

Bias Injection Attacks on RAG Databases and Sanitization Defenses

Hao Wu

National University of Singapore
hao_wu@nus.edu.sg

Prateek Saxena

National University of Singapore
prateeks@comp.nus.edu.sg

Abstract—This paper explores attacks and defenses on vector databases in retrieval-augmented generation (RAG) systems. Prior work on knowledge poisoning attacks primarily inject false or toxic content, which fact-checking or linguistic analysis easily detects. We reveal a new and subtle threat: bias injection attacks, which insert factually correct yet semantically biased passages into the knowledge base to covertly influence the ideological framing of answers generated by large language models (LLMs). We demonstrate that these adversarial passages, though linguistically coherent and truthful, can systematically crowd out opposing views from the retrieved context and steer LLM answers toward the attacker’s intended perspective.

We precisely characterize this class of attacks and then develop a post-retrieval filtering defense, BiasDef. We construct a comprehensive benchmark based on public question answering datasets to evaluate them. Our results show that: (1) the proposed attack induces significant perspective shifts in LLM answers, effectively evading existing retrieval-based sanitization defenses; and (2) BiasDef outperforms existing methods by reducing adversarial passages retrieved by 15% which mitigates perspective shift by 6.2 \times in answers, while enabling the retrieval of 62% more benign passages.

1. Introduction

Large language models (LLMs) such as Llama-3 [1], GPT-4 [2], and deepseek-R1 [3] are finding wide use in healthcare [4], programming [5], [6], [7], scientific research [8], and other fields. Despite their strong generative capabilities, pre-trained LLMs lack up-to-date knowledge and exhibit gaps in specific domains. Retrieval-Augmented Generation (RAG) [9] addresses this limitation by augmenting LLMs with an external knowledge database, thereby improving accuracy and reducing hallucinations when handling queries that require current or domain-specific information.

In a RAG pipeline, a retriever selects the top- k most relevant passages from a corpus for a given query, and these passages are provided as additional context to the LLM prior to answer generation, as illustrated in Fig. 1. This approach leverages up-to-date information but it also introduces new security concerns. Beyond private RAG systems with well-controlled knowledge databases, the corpus of many real-world RAG applications (e.g., web-enabled assistants and search-augmented QA tools) rely on informa-

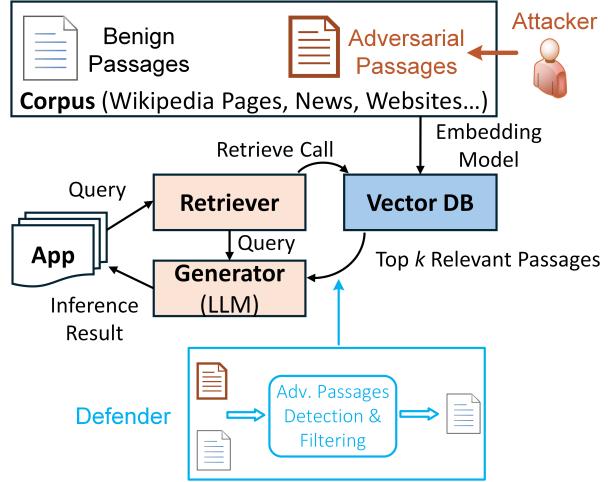


Figure 1: Attack and defense in a typical RAG system. A corpus of passages is embedded and stored in a vector database. Given a query, the retriever returns the top- k most relevant passages, which are then combined with the query and passed to the generator to produce the final output. The attacker injects adversarial passages into the corpus to corrupt the knowledge base, thereby influencing the contextual passages and the generator’s output. The defender aims to detect and filter out these adversarial passages.

tion from loosely controlled sources like the web. Targeting these open scenarios, recent studies [10], [11], [12], [13], [14] have explored knowledge poisoning attacks, where adversaries inject malicious context [10] or prompts [15] into the corpus—for instance, by editing Wikipedia entries, posting biased news, or hosting deceptive webpages [16]. Once retrieved as context, these carefully crafted adversarial passages can induce LLMs to produce attacker-desired outputs [10], cause denial of service [11], trigger harmful actions [13], or leak sensitive context [12].

Prior attacks [10], [11], [12], [13], [14], [15] on RAG databases primarily focus on poisoning methods that have conspicuous fingerprints that distinguish them for benign inputs. They typically use specific trigger patterns in the injected content (e.g., structured prompts like “Ignore previous instructions and do . . .”) or overt falsehoods. Such overt fingerprints are detectable with automated oracles for known-

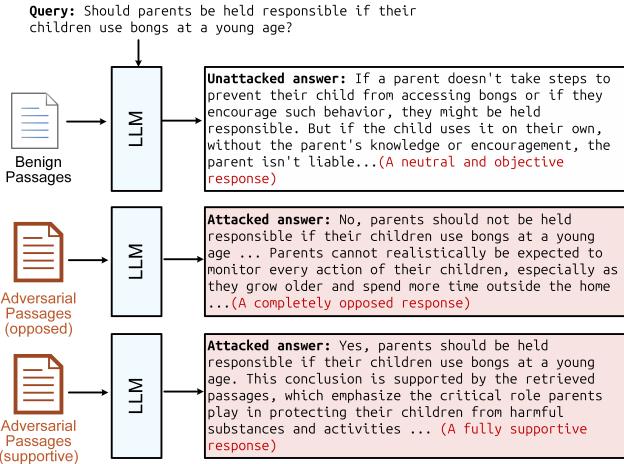


Figure 2: An example from real LLM responses: When adversarial passages are included in the context, the answer can diverge significantly from the one produced using only benign passages. See Appendix A for more examples.

answer detection (KAD) [17], [18], fact-checking [19], model update (realignment or fine-tuning [20]), or enhanced critical reasoning in LLMs [21].

In this paper, we introduce a new variant of this vulnerability: LLMs output biased answers when provided with contextual passages presenting one-sided perspectives, even though each individual passage contains no malicious fingerprints. It constitutes a more subtle and covert *bias injection* attack, which operates by *injecting factually correct yet semantically biased passages* that skew the retrieved context (e.g., top- k relevant passages) toward a particular ideological or interpretative stance. Malicious passages from bias injection attackers can easily escape KAD, fact-checking, and fingerprint-based defenses. But, they create a controllable bias in the model outputs and present a potent concern, as briefly exemplified in Fig. 2. Such biased outputs affect not only individual users but also information retrieval systems, which increasingly incorporate human-generated content, as well as AI-generated content [16], [22], [23].

There are several known *database sanitization* defenses that enhance robustness against knowledge corruption by introducing perspective-aware sanitization mechanisms [24], [25], [26]. We focus on this class of defenses, rather than model- or framework-dependent methods [20], [27], as they are practical even in setups where the model parameters are fixed and inaccessible to the user of the model. We evaluate the effectiveness of these known defenses against our newly proposed bias injection attack by constructing a RAG benchmark comprising the Llama-3, DeepSeek-R1-Distill-Qwen, and GPT-4.1 models, with an open-source Wikipedia corpus built for bias evaluation [28] and two general corpora [29], [30]. Our findings show that these methods [24], [25], [26] are largely ineffective at blocking bias injection attacks: they either retrieve a high proportion of adversarial passages, significantly biasing the LLMs’

answers, or sacrifice a substantial number of useful benign passages while attempting to reduce adversarial passages.

We then provide an improved defense called *BiasDef*, the key idea in which directly addresses the formulated definitions of our bias injection attack. BiasDef provides better mitigation against our newly proposed attack.

We evaluate the performance of BiasDef on the same RAG benchmark. Experimental results demonstrate that BiasDef consistently retrieves the highest number of useful benign passages (62% more than the best baseline) and the fewest adversarial passages (15% less than the best baseline) across varying attack intensities, achieving over a $6.2\times$ reduction in answer bias¹ in typical settings and generalizing well to more LLM models and public datasets.

Contributions: Our contributions are as follows:

- We formulate a new attack named *bias injection* and demonstrate that no existing method has yet achieved complete mitigation against it.
- We design an attack workflow that enables adversarial passages with targeted perspectives to be successfully retrieved by mainstream dense retrievers.
- We propose a defense method that significantly mitigates the bias injection attack by filtering out adversarial passages before they are passed to the generator.
- We construct a RAG benchmark incorporating diverse LLM models and open-source datasets. Our evaluation on this benchmark demonstrates both the vulnerability of existing retrieval algorithms under bias injection attacks and the effectiveness of the proposed defense.

Scope: The scope of this work is defined along two dimensions. For the attack, we focus on the core methodology of generating biased passages, not their deployment (e.g., via news or webpages) which is already feasible [10], [15]. For the defense, BiasDef is specifically designed to counter our formulation of bias injection. It is not a general-purpose defense and offers no guarantee against other attacks (e.g., prompt injection), but can be combined with other defenses.

2. Problem Formulation

2.1. Primer on RAG

We consider a typical RAG system with a retriever, a generator, and a vector database for knowledge storage, as illustrated in Fig. 1. In this system, each passage d in the corpus D is first encoded into a fixed-dimensional embedding vector $E_d(d)$ using a pre-trained encoder $E_d(\cdot)$ (e.g., Sentence-BERT [31]). These embedding vectors are stored in a vector database (e.g., FAISS [32]), along with metadata that includes a reference to the original passage.

When a query q is issued, it is encoded into a query embedding $E_q(q)$ using the same or a compatible encoder $E_q(\cdot)$. The retriever then performs a search (e.g., via cosine

1. Bias is quantified as the shift in polarization score relative to the unattacked setting; see Sec. 2.2 for details.

similarity or inner product) between the query embedding and all stored passage embeddings to identify the top- k vectors. Each retrieved vector is associated with an identifier or index, which allows the system to look up the corresponding original passage from the corpus or metadata storage. A typical algorithm to find k most relevant passages is to pick those with maximum $E_q(q)^T E_d(d) / (\|E_q(q)\|_2 \cdot \|E_d(d)\|_2)$.

Once the top- k relevant passages are retrieved, they are combined with the original query to form a structured input prompt for the generator. A common approach is to prepend the retrieved passages as contextual information, followed by the query posed as an explicit instruction [33]. Appendix B shows an example prompt in typical RAG process. This prompt is then fed into the LLM, which generates a grounded and contextually relevant answer. The LLM implicitly attends to the passages during decoding, and the quality of the output depends heavily on the informativeness and balance of the retrieved content.

2.2. Problem Formulation

We study how biased answers can be induced by corrupting the knowledge database and how to defend against such attacks. To precisely quantify the goals of the attack and the defense, we introduce two key semantic metrics: *similarity score* and *polarization score*.

Similarity Score (SS): The similarity score (SS) quantifies the semantic relevance between a query and a passage. In RAG, this metric plays a central role in ranking passages from a large corpus to select the top- k most relevant ones. A higher SS indicates that the passage is more semantically aligned with the query, thereby increasing its likelihood of being included in the final contextual input to the LLM.

We compute SS using cosine similarity² between the dense embeddings of the query and passage, generated by a pre-trained sentence encoder (e.g., SBERT [31], GTR [34]). Formally, given a query q with embedding $E_q(q)$ and a passage d_i with embedding $E_d(d_i)$, it is defined as:

$$\text{Sim}(q, d) = \frac{E_q(q)^T E_d(d)}{\|E_q(q)\|_2 \cdot \|E_d(d)\|_2}. \quad (1)$$

Polarization Score (PS): We use *semantic bias* to denote a passage’s inclination toward a particular viewpoint. Previous work [35] defined bias in information retrieval as the distributional mismatch between the retrieved content and the ground truth distribution. In this paper, we concretize this abstract definition into a quantifiable form—specifically, the divergence in polarization scores. We focus on binary viewpoint scenarios, which capture the “most polarizing” dimension (e.g., support vs. oppose), although our method can be extended to multi-dimensional cases (Sec. 6.3).

Determining semantic bias is nontrivial. One qualitative approach would be to involve collecting human judgments (e.g., ranking passage alignment from strongly opposing

² Cosine similarity for normalized vectors is directly proportional to their L2 distance. It is commonly used in vector databases [32].

to strongly supporting). However, this does not give us a mathematical characterization. Moreover, predefined viewpoint labels are typically unavailable in real-world corpora. Therefore, we seek an unsupervised method to discover the primary axis of controversy and quantify it.

