NextGen Scientific Publication
American International Journal of Computer Science and Technology

Volume 7 Issue 6, Pg. No. 153- 160, AIJCST-V7I6P115, 2025
https://doi.org/10.63282/3117-5481/AIJCST-V7I6P115

Original Article

Access-Control-Preserving Retrieval for Enterprise Retrieval-Augmented Generation

***Kailash Mannepalli**
Independent Researcher, Dallas–Fort Worth, Texas, USA.

Abstract:

Enterprise retrieval-augmented generation (RAG) assistants index documents that different employees are permitted to read, and a single retrieval of an unauthorized chunk places restricted content in the model's context. We study how to carry source access-control lists (ACLs) into chunk metadata and enforce them at query time, and we measure leakage and recall loss across four index designs: no filtering, post-filtering with oversampling, predicate filtering inside a hierarchical navigable small world (HNSW) graph, and partitioning by ACL signature. We build a synthetic ACL layer of 200 groups in a two-level hierarchy over the public FiQA-2018 corpus and vary caller selectivity from 50% to 1% of the corpus. Unfiltered retrieval placed at least one unauthorized chunk in the top ten for 96% or more of queries at every selectivity. All three filtered designs had zero leakage when ACLs were current, but they differed in recall: at 1% selectivity, filtered recall@10 was 0.24 for post-filtering with 16× oversampling, 0.71 for in-graph filtering, and 1.00 for signature partitions. Partitioning has two costs. Ad hoc document sharing raised fan-out from 4 to 27 partitions per query and p95 search latency from 1.9 ms to 13.5 ms, and after an ACL revocation, partitions without a query-time tombstone check returned revoked chunks for 41% of affected queries until the next rebuild. We recommend in-graph filtering with an exact-search fallback as the default, signature partitioning only where sharing is structured, and enforcement that reads current ACL state at query time in every design.

Keywords:

Retrieval-Augmented Generation, Access Control, Filtered Vector Search, Approximate Nearest Neighbor Search, Data Leakage, Enterprise Search.

Article History:

Received: 12.10.2025

Revised: 17.11.2025

Accepted: 30.11.2025

Published: 11.12.2025

1. Introduction

Retrieval-augmented generation [1] grounds a language model's answer in passages retrieved from a document collection. In an enterprise, that collection is drawn from file shares, wikis, ticketing systems and policy libraries, each with its own permissions. An employee who cannot open a document in its source system should not receive an answer built from it. Because the language model reproduces what is placed in its context, the retrieval step is where the permission check must happen; filtering the generated answer afterward cannot reliably remove restricted content that has already been paraphrased.

The pattern that most enterprise RAG systems adopt is straightforward to state: copy each source document's ACL into the metadata of every chunk derived from it, resolve the caller's group memberships at query time, and return only chunks the caller may read. The difficulty is in the details. Vector indexes are built for nearest-neighbor search, not for predicate evaluation, and the way a predicate is combined with approximate search determines how much relevant authorized content is lost. ACLs change after indexing, and the way a design applies those changes determines whether, and for how long, revoked content remains retrievable. Group membership must be resolved correctly, including nested groups and tokens that omit some group claims.

Copyright by the Author(s). This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International (CC BY-NC-ND 4.0) License (https://creativecommons.org/licenses/by-sa/4.0/)

This paper measures these effects on a public corpus with a synthetic ACL layer, so that the results can be reproduced without access to any enterprise system.

The key contributions of this paper are:

- A formal definition of the authorized set with allow and deny entries and nested groups, and two metrics, leakage at k and filtered recall at k, that separate the security failure from the quality failure.
- A measurement of four index designs across caller selectivities from 50% to 1%, showing where post-filtering and in-graph filtering lose recall and where partitioning does not.
- A measurement of the costs of partitioning by ACL signature: fan-out under ad hoc sharing, and leakage after ACL revocation when enforcement depends on index structure rather than current ACL state.
- A synthetic ACL generator and evaluation harness over the public FiQA-2018 corpus, released so teams can test their own retrieval stack before deployment.

