

When Clusters Lie: Post-Partition Injection Breaks Partition-Based RAG Defenses

Md Tayeb Adnan, Shuvam Agarwala, Lorenzo De Carli and Hadis Karimipour

Department of Electrical and Software Engineering

University of Calgary, Calgary, Alberta, Canada

{mdtayeb.adnan,shuvam.agarwala,lorenzo.decarli,hadis.karimipour}@ucalgary.ca

Abstract—Retrieval-Augmented Generation (RAG) systems are widely deployed to ground large language model outputs in external knowledge, but their reliance on dynamically updated corpora introduces a critical security vulnerability. Recent defenses against corpus poisoning, particularly partition-based methods, assume that adversarial documents can be identified as outliers in embedding space. In this paper, we show that this assumption is fundamentally flawed under adaptive adversaries who interact with the retrieval interface. We present Post-Partition Injection (PPI), a corpus poisoning attack that exploits the structural gap between cluster formation and document insertion in clustering-based RAG defenses. By issuing probe queries to infer cluster structure, PPI constructs poisoned documents whose embeddings lie within benign clusters, thereby bypassing partition-based filtering while steering model outputs toward attacker-chosen answers. Unlike prior poisoning attacks that rely on outlier behavior, PPI operates entirely within the distribution of clean data, making it difficult to detect using existing defenses. To address this threat, we propose Partition-Aware Filtering (PAF). PAF anchors the cluster structure to a trusted corpus snapshot and flags documents inserted afterward whose embeddings deviate from the geometry of that snapshot. PAF integrates document-isolated generation with a single-pass Natural Language Inference (NLI)-based majority vote, providing a bounded robustness guarantee under partial adversarial control. Empirical results show that PAF significantly reduces attack success.

1. Introduction

Retrieval-Augmented Generation (RAG) [1] has become one of the dominant paradigms for enhancing Large Language Models (LLMs) with external, up-to-date knowledge sources, addressing the limitations of purely parametric memory [1], [2], [3], [4]. By retrieving relevant passages from an external corpus at inference time and conditioning generation on these retrieved documents, RAG systems improve factual accuracy, adaptability, and domain grounding compared to model fine-tuning alone. As a result, RAG has been widely deployed in real-world applications such as enterprise knowledge systems, legal analysis tools, and medical question answering systems. Prominent examples include Microsoft Bing Chat [5], Perplexity AI [6], and Google Search with AI-generated overviews, as well as

widely used development frameworks such as LangChain [7] and LlamaIndex [8]

Despite these advantages, the reliance on external corpora introduces a fundamental security vulnerability: the retrieval corpus is a writable and continuously accessible attack surface. Prior work demonstrates that injecting as few as five carefully crafted documents can steer downstream LLM outputs toward attacker-specified answers for targeted queries [9]. This class of threats, broadly known as corpus poisoning or knowledge injection attacks [9], [10], poses serious risks for any system that indexes externally contributed or frequently updated content. For example, in enterprise knowledge bases populated through ticketing systems or collaborative platforms (e.g., Confluence and Notion), employees or contractors with write access may act as insider adversaries. Similarly, legal and financial RAG systems that ingest public filings, web content, or third-party data feeds inherently rely on information sources that may be controlled by external and potentially adversarial actors.

In response, a growing line of defense mechanisms has been proposed. RobustRAG [11] introduced the first certifiable defense via an isolate-then-aggregate scheme with formal guarantees when fewer than half of retrieved passages are adversarial. More recent approaches, including TrustRAG [12], RAGDefender [13], SeCon-RAG [14], adopt partition-based strategies. These methods cluster the corpus in embedding space, at query time or over the indexed corpus, and filter documents that fall into anomalous or isolated clusters. The underlying intuition is that adversarially injected documents, which are typically optimized toward specific query embeddings, will be more similar to one another and will form anomalously tight groups or outlier clusters that can be detected and removed.

In this work, we challenge this intuition and show that this intuition fails under adaptive adversaries. The critical vulnerability we identify is the outlier assumption: partition-based defenses compute cluster structure from the documents present at partition time and treat that geometry as the benign reference. The live system, however, continues to expose a queryable retrieval interface and accept new documents. An adversary who interacts with this interface before injecting documents can infer the latent cluster structure from retrieval patterns alone, without any privileged access. Armed with this structural knowledge, the adversary can place poisoned documents inside benign clusters rather than as outliers,

defeating anomaly detection.

We formalize this strategy as Post-Partition Injection (PPI), a cluster-aware corpus poisoning attack designed to bypass partition-based RAG defenses. PPI operates in four stages: (i) Phase 1 (Probing): issue benign probe queries and collect co-retrieval statistics, (ii) Phase 2 (Cluster Inference): apply spectral methods to reconstruct the cluster partition, (iii) Phase 3 (Targeted Construction): generate poisoned documents with embeddings inside the target cluster; and (iv) Phase 4 (Attack Execution): inject after clustering is formed.

On all three standard benchmarks, PPI achieves an Attack Success Rate (ASR) of 50.9–79.2%. This demonstrates that PPI represents a qualitatively new threat: it defeats defenses that successfully block prior attacks.

We then present Partition-Aware Filtering (PAF), a defense designed specifically against PPI. PAF operates in three coordinated components: (A) it anchors the partition to a corpus snapshot so that frozen centroids cannot be circumvented by post-deployment injection timing; (B) it detects embedding drift in newly inserted documents relative to the frozen anchor geometry, quarantining injections before they enter retrieval; and (C) it applies document-isolated generation with a single-pass NLI majority vote that provides a bounded robustness certificate (Theorem 5.1) for the documents that pass the filter.

Our main contributions are:

- We identify a fundamental vulnerability in partition-based RAG defenses arising from the mismatch between clustering and adversarial corpus manipulation.
- We propose PPI, the first adaptive corpus poisoning attack that reconstructs latent partition structure from retrieval interaction signals and achieves in-cluster adversarial document placement with no privileged access.
- We propose PAF, a defense that mitigates PPI through corpus anchoring, drift-based insertion filtering, and certified NLI aggregation.
- We empirically demonstrate that existing partition-based defenses are insufficient under realistic query-access adversaries, and show that PPI substantially degrades their robustness while PAF effectively restores reliable performance.

The implementation of PPI and PAF, together with all evaluation scripts, can be found at https://anonymous.4open.science/r/when_clusters_lie-779B/README.md

2. Related Work

Corpus Poisoning Attacks. Corpus poisoning attacks compromise Retrieval-Augmented Generation (RAG) systems by inserting malicious documents into the retrieval corpus [15], [16], [17], [18], [19]. The foundational study of adversarial passage injection in RAG was presented by Zhong et. al. [20], who showed that documents optimized toward a target query embedding can reliably appear in the top- k results while asserting false answers. PoisonedRAG [9] operationalized this into a practical attack achieving over

90% ASR using five injected passages and provided the first systematic attack benchmark now widely used for evaluation. Prompt Injection Attacks (PIA) [21] embed adversarial instructions inside retrieved content to override the generator system prompt, representing a distinct but complementary threat surface where the attack target is the generator rather than the retrieval result.

All prior corpus poisoning attacks share a key characteristic: injected documents are crafted to maximize retrieval relevance to the target query, pulling their embeddings toward the query direction. This outlier behavior is the explicit target of partition-based defenses. PPI departs from this by targeting the cluster *interior*, making it the first attack to simultaneously satisfy retrieval relevance and distribution conformance objectives.

RAG Defenses. Existing defenses against RAG poisoning attacks can be broadly categorized into retrieval filtering, response aggregation, and partition-based verification. Retrieval filtering approaches attempt to identify malicious documents before generation, while aggregation-based methods combine evidence from multiple retrieved documents to reduce the influence of any single adversarial source. More recently, partition-based approaches leverage clustering and semantic consistency analysis to detect suspicious retrieval results.

Among aggregation-based approaches, RobustRAG [11] provides the strongest formal robustness guarantee, applying an isolate-then-aggregate strategy. It proves that if fewer than half of the retrieved passages are adversarial, the majority-vote answer remains correct. Crucially, RobustRAG’s guarantee is agnostic to how documents entered the corpus.

Several recent defenses exploit structural properties of the retrieval space. TrustRAG [12] uses k -means clustering at retrieval time to separate malicious from clean clusters via cosine similarity and ROUGE overlap. RAGDefender [13] combines agglomerative clustering and TF-IDF with concentration-based grouping to isolate potentially poisoned passages. SeCon-RAG [14] further incorporates entity-intent-relation extraction to improve semantic consistency verification. In contrast, ASTUTE RAG [4] focuses on conflict resolution by eliciting the model’s internal knowledge and consolidating it with retrieved evidence to identify contradictory or misleading contexts. ReliabilityRAG [10] exploits document rank as a reliability signal and provides a partial formal guarantee via maximum independent set reasoning.

Despite their differences, existing defenses share a common assumption that poisoned documents remain distinguishable from benign corpus content. Through clustering, anomaly detection, consistency verification, or ranking analysis, these methods rely on adversarial passages exhibiting abnormal retrieval behavior or detectable semantic inconsistencies. However, recent attacks show that poisoned documents can be crafted to align closely with legitimate semantic partitions, reducing the effectiveness of partition-based defenses. To address this limitation, we propose Partition-Aware Filtering (PAF), which combines corpus-level drift verification with retrieval-time consensus validation to improve robustness against partition-conforming poisoning

attacks.

3. Problem Formulation

3.1. Retrieval-Augmented Generation

Let $\mathcal{D} = \{d_1, d_2, \dots, d_N\}$ denote a document corpus over which a RAG system operates. A RAG system comprises a retriever $\mathcal{R}$ and a generator $\mathcal{G}$. The retriever $\mathcal{R}$ maps a query q to a ranked set of k documents via an embedding function $\phi : \mathcal{X} \rightarrow \mathbb{R}^m$, selecting the top- k passages by cosine similarity:

$$\mathcal{R}(q, \mathcal{D}) = \text{top-}k_{d \in \mathcal{D}} \cos(\phi(q), \phi(d)) \quad (1)$$

The generator $\mathcal{G}$ then produces a response conditioned on the query and retrieved context:

$$y = \mathcal{G}(q, \mathcal{R}(q, \mathcal{D})) \quad (2)$$

We assume $\mathcal{D}$ is a *dynamic* corpus: documents can be inserted or removed over time, reflecting real-world deployments where corpora are continuously updated.

3.2. Partition-based defenses.

A partition-based defense applies a clustering algorithm $\mathcal{C}$ to a document set $\mathcal{S}$, producing partitions $\mathcal{P} = \{P_1, P_2, \dots, P_K\}$ such that $\bigcup_i P_i = \mathcal{S}$ and $P_i \cap P_j = \emptyset$ for $i \neq j$. Here, $\mathcal{S}$ is either the set $\mathcal{R}_q$ retrieved for a query q or the full corpus $\mathcal{D}$. Each cluster P_i is associated with a centroid:

$$\mu_i = \frac{1}{|P_i|} \sum_{d \in P_i} \phi(d). \quad (3)$$

These defenses assume that poisoned documents crafted for the same target query share similar phrasing and are therefore more similar to one another than to benign documents. As a result, they form an anomalous cluster, or appear as outliers, in a region separable from the benign clusters. Suspicious documents are therefore identified by excess similarity, scoring each document by its similarity to its own centroid:

$$\text{score}(d) = \cos(\phi(d), \mu_{c(d)}), \quad \text{filter if } \text{score}(d) > \tau, \quad (4)$$

where $c(d)$ denotes the index of the cluster containing d . Individual defenses instantiate τ differently: as a fixed similarity threshold, or implicitly, through a data-dependent cutoff derived from set-level similarity statistics or an estimated number of adversarial documents. Defenses without centroids use the analogous pairwise score, the mean cosine similarity of d to the other documents in $\mathcal{S}$. In other words, a document is removed for being too similar to its neighbors.

TrustRAG [12], RAGDefender [13], and SeConRAG [14] each implement variants or a close analogue of this clustering-based filtering paradigm, assuming adversarially crafted documents will manifest as an anomalous cluster in embedding space, separable from the benign corpus distribution. This shared assumption constitutes the structural vulnerability we exploit.

TABLE 1: Capability comparison of corpus poisoning attacks

Capability	PoisonedRAG [9]	PIA [21]	PPI (Ours)
Query interface access	X	X	✓
Public encoder (surrogate)	✓	X	✓
Corpus write access	✓	✓	✓
Cluster structure knowledge	X	X	Inferred via probing
Internal system state	X	X	X
Gradient access	X	X	X

3.3. Threat Model

We define the threat model in terms of the attacker’s objectives, background knowledge, and capabilities. We summarize the proposed threat model and provide a comparison with existing poisoning attacks in Table 1. Cluster structure is listed as inferred rather than assumed. It is not an input to PPI but a quantity the attack reconstructs from query access (Section 4.1).

