Hybrid Retrieval and Reranking in RAG: A Dual-Stage Approach to Improve Information Recall and Precision

Varadrajan Kunsavalikar

March 13, 2025

1 Abstract

Retrieval-Augmented Generation (RAG) systems rely on efficient retrieval mechanisms to fetch relevant information for downstream generative tasks. Traditional retrieval methods, such as BM25 (lexical matching) and dense embeddings (semantic retrieval via cosine similarity), have inherent limitations—BM25 struggles with semantic understanding, while dense retrieval often overlooks exact keyword matches. To address these challenges, we propose a hybrid retrieval approach that combines BM25 and cosine similarity-based dense retrieval to maximize recall and precision.

In our approach, we first retrieve candidate chunks separately using BM25 and cosine similarity. We then merge the unique chunks from both methods to create a diverse and enriched candidate pool. To further refine the selection, we employ a cross-encoder reranking model from the Sentence Transformers library, which evaluates query-chunk pairs more contextually, leading to improved relevance scoring. This additional reranking step significantly enhances true positive retrievals while reducing false positives.

Through rigorous experimentation, we have found that our Hybrid Retrieval + Reranking pipeline outperforms individual retrieval methods in precision, recall. Our results indicate that this dual-stage approach is particularly effective in improving retrieval quality in RAG-based applications, making it suitable for enterprise search, document processing, and conversational AI. This paper details the methodology, implementation, and key insights into optimizing retrieval strategies for modern AI workflows.

2 Introduction

2.1 Problem Statement & Importance of Retrieval in RAG

Retrieval-Augmented Generation (RAG) is a powerful technique that enhances generative AI models by incorporating external knowledge retrieval. It is widely used in applications such as **document search, legal and medical text retrieval, customer support automation, and enterprise AI systems**, where retrieving **accurate and contextually relevant information** is critical for downstream processing.

The success of a RAG system depends **not only on the generative model but also on the quality of retrieved information**. If irrelevant or incomplete data is retrieved, the generated responses will be unreliable. Thus, retrieval plays a fundamental role in ensuring that generative models have access to the most relevant knowledge before generating responses.

2.2 Challenges with Traditional Retrieval Methods

Traditional retrieval approaches can be categorized into **lexical retrieval** and **semantic retrieval**, each with its strengths and weaknesses:

- **Lexical Retrieval (e.g., BM25)**
  - Efficient for keyword-based queries.
  - Lacks semantic understanding, leading to **missed relevant documents** that use different wording.
- **Semantic Retrieval (Dense Embeddings + Cosine Similarity)**
  - Captures contextual meaning, enabling retrieval of conceptually similar documents.
  - May ignore **exact keyword matches**, leading to loss of critical information.

These limitations lead to **false positives (retrieved irrelevant data)** and **false negatives (missed relevant data)**, reducing the effectiveness of RAG-based applications.

2.3 Why Hybrid Retrieval is Necessary

Since neither lexical nor semantic retrieval alone is sufficient for all scenarios, **a hybrid approach is necessary** to maximize retrieval accuracy. By **combining both BM25 and dense retrieval**, we can:

- Improve recall by retrieving results from **both lexical and semantic search**.
- Capture both **keyword-specific and contextually relevant** documents.
- Reduce the risk of missing important information due to the weaknesses of a single retriever.

However, **combining two retrieval methods introduces new challenges**—we need a way to prioritize and rank the retrieved results effectively. This is where **reranking with a cross-encoder** becomes essential.

2.4 Introduction to Our Approach

To enhance retrieval effectiveness, we implemented a **Hybrid Retrieval + Reranking approach**:

1. **Hybrid Retrieval:**
   - Perform **BM25 retrieval** and **cosine similarity retrieval** separately.
   - Extract **unique** retrieved chunks from both methods.
2. **Cross-Encoder Reranking:**
   - Use a **cross-encoder model** from the Sentence Transformers library to **rerank** the results based on deeper contextual understanding.
   - Prioritize **true positive** matches while reducing **false positives**.

This approach ensures that only **the most relevant information** is retrieved and passed to the generative model, **improving retrieval quality in RAG workflows**.