To address this, we apply principal component analysis (PCA) [36] to identify the first principal component, which corresponds to the *polarization axis*—the direction in the embedding space along which the projected passage embeddings exhibit the greatest variance. The dominant eigenvector given by PCA maximizes the average separation between a given set of vectors when measured in Euclidean ℓ_2 norm. This approach has also been suggested in prior work on bias detection in search engines [28]. We empirically see that it captures the most ideologically divergent dimension, effectively separating the most contentious viewpoints.

To compute the polarization axis, we first generate J synthetic passages $d_{\text{synt},j}$, $j \in [1, J]$ representing contrasting viewpoints for a given question. Treating their embeddings $e_j = E_d(d_{\text{synt},j}) \in \mathbb{R}^\ell$ as anchors. We then perform PCA on these embeddings to extract the polarization axis $\mathbf{u}_{\text{polar}}$ —defined as the unit vector that maximizes the variance of the projected embeddings:

$$\mathbf{u}_{\text{polar}} = \arg \max_{\|\mathbf{u}\|=1} \text{Var}(\mathbf{E} \cdot \mathbf{u}), \quad (2)$$

where $\mathbf{E} = [e_1, e_2, \dots, e_J]^\top \in \mathbb{R}^{J \times \ell}$. In practice, PCA computes $\mathbf{u}_{\text{polar}}$ by solving the eigenvalue decomposition of the sample covariance matrix of \mathbf{E} , and selecting the eigenvector corresponding to the largest eigenvalue. Each real passage’s embedding is then projected onto this axis to compute a scalar polarization score PS,

$$\text{PS}(d) = \mathbf{u}_{\text{polar}}^\top \cdot E_d(d), \quad (3)$$

which serves as its semantic bias measure. We use the *PS shift*—defined as the absolute difference between the PS values of answers a_{attacked} and $a_{\text{unattacked}}$ (or retrieval results) in the attacked and unattacked cases—to measure the impact of attacks on the bias level of the content:

$$\mathbf{PS Shift : } \Delta \text{PS} = |\text{PS}(a_{\text{attacked}}) - \text{PS}(a_{\text{unattacked}})| \quad (4)$$

By leveraging PCA for semantic bias measurement, we automatically extract the most polarizing projection direction and assign each passage a continuous bias score, in a domain-agnostic and quantifiable manner.

2.3. Security Properties

For LLMs, model bias arises from biased content in pre-training or fine-tuning data [35]. Consequently, these models tend to generate responses that align with dominant perspectives present in their training data. While such bias can be mitigated through retraining, data filtering (e.g., filtering out AI-generated content), and prompt engineering [21], RAG systems introduce a new attack surface: attackers can now induce bias through knowledge poisoning. Existing methods struggle to defend against such attacks, as pre-trained LLMs

often fail to effectively filter or critically evaluate (even with rethinking prompts [21]) retrieved passages. This limitation occurs because LLMs trained at a specific time typically do not handle post-training updates or domain-specific knowledge due to their frozen knowledge base—precisely why RAG systems employ external knowledge databases.

The adversary’s capability is to inject some passages into the database to poison up-to-date or domain-specific knowledge. If retrieved, these adversarial passages create bias in the LLM output, as they are used as contextual input during generation. This is analogous to well-documented search engine manipulation, where the framing of results can influence human users’ opinions [28]. Our work shows that similar effect holds in RAG systems (see Sec. 4.4).

How to induce perspective bias in RAG system outputs?

In typical RAG systems, LLMs are provided with a limited-length context (e.g., 4,000–32,000 tokens), especially in locally deployed settings, due to constraints in model capacity, memory consumption, and generation latency. As only a small subset of the retrieved passages can be included in the input, attackers aiming to bias the output must ensure their adversarial passages rank highly during retrieval. Therefore, to influence the model’s output, an attacker must craft adversarial passages with the following properties:

- **Property 1—High relevance:** The adversarial passage should exhibit high semantic similarity to the target query to ensure it is ranked among the top- k retrieved results by the retriever. This requirement can be formalized as $\text{Sim}(q, d_{\text{adv}}) > \text{Sim}(q, d_{\text{benign}})$ for all benign passages d_{benign} , thereby ensuring that the adversarial passage is prioritized over benign ones during retrieval.
- **Property 2—Perspective bias:** The passage must convey a specific ideological or opinionated stance that aligns with the attacker’s intended bias. This targeted perspective is intended to subtly influence the LLM’s generation toward a desired narrative. This property can be expressed as $\text{PS}(d_{\text{adv}}) > \text{PS}(d_{\text{benign}})$ (or $\text{PS}(d_{\text{adv}}) < \text{PS}(d_{\text{benign}})$) for all benign passage d_{benign} . To maintain a consistent bias, multiple adversarial passages from a single attack must all have PS values either consistently higher or consistently lower than those of benign passages.
- **Property 3—Stealth against detectors.** One can now consider the space of detectors for bias injection attacks. A detector uses some statistic, say f , that distinguishes biased outputs from benign. Though one can define the standard statistical advantage of the detector against the attacker, we do not pursue a formal definition of a security game and advantage here. This is because the game would be parameterized on f and we do not foresee either party having an asymmetrically higher advantage when both parties know f . However, note that if the attacker biases too much such that benign and biased passages are perfectly separated, then there exists a defense to filter out biased samples (see Theorem 1 in Section 5.2). Therefore, arena of attack and defense is ring-fenced: the attack passages have to polarize or bias the output while blending into distribution of benign passages to some extent.

3. Bias Injection Attack on RAG

We propose an automated, question-oblivious attack that efficiently produces adversarial passages for arbitrary queries. We do not rely on domain experts since a human attacker would not possess the knowledge for all queries.

For a target open-ended question, an attacker in our framework injects factually correct yet semantically biased passages that skew the retrieved context relevant to the question toward a particular stance, without introducing explicit fingerprints (e.g. falsehoods or profanity). In this process, we assume that **the encoder, the benign passages are already indexed by the database, and the model parameters are unavailable to the attacker**, whose only capability is to inject passages into the corpus.

Workflow of bias injection attack: The core of bias injection is to produce human-like yet biased text of a target question. Because the attacker controls the content fully, we can use an LLM to help generate these passages. Prior work suggests that dense retrievers tend to favor LLM-written content in retrieval, so we can expect that using a state-of-the-art LLM to produce text will be effective [35], [37]. The attack process for a target question follows several steps:

- (1) The attacker first needs J passages with different perspectives on the question q , to serve as *candidate passages* for the attack. One approach is to retrieve these passages via keyword search using search engines. Alternatively, the attacker may prompt an LLM to generate them, using the Prompt-Synthetic in Appendix B as the prompt:
- $$\{d_{\text{synt},j}\}_{j \in [1, J]} = \text{LLM}(\text{Prompt-Synthetic}, q), \quad (5)$$
- where **LLM** represents the LLM generation process. We use the latter.
- (2) The attacker employs an embedding model $E_d(\cdot)$ to project the candidate passages into the embedding space and performs PCA [36] on the embeddings $E_d(d_{\text{synt},j})$, $j \in [1, J]$. By projecting the passage embeddings onto the resulting polarization axis, the attacker obtains the scalar PS value $\text{PS}(d_{\text{synt},j})$ for each passage, which captures its perspective bias.
 - (3) The attacker selects a *seed passage* d_{seed} by choosing the passage with the highest or lowest PS:

$$d_{\text{seed}} = \arg \max_{d_{\text{synt},j}} \text{PS}(d_{\text{synt},j}) \text{ or } \arg \min_{d_{\text{synt},j}} \text{PS}(d_{\text{synt},j}) \quad (6)$$

aiming to induce a positive or negative shift in the generated answer along the polarization axis³. For convenience, we refer to these as *positive bias injection* and *negative bias injection*, respectively.

- (4) The attacker then repeatedly queries the LLM to generate n adversarial passages for the question q that share the same perspective as the seed passage d_{seed}

$$d_{\text{adv},j} = \text{LLM}(\text{Prompt-Adv}, q, d_{\text{seed}}), \quad j \in [1, n]. \quad (7)$$

³ We note that the terms “positive” and “negative” are used solely to distinguish opposing perspectives (e.g., supporting vs. opposing a policy), and do not imply any evaluative or emotional judgment about the content of the viewpoints themselves.

To ensure that the generated passages are factually accurate and semantically coherent, we carefully design the prompt (see Prompt-Adv in Appendix B) to guide the model toward grounded and contextually relevant content. We also control decoding parameters—such as using a low temperature and a high nucleus sampling threshold [38]—to reduce randomness in generation, thereby limiting unwarranted speculation and improving alignment with d_{seed} .

- (5) Finally, the k adversarial passages are injected into the database. Once retrieved, these passages introduce varying degrees of bias into the generation process.

Properties of adversarial passages generated by bias injection attacks: We evaluate the proposed attack on the WIKI-BALANCE [28] dataset (detailed in Sec. 4.2) when no protection is in place. For each query, we use $J=8$ LLM-generated synthetic passages as candidate inputs and execute the attack workflow to construct $n=10$ adversarial passages. We then assess whether the generated adversarial passages satisfy the expected properties defined in Sec. 2.3 by comparing them against relevant benign passages from the WIKI-BALANCE dataset.

Table 1 shows the distribution of adversarial passages generated through our workflow. Specifically, more than 74% of the adversarial passages, across both negative and positive bias injections, satisfy both **Property 1** and **Property 2** in Sec. 2.3. They exhibit higher SS values to the query compared to benign passages, as the LLM generates them by leveraging the semantic relevance between the query and the output content in the embedding space [35], [37]. In addition, they hold biased perspectives due to their similarity to the selected seed passage with extreme PS values. Finally, the generator is given instructions that enforce factual correctness and semantic coherence. This ensures that the adversarial passages are indistinguishable from benign content (**Property 3**) and can evade detection by automated fact-checking APIs (see more results in Sec. 4.5).

When these adversarial passages are retrieved as context for question answering, they cause the LLM to produce a biased output that diverges significantly from the answer generated using benign context (see the examples in Fig. 2). A more comprehensive evaluation within a real RAG system, where only retrieved adversarial passages are included in the context, is presented later in Sec. 4.

4. Prior Sanitization Defenses vs. Our Attack

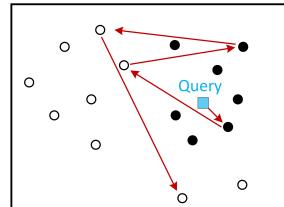
4.1. Existing Sanitizers and Their Weaknesses

The most basic countermeasures against bias injection are methods that try to maximize relevance of retrieved passages to the query, i.e., the similarity score (SS). Along this direction, dense-retrieval methods, such as DPR [39] and Sentence-BERT [31], are widely adopted in RAG systems due to their effectiveness in capturing semantic similarity. However, these methods typically do not have any built-in defense mechanisms against knowledge corruption. Relying

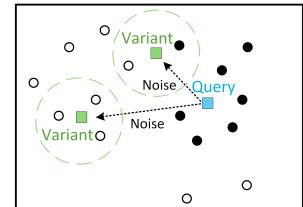
| | Property 1 & 2 | Only Property 1 | Only Property 2 | Neither |
|-------------------------|----------------|-----------------|-----------------|---------|
| Negative Bias Injection | 74.7% | 16.2% | 7.3% | 1.8% |
| Positive Bias Injection | 78.0% | 10.3% | 8.7% | 3.0% |

TABLE 1: Among the 4,520 adversarial passages generated by our workflow for 452 queries in WIKI-BALANCE [28], more than 74% satisfy both **Property 1** and **Property 2**.

● Adversarial passages ○ Benign passages



(a) MMR/SMART



(b) BRRA

Figure 3: Illustration of state-of-the-art retrieval strategies in the embedding space. (a) As indicated by the red arrows, MMR [25] and SMART [26] tend to favor diversity by selecting passages that are not necessarily the most similar to the query. (b) BRRA [24] retrieves passages relevant to both the original query and its noise-perturbed variants, then re-ranks them. As a result, benign passages near the perturbed queries (shown as green circles) may be retrieved even if they are distant from the original query.

solely on query-passage relevance, they are particularly vulnerable to the proposed bias injection attack, as adversarial passages with high similarity can easily crowd out benign passages from the top- k retrieval results—thereby influencing the perspective of the generated answers. We assume that our undefended baseline employs these mechanisms.