3.3.1. Attacker’s Objective. The adversary targets a set of T queries $Q^* = \{q_1^*, \dots, q_T^*\}$ and a corresponding set of attacker-specified answers $\mathcal{A}^* = \{a_1^*, \dots, a_T^*\}$, where each a_t^* is factually incorrect or otherwise harmful. The attacker’s goal is to cause the RAG system to produce outputs semantically consistent with a_t^* for each q_t^* , while passing undetected through the deployed defenses. Formally, the attacker seeks to inject a set of documents $\mathcal{D}^{\text{adv}}$ into $\mathcal{D}$ such that for each target query q_t^* :

$$\mathcal{G}(q_t^*, \mathcal{R}(q_t^*, \mathcal{D}')) \models a_t^* \wedge \text{anom}(d) = 0 \quad \forall d \in \mathcal{D}^{\text{adv}}, \quad (5)$$

where $\mathcal{D}' = \mathcal{D} \cup \mathcal{D}^{\text{adv}}$ denotes the poisoned corpus, and $\models$ denotes semantic entailment. The two conditions capture the dual objective of the attack: *effectiveness* (steering the generator toward a_t^*) and *evasion* (bypassing the partition-based anomaly filter). Unlike prior poisoning attacks that satisfy only the first condition, PPI is specifically designed to satisfy both simultaneously.

Attack success. We define a per-query attack as successful if the model output y semantically entails the target answer a_t^* , evaluated using exact match and NLI-based entailment scoring (Section 6). The overall *Attack Success Rate* (ASR) is the fraction of target queries for which this condition holds.

3.3.2. Attacker’s Capabilities. PPI is a black-box attack under the standard adversarial machine learning definition: the adversary observes only retrieval outputs and never accesses internal system parameters, gradients, cluster assignments, or any other defended system state. We characterize the adversary by five capability dimensions.

Query interface access. Query access is the default interaction mode of any deployed RAG (search assistants, enterprise Q&A), and available to ordinary users. The adversary can issue a bounded number of probe queries to the retrieval interface and observe the top- k retrieved documents for each query. As in standard RAG deployments, retrieval results expose document identifiers together with the retrieved document text (or passage content). However,

the adversary receives no embedding vectors, cluster assignments, centroids, anomaly scores, filtering decisions, or other internal system state.

Corpus write access. The adversary can inject a bounded number of documents $\mathcal{D}^{\text{adv}} = \{d_1^{\text{adv}}, \dots, d_M^{\text{adv}}\}$ into the corpus after the initial clustering phase has been completed. We denote the post-injection corpus as $\mathcal{D}' = \mathcal{D}_0 \cup \mathcal{D}^{\text{adv}}$. This is practical for an adversary, especially when the external corpus allows content from unverified sources, such as online platforms [22] and email systems [23].

Public encoder access. The adversary has access to the public retriever encoder φ (e.g., Contriever) and can use it to embed a bounded number of texts of their own authoring. This is the standard assumption in the corpus poisoning literature [9] and is explicitly not white-box access to the defended system. Importantly, encoder access does not provide access to the defender’s corpus or defense state. The adversary can encode arbitrary text using the public encoder but does not possess a complete copy of the indexed corpus ($\mathcal{D}_0$) or observe cluster assignments, centroids, clustering parameters, anomaly scores, filtering thresholds, retrieval logs, or other internal defense information. Thus, encoder access alone does not reveal the partition structure used by partition-based defenses.

No privileged access. The adversary has no access to the embedding model weights, the clustering parameters (number of clusters K , centroids $\{\mu_i\}$, or assignment threshold τ), or any internal system state beyond what is observable through the query interface.

3.3.3. Defense Objective. A robust defense against PPI-style attacks must substantially reduce the ASR of PPI-style attacks, preventing the adversary from reliably steering model outputs toward attacker-specified answers. In addition, the defense must remain effective under adaptive threat models, where the adversary can craft poisoned documents specifically designed to evade detection or mitigation. This rules out defenses whose security relies on the adversary not knowing the defense strategy (*security through obscurity*).

4. Post-Partition Injection (PPI)

PPI proceeds in four stages illustrated in Figure 1. We describe each phase formally and then analyze the attack’s evasion guarantee against partition-based defenses.

4.1. Phase 1: Probing and Reconnaissance

The first stage of PPI performs reconnaissance to infer the latent partition structure maintained by the defense. Although the retriever encoder is publicly available, encoder access alone is insufficient to mount PPI. The encoder (ϕ) maps text to embeddings, but it does not provide access to the defender’s indexed corpus ($\mathcal{D}_0$), cluster assignments, or cluster centroids. Since partition-based defenses compute clusters over documents drawn from the defender’s private

corpus, either per-query retrieved sets or the full index, these quantities cannot be recovered from encoder access alone.

To obtain information about the corpus, the adversary probes the retrieval interface using a set of exploratory queries $[Q_p = q_1, q_2, \dots, q_{T_{\text{probe}}}]$. For each query (q_t), the adversary records the returned top-(k) retrieved documents $[R(q_t) = d_1, \dots, d_k]$. Through the RAG interface, probing gradually exposes a subset of corpus documents together with their textual content. These retrieved documents constitute the attacker’s observable view of the corpus.

From these observations, the adversary constructs a *co-retrieval matrix* $M \in \mathbb{R}^{N \times N}$, where N denotes the number of unique documents observed during probing. Each entry ($M_{i,j}$) records the frequency with which documents d_i and d_j are retrieved together in the same result set:

$$M_{ij} = \sum_{t=1}^{T_{\text{probe}}} \mathbf{1}[d_i \in R(q_t)] \mathbf{1}[d_j \in R(q_t)]. \quad (6)$$

Intuitively, documents that belong to the same cluster share similar embeddings and therefore tend to co-appear under semantically related queries; documents from different clusters rarely co-appear. As a result, the block structure of M approximates the latent partition $\mathcal{P}$.

Two capabilities are jointly required at this stage. First, probing surfaces document texts from the retrieval interface, providing a corpus sample that would otherwise be unavailable to the attacker. Second, the public encoder enables local embedding computation over these retrieved documents. Neither capability alone is sufficient: encoder access without corpus observations cannot estimate cluster geometry, while corpus observations without the encoder cannot produce the embedding-aligned poison documents used in later stages of the attack.

The co-retrieval graph produced during reconnaissance serves as an approximation of the hidden partition structure maintained by the defense. In Phase 2, PPI applies spectral clustering to this graph to infer latent cluster memberships and estimate cluster geometry for subsequent poison generation.

Query selection. To maximise the informativeness of M , probe queries are sampled to cover a broad range of semantic topics rather than concentrated around the attack targets. In our experiments, $T_{\text{probe}} = 100$ queries were drawn from a held-out pool drawn from the same distribution as the target benchmark, though we show in Table 5 that attack success is robust to reductions down to $T_{\text{probe}} = 50$.

4.2. Phase 2: Cluster Inference

The adversary treats M as a similarity graph and applies spectral clustering to recover an approximate partition $\hat{\mathcal{P}} = \{\hat{C}_1, \dots, \hat{C}_m\}$. Let $L = I - D^{-1/2} M D^{-1/2}$ denote the normalised graph Laplacian, where $D_{ii} = \sum_j M_{ij}$. The adversary computes the m smallest eigenvectors of L and applies k -means to the resulting spectral embedding to obtain cluster assignments.

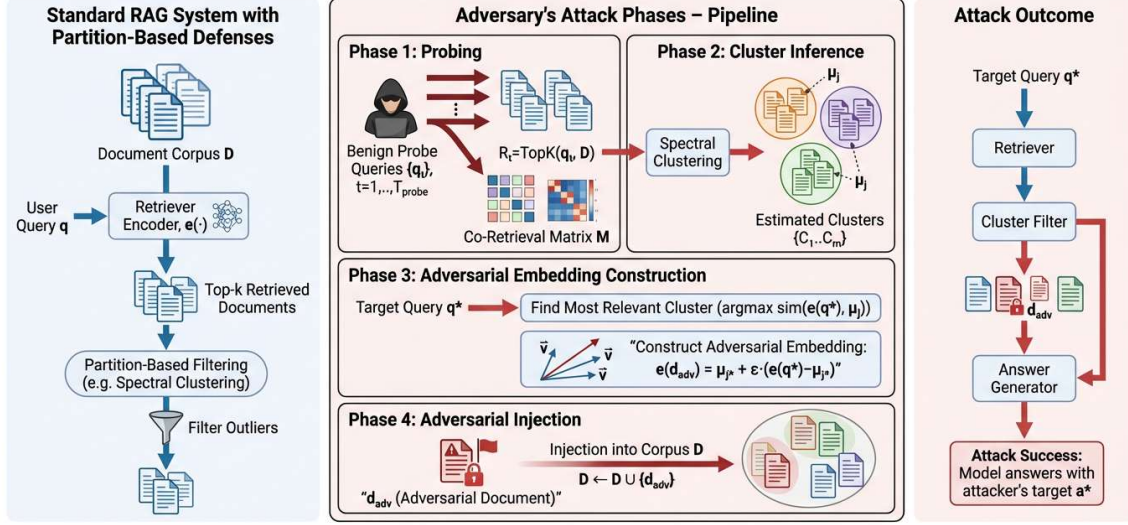


Figure 1: Overview of the PPI attack. PPI constructs a co-retrieval matrix from probe queries (phase-1), infers retrieval clusters via spectral analysis (phase-2), and generates cluster-aligned poisoned documents (phase-3). Injection occurs after partitioning to evade detection and influence generation (phase 4).

Estimating cluster count. Since K is not directly observable, the adversary estimates m using the *eigengap heuristic*: the number of clusters is set to the index at which the eigenvalue sequence of L exhibits the largest relative gap [24]. We find empirically that this heuristic recovers a partition whose cluster boundaries closely approximate those of $\mathcal{P}$ across all three benchmarks.

Centroid estimation. For each inferred cluster $\hat{C}_j$, the adversary estimates the cluster centroid by computing the mean embedding locally using the publicly accessible encoder ϕ :

$$\hat{\mu}_j = \frac{1}{|\hat{C}_j|} \sum_{d \in \hat{C}_j} \phi(d). \quad (7)$$

4.3. Phase 3: Targeted Construction

Given the inferred partition $\hat{\mathcal{P}}$, a target query $q^* \in Q^*$, and a target answer $a^* \in \mathcal{A}^*$, the adversary constructs a poisoned document d^{adv} whose embedding (i) lies within the cluster most likely to be retrieved for q^* and (ii) steers generation toward a^* .

Target cluster selection. The adversary selects the target cluster j^* as the one whose estimated centroid is most similar to the query embedding:

$$j^* = \arg \max_{j \in [m]} \cos(\phi(q^*), \hat{\mu}_j) \quad (8)$$

Target embedding construction. A naive approach would set the target embedding equal to $\phi(q^*)$, maximising retrieval relevance. However, several documents placed along the query direction would be near-duplicates of one another and form an anomalous cluster flagged by partition-based defenses (Equation 4). We therefore construct a *convex*

interpolation between the estimated cluster centroid $\hat{\mu}_{j^*}$ and the query-aligned direction:

$$\hat{z} = \hat{\mu}_{j^*} + \varepsilon \cdot (\phi(q^*) - \hat{\mu}_{j^*}), \quad \varepsilon \in (0, 1). \quad (9)$$

Here ε controls the trade-off between evasion and retrieval effectiveness: small ε places the target embedding close to the centroid (high conformity, lower relevance), while $\varepsilon \rightarrow 1$ recovers the query direction (high relevance, increased detection risk). We select ε by a simple grid search over $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, evaluated on a small held-out set of non-target queries (see Section 6).

Adversarial text generation. Given the target embedding $\hat{z}$, the adversary generates a text document d^{adv} that (i) contains the target answer a^* and (ii) satisfies $\phi(d^{\text{adv}}) \approx \hat{z}$. We achieve this via a two-stage procedure. First, we generate a candidate document using a prompted LLM conditioned on a^* and the target query q^* :

$$d_0^{\text{adv}} \sim \mathcal{G}(\text{Write a passage that answers } q^* \text{ with } a^*). \quad (10)$$

Second, we fine-tune the token sequence of d_0^{adv} using a greedy embedding-space optimisation, iteratively substituting tokens to minimise $\|\phi(d^{\text{adv}}) - \hat{z}\|_2$ subject to the constraint that a^* is preserved verbatim. This follows the general approach of HotFlip-style token substitution [25], applied in the document embedding space rather than the classification logit space.