3 Related Work & Background

3.1 Retrieval Methods Overview

3.1.1 Lexical Retrieval (BM25)

BM25 (Best Matching 25) is a ranking function based on term frequency-inverse document frequency (TF-IDF) weighting. It scores documents based on the presence of query terms using the following formula[2]:

$$\text{BM25}(D, Q) = \sum_{t \in Q} IDF(t) \cdot \frac{f(t, D) \cdot (k_1 + 1)}{f(t, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{\text{avgD}})} \tag{1}$$

where:

- $f(t, D)$ is the term frequency of term t in document D.
- $|D|$ is the document length.
- avgD is the average document length in the corpus.
- k_1 and b are hyperparameters controlling term frequency scaling and document length normalization.
- $IDF(t)$ is the inverse document frequency.

BM25 is efficient for **keyword-based retrieval** but lacks **semantic understanding**, leading to failure in retrieving conceptually similar content.

3.1.2 Dense Retrieval (Embedding-Based)

Dense retrieval relies on **transformer-based embeddings** to capture the semantic meaning of words and phrases. A common approach is to represent documents and queries in a high-dimensional vector space and compute similarity using cosine similarity [3]:

$$\cos(\theta) = \frac{\mathbf{q} \cdot \mathbf{d}}{\|\mathbf{q}\| \|\mathbf{d}\|} \tag{2}$$

where:

- $\mathbf{q}$ is the query embedding.
- $\mathbf{d}$ is the document embedding.
- $\cos(\theta)$ measures the similarity between the two vectors.

While **dense retrieval** improves upon lexical methods by capturing **semantic meaning**, it struggles with **exact keyword matches**, making it prone to **false negatives** in certain use cases.

3.1.3 Distance-Based Retrieval Techniques

Various **distance-based retrieval techniques** are used in high-dimensional vector spaces:

- **Euclidean Distance:** Measures the straight-line distance between vectors.
- **Manhattan Distance:** Computes the sum of absolute differences.

For **fast similarity search**, Approximate Nearest Neighbor (ANN) techniques like **FAISS**, **Annoy**, and **ScaNN** are employed to efficiently search large embedding spaces.

3.1.4 Cross-Encoders for Reranking

Cross-encoders evaluate query-document pairs jointly, providing **more precise relevance scores** compared to bi-encoders. Instead of retrieving independently scored documents, cross-encoders compute similarity by passing **both query and document together** through a transformer model:

$$\text{score}(q, d) = f([\text{CLS}, q, d]) \tag{3}$$

where f is a deep transformer-based model (e.g., BERT, RoBERTa), and the '[CLS]' token output is used for ranking.

Cross-encoders significantly improve **precision** but come at a **higher computational cost**, making them more suitable for **reranking retrieved candidates rather than full retrieval**.

3.2 Industry Practices & Existing Research

Several hybrid retrieval approaches have been explored in research and industry:

- **Multi-stage retrieval** in enterprise search (e.g., combining BM25 and dense retrieval in **Elasticsearch** and **FAISS**).
- Hybrid retrieval methods used in **OpenAI's search systems**, combining **keyword matching with embeddings**.
- **Dense Passage Retrieval (DPR)**, developed by Facebook, which trains a retriever jointly with a cross-encoder.

Our approach aligns with these industry best practices but **further optimizes retrieval precision by leveraging cross-encoder reranking**.

3.3 Comparative Analysis of Retrieval Methods

The following table compares different retrieval techniques in terms of accuracy, efficiency, and limitations.

Method	**Strengths**	**Weaknesses**	**Use Case**
BM25	Fast, keyword-matching	No semantic understanding	Structured documents
Dense Retrieval	Captures semantic meaning	Struggles with exact matches	Conversational AI
Hybrid Retrieval	Balances precision and recall	Computationally expensive	Enterprise search
Cross-Encoders	Highly precise rankings	High latency	Final-stage reranking

Table 1: Comparison of retrieval methods.

4 Methodology

4.1 Retrieval Strategy

4.1.1 Two-Stage Retrieval

Our hybrid retrieval approach combines **BM25** for lexical retrieval and **dense embeddings with cosine similarity** for semantic retrieval. This ensures that we retrieve both keyword-based matches and semantically similar content.

Stage 1: BM25 for Lexical Matching BM25 is applied to retrieve candidate chunks that contain exact or near-exact keyword matches. As discussed in Section 3, BM25 is an improved form of TF-IDF and assigns scores based on term frequency and inverse document frequency.

