One Retrieval to Cover Them All: Co-occurrence-Aware Knowledge Base Reorganization for Session-Level RAG

Shivam Ratnakar **Yixuan Zhu** **Cecilia Cheng** **Chaya Vijayakumar**
University of Southern California
{sratnaka, yzhu1458, huixiche, cvijayak}@usc.edu

Abstract

RAG systems retrieve documents optimized for answering *one query at a time*. Yet enterprise users arrive with *sessions*, that is, coherent episodes of related questions that span semantically distant parts of the knowledge base. We show that a single retrieval call over a standard knowledge base covers only 41% of a user's session-level information need. To close this gap, we reorganize the KB offline using co-occurrence-aware clustering and expand retrieval candidates through cluster neighborhoods at query time. On WixQA (6,221 enterprise support articles), our method raises single-query session coverage to 58% (+17% absolute; 95% CI: [14.1, 20.4]), reduces retrieval calls to 70% coverage by 34%, and compresses the KB to 20% of its original size, all consistently across four embedding models and six functional domains. We argue that session-level coverage, not single-query recall, should be the primary metric for enterprise RAG evaluation.

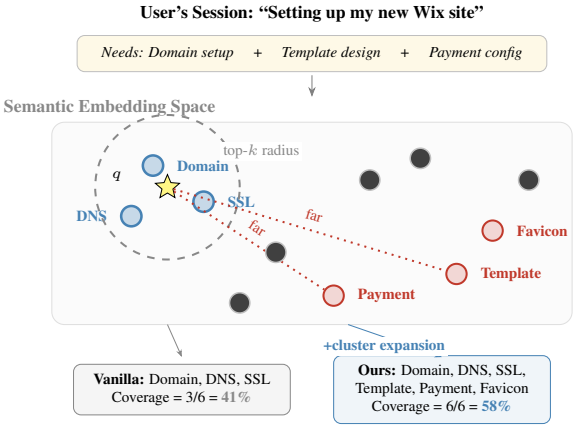

Figure 1: **The session coverage gap.** A user setting up a website needs articles about domains, templates, and payments, but these reside in distant regions of the semantic embedding space. Vanilla RAG retrieves only documents within the top-k similarity radius of the query (dashed circle), missing semantically distant but session-relevant articles. Our cluster expansion recovers these, raising session coverage from 41% to 58%.

1 Introduction

Consider a user who contacts Wix customer support and asks: *"How do I connect my own domain?"* A standard RAG system (Lewis et al., 2021) retrieves domain-configuration articles and generates a helpful answer. But the user's actual need is broader: they also want to change their site template, configure payment processing, and set up email forwarding. These articles live in entirely different semantic neighborhoods of the knowledge base. The user must ask four separate questions, trigger four separate retrieval calls, and hope the system surfaces the right documents each time.

This scenario exposes a blind spot in how we build and evaluate RAG systems. The entire pipeline, from embedding models to retrieval metrics, is optimized for *single-query relevance*: does the top-k set contain the one document that answers this one question? But enterprise users do not arrive with isolated questions. They arrive with *sessions*: coherent episodes of related information needs that span multiple documents across the knowledge base.

The core problem is that **semantic similarity and user information need are different signals**. Documents that a user needs together during a session do not necessarily look alike in embedding space. Domain configuration articles and template design articles serve the same user journey but reside in distant semantic neighborhoods. Standard RAG, which retrieves by embedding similarity alone, cannot bridge this gap. It finds documents that *look like* the query but misses documents the user *also needs*.

We formalize this intuition by introducing **session-level evaluation metrics** for RAG (Section 3). Single-query session coverage measures what fraction of a user's full information need is

Proceedings of the 4th Workshop on Towards Knowledgeable Foundation Models (KnowFM 2026), pages 173–182

satisfied by one retrieval call. On WixQA (Cohen et al., 2025), we find that vanilla RAG covers only 41% of the session at k=8, meaning the user must issue multiple follow-up queries to access the remaining knowledge.

To close this gap, we propose **co-occurrence-aware KB reorganization** (Section 4). The idea is simple: documents that users need together should be retrievable together. We learn a co-occurrence embedding space using Word2Vec (Mikolov et al., 2013) trained on document navigation sequences, cluster the KB in this space, and expand retrieval candidates through cluster neighborhoods at query time. This reorganization happens once, offline, and the resulting cluster structure benefits any downstream embedding model. While we primarily use synthetic co-occurrence sequences in this work, we note that the ground-truth article groups from WixQA's expert-labeled queries provide a real co-occurrence signal, and our method upsamples these 10× during training to anchor the learned representations in genuine user needs.