4.4. Phase 4: Attack Execution

The adversary injects d^{adv} into the live corpus $\mathcal{D}$ at time $t > t_0$, where t_0 denotes when the benign reference partition is established. Whether computed from D_0 or per query from R_q , this partition is dominated by benign documents present before injection. The filter therefore scores d^{adv} against the

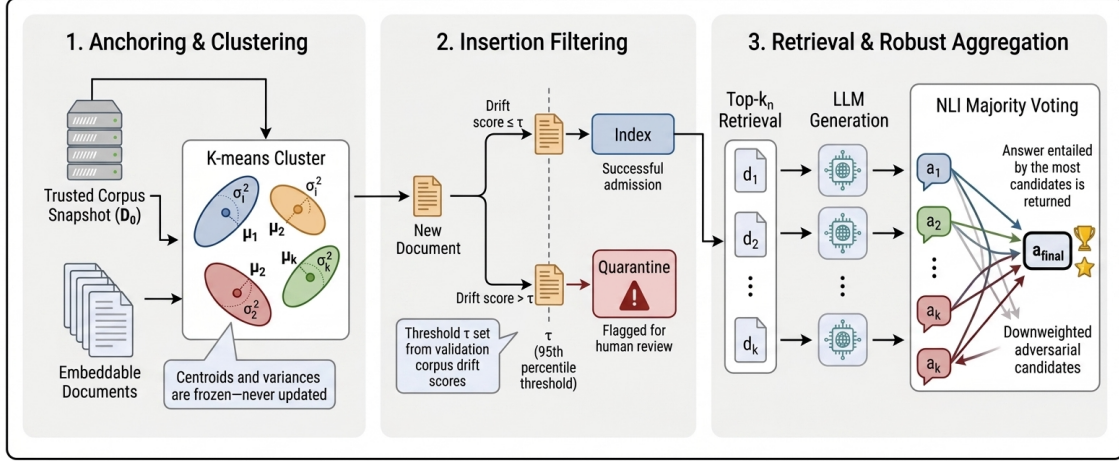


Figure 2: PAF-RAG pipeline. (1) K-means anchors D_0 by clustering and stores centroids μ_j . (2) Documents are assigned a drift score based on distance to the nearest centroid and are quarantined if it exceeds threshold τ . (3) A KV-cached LLM generates per-document candidates, which are aggregated via NLI-based majority voting.

centroid of the benign-dominated cluster it joins. For an appropriate ε , $\phi(d^{\text{adv}}) \approx \hat{z}$ lies inside the benign region of that cluster, so d^{adv} is absorbed rather than isolated, and its score matches that of typical benign members, so the filter accepts d^{adv} as a benign document:

$$\text{score}(d^{\text{adv}}) \approx \text{score}(d) \leq \tau, \quad d \in P_{c(d^{\text{adv}})}. \quad (11)$$

Simultaneously, the interpolated embedding makes it likely that d^{adv} ranks highly in the top- k retrieval results for q^* , in which case $\mathcal{G}(q^*, \mathcal{R}(q^*, \mathcal{D} \cup \{d^{\text{adv}}\}))$ can produce an output entailing a^* .

Multi-document injection. For injection budgets $M > 1$, the adversary generates M poisoned documents using distinct values of ε drawn from a uniform grid, encouraging diversity in both embedding position and surface text. This increases the probability that at least one injected document appears in the top- k result set for q^* while making the injected documents less likely to form an anomalous cluster.

5. Partition-Aware Filtering (PAF)

The vulnerability exposed by PPI stems from the outlier assumption shared by existing partition-based defenses. Because these defenses flag only anomalously compact clusters and recompute their partition after injection, poisoned documents absorbed into a benign cluster are accepted (Section 4.4). PAF addresses this in three ways. First, it anchors the reference geometry to a trusted snapshot, so injected content cannot shift the detection baseline. Second, it screens each new document by its drift from this frozen geometry, forcing a PPI adversary to trade retrieval relevance for evasion (Eq. 15). Third, it bounds the influence of any document that still passes through document-isolated generation and NLI-based aggregation. The defense comprises three layers, illustrated in Figure 2: (1) corpus anchoring and clustering, (2) drift-based

insertion filtering, and (3) document-isolated generation with NLI-based aggregation.

5.1. Corpus Anchoring and Clustering

At deployment time t_0 , PAF computes a trusted corpus snapshot $\mathcal{D}_0$. This snapshot serves as a tamper-evident reference: any subsequent modification to $\mathcal{D}_0$ is detectable by recomputation.

Standard k -means clustering is then applied to the embeddings $\{\phi(d)\}_{d \in \mathcal{D}_0}$ to produce a partition $\mathcal{P} = \{P_1, \dots, P_K\}$ with centroids $\{\mu_j\}_{j=1}^K$. Crucially, the centroid set $\{\mu_j\}$ and the empirical intra-cluster distance distributions $\{\sigma_j^2\}$ are *frozen* against the anchor snapshot and are never recomputed as new documents arrive. This anchoring strategy is the key departure from existing defenses, which re-cluster per query over the retrieved set [12], [13] or over the contaminated corpus [14], without anchoring to a trusted snapshot. By tying the reference geometry to $\mathcal{D}_0$, PAF ensures that a post-partition adversary cannot shift the anomaly detection baseline by injecting many weakly-adversarial documents before the target attack.

5.2. Document Insertion with Drift Detection

When a new document d_{new} arrives for insertion, PAF computes its *drift score* as the minimum distance from its embedding to the frozen centroid set:

$$\text{drift}(d_{\text{new}}) = \min_{j \in [K]} \|\phi(d_{\text{new}}) - \mu_j\|_2. \quad (12)$$

The document is admitted to the live corpus if and only if its drift score falls below an admission threshold τ :

$$\text{admit}(d_{\text{new}}) = \begin{cases} \text{PASS} & \text{if } \text{drift}(d_{\text{new}}) \leq \tau \\ \text{QUARANTINE} & \text{otherwise.} \end{cases} \quad (13)$$

Algorithm 1 Partition-Aware Filtering (PAF)

Offline Phase (at deployment time t_0)**Require:** Corpus D_0 , validation set D_{val} ($|D_{\text{val}}| = 1000$)

- 1: Run KMeans on $\phi(D_0)$
- 2: Obtain partitions $P = \{P_1, \dots, P_K\}$ and centroids $\{\mu_j\}_{j=1}^K$ (frozen)
- 3: Compute cluster variances $\{\sigma_j^2\}_{j=1}^K$ (frozen)
- 4: $\tau \leftarrow \text{Percentile}_{95}(\{\min_j \|\phi(d) - \mu_j\|_2 : d \in D_{\text{val}}\})$

Ingestion Phase ($t > t_0$)

- 5: **for all** new document d_{new} **do**
 - 6: $\text{drift}(d_{\text{new}}) \leftarrow \min_j \|\phi(d_{\text{new}}) - \mu_j\|_2$
 - 7: **if** $\text{drift}(d_{\text{new}}) \leq \tau$ **then**
 - 8: Admit d_{new} to index
 - 9: **else**
 - 10: Quarantine d_{new}
 - 11: **end if**
 - 12: **end for**
 - 13: **Retrieval Phase (for query q)**
 - 14: $F \leftarrow \text{top-}k \text{ retrieval results } (|F| = k_f \leq 10)$
 - 15: **for all** $d_i \in F$ **do**
 - 16: $a_i \leftarrow G(q, d_i)$ $\triangleright$ isolated generation
 - 17: **end for**
 - 18: $v(a_i) \leftarrow |\{j \neq i : \text{NLI}(a_j \rightarrow a_i) = \text{ENTAIL}\}|$
 - 19: $a_{\text{final}} \leftarrow \arg \max_{a_i} v(a_i)$
 - 20: **return** a_{final}
-

Quarantined documents are flagged for human review and excluded from retrieval pending verification.

Threshold calibration. The threshold τ is set to the 95th percentile of drift scores computed over a held-out set of known-clean documents drawn from the same distribution as $\mathcal{D}_0$:

$$\tau = \text{Percentile}_{95}(\{\text{drift}(d) : d \in \mathcal{D}_{\text{val}}\}). \quad (14)$$

This calibration ensures that legitimate documents, including those written after t_0 , are admitted at a high true-positive rate while documents whose embeddings deviate from the anchor geometry are flagged.

Relationship to PPI. An adversary executing PPI with interpolation parameter ε (Equation 9) produces a document whose drift score satisfies:

$$\text{drift}(d^{\text{adv}}) \approx \varepsilon \cdot \|\phi(q^*) - \mu_{j^*}\|_2. \quad (15)$$

When $\|\phi(q^*) - \mu_{j^*}\|_2$ exceeds τ , the adversary must choose small ε to evade PAF, which by Equation 9 simultaneously reduces the document’s retrieval relevance to the target query. This formalises the evasion–effectiveness trade-off inherent in PPI under PAF: reducing drift to evade detection constrains the adversary’s ability to significantly shift retrieval relevance toward the target query.

5.3. Document-Isolated Generation and NLI Aggregation

Even with drift filtering, a small number of adversarial documents may pass the admission threshold, particularly at

small ε or under high-quality cluster mimicry. To bound the impact of such documents, PAF adopts a *document-isolated generation* strategy inspired by RobustRAG [11], combined with a single-pass NLI-based majority vote.

Document-isolated generation. Let $\mathcal{F} = \{d_1, \dots, d_{k_f}\} \subseteq \mathcal{D}'$ denote the set of up to k_f documents that pass the drift filter and are retrieved for query q^* . Rather than concatenating all retrieved documents into a single context, PAF generates one candidate answer per document in isolation:

$$a_i = \mathcal{G}(q^*, d_i), \quad i = 1, \dots, k_f. \quad (16)$$

Each generation call uses a key-value cached single-pass LLM with document-isolated attention, ensuring that adversarial content in one document cannot influence the generation conditioned on another. This isolation property enables the robustness certificate in Theorem 5.1.

NLI majority vote. Given the candidate answer set $\{a_1, \dots, a_{k_f}\}$, PAF aggregates using pairwise Natural Language Inference (NLI) entailment scoring via a lightweight cross-encoder (DeBERTa-v3-small). The aggregate answer a^* is selected as the candidate that is *entailed* by the largest number of other candidates:

$$a^* = \arg \max_{a_i} |\{j \neq i : \text{NLI}(a_j \rightarrow a_i) = \text{ENTAIL}\}|. \quad (17)$$

Intuitively, the correct answer is supported by the majority of clean retrieved documents, while adversarial documents—which by design assert a single attacker-specified answer a_{adv}^* —form a minority coalition. The NLI vote down-weights outlier candidates without requiring an explicit label for which documents are adversarial.

5.4. Robustness Guarantee

We state a formal robustness guarantee for PAF under partial adversarial control. We first make explicit the condition under which the guarantee holds, then validate it empirically in Section 5.5.

Assumption 1 (Bounded Adversarial Infiltration). *Among the k_f documents passed to the generation stage after drift filtering, the number of adversarial documents satisfies $k_{\text{adv}} < \frac{k_f}{2}$.*

Assumption 1 requires that adversarial content remains a strict minority of the filtered retrieval set after drift detection. Section 5.5 provides direct empirical evidence that this condition holds for injection budgets $M \leq 3$ across all benchmarks, and holds with high probability (93%) for $M \leq 10$ on MS-MARCO.

Theorem 5.1 (Conditional PAF Robustness Certificate). *Let k_f denote the number of filtered documents passed to generation, and let $k_{\text{adv}} \leq k_f$ denote the number of adversarial documents among them. Let ε_{NLI} denote the per-comparison error rate of the NLI cross-encoder. Under Assumption 1, if $\varepsilon_{\text{NLI}} < \frac{k_f - 2k_{\text{adv}}}{2k_f}$, then the NLI majority vote returns a correct answer with probability at least $1 - k_f \varepsilon_{\text{NLI}}$.*

TABLE 2: Post-filter adversarial fraction k_{adv}/k_f (mean $\pm$ std, 100 queries) and Assumption 5.1 violation rate (%). $k_f = 10$. * denotes violation rate $> 5\%$.

M	MS-MARCO		NQ		HotpotQA	
	Adv. frac. ($\mu \pm \sigma$)	Viol. (%)	Adv. frac. ($\mu \pm \sigma$)	Viol. (%)	Adv. frac. ($\mu \pm \sigma$)	Viol. (%)
1	0.022 $\pm$ 0.041	0.0	0.067 $\pm$ 0.047	0.0	0.083 $\pm$ 0.038	0.0
2	0.038 $\pm$ 0.075	0.0	0.127 $\pm$ 0.093	0.0	0.166 $\pm$ 0.075	0.0
3	0.050 $\pm$ 0.102	0.0	0.178 $\pm$ 0.136	0.0	0.245 $\pm$ 0.113	0.0
5	0.068 $\pm$ 0.150	4.0*	0.263 $\pm$ 0.220	19.0*	0.379 $\pm$ 0.188	23.0*
7	0.079 $\pm$ 0.184	7.0*	0.320 $\pm$ 0.287	28.0*	0.487 $\pm$ 0.265	36.0*
10	0.088 $\pm$ 0.217	8.0*	0.351 $\pm$ 0.334	35.0*	0.564 $\pm$ 0.339	57.0*

Proof sketch. After drift filtering, let $k_c = k_f - k_{\text{adv}}$ denote the number of clean documents. Under Assumption 1, clean documents form a strict majority ($k_c > k_{\text{adv}}$). Each document independently generates a candidate answer, and the NLI aggregation selects the answer with the largest number of pairwise entailment votes. Under an ideal NLI oracle, all clean candidates are mutually entailing and receive $k_c - 1$ votes, while adversarial candidates receive at most $k_{\text{adv}} - 1$ votes; clean candidates therefore strictly dominate.

