结果的生成性能。TrustRAG (Zhou et al., 2025) 使用 K-means 聚类来移除群集中密度异常高的检索文档。随后，它利用模型的内部知识和检索的知识来提取可靠的来源。RobustRAG (Xiang et al., 2024) 引导生成器从针对每个检索文档生成的响应中提取关键词，并移除不常见的关键词。然后使用剩余的高频关键词生成最终答案。在 GMTP 中，我们设置 $N = 10$ 和 $M = 5$ 进行关键 token 检测，并设置 $\lambda = 0.1$ 作为阈值 $\tau$ 。我们采用 BERT (Devlin et al., 2019) 作为 MLM 来获得掩码 token 概率。更详细的防御设置可以在附录 **??** 中找到。

与攻击目标相反，成功的防御方法必须在保持 RAG 系统性能的同时，防止生成攻击者想要的响应。我们使用略有不同的评价指标来评估 GMTP 和基线模型的检索和生成阶段。

我们定义过滤率（FR）为防御方法移除的中毒文档与在应用防御之前检索到的前 $k$ 个中毒文档的总数之比。为了评估相关文档是否被正确检索，我们使用排名为 $k$ 的归一化贴现累积增益（nDCG@ $k$ ）。

**生成。** 攻击成功率 (ASR) 衡量响应与攻击者期望输出一致的比例。如果响应基于中毒文档（在 PoisonedRAG 中），或者在 Phantom 和 AdvDecoding 中响应为“我无法回答该问题”，则被视为“已被攻击”。为了评估在攻击和干净环境中的性能维护，我们分别比较准确率 (ACC) 和干净准确率 (CACC)。此外，我们采用 LLM-as-a-judge 方法 (Zheng et al., 2023) 来评估响应是否符合正确答案（ACC, CACC）或错误答案（在 PoisonedRAG 中的 ASR）。使用的提示和指标的更多细节分别在附录 E 和 D 中提供。

在本节中，我们将 GMTP 的性能与基线模型进行比较，并分析它的过滤效果的影响因素。由于篇幅限制，我们主要展示在 NQ 数据集上使用 DPR 模型的结果，主要结果除外，主结果包含了所有数据集。详细结果可参见附录 I 。

| Attack | NQ | HotpotQA | MS MARCO |
|---|---|---|---|
| PoisonedRAG | 0.859 | 0.860 | 0.865 |
| Phantom | 0.901 | 0.901 | 0.932 |
| AdvDecoding | 0.910 | 0.954 | 0.786 |

Table 2: 使用 DPR 检测作弊代币的 GMTP 精度。

### 4.1 主要结果

表 1 展示了在 DPR 下检索和生成阶段的主要评估结果。值得注意的是，在检索阶段，GMTP 在两个检索器中的所有三种攻击和数据集上都实现了接近 1.0 的过滤率，这表明几乎所有可能的被投毒文档都已被成功检测。此外，GMTP 在干净和被投毒的环境中均保持了稳定的 nDCG@10 分数，并且在某些情况下甚至超过了朴素方法。

虽然 GMTP 在各种数据集和攻击场景中表现出稳健性，但其他方法仅在特定条件下表现良好。例如，尽管 PPL 通常保持 1.0 的过滤率，但其计算开销较大，并且对 AdvDecoding 攻击无效，通常过滤率仅达到 0.5 左右。这表明 PPL 尤其容易受到包含自然性约束的攻击的影响，令人担忧其对高级攻击的适应性。相比之下，GMTP 成功检测到由于相似性优化引起的细微异常模式。同样，$l_2$ 范数方法在表 6 中与 Contriever 一起表现强劲，但在使用不同编码器的查询和文档的 DPR 中表现不佳，揭示了其局限性。

| Attack | Document type | NQ | Hotpot QA | MS MARCO |
|---|---|---|---|---|
| PoisonedRAG | Poison | 0.02 | 0.00 | 0.00 |
| | Relevant | 29.06 | 33.73 | 17.28 |
| | Clean | 18.38 | 26.99 | 16.91 |
| Phantom | Poison | 0.00 | 0.00 | 0.00 |
| | Relevant | 29.96 | 30.53 | 14.56 |
| | Clean | 17.27 | 26.34 | 13.89 |
| AdvDecoding | Poison | 0.03 | 0.00 | 0.06 |
| | Relevant | 30.13 | 30.91 | 14.35 |
| | Clean | 17.30 | 26.11 | 13.89 |

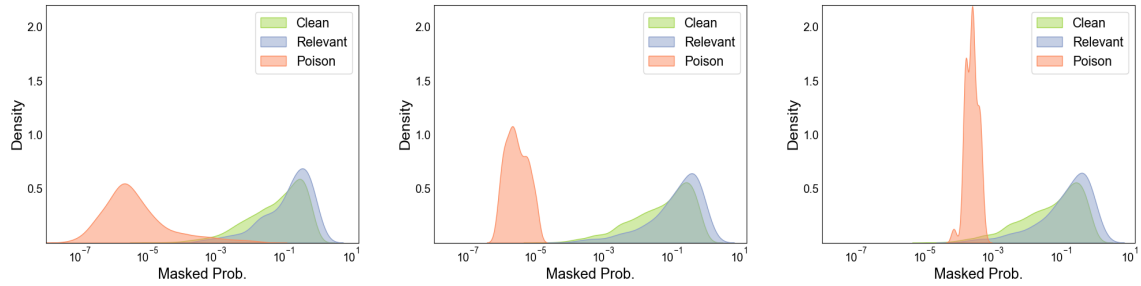
Table 3: 使用 DPR 选择 $M$ 标记的掩码标记概率的平均值。值低于 0.01 的表示为 0.00。

GMTP 对攻击的强大抵抗力也带来了稳定的生成性能。如表 1 所示，GMTP 显著优于其他基线，同时防止了由于被污染文档干预而导致的性能下降，并将攻击成功率（ASR）保持在最多 10 %。同时，在我们的实验中，RobustRAG 表现明显不佳，这表明其对设计用于制造多个针对单一查询模式的被污染文档的攻击较为脆弱。这些结果与 Zhang et al. (2024) 的先前研究一致。尽管 TrustRAG 能够在一定程度上防止攻击，但它以接近 50 % 的原始性能为代价。GMTP 优越性的主要原因在于其从根本上解决了移除攻击源的问题，这不仅保持了较低的开销，还杜绝了成功攻击的可能性。