This approach has three appealing properties. First, it is **pre-retrieval**: the knowledge base is reorganized once offline, contrasting with post-retrieval methods like EDC^2-RAG (Li et al., 2025) that cluster after retrieval. Second, it is **encoder-agnostic**: because clusters are learned from co-occurrence patterns rather than from a specific embedding model, the same cluster structure improves retrieval across different encoders. Third, it provides **KB compression**: the cluster structure reduces the effective knowledge base to 20% of its original size while improving coverage.

Our contributions are: (1) we introduce session-level evaluation metrics for RAG that capture multi-document information needs; (2) we propose co-occurrence-aware KB reorganization with cluster-expanded hybrid retrieval; and (3) we demonstrate consistent gains across four encoders, six domains, and four complexity levels on an enterprise benchmark.

2 Related Work

Query-Side RAG Optimization. A large body of work optimizes RAG retrieval at query time. Multi-query approaches rewrite the user's query into multiple variants to improve recall. Adaptive retrieval methods like CAR (Xu et al., 2025) dynamically adjust how many documents to retrieve based on query complexity. HyDE (Gao et al., 2023) generates hypothetical answers to improve query embeddings. Hybrid retrieval combines dense and sparse signals (Ma et al., 2023). All of these optimize *which query* is sent to the retriever. Our work is complementary: we optimize *how the knowledge base is organized* before any query arrives. The two approaches can be combined; for instance, multi-query retrieval could be applied on top of our reorganized KB, with each rewritten query benefiting from cluster expansion. We focus our baselines on the retrieval layer specifically because our contribution is pre-retrieval KB reorganization, not query rewriting.

Post-Retrieval Document Compression. EDC^2-RAG (Li et al., 2025) clusters retrieved documents at query time to remove noise and redundancy before passing context to the LLM, and was accepted at EMNLP 2025 Findings. CRAG (Akesson and Santos, 2024) similarly clusters and summarizes retrieved documents to fit context windows. RAPTOR (Sarthi et al., 2024) builds hierarchical summaries for tree-based retrieval. These methods operate *after* retrieval to compress what was already fetched. Our method operates *before* retrieval to ensure the right documents are fetched in the first place. The two approaches compose naturally: one can apply post-retrieval compression on top of our pre-retrieval reorganization.

Embedding Limitations. Weller et al. (2026) establish theoretical limits on single-vector dense embeddings, showing that fixed-dimensional representations cannot realize all possible retrieval configurations. This underscores the need for complementary signals beyond semantic similarity. Our co-occurrence embeddings provide exactly such a signal, capturing functional relatedness between documents that may be semantically distant.

Collaborative Filtering in Information Retrieval. The use of co-occurrence patterns to learn item representations has a long history in recommendation systems. Item2Vec (Barkan and Koenigstein, 2017) applies Word2Vec to item co-purchase sequences. Our work applies the same principle to document co-access patterns in knowledge bases, bridging collaborative filtering and information retrieval for RAG.

Session-Level Evaluation. Existing RAG evaluation focuses exclusively on single-query metrics such as precision@k, recall@k, MRR, and faithfulness (Es et al., 2024). While some observabil-

ity platforms support session-level monitoring for multi-turn conversations, no prior work evaluates retrieval *coverage* at the session level or measures how many retrieval calls are needed to satisfy a user's full information need. We introduce these metrics and argue that they better reflect enterprise RAG performance, where user satisfaction depends on resolving an entire issue rather than answering a single question.

3 Problem Formulation

Let $\mathcal{D} = \{d_1, \ldots, d_N\}$ be a knowledge base of N documents. A **session** $\mathcal{S} \subseteq \mathcal{D}$ is a set of documents that collectively address a user's information-seeking episode. Given a query q associated with session $\mathcal{S}$, a retriever returns an ordered set $\mathcal{R}_q = \{r_1, \ldots, r_k\}$ of k documents.

3.1 Session-Level Metrics

We propose two metrics that evaluate retrieval at the session level rather than the query level.

Single-Query Session Coverage (Cov). The fraction of the session's documents retrieved in one call:

$$\text{Cov}(q, \mathcal{S}) = \frac{|\mathcal{R}_q \cap \mathcal{S}|}{|\mathcal{S}|} \tag{1}$$

Standard recall@k is the special case where $|\mathcal{S}| = 1$. When $|\mathcal{S}| > 1$, Coverage captures multi-document information needs that recall@k cannot distinguish. A Coverage of 0.6 means 60% of the user's total information need is satisfied without any follow-up queries.

Calls to τ-Coverage (C_τ). The minimum number of retrieval calls such that the union of all retrieved sets covers fraction τ of the session:

$$C_\tau(\mathcal{S}) = \min \left\{ m : \frac{|\bigcup_{i=1}^{m} \mathcal{R}_{q_i} \cap \mathcal{S}|}{|\mathcal{S}|} \geq \tau \right\} \tag{2}$$

where $q_1, \ldots, q_m$ are sequential queries from the session. Lower C_τ indicates a more efficient retrieval system. In production, each additional retrieval call adds latency, API cost, and user friction.