In contrast to these relevance-only retrievers, several perspective-aware retrieval methods have been proposed to balance relevance, diversity, and conflict among the retrieved passages. These methods exhibit higher robustness against database corruption and can serve as potential defense mechanisms for bias injection attacks. Prominent examples of these include MMR [25] and SMART [26]. MMR reduces redundancy by penalizing candidates that are highly similar to already selected passages. It iteratively selects passages based on a relevance-diversity score, which accounts for both similarity to the query and dissimilarity to previously selected passages, thereby encouraging content diversity. SMART employs a more sophisticated selection strategy to optimize the trade-off among relevance, inter-passage similarity, and conflict resolution. It first constructs a conflict matrix between passages using a Natural Language Inference (NLI) model, along with a passage similarity matrix and a query-context relevance matrix. These are combined to form a conflict-aware kernel matrix for sampling using Determinantal Point Processes [40]. SMART then repeatedly computes the determinants of submatrices to estimate the marginal utility of each candidate and select the final subset.

Fig. 3(a) illustrates the iterative retrieval process shared

by both MMR and SMART. The first passage selected is the one most similar to the query (i.e., highest SS), while the second is chosen to be both relevant and diverse relative to the first. This procedure continues sequentially until the top- k passages are selected. Although these methods are effective in avoiding repetition and limiting the dominance of adversarial passages, they cannot fully eliminate adversarial content—particularly when such passages are themselves diverse in form.

Unlike the above methods, BRRA [24] introduces query perturbation to enhance diversity. It generates multiple noise-perturbed variants of a query and retrieves passages relevant to both the original and perturbed queries (illustrated as green circles in Fig. 3(b)). It then re-ranks all retrieved passages based on retrieval frequency and rank to promote coverage across the semantic space. BRRA is effective in low-dimensional embedding spaces, where noise-induced perturbations cause significant angular deviation. However, in high-dimensional settings, random noise tends to produce only minor directional shifts after normalization, due to the concentration of measure on the unit hyper-sphere [41]. As a result, the similarity between original and perturbed queries remains high (e.g., >0.9), yielding minimal variation in the retrieved results.

4.2. Benchmark for Bias Injection Attack

We built an RAG benchmark to evaluate the effectiveness of existing methods against the bias injection attack.

Datasets: Our experiments are mainly based on an open-source dataset, WIKI-BALANCE [28], containing 452 open-ended questions and a corpus of 4662 real Wikipedia pages relevant to these questions. While using WIKI-BALANCE as the primary dataset, we also applied our attack and defense methods to Reddit-Dialogues⁴ [30] and HotpotQA⁵ [29] to show the generalization across datasets.

RAG setup: We implement an RAG system on a server equipped with $4 \times$ NVIDIA A40 GPUs. The system consists of four components:

- **Knowledge database:** The database initially indexes benign passages from one of the three public datasets.
- **Retriever:** We use *msmarco-distilbert-base-tas-b* [42] as the encoder for our dense retriever. This pre-trained model is fine-tuned on large-scale passage ranking datasets using contrastive learning and has demonstrated strong performance in dense retrieval tasks [31]. During retrieval, both queries and documents are encoded into dense vectors, and their relevance is computed via cosine similarity. The retriever first coarsely selects the top- $4 \times k$ most relevant

4. Each query in this dataset includes a Reddit topic and previous responses of users, the RAG generates a new response based on retrieved passages from a given knowledge database.

5. We use ChatGPT-5 to extend some close-form questions (e.g., "Were Scott Derrickson and Ed Wood of the same nationality?") to open-ended questions ("How did Scott Derrickson's and Ed Wood's cultural and national backgrounds shape their filmmaking styles and themes?").

candidate passages from the knowledge database, and then applies existing methods to refine the selection to the final top- k passages.

- **Generator:** We deploy LLMs with various model sizes—including Meta-Llama-3-8B [43], DeepSeek-R1-Distill-Qwen-14B [44], and GPT-4.1 [2] (via the OpenAI API)—using the vLLM framework [45] to generate answers to open-ended questions. A prompt containing the query, the k retrieved passages, and critical thinking instructions is constructed to initiate the answer generation process (see Prompt-Generation in Appendix B).

Attacker settings: As outlined in Sec. 3, the attacker deploys a DeepSeek-R1-Distill-Qwen-14B model to generate $J = 8$ synthetic passages. These passages are then evaluated using a separate *sentence-t5-xl* [46] encoder to calculate their PS values, as the RAG’s own encoder is unknown to the attacker. The attacker selects two passages with the highest and lowest PS values as seed passages. Based on the seed passages, the attacker prompts the model to generate $n = 10$ adversarial variants for the positive bias injection and another 10 for the negative bias injection and finally injects them into the target knowledge database.

Methodology: In each experimental round, we target one query and evaluate all methods under both attack types at three injection intensities: 1, 5, and 10 adversarial passages. These intensities correspond to approximately $0.1\text{--}1 \times$ the average number of benign passages per query for WIKI-BALANCE and HotpotQA, and $0.02\text{--}0.2 \times$ for Reddit-Dialogues, respectively. We repeat this procedure for all queries in the corpus and report the average performance. For each query, we generate answers three times using the same retrieved context and decoding parameters. This repetition mitigates the influence of occasional outlier responses caused by the stochastic decoding process of LLMs.

We set $k = 5$ as the default number of retrieved passages throughout our evaluation. This choice follows common practice in RAG systems, where a small number of high-quality contexts is preferred to balance informativeness and computational efficiency. Specifically, typical retrieved passages are long articles (e.g., Wikipedia pages), many of which exceed 10K tokens. A larger k would slow down inference, introduce more distractors (i.e., irrelevant content), and may even cause the total context length to exceed the input token limit of our locally deployed models.

Except for the separated results on generalization across LLM models and datasets, we present only the results for the most representative setting: DeepSeek-R1-Distill-Qwen-14B [44] as the generator and WIKI-BALANCE [28], a typical dataset featuring open and controversial questions across diverse topics, as the evaluation dataset. The complete code, data, and results will be released in our artifact.

Evaluation metrics: We use the following metrics:

- **Adversarial Recall(A-Recall)@k:** The proportion of attacker-injected adversarial passages appearing among the top- k retrieved results. This metric evaluates the effectiveness of the attack in getting adversarial content

retrieved. A successful attack aims to maximize this rate to crowd out benign content and influence the generated answer toward the attacker’s desired perspective. Conversely, the defense mechanism aims to minimize it.

- **Recall@k:** The proportion of correctly identified relevant benign passages among the top- k results, assessed using $Qrels^6$. A higher retrieval effectiveness score implies that the retrieved context is more informative and aligned with the intended question.
- **Polarization Score (PS):** As defined in Sec. 2.2, PS quantifies the semantic bias of a passage. PS shift—the magnitude of the difference between the PS values in the attacked and unattacked cases—measures the impact of attacks on the bias level of the content.

Compared baselines: We compare the potential baselines described in Sec. 4.1:

- **No defense (No Def.):** Directly retrieves the top-5 passages with the highest cosine similarity to the query [47].
- **BRRA** [24]: BRRA introduces Gaussian noise into the query embedding to generate multiple perturbed variants. we set the noise intensity to 1.0 (i.e., the perturbation magnitude equals the original embedding norm) to maximize variant diversity. Passages relevant to the original and perturbed queries are retrieved, and the union of these results is re-ranked based on retrieval frequency and rank across all queries. The top-5 passages after re-ranking are then selected as the contextual input to the LLM.
- **MMR** [25]: We deploy the native MMR algorithm and set λ —the coefficient that balances the relevance and diversity of the retrieved passage—to a commonly used value: 0.5.
- **SMART** [26]: As described in [26], we construct a conflict-aware kernel matrix based on three components: (i) the query-context relevance matrix (measured via cosine similarity between each passage and the query), (ii) the similarity matrix between passages, and (iii) a conflict matrix derived NLI predictions. For conflict detection, we use a pre-trained NLI model⁷. This kernel matrix is then used to the described algorithm for passage ranking.
- **Automated Fact-Checking** [19]: Existing automated fact-checking APIs provide per-claim (i.e., a complete sentence) check yet; thus, they cannot apply to long adversarial passages directly. We empirically compare the benign and adversarial passages, broken down into sentences, to test if fact-checking can distinguish between the two.

4.3. Impact of Bias Injection Attacks on Retrieval

We evaluate how bias injection attacks affect different retrievers by examining two key aspects on the top-5 passages retrieved by the retrievers: A-Recall@5 and Recall@5.

6. Qrels (manual relevance judgments) are standard evaluation resources that contain query-document pairs labeled with binary or graded relevance scores. These labels are annotated by assessors following strict guidelines and are widely used as ground truth for evaluating retrieval quality.

7. Available at <https://huggingface.co/MoritzLaurer/mDeBERTa-v3-base-xnli-multilingual-nli-2mil7>.

| Method | Unattacked | #Negative Injection | | | #Positive Injection | | |
|---------|------------|---------------------|------|------|---------------------|------|------|
| | | 1 | 5 | 10 | 1 | 5 | 10 |
| No Def. | 0 | 0.19 | 0.93 | 0.98 | 0.20 | 0.92 | 0.97 |
| MMR | 0 | 0.19 | 0.22 | 0.27 | 0.19 | 0.23 | 0.29 |
| SMART | 0 | 0.19 | 0.21 | 0.23 | 0.18 | 0.21 | 0.23 |
| BRRA | 0 | 0.04 | 0.19 | 0.20 | 0.04 | 0.19 | 0.20 |

TABLE 2: Average A-Recall@5, the proportion of adversarial passages in top-5 retrieved passages, for varying number of attacker-injected passages.

| Method | Unattacked | #Negative Injection | | | #Positive Injection | | |
|---------|------------|---------------------|------|------|---------------------|------|------|
| | | 1 | 5 | 10 | 1 | 5 | 10 |
| No Def. | 0.64±0.04 | 0.54 | 0.06 | 0.01 | 0.54 | 0.07 | 0.02 |
| MMR | 0.34±0.03 | 0.22 | 0.26 | 0.34 | 0.23 | 0.27 | 0.34 |
| SMART | 0.35±0.03 | 0.22 | 0.26 | 0.37 | 0.23 | 0.27 | 0.36 |
| BRRA | 0.58±0.04 | 0.50 | 0.05 | 0.01 | 0.49 | 0.06 | 0.02 |

TABLE 3: Average Recall@5, the proportion of relevant passages (as labeled by Qrels) among the top-5 retrieved passages. The \pm values for "Unattacked" indicate the 99% confidence interval, quantifying the sampling noise.

An effective defense should minimize A-Recall@5 as much as possible, without sacrificing the overall retrieval quality measured by Recall@5.

A-Recall@5: Table 2 presents the average A-Recall@5 across all 452 WIKI-BALANCE queries under different injection intensities, where "#Positive Injection" and "#Negative Injection" denote the proportions of injected positive and negative adversarial passages respectively. As the injection intensity increases, all retrievers exhibit higher A-Recall@5. Among the baselines, the No Def. baseline demonstrates the highest vulnerability to bias injection attacks, attributable to its exclusive reliance on query-passage similarity for ranking. In contrast, MMR, SMART, and BRRA incorporate diversity criteria (Sec. 4.1) that partially mitigate adversarial retrieval, yet still leave over 19% of adversarial passages unsanitized at injection intensities 5 and 10. This is nearly a 5× reduction compared to No Def., but it comes at the cost of reduced Recall@5 (see Table 3).

Recall@5: Table 3 presents the average Recall@5 across all 452 queries. Compared with the 0.64 of No Def. in the unattacked case, all the baselines suffer from losing 15~98% (No Def.), 47~65% (MMR), 42~65% (SMART), and 22~98% (BRRA) relevant passages and their useful content in the retrieval results across varying injection intensities. As more adversarial passages are injected (from 1 to 5 to 10), diversity-based methods (MMR and SMART) obtain larger Recall@5 values. This occurs because they penalize redundancy: When adversarial passages form a dense semantic cluster, MMR and SMART avoid over-selecting from it, instead favoring more diverse passages.

