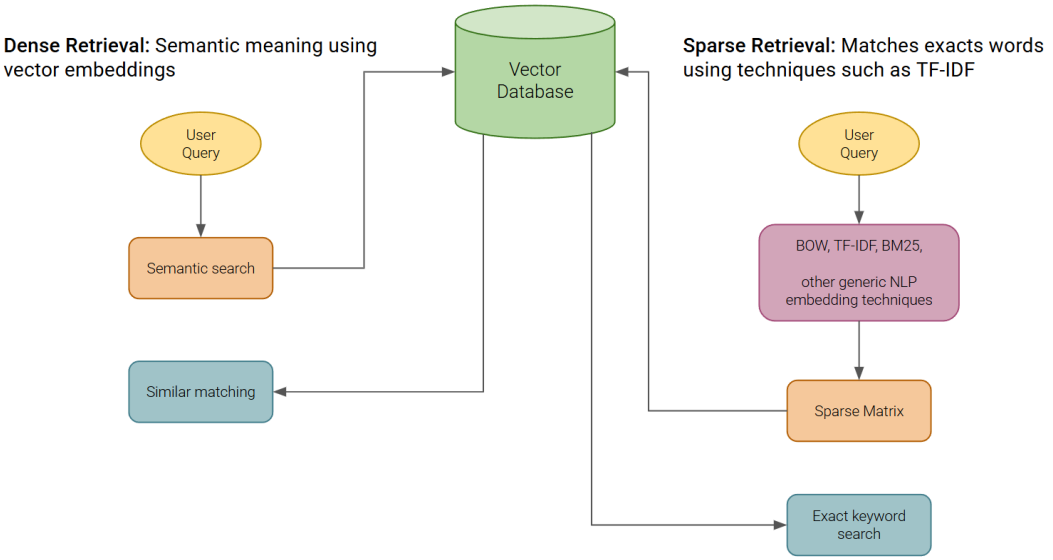
Hybrid Search Strategies

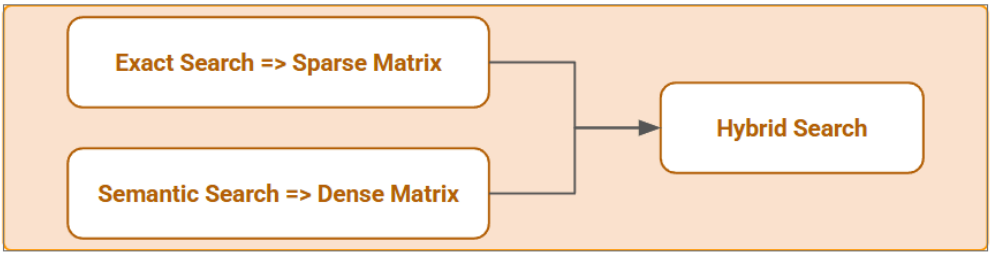
Combining Dense and Sparse Matrix

Hybrid retrieval combines both dense and sparse scores (e.g. using a weighted sum or learning-to-rank methods) to improve recall and relevance

You get the semantic power of embeddings, and the exact match of keywords - best of both worlds.



Doc ID	Content
D1	LangChain helps build LLM apps
D2	Pinecone is used for vector search
D3	The Eiffel Tower is in Paris
Query	Build application using LLM



$$Score_{hybrid} = (\alpha * Score_{dense}) + [(1 - \alpha) * Score_{sparse}]$$

$$Score_{dense} = Cosine\ Similarity\ (Input, VectorStore)$$

$$Score_{sparse} = TF - IDF\ (Input, Vector\ Representation)$$

$$\alpha = Weightage = 0.5$$

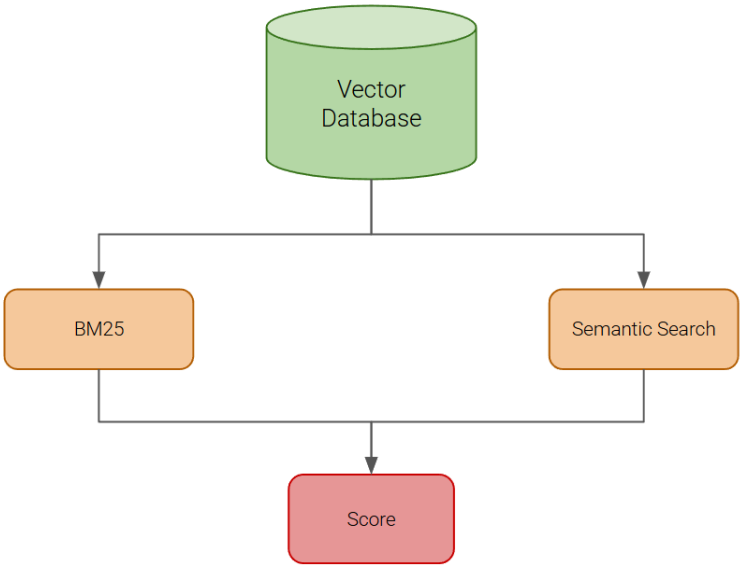
	Dense Embedding (Cosine Similarity)	Sparse Matrix (TF-IDF)
D1, Query	0.85	0.60
D2, Query	0.40	0.20
D3. Query	0.10	0.10