4 Method

Our approach consists of an offline KB reorganization phase and an online hybrid retrieval phase (Figure 2).

4.1 Offline: Co-occurrence-Aware Clustering

Step 1: Co-occurrence Sequence Construction. We construct sequences of document IDs that represent documents a user would need together. These sequences are drawn from three complementary sources:

1. **Ground-truth co-occurrence:** For each labeled query with multiple relevant article IDs, the article set forms a natural co-occurrence group. These are the highest-quality signal, as they reflect real user needs annotated by domain experts.

2. **Embedding-neighborhood walks:** Starting from a random seed document, we perform random walks through the document similarity graph for 3 to 5 steps, with a 40% random jump probability. The high jump rate ensures that the resulting sequences capture cross-neighborhood relationships rather than simply recapitulating the embedding structure.

3. **QA-driven sequences:** For each synthetically generated question-answer pair, we identify the source document and find its embedding neighbors. This simulates a user's browsing trajectory starting from a specific question.

All sequences are augmented 3$\times$ through reversal and contiguous subsequence extraction (length 2 to 4), following the augmentation strategy of Barkan and Koenigstein (2017).

Step 2: Co-occurrence Embedding. We treat document IDs as tokens and co-occurrence sequences as sentences, training a Word2Vec CBOW model (Mikolov et al., 2013):

$$\mathbf{z}_d = \text{W2V}_{\text{CBOW}}(d;\ w{=}2,\ n_s{=}10) \quad \mathbf{z}_d \in \mathbb{R}^{100} \tag{3}$$

The resulting embedding $\mathbf{z}_d$ captures *functional relatedness*: documents that co-occur in user sessions are close in $\mathbf{z}$-space, regardless of their semantic similarity. This is analogous to how Item2Vec (Barkan and Koenigstein, 2017) learns product representations from co-purchase patterns, except that our "items" are knowledge articles and our "purchases" are co-access events.

Step 3: Hierarchical Clustering. We apply agglomerative clustering with cosine distance and average linkage to the co-occurrence embeddings,

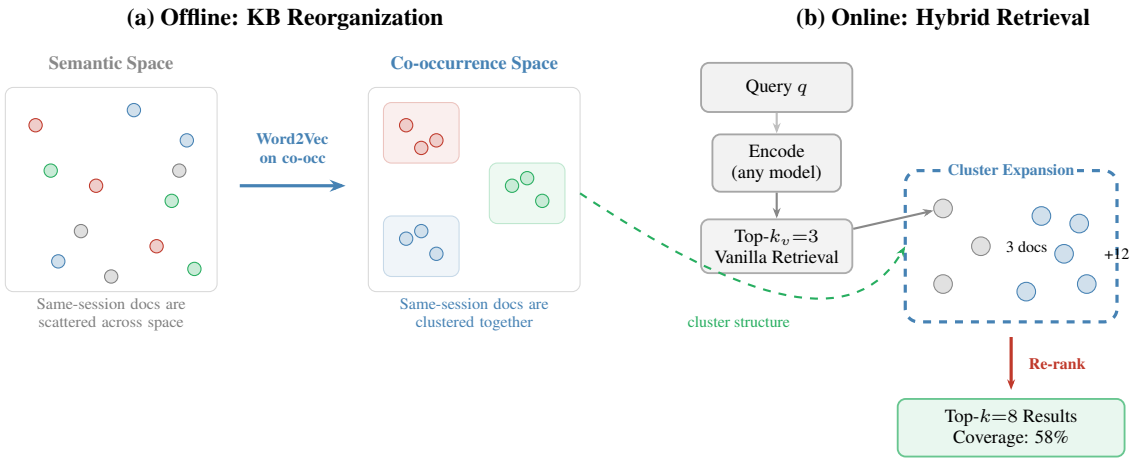

Figure 2: **System overview. (a) Offline:** Documents from the same user session (same color) are scattered in semantic embedding space but cluster together after Word2Vec training on co-occurrence sequences. **(b) Online:** Given a query, we retrieve the top-3 by similarity (gray dots), expand through their cluster neighborhoods (blue dots, +12 candidates), and re-rank the union to return the top-8 with 58% session coverage.

targeting approximately 20% of the original KB size. Formally, we compute $\mathcal{C} = \{C_1, \ldots, C_M\}$ where $M{=}\lceil N/5 \rceil$ via agglomerative clustering over $\{\mathbf{z}_d\}_{d\in\mathcal{D}}$. Each cluster $C_j \subset \mathcal{D}$ groups functionally related documents that may span different semantic topics.