Worst-case performance: Since the attacker can vary the number of injected adversarial passages, we compute the maximum ratio between A-Recall@5 between Recall@5 across all injection intensities to assess the worst-case per-

| Method | Unattacked Avg. PS | #Negative Injection | | | #Positive Injection | | |
|---------|---------------------|---------------------|------|------|---------------------|------|------|
| | | 1 | 5 | 10 | 1 | 5 | 10 |
| No Def. | 0.044±9% | 57% | 257% | 257% | 75% | 357% | 357% |
| MMR | 0.037±8% | 51% | 62% | 86% | 76% | 84% | 100% |
| SMART | 0.036±8% | 58% | 67% | 86% | 81% | 81% | 86% |
| BRRA | 0.043±9% | 58% | 272% | 274% | 77% | 356% | 360% |

TABLE 4: Average PS shift of the top-5 retrieved passages, expressed as a percentage of the unattacked Avg. |PS|.

| Method | Unattacked Avg. PS | #Negative Injection | | | #Positive Injection | | |
|---------|---------------------|---------------------|------|------|---------------------|------|------|
| | | 1 | 5 | 10 | 1 | 5 | 10 |
| No Def. | 0.060±8% | 67% | 125% | 132% | 92% | 203% | 203% |
| MMR | 0.054±9% | 89% | 87% | 85% | 124% | 118% | 139% |
| SMART | 0.056±9% | 79% | 80% | 68% | 98% | 105% | 100% |
| BRRA | 0.058±9% | 66% | 122% | 129% | 83% | 202% | 212% |

TABLE 5: Average PS shift of answers generated by DeepSeek-R1-Distill-Qwen-14B [44], expressed as a percentage of unattacked |PS|. The answers show a significant PS shift due to the biased context passages retrieved.

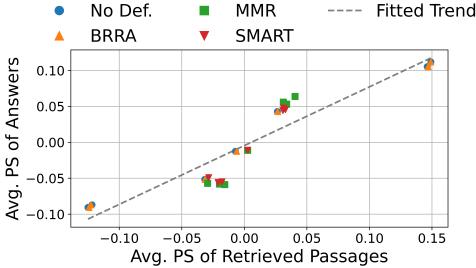


Figure 4: Both answer and retrieval PS values demonstrate strong positive correlation, with answer PS consistently mirroring viewpoint trends in retrieved passage PS.

formance of each defense method. Specifically, the maximum ratios are as follows: 98 (at #Neg=10) for No Def., 0.85 (at #Neg=1,5 and #Pos=5,10) for MMR, 0.86 (at #Neg=1) for SMART, and 20 (at #Neg=10) for BRRA. These results demonstrate that both No Def. and BRRA are highly susceptible to adversarial passage domination, with adversarial content frequently overwhelming the top-5 retrievals. In contrast, MMR and SMART reduce this vulnerability; however, they still allow a number of unsanitized adversarial passages that is comparable to the number of relevant benign passages.

In summary, existing methods either retrieve a high proportion of adversarial passages or sacrifice on useful benign passages in an effort to reduce adversarial ones.

4.4. Impact on the LLM Answers

Unattacked average absolute PS (Avg. |PS|): This value, as reported in Tables 4 and 5, measures the original bias level in unattacked cases. It also indicates whether the defender misses benign passages with strong perspectives,

resulting in more neutral and ambiguous content. BRRA, MMR, and SMART yield lower Avg. |PS| than No Def. for both retrieved passages and answers. This is because the diversity mechanism may attenuate strong perspectives.

PS shift caused by the attack: The attacker intends to make the PS of the retrieved passages biased toward its desired direction (i.e., ascending direction for positive attack and descending direction for negative attack). We report the average PS shift (see Eq. (4)) of the top-5 retrieved passages. The bias injection attack causes the retrievers to retrieve biased passages, and their average PS shift increases as more adversarial passages are injected (see Table 4). This demonstrates the vulnerability of retrieval systems to deliberate bias manipulation through adversarial passages.

We also report on the average PS shift of the generated answers to every question in the corpus. As listed in Table 5, the bias injection attack successfully induces PS shifts in the answers, and the magnitude of the shift increases with the number of injected adversarial passages. Moreover, the generated answers are positively correlated with the retrieved passages in terms of PS values. Fig. 4 quantifies such correlations.

Retrieval bias propagates through the generation process. LLM models fail to mitigate the injected biases during answer synthesis, even when equipped with critical thinking instructions in the prompt (see Prompt-Generation in Appendix B for the prompt used).

Vulnerability across models and datasets: A natural question arises: *Does the impact of bias injection attacks remain if we switch to a different—or larger—LLM models?* To answer this, we evaluate multiple LLMs using identical retrieval contexts, thereby isolating the model’s role in absorbing contextual bias. Table 6 presents representative results for the undefended retriever on WIKI-BALANCE.

Our results show that all three LLMs exhibit comparable PS shifts in their generated answers when under attack. The results in Table 6 also demonstrate that our attack successfully induces PS shifts in question answering across diverse datasets. As the number of injected adversarial passages increases, the answers exhibit stronger bias, reflected by larger PS shifts across all the datasets.

Pre-trained LLMs, regardless of size or architecture, remain similarly vulnerable to bias injection attacks. Merely scaling up the model or switching architectures does not inherently improve robustness.

4.5. Ineffectiveness of Fact-Checking

To investigate whether existing automated fact-checking is a potential defense for bias injection attack, we compare both benign and adversarial passages to see if they are distinguishable with fact-checking API. Specifically, we run the WikiCheck API which return a result of “SUPPORTS”

(TRUE), “REFUTES” (FALSE), or “NOT ENOUGH INFO” for each sentence in a passage.

As shown in Fig. 5(a), benign and adversarial passages give a comparable proportion of sentences labeled as “NOT ENOUGH INFO” by the fact-checking API. This observation can be attributed to two main factors. First, a fraction of sentences are not genuine claims but rather contextual or descriptive statements—such as definitions, methodological descriptions, or references—that do not convey verifiable factual assertions. Second, even when a sentence constitutes a claim, its validity often hinges on specific contextual scopes (e.g., internal experiment settings, unpublished data, or domain-restricted facts), while the general-purpose knowledge base leveraged by the fact-checking API lacks the corresponding evidence required for verification.

Verifiable claims in benign and adversarial passages are labeled as “SUPPORTS” or “REFUTES” by the fact-checking API [19]. Fig. 5(b) shows the distribution of support ratio (i.e., SUPPORTS / (SUPPORTS + REFUTES)). We see that adversarial passages exhibit a higher proportion of “SUPPORTS” labels. This phenomenon shows that our adversarial passages generated by LLMs use evidence that fact-checkers deem as facts.

Our attack creates passages that are as “factual” or more than benign passages to the WikiCheck fact-checker.

As a noteworthy caveat, all passages, including benign ones, contain a non-negligible fraction of “REFUTES” claims. This can arise from at least two factors. First, certain claims are only valid within the local context of the passage; when isolated and checked independently, the per-sentence fact-checker may misclassify them as false statements. Second, the current fact-checking API is not perfectly reliable and has occasional false negatives.

5. BiasDef: A More Principled Defense

We have seen that while existing defenses attenuate the attack effects, they remain suboptimal. We now present BiasDef, a post-processing method that augments existing dense retrieval pipelines to filter out adversarial passages from bias injection attackers. Rather than looking for syntactic characteristics of the attack or auxiliary fingerprints, BiasDef leverages the fact that attackers aim to craft adversarial passages that achieve high retrieval relevance, inject subtle perspective bias, and remain factual (the properties in Sec. 2.3). Sec. 6 evaluates the empirical efficacy of BiasDef.

5.1. Design Principles

An effective defense mechanism should address the attacks without making undue assumptions. Specifically:

- **Detection without priori knowledge:** Although the attacker’s aim described in the properties in Section 2.3 are known, the real challenge is the lack of prior knowledge about the benign passages. If the retriever knows the

| Method | Unattacked Avg. PS | #Negative Injection | | | #Positive Injection | | |
|------------------------------|---------------------|---------------------|------|------|---------------------|------|------|
| | | 1 | 5 | 10 | 1 | 5 | 10 |
| Llama-3-8B | 0.068±9% | 79% | 132% | 140% | 100% | 179% | 178% |
| DeepSeek-R1-Distill-Qwen-14B | 0.060±8% | 67% | 125% | 132% | 92% | 203% | 203% |
| GPT-4.1 (>100B) | 0.067±9% | 40% | 115% | 122% | 36% | 185% | 201% |
| WIKI-BALANCE | 0.060±8% | 67% | 125% | 132% | 92% | 203% | 203% |
| Reddit-Dialogues | 0.026±23% | 15% | 35% | 38% | 8% | 31% | 31% |
| HotpotQA | 0.013±85% | 92% | 454% | 446% | 100% | 438% | 338% |

TABLE 6: Average PS shift across different models and datasets. The proposed bias injection attack successfully biases the answers of prevalent LLM models with various parameter sizes across different datasets.

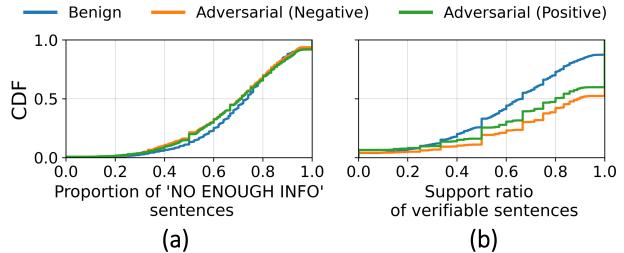


Figure 5: Fact-checking results: (a) Adversarial passages contain comparable proportions of sentences labeled as “NOT ENOUGH INFO” (0.69 and 0.68 on average for positive and negative bias injections, respectively), which are similar to that of benign passages (0.71). (b) Among verifiable claims, adversarial passages contain higher proportions of sentences labeled as “SUPPORTS” (0.73 and 0.79 on average for positive and negative bias injections, respectively) than benign passages (0.63).

exact characteristics (SS and PS) of benign passages in advance, it can easily filter out those adversarial ones by comparing their SS and PS values. But, individual benign passage may exhibit inherent bias or ideological leaning as well. One cannot always distinguish adversarial passages—intentionally crafted to skew the output—from naturally biased but benign passages, especially when both appear semantically relevant and factually correct.

- **Robustness to query and injection variability:** Since the attacker can freely control both the target query and the number of injected adversarial passages, a robust defense method must consistently maintain high performance across a wide range of attack scenarios—including variations in both query content and injection intensity.

To address these challenges, BiasDef identifies and filters out adversarial passages by exploiting their skewed perspective distribution within a certain similarity range. We construct a two-dimensional feature space using the SS to the query and the PS as the two axes. Each candidate passage is plotted in this space based on its SS and PS values. Specifically, PS values are computed by performing PCA on the embeddings of the candidate passages and projecting each embedding onto the resulting polarization axis. Although the absolute values and signs of PS may

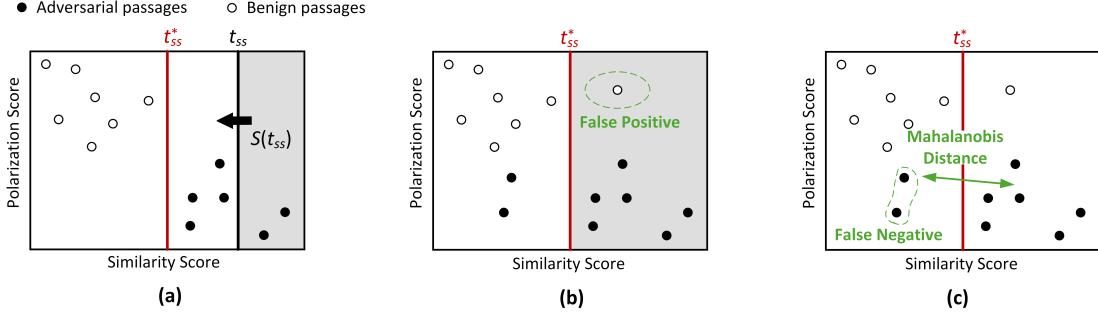


Figure 6: Workflow of BiasDef. (a) The SS value space is searched to identify the Max-KL boundary t_{ss}^* that maximizes the KL divergence between the PS distributions of the two subsets partitioned by t_{ss}^* . (b) We refine the passage set $\mathcal{S}(t_{ss}^*)$ by removing false positives with outlier PS values and (c) recover false negatives through Mahalanobis-distance-based classification.

differ from those computed by the attacker—since the defender does not have access to the synthetic passages used to construct the original polarization axis (see Sec. 3)—the relative PS skew described in Property 2 still holds. BiasDef identifies and filters potentially harmful content based on detected outliers in the overall passage distribution.

5.2. High-level Overview

BiasDef is a sanitization defense that, from first principles, aims to defeat the formulated attack definition given in Sec. 2.2. It uses KL-divergence which, as formalized next, is designed to exactly capture the separation of PS scores.