Stage 2: Dense Retrieval with Cosine Similarity We leverage transformer-generated dense embeddings to perform semantic retrieval using cosine similarity. However, to avoid magnitude bias in vector similarity calculations, **we normalize all embeddings**, ensuring that similarity scores are only influenced by vector direction and not their magnitude. This prevents highly confident but irrelevant embeddings from dominating retrieval.

Selecting Unique Chunks Once both retrieval methods have independently returned their top results:

- We extract **unique** chunks from both BM25 and dense retrieval results.
- If a chunk appears in both retrieval methods, we retain only one using chunk ids metadata.
- This set ensures we do not introduce duplicates while maximizing recall.

4.2 Reranking with Cross-Encoder

4.2.1 Need for Reranking

Despite using hybrid retrieval, **false positives** can still occur:

- BM25 retrieves keyword-matching documents that might not be contextually relevant.
- Dense retrieval retrieves semantically similar documents, but some may be loosely related.

To refine the retrieved results, we apply **cross-encoder-based reranking**.

4.2.2 Cross-Encoder Implementation

We employ the `cross-encoder/ms-marco-MiniLM-L6-v2` model from the Sentence Transformers library. Unlike bi-encoders, cross-encoders process **query-document pairs together**, generating **a single, refined relevance score**. This model is computationally expensive, so we only apply it to **rerank the unique chunks** rather than the entire corpus[6].

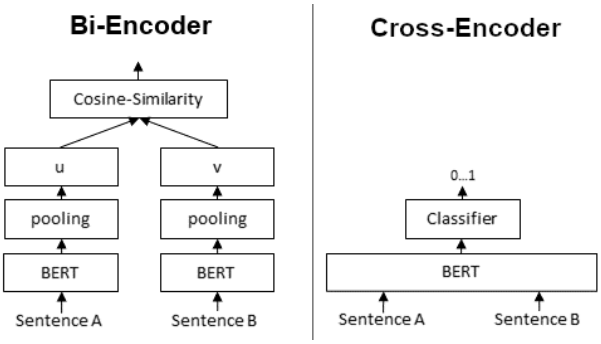

Figure 1: Bi-encoder vs Cross-encoder Working

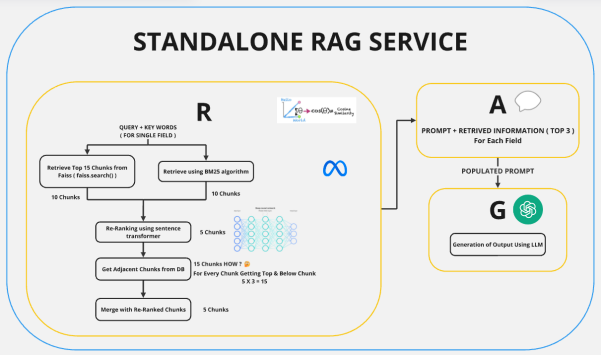

Figure 2: Implemented Methodology

Rescoring Mechanism

1. The cross-encoder takes the unique candidate chunks and assigns a new relevance score.
2. Chunks are **re-ranked based on these scores**, prioritizing **true positives**.
3. The final top-k results are sent to the RAG pipeline for further processing.

4.2.3 Trade-offs and Considerations

While this reranking approach significantly improves retrieval precision, it comes with computational overhead. To balance performance, we:

- Limit reranking to the **top k** retrieved chunks instead of the full dataset.
- Optimize retrieval latency by using efficient indexing techniques.
- Consider future improvements such as **efficient knowledge distillation** to create a lightweight cross encoder.

This approach ensures that the **retrieval pipeline optimally balances recall and precision**, improving retrieval quality for RAG-based workflows.

5 Challenges & Limitations

While our hybrid retrieval approach significantly improves recall and precision, it also introduces certain challenges and limitations that must be considered.

5.1 Computational Cost of Cross-Encoder Reranking

The cross-encoder model used for reranking provides highly precise relevance scoring but comes at a **computational cost**. Unlike bi-encoders that independently generate embeddings for queries and documents, cross-encoders process each **query-document pair jointly**, making them computationally expensive, especially for large-scale retrieval.

- Applying reranking to **all retrieved candidates** would be infeasible due to latency constraints.
- To mitigate this, we limit reranking to the **top 20-50** retrieved chunks instead of processing the entire corpus.