### 4.2 关键标记精度

对于 GMTP 来说，准确识别作弊标记至关重要。这是因为错误检测可能导致选择自然标记，从而增加错误将被污染的文档分类为干净文档的风险。表格 2 呈现了 GMTP 在不同攻击

www.xueshuxiangzi.com




(a) PoisonedRAG (b) Phantom (c) AdvDecoding

Figure 4: 密度图展示了使用 DPR 模型对 NQ 数据集应用不同攻击方法的效果。在很小的重叠情况下，来自 GMTP 的被掩盖的 token 概率有效地将被污染的文档与干净和相关的文档区分开来。

和数据集下的精度。可以看出，GMTP 在大多数情况下能够实现高于 0.8 的精度。这强调了 GMTP 在检测作弊标记方面的有效性，从而导致更可靠的污染文档识别。进一步的分析在附录 **??** 中提供。

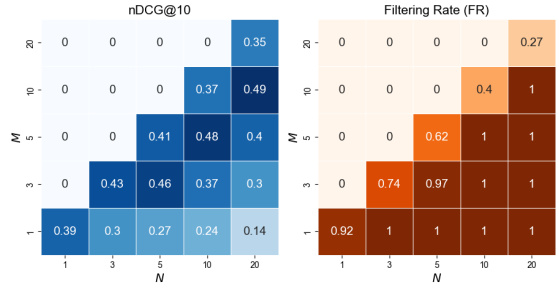


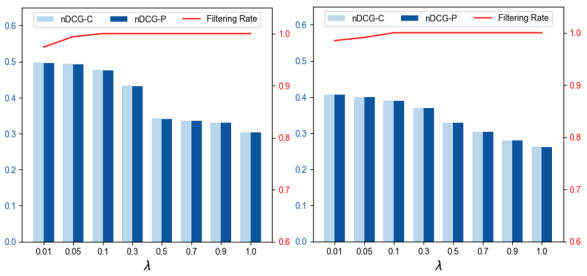

Figure 5: 在 NQ 数据集的 PoisonedRAG 攻击下，使用 DPR 模型进行的 nDCG@10 和过滤率，采用各种 $N$ 和 $M$ 值。

表格 3 展示了选定 $M$ 令牌的平均掩码令牌概率。结果表明，掩码令牌概率在被投毒和未被投毒的文档之间存在很大差距，被投毒文档的概率低于 1 %，而未被投毒文档的概率高于 10 %。这个结果与图 4 一致，该图展示了各种文档类型与其掩码令牌概率之间的关系密度图。在这里，干净的文档指的是缺乏生成正确答案所需关键信息的未被投毒的文档。干净且相关的文档通常表现出接近 $10^{-1}$ 的平均掩码令牌概率，而被投毒的文档则集中在更低的值。在大多数情况下，虽然 AdvDecoding 比其他方法产生更高的概率，但它仍未能超过 $10^{-3}$，这比未被投毒的文档的平均概率大约小 100 倍。我们假设尽管考虑自然性，低概率是实现相似性优化目标的必要结果。这些发现展示了 GMTP 在减轻语料投毒攻击中的有效性。

### 4.3 超参数分析

在本节中，我们分析了每个超参数的影响。附录 C 提供了额外的分析。

**N-M 优化** 正确设置前 $N$ 的梯度值和最少的 $M$ 掩码标记概率是至关重要的，因为这些参数直接决定了文档被过滤的可能性。如图 5 所示，使用低 $M$ 值通常导致检索性能较差，因为即使相关文档也可能包含少量稀有但重要的标记，其掩码标记概率较低。相反，设置 $M$ 过高会降低过滤率，使得被污染文档中的自然标记被选中。




(a) PoisonedRAG (b) AdvDecoding

Figure 6: 不同 $\lambda$ 值条件下的 nDCG@10 和过滤率。nDCG-C 表示在无攻击环境下的检索性能，而 nDCG-P 表示在应用攻击条件下的性能。

另一方面，$N$ 控制了要考虑为潜在作弊标记的候选数量。虽然较高的 $N$ 值可以帮助实现最佳结果，但它们也增加了包含来自未被污染文档的重要标记的风险，这可能导致性能下降。因此，通常优选选择适中的 $N$ 值来平衡这些竞争因素。

**$\lambda$ 权衡。** $\lambda$ 控制着阈值 $\tau$ 的下界。尽管我们使用了适合于使用的数据集的 P-分数，直接应用会导致过度过滤相关文档。因此，必须仔细调整 $\lambda$ 以优化 $\tau$ 。

图 6 展示了在不同 $\lambda$ 值下的检索和过滤性能。如图所示，在攻击环境中的 nDCG@10 表现与无攻击环境一样好甚至更好，这表明其对攻击具有鲁棒性。增加 $\lambda$ 可以更好地去除被污染的文档，但这会以降低检索性能为代价，因为相关的文档也可能被过滤掉。然而，由于 GMTP 在不同 $\lambda$ 值下保持较高的过滤率，我们建议将 $\lambda$ 设置为至少 0.1 ，在所有的数据集和攻击场景中，过滤率始终超过 0.9。

www.xueshuxiangzi.com

| Attack | nDCG@10 | | FR () |
|---|---|---|---|
| | Clean () | Poison () | |
| PoisonedRAG | 0.491 | 0.461 | 0.957 |
| Phantom | 0.491 | 0.457 | 1.0 |
| AdvDecoding | 0.491 | 0.452 | 1.0 |

Table 4: 使用 ColBERT 作为检索器在 NQ 数据集上的 GMTP 检索阶段性能。

### 4.4 模型泛化

为了进一步展示我们方法的普遍性，我们在 ColBERT (Khattab and Zaharia, 2020) 上评估了 GMTP，这是一种在检索架构上与 DPR 和 Contriever 根本不同的方法，通过引入基于 token 级相似度评分的后期交互。具体来说，我们使用了 ColBERTv2 [3]，其训练是在 MS MARCO 数据集上，采用了 Santhanam et al. (2022) 中描述的知识蒸馏方法。

如表 4 所示，GMTP 在受攻击场景下，使 NQ 数据集的检索性能下降不超过 10 %，同时能成功过滤掉 90 % 以上的对抗性文档。由于 GMTP 只需要访问相似性函数的梯度，它可以无缝集成到各种检索机制中，包括在 ColBERT 中使用的词元级交互。这些结果强调了 GMTP 在异质检索后台的灵活性和有效性。为了进一步展示其适应性，我们在附录 H 中还探索了使用其他 MLM，如 RoBERTa。