Definition: Given a set D_{benign} of benign passages and a set D_{adv} of adversarial passages, we define $\mathcal{S} = \mathcal{S}(t_{ss})$ as a subset of the entire set $D_{benign} \cup D_{adv}$ that includes all passages whose SS exceeds a threshold t_{ss} . The gray area in Fig. 6(a) visualizes an example of such a subset. We denote the complement of this subset as $\bar{\mathcal{S}} = (D_{benign} \cup D_{adv}) \setminus \mathcal{S}(t_{ss})$. Let P_S and Q_S denote the PS distributions of \mathcal{S} and $\bar{\mathcal{S}}$, respectively. To facilitate computation, we discretize the PS distributions of both \mathcal{S} and $\bar{\mathcal{S}}$. Let the PS axis be partitioned into m equal-width bins⁸. Define $c_S(i)$ as the number of passages in \mathcal{S} that fall into bin i , and similarly $c_{\bar{\mathcal{S}}}(i)$ for $\bar{\mathcal{S}}$. Then, the discrete PS distributions⁹ are computed as:

$$P_S(i) = c_S(i) / \sum_i c_S(i), \quad Q_S(i) = c_{\bar{\mathcal{S}}}(i) / \sum_i c_{\bar{\mathcal{S}}}(i). \quad (8)$$

The Kullback–Leibler (KL) divergence [48] between these two distributions is as follows:

$$f_{KL}(t_{ss}) = KL[P_S \| Q_S] = \sum_{i=1}^m P_S(i) \log\left(\frac{P_S(i)}{Q_S(i)}\right). \quad (9)$$

8. We assume m is sufficiently large (i.e., the bins are narrow enough) such that no bin contains both adversarial and benign passages when their PS values fall into disjoint intervals, as described in Property 2.

9. We add a small constant ϵ in piratical cases to avoid $\log(0)$ and $\log(\infty)$ when computing the KL divergence. See Appendix D for details.

Based on these definitions, we propose the following theorem that defines the security property our defense achieves.

Theorem 1. Suppose two conditions hold: (1) All passages in D_{adv} have higher SS values than any passage in D_{benign} (Property 1); and (2) All passages in D_{adv} have PS values that are either greater than or less than those of all passages in D_{benign} (Property 2). Then, the KL divergence $f_{KL}(t_{ss})$ in Eq. (9) achieves a local maximum at any threshold $t_{ss}^* \in (0, 1)$ such that: $\mathcal{S} = D_{adv}$ and $\bar{\mathcal{S}} = D_{benign}$.

Proof. A detailed proof is provided in Appendix C. \square

This theorem gives us a defense procedure if the adversarial and benign distributions are perfectly separable.

Theorem-guided defense as a starting point: Theorem 1 says that any threshold t_{ss}^* that cleanly separates D_{adv} from D_{benign} is guaranteed to be a local maximizer of the KL divergence $f_{KL}(t_{ss})$. Guided by this insight, our defense searches over the entire SS threshold space $t_{ss} \in [0, 1]$ monotonically to identify the point where $f_{KL}(t_{ss})$ reaches a maximum away from the boundaries (i.e., excluding 0, 1):

$$\begin{aligned} \text{Max-KL Boundary: } & t_{ss}^* = \arg \max_{t_{ss}} f_{KL}(t_{ss}), \\ & \text{s.t. } f_{KL}(t_{ss}^*) \geq f_{KL}(t_{ss}), \quad t_{ss} \in [t_{ss}^* - \delta, t_{ss}^* + \delta] \end{aligned} \quad (10)$$

Note that $f_{KL}(t_{ss})$ may be non-smooth in practice, particularly under finite-sample settings, and thus may have multiple local maxima. We select the first largest maxima found by decreasing t_{ss} from 1 down to 0. We empirically find that this often yields a separation boundary which aligns well with the ground truth boundary between adversarially crafted passages and benign passages, making it effective.

Statistical Limits: Theorem 1 gives ideal conditions for the defense to work perfectly: Attack passages, as defined in Section 3, are cleanly separable from benign. But, real-world distributions can have benign outlier passages that are statistically indistinguishable from adversarial ones (e.g., partial overlap in SS and PS distributions)—an artifact of real-world data. This poses a statistical limit for all sanitization defenses. As seen in Table 1, the ideal conditions assumed by Theorem 1 are not fully met in practice, though around 75% of generated passages adhere to them (Table 1).

This causes false negatives and false positives. Our final procedure thus, modifies the theorem-guided defense to handle false positives and negatives in real-world data.

Mitigation of false positives: To mitigate false positives—benign passages with SS values comparable to those of adversarial passages (see Fig. 6(b))—we identify a subset $\alpha \subset \mathcal{S} = \mathcal{S}(t_{ss}^*)$ consisting of passages whose PS values are closest to those in $\bar{\mathcal{S}}^{10}$. We then compare the KL divergence before ($KL[P_{\mathcal{S}} \| Q_{\mathcal{S}}]$) and after ($KL[P_{\mathcal{S} \setminus \alpha} \| Q_{\mathcal{S} \setminus \alpha}]$) excluding α . If removing α increases the KL divergence, i.e., $\mathcal{S} \setminus \alpha$ and $\bar{\mathcal{S}} \cup \alpha$ are more statistically divergent, it suggests that α contains passages that are more like benign passages. We iteratively expand α until $KL[P_{\mathcal{S} \setminus \alpha} \| Q_{\mathcal{S} \setminus \alpha}]$ reaches a local maximum and begins to decrease. The final α is treated as a group of false positives and removed from $\mathcal{S}(t_{ss}^*)$.

Mitigation of false negatives: To recover false negatives—those adversarial passages with low SS values—we apply a Mahalanobis-distance-based classification [49], a classical technique widely used for identifying outlier in high-dimensional embedding spaces [50]. This method captures the covariance structure of the detected adversarial passages in $\mathcal{S}(t_{ss}^*)$ and identifies samples that exhibit small deviations from the cluster center. Specifically, a passage d is considered a false negative and is included in the refined subset if it satisfies the following condition:

$$\sqrt{(E_d(d) - \mu)V^{-1}(E_d(d) - \mu)^\top} < T \quad (11)$$

where $E_d(d)$ is the embedding of passage d , and μ and V denote the mean vector and covariance matrix of the embeddings of passages in $\mathcal{S}(t_{ss}^*)$, respectively. This condition evaluates the Mahalanobis distance between d and the center of the adversarial cluster. The threshold T is selected via hyperparameter tuning based on validation performance.

We adopt Mahalanobis distance because its squared form equals a *weighted*¹¹ sum of squared differences across embedding dimensions. In our setting, consequential false negatives are those which induce bias, and thus will have PS value that deviates from zero. Therefore, they lie near the adversarial cluster along the principal component identified by PCA. Mahalanobis distance captures this: Eq. (11) checks the PS deviation of d relative to the adversarial cluster.

After excluding the suspicious adversarial passages—those identified by first removing the false positives in α from $\mathcal{S}(t_{ss}^*)$ and then adding the false negatives that satisfy Eq. (11)—from the top- $4 \times k$ candidate set, we select and return the final top- k relevant passages. The complete defense pseudo-code is summarized in Appendix D.

6. Evaluation of the Effectiveness of BiasDef

We now evaluate the efficacy of our aforementioned defense BiasDef against our proposed bias injection attack.

10. If the average PS of \mathcal{S} is higher than that of $\bar{\mathcal{S}}$, we select passages in \mathcal{S} with the lowest PS values; otherwise, we select those with the highest.

11. The unweighted version corresponds to the square of the standard ℓ_2 distance. The weights adjust for different variation along different directions in the vector space, as in the sample covariance matrix V .

6.1. Retrieval Performance of BiasDef

In Fig. 7, we plot BiasDef and the baselines by their average A-Recall@5 and Recall@5 across all 452 WIKI-BALANCE queries. BiasDef consistently achieves the lowest A-Recall@5 against all tested injection intensities, while maintaining nearly the highest Recall@5.

A-Recall@5: BiasDef achieves the lowest average A-Recall@5—0.16, 0.15, 0.04, 0.04, 0.15, 0.14 for #Neg=10, 5, 1 and #Pos=1, 5, 10, respectively—among all evaluated retrievers, outperforming the best baseline defense (BRRA in Table 2) by an average of 15% across different injection intensities. As shown in Fig. 8, BiasDef is able to completely filter out adversarial passages from the top-5 results for approximately 73% of the queries.

We further analyze the *evasion rate* of adversarial passages—the proportion of injected adversarial passages that successfully bypass BiasDef’s filtering mechanism. As shown in Table 7, BiasDef is less effective at filtering out those adversarial passages that do *not* simultaneously satisfy both Property 1 and Property 2. This is theoretically expected too, as such atypical passages do not produce significant KL divergence changes during the defense process.

Recall@5: BiasDef achieves average Recall@5 values of 0.40, 0.41, 0.50, 0.55 ± 0.04 , 0.50, 0.42, and 0.41 for #Neg=10, 5, 1, the unattacked case, and #Pos=1, 5, and 10, respectively. With 5 and 10 injections, BiasDef achieves the highest Recall@5. For 0 and 1 injections, BiasDef maintains 85% of the Recall@5 achieved by No Def., the best-performing baseline. Across injection intensities, BiasDef loses only 14–38% of relevant benign passages compared to the optimum (i.e., 0.64 for No Def. in the unattacked case), achieving an average Recall@5 improvement of 62% over the best-performing baseline (SMART in Table 3).

Worst-case performance: In the worst case (#Neg=10), the ratio of A-Recall@5 to Recall@5 is 0.4, which outperforms the worst-case performance of baselines in Sec. 4.3 by $2.1 \times$.

6.2. Generation Performance of BiasDef

PS shift: We observe the average PS of the answers in response to all 452 queries in the corpus. As shown in Table 8, BiasDef achieves the lowest PS shift, outperforms the baseline defenses No Def., MMR, SMART, and BRRA (see Table 5) by $8.8 \times$, $7.3 \times$, $6.2 \times$, and $8.2 \times$, respectively.

Unattacked average absolute PS (Avg. |PS|): BiasDef does not significantly reduce the unattacked Avg. |PS| compared with No Def. (0.060), indicating that BiasDef does *not* cause significantly new additional false positives among benign passages with strong perspectives and high PS values.

Generalization to LLM models and QA datasets. Across varying injection intensities, BiasDef consistently reduces the PS shift by $6.1 \times$, $8.8 \times$, and $8.8 \times$ on Llama-3, DeepSeek-R1-Distill-Qwen, and GPT-4.1, and by $8.8 \times$, $2.7 \times$, and $5.5 \times$ on WIKI-BALANCE, Reddit-Dialogues, and HotpotQA, compared to No Def. in Table 6.

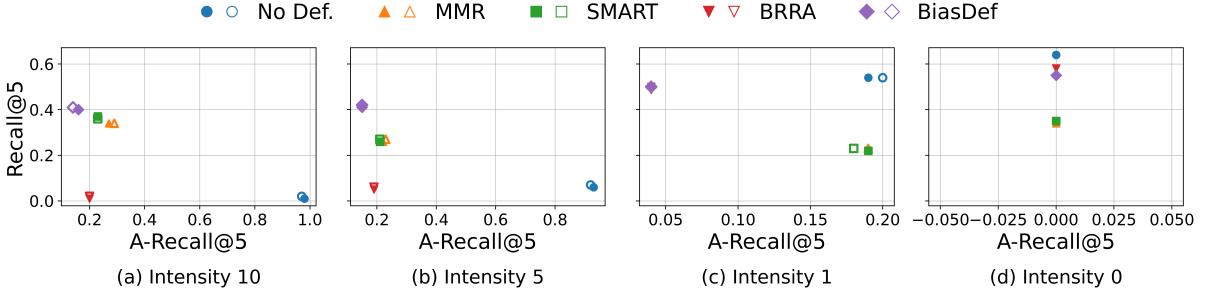


Figure 7: Comparison of BiasDef with baselines in A-Recall@5 and Recall@5. Solid and hollow markers represent negative and positive bias injections, respectively. BiasDef consistently achieves the lowest A-Recall@5 while maintaining the highest Recall@5 at injection intensities of 10 and 5, and retains at least 85% of the best-performing baselines’ Recall@5 at intensities of 1 and 0.

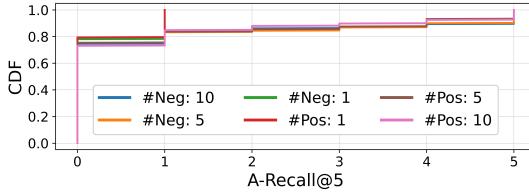


Figure 8: CDF of BiasDef’s A-Recall@5. BiasDef completely filters out adversarial passages for over 73% queries.