Let $\Delta = k_c - k_{\text{adv}}$ denote the vote margin. Under stochastic NLI errors with rate ε_{NLI} , each candidate’s vote score can shift by at most $k_f \varepsilon_{\text{NLI}}$. The margin is preserved whenever $2k_f \varepsilon_{\text{NLI}} < \Delta$, i.e., $\varepsilon_{\text{NLI}} < (k_f - 2k_{\text{adv}})/(2k_f)$. Applying a union bound over the k_f candidates yields the stated success probability $1 - k_f \varepsilon_{\text{NLI}}$.

The full formal proof is given in Appendix B.1. $\square$

Practical implications. The certificate requires: (i) adversarial documents to remain a minority after drift filtering (Assumption 1), and (ii) the NLI error rate to be sufficiently small relative to the vote margin. Condition (ii) is satisfied by DeBERTa-v3-small [26]. Condition (i) depends on the effectiveness of the drift-filtering stage and is validated empirically in Section 5.5. Appendix C analyzes the scope of the guarantee: the deployment conditions under which (i) holds (C.1), and three failure modes—large injection budgets (C.2), sparse retrieval density (C.3), and high-quality embedding mimicry (C.4).

5.5. Validation of Assumption

Assumption 1 requires that adversarial documents remain a strict minority after semantic drift filtering, i.e., $k_{\text{adv}} < k_f/2$. Since PPI is designed to generate cluster-aligned adversarial documents that evade semantic drift detection, this assumption is not guaranteed by construction and must therefore be validated empirically.

We evaluate adversarial injection budgets $M \in \{1, 2, 3, 5, 7, 10\}$ on MS-MARCO, NQ, and HotpotQA. For each $(M, \text{benchmark})$ pair, 100 target queries are executed with M injected PPI documents. After applying the Layer-1 semantic drift filter, we measure the post-filter adversarial fraction k_{adv}/k_f ($k_f = 10$) and the percentage of queries satisfying $k_{\text{adv}} \geq k_f/2$, corresponding to a violation of Assumption 5.1.

Table 2 shows that Assumption 1 holds for 100% of queries when $M \leq 3$, validating the robustness certificate in

its intended operating regime. Although some PPI successfully bypasses the semantic drift filter, adversarial documents do not dominate the retrieved top- k_f set under these injection budgets, resulting in low post-filter adversarial fractions across all three benchmarks.

For larger injection budgets ($M \geq 5$), assumption violations become increasingly frequent, reaching 8% on MS-MARCO, 35% on NQ, and 57% on HotpotQA. These results indicate that sufficiently large-scale injections can cause adversarial documents to constitute a majority of the filtered retrieval set, placing the system outside the certified regime. Importantly, this does not imply that PAF becomes ineffective. As shown in Section 6, the complete PAF pipeline continues to reduce ASR against PPI to 3.2–8.6%.

Overall, the results empirically validate Assumption 1 for $M \leq 3$ and establish the scope of the robustness certificate. Beyond this regime, PAF continues to demonstrate strong empirical robustness, even though the theoretical certificate no longer universally applies.

6. Evaluation

We organize our evaluation into two phases. The first evaluates the Post-Partition Injection (PPI) attack against both undefended and defended RAG systems. The second evaluates Partition-Aware Filtering (PAF) against adaptive poisoning attacks and assesses its impact on benign retrieval quality and downstream question-answering performance.

6.1. Experimental Setup

In this section, we describe our experimental setup. The effectiveness and robustness of PPI and PAF are examined under multiple adversarial conditions. All results are computed as the mean over 5 independent runs, with an uncertainty of approximately $\pm 1\%$.

6.1.1. Datasets. We evaluate on three datasets covering multiple QA and generation settings: MS-MARCO (RQA) [27], HotpotQA [28], and Natural Questions (NQ) [29]. These benchmarks are widely used to assess both accuracy and robustness in RAG systems, making them suitable for evaluating our approach (see Appendix A for details).

6.1.2. Evaluation Metrics. Following prior work, we adopt two primary metrics to evaluate the performance of all defense methods in Retrieval-Augmented Generation (RAG) systems: (1) **Accuracy (ACC)**, which measures the proportion of correct responses produced under standard (non-adversarial) conditions, thereby reflecting the reliability of retrieval and generation; and (2) **Attack Success Rate (ASR)**, which captures system vulnerability by measuring the fraction of incorrect outputs triggered by adversarial inputs.

6.1.3. Attack configurations. As illustrated in Table 7, we evaluate the robustness of different RAG frameworks across multiple LLMs (Mistral7b, Llama8b, and Qwen7b) under three adversarial settings: prompt injection attacks (PIA),

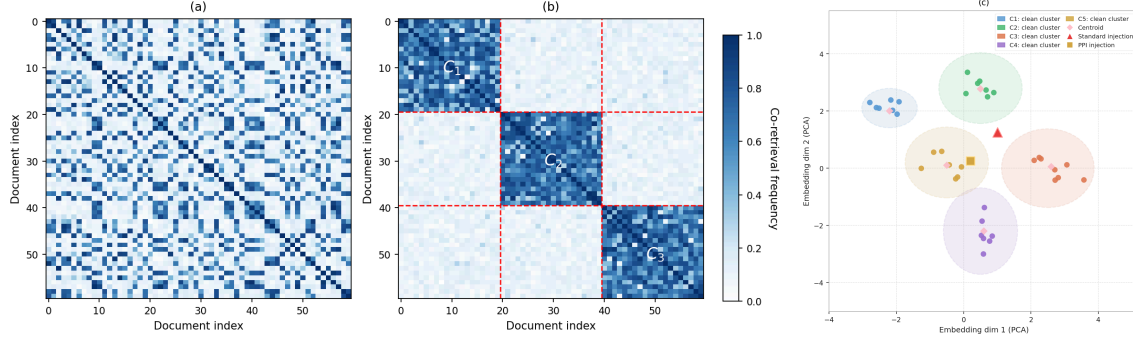


Figure 3: Co-retrieval matrix structure and embedding visualization: (a) Co-retrieval Matrix M Before, and (b) After Spectral Reordering, and (c) Embedding Space: Standard Injection vs Post-Partition Injection (PPI)

TABLE 3: Baseline vulnerability of Vanilla RAG (RQ1). ACC $\uparrow$ / ASR $\downarrow$ for PIA, PoisonedRAG, and PPI across three generator models and three datasets.

Model	Natural Questions										HotpotQA										MS-MARCO									
	PIA		PR		PPI *						PIA		PR		PPI *						PIA		PR		PPI *					
					PN-1		PN-2		PN-3						PN-1		PN-2		PN-3						PN-1		PN-2		PN-3	
	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR	ACC	ASR
Mistral7b	46.1	49.7	41.8	52.4	25.6	75.5	19.7	77.8	18.9	79.2	48.7	50.2	51.4	44.6	24.8	68.5	23.5	69.4	22.7	71.2	44.3	48.6	27.5	69.3	23.4	76.2	22.1	77.5	20.8	79.1
Llama8b	43.4	54.6	27.8	72.2	35.5	67.6	34.6	69.1	33.8	70.7	53.6	47.4	29.4	69.8	41.3	65.6	40.5	67.3	39.7	69.5	54.5	58.2	26.3	72.6	26.7	74.3	23.4	76.8	22.2	78.1
Qwen7b	53.8	40.5	57.4	38.3	45.7	50.9	43.9	55.4	38.3	60.8	68.4	27.3	66.8	25.6	33.6	64.4	32.4	65.6	30.7	67.1	61.5	36.4	51.2	49.8	25.6	70.7	24.8	72.1	23.7	73.9

corpus poisoning attacks (PoisonedRAG), and the proposed post-partition injection (PPI).

6.2. Attack Analysis (PPI)

This subsection addresses four research questions about the effectiveness and operational characteristics of Post-Partition Injection.

- **RQ-A1:** How effective is PPI compared to existing corpus poisoning attacks (PIA, PoisonedRAG) against traditional RAG systems?
- **RQ-A2:** Does PPI successfully bypass existing partition-based defenses (TrustRAG, SeConRAG etc.), and by how much compared to prior attacks?
- **RQ-A3:** How does the number of probe queries (T_{probe}) affect PPI’s ability to reconstruct the cluster partition and mount a successful attack?
- **RQ-A4:** How does the interpolation parameter ϵ affect attack effectiveness?

6.2.1. PPI Against Vanilla RAG (RQ-A1). We first evaluate PPI against an undefended, vanilla RAG pipeline to establish its baseline effectiveness relative to two well-known corpus poisoning baselines: PIA and PoisonedRAG (PR). Table 3 reports Attack Success Rate (ASR) and Clean Accuracy (ACC) across three benchmark datasets (Natural Questions (NQ), HotpotQA, and MS-MARCO) and three backbone LLMs (Mistral-7B, LLaMA-8B, Qwen-7B).

PPI is evaluated under three poisoning-count variants (PN-1, PN-2, and PN-3), corresponding to one, two, and three injected poison documents per query, respectively. Across the evaluated datasets and backbone LLMs, PPI achieves substantially higher ASR than both PIA and PR

in most configurations, in several cases already under the weakest configuration (PN-1). For instance, on Natural Questions with Mistral-7B, PIA achieves an ASR of 49.7% and PoisonedRAG achieves 52.4%, while PPI PN-1 already reaches 75.5%, scaling to 79.2% under PN-3. A similar pattern holds on HotpotQA and MS-MARCO.

Across the three models, Qwen-7B exhibits the most robust resistance to all attacks, maintaining higher ACC and lower ASR compared to the other models; however, PPI still substantially degrades its performance. This consistent behavior across models indicates that PPI is model-agnostic and does not depend on a specific architecture or vocabulary.

Overall, these results demonstrate that PPI outperforms PIA and PoisonedRAG in most settings, achieving strong effectiveness even under minimal poisoning. Its impact further increases with the number of injected documents and remains stable across diverse backbone LLMs, validating the core premise that partition-aware injection constitutes a qualitatively more powerful threat than unstructured poisoning.

6.2.2. PPI Against Partition-Based Defenses (RQ-A2).

A central motivation for PPI is the observation that existing partition-based defenses, including TrustRAG and SeConRAG, rely on the implicit assumption that adversarial documents will form an anomalous cluster separable from benign ones and can therefore be detected by their excess similarity. PPI directly challenges this assumption by constructing adversarial documents whose embeddings lie inside legitimate clusters. Figure 3(c) illustrates this in embedding space. Traditional poisoning attacks produce embeddings that are near-duplicates of one another and lie apart from the benign clusters, causing them to be flagged as anomalous (red triangle). In contrast, PPI documents fall within the distribution of the target cluster while remaining retrievable

TABLE 4: Attack Success Rate (ASR ↓) of different defense frameworks across three datasets and three backbone LLMs. PPI attack is evaluated under three poisoning variants (PN-1, PN-2, PN-3), where PN refers to the number of poison documents. Bold entries indicate the lowest ASR per column within the same model block.