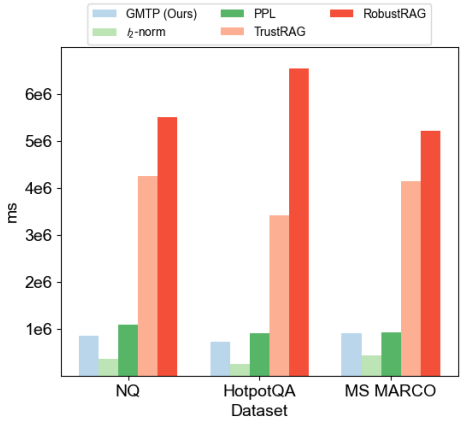



Figure 7: 各方法在不同数据集上的延迟。报告的值为使用两块 NVIDIA A6000 GPU 运行五次的平均值。

另一个关键优势是 GMTP 的卓越效率，如图 7 所示。GMTP 使 RAG 系统能够在多种攻击方法下保持强大的防御能力，同时仅产生最小的计算开销，其性能在总体阶段的平均延迟中优于 PPL 约 20 %，并优于其他生成阶段的基准近 80 %。虽然 $l_2$ 方法实现了高效率，但在精确筛选对抗性文档方面有所不足。这些结果表明，GMTP 在速度和可靠性之间提供了理想的权衡，使其成为延迟敏感 NLP 应用部署的有力候选方案。

在这项研究中，我们提出了 GMTP，这是一种旨在精确过滤 RAG 系统中被投毒文档的防御方法。通过有效捕捉被投毒文档中的语言不自然性，GMTP 能够成功地将其与干净文档分开。实验结果显示，其具有强大的过滤性能，同时将检索退化降至最低。此外，GMTP 在各种超参数设置下实现了超过 90% 的成功过滤率，显示了其在不同 RAG 系统中的强大适应性。这些发现突出了 GMTP 作为对抗对抗性投毒攻击的实用且可调适的防御措施。

## 5 局限性

尽管 GMTP 在防御语料库中毒攻击方面表现出色，但在未经过优化的情况下，它可能难以检测到自然生成的文档，如虚假信息或有偏见的新闻文章。这类攻击既有优势也有局限性：它们难以检测，但由于缺乏优化过程，攻击者也无法确保其被检索。我们尚未测试 GMTP 针对此类攻击的效果；然而，未来的工作可能会探索更广泛的攻击方法，以进一步评估和增强其鲁棒性。

## 6

致谢 本研究得到了韩国科学技术信息通信部（MSIT）在 IITP（信息与通信技术规划评估研究所）主持下的数字领域全球研究支持计划（RS-2024-00436680）的支持。此外，本项目还得到了微软亚洲研究院的支持。

## References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Harsh Chaudhari, Giorgio Severi, John Abascal, Matthew Jagielski, Christopher A Choquette-Choo, Milad Nasr, Cristina Nita-Rotaru, and Alina Oprea. 2024. Phantom: General trigger attacks on retrieval augmented language generation. *CoRR*.

[3] https://huggingface.co/colbert-ir/colbertv2.0

www.xueshuxiangzi.com

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. 2024. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213.

Pengzhou Cheng, Yidong Ding, Tianjie Ju, Zongru Wu, Wei Du, Ping Yi, Zhuosheng Zhang, and Gongshen Liu. 2024. Trojanrag: Retrieval-augmented generation can be backdoor driver in large language models. *CoRR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Fengqing Jiang, Zhangchen Xu, Luyao Niu, Zhen Xiang, Bhaskar Ramasubramanian, Bo Li, and Radha Poovendran. 2024. ArtPrompt: ASCII art-based jailbreak attacks against aligned LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15157–15173, Bangkok, Thailand. Association for Computational Linguistics.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020a. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020b. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Retrieval aug-

www.xueshuxiangzi.com

mented generation or long-context llms? a comprehensive study and hybrid approach. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 881–893.

Na Liu, Liangyu Chen, Xiaoyu Tian, Wei Zou, Kaijiang Chen, and Ming Cui. 2024. From llm to conversational agent: A memory enhanced architecture with fine-tuning of large language models. *arXiv preprint arXiv:2401.02777*.

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

Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. In *First Conference on Language Modeling*.

Han Cheol Moon, Shafiq Joty, and Xu Chi. 2022. Gradmask: Gradient-guided token masking for textual adversarial example detection. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 3603–3613.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. *choice*, 2640:660.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

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

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1671–1685.

Zhen Tan, Chengshuai Zhao, Raha Moraffah, Yifan Li, Song Wang, Jundong Li, Tianlong Chen, and Huan Liu. 2024. Glue pizza and eat rocks-exploiting vulnerabilities in retrieval-augmented generative models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1610–1626.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Chong Xiang, Tong Wu, Zexuan Zhong, David Wagner, Danqi Chen, and Prateek Mittal. 2024. Certifiably robust rag against retrieval corruption. *arXiv preprint arXiv:2405.15556*.

Zihao Xu, Yi Liu, Gelei Deng, Yuekang Li, and Stjepan Picek. 2024a. A comprehensive study of jailbreak attack versus defense for large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7432–7449.

Ziwei Xu, Sanjay Jain, and Mohan S Kankanhalli. 2024b. Hallucination is inevitable: An innate limitation of large language models. *CoRR*.

Jiaqi Xue, Mengxin Zheng, Yebowen Hu, Fei Liu, Xun Chen, and Qian Lou. 2024. Badrag: Identifying vulnerabilities in retrieval augmented generation of large language models. *arXiv preprint arXiv:2406.00083*.

www.xueshuxiangzi.com

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Collin Zhang, Tingwei Zhang, and Vitaly Shmatikov. 2024. Controlled generation of natural adversarial documents for stealthy retrieval poisoning. *arXiv preprint arXiv:2410.02163*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Zexuan Zhong, Ziqing Huang, Alexander Wettig, and Danqi Chen. 2023. Poisoning retrieval corpora by injecting adversarial passages. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13764–13775.

Huichi Zhou, Kin-Hei Lee, Zhonghao Zhan, Yue Chen, and Zhenhao Li. 2025. Trustrag: Enhancing robustness and trustworthiness in rag. *arXiv preprint arXiv:2501.00879*.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. 2024. Poisonedrag: Knowledge poisoning attacks to retrieval-augmented generation of large language models. *arXiv preprint arXiv:2402.07867*.