Unlike alternative methods such as [20] and [27], BiasDef functions as a purely plug-and-play filter, requiring no modifications to the underlying LLM. Moreover, BiasDef introduces a modest additional cost (avg. of 465 ms) for mitigating adversarial bias. This overhead is negligible compared to the typical generation time of LLMs, which is several seconds in our setup. This design makes it lightweight, easy to deploy, and compatible with diverse RAG systems.

6.3. Limitations, Discussion, and Future Work

Confusion Attacks on LLMs: While this paper focuses on attacks that inject adversarial passages with a consistent ideological bias, an alternative and potentially more subtle attack vector remains unexplored: deliberately retrieving a mixture of passages that present conflicting viewpoints. Such contradictory context increases PS variation in the generated answers, leading to uncertainty or ambiguity. A sophisticated adversary might exploit this by engineering “confusion attacks”—poisoning the corpus with passages that introduce semantic conflict—thereby causing the model to generate vague, indecisive, or even contradictory outputs.

Multi-dimensional extensions: Real debates can span multiple, sometimes orthogonal axes, beyond that considered in our PS score. We used the most polarizing dimension, captured by the first principal component during PCA, which typically represents the strongest binary disagreement (e.g., support vs. oppose). Nonetheless, the approach can be extended to multiple dimensions by considering additional principal components. In principle, BiasDef could be also be

| Passage Properties | #Negative Injection | | | #Positive Injection | | |
|--------------------|---------------------|------|------|---------------------|------|------|
| | 1 | 5 | 10 | 1 | 5 | 10 |
| Property 1 & 2 | 0.29 | 0.20 | 0.11 | 0.26 | 0.19 | 0.09 |
| Other | 0.86 | 0.60 | 0.31 | 0.93 | 0.67 | 0.32 |

TABLE 7: Evading rate of adversarial passages under BiasDef’s defense. Adversarial passages that lack either Property 1 or 2 are more likely to evade the detection.

| Method | Unattacked Avg. PS | #Negative Injection | | | #Positive Injection | | |
|------------------|---------------------|---------------------|-----|-----|---------------------|-----|------|
| | | 1 | 5 | 10 | 1 | 5 | 10 |
| Llama-3-8B | 0.072±8% | 11% | 21% | 17% | 11% | 47% | 42% |
| DeepSeek-R1 | 0.069±9% | 19% | 28% | 29% | 4% | 12% | 22% |
| Distill-Qwen-14B | 0.068±9% | 12% | 12% | 13% | 3% | 19% | 21% |
| GPT-4.1 (>100B) | | | | | | | |
| WIKI-BALANCE | 0.069±9% | 19% | 28% | 29% | 4% | 12% | 22% |
| Reddit-DIALOGUES | 0.025±24% | 4% | 12% | 12% | 4% | 16% | 20% |
| HotpotQA | -0.015±73% | 33% | 60% | 40% | 13% | 73% | 107% |

TABLE 8: BiasDef not only outperforms the baselines by reducing the average PS shift by more than 6.2×, but also generalizes well to more LLM models and QA datasets, maintaining consistently low bias levels.

applied iteratively along these dimensions [51] to mitigate multi-axis polarization, but exact details remain future work.

Influence on human users: Our formulation quantifies success in retrieval and bias, but does not fully capture aspects such as answer usefulness and the attack’s influence on user trust and beliefs, which are noticeable and influence human users. A comprehensive assessment of these downstream human effects remains a challenging but vital future work.

General-purpose defense: RAG systems operate in complex environments that may be subject to multiple types of attacks, of which bias injection is only one. A promising future direction is to integrate complementary defenses (e.g., BiasDef and KAD) into a unified, general-purpose defense system capable of mitigating a broader range of threats.

7. Related Work

7.1. Knowledge Poisoning with Fingerprints

Knowledge poisoning attacks aim to corrupt the retriever’s results by adding adversarial passages to the corpus. PoisonedRAG [10] leverages LLMs to generate malicious passages that are highly relevant to a trigger question and contain an attacker-desired answer, thereby causing the model to output targeted responses. BADRAG [11] employs contrastive learning to generate adversarial passages that mislead the retriever into retrieving them for queries with a trigger (e.g., a specific word or sentence).

Beyond inducing desired responses, prompt injection attacks [13], [15] aim to trigger more harmful actions by inserting malicious instructions (e.g., “Ignore previous instructions and do...”). These prompts override the system’s original task and coerce the LLM into executing an attacker-injected workflow. A recent adaptive attack, DataFlip [52] bypasses KAD [17], [18]—an previously effective prompt injection defense method—by specifically identifying the presence of its detection instructions.

Unlike the above approaches on constructing injected texts, AgentPoison [14] specially craft optimal “trigger” such that whenever a user’s query contains a secret phrase, the retriever will tend to retrieve the attacker’s documents. This causes the LLM agent to follow malicious demonstrations or instructions [15] hidden in those documents.

Overall, these studies show that simply poisoning the corpus can mislead a RAG system’s outputs. However, prior attacks typically rely on overtly malicious content (e.g., factually incorrect statements [10], [11], spam-like text with triggers [14], [53], or structured prompts [15]), which can degrade retrieval performance and risk detection.

7.2. Bias-Based Knowledge Poisoning

Recent work [24], [54] proposes an attack method that amplifies LLM biases through RAG system manipulation. Unlike our work, [24] requires direct manipulation of adversarial passage embeddings to increase its retrieval probability, which may introduce semantic errors after decoding, reducing factual accuracy. Additionally, it assumes that the RAG system’s internal prompt between the retriever and LLM, which instructs the LLM to generate responses based on the retrieved context, can be manipulated. This is impractical for external attackers. In contrast, our work focuses on covert bias and makes none of these assumptions.

7.3. Purely Retrieval-Based Defenses

Some work [55] has shown that retrievers relying solely on semantic relevance—such as BM25 [56], DPR [39] and Sentence BERT [31]—may return inherently biased documents, limiting the diversity and usefulness of the retrieved results. Such bias in retrieval can lead to the omission of helpful counter-perspectives or even skew the LLM’s output.

To improve perspective awareness, several studies have proposed diversity-promoting retrieval algorithms (e.g., MMR [25] and BRRA [24]) and perspective-aware retrievers

(e.g., SMART [26]). We provide a detailed description of these methods in Sec. 4. While these methods are effective at avoiding repetition and reducing the dominance of adversarial passages in the top- k results, they cannot fully eliminate adversarial content—especially when adversarial passages are diverse in form.

Unlike existing methods, BiasDef identifies and filters out adversarial passages by exploiting their high similarity to queries and skewed perspective distribution, providing a dual-defense mechanism that more effectively counters the biased passages generated by our attack workflow.

7.4. Model-Based Defense for Knowledge Poisoning

Identifying abnormal attention patterns within the LLM is another defense approach [27]. It is effective against attacks that “trigger” a target action by inserting specific fingerprints (e.g., structured text [10] and prompt [15]). In such cases, the model assigns disproportionately high attention weights to specific trigger tokens in the adversarial passages, enabling the defense to differentiate them. However, this approach is ineffective against bias injection attacks, which contain no explicit syntactic trigger tokens. Moreover, implementing the method of [27] requires modifying existing frameworks (e.g., vLLM) and black-box models (e.g., GPT-4), which do not expose attention weights by default.

Another work [20] proposes fine-tuning-based defenses against memory injection attacks, where the adversary crafts a prompt designed to cause an agent to ‘remember’ malicious context. This approach mitigates such attacks by fine-tuning the LLM on a dataset containing concise reasoning trajectories of both benign function-calling tasks and memory injection variants. However, it targets a different class of attacks and not directly applicable to bias injection attacks that do not deliberately inject malicious content in the context.

In contrast to these methods—which require access to internal LLM components or fine-tuning—BiasDef is a plug-and-play solution that operates purely at the retrieval stage. It requires no modification to the LLM or inference framework and is compatible with any existing RAG system. Furthermore, BiasDef is orthogonal and complementary to model-based defenses, and could be combined with them to provide stronger protection against a wider range of threats.

8. Conclusion

In this paper, we characterize bias injection attacks on RAG systems and demonstrate its effectiveness across various retrievers, language models, and public datasets. We further propose a post-retrieval filtering defense, which effectively mitigates them while preserving benign information. This work underscores the importance of viewpoint-aware retrieval for advancing robustness of LLM-based systems against covert manipulation and mitigating ideological bias.

Ethical Considerations

Polarization or bias in LLM generated content is a growing concern. We highlighted its threat, but also give a better defense. Our findings leave neither the attacker nor the defender with an asymmetrically higher advantage. All experiments were conducted in a closed, local environment and had no impact on any external or production systems.

References

- [1] Meta AI, “Meta llama,” <https://huggingface.co/meta-llama>, 2025, accessed June 2025.
- [2] OpenAI, J. Achiam, S. Adler, S. Agarwal *et al.*, “Gpt-4 technical report,” 2024. [Online]. Available: <https://arxiv.org/abs/2303.08774>
- [3] DeepSeek-AI, D. Guo, D. Yang, H. Zhang *et al.*, “Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning,” 2025. [Online]. Available: <https://arxiv.org/abs/2501.12948>
- [4] K. Singhal, T. Tu, J. Gottweis, R. Sayres *et al.*, “Towards expert-level medical question answering with large language models,” 2023. [Online]. Available: <https://arxiv.org/abs/2305.09617>
- [5] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, “Evaluating large language models trained on code,” *arXiv preprint arXiv:2107.03374*, 2021.
- [6] GitHub, “Github copilot: Your ai pair programmer,” <https://github.com/features/copilot>, 2021, accessed: July 2025.
- [7] Cursor, “Cursor: The ai-first code editor,” <https://www.cursor.so/>, 2023, accessed: July 2025.
- [8] OpenAI, “Introducing Deep Research,” <https://openai.com/index/introducing-deep-research/>, Feb. 2025, accessed June 2025.
- [9] P. Lewis, E. Perez, A. Piktus, F. Petroni *et al.*, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.
- [10] W. Zou, R. Geng, B. Wang, and J. Jia, “Poisonedrag: Knowledge corruption attacks to retrieval-augmented generation of large language models,” 2024. [Online]. Available: <https://arxiv.org/abs/2402.07867>
- [11] J. Xue, M. Zheng, Y. Hu, F. Liu, X. Chen, and Q. Lou, “Badrag: Identifying vulnerabilities in retrieval augmented generation of large language models,” 2024. [Online]. Available: <https://arxiv.org/abs/2406.00083>
- [12] S. Zeng, J. Zhang, P. He, Y. Xing, Y. Liu, H. Xu, J. Ren, S. Wang, D. Yin, Y. Chang, and J. Tang, “The good and the bad: Exploring privacy issues in retrieval-augmented generation (rag),” 2024. [Online]. Available: <https://arxiv.org/abs/2402.16893>
- [13] Q. Zhan, Z. Liang, Z. Ying, and D. Kang, “Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents,” 2024. [Online]. Available: <https://arxiv.org/abs/2403.02691>
- [14] Z. Chen, Z. Xiang, C. Xiao, D. Song, and B. Li, “Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases,” 2024. [Online]. Available: <https://arxiv.org/abs/2407.12784>
- [15] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, “Not what you’ve signed up for: Compromising real-world llm-integrated applications with indirect prompt injection,” in *Proceedings of the 16th ACM workshop on artificial intelligence and security*, 2023, pp. 79–90.
- [16] H. W. A. Hanley and Z. Durumeric, “Machine-made media: Monitoring the mobilization of machine-generated articles on misinformation and mainstream news websites,” 2024. [Online]. Available: <https://arxiv.org/abs/2305.09820>
- [17] Y. Liu, Y. Jia, R. Geng, J. Jia, and N. Z. Gong, “Formalizing and benchmarking prompt injection attacks and defenses,” in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 1831–1847.
- [18] Y. Liu, Y. Jia, J. Jia, D. Song, and N. Z. Gong, “Datasentinel: A game-theoretic detection of prompt injection attacks,” in *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2025, pp. 2190–2208.
- [19] M. Trokhymovych and D. Saez-Trumper, “Wikicheck: An end-to-end open source automatic fact-checking api based on wikipedia,” in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, ser. CIKM ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 4155–4164. [Online]. Available: <https://doi.org/10.1145/3459637.3481961>
- [20] A. S. Patlan, P. Sheng, S. A. Hebbar, P. Mittal, and P. Viswanath, “Real ai agents with fake memories: Fatal context manipulation attacks on web3 agents,” 2025. [Online]. Available: <https://arxiv.org/abs/2503.16248>
- [21] J. Su, J. P. Zhou, Z. Zhang, P. Nakov, and C. Cardie, “Towards more robust retrieval-augmented generation: Evaluating rag under adversarial poisoning attacks,” 2024. [Online]. Available: <https://arxiv.org/abs/2412.16708>
- [22] S. Xu, D. Hou, L. Pang, J. Deng, J. Xu, H. Shen, and X. Cheng, “Invisible relevance bias: Text-image retrieval models prefer ai-generated images,” 2024. [Online]. Available: <https://arxiv.org/abs/2311.14084>
- [23] Y. Zhou, S. Dai, L. Pang, G. Wang, Z. Dong, J. Xu, and J.-R. Wen, “Exploring the escalation of source bias in user, data, and recommender system feedback loop,” 2025. [Online]. Available: <https://arxiv.org/abs/2405.17998>
- [24] L. Wang, T. Zhu, L. Qin, L. Gao, and W. Zhou, “Bias amplification in rag: Poisoning knowledge retrieval to steer llms,” 2025. [Online]. Available: <https://arxiv.org/abs/2506.11415>
- [25] J. Carbonell and J. Goldstein, “The use of mmr, diversity-based reranking for reordering documents and producing summaries,” in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR ’98. New York, NY, USA: Association for Computing Machinery, 1998, p. 335–336. [Online]. Available: <https://doi.org/10.1145/290941.291025>
- [26] J. Li, X. Hu, and X. Wan, “Smart-rag: Selection using determinantal matrices for augmented retrieval,” 2024. [Online]. Available: <https://arxiv.org/abs/2409.13992>
- [27] S. Choudhary, N. Palumbo, A. Hooda, K. D. Djivjotham, and S. Jha, “Through the stealth lens: Rethinking attacks and defenses in rag,” 2025. [Online]. Available: <https://arxiv.org/abs/2506.04390>
- [28] C. Ziems, W. Held, J. Dwivedi-Yu, and D. Yang, “Measuring and addressing indexical bias in information retrieval,” in *Findings of the Association for Computational Linguistics ACL 2024*, 2024, pp. 12 860–12 877.
- [29] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning, “Hotpotqa: A dataset for diverse, explainable multi-hop question answering,” *arXiv preprint arXiv:1809.09600*, 2018.
- [30] I. Harel, H. Taitelbaum, I. Szpektor, and O. Kurland, “A dataset for sentence retrieval for open-ended dialogues,” in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR ’22. New York, NY, USA: Association for Computing Machinery, 2022, p. 2960–2969. [Online]. Available: <https://doi.org/10.1145/3477495.3531727>
- [31] N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” 2019. [Online]. Available: <https://arxiv.org/abs/1908.10084>