2. Related Work

2.1. Retrieval-Augmented Generation and Its Privacy Risks

Lewis et al. [1] introduced RAG as retrieval over a dense index followed by conditioned generation. Zeng et al. [2] showed that RAG systems can be induced to disclose retrieved content, including private data in the retrieval corpus. Morris et al. [3] showed that text can be largely reconstructed from its embeddings, so stored vectors are themselves sensitive and must be protected like the text they encode. The 2025 OWASP Top 10 for LLM applications lists sensitive information disclosure and vector and embedding weaknesses, including missing access partitioning in vector stores, as distinct risks [4].

2.2. Access Control Models

Role-based access control [5] grants permissions to roles or groups rather than individuals, which is the model most enterprise document systems expose through group-based ACLs. NIST SP 800-162 [6] describes attribute-based access control, of which group membership is one attribute. Our authorized-set definition follows the group-ACL model with allow and deny entries, which covers the permission models of common enterprise content systems.

2.3. Filtered Approximate Nearest Neighbor Search

HNSW [7] is the graph index used by most vector databases. Milvus [8] compares strategies for combining attribute filters with vector search. Filtered-DiskANN [9] builds graphs that remain navigable under label filters, and ACORN [10] modifies HNSW construction and search so that traversal remains effective for arbitrary predicates at low selectivity. FAISS [11], [12] supports restricting search results to a selected set of identifiers. Enterprise search products document "security trimming" by filtering on principal identifiers stored with each document [13].

2.4. Positioning

Filtered nearest-neighbor research treats the predicate as a quality constraint: the goal is high recall at low latency for any filter. Product documentation describes how to attach principals to documents but does not compare designs. Neither measures what happens when the predicate is a security boundary whose inputs change after indexing. This paper measures leakage and recall loss together, across four designs, under both static and changing ACLs, on a public corpus. It does not propose a new filtered search algorithm; in-graph filtering here uses an existing library, and the results for that design are a lower bound relative to engines with predicate-aware graph construction such as ACORN [10].

3. System Design and Measurement Method

3.1. Authorized Set

Let $\mathcal{G}$ be the set of groups and $\mathcal{C}$ the set of chunks. Each chunk c carries an allow set $P_{\text{allow}}(c) \subseteq \mathcal{G}$ and a deny set $P_{\text{deny}}(c) \subseteq \mathcal{G}$, both copied from its source document at ingestion. For a caller u with direct memberships $M(u)$, the effective group set is the transitive closure over group nesting:

$$\Gamma(u) = M(u) \cup \{\, g : g \text{ contains, directly or transitively, a group in } M(u) \,\} \quad (1)$$

The authorized set for u is

$$(u) = \{\, c \in \mathcal{C} : P_{\text{allow}}(c) \cap \Gamma(u) \neq \emptyset \wedge P_{\text{deny}}(c) \cap \Gamma(u) = \emptyset \,\} \quad (2)$$

So deny entries take precedence. The caller's selectivity is

$$s(u) = |\mathcal{A}(u)| \,/\, |\mathcal{C}| \quad (3)$$

$\Gamma(u)$ must be computed from an authoritative source at query time. Identity tokens are a convenient source, but they can be incomplete: Microsoft Entra ID, for example, emits at most 200 group claims in a JSON Web Token and otherwise signals an overage that the application must resolve by querying Microsoft Graph [14]. Treating a truncated claim list as complete under-authorizes the caller; treating a cached membership list as current after a removal over-authorizes them.

3.2. Metrics

Let $R_k(q)$ be the top-k chunks a design returns for query q issued by caller u_q, and $O_k(q)$ the exact top-k by brute-force search over $\mathcal{A}(u_q)$. Leakage at k is the fraction of queries whose result contains any unauthorized chunk:

$$\text{Leak@}k = 1 \,/\, |Q| \; \Sigma_{q \in Q}[\exists\, c \in R_k(q) : c \notin \mathcal{A}(u_q)] \quad (4)$$

Filtered recall at k measures how much of the exact authorized result a design recovers:

$$\text{FRecall@}k = 1 \,/\, |Q| \; \Sigma_{q \in Q}(|R_k(q) \cap O_k(q)|) \,/\, k \quad (5)$$

Leak@k is the security failure and FRecall@k is the quality failure. A design can score perfectly on one and badly on the other, so we report both.