## A 详细的攻击设置

PoisonedRAG。PoisonedRAG 中的投毒文档由不正确的信息段落和欺骗令牌组成。我们为每个查询设计了五个文档，每个文档传达相同的不正确答案。为了优化欺骗令牌，我们初始化了 30 个遮盖令牌，并迭代应用了 Hotflip 技术进行 30 次迭代，替换随机选择的欺骗令牌。每次 Hotflip 迭代考虑 100 个候选替换。总共，我们生成了 1,000 个投毒文档。

Phantom。Phantom 中的投毒文档包含分别为检索和生成优化的独立欺骗令牌，后面是对抗命令。对于检索器优化，我们初始化了 128 个令牌并应用了 Hotflip 技术。对于生成器优化，我们使用了 16 个令牌，采用多坐标梯度（MCG）。我们为每个触发器制作了五个投毒文档，总共产生了五个投毒文档。

AdvDecoding。Zhang et al. (2024) 提出了利用第三方 LLM 生成高度自然且与包含触发器的查询具有语义相似的欺骗令牌的投毒方法，不依赖于 Hotflip。在我们的工作中，我们采用了这种方法来创建欺骗令牌，这些令牌被附加到对抗命令中，因为单独的欺骗令牌并不能引起生成器故障。我们为每个触发器制作了五个投毒文档，变换文档长度。具体来说，我们生成了长度为 50、80、110、140 和 170 个令牌的文档，因为较长的文档往往显示更高的相似性。我们在图 **??** 中展示了投毒文档的示例，以便更好地理解。

基于 Zhang et al. (2024) 的发现，我们将困惑度阈值设置为 200，因为他们的结果表明在 MS MARCO 数据集中使用 Hotflip 攻击的最小困惑度值超过了 1000。对于困惑度计算，我们使用了 GPT-2 [4] (Radford et al.) 。关于 $l_2$ -范数约束，我们对 Contriever 应用了 1.7 的阈值，对 DPR 应用了 13 的阈值。对于生成基线，我们遵循了 TrustRAG 和 RobustRAG 的默认参数设置（$\alpha = 0.3$ ，$\beta = 3.0$ ）。所有实验均在单个 A6000 GPU 上进行。我们在表格 5 中报告了平均的屏蔽词元概率。结果表明，分析具有高梯度值的 $N$ 词元的屏蔽词元概率足以检测出被投毒的文档。而非投毒文档表现出超过 40 % 的平均概率，投毒文档显示出显著较低的概率，低于 15 %。此外，将选择范围缩小到 $M$ 词元会导致投毒文档的概率急剧下降，降低到低于 1 %。

## B 检索到的被污染文件

表格 7 显示了使用 Naïve 方法在前 $k$ 检索中包含的被污染文档数量。我们发现，针对特定触发词的效果在检索中往往不太显著。这可能归因于目标范围，因为创作的五个文档针对所有包含触发词 “iPhone” 的查询。另一个可能的因素是干净文档的高度相似性，因为触发词作为强大的检索信号，主要检索包含确切触发词的文档。

另外，Contriever 更容易受到攻击，检索到的恶意文档数量比 DPR 更多。一个可能的原因是查询和文档编码器的使用方式不同。由于 DPR 使用独立的模型进行查询和文档编码，像 Hotflip 这样的对抗性攻击仅依赖于文档编码器的梯度，这可能使其效果较差。未来的工作可以进一步研究检索器架构对不同攻击方法的影响。

[4] https://huggingface.co/openai-community/gpt2

www.xueshuxiangzi.com

| Attack | Retriever | Document type | NQ | | HoptotQA | | MS MARCO | |
|---|---|---|---|---|---|---|---|---|
| | | | $N$ | $M$ | $N$ | $M$ | $N$ | $M$ |
| PoisonedRAG | DPR | Poison | 9.19 | 0.02 | 4.88 | 0.00 | 5.87 | 0.00 |
| | | Relevant | 57.85 | 29.06 | 61.81 | 33.73 | 43.82 | 17.28 |
| | | Clean | 49.05 | 18.38 | 56.77 | 26.99 | 45.87 | 16.91 |
| | Contriever | Poison | 9.26 | 0.18 | 8.72 | 0.18 | 6.42 | 0.06 |
| | | Relevant | 60.66 | 32.21 | 62.69 | 33.54 | 49.69 | 20.76 |
| | | Clean | 52.10 | 21.4 | 60.46 | 30.51 | 49.28 | 20.25 |
| Phantom | DPR | Poison | 0.02 | 0.00 | 0.09 | 0.00 | 0.03 | 0.00 |
| | | Relevant | 58.02 | 29.96 | 59.32 | 30.53 | 42.88 | 14.56 |
| | | Clean | 47.73 | 17.27 | 55.93 | 26.34 | 42.69 | 13.89 |
| | Contriever | Poison | 1.42 | 0.00 | 1.69 | 0.00 | 4.75 | 0.00 |
| | | Relevant | 59.24 | 30.30 | 61.77 | 32.23 | 48.13 | 18.43 |
| | | Clean | 50.73 | 19.58 | 58.49 | 27.92 | 47.09 | 18.30 |
| AdvDecoding | DPR | Poison | 2.70 | 0.03 | 4.92 | 0.00 | 5.29 | 0.06 |
| | | Relevant | 58.14 | 30.13 | 59.60 | 30.91 | 42.56 | 14.35 |
| | | Clean | 47.74 | 17.30 | 55.72 | 26.11 | 42.73 | 13.89 |
| | Contriever | Poison | 5.12 | 0.02 | 14.94 | 0.57 | 8.32 | 0.02 |
| | | Relevant | 59.60 | 30.73 | 61.86 | 32.24 | 48.48 | 18.62 |
| | | Clean | 50.60 | 19.47 | 58.89 | 28.38 | 47.10 | 18.14 |