Model	Defense	NQ ↓					HotpotQA ↓					MS-MARCO ↓				
		PIA	PR	PN-1	PPI PN-2	PN-3	PIA	PR	PN-1	PPI PN-2	PN-3	PIA	PR	PN-1	PPI PN-2	PN-3
Mistral7b	RobustRAG	5.6	8.2	25.7	23.6	24.8	7.3	8.1	10.6	9.8	11.2	24.5	28.4	31.3	29.7	30.5
	InstructRAG	42.5	39.4	43.6	41.8	44.5	36.8	37.1	29.5	31.2	30.7	63.5	60.2	48.6	50.4	47.8
	ASTUTE RAG	18.4	11.3	15.4	14.7	16.1	25.4	15.8	16.3	15.5	17.4	27.6	25.7	19.7	20.5	18.9
	TrustRAG	9.8	10.4	17.5	18.1	16.8	8.9	13.1	21.6	19.7	20.4	10.2	12.6	23.8	22.6	24.5
	SeconRAG	10.2	11.8	40.1	38.9	39.6	11.4	12.7	38.6	37.5	39.2	11.7	10.5	42.4	40.8	41.6
	ReliabilityRAG	25.3	53.7	61.4	62.7	65.3	17.8	33.4	57.3	60.8	60.5	22.8	24.7	61.8	63.5	64.2
Llama8b	GRADA	6.5	17.2	38.1	40.3	41.6	12.4	18.7	35.6	36.2	38.5	10.8	29.4	38.5	40.1	41.6
	RobustRAG	10.3	28.7	25.8	24.1	26.2	16.4	20.2	14.5	15.1	14.8	5.4	49.3	48.2	47.5	49.1
	InstructRAG	20.4	38.6	33.7	32.9	34.4	18.9	34.8	25.3	24.8	26.1	28.5	40.3	52.6	51.7	50.9
	ASTUTE RAG	13.4	12.8	15.6	16.2	15.9	24.8	40.5	16.7	17.1	16.3	27.4	17.5	21.7	20.1	18.7
	TrustRAG	11.6	8.2	17.7	18.5	17.1	13.2	10.8	15.9	16.2	15.4	8.3	19.6	19.3	20.8	22.3
	SeconRAG	10.8	13.4	31.5	30.7	32.1	8.5	13.1	39.6	38.4	40.2	11.5	10.8	34.7	35.2	33.8
Qwen7b	ReliabilityRAG	19.8	43.7	59.3	60.4	62.1	16.3	39.8	62.5	62.8	63.4	16.9	30.4	56.3	56.8	60.7
	GRADA	7.2	21.4	25.7	27.3	30.6	9.7	25.3	32.6	32.8	33.5	6.8	23.7	20.3	24.7	25.9
	RobustRAG	39.5	29.3	34.7	33.5	35.2	28.7	21.5	19.8	20.4	18.9	9.6	26.4	39.8	38.7	40.1
	InstructRAG	35.7	40.6	25.3	24.6	26.1	27.6	36.4	17.5	18.1	16.9	42.6	62.7	39.5	40.2	38.9
	ASTUTE RAG	5.4	7.6	13.7	14.2	13.9	12.8	12.5	13.6	14.1	13.8	15.4	23.6	14.5	15.1	14.8
	TrustRAG	7.6	12.8	23.7	22.9	24.1	6.4	8.2	19.5	18.7	20.2	11.7	8.5	20.8	19.6	21.1
	SeconRAG	12.5	14.2	45.5	44.1	46.2	11.9	11.8	41.6	40.2	42.4	11.9	13.5	40.8	39.7	41.2
	ReliabilityRAG	22.7	52.2	64.6	65.1	65.9	13.5	43.2	61.7	63.2	65.6	16.9	27.2	57.3	59.5	61.2
	GRADA	8.6	14.2	30.1	32.5	34.2	10.2	19.4	31.6	31.8	32.4	12.5	17.2	35.8	36.4	36.9

TABLE 5: PPI attack performance vs. number of probe queries (T_{probe}). NMI measures cluster reconstruction quality against ground truth assignments. Mistral7b and $T_{\text{probe}} = 100$ are the default settings used in all main experiments.

T_{probe}	Cluster NMI	HotpotQA ASR (%)	NQ ASR (%)	MS-MARCO ASR (%)
25	0.45 ± 0.03	53.1 ± 2.1	49.8 ± 1.9	58.2 ± 2.3
50	0.59 ± 0.02	57.4 ± 1.7	57.3 ± 1.8	60.9 ± 1.6
100	0.71 ± 0.02	69.8 ± 1.3	75.2 ± 1.4	74.4 ± 1.2
150	0.81 ± 0.01	70.3 ± 1.8	76.6 ± 1.0	76.7 ± 0.9
200	0.93 ± 0.01	76.1 ± 0.7	80.2 ± 0.9	79.3 ± 0.8

TABLE 6: Attacker-side impact of ε on evasion vs. effectiveness. Drift is computed against the attacker’s estimated centroids (Eq. 7), and “Below $\hat{\tau}$ ” is the attacker’s anticipated evasion rate, not PAF’s admission rate. ASR is measured on undefended RAG. Results are reported on MS-MARCO (5 independent runs) using Mistral7b; NQ and HotpotQA follow the same trend.

ε	Drift score (mean)	Below τ ?	ASR (%)	Retrieval hit rate (%)	Retrieval rank (mean)
0.1	0.18 ± 0.03	97%	35.9 ± 1.1	65.4 ± 3.2	4.2 ± 0.6
0.3	0.29 ± 0.04	94%	63.2 ± 1.2	71.8 ± 2.7	3.5 ± 0.5
0.5	0.41 ± 0.05	89%	76.6 ± 0.9	82.6 ± 2.5	2.9 ± 0.4
0.7	0.56 ± 0.06	65%	79.8 ± 0.7	88.1 ± 2.1	2.2 ± 0.4
0.9	0.74 ± 0.08	42%	81.7 ± 0.8	91.3 ± 1.8	1.9 ± 0.3

for the target query. Table 4 reports ASR under these defense frameworks across Mistral-7B, LLaMA-8B, and Qwen-7B on all three datasets.

On the partition-based defenses, the pattern is consistent. TrustRAG reduces PIA and PoisonedRAG to single digits in multiple configurations (e.g., PIA 6.4% with Qwen-7B on HotpotQA) yet fails to suppress PPI below 16.8% on NQ with Mistral-7B. Against SeConRAG, which achieves one of the strongest suppressions of PIA (10.8% on NQ with Llama-8B), PPI still reaches 31.5%, 39.6%, and 34.7% on NQ, HotpotQA, and MS-MARCO at a single-document budget.

Beyond partition-based defenses, we examine PPI against non-clustering-based defenses. ReliabilityRAG [10], which

uses retrieval rank as a reliability prior, and GRADA [30], which reranks documents using a document-similarity graph. Neither effectively suppresses PPI. ReliabilityRAG yields 56.3–65.9% ASR because PPI documents are designed to rank highly for (q^*), directly exploiting its trust signal. GRADA reduces PIA and PoisonedRAG to 6.5–29.4% ASR but still allows 20.3–41.6% PPI ASR, as PPI documents remain well connected to benign neighbours.

However, PPI’s ASR under some defenses is moderate in Table 4, consistent with its design. Its advantage is strongest against cluster-based filtering, reaching 46.2% against SeConRAG with Qwen-7B on NQ, but is smaller against cross-document defenses such as ASTUTE RAG, which limits PPI to 13.6%. This difference arises because ASTUTE RAG does

not rely on partitioning; it cross-checks retrieved content against the model’s parametric knowledge, limiting the poisoned document’s influence. Nevertheless, PPI achieves higher ASR than prior attacks for most evaluated defenses, with the largest gains against partition-based defenses.

Overall, PPI remains effective against state-of-the-art partition-based defenses, indicating that partition consistency alone cannot distinguish carefully crafted poisoned documents from legitimate content.

6.2.3. Probe Query Sensitivity (RQ-A3). Table 5 reports PPI attack success rate (ASR) and cluster reconstruction quality, measured via Normalized Mutual Information (NMI). The ASR increases monotonically with cluster reconstruction fidelity. With a moderate reconnaissance budget of $T_{probe}=100$, PPI achieves an NMI of 0.71, yielding an ASR of 68–74% and representing a strong cost–effectiveness trade-off. Increasing the budget to $T_{probe}=200$ further improves reconstruction to near-perfect accuracy (NMI = 0.93), resulting in an additional 10–12% ASR gain. Overall, these results demonstrate diminishing returns beyond moderate probing and highlight a clear trade-off between reconnaissance cost and attack efficacy.

Figure 3(a-b) illustrates the reconstruction qualitatively. Under the original document ordering, the co-retrieval matrix (M) appears unstructured, with frequent co-retrievals dispersed across the matrix. Reordering rows and columns according to the spectral cluster assignments inferred in Phase 2 reveals a clear block-diagonal structure, with dense within-cluster and sparse cross-cluster co-retrieval. This confirms that co-retrieval statistics are sufficient to recover the latent partition (see Appendix B.0.1 for details).

6.2.4. Interpolation Parameter Analysis (RQ-A4). The interpolation parameter ε controls the trajectory of adversarial embeddings, inducing a trade-off between semantic consistency and retrieval effectiveness. Lower ε values constrain embeddings near their anchor centroids, preserving semantic consistency with the benign embedding manifold but limiting alignment with target queries. Larger ε values increasingly shift embeddings toward the target query region, improving retrieval effectiveness at the cost of higher embedding drift. Table 6 reports this trade-off on MS-MARCO, with NQ and HotpotQA exhibiting similar trends.

As ε increases from 0.1 to 0.9, the drift score rises monotonically from 0.18 ± 0.03 to 0.74 ± 0.08 , indicating progressively larger deviations from the anchor distribution. This shift substantially improves attack effectiveness: ASR increases from $35.9 \pm 1.1\%$ to $81.7 \pm 0.8\%$, retrieval hit rate improves from 65.4% to 91.3%, and mean retrieval rank improves from 4.2 to 1.9. These results indicate that stronger interpolation produces embeddings that are increasingly optimized for target-query retrieval.

However, the improved retrieval alignment at larger ε values comes at the cost of reduced semantic locality. The intermediate setting $\varepsilon = 0.5$ provides the best balance between embedding stability and retrieval effectiveness, achieving $76.6 \pm 0.9\%$ ASR while maintaining moderate

drift. Overall, the results show that interpolation strength is the primary factor governing the effectiveness and geometric behavior of PPI-generated adversarial embeddings.

6.3. Defense Analysis (PAF)

This section evaluates the proposed Partition-Aware Filtering (PAF) defense across five research questions covering cross-attack robustness, component contribution, runtime cost, and resilience under constrained conditions.

- **RQ-D1:** How does PAF perform compared to existing defenses in terms of robustness (ASR) and accuracy (ACC) against PPI?
- **RQ-D2:** Does PAF’s robustness generalise to attacks it was not specifically designed for (PIA, PoisonedRAG)?
- **RQ-D3:** What runtime overhead does PAF introduce compared to baseline approaches, and does it remain practical for deployment?
- **RQ-D4:** Does PAF remain robust under embedding model variation and encoder mismatch?
- **RQ-D5:** What is the relative contribution of each PAF component, and are the components complementary or redundant?

Table 7 presents a comprehensive evaluation of all defense methods across three models (Mistral7b, Llama8b, and Qwen7b) and three datasets (MS-MARCO, NQ, and HotpotQA) under multiple attack settings (PIA, PoisonedRAG, and PPI). Across all configurations, PAF RAG achieves the lowest attack success rate (ASR) under the proposed PPI attack while maintaining comparable clean accuracy (ACC), demonstrating a favorable robustness–utility trade-off.

6.3.1. Robustness Against PPI (RQ-D1). Compared to the strongest prior defense (typically TrustRAG), PAF RAG yields substantial reductions in PPI ASR across most of the evaluated models and datasets. Under Mistral7b, ASR is reduced to 3.2% on MS-MARCO, 6.3% on NQ (vs. 17.5%), and 6.6% on HotpotQA (vs. 21.6%), corresponding to improvements of up to 10 to 15 percentage points. Similar trends hold for Llama8b, where PAF RAG lowers ASR to 5.6%, 6.4%, and 5.4% across the three datasets, outperforming TrustRAG by consistent margins. Under Qwen7b, which is overall more robust, PAF RAG further pushes ASR down to 6.3%, 8.6%, and 6.5%, again establishing the best performance. These results confirm that PAF RAG is consistently the most effective defense against post-partition injection, regardless of model scale or dataset characteristics.

6.3.2. Generalization to Other Attacks (RQ-D2). Although PAF RAG is designed for PPI, it also demonstrates strong robustness under traditional attacks (PIA and PoisonedRAG). In most cases, it achieves the lowest ASR among all defenses. Under Mistral7b, it achieves an ASR of 5.3% against PIA on MS-MARCO (vs. 10.2% for TrustRAG), and 6.5% against PoisonedRAG (vs. 12.6%). This demonstrates that the defense generalizes across attack types, as expected from its design.

TABLE 7: The result table presents the performance of various defense frameworks applied to different LLMs integrated with RAG systems against prompt injection attacks (PIA), corpus poisoning attacks (PoisonedRAG), and the proposed post-partition injection (PPI) attack. The best performance is in **bold**.