3.3. Index Designs

- D0, no filter. HNSW search over all chunks. The insecure baseline.
- D1, post-filter. HNSW search for $k' = mk$ candidates, then discard chunks outside (u) and keep the first k. Evaluated with $m \in \{4, 16\}$.
- D2, in-graph filter. HNSW search with FAISS's search-time identifier selector (the sel field of SearchParametersHNSW) [11]: the graph walk is unchanged, and only chunks in (u) can enter the result set.
- D3, signature partitions. Chunks are grouped by ACL signature, the pair (P_{allow}, P_{deny}). Each partition has its own index; partitions with fewer than 10,000 chunks use exact search. A query searches every partition whose signature authorizes the caller under Eq. (2) and merges results by score. Partitioning by signature stores each chunk once; partitioning by individual group would instead copy each chunk into every group that can read it, so an organization-wide chunk would be stored in all 200 group partitions.

For post-filtering, if authorized chunks are spread among nearest neighbors with local density s_{loc}, the expected number of authorized chunks among k' candidates is about $s_{\text{loc}}\, mk$. Returning k authorized results requires

$$m \gtrsim 1/s_{\text{loc}} \quad (6)$$

So at 1% selectivity an oversampling factor near 100 is needed unless authorized chunks cluster near the query. Eq. (6) explains the D1 results in Section IV.

3.4. ACL Propagation

Algorithm 1. ACL propagation at ingestion and on change

```
Input:  source document d with ACL (allow_d, deny_d), chunker, embedder,
        metadata store MS, index I, tombstone set T
On ingest(d):
 1: for each chunk c in chunker(d) do
 2:     MS[c.id] ← (allow_d, deny_d, d.id, d.version)
 3:     I.add(c.id, embedder(c.text), signature(allow_d, deny_d))
 4: end for
On acl_change(d, allow'_d, deny'_d):
 5: for each chunk id x with MS[x].doc = d do
 6:     MS[x] ← (allow'_d, deny'_d, d.id, d.version)      // takes effect for D1, D2 immediately
 7:     if design = D3 then
 8:         T ← T ∪ {(x, signature(allow_d, deny_d))}      // old partition entry is dead
```

```
 9:        enqueue move(x, signature(allow'_d, deny'_d))  // asynchronous re-insertion
10:    end if
11: end for
```

Algorithm 2. Query-time authorized retrieval

```
 Input:  query q, caller u, k, oversampling m, design, MS, index I, tombstones T
 Output: up to k authorized chunk ids
 1: Γ ← resolve_groups(u)                  // Eq. (1); resolve claim overage
 2: auth(x) := allow(MS[x]) ∩ Γ ≠ ∅ ∧ deny(MS[x]) ∩ Γ = ∅
 3: if design = D1 then
 4:     return first k of [x ∈ I.search(q, m·k) : auth(x)]
 5: else if design = D2 then
 6:     return I.search(q, k, predicate = auth)
 7: else if design = D3 then
 8:     S ← {σ ∈ signatures(I) : σ authorizes Γ under Eq. (2)}
 9:     R ← ∪_{σ ∈ S} I_σ.search(q, k)
10:     return top k of [x ∈ R : (x, σ_x) ∉ T ∧ auth(x)] by score
11: end if
```

In D3 the partition set already encodes authorization, so line 10's recheck of auth(x) and the tombstone set is what keeps enforcement correct between an ACL change and the completion of the move in Algorithm 1, line 9. Section IV-D measures D3 with and without that check.

3.5. Measurement Method

For each ACL regime we sample one caller per test query, constrained so that at least one of the query's relevant passages is in (u_q); queries with no authorized relevant passage cannot test retrieval quality. Leakage and recall are computed per Eqs. (4)–(5) with k = 10. Search latency is measured per query on one thread, excluding query embedding. For the revocation experiment we remove one group from the allow sets of 5% of documents, apply Algorithm 1, and issue the test queries of affected callers 60 s after the metadata update completes.