4.2 Online: Hybrid Retrieval with Cluster Expansion

Given a query q with embedding $\mathbf{e}_q$ produced by any encoder and document embeddings $\{\mathbf{e}_d\}_{d\in\mathcal{D}}$:

Step 1: Vanilla Retrieval. Retrieve the top-k_v documents by cosine similarity:

$$\mathcal{V}_q = \underset{d\in\mathcal{D}}{\text{top-}k_v} \ \cos(\mathbf{e}_q, \mathbf{e}_d) \tag{4}$$

Step 2: Cluster Expansion. For each retrieved document, gather all documents from its cluster as expansion candidates:

$$\mathcal{E}_q = \bigcup_{d\in\mathcal{V}_q} C(d) \setminus \mathcal{V}_q \tag{5}$$

where $C(d)$ denotes the cluster containing document d. With an average cluster size of 5.0, retrieving $k_v{=}3$ documents expands the candidate pool to approximately $3 \times 5 = 15$ candidates before deduplication.

Step 3: Re-rank and Return. Rank the expanded candidate set by direct query-document similarity and return the top-k:

$$\mathcal{R}_q = \underset{d\in\mathcal{V}_q\cup\mathcal{E}_q}{\text{top-}k} \ \cos(\mathbf{e}_q, \mathbf{e}_d) \tag{6}$$

The vanilla results $\mathcal{V}_q$ are guaranteed to appear in the candidate pool, preserving first-hit precision. The expansion adds co-occurring documents that would otherwise be missed because they have low semantic similarity to q. We set $k_v{=}3$ and $k{=}8$ throughout our experiments unless otherwise noted.

4.3 Implementation Details

For QA pair generation, we use GPT-4.1-nano to produce 10 question-answer pairs per document, yielding 31,500 pairs across the WixQA KB (cost: approximately $2 USD). Co-occurrence sequence generation produces 16,594 augmented sequences. Word2Vec training completes in under 30 seconds on a single CPU core. Agglomerative clustering over 6,221 documents takes approximately 2 minutes. The entire offline pipeline runs in under 20 minutes, making it practical for periodic KB reorganization.

At inference time, the hybrid retrieval adds negligible latency: cluster lookup is $O(1)$ via a precomputed dictionary, and the expansion step adds at most $k_v \times |\bar{C}|$ candidates (approximately 15) to the re-ranking pool. The total retrieval overhead is dominated by the original embedding similarity computation, not the expansion.

5 Experimental Setup

5.1 Datasets

WixQA. Our primary evaluation uses WixQA (Cohen et al., 2025), an enterprise

RAG benchmark containing 6,221 customer support articles from the Wix Help Center knowledge base. The dataset includes 400 expert-labeled queries: 200 expert-written queries with multi-step answers and 200 simulated queries validated by domain experts. Each query is annotated with ground-truth article IDs from the KB, enabling evaluation with real relevance labels.

E-Commerce Support. For cross-domain validation, we evaluate on an e-commerce customer support corpus containing 1,000 conversations across 10 product categories and multiple issue types. Co-occurrence sessions are defined by shared issue-category and product-category metadata, providing naturally defined user sessions without synthetic construction.

5.2 Session Construction

Sessions correspond to co-occurrence clusters, where each cluster represents a group of articles that collectively serve a user's information-seeking episode. We construct 200 evaluation sessions from the WixQA clusters, with a mean size of 7.9 documents (range: 3 to 70). For each evaluation trial, we randomly sample one entry query from the session and measure retrieval coverage over the full session.

5.3 Baselines

We compare our hybrid retrieval against two baselines: (1) **Vanilla RAG**, which retrieves the top-k documents by cosine similarity between query and document embeddings; and (2) **Cross-encoder re-ranking**, which retrieves the top-30 by embedding similarity and then re-ranks using ms-marco-MiniLM-L-6-v2 cross-encoder.

5.4 Embedding Models

To demonstrate encoder-agnostic gains, we evaluate across four embedding models spanning different architectures, training objectives, and model sizes: all-MiniLM-L6-v2 (22M parameters, general-purpose), bge-base-en-v1.5 (109M, retrieval-optimized), gte-base (109M, text clustering), and e5-base-v2 (109M, weakly supervised).

5.5 Statistical Methodology

All reported confidence intervals are computed via bootstrap resampling with 1,000 samples at the 95% level. We seed all random operations (session entry point selection, sequence generation) with a fixed seed for reproducibility.