5.2 Latency Trade-offs

Hybrid retrieval is inherently **slower than single-method retrieval**. Combining BM25 with dense retrieval increases processing time due to:

- **BM25 computation overhead**: While efficient, BM25 retrieval must be combined with additional filtering.
- **Dense retrieval vector search**: Searching in high-dimensional vector spaces requires approximate nearest neighbor (ANN) techniques.
- **Reranking delay**: Cross-encoder reranking adds additional processing time, making real-time applications more challenging.

Strategies such as **FAISS optimizations, retrieval caching, and indexing refinements** can help reduce these latency issues.

5.3 Challenges with Long Document Retrieval

One limitation identified is the handling of **long documents** where retrieved answers span **2-3 pages**. In such cases:

- Dense embeddings often fail to differentiate between **tables, paragraphs, and section headers**.
- BM25 may prioritize keyword matches without considering the **structural organization** of the document.

To address this, a **document layout detection mechanism** can be introduced:

1. **Structural Parsing**: Identify **paragraphs, tables, lists, and headers** within documents.
2. **Embedding Differentiation**: Embed tables separately from body text to ensure accurate retrieval.
3. **Pre-filtering in Retrieval**: Use document layout as a filter before hybrid retrieval to prioritize semantically structured chunks.

This improvement would ensure that **tabular data, structured information, and free-text content are retrieved appropriately**, enhancing the effectiveness of RAG in handling long-form responses.

5.4 Ambiguous Queries

The retrieval model sometimes struggles with **ambiguous queries** that lack clear context. For example:

- If a query contains vague terms (e.g., "billing process"), the system may retrieve **multiple related documents** without clearly identifying the most relevant one.
- BM25 tends to favor **term frequency**, potentially over-ranking **irrelevant sections**.
- Dense retrieval may rank **semantically similar but contextually different** chunks higher.

One possible solution is to integrate a **query disambiguation step** that:

- Expands the query using **synonym-based reformulation**.
- Uses **zero-shot reranking** to prioritize contextually relevant answers.

5.5 Future Considerations

Addressing these limitations will involve:

- Exploring **layout-aware embeddings** to distinguish between structured and unstructured text.
- Improving **retrieval speed** by optimizing index-based lookups.
- Enhancing **contextual awareness** by refining reranking models.

Despite these challenges, the hybrid retrieval approach provides a strong balance of recall and precision, making it a viable solution for RAG-based applications.

6 Business Impact & Use Cases

Our hybrid retrieval approach has significant implications across various industries and AI-driven applications. By combining lexical and semantic retrieval methods with cross-encoder reranking, we enhance the quality of retrieved information, making RAG-based systems more effective and reliable.

6.1 Enterprise Search

In large organizations, vast amounts of structured and unstructured documents exist across multiple repositories. Traditional keyword-based search engines often return incomplete or overly broad results. Our hybrid retrieval approach:

- Improves information retrieval by combining **BM25 (lexical matching)** and **dense retrieval (semantic understanding)**.
- Enhances **precision and recall** by eliminating false positives and retrieving more contextually relevant documents.
- Enables **faster decision-making** by providing more accurate search results in knowledge management systems.

6.2 Legal & Medical Document Retrieval

Legal and healthcare industries require high-precision document retrieval due to strict compliance and regulatory constraints.

- In **legal applications**, our approach retrieves relevant case laws, contracts, and precedents, ensuring high factual accuracy.
- In **medical applications**, it assists in fetching clinical guidelines, patient records, and research articles while maintaining compliance with regulatory requirements (e.g., HIPAA).
- The use of **document layout detection** ensures accurate extraction of tables and structured content, improving retrieval for legal and medical use cases.

6.3 Customer Support & Conversational AI

RAG-based chatbots and virtual assistants often rely on retrieval mechanisms to fetch relevant information before generating responses. Our approach enhances:

- **Chatbot accuracy**: Ensures retrieval of factually correct responses instead of hallucinations.
- **Context-awareness**: Retrieves answers based on **semantic meaning and keyword relevance**, leading to more natural and informative responses.
- **Scalability**: Supports large-scale knowledge base retrieval across industries like e-commerce, customer service, and technical support.

6.4 Financial, Compliance, & Risk Assessment Applications

Financial institutions and regulatory bodies require precise document retrieval to analyze compliance reports, fraud detection patterns, and investment research.

- **Regulatory compliance**: Retrieves policy documents, legal frameworks, and risk management reports efficiently.
- **Fraud detection**: Enables semantic search of fraudulent transaction records and suspicious activity reports.
- **Investment research**: Helps analysts fetch company filings, earnings reports, and news articles with high recall and precision.