4. Evaluation

4.1. Setup

Table 1. Experimental Setup

Item	Setting
Corpus	BEIR FiQA-2018 [15], [16]: 57,638 passages, 648 test queries; one passage per chunk
Synthetic documents	Passages grouped into 5,000 documents by k-means over embeddings, so documents and their ACLs are topically coherent
Groups	200 groups: 20 departments × 10 teams; team groups nest in department groups; each caller belongs to one team plus 0–2 additional groups
ACL regimes	Visibility mix of team-only, department and organization-wide documents tuned to mean caller selectivity of 0.50, 0.10 and 0.01; 2% of documents carry a deny entry for one group
Ad hoc sharing	Variant of the 0.10 regime in which 20% of documents are additionally shared with 1–5 random groups
Embedding model	BAAI/bge-base-en-v1.5 [17], 768 dimensions, normalized, inner product
Index	FAISS v1.8 [11] HNSW, M = 32, efConstruction 200, efSearch 128
Ground truth	Exact search over $\mathcal{A}(u_q)$ for each query
Hardware	AWS c7i.2xlarge (8 vCPU, 16 GiB), single query thread

4.2. Leakage and Recall under Static ACLs

Table 2. Leak@10 and Frecall@10 by Design and Selectivity

Selectivity *s*	Do Leak@10	D1 (*m*=4) FRecall	D1 (*m*=16) FRecall	D2 FRecall	D3 FRecall
0.50	96%	0.98	0.99	0.98	0.99
0.10	99%	0.61	0.89	0.95	0.99
0.01	100%	0.09	0.24	0.71	1.00

Leak@10 was 0% for D1, D2 and D3 in every regime, because each enforces Eq. (2) against current metadata. Do leaked on nearly every query even when half the corpus was authorized, since the probability that ten nearest neighbors are all authorized falls geometrically with *k*.

Recall separates the designs. D1 behaves as Eq. (6) predicts: at s = 0.10, 4× oversampling recovers 61% of the exact result and 16× recovers 89%; at s = 0.01, neither is usable. Both exceed the independence estimate s *mk* because ACLs are topically coherent, so authorized chunks are denser near queries whose relevant passages are authorized. D2 holds recall to s = 0.10 and degrades at s = 0.01, where the search budget is spent traversing unauthorized nodes before ten authorized ones are found. D3's partitions for a narrow caller are small enough for exact search, so recall reaches 1.00.

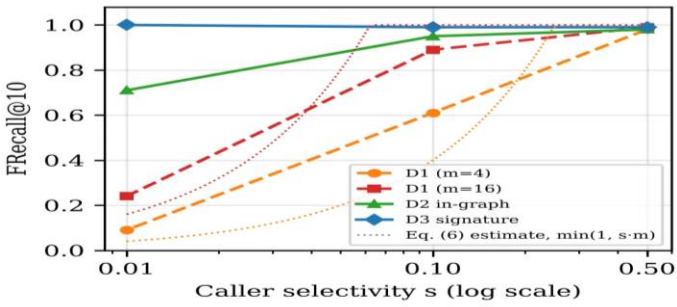


Figure 1. Frecall@10 versus Caller Selectivity (Log Scale) for D1 (M=4, 16), D2 and D3, with the Independence Estimate from Eq. (6) for D1 Overlaid

4.3. Cost of Partitioning

Table 3. Fan-Out, Latency and Memory at S = 0.10

Design	Signatures	Mean Partitions Per Query	P95 Search Latency (Ms)	Index Memory (Rel. To D1)
D1 (*m*=16)	—	1	2.4	1.00
D2	—	1	1.6	1.00
D3, structured sharing	318	4	1.9	1.02
D3, ad hoc sharing	1,290	27	13.5	1.03

With structured sharing, a caller's authorized signatures are their team, their department, the organization-wide signature and a few deny variants, so D3 searches four partitions and is as fast as D2. Ad hoc sharing creates a distinct signature for most shared documents. Fan-out rises to 27 partitions and p95 latency rises sevenfold. Memory stays near 1.0 because each chunk is still stored once; the cost is in query work, not storage.