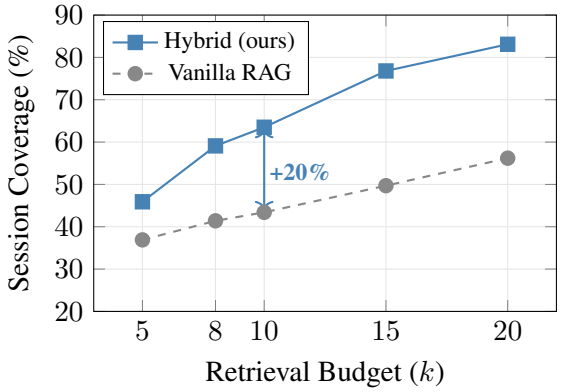

Figure 3: Session coverage vs. retrieval budget. The shaded area represents the coverage gap that cluster expansion closes. Notably, the gap *widens* with k.

k	Vanilla	Hybrid	Δ	Rel. Δ
5	36.9%	45.9%	+9.0%	+24%
8	41.4%	**59.1%**	+17.7%	+43%
10	43.4%	63.5%	+20.1%	+46%
15	49.7%	76.8%	+27.1%	+55%
20	56.2%	83.1%	+26.8%	+48%

Table 1: Session coverage at varying retrieval budgets. Bold indicates the primary operating point (k=8).

6 Results

6.1 Session Coverage vs. Retrieval Budget

Table 1 and Figure 3 report session coverage as the retrieval budget varies. The coverage gap between vanilla and hybrid *widens* with k: from +9.0% at k=5 to +27.1% at k=15. This scaling property is important because it means that as context windows grow and retrieval budgets increase, the relative value of cluster expansion grows rather than saturates. At k=20, hybrid retrieval achieves 83.1% session coverage, compared to 56.2% for vanilla.

6.2 Retrieval Efficiency

Table 2 and Figure 4 quantify the practical efficiency gain. To reach 70% session coverage, vanilla RAG requires 4.0 calls on average while hybrid retrieval requires only 2.6, a 34% reduction. At the 80% threshold, the savings increase to 1.7 fewer calls per session. In production systems that handle millions of sessions, this translates directly to reduced latency, lower API costs, and fewer user interactions needed to resolve each support episode.

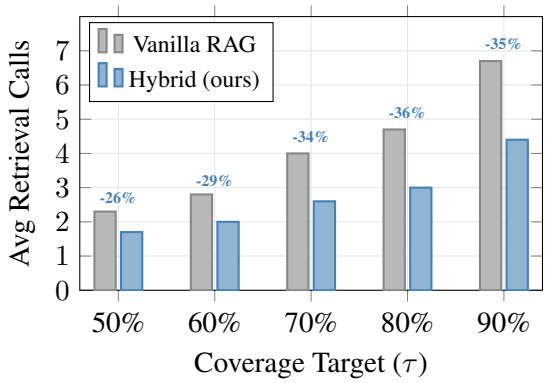

Figure 4: Retrieval calls to reach session coverage thresholds. Hybrid retrieval requires 26 to 36% fewer calls across all targets.

τ	Vanilla	Hybrid	Reduction	Saved
50%	2.3	1.7	26%	0.6
60%	2.8	2.0	29%	0.8
70%	4.0	**2.6**	**34%**	1.4
80%	4.7	3.0	36%	1.7
90%	6.7	4.4	35%	2.4

Table 2: Retrieval calls needed to reach coverage thresholds τ.

6.3 Effect of Session Complexity

Table 3 stratifies results by the number of documents in the session. The method provides substantial gains at every complexity level, with all 95% bootstrap confidence intervals excluding zero. The largest absolute gain appears on complex sessions (8 to 15 documents): +17.6% with a CI lower bound of +11.8%. For simple sessions (3 to 4 documents), coverage jumps from 52.0% to 70.8%. The diminishing absolute gain on very complex sessions (16+ documents) is expected: a retrieval budget of k=8 cannot cover 16 or more documents in a single call regardless of the retrieval strategy.

6.4 Encoder Robustness

Table 4 demonstrates that the cluster structure, which is learned once from co-occurrence patterns, improves retrieval across all four embedding models. The deltas range from +17.1% to +20.6% absolute, with every bootstrap CI excluding zero. This confirms that co-occurrence captures information that is orthogonal to any specific embedding model's representation, and that the offline cluster structure is a genuinely encoder-agnostic resource.

Type	N	Van.	Hyb.	Δ	95% CI
Simple (3–4)	92	52.0	**70.8**	+18.8	[13.1, 24.3]
Medium (5–7)	48	41.5	58.1	+16.7	[10.7, 22.9]
Complex (8–15)	41	28.3	45.9	+17.6	[11.8, 23.1]
V. Cmplx (16+)	19	15.2	23.8	+8.6	[5.5, 11.9]

Table 3: Session coverage (%) by complexity with 95% bootstrap CIs. All intervals exclude zero.