6.5 Impact on Retrieval-Based AI Applications

Our hybrid retrieval approach improves RAG-based AI applications by:

- **Reducing False Positives**: The cross-encoder reranking step ensures that only **highly relevant** retrieved documents are passed to the generative model.
- **Enhancing Recall**: By merging BM25 and dense retrieval results, we retrieve a more comprehensive set of documents.
- **Supporting Multi-Modal Data**: The ability to distinguish between **text, tables, and structured content** improves retrieval effectiveness in document-heavy domains.

By improving retrieval accuracy and relevance, our approach enhances AI-driven workflows across multiple industries, ensuring that RAG-based models generate more precise and informative responses.

7 Conclusion & Future Work

7.1 Summary of Findings

In this work, we successfully implemented a **hybrid retrieval approach** that combines **BM25 for lexical retrieval** and **dense retrieval using cosine similarity**, followed by **cross-encoder reranking** to refine results. Our approach effectively improves the precision and recall of retrieved documents, making it a valuable enhancement for **RAG-based AI applications**.

Key takeaways from our research:

- Hybrid retrieval balances the strengths of BM25 (exact keyword matching) and dense embeddings (semantic similarity).
- Normalizing vector embeddings prevents magnitude bias, ensuring fair similarity scoring.
- Cross-encoder reranking significantly reduces false positives by refining retrieved candidates based on contextual meaning.
- Our approach is applicable across multiple domains, including **enterprise search, legal and medical document retrieval, customer support, and compliance monitoring**.
- A limitation observed in long document retrieval suggests the need for **document layout detection** to handle structured data differently.

7.2 Potential Improvements and Future Work

While our hybrid retrieval system enhances retrieval quality, there are areas for further improvement:

7.2.1 Fine-Tuning the Reranker on Domain-Specific Data

- The cross-encoder reranker can be fine-tuned on **domain-specific corpora** to improve contextual ranking.
- Future experiments can explore **adapting pre-trained rerankers** for specialized datasets, such as legal or medical text retrieval.

7.2.2 Exploring Retrieval Fusion Techniques

- Investigate **weighted hybrid retrieval**, where BM25 and dense retrieval contributions are dynamically adjusted based on query type.
- Implement **query-dependent retrieval strategies** that select the most suitable retrieval mechanism (lexical, semantic, or both) based on query characteristics.

7.2.3 Optimizing for Real-Time Applications

- Reduce the latency of hybrid retrieval by optimizing **vector search algorithms** (e.g., FAISS, HNSW-based ANN search).
- Introduce **knowledge distillation** to create a lightweight cross-encoder for faster reranking.

7.2.4 Handling Long-Form Documents with Layout-Aware Embeddings

- Introduce a **document layout detection step** before retrieval to differentiate between **paragraphs, tables, lists, and metadata**.
- Embed **structured (tabular) and unstructured (text) content separately** to improve relevance scoring.
- Use layout-aware embedding models to ensure accurate chunk retrieval when queries relate to structured content.

7.2.5 Incremental Learning for Adaptive Retrieval

- Implement **continuous learning mechanisms** to update retrieval embeddings based on new document trends.
- Explore **adaptive retrievers** that refine search results dynamically as user queries evolve.

7.3 Final Thoughts

The hybrid retrieval approach significantly enhances RAG-based systems by ensuring that retrieved content is both **semantically and contextually relevant**. While our method improves precision and recall, **future work will focus on scalability, efficiency, and domain-specific optimizations**. By refining retrieval fusion techniques, optimizing reranking strategies, and integrating layout-aware embeddings, we aim to further enhance the reliability and accuracy of retrieval-augmented generation in enterprise applications.

8 References

References

[1] S. Robertson and H. Zaragoza, "The Probabilistic Relevance Framework: BM25 and Beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009. Available: `https://www.staff.city.ac.uk/~sbrp622/papers/foundations_bm25_review.pdf`

[2] BM25 algorithm: `https://en.wikipedia.org/wiki/Okapi_BM25`

[3] Cosine Similarity algorithm: `https://en.wikipedia.org/wiki/Cosine_similarity`

[4] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. Yih, "Dense Passage Retrieval for Open-Domain Question Answering," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[5] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

[6] N. Reimers and I. Gurevych, "Sentence-Transformers: Multilingual Sentence Embeddings using BERT," 2019. Available: `https://www.sbert.net`.