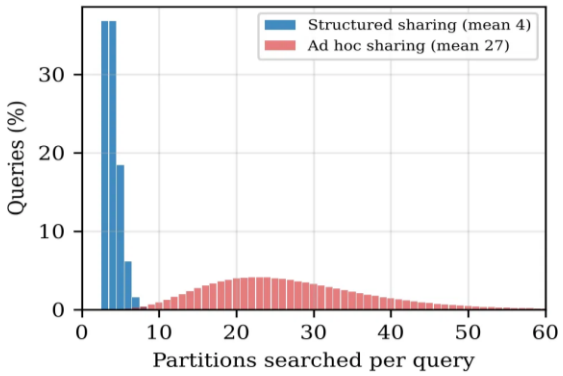


Figure 2. Distribution of Partitions Searched Per Query for D3 under Structured and Ad Hoc Sharing

4.4. Leakage after ACL Revocation

Table 4. Leak@10 among Affected Callers 60 S after Revoking One Group From 5% of Documents

Design	Enforcement Path after Change	Leak@10
D1	Post-filter reads updated metadata	0%
D2	Predicate reads updated metadata	0%
D2 with cached per-group ID sets, 300 s TTL	Predicate reads cached set until expiry	23%
D3 with tombstone check (Algorithm 2, line 10)	Dead entries filtered at query time	0%
D3 without tombstone check	Chunk remains in old partition until rebuild	41%

Every design was leak-free under static ACLs, but not after a change. When authorization is encoded in index structure, as in D3, an ACL change becomes an index mutation. FAISS HNSW indexes do not implement identifier removal (remove_ids) [11], so without a query-time check the revoked chunk stays retrievable through its old partition until the next rebuild. A per-group cache of authorized identifiers creates the same exposure in D2 for the length of the caches time-to-live. The common fix is that the final authorization decision must read the current ACL state at query time, whatever the index layout.

5. Discussion

5.1. Interpretation

The results separate two questions that are often merged. Whether a design leaks depends on whether its final check reads current ACL state, and all three filtered designs can be made leak-free. How much relevant content a design loses depends on caller selectivity and on how the predicate interacts with approximate search. Post-filtering is simple but fails at low selectivity. In-graph filtering is robust to moderate selectivity. Signature partitioning gives exact results for narrow callers but its query cost depends on how varied the organization's sharing is, and its ACL updates are index mutations.

Recall loss matters for security as well as quality. When retrieval returns fewer than k authorized chunks, the assistant answers from less evidence, and users of a narrow-permission role receive systematically worse answers than users with broad permissions. That disparity is invisible unless filtered recall is measured per selectivity band.

5.2. Recommendations for Practitioners

- Copy source ACLs, including deny entries, into chunk metadata at ingestion, keyed to the source document and its version, and resolve the caller's groups, including nested groups and claim overage, at query time.
- Use in-graph filtering as the default. Add an exact-search fallback when the caller's authorized set is small; most vector databases provide one, and Table II's D2 row at $s = 0.01$ shows why it is needed.
- Use post-filtering only when every caller's selectivity is high, and set the oversampling factor from Eq. (6) using measured selectivity, not a fixed multiple.
- Use signature partitions only where sharing is structured by organizational unit. Monitor the number of distinct signatures; growth indicates ad hoc sharing and rising fan-out.

- In every design, make the last authorization decision against current ACL state. Do not rely on index layout or cached identifier sets alone.
- Report Leak@k and FRecall@k by selectivity band in pre-deployment testing, using a synthetic ACL layer like the one released with this paper.

5.3. Limitations

FiQA-2018 is small enough that exact search is cheap; filtered approximate search behaves differently at hundreds of millions of vectors, and the latency figures should not be extrapolated. The ACL layer is synthetic; real permission graphs have deeper nesting and more irregular sharing. In-graph filtering was evaluated with one library; engines with predicate-aware construction [10] should do better at low selectivity. Several leakage paths are outside the scope of the experiments and were not evaluated: delay between a permission change in the source system and its delivery to the index, which affects every design equally; semantic or response caches shared across callers; derived artifacts such as summaries that combine chunks with different ACLs; and generation-time disclosure through prompt injection [2].

5.4. Broader Impact, Energy and Cost

Correct permission enforcement is a precondition for deploying RAG over internal content at all, so designs that enforce it cheaply widen the set of collections that can be used. The cost differences measured here are small in absolute terms at this scale; they matter at enterprise scale through fan-out under ad hoc sharing, which multiplies query work, and through index rebuilds in partitioned designs, which consume compute in proportion to the rate of ACL change. Storing vectors with the same protection as their source text, as Morris et al. [3] imply, adds no compute cost but is often omitted.