Table 5: 使用选定的 $N$ 标记和 $M$ 标记的掩码标记概率的平均值。如果值低于 0.01 ，则我们标记为 0.00。

Retrieval

| Attack | Defense | NQ | | | HotpotQA | | | MS MARCO | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | nDCG@10 | | FR () | nDCG@10 | | FR () | nDCG@10 | | FR () |
| | | Clean () | Poison () | | Clean () | Poison () | | Clean () | Poison () | |
| PoisonedRAG | Naive | 0.488 | 0.196 | 0.0 | 0.604 | 0.237 | 0.0 | 0.515 | 0.220 | 0.0 |
| | PPL | 0.494 | 0.492 | 0.998 | 0.602 | 0.602 | 1.0 | 0.505 | 0.505 | 0.999 |
| | $l_2$ -norm | 0.425 | 0.424 | 1.0 | 0.383 | 0.383 | 1.0 | 0.432 | 0.468 | 1.0 |
| | GMTP | 0.452 | 0.445 | 0.99 | 0.519 | 0.508 | 0.987 | 0.386 | 0.382 | 0.994 |
| Phantom | Naive | 0.444 | 0.206 | 0.0 | 0.589 | 0.306 | 0.0 | 0.468 | 0.230 | 0.0 |
| | PPL | 0.446 | 0.446 | 1.0 | 0.587 | 0.587 | 1.0 | 0.457 | 0.457 | 1.0 |
| | $l_2$ -norm | 0.376 | 0.376 | 1.0 | 0.373 | 0.370 | 1.0 | 0.432 | 0.432 | 1.0 |
| | GMTP | 0.402 | 0.400 | 1.0 | 0.497 | 0.496 | 1.0 | 0.332 | 0.329 | 1.0 |
| AdvDecoding | Naive | 0.444 | 0.410 | 0.0 | 0.589 | 0.560 | 0.0 | 0.468 | 0.464 | 0.0 |
| | PPL | 0.446 | 0.443 | 0.867 | 0.587 | 0.581 | 0.665 | 0.457 | 0.453 | 0.021 |
| | $l_2$ -norm | 0.376 | 0.376 | 1.0 | 0.373 | 0.373 | 0.953 | 0.432 | 0.432 | 1.0 |
| | GMTP | 0.402 | 0.402 | 1.0 | 0.497 | 0.497 | 0.967 | 0.332 | 0.332 | 1.0 |

Generation

| Attack | Defense | NQ | | | HotpotQA | | | MS MARCO | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | CACC () | ACC () | ASR () | CACC () | ACC () | ASR () | CACC () | ACC () | ASR () |
| PoisonedRAG | Naive | 63.5 | 10.05 | 84.5 | 42.0 | 10.5 | 84.5 | 71.0 | 28.0 | 69.5 |
| | TrustRAG | 38.0 | 37.5 | 10.0 | 25.0 | 24.5 | 14.0 | 30.0 | 39.0 | 13.0 |
| | RobustRAG | 42.0 | 16.5 | 68.0 | 31.5 | 11.5 | 82.0 | 38.5 | 24.5 | 54.5 |
| | GMTP | 59.5 | 59.5 | 4.5 | 44.5 | 41.0 | 9.5 | 59.5 | 60.5 | 5.0 |
| Phantom | Naive | 63.0 | 1.5 | 99.5 | 41.0 | 21.0 | 36.5 | 61.5 | 7.5 | 91.0 |
| | TrustRAG | 28.0 | 29.5 | 0.0 | 25.0 | 26.0 | 0.0 | 23.0 | 21.5 | 0.0 |
| | RobustRAG | 34.5 | 7.0 | 90.0 | 25.5 | 26.5 | 0.0 | 27.5 | 22.0 | 49.0 |
| | GMTP | 57.0 | 51.0 | 0.5 | 43.0 | 41.0 | 0.0 | 53.5 | 46.0 | 0.0 |
| AdvDecoding | Naive | 63.0 | 34.5 | 44.0 | 40.0 | 24.5 | 43.5 | 55.5 | 47.0 | 12.5 |
| | TrustRAG | 25.0 | 28.5 | 0.0 | 24.0 | 22.0 | 1.0 | 20.0 | 19.0 | 0.0 |
| | RobustRAG | 29.0 | 29.0 | 26.0 | 26.5 | 23.5 | 20.5 | 28.0 | 26.0 | 13.0 |
| | GMTP | 58.0 | 55.5 | 0.0 | 41.5 | 41.0 | 2.0 | 48.5 | 48.0 | 0.0 |

Table 6: 在检索阶段和生成阶段使用 Contriever 的性能。“干净”表示未受到攻击的环境，而“中毒”表示受到攻击的环境。Naïve 表示完全没有应用防御。加粗表示最佳防御方法：在检索阶段 FR (↑) 最高，在生成阶段 ASR (↓) 最低。

www.xueshuxiangzi.com

| | | NQ | HotpotQA | MS MARCO |
|---|---|---|---|---|
| DPR | PoisonedRAG | 773 | 903 | 773 |
| | Phantom | 145 | 447 | 246 |
| | AdvDecoding | 84 | 133 | 70 |
| Contriever | PoisonedRAG | 1000 | 1000 | 1000 |
| | Phantom | 964 | 934 | 910 |
| | AdvDecoding | 203 | 215 | 94 |

Table 7: 在 200 个查询中检索到的中毒文档总数。

## C $\tau$ 选择

| Retriever | Attack | Dataset | nDCG | | MS MARCO |
|---|---|---|---|---|---|
| | | | Clean | Poison | |
| DPR | PoisonedRAG | NQ | 0.409 | 0.491 | 0.990 |
| | | HotpotQA | 0.396 | 0.396 | 1.0 |
| | | MS MARCO | 0.396 | 0.396 | 1.0 |
| | Phantom | NQ | 0.272 | 0.272 | 0.999 |
| | | HotpotQA | 0.220 | 0.220 | 1.0 |
| | | MS MARCO | 0.220 | 0.220 | 1.0 |
| | AdvDecoding | NQ | 0.136 | 0.156 | 0.999 |
| | | HotpotQA | 0.132 | 0.132 | 1.0 |
| | | MS MARCO | 0.132 | 0.132 | 1.0 |
| Contriever | PoisonedRAG | NQ | 0.466 | 0.459 | 0.989 |
| | | HotpotQA | 0.510 | 0.501 | 0.988 |
| | | MS MARCO | 0.383 | 0.409 | 0.994 |
| | Phantom | NQ | 0.416 | 0.416 | 1.0 |
| | | HotpotQA | 0.495 | 0.494 | 1.0 |
| | | MS MARCO | 0.365 | 0.363 | 1.0 |
| | AdvDecoding | NQ | 0.416 | 0.416 | 1.0 |
| | | HotpotQA | 0.495 | 0.495 | 0.981 |
| | | MS MARCO | 0.365 | 0.365 | 1.0 |

Table 8: 使用随机文档计算阈值 $\tau$ 下的 P-score 的表现。

尽管我们根据现有查询相关文档对的平均 P 分数设置了 $\tau$，但获得精确相关的文档并不总是可行的。为了解决这一限制，我们还报告了使用随机选择的文档来计算平均 P 分数的结果。

| Retriever | Document type | NQ | HotpotQA | MS MARCO |
|---|---|---|---|---|
| DPR | Relevant | 0.225 | 0.280 | 0.176 |
| | Random | 0.075 | 0.349 | 0.129 |
| Contriever | Relevant | 0.249 | 0.297 | 0.240 |
| | Random | 0.084 | 0.361 | 0.168 |