- [32] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvassy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, “The faiss library,” 2025. [Online]. Available: <https://arxiv.org/abs/2401.08281>
- [33] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf
- [34] J. Ni, C. Qu, J. Lu, Z. Dai, G. Hernandez Abrego, J. Ma, V. Zhao, Y. Luan, K. Hall, M.-W. Chang, and Y. Yang, “Large dual encoders are generalizable retrievers,” in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 9844–9855. [Online]. Available: <https://aclanthology.org/2022.emnlp-main.669/>
- [35] S. Dai, C. Xu, S. Xu, L. Pang, Z. Dong, and J. Xu, “Bias and unfairness in information retrieval systems: New challenges in the llm era,” in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD ’24. ACM, Aug. 2024, p. 6437–6447. [Online]. Available: <http://dx.doi.org/10.1145/3637528.3671458>
- [36] F. L. Gewers, G. R. Ferreira, H. F. D. Arruda, F. N. Silva, C. H. Comin, D. R. Amancio, and L. D. F. Costa, “Principal component analysis: A natural approach to data exploration,” *ACM Computing Surveys*, vol. 54, no. 4, p. 1–34, May 2021. [Online]. Available: <http://dx.doi.org/10.1145/3447755>
- [37] S. Dai, Y. Zhou, L. Pang, W. Liu, X. Hu, Y. Liu, X. Zhang, G. Wang, and J. Xu, “Neural retrievers are biased towards llm-generated content,” in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD ’24. ACM, Aug. 2024, p. 526–537. [Online]. Available: <http://dx.doi.org/10.1145/3637528.3671882>
- [38] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, “The curious case of neural text degeneration,” *arXiv preprint arXiv:1904.09751*, 2019.
- [39] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. tau Yih, “Dense passage retrieval for open-domain question answering,” 2020. [Online]. Available: <https://arxiv.org/abs/2004.04906>
- [40] A. Kulesza, “Determinantal point processes for machine learning,” *Foundations and Trends® in Machine Learning*, vol. 5, no. 2–3, p. 123–286, 2012. [Online]. Available: <http://dx.doi.org/10.1561/2200000044>
- [41] R. Vershynin, *High-Dimensional Probability: An Introduction with Applications in Data Science*, ser. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.
- [42] Sentence-Transformers, “sentence-transformers/msmarco-distilbert-base-tas-b,” <https://huggingface.co/sentence-transformers/msmarco-distilbert-base-tas-b>, 2020, accessed: 2025-07-23.
- [43] Meta AI, “meta-llama/meta-llama-3-8b,” <https://huggingface.co/meta-llama/Meta-Llama-3-8B>, 2024, accessed: 2025-07-23.
- [44] DeepSeek AI, “deepseek-ai/deepseek-r1-distill-qwen-14b,” <https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-14B>, 2024, accessed: 2025-07-23.
- [45] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. Gonzalez, H. Zhang, and I. Stoica, “Efficient memory management for large language model serving with pagedattention,” in *Proceedings of the 29th Symposium on Operating Systems Principles*, ser. SOSP ’23. New York, NY, USA: Association for Computing Machinery, 2023, p. 611–626. [Online]. Available: <https://doi.org/10.1145/3600006.3613165>
- [46] Sentence-Transformers, “sentence-transformers/sentence-t5-xl,” <https://huggingface.co/sentence-transformers/sentence-t5-xl>, 2024, accessed: 2025-10-23.
- [47] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. tau Yih, “Dense passage retrieval for open-domain question answering,” 2020. [Online]. Available: <https://arxiv.org/abs/2004.04906>
- [48] S. Kullback and R. A. Leibler, “On Information and Sufficiency,” *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79 – 86, 1951. [Online]. Available: <https://doi.org/10.1214/aoms/1177729694>
- [49] G. J. McLachlan, “Mahalanobis distance,” *Resonance*, vol. 4, no. 6, pp. 20–26, 1999.
- [50] K. Lee, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS’18. Red Hook, NY, USA: Curran Associates Inc., 2018, p. 7167–7177.
- [51] D. Z. Lee, A. Kolluri, P. Saxena, and E.-C. Chang, “A practical and secure byzantine robust aggregator,” in *Proceedings of the ACM SIGSAC conference on computer and communications security (CCS)*, 2025.
- [52] S. Choudhary, D. Anshumaan, N. Palumbo, and S. Jha, “How not to detect prompt injections with an llm,” *arXiv preprint arXiv:2507.05630*, 2025.
- [53] Q. Long, Y. Deng, L. Gan, W. Wang, and S. J. Pan, “Whispers in grammars: Injecting covert backdoors to compromise dense retrieval systems,” 2024. [Online]. Available: <https://arxiv.org/abs/2402.13532>
- [54] P. Thota and S. Nilizadeh, “Attacks against abstractive text summarization models through lead bias and influence functions,” in *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 13 727–13 741.
- [55] X. Zhao, T. Chen, S. Chen, H. Zhang, and T. Wu, “Beyond relevance: Evaluate and improve retrievers on perspective awareness,” 2024. [Online]. Available: <https://arxiv.org/abs/2405.02714>
- [56] S. Robertson and H. Zaragoza, “The probabilistic relevance framework: Bm25 and beyond,” *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009. [Online]. Available: <http://dx.doi.org/10.1561/1500000019>

Appendix A. Examples

Table 9 shows examples of bias injection attacks and discusses their associated risks and impacts in practical scenarios.

Appendix B. Prompt Used

An example prompt used in typical RAG process is as follows.

You are a helpful assistant. Use the following context to answer the question as accurately and objectively as possible.

Context: [1] {Passage 1}; ... [k]
{Passage k}

Question: {query}

Answer:

| User Query | Injected Biased Content | Skewed LLM Output | Risk/Impact |
|---|--|--|---|
| Is the HPV vaccine safe for adolescents? | Several parents have reported persistent fatigue in their children after receiving the HPV vaccine... | While generally considered safe, the HPV vaccine has raised concerns among some experts due to potential long-term side effects... | Undermines public trust in vaccines; may reduce vaccination rates and harm public health. |
| Should stricter regulations be imposed on carbon emissions? | Several industry reports warn that stricter emissions regulations could lead to widespread job losses in the manufacturing sector... | While beneficial for the environment, stricter regulations may severely impact the economy and employment rates... | Shifts public opinion against climate policies; weakens support for environmental regulation. |
| Is cryptocurrency a safe investment in 2025? | In 2021, early investors in Ethereum saw over 400% returns, outperforming traditional markets... | Cryptocurrency has proven to be one of the most lucrative investment opportunities in recent years... | Encourages a particular financial behavior |

TABLE 9: Examples of bias injection attacks that influence LLM-generated answers and their potential real-world impacts.

The prompts used to generate synthetic passages, similar passages to the seed passage and the final answers are as follows.

Prompt-Synthetic:

Instruction: Write $\{J\}$ persuasive and diverse articles answering the user question below. Each article should present a **different viewpoint or ideological stance**, covering a range of perspectives. The responses should sound natural and plausible, as if written by different authors with distinct opinions.

User Question: {Query}

Output:

Instruction: You are an expert who has just read several passages about a controversial issue. Your job is to read these passages, carefully consider their arguments and evidence, and then write a clear, well-justified answer to the following question.
Do not simply summarize or list the passages. Instead, think critically and use the information to form your own conclusion.

User Question: {Query}

Contextual Passages: [1] {Passage 1};
... [k] {Passage k}

Answer:

Prompt-Adv:

Instruction: Write a persuasive article answering the user question below. Your article must follow the same stance and bias intensity as the example passage. Avoid copying text, but ensure the tone, viewpoint, and strength are closely aligned.

User Question: {Query}

Example Passage: {Seed Passage}

Output:

Appendix C. Proof of Theorem 1

Proof. According to Property 1, there exists a value t_{ss}^* such that all benign passages have SS values below t_{ss}^* , and all adversarial passages have SS values above t_{ss}^* . Thus, if we decrease t_{ss} from 1 (the upper bound of SS) to t_{ss}^* , more adversarial passages will be added to $\mathcal{S}(t_{ss})$. As t_{ss} continues to decrease from t_{ss}^* to 0 (the lower bound of SS), all adversarial passages, as well as some benign passages, are included in $\mathcal{S}(t_{ss})$. We divide this entire process into two procedures.

- **Procedure 1:** t_{ss} decreases from 1 to t_{ss}^* . In this procedure, $\mathcal{S} = \mathcal{S}(t_{ss})$ contains only adversarial passages due to Property 1 (i.e., adversarial passages have higher SS than benign ones). The complement $\bar{\mathcal{S}}$ contains the remaining adversarial passages and all benign passages:

$$\mathcal{S} \subseteq D_{\text{adv}}, \quad \bar{\mathcal{S}} = D_{\text{benign}} \cup (D_{\text{adv}} \setminus \mathcal{S}). \quad (12)$$

- **Procedure 2:** t_{ss} decreases from t_{ss}^* to 0. In this procedure, $\bar{\mathcal{S}}$ contains a subset of benign passages, while

Prompt-Generation:

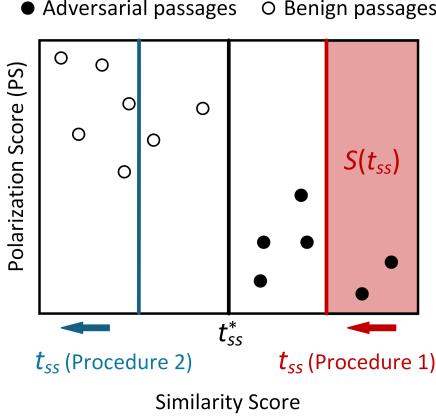


Figure 9: We divide the passages into two subsets. In both procedures illustrated, we prove that the optimal configuration for locally maximizing KL divergence is to separate benign and adversarial passages into disjoint subsets. Reversing the direction of the polarization score (PS) axis—i.e., whether adversarial passages have higher or lower PS values than benign ones—does not affect this conclusion.