Models	Defense	MS-MARCO				NQ				HotpotQA			
		PIA ACC↑ / ASR↓	PoisonedRAG ACC↑ / ASR↓	PPI ACC↑ / ASR↓	Clean ACC↑	PIA ACC↑ / ASR↓	PoisonedRAG ACC↑ / ASR↓	PPI ACC↑ / ASR↓	Clean ACC↑	PIA ACC↑ / ASR↓	PoisonedRAG ACC↑ / ASR↓	PPI ACC↑ / ASR↓	Clean ACC↑
Mistral7b	RobustRAG	56.7/24.5	50.2/28.4	51.6/31.3	53.2	54.4/5.6	55.7/8.2	53.5/25.7	58.6	74.2/7.3	73.9/8.1	71.4/10.6	71.5
	InstructRAG	32.6/63.5	35.7/60.2	38.3/48.6	74.8	52.3/42.5	49.6/39.4	51.8/43.6	66.2	58.4/36.8	56.5/37.1	62.7/29.5	82.3
	ASTUTE RAG	60.8/27.6	59.4/25.7	63.5/19.7	75.1	63.7/18.4	68.6/11.3	65.7/15.4	73.2	71.6/25.4	78.5/15.8	83.4/16.3	85.2
	TrustRAG	75.6/10.2	74.8/12.6	68.5/23.8	79.3	65.2/9.8	68.7/10.4	66.6/17.5	70.4	74.5/8.9	76.2/13.1	64.8/21.6	77.3
	SeconRAG	67.3/11.7	66.6/10.5	46.1/42.4	69.6	62.8/10.2	64.5/11.8	52.3/40.1	65.7	72.4/11.4	73.2/12.7	54.8/38.6	78.7
	PAF RAG	76.4/5.3	75.3/6.5	70.6/3.2	80.5	65.9/5.2	69.5/8.4	68.4/6.3	72.6	76.6/6.1	78.9/9.5	77.5/6.6	84.2
Llama8b	RobustRAG	54.3/5.4	41.6/49.3	50.7/48.2	55.1	45.2/10.3	50.4/28.7	60.5/25.8	60.6	68.3/16.4	66.7/20.2	72.8/14.5	73.2
	InstructRAG	63.4/28.5	55.8/40.3	45.2/52.6	84.6	54.7/20.4	57.2/38.6	62.5/33.7	69.4	58.6/18.9	62.3/34.8	71.7/25.3	88.2
	ASTUTE RAG	52.7/27.4	64.8/17.5	66.6/21.7	64.3	71.6/13.4	76.3/12.8	80.5/15.6	74.7	72.5/24.8	55.9/40.5	84.3/16.7	84.6
	TrustRAG	74.6/8.3	65.4/19.6	69.7/19.3	75.2	62.5/11.6	73.6/8.2	71.8/17.7	81.4	75.6/13.2	74.2/10.8	71.7/15.9	86.3
	SeconRAG	71.7/11.5	76.1/10.8	58.5/34.7	77.3	55.2/12.8	58.7/16.4	56.6/31.5	69.7	82.4/8.5	83.6/13.1	54.7/39.6	81.7
	PAF RAG	76.5/5.2	67.3/15.6	74.4/5.6	78.6	72.7/10.5	75.4/7.2	77.3/6.4	80.5	82.4/5.6	80.6/8.5	83.8/5.4	86.5
Qwen7b	RobustRAG	59.2/9.6	52.7/26.4	42.3/39.8	55.3	32.6/39.5	38.4/29.3	36.5/34.7	44.5	49.6/28.7	51.2/21.5	49.7/19.8	57.4
	InstructRAG	57.8/42.6	54.6/62.7	56.3/39.5	85.6	62.4/35.7	52.9/40.6	66.7/25.3	80.8	70.3/27.6	58.8/36.4	72.6/17.5	80.6
	ASTUTE RAG	75.6/15.4	77.2/23.6	79.8/14.5	79.5	80.4/5.4	81.7/7.6	76.9/13.7	82.6	85.5/12.8	78.6/12.5	83.4/13.6	86.7
	TrustRAG	82.3/11.7	85.6/8.5	78.4/20.8	85.8	84.5/7.6	82.3/12.8	69.6/23.7	83.5	90.4/6.4	87.6/8.2	72.9/19.5	88.6
	SeconRAG	71.3/11.9	72.1/13.5	49.7/40.8	75.3	67.2/12.5	65.9/14.2	52.7/45.5	74.4	74.7/11.9	77.4/11.8	55/41.6	81.7
	PAF RAG	84.6/11.5	85.4/4.6	80.7/6.3	86.4	80.7/9.6	83.5/7.4	77.8/8.6	85.7	85.6/5.3	87.8/6.7	79.6/6.5	88.5

TABLE 8: Comparison of PAF-RAG performance under different embedding models across varying poisoning settings on three datasets. Higher accuracy (↑) and lower ASR (↓) are better.

Model	Setting	HotpotQA			NQ			MS-MARCO		
		PN-1 ACC/ASR	PN-2 ACC/ASR	PN-3 ACC/ASR	PN-1 ACC/ASR	PN-2 ACC/ASR	PN-3 ACC/ASR	PN-1 ACC/ASR	PN-2 ACC/ASR	PN-3 ACC/ASR
Mistral7b	MiniLM	77.5 / 6.6	70.1 / 6.9	65.3 / 8.2	68.4 / 6.3	62.0 / 7.1	56.1 / 9.7	70.8 / 3.2	64.2 / 4.7	60.5 / 5.6
Mistral7b	SimCSE	75.3 / 7.4	67.4 / 7.6	66.8 / 8.1	65.9 / 6.7	64.5 / 7.4	58.3 / 9.7	72.6 / 3.7	64.7 / 5.4	62.3 / 6.4
Mistral7b	BGE	78.3 / 6.7	70.1 / 6.4	66.7 / 7.3	68.4 / 5.7	63.6 / 6.5	56.4 / 8.6	70.9 / 2.8	64.8 / 4.6	61.2 / 5.9
Mistral7b	BERT	75.9 / 7.3	69.1 / 6.5	65.7 / 9.6	67.6 / 7.5	61.7 / 7.9	56.0 / 10.6	69.4 / 4.6	64.1 / 5.4	59.8 / 6.9

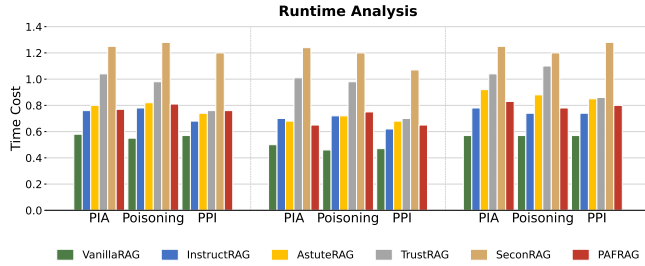


Figure 4: The average runtime per batch (in minutes) on three datasets under different adversarial settings. From left to right are MS-MARCO, HotpotQA, NQ

Clean Accuracy Trade-off. A key advantage of PAF RAG is that its robustness gains do not come at the cost of utility. PAF-RAG’s clean accuracy remains close to that of the best competing defense and is the highest in several settings. For instance, under Mistral7b, it achieves 80.5% on MS-MARCO (vs. 79.3% for TrustRAG) and 84.2% on HotpotQA (vs. 77.3%). Similar improvements are observed for Llama8b and Qwen7b, where PAF RAG consistently ranks among the top-performing methods in terms of clean accuracy.

6.3.3. Runtime Analysis (RQ-D3). We compare the runtime overhead of PAF-RAG against five representative RAG baselines, VanillaRAG, InstructRAG, AstuteRAG, TrustRAG, and SeconRAG, across three benchmark datasets (MS-MARCO, NQ, and HotpotQA) under PIA, Poisoning, and PPI attack

settings (Figure 4). PAF-RAG achieves the lowest ASR in most evaluated attack scenarios while maintaining moderate runtime overhead. Specifically, PAF-RAG requires approximately 0.65–0.83 minutes per batch across all datasets and attack settings, making it more computationally efficient than heavier defense-oriented methods such as TrustRAG and SeconRAG, while remaining only marginally slower than lightweight baselines such as VanillaRAG and InstructRAG. These results indicate that PAF-RAG’s additional security mechanisms introduce limited computational cost and maintain consistent runtime across different retrieval environments. Overall, PAF-RAG demonstrates a favorable trade-off between robustness and runtime efficiency, supporting its practical deployment in secure RAG systems.

6.3.4. Embedding Models (RQ-D4). We further evaluate PAF-RAG using four widely adopted embedding models: MiniLM [31], SimCSE [32], BERT [33], and BGE [34]. Table 8 reports results across three benchmark datasets under different poisoning numbers of documents. Overall, PAF-RAG consistently maintains strong robustness across different embedding spaces, achieving relatively high QA accuracy and low ASR even as the poisoning level increases. Among the evaluated models, MiniLM and BGE generally provide stronger robustness, while BERT exhibits comparatively lower performance under severe poisoning. These results demonstrate that PAF-RAG generalizes effectively across diverse retrieval backbones without relying on a specific embedding model.

TABLE 9: PPI ASR under encoder mismatch between attack and system on MS-MARCO using Mistral7b, reported as mean $\pm$ std over 5 runs. “Matched” indicates that the attack and system use the same encoder. PAF uses the full three-component defense.

Attack Encoder	System Encoder	PPI ASR (no PAF)	PPI ASR (+ PAF)
Contriever (matched)	Contriever	76.3 ± 0.8	3.2 ± 0.7
E5-large-v2 (mismatch)	Contriever	68.2 ± 1.4	7.4 ± 0.9
BGE-large-en (mismatch)	Contriever	64.7 ± 1.6	8.3 ± 1.0
GTE-large (mismatch)	Contriever	61.3 ± 1.8	7.3 ± 1.1
Contriever (mismatch)	E5-large-v2	74.9 ± 1.1	7.6 ± 0.8
E5-large-v2 (matched)	E5-large-v2	75.1 ± 0.7	6.9 ± 0.7
Hybrid (matched)	Hybrid	67 ± 0.8	5.5 ± 0.6
Hybrid (mismatch)	Contriever	60 ± 1.0	8.3 ± 0.9

TABLE 10: Ablation study. Each row removes one or two PAF components. ASR is reported using Mistral7b, averaged over 5 runs (mean $\pm$ std). Full PAF is the default configuration.

Configuration	HotpotQA ASR ($\downarrow$)	NQ ASR ($\downarrow$)	MS-MARCO ASR ($\downarrow$)
Drift Filter only	13.8 ± 1.3	12.4 ± 1.4	13.1 ± 1.1
Isolated Gen. only (no NLI)	24.5 ± 1.4	22.8 ± 1.8	23.6 ± 1.5
NLI Voting only (no drift)	19.7 ± 1.2	17.4 ± 1.6	18.3 ± 1.1
Drift + Isolated Gen. (no NLI)	11.2 ± 0.9	9.8 ± 1.0	10.3 ± 0.8
Drift + NLI (no Isolated Gen.)	15.6 ± 1.0	14.1 ± 1.2	14.4 ± 0.9
Full PAF (all three components)	6.6 ± 0.6	6.3 ± 0.5	3.2 ± 0.6

6.3.5. Encoder Mismatch (RQ-D4). In practice, the adversary may not know the exact embedding model used by the target system. Table 9 evaluates PPI under encoder mismatch between the attack and system. When using a mismatched encoder (e.g., E5-large-v2 for attack vs. Contriever for system), the undefended PPI ASR drops from 76.3% to 68.2%. While encoder mismatch reduces attack effectiveness, the attack remains highly successful, achieving a 68.2% ASR, which demonstrates that PPI transfers effectively across different embedding models. A hybrid retriever (BM25 + Contriever) exhibits a similar trend, with PPI achieving 67.0% ASR under matched encoders and 60.0% under mismatch.

Importantly, PAF maintains strong robustness across all mismatch conditions, with ASR consistently in the 6.9%–8.3% range. This indicates that the defense does not rely on encoder-specific properties of the attack, and remains effective even under cross-encoder transfer settings.

6.3.6. Ablation Study (RQ-D5). We conduct an ablation study with Mistral-7B to quantify the contribution of each PAF-RAG component to robustness against PPIs. Table 10 reports PPI attack success rate (ASR) on HotpotQA, NQ, and MS-MARCO, averaged over five independent runs. Each component provides a measurable robustness benefit.

Drift filtering alone reduces ASR to 13.8%, 12.4%, and 13.1% on HotpotQA, NQ, and MS-MARCO, respectively, indicating that anchor-based drift detection blocks a substantial fraction of PPI-poisoned documents before answer generation. Isolated generation alone lowers ASR to 24.5%, 22.8%, and 23.6% by preventing contamination across retrieved documents. NLI voting alone achieves ASRs of 19.7%, 17.4%, and 18.3%, demonstrating the effectiveness of consensus-based aggregation in suppressing outlier responses.

Combining all three components yields ASRs of 6.6%, 6.3%, and 3.2%, substantially outperforming any individual

mechanism. These results indicate that drift filtering, isolated generation, and NLI voting address complementary failure modes and together provide significantly stronger protection against PPIs.