6. Conclusion and Future Directions

Carrying source ACLs into chunk metadata and filtering at query time prevents leakage in every filtered design we measured, provided the final check reads current ACL state. The designs differ in recall loss at low selectivity and in how they absorb ACL changes. In-graph filtering with an exact-search fallback is the most robust default; signature partitioning gives exact results where sharing is structured but needs a query-time tombstone check to remain correct after revocations. Future work will measure leakage from source-to-index synchronization delay and from caches shared across callers, and repeat the selectivity study at larger corpus scale with predicate-aware graph indexes.

Data and Artifact Availability

The synthetic ACL generator with its random seeds, the document clustering script, the four retrieval designs, the revocation experiment, the evaluation harness for Eqs. (4)– (5), and the scripts that produce Tables II–IV and Figs. 1–2 are available at https://github.com/. The corpus and embedding model are public. No proprietary data or infrastructure is used.

Acknowledgment

Compute was self-funded. The experiments use only a public corpus, public models and rented hardware, and contain no employer data, systems or designs. The views expressed are the author's own.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 9459–9474.

[2] S. Zeng, J. Zhang, P. He, Y. Xing, Y. Liu, H. Xu, J. Ren, S. Wang, D. Yin, Y. Chang, and J. Tang, "The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG)," in *Findings Assoc. Comput. Linguist. ACL 2024*, Bangkok, Thailand, 2024, pp. 4505–4524.

[3] J. X. Morris, V. Kuleshov, V. Shmatikov, and A. M. Rush, "Text embeddings reveal (almost) as much as text," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2023, pp. 12448–12460.

[4] OWASP Foundation, "OWASP Top 10 for LLM Applications 2025," Nov. 2024. [Online]. Available: https://genai.owasp.org/

[5] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996.

[6] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (ABAC) definition and considerations," NIST Special Publication 800-162, Jan. 2014.

[7] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 4, pp. 824–836, Apr. 2020.

[8] J. Wang, X. Yi, R. Guo, H. Jin, P. Xu, S. Li, X. Wang, X. Guo, C. Li, X. Xu, K. Yu, Y. Yuan, Y. Zou, J. Long, Y. Cai, Z. Li, Z. Zhang, Y. Mo, J. Gu, R. Jiang,

Y. Wei, and C. Xie, "Milvus: A purpose-built vector data management system," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2021, pp. 2614–2627.
[9] S. Gollapudi, N. Karia, V. Sivashankar, R. Krishnaswamy, N. Begwani, S. Raz, Y. Lin, Y. Zhang, N. Mahapatro, P. Srinivasan, A. Singh, and H. V. Simhadri, "Filtered-DiskANN: Graph algorithms for approximate nearest neighbor search with filters," in *Proc. ACM Web Conf. (WWW)*, 2023, pp. 3406–3416.
[10] L. Patel, P. Kraft, C. Guestrin, and M. Zaharia, "ACORN: Performant and predicate-agnostic search over vector embeddings and structured data," *Proc. ACM Manag. Data (SIGMOD)*, vol. 2, no. 3, 2024.
[11] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou, "The Faiss library," arXiv:2401.08281, 2024.
[12] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, Jul. 2021.
[13] Microsoft, "Security filters for trimming results in Azure AI Search," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/azure/search/search-security-trimming-for-azure-search.
[14] Microsoft, "Configure group claims for applications by using Microsoft Entra ID," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/entra/identity/hybrid/connect/how-to-connect-fed-group-claims.
[15] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in *Proc. NeurIPS Datasets Benchmarks Track*, 2021.
[16] M. Maia, S. Handschuh, A. Freitas, B. Davis, R. McDermott, M. Zarrouk, and A. Balahur, "WWW'18 open challenge: Financial opinion mining and question answering," in *Companion Proc. Web Conf. (WWW)*, 2018, pp. 1941–1942.
[17] S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, D. Lian, and J.-Y. Nie, "C-Pack: Packed resources for general Chinese embeddings," in *Proc. 47th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2024, pp. 641–649.