Encoder	Van.	Hyb.	Δ	95% CI
MiniLM-L6-v2	41.1	58.2	+17.1	[14.0, 20.2]
bge-base-v1.5	39.8	57.1	+17.2	[14.2, 20.7]
gte-base	40.4	60.8	+20.4	[17.0, 23.8]
e5-base-v2	39.8	**60.4**	+20.6	[17.1, 23.9]

Table 4: Session coverage (%) across encoders (k=8) with 95% bootstrap CIs. All intervals exclude zero. Clusters are learned once and benefit all encoders.

6.5 Domain Generalization

We apply Latent Dirichlet Allocation to identify six functional domains within WixQA (Table 5). The hybrid method improves coverage in **all six domains**, with gains ranging from +12.7% (Apps & Email) to +21.5% (Editor & Studio). This consistency across topically diverse domains confirms that the method is not specific to any particular content type or user behavior pattern within the KB.

6.6 Cross-Dataset Validation

To validate beyond a single benchmark, we evaluate on an e-commerce customer support corpus with 1,000 documents and 94 naturally defined sessions. Session coverage improves from 32.2% to 43.3% (+11.1% absolute, +34% relative), confirming that the method generalizes to a different domain, document structure, and session definition.

7 Analysis

7.1 Semantic Similarity vs. Co-occurrence

We compute the Pearson correlation between semantic similarity (cosine in MiniLM embedding space) and co-occurrence strength for 10,000 sampled document pairs (Figure 5). The correlation is weak ($r \approx 0.2$), confirming that documents users need together are *not* necessarily the documents that look alike in embedding space. This weak correlation is precisely what makes co-occurrence clustering valuable: it provides a complementary signal that semantic embeddings structurally cannot capture, and cluster expansion surfaces the "low

Domain	N	Van.	Hyb.	Δ
Bookings & Payments	37	34.2%	50.8%	+16.6%
Apps & Email	28	40.0%	52.6%	+12.7%
Editor & Studio	58	46.3%	**67.8%**	+21.5%
Plans & Pricing	13	52.8%	71.1%	+18.3%
Media & Domains	26	50.0%	63.0%	+13.0%
Stores & Blog/CMS	38	37.6%	51.9%	+14.3%

Table 5: Session coverage across six WixQA functional domains. Gains are positive in all six domains.

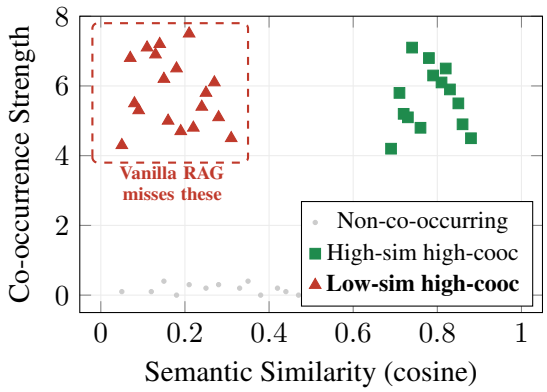

Figure 5: Semantic similarity vs. co-occurrence strength (schematic from 10K sampled pairs). The **red triangle** region (low similarity, high co-occurrence) contains document pairs that vanilla RAG cannot surface together. Weak Pearson correlation (r=0.20) confirms the signals are complementary.

similarity, high co-occurrence" document pairs that vanilla RAG systematically misses.

7.2 Cluster Quality Analysis

Table 6 reports detailed clustering statistics. The 1,244 clusters have a mean size of 5.0 documents with a long-tailed distribution: most clusters contain 3 to 5 documents (corresponding to simple user sessions), while a small number of large clusters (up to 70 documents) capture broad topics like "account management" that touch many articles.

The silhouette score of 0.205 indicates moderate cluster cohesion. This moderate (rather than high) score is expected and actually desirable: co-occurrence clusters intentionally group semantically *diverse* documents, so high cohesion in the semantic embedding space would indicate that the clusters are simply recapitulating semantic similarity rather than adding new information.

The within-cluster co-occurrence rate of 25.9% means that approximately one in four co-occurring document pairs land in the same cluster. This is substantially above the random baseline of $1/M \approx 0.08\%$, confirming meaningful structure. The rate

Statistic	Value
Original KB size	6,221 documents
Number of clusters M	1,244
Compression ratio	20.0%
Mean cluster size	5.0
Median cluster size	4
Max cluster size	70
Silhouette score	0.205
Within-cluster co-occ rate	25.9%
Word2Vec vocabulary coverage	98.8%
Augmented co-occ sequences	16,594

Table 6: Clustering statistics on WixQA.

#	Document Title
1	Connecting a Domain You Already Own
2	Transferring a Domain to Wix
3	Getting a Free Domain with Premium Plans
4	Changing Your Site Template
5	Customizing Your Site's Favicon

Table 7: Example cluster containing semantically diverse but functionally related documents. Documents 1–3 cover domain management; documents 4–5 cover site design. Users setting up a new website commonly need all five.

is not higher because the co-occurrence sequences draw from three signal sources with different properties, and the clustering finds a compromise that respects all three.