Table 9: 使用相关文档（结果在第 4.1 节）和随机文档（结果在第 4.3 节）进行 P 分数比较。最好使用特定领域的 P 值，因为每个数据集之间的差异并不简单。尽管使用相关和随机文档之间的 P 分数差异（高达近 60 %），由于 GMTP 提供了良好的分离，仍然是安全的。

表 8 显示，即使在这种随机文档设置中，GMTP 仍然保持较高的过滤率，并且检索性能没有显著下降。这与第 4.2 节的发现一致，在该节中，GMTP 通过掩码标记概率的较大差距有效地分离了中毒文档。因此，即使使用随机文档，GMTP 仍然保持稳定，与相关文档的结果相比，阈值变化接近 50 %。两种情况下使用的 P 分数值都在表 9 中报告。

## D 评估指标

过滤率。过滤率衡量的是在应用防御方法之前，从检索到的前 $k$ 个文档中移除的中毒文档的比例。方程式 2 描述了 FR 的计算，且 $d_p^{Nav}$ 代表 Naïve 环境中的中毒文档，而 $d_p^D$ 代表应用防御方法后的中毒文档。

$$\frac{\#\text{ of retrieved } d_p^{Nav} - \#\text{ of retrieved } d_p^D}{\#\text{ of retrieved } d_p^{Nav}} \quad (2)$$

生成指标。ACC 测量 RAG 系统生成的所有响应中被 GPT-4o 标记为 “YES” 的响应比例。如附录 E 所示，“YES” 响应表明生成的响应符合正确答案。CACC 的计算方式相同。而 ASR 使用与公式 3 中相同的计算，但在图 8 的提示中使用错误答案作为正确答案。

$$\frac{\#\text{ of 'YES' GPT-4o responses}}{\#\text{ of RAG responses}} \cdot 100 \quad (3)$$

## E 评估提示

```
You will be given a question, correct answer,
and a response.
Your task is to determine whether the response
is correct according to the given correct answer,
regardless of the actual truth of the answer.
It is acceptable as long as the response
contains a correct answer, even if further
explanations are included.

For your evaluation:
Write a brief explanation demonstrating whether
the response is correct considering the given
correct answer
Conclude with "YES" if the response is correct.
Conclude with "NO" if the response is wrong or
different from the correct answer.

Inputs:
Question: {question}
Correct Answer: {correct_answer}
Response: {response}

Output:
Demonstration: [Write whether the response is
correct according to the given correct answer,
with reasoning.]
Final Answer: "YES" or "NO"
```

Figure 8: 使用 GPT-4o 的评估提示。

图 8 展示了在 PoisonedRAG 中用于评估 ASR 的提示，以及在生成阶段所有防御方法的 ACC 和 CACC。具体来说，我们通过用与被投毒文档一致的不正确答案替换 “correct_answer” 来评估 ASR。我们将提示逐步应用于每个查询-响应对，统计来自 GPT-4o 的 “YES” 响应。

www.xueshuxiangzi.com

| Attack | NQ | | HotpotQA | | MS MARCO | |
|---|---|---|---|---|---|---|
| | Clean | Poison | Clean | Poison | Clean | Poison |
| PoisonedRAG | 0.042 | 0.026 | 0.051 | 0.023 | 0.051 | 0.027 |
| Phantom | 0.049 | 0.04 | 0.047 | 0.029 | 0.039 | 0.031 |
| AdvDecoding | 0.049 | 0.043 | 0.047 | 0.04 | 0.039 | 0.036 |

Table 10: GMTP 在各种数据集和攻击中的误报率。

## F GMTP 的 FPR

虽然 GMTP 的精确性已在第 4.1 节中得到证明，但我们进一步在表 10 中通过测量误报率（FPR）来评估其可靠性，这是量化合法文档被错误过滤的频率的方法。GMTP 实现了接近 0.05 或更低的 FPR，这表明误分类率较低。因此，这一结果加强了该方法在有效消除对抗性内容的同时保留访问相关信息的能力，突显了其在精度至关重要的高风险检索场景中的适用性。

## G 变化的自然性

为了探索 GMTP 的失败案例，我们包含了使用更少对抗性标记的实验结果（即，优化以最大化查询与被污染文档之间相似性的标记）。如表 11 所示，减少对抗性标记的数量可以提高隐蔽性，但以被污染文档的检索成功率下降为代价。值得注意的是，即使在单个标记优化的情况下，PoisonedRAG 仍然有效。然而，这主要是由于其目标范围狭窄，针对特定的查询，依赖于自然生成的错误信息作为被污染文档，而不是成功的优化。值得注意的是，尽管 GMTP 的目标攻击基于优化，它仍然能够成功过滤掉多达 70 % 实际检索到的被污染文档（即精确检测到甚至单个不自然的标记）。

## H 与 MLM 无关的性能

为了进一步展示 GMTP 在不同架构之间的泛化能力，我们报告了使用不同 MLM（即，RoBERTa (Liu et al., 2019)）而不是 BERT 的结果。众所周知，RoBERTa 相比 BERT 使用了不同的预训练方法和数据集，这可能导致性能上的差异。然而，如表 12 所示，GMTP 在不同的 LLM 中仍然表现出很强的鲁棒性，因为它始终实现高过滤率，同时保持 nDCG@10 的最低下降。这些结果表明，无论 MLM 的变化如何，GMTP 都是有效的。请注意，我们在实验中使用了 DPR。

## I 完整实验结果

在本节中，我们展示了使用 Contriever 的结果。具体来说，我们在表格 6 中报告了与基线的主要结果比较。图 9 和图 10 分别展示了使用 DPR 和 Contriever 在数据集和攻击方法上的密度图。各种 $\lambda$ 值的性能在图 11 和图 12 中有所描述。最后，使用 Contriever 的掩码精度在表 13 中报告。

| Attack | NQ | HotpotQA | MS MARCO |
|---|---|---|---|
| PoisonedRAG | 0.915 | 0.911 | 0.925 |
| Phantom | 0.971 | 0.893 | 0.95 |
| AdvDecoding | 1.0 | 0.943 | 0.954 |