7. Conclusion

We introduced Post-Partition Injection (PPI), the first adaptive corpus poisoning attack that exploits the structural timing asymmetry in partition-based RAG defenses. By reconstructing latent cluster structure from co-retrieval signals and placing injected documents within benign clusters, PPI achieves high attack success rates across multiple benchmarks. We then proposed Partition-Aware Filtering (PAF), a three-component defense comprising corpus anchoring, drift detection, and NLI-based voting. PAF reduces attack success rates between 3.2% to 8.6% across the most evaluated settings and is supported by a formal robustness certificate. Together, these contributions show that partition-based defenses require fundamental redesign to remain secure against adaptive adversaries and provide a principled, formally grounded framework for achieving such robustness.

Limitations and Future work. While PAF-RAG demonstrates robustness against various poisoning attacks, including PPI, several limitations point to important directions for future research. First, our threat model assumes that the adversary cannot modify existing corpus documents or intercept retrieval results; relaxing either assumption introduces a distinct and potentially stronger attack surface. Second, PAF’s drift detection mechanism depends on a fixed embedding model after time t_0 ; updating the encoder would require re-anchoring, which may introduce a temporary vulnerability window. Third, our evaluation is restricted to three English-language benchmarks, leaving performance on

multilingual and domain-specific corpora (e.g., medical or legal domains) an open question. Finally, Theorem 5.1 does not address adaptive adversaries that jointly optimize attacks to evade drift detection and subvert NLI voting through document-level semantic manipulation, motivating further theoretical and empirical investigation.

References

- [1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.
- [2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [3] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, “Retrieval augmented language model pre-training,” in *International conference on machine learning*. PMLR, 2020, pp. 3929–3938.
- [4] F. Wang, X. Wan, R. Sun, J. Chen, and S. O. Arik, “Astute rag: Overcoming imperfect retrieval augmentation and knowledge conflicts for large language models,” in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 30 553–30 571.
- [5] Microsoft, “Bing chat,” <https://www.microsoft.com/en-us/edge/features/bing-chat>, 2024.
- [6] Perplexity AI, “Perplexity ai,” <https://www.perplexity.ai/>, 2024, accessed: 2024.
- [7] LangChain, “Langchain,” <https://github.com/langchain-ai/langchain>, 2024, gitHub repository.
- [8] J. Liu, “Llamaindex,” <https://www.llamaindex.ai/>, 11 2022, formerly known as GPT Index.
- [9] W. Zou, R. Geng, B. Wang, and J. Jia, “PoisonedRAG: Knowledge corruption attacks to Retrieval-Augmented generation of large language models,” in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 3827–3844.
- [10] Z. Shen, B. Imana, T. Wu, C. Xiang, P. Mittal, and A. Korolova, “Reliabilityrag: Effective and provably robust defense for rag-based web-search,” in *Advances in Neural Information Processing Systems*, vol. 38, 2025, pp. 45 662–45 702.
- [11] C. Xiang, T. Wu, Z. Zhong, D. Wagner, D. Chen, and P. Mittal, “Certifiably robust rag against retrieval corruption,” *arXiv preprint arXiv:2405.15556*, 2024.
- [12] H. Zhou, K.-H. Lee, Z. Zhan, Y. Chen, Z. Li, Z. Wang, H. Haddadi, and E. Yilmaz, “Trustrag: enhancing robustness and trustworthiness in retrieval-augmented generation,” *arXiv preprint arXiv:2501.00879*, 2025.
- [13] M. Kim, H. Lee, and H. Koo, “Rescuing the unpoisoned: Efficient defense against knowledge corruption attacks on rag systems,” *arXiv preprint arXiv:2511.01268*, 2025.
- [14] X. Si, M. Zhu, S. Qin, L. Yu, L. Zhang, S. Liu, X. Li, R. Duan, Y. Liu, and X. Jia, “Secon-rag: A two-stage semantic filtering and conflict-free framework for trustworthy rag,” *Advances in Neural Information Processing Systems*, vol. 38, pp. 70 652–70 681, 2025.
- [15] Z. Chen, Z. Xiang, C. Xiao, D. Song, and B. Li, “Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 130 185–130 213, 2024.
- [16] R. Sui, “Ctrlrag: Black-box adversarial attacks based on masked language models in retrieval-augmented language generation,” *arXiv preprint arXiv:2503.06950*, 2025.
- [17] A. Shafran, R. Schuster, and V. Shmatikov, “Machine against the {RAG}: Jamming {Retrieval-Augmented} generation with blocker documents,” in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 3787–3806.
- [18] B. Zhang, Y. Chen, Z. Liu, L. Nie, T. Li, Z. Liu, and M. Fang, “Practical poisoning attacks against retrieval-augmented generation,” *arXiv preprint arXiv:2504.03957*, 2025.
- [19] J. Xue, M. Zheng, Y. Hu, F. Liu, X. Chen, and Q. Lou, “Badrag: Identifying vulnerabilities in retrieval augmented generation of large language models,” *arXiv preprint arXiv:2406.00083*, 2024.
- [20] Z. Zhong, Z. Huang, A. Wettig, and D. Chen, “Poisoning retrieval corpora by injecting adversarial passages,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 13 764–13 775.
- [21] 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.
- [22] N. Carlini, M. Jagielski, C. A. Choquette-Choo, D. Paleka, W. Pearce, H. Anderson, A. Terzis, K. Thomas, and F. Tramèr, “Poisoning web-scale training datasets is practical,” in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2024, pp. 407–425.
- [23] S. Cohen, R. Bitton, and B. Nassi, “Here comes the ai worm: Unleashing zero-click worms that target genai-powered applications,” *arXiv preprint arXiv:2403.02817*, 2024.
- [24] U. von Luxburg, “A tutorial on spectral clustering. statistics and computing,” *Data Structures and Algorithms (cs. DS); Machine Learning*, pp. 395–416, 2007.
- [25] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, “Hotflip: White-box adversarial examples for text classification,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 31–36.
- [26] P. He, X. Liu, J. Gao, and W. Chen, “Deberta: Decoding-enhanced bert with disentangled attention,” *arXiv preprint arXiv:2006.03654*, 2020.
- [27] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen *et al.*, “Ms marco: A human generated machine reading comprehension dataset,” *arXiv preprint arXiv:1611.09268*, 2016.
- [28] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning, “HotpotQA: A dataset for diverse, explainable multi-hop question answering,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.
- [29] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee *et al.*, “Natural questions: a benchmark for question answering research,” *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 453–466, 2019.
- [30] J. Zheng, A. P. Gema, G. Hong, X. He, P. Minervini, Y. Sun, and Q. Xu, “Grada: graph-based reranking against adversarial documents attack,” in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025, pp. 22 255–22 277.
- [31] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers,” *Advances in neural information processing systems*, vol. 33, pp. 5776–5788, 2020.
- [32] T. Gao, X. Yao, and D. Chen, “Simcse: Simple contrastive learning of sentence embeddings,” in *Proceedings of the 2021 conference on empirical methods in natural language processing*, 2021, pp. 6894–6910.

- [33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.
- [34] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, “Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation,” *arXiv preprint arXiv:2402.03216*, vol. 4, no. 5, 2024.

Appendix A. Datasets

MS-MARCO. MS MARCO (Microsoft MACHINE Reading COMprehension) is a large-scale dataset widely used for NLP and information retrieval (IR) tasks, including question answering, passage ranking, and document ranking. It provides download tables for the roughly 3.2 million-document corpus with 367,013 training queries and the 8.8 million-passage collection with over a million queries, in TSV and TREC formats.

Natural Questions(NQ). NQ is an open-domain QA benchmark built from genuine, anonymized queries submitted to Google search rather than from crowdworker-authored questions. For each query, an annotator inspects a Wikipedia article drawn from the top five search results and marks both a long answer, typically a single paragraph, and a short answer consisting of one or more entities, leaving either field empty when the page does not support one. The public release comprises 307,373 singly annotated training instances, 7,830 development instances with five annotations each, and 7,842 similarly annotated test instances held out for evaluation. The authors additionally report a 25-way annotation study on 302 examples to quantify human disagreement, propose evaluation metrics with high human upper bounds, and provide baselines from competitive prior systems.

HotpotQA. HotpotQA targets a gap in earlier QA resources, which rarely require multi-step reasoning or justified answers. It contains roughly 113k Wikipedia-based question-answer pairs where the evidence needed is typically spread across more than one document, and because the questions are not tied to a fixed knowledge base schema, they vary widely in topic and phrasing. Each instance carries sentence-level supporting facts, giving strong supervision for the reasoning chain and enabling models to explain their predictions. The dataset also introduces comparison questions that require contrasting facts about two entities, and remains challenging for current systems.

Appendix B. Analysis

B.0.1. Why PPI Works?. The fundamental reason PPI succeeds against partition-based defenses is the assumption shared by the partition-based defenses, identified in Section 3: that adversarially crafted documents will appear as embedding outliers relative to the benign corpus distribution, forming anomalous clusters separable from benign ones. An

adaptive adversary who interacts with the retrieval interface can place poisoned documents inside a benign cluster. The probe phase reconstructs the partition with $\text{NMI} > 0.7$ using only retrieval interface signals, without access to embeddings or clustering parameters.

The interpolation construction in Eq. 9 places the injected document’s embedding inside the target cluster, so that, for an appropriate ε , it is absorbed into that cluster rather than isolated. Simultaneously, the non-zero ε component in the query direction ensures sufficient retrieval relevance. In contrast, prior attacks such as PoisonedRAG optimize solely for retrieval relevance, producing embeddings that cluster near the query direction and therefore near one another, precisely the pattern that partition-based anomaly detectors are designed to flag.

Figure 3 provides empirical insight into why PPI achieves high attack success rates. Figure 3 (a) shows the co-retrieval matrix constructed from probe queries, where each entry M_{ij} counts how often documents d_i and d_j co-appeared in the same top- k retrieval result across $T_{\text{probe}} = 100$ probe queries. However, high co-retrieval values appear dispersed due to arbitrary document ordering, obscuring the latent partition. In Figure 3(b), spectral clustering is applied to the normalized graph Laplacian of M . Reordering documents according to their inferred cluster assignments reveals a clear block-diagonal structure: documents within the same cluster (e.g., C_1, C_2, C_3) exhibit dense mutual co-retrieval, while cross-cluster interactions remain sparse. This demonstrates that co-retrieval statistics alone are sufficient to recover the underlying partition.

Figure 3(c) illustrates the failure mode of partition-based defenses. Traditional poisoning attacks produce embeddings that are near-duplicates of one another and lie apart from the benign clusters, and are therefore flagged as anomalous (red triangle). In contrast, PPI constructs poisoned documents whose embeddings lie within the target cluster. Specifically, the cluster whose centroid is most similar to the query embedding. As a result, these injected documents (yellow marker) conform to the cluster distribution and evade anomaly detection while remaining highly likely to be retrieved.

B.1. PAF Robustness Certificate (Theorem 5.1)

We provide the full proof of the PAF robustness certificate stated in Section 5.4. We first establish notation, then prove two lemmas before assembling the main result.

Setup and Notation

Let $\mathcal{F} = \mathcal{F}_c \cup \mathcal{F}_a$ denote the set of k_f documents passed to the generation stage after drift filtering, where $\mathcal{F}_c$ contains the $k_c = k_f - k_{\text{adv}}$ clean documents and $\mathcal{F}_a$ contains the k_{adv} adversarial documents. By assumption, $k_{\text{adv}} < k_f/2$, so $k_c > k_{\text{adv}}$ and clean documents form a strict majority.

For document $d_i \in \mathcal{F}$, let $a_i = \mathcal{G}(q^*, d_i)$ denote the candidate answer generated in isolation (Equation 16). We say

a_i is a *correct candidate* if it semantically entails the ground-truth answer $a^\dagger$, and an *adversarial candidate* otherwise. We write $\text{NLI}(a_j \rightarrow a_i) \in \{\text{ENTAIL}, \text{NEUTRAL}, \text{CONTRADICT}\}$ for the output of the cross-encoder on the premise a_j and hypothesis a_i , and define the indicator:

$$e_{ji} = \mathbf{1}[\text{NLI}(a_j \rightarrow a_i) = \text{ENTAIL}]. \quad (18)$$

The NLI vote score for candidate a_i is $V_i = \sum_{j \neq i} e_{ji}$, and the aggregate answer is $a^* = a_{i^*}$ where $i^* = \arg \max_i V_i$ (Equation 17).

Ideal NLI oracle. We first analyze the case of a perfect NLI oracle ($\varepsilon_{\text{NLI}} = 0$), then introduce errors in Lemma B.2. Under the ideal oracle, $e_{ji} = 1$ if and only if a_j and a_i are both correct candidates (i.e., both entail $a^\dagger$), or both adversarial candidates, and $e_{ji} = 0$ whenever one is correct and the other is adversarial.