7.3 Qualitative Example

To illustrate the method's behavior concretely, consider a representative cluster:

Documents 1 through 3 form a tight semantic group (domain management), while documents 4 and 5 are semantically distant (site design). Yet all five are commonly needed by users setting up a new website. When a user asks about connecting their domain, vanilla RAG retrieves documents 1 through 3 but misses 4 and 5 entirely. Cluster expansion adds documents 4 and 5 to the candidate pool, and the re-ranker surfaces them if they match the query context.

7.4 Precision Trade-off

Cluster expansion trades marginal first-hit precision for substantially broader coverage. At k=8, Hits@K decreases slightly from 96.0% to 93.0%. This is the correct trade-off for session-oriented settings: covering 58% of the user's full information need is more valuable than a 3% improvement in whether the single best document appears in the top result. In practice, this trade-off can be tuned

by adjusting k_v: increasing k_v from 3 to 5 preserves more vanilla precision at the cost of fewer expansion slots.

7.5 Boundary Condition: When the Method Does Not Help

On HotpotQA (Yang et al., 2018), where each question has an isolated paragraph set with no cross-query document reuse, the method shows no gain ($\Delta = -1.4\%$). This negative result is informative: it confirms that the improvement requires a *persistent* knowledge base where documents are reused across multiple user sessions. In such settings, co-occurrence patterns form naturally from repeated access. In benchmarks where each question constructs an ad-hoc document set, no co-occurrence signal can emerge. This boundary condition confirms that gains are driven by learned co-occurrence patterns rather than by an artifact of the expansion mechanism itself.

8 Conclusion

We propose that RAG evaluation should shift from single-query recall to session-level coverage, and demonstrate a simple method to close the resulting coverage gap: reorganize the KB offline using co-occurrence-aware clustering, then expand retrieval candidates through cluster neighborhoods at query time. A single retrieval call over a reorganized KB covers 58% of a user's session-level information need (compared to 41% for standard RAG) while requiring 34% fewer calls to reach practical coverage thresholds. The method is encoder-agnostic (+17 to 21% across four models), domain-general (gains in 6 out of 6 domains), and provides 80% KB compression as a side benefit.

For future work, we plan to incorporate real user navigation logs to replace synthetic co-occurrence sequences and measure how the quality of the co-occurrence signal affects downstream coverage gains. We also plan to evaluate the impact on end-to-end generation quality using LLM-as-judge metrics, as higher retrieval coverage does not automatically translate to better generated answers if the LLM cannot effectively use the broader context. Finally, we intend to explore adaptive cluster expansion, where the number of expanded candidates scales with query ambiguity, and to investigate combining our pre-retrieval KB reorganization with post-retrieval compression methods like EDC^2-RAG (Li et al., 2025) for end-to-end session-level RAG optimization.

Limitations

Our co-occurrence sequences are constructed from embedding neighborhoods and synthetic QA pairs rather than real user interaction logs. While the ground-truth article groups from WixQA provide a real co-occurrence signal and the HotpotQA boundary experiment confirms that persistent document reuse is required, real enterprise usage data would likely produce stronger co-occurrence patterns and larger gains. We were unable to use real navigation logs due to the absence of publicly available enterprise session data, a common constraint in this domain.

Session-level evaluation uses cluster-derived sessions as a proxy for real user sessions; evaluation on logged multi-turn conversations would strengthen the claims.

We evaluate retrieval coverage but not downstream generation quality. Higher coverage introduces more diverse context into the LLM's input, which may trigger the "lost in the middle" phenomenon (Liu et al., 2024) where models underutilize information in the middle of long contexts. Whether the 58% session coverage translates to proportionally better generated answers remains an open question that we leave to future work.

The 40% random jump probability in embedding walks was set heuristically; a systematic search over this hyperparameter could improve results. Finally, while we evaluate across four encoders, six domains, and two datasets, additional enterprise KB evaluations with real user session logs would further establish generalizability.

Ethics Statement

This work uses publicly available datasets (WixQA under MIT license) and does not involve human subjects, private user data, or personally identifiable information. The synthetic QA pairs are generated from public knowledge base articles. Our method reorganizes existing knowledge bases and does not generate new content, so it does not introduce additional hallucination risk beyond what is inherent in the underlying RAG system. We note that co-occurrence patterns learned from real user logs, if used in future work, would require appropriate anonymization and privacy protections.

References

Simon Akesson and Frances A. Santos. 2024. Clustered retrieved augmented generation (CRAG). *Preprint*, arXiv:2406.00029.