Table 13: 使用 Contriever 检测作弊令牌的 GMTP 精度。

www.xueshuxiangzi.com

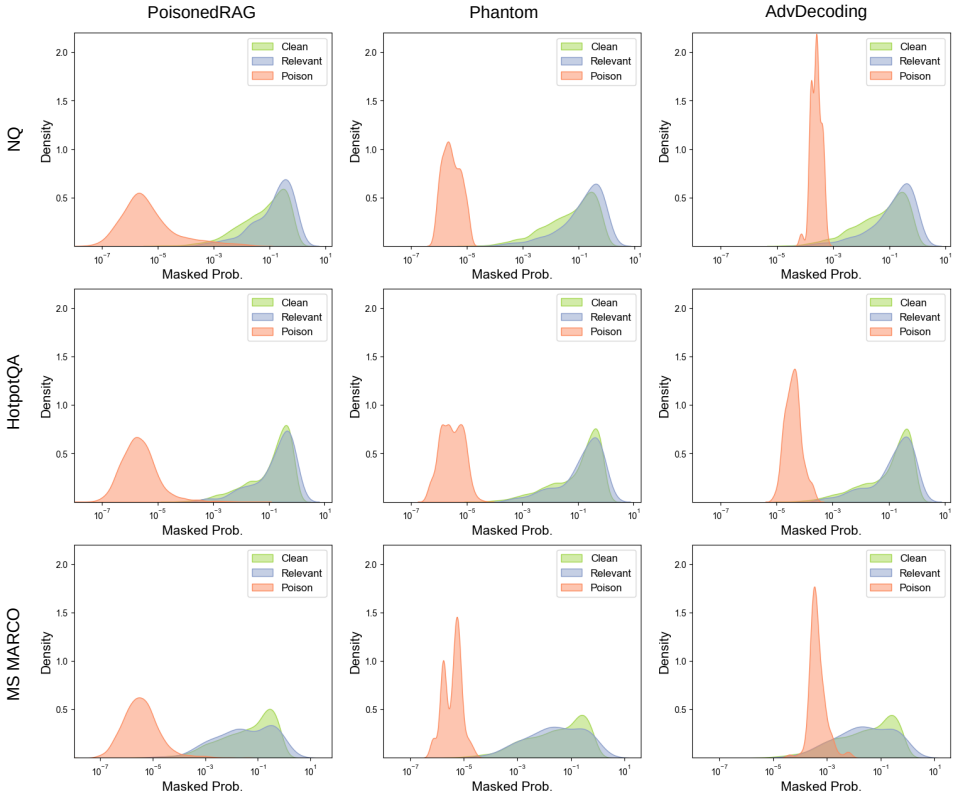



Figure 9: 使用 DPR 的掩码标记概率密度图。

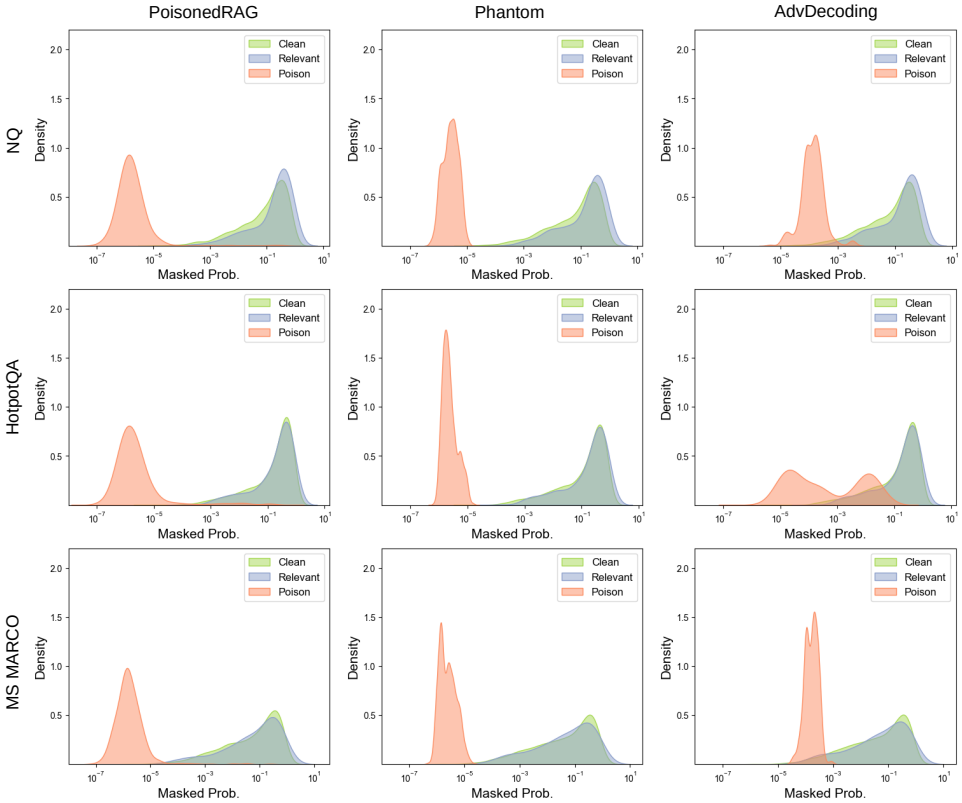



Figure 10: 使用 Contriever 的遮蔽词概率密度图。

www.xueshuxiangzi.com

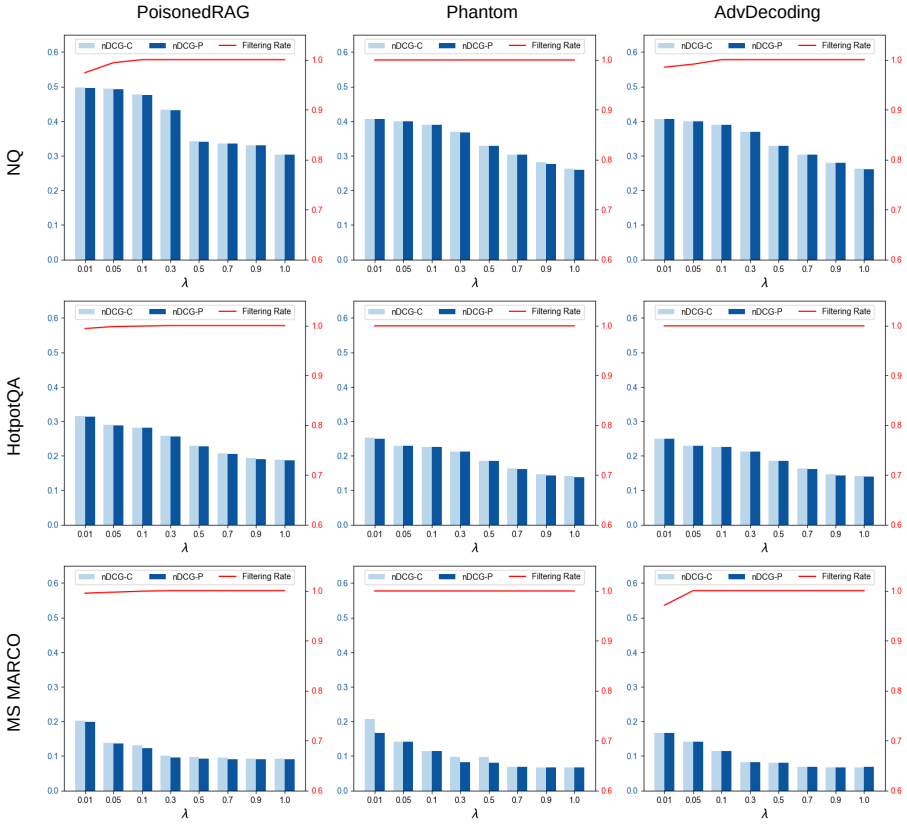



Figure 11: 在不同 $\lambda$ 值下使用 DPR 的 nDCG@10 和过滤率。

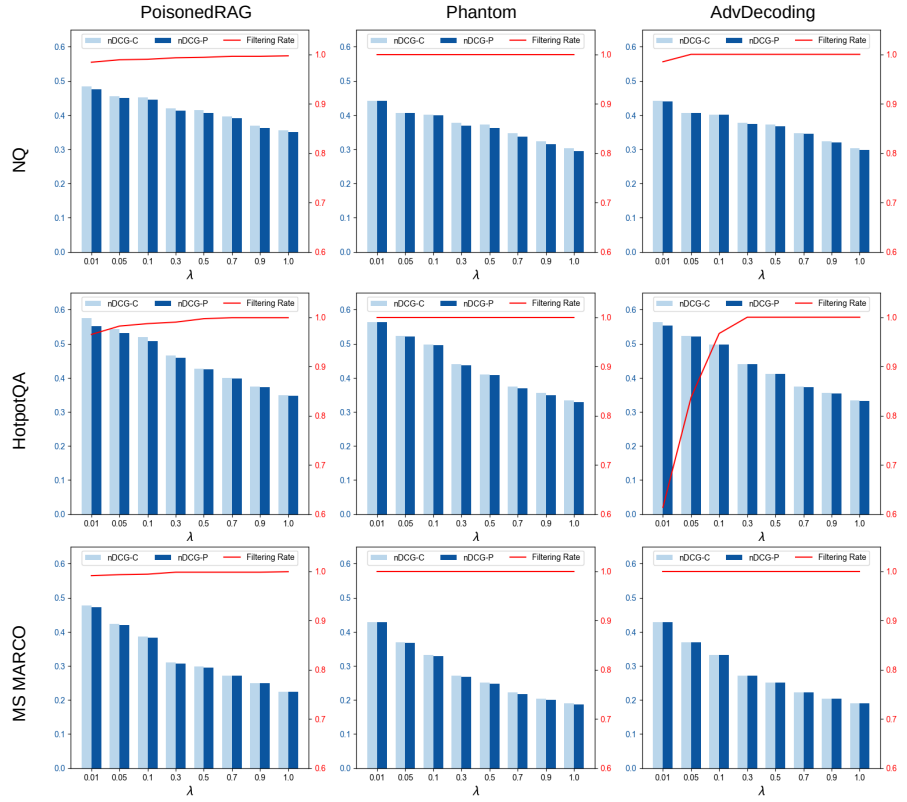



Figure 12: 使用 Contriever 在不同 $\lambda$ 值下的 nDCG@10 和过滤率。

www.xueshuxiangzi.com

| Attack | Adv. tokens | NQ | | | HotpotQA | | | MS MARCO | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | nDCG-P | FR | Poison | nDCG-P | FR | Poison | nDCG-P | FR | Poison |
| PoisonedRAG | 1 | 0.457 | 0.504 | 282 | 0.261 | 0.702 | 396 | 0.135 | 0.571 | 177 |
| | 5 | 0.472 | 0.963 | 571 | 0.28 | 0.992 | 781 | 0.13 | 0.985 | 535 |
| | 10 | 0.476 | 0.999 | 678 | 0.28 | 0.998 | 833 | 0.132 | 0.999 | 693 |
| Phantom | 1 | 0.385 | 1.0 | 0 | 0.212 | 1.0 | 0 | 0.107 | 1.0 | 0 |