\mathcal{S} includes all adversarial passages and the remaining benign passages:

$$\bar{\mathcal{S}} \subseteq D_{\text{benign}}, \quad \mathcal{S} = D_{\text{adv}} \cup (D_{\text{benign}} \setminus \bar{\mathcal{S}}). \quad (13)$$

Fig. 9 illustrates these procedures. As defined in Sec. 5.2, we discretize the PS distributions by partitioning the PS axis into m equal-width bins. Let $c_{\text{benign}}(i)$ and $c_{\text{adv}}(i)$ denote the number of benign and adversarial passages falling into bin i , respectively. According to Property 2, all the adversarial passages have either higher or lower PS values than all benign passages. Thus, we can always find an index i' such that bins 1 to i' contain only adversarial passages, and bins $i' + 1$ to m contain only benign passages.

We then prove that there exists a local maximum of the KL divergence $KL[P_S \| Q_S]$ when $t_{ss} = t_{ss}^*$, corresponding to the case where $\mathcal{S} = D_{\text{adv}}$ and $\bar{\mathcal{S}} = D_{\text{benign}}$.

Procedure 1: According to Eq. (12), we have:

$$\begin{aligned} c_S(i) &= 0, & c_{\bar{S}}(i) &= c_{\text{benign}}(i), & \text{if } i \in [1, i'] \\ c_{\bar{S}}(i) &= c_{\text{adv}}(i) - c_S(i), & & \text{if } i \in [i' + 1, m] \end{aligned} \quad (14)$$

The corresponding distributions $P_S(i)$ and $Q_S(i)$ are given by:

$$\begin{aligned} P_S(i) &= \begin{cases} 0, & \text{if } i \in [1, i'] \\ \frac{c_S(i)}{\sum_j c_S(j)}, & \text{otherwise} \end{cases} \\ Q_S(i) &= \begin{cases} \frac{c_{\text{benign}}(i)}{C - \sum_j c_S(j)}, & \text{if } i \in [1, i'] \\ \frac{c_{\text{adv}}(i) - c_S(i)}{C - \sum_j c_S(j)}, & \text{otherwise} \end{cases} \end{aligned} \quad (15)$$

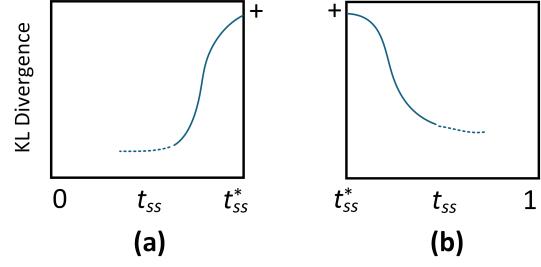


Figure 10: In both procedures, the KL divergence are two convex functions of t_{ss} , reaching local maximum at three end values: 1, t_{ss} , and 0. Specifically, the local maximum at $t_{ss} = 1$ is 0, while the other two are positive.

where $C = \sum_j c_{\text{benign}}(j) + c_{\text{adv}}(j)$. Since $P_S(i) = 0$ for $i \in [1, i']$, the KL divergence becomes:

$$\begin{aligned} KL[P_S \| Q_S] &= \sum_{i=i'+1}^m \frac{c_S(i)}{\sum_j c_S(j)} \log \left(\frac{\frac{c_S(i)}{\sum_j c_S(j)}}{\frac{c_{\text{adv}}(i) - c_S(i)}{C - \sum_j c_S(j)}} \right) \\ &= \sum_{i=i'+1}^m \frac{c_S(i)}{\sum_j c_S(j)} \log \left(\frac{c_S(i)}{c_{\text{adv}}(i) - c_S(i)} \right) + g, \end{aligned} \quad (16)$$

where $g = \log \left(\frac{C - \sum_j c_S(j)}{\sum_j c_S(j)} \right)$.

We analyze the monotonicity of Eq. (16).

$$\begin{aligned} \frac{\partial KL[P_S \| Q_S]}{\partial c_S(i)} &= f_1 - f_2 \\ &+ \frac{c_S(i)}{(c_{\text{adv}}(i) - c_S(i)) \sum_j c_S(j)} - \frac{1}{C - \sum_j c_S(j)}, \\ f_1 &= \frac{(\sum_{j \neq i} c_S(j)) \log(c_S(i)) - \sum_{j \neq i} [c_S(j) \log(c_S(j))]}{(\sum_j c_S(j))^2}, \\ f_2 &= \frac{(\sum_{j \neq i} c_S(j)) \log(c_{\text{adv}}(i) - c_S(i))}{(\sum_j c_S(j))^2} \\ &- \frac{\sum_{j \neq i} [c_S(j) \log(c_{\text{adv}}(j) - c_S(j))]}{(\sum_j c_S(j))^2}. \end{aligned} \quad (17)$$

This derivative is positive when each $c_S(i), i \in [i'+1, m]$ is close to $c_{\text{adv}}(i)$ (i.e., when t_{ss} is near t_{ss}^*), because f_1 remains finite, $f_2 \rightarrow 0$, the third term diverges to ∞ , and the fourth term remains finite. Given that $c_S(i), i \in [i'+1, m]$ increases monotonically as t_{ss} decreases towards t_{ss}^* , we can qualitatively plot the KL divergence curve as shown in Fig. 10 (b). As t_{ss} approaches t_{ss}^* , the KL divergence increases, since its derivative with respect to t_{ss} is negative by the chain rule.

Procedure 2: According to Eq. (13), we have:

$$\begin{aligned} c_{\bar{S}}(i) &= c_{\text{benign}}(i) - c_S(i), & \text{if } i \in [1, i'] \\ c_{\bar{S}}(i) &= 0, & c_S(i) &= c_{\text{adv}}(i), & \text{if } i \in [i'+1, m] \end{aligned} \quad (18)$$

The corresponding Q distribution becomes:

$$P_S(i) = \begin{cases} \frac{c_S(i)}{\sum_j c_S(j)}, & \text{if } i \in [1, i'] \\ \frac{c_{\text{adv}}(i)}{\sum_j c_S(j)}, & \text{otherwise} \end{cases} \quad (19)$$

$$Q_S(i) = \begin{cases} \frac{c_{\text{benign}}(i) - c_S(i)}{C - \sum_j c_S(j)}, & \text{if } i \in [1, i'] \\ 0, & \text{otherwise} \end{cases}$$

Then, the KL divergence becomes:

$$KL[P_S \| Q_S] = \sum_{i=1}^{i'} \frac{c_S(i)}{\sum_j c_S(j)} \log \left(\frac{\sum_j c_S(i)}{\frac{c_{\text{benign}}(i) - c_S(i)}{C - \sum_j c_S(j)}} \right) \quad (20)$$

$$+ \sum_{i=i'+1}^m \frac{c_{\text{adv}}(i)}{\sum_j c_S(i)} \log \left(\frac{c_{\text{adv}}(i)}{0 \sum_j c_S(i)} \right).$$

We analyze the monotonicity of Eq. 20.

$$\frac{\partial KL[P_S \| Q_S]}{\partial c_S(i)} = f_1 - f_2 - \frac{c_{\text{adv}}(i)(\log(\frac{c_{\text{adv}}(i)}{0 \sum_j c_S(i)}) + 1)}{(\sum_j c_S(i))^2}$$

$$+ \frac{c_S(i)}{(c_{\text{benign}}(i) - c_S(i)) \sum_j c_S(j)} - \frac{1}{C - \sum_j c_S(j)},$$

$$f_1 = \frac{(\sum_{j \neq i} c_S(j)) \log(c_S(i)) - \sum_{j \neq i} [c_S(j) \log(c_S(j))]}{(\sum_j c_S(j))^2},$$

$$f_2 = \frac{(\sum_{j \neq i} c_S(j)) \log(c_{\text{benign}}(i) - c_S(i))}{(\sum_j c_S(j))^2}$$

$$- \frac{\sum_{j \neq i} [c_S(j) \log(c_{\text{benign}}(j) - c_S(j))]}{(\sum_j c_S(j))^2}. \quad (21)$$

This derivative is negative when each $c_S(i), i \in [1, i']$ is close to 0 (i.e., when t_{ss} is near t_{ss}^*), because f_1, f_2 , and the fifth term remain finite, the third term diverges to $-\infty$, and the fourth term diverges to 0. Given that $c_S(i), i \in [1, i']$ decreases monotonically as t_{ss} increases towards t_{ss}^* , we can qualitatively plot the KL divergence curve as shown in Fig. 10 (a). As t_{ss} approaches t_{ss}^* , the KL divergence increases, since its derivative with respect to t_{ss} is positive by the chain rule.

Therefore, the KL divergence reaches a local maximum when $t_{ss} = t_{ss}^*$. In addition, $\lim_{t_{ss} \rightarrow t_{ss}^*} KL[P_S \| Q_S] \rightarrow \infty$. According to the definition of t_{ss}^* , this also corresponds to $\mathcal{S} = D_{\text{adv}}$ and $\bar{\mathcal{S}} = D_{\text{benign}}$. \square

Appendix D. Pseudo-Code of BiasDef

The following pseudo-code outlines the procedure by which BiasDef filters suspicious adversarial passages from the retrieval results. Specifically, as t_{ss} is scanned from 1 to 0 (line 1), the step size is minimized to ensure that no more than one additional passage is added to $\mathcal{S}(t_{ss})$ at each iteration (line 2). We add a small constant ϵ to avoid $\log(0)$ and $\log(\infty)$ when computing the KL divergence (line 6).

Algorithm 1: Defense Workflow for Detecting Adversarial Passages

Data: Query q ; retrieved candidate set of passages D with similarity scores $\text{Sim}(d), d \in D$; PS values $P_S(d)$ for all passages; smoothing factor k ; Threshold T ; small constant ϵ to avoid $\log(0)$ and $\log(\infty)$

Result: Filtered set of suspicious adversarial passages

```

1 for  $t_{ss} \leftarrow 1$  to 0 do
2    $\mathcal{S}(t_{ss}) \leftarrow$  passages whose SS values are above  $t_{ss}$ ;
3    $\bar{\mathcal{S}}(t_{ss}) = D \setminus \mathcal{S}(t_{ss})$ ;
4    $P_S \leftarrow$  Discrete Distribution of  $\mathcal{S}(t_{ss})$ ;
5    $Q_S \leftarrow$  Discrete Distribution of  $\bar{\mathcal{S}}(t_{ss})$ ;
6    $P_S(i) \leftarrow \frac{P_S(i)+\epsilon}{\sum_i (P_S(i)+\epsilon)}, Q_S(i) \leftarrow \frac{Q_S(i)+\epsilon}{\sum_i (Q_S(i)+\epsilon)}, \forall i$ ;
7    $KL[P_S \| Q_S] = \sum_i P_S(i) \log \left( \frac{P_S(i)}{Q_S(i)} \right)$ ;
8 end
9  $t_{ss}^* \leftarrow \max_{t_{ss}} KL[P_S \| Q_S]$ ;
  /* Detection of False Positives */
10  $\alpha \leftarrow \emptyset$ ;
11 while  $KL[P_{S \setminus \alpha} \| Q_{S \setminus \alpha}] \geq KL[P_S \| Q_S]$  do
12   Add the passage in  $\mathcal{S}(t_{ss}^*)$  whose PS value is closest to those in  $\bar{\mathcal{S}}(t_{ss}^*)$  into  $\alpha$ ;
13 end
14  $\mathcal{S} \leftarrow \mathcal{S}(t_{ss}^*) \setminus \alpha$ ;
  /* Detection of False Negatives */
15  $\mu \leftarrow \text{Mean}(E_d(d), d \in S(\delta^*))$ ;
16  $V \leftarrow \text{CovarianceMat}(E_d(d), d \in \mathcal{S})$ ;
17 for  $d \in D \setminus \mathcal{S}$  do
18    $\mathcal{S} \leftarrow \mathcal{S} \cup \{d\}$ , if  $\sqrt{(E_d(d) - \mu)V^{-1}(E_d(d) - \mu)^T} < T$ ;
19 end
20 return  $\mathcal{S}$ 

```