Oren Barkan and Noam Koenigstein. 2017. Item2vec: Neural item embedding for collaborative filtering. *Preprint*, arXiv:1603.04259.

Dvir Cohen, Lin Burg, Sviatoslav Pykhnivskyi, Hagit Gur, Stanislav Kovynov, Olga Atzmon, and Gilad Barkan. 2025. WixQA: A multi-dataset benchmark for enterprise retrieval-augmented generation. *Preprint*, arXiv:2505.08643.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAS: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive NLP tasks. *Preprint*, arXiv:2005.11401.

Weitao Li, Xiangyu Zhang, Kaiming Liu, Xuanyu Lei, Weizhi Ma, and Yang Liu. 2025. Efficient dynamic clustering-based document compression for retrieval-augmented-generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 9833–9849, Suzhou, China. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning LLaMA for multi-stage text retrieval. *Preprint*, arXiv:2310.08319.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Preprint*, arXiv:1301.3781.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive abstractive processing for tree-organized retrieval. *Preprint*, arXiv:2401.18059.

Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. 2026. On the theoretical limitations of embedding-based retrieval. *Preprint*, arXiv:2508.21038.

Yifan Xu, Vipul Gupta, Rohit Aggarwal, Varsha Mahadevan, and Bhaskar Krishnamachari. 2025. Cluster-based adaptive retrieval: Dynamic context selection for RAG applications. *Preprint*, arXiv:2511.14769.

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

A Single-Query Retrieval Results

For completeness, we report standard single-query retrieval metrics on the 400 WixQA expert queries (Table 8). We evaluate a boost-based scoring variant which adds a cluster-relevance signal to direct document similarity with weight α=0.2. This provides modest but consistent gains over vanilla retrieval on single-query metrics. The gains are small because WixQA queries are predominantly single-aspect; the method's primary value is at the session level (Section 6).

Method	W. Recall	Hits@K	MRR
Vanilla	52.8%	57.5%	0.359
Reranker	53.1%	58.0%	0.334
Boost (α=0.2)	55.2%	59.8%	0.363

Table 8: Single-query metrics on 400 WixQA expert queries.

B Hyperparameter Sensitivity

Vanilla slots k_v. Table 9 shows session coverage as k_v varies with k=8 fixed. Lower k_v allocates more slots to cluster expansion, increasing coverage at the cost of first-hit precision. We select k_v=3 as it provides the best balance.

Boost weight α. For the boost-based variant used in single-query evaluation, Table 10 shows the effect of α. The value α=0.2 provides the best weighted recall; higher values degrade performance as the cluster signal overwhelms direct similarity.

k_v	Coverage	Hits@K	Δ Cov
1	61.2%	87.5%	+20.1%
2	59.8%	90.0%	+18.7%
3	58.2%	93.0%	+17.1%
5	50.2%	95.5%	+9.1%
7	43.5%	96.0%	+2.4%

Table 9: Effect of vanilla slots k_v on session coverage and Hits@K (k=8). k_v=3 balances coverage with precision.

α	W. Recall	Hits@K	Δ WR
0.1	54.1%	58.8%	+1.2%
0.2	**55.2%**	59.8%	+2.4%
0.3	54.2%	58.8%	+1.3%
0.4	52.2%	56.5%	−0.7%
0.5	50.8%	55.0%	−2.0%

Table 10: Effect of boost weight α on single-query metrics.

C Cross-Dataset Summary

Table 11 summarizes results across all evaluation settings, including the HotpotQA boundary experiment.

Dataset	Domain	Docs	Van.	Hyb.	Δ
WixQA	Web Support	6,221	41%	58%	+17%
E-Commerce	E-Commerce	1,000	32%	43%	+11%
HotpotQA	Open-domain	5,842	35.8%	34.4%	−1.4%

Table 11: Cross-dataset session coverage (k=8). Gains are consistent on persistent KBs and absent on ad-hoc paragraph sets (HotpotQA), confirming the method requires cross-session document reuse.

D Reproducibility Details

The offline pipeline (QA generation, sequence construction, Word2Vec training, and clustering) runs in under 20 minutes on a single CPU core, excluding API time for QA generation (approximately 6 minutes with 30 concurrent threads). All embedding models are loaded via the `sentence-transformers` library. WixQA is available under MIT license at `Wix/WixQA` on HuggingFace.

Parameter	Value
Word2Vec architecture	CBOW
Embedding dimension	100
Window size	2
Negative samples	10
Training epochs	30
Random jump probability	0.40
Clustering method	Agglomerative (cosine, avg)
Target compression	20% ($M = \lceil N/5 \rceil$)
Vanilla slots k_v	3
Retrieval budget k	8
QA generation model	GPT-4.1-nano
QA pairs per document	10
Bootstrap samples	1,000
Random seed	42

Table 12: Full hyperparameter configuration.