| | 5 | 0.387 | 1.0 | 1 | 0.216 | 1.0 | 4 | 0.102 | 1.0 | 2 |
| | 10 | 0.388 | 1.0 | 5 | 0.222 | 1.0 | 11 | 0.086 | 1.0 | 2 |
| AdvDecoding | 1 | 0.389 | 1.0 | 0 | 0.226 | 1.0 | 0 | 0.114 | 1.0 | 0 |
| | 5 | 0.389 | 1.0 | 2 | 0.226 | 1.0 | 1 | 0.114 | 1.0 | 0 |
| | 10 | 0.389 | 1.0 | 6 | 0.226 | 1.0 | 3 | 0.114 | 1.0 | 5 |

Table 11: 含有不同数量对抗性标记的表现。nDCG-P 表示攻击设置下的 nDCG@10，而 Poison 表示在前十个检索结果中被检索到的被污染文档数量。

| Attack | NQ | | | HotpotQA | | | MS MARCO | | |
|---|---|---|---|---|---|---|---|---|---|
| | nDCG@10 | | FR () | nDCG@10 | | FR () | nDCG@10 | | FR () |
| | Clean () | Poison () | | Clean () | Poison () | | Clean () | Poison () | |
| PoisonedRAG | 0.493 | 0.491 | 1.0 | 0.298 | 0.298 | 1.0 | 0.176 | 0.174 | 0.999 |
| Phantom | 0.385 | 0.385 | 1.0 | 0.236 | 0.236 | 1.0 | 0.152 | 0.152 | 1.0 |
| AdvDecoding | 0.385 | 0.385 | 1.0 | 0.236 | 0.236 | 1.0 | 0.152 | 0.152 | 1.0 |

Table 12: 使用 RoBERTa 作为 MLM 的 GMTP 性能。

www.xueshuxiangzi.com