Adversarial candidate model. By construction of PPI, every adversarial document $d \in \mathcal{F}_a$ contains the attacker-specified answer a_{adv}^* , so $\mathcal{G}(q^*, d) \approx a_{\text{adv}}^*$ for all $d \in \mathcal{F}_a$. All adversarial candidates are therefore mutually entailing: $e_{ji} = 1$ for all $d_i, d_j \in \mathcal{F}_a$, $i \neq j$. Clean candidates are mutually entailing under the ideal oracle for the analogous reason: they all entail $a^\dagger$.

Lemma 1: Vote Scores Under the Ideal Oracle

Lemma B.1. *Under the ideal NLI oracle, the vote score of every correct candidate a_i (with $d_i \in \mathcal{F}_c$) satisfies $V_i = k_c - 1$, and the vote score of every adversarial candidate a_i (with $d_i \in \mathcal{F}_a$) satisfies $V_i = k_{\text{adv}} - 1$. Consequently, $V_i > V_j$ for all $d_i \in \mathcal{F}_c$ and $d_j \in \mathcal{F}_a$ whenever $k_c > k_{\text{adv}}$.*

Proof. Fix a correct candidate a_i with $d_i \in \mathcal{F}_c$. Under the ideal oracle, $e_{ji} = 1$ if and only if a_j also entails $a^\dagger$, i.e., $d_j \in \mathcal{F}_c$. There are $k_c - 1$ such documents (all clean documents except d_i itself), so $V_i = k_c - 1$.

Fix an adversarial candidate a_i with $d_i \in \mathcal{F}_a$. Under the ideal oracle, $e_{ji} = 1$ if and only if a_j also entails a_{adv}^* , i.e., $d_j \in \mathcal{F}_a$. There are $k_{\text{adv}} - 1$ such documents, so $V_i = k_{\text{adv}} - 1$.

Since $k_c > k_{\text{adv}}$ by assumption, we have $k_c - 1 \geq k_{\text{adv}} > k_{\text{adv}} - 1$, so every correct candidate strictly outscores every adversarial candidate. The $\arg \max$ in Equation 17 therefore selects a correct candidate, and the NLI vote returns an answer entailing $a^\dagger$. $\square$

Lemma 2: Stability Under NLI Errors

We now introduce a stochastic NLI model. Each indicator e_{ji} is subject to independent error: with probability ε_{NLI} , the NLI cross-encoder flips the true entailment decision. Formally, let e_{ji}^* denote the ideal-oracle indicator and $\xi_{ji} \sim \text{Bernoulli}(\varepsilon_{\text{NLI}})$ be an independent error indicator; then the observed indicator is $e_{ji} = e_{ji}^* \oplus \xi_{ji}$, where $\oplus$ denotes XOR.

Lemma B.2. *Under the stochastic NLI model with per-comparison error rate ε_{NLI} , define the vote margin $\Delta = (k_c - 1) - (k_{\text{adv}} - 1) = k_c - k_{\text{adv}}$. If*

$$\varepsilon_{\text{NLI}} < \frac{\Delta}{2k_f} = \frac{k_f - 2k_{\text{adv}}}{2k_f}, \quad (19)$$

then with probability at least $1 - k_f \varepsilon_{\text{NLI}}$, every correct candidate retains a higher vote score than every adversarial candidate, and the NLI vote selects a correct answer.

Proof. For a fixed correct candidate a_i ($d_i \in \mathcal{F}_c$), let $\tilde{V}_i = V_i^* + \eta_i$ where $V_i^* = k_c - 1$ is the ideal-oracle vote score and $\eta_i = \sum_{j \neq i} (\xi_{ji}^{(1)} - \xi_{ji}^{(0)})$ is the net change due to errors. Here $\xi_{ji}^{(1)}$ are errors that incorrectly add a vote (false entailment from a non-entailing pair) and $\xi_{ji}^{(0)}$ are errors that incorrectly remove a vote (missed entailment from an entailing pair). Each $\xi_{ji}^{(b)}$ is Bernoulli(ε_{NLI}), independently across pairs.

The expected perturbation is $\mathbb{E}[\eta_i] = 0$. By a union bound over the $k_f - 1 \leq k_f$ comparisons involving a_i , the probability that any single comparison is flipped is at most $k_f \varepsilon_{\text{NLI}}$.

Analogously, for a fixed adversarial candidate a_j ($d_j \in \mathcal{F}_a$), the perturbed score satisfies $\tilde{V}_j \leq (k_{\text{adv}} - 1) + \eta_j^+$ where η_j^+ counts false-positive entailment votes gained from errors. In the worst case, every one of the k_c cross-comparisons with correct candidates is erroneously flipped to entail, giving $\eta_j^+ \leq k_f \varepsilon_{\text{NLI}}$ in expectation.

For the vote to be overturned we need $\tilde{V}_j \geq \tilde{V}_i$, i.e., $(k_{\text{adv}} - 1) + k_f \varepsilon_{\text{NLI}} \geq (k_c - 1) - k_f \varepsilon_{\text{NLI}}$, which simplifies to:

$$2k_f \varepsilon_{\text{NLI}} \geq k_c - k_{\text{adv}} = \Delta. \quad (20)$$

Equation 20 is violated precisely when Equation 19 holds. Therefore, if $\varepsilon_{\text{NLI}} < \Delta/(2k_f)$, no adversarial candidate can outscore a correct candidate, even accounting for errors. By the union bound over all k_f candidates, the probability that this guarantee fails for any candidate is at most $k_f \varepsilon_{\text{NLI}}$. $\square$

Appendix C.

Boundary Conditions of the PAF Certificate

C.1. Deployment Scope of Assumption 1

Assumption 1 constrains k_{adv} , the number of adversarial documents among the k_f passed to generation. Because each injected document occupies at most one retrieval slot and drift filtering only removes documents, $k_{\text{adv}} \leq M$, where M is the adversary’s per-query injection budget. The assumption therefore holds by construction whenever $M < k_f/2$ ($M \leq 4$ for $k_f = 10$), which accounts for the zero violation rate observed $M \leq 3$ in Table 2. When quarantining reduces the number of admitted documents, the bound applies to the realised k_f rather than to its maximum, so operators should calibrate against the smallest k_f they expect to observe in practice. The operative question for a deployment is therefore not whether PPI evades the drift filter but how many documents a single adversary can place for one target query. This is determined by the corpus write path rather than by the defense.

Conditions favoring the bound. The bound on M is easier to enforce when corpus contributions are authenticated, rate-limited per account, or reviewed before indexing. Enterprise knowledge bases, curated document stores, and

systems that ingest content from a fixed set of vetted feeds provide such controls. Retrieval policies can also limit the number of near-duplicate or same-source documents in the top- k , constraining k_{adv} even when M is large C.2. Finally, topics with dense clean coverage provide a statistical form of protection because many benign passages compete for the available retrieval slots C.3

Conditions defeating the bound. open-write corpora may allow a single contributor to submit many documents without a strict per-contributor limit. Public wikis, forums, and comment sections may accept content from unauthenticated sources. Identity-based rate limits can also be bypassed using multiple accounts. Therefore, if an adversary can inject enough documents for the same target query, the $k_f/2$ bound can be exceeded.

Implications for operators. Deployments in the second category should not rely solely on the certificate’s guarantee and should instead consider PAF’s empirical robustness, which reduces PPI’s ASR to 3.2-8.6%. Alternatively, increasing k_f restores the bound $k_{adv} \leq M < k_f/2$ for a larger M , at the cost of one additional isolated generation call per document. The runtime measurements in Section 6.3.3 indicate this scales linearly and remains practical for the values of k_f considered here.

C.2. Large Injection Budgets ($M \geq 5$)

Root cause. PPI’s cluster-aligned construction (Eq. 9) is explicitly designed to satisfy the drift constraint while remaining competitive under cosine retrieval. As M increases, the probability that at least $\lceil k_f/2 \rceil$ injected documents both pass filtering and appear in the top- k_f retrieved set increases correspondingly. This limitation follows from attacker access to the retrieval function rather than a deficiency of the drift filter itself. Empirically, Table 2 reports violation rates of 4%, 19%, and 23% on MS-MARCO, NQ, and HotpotQA, respectively.

Potential extensions. The certified budget may be increased through: (i) a query-adaptive threshold

$$\tau(q) = \text{Percentile}_{95}(\{\text{drift}(d) : d \in \text{top-}k(q, D_0)\}),$$

computed from the trusted corpus D_0 , which tightens filtering within the query-relevant cluster; and (ii) a retrieval diversity constraint enforcing a minimum pairwise cosine distance δ among retrieved documents, preventing multiple near-duplicate PPI injections from simultaneously occupying top- k positions. Similar diversity constraints are widely used in production retrieval systems.

C.3. Dataset Retrieval Sparsity

The violation rate at $M \geq 5$ varies substantially across datasets, ranging from 4% on MS-MARCO to 19% on HotpotQA. This variation is largely explained by *retrieval density*, i.e., the average cosine similarity between the query and its top- k clean passages. Datasets such as MS-MARCO contain single-intent queries for which relevant passages

often achieve high retrieval scores, leaving less opportunity for adversarial documents to enter the top- k . In contrast, HotpotQA’s multi-hop questions distribute relevance across multiple passages, producing a flatter ranking landscape in which cluster-aligned adversarial documents can compete more effectively. This sensitivity is inherent to cosine-similarity retrieval and is not specific to PAF; any defense whose guarantees depend on the retrieval ranking stage will exhibit analogous dataset-dependent behavior.

C.4. High-Quality Embedding Mimicry

Lemma B.1 assumes that adversarial candidates are not entailed by the ground-truth answer $a^\dagger$. This assumption can fail under *embedding mimicry*: an adversary may state $a^\dagger$ in an initial sentence while embedding the target misinformation in subsequent content. Such candidates can be classified as entailing $a^\dagger$ by the NLI cross-encoder, increasing their vote weight and reducing the effective voting margin Δ .

Practical relevance. Successful mimicry requires knowledge of both the ground-truth answer $a^\dagger$ and the target incorrect answer a_{adv}^* . Under the threat model of Section 3.3, the adversary observes only a_{adv}^* and has no access to $a^\dagger$, making this attack substantially more difficult in practice. Nevertheless, mimicry remains plausible in closed-domain settings where the correct answer is publicly known (e.g., entity-centric or fact-verification tasks). Addressing this failure mode may require stronger fact-level entailment verification mechanisms beyond sentence-level NLI.

Appendix D. Ethics Considerations

This work identifies a previously overlooked vulnerability in partition-based RAG defenses and proposes a corresponding mitigation. Although the proposed PPI attack demonstrates how an adversary can exploit the gap between partitioning and corpus updates, our goal is to improve the security of RAG systems rather than enable misuse. We validated our approach in a controlled experimental setting using publicly available datasets and models, and did not target any real-world deployed systems.

Many real-world RAG applications rely on continuously updated knowledge sources, making it important to understand how adaptive adversaries may manipulate retrieved information. By exposing this weakness and introducing PAF as a defense, we aim to help researchers and practitioners evaluate the robustness of deployed systems and develop stronger safeguards against corpus poisoning. We hope this work contributes to the secure, transparent, and responsible deployment of retrieval-augmented AI systems.

Appendix E.

Meta-Review

The following meta-review was prepared by the program committee for the 2027 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

E.1. Summary

This paper studies poisoning attacks against Retrieval-Augmented Generation systems, focusing on defenses that rely on partitioning of the retrieval corpus. It introduces Post-Partition Injection (PPI), an adaptive attack that probes the retrieval interface to infer the latent cluster structure and crafts poisoned documents that remain within benign clusters while influencing retrieval and generation. The paper also proposes Partition-Aware Filtering (PAF), which combines trusted corpus anchoring, drift-based insertion filtering, document-isolated generation, and NLI-based aggregation. The evaluation spans multiple datasets, LLMs, attacks, and defenses.

E.2. Scientific Contributions

- Identifies an Impactful Vulnerability
- Provides a Valuable Step Forward in an Established Field

E.3. Reasons for Acceptance

- 1) The paper identifies an important weakness in clustering-based RAG defenses and presents a concrete adaptive attack that exploits the gap between partition construction and subsequent corpus updates.
- 2) PPI is technically well motivated and empirically effective, particularly against defenses that rely on embedding-space clustering or partitioning.
- 3) The paper complements the attack with PAF, a defense that combines trusted corpus anchoring, insertion filtering, isolated generation, and NLI-based aggregation, and substantially reduces attack success rates across the evaluated settings.
- 4) The evaluation is broad, covering multiple datasets, generator models, attacks, defenses, embedding models, sensitivity analyses, ablations, and runtime overheads. The conditional robustness analysis and public implementation further strengthen the work.