$$D1 = (0.5 * 0.85) + [(1 - 0.5) * 0.60] = 0.725$$

$$D2 = (0.5 * 0.40) + [(1 - 0.5) * 0.20] = 0.300$$

$$D3 = (0.5 * 0.10) + [(1 - 0.5) * 0.10] = 0.100$$

Benefits of combining Dense and Sparse Search



Benefits	Explanation
Boosts Recall	BM25 may catch exact keyword matches that dense retrieval misses; dense captures meaning even if keywords differ. Together , you minimize the risk of missing relevant documents
Handles Synonyms and Rephrasing	Semantic search can match "create app" with "build LLM system", even if there are no shared words. BM25 will catch the exact "LLM" and "app".
Improves Retrieval Robustness	Covers both users who search by specific terms (e.g., "LangChain agent") and those who use natural phrasing (e.g., "how do I use LangChain to talk to tools").
Supports Lexical Importance	BM25 scores rare keywords higher - this is crucial in technical/legal/medical contexts where a rare term (like "osteroporosis") should weight heavily.
Bridges Document Diversity	In large corpora (e.g., web [ages, PDFs, blogs), you often have a mix of well-structured and loosely written text. Hybrid retrieval adapts to both.
Easy to Tune via Weights	You can easily adjust the influence of each method: e.g., score = 0.7 * dense + 0.3 * sparse.
Helps with Misspellings or Variants	Dense models are more tolerant to typos and misspelled words (e.g., "LLM application"). BM25 may fail here.

When to use BM25 + Semantic Search

Use Case	Why Hybrid Retrieval Helps
RAG Pipelines	Prevents retrieval hallucination by ensuring both exact and fuzzy matches are considered.
Technical Documentation Search	Developers may search "how to use API" while the doc says "API usage" — BM25 and semantic together improve hit rate
Legal / Medical QA	Some queries require precise term matching (BM25), while others need general understanding (dense)
E-commerce / Product Search	"Cheap noise-cancelling headphones" could match "affordable" ANC earbuds" — dense helps; BM25 confirms "ANC"
Multilingual or Cross-lingual Retrieval	Semantic models can bridge language differences; BM25 ensures matching if terms are in the same language
Customer Support	Real users may type vague or keyword-heavy queries — hybrid improves retrieval reliability in chatbots / QA
Transcripts / Unstructured Data	Speech data or emails may contain inconsistent phrasing — dense retrieval picks meaning, sparse supports clarity

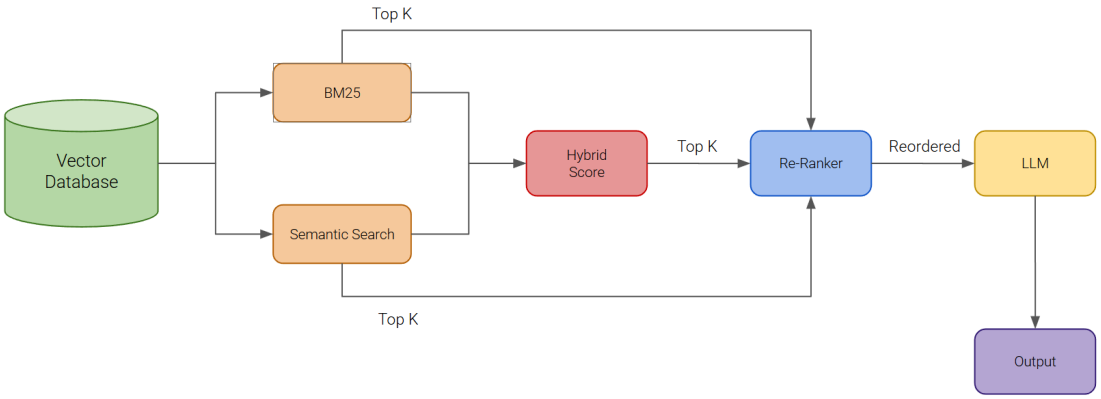
Re-Ranking Hybrid Search

Re-ranking is a second-stage filtering process in retrieval systems, especially in RAG pipelines, where we:

1. Use a fast retriever (like BM25, FAISS, hybrid) to fetch top-k documents quickly.

2. Use a more accurate but slower model (like a cross-encoder or LLM) to re-score and reorder those documents by relevance to the query.

It ensures that the most relevant document appears at the top, improving the final answer from the LLM.



Maximum Marginal Relevance

MMR (Maximum Marginal Relevance) is a powerful diversity-aware retrieval technique used in information retrieval and RAG pipelines to balance relevance and novelty when selecting documents

MMR select documents that are both:

1. Relevant to the query
2. Diverse from each other (non-redundant)

It prevents the retriever from returning very similar documents that repeat the same content.

$$MMR\,(d) \;=\; \lambda \;*\; Sim\,(d, q) \;-\; (1 \;-\; \lambda) \;* max_{s\,\epsilon\,S}\, Sim\,(d,\, s)$$

- q = Query
- Candidate document set (D)
- Select Document (s)
- Similarity function sim (a, b) => Cosine similarity
- Tunable parameter
    - Relevance to the q (higher λ)
    - Diversity (lower λ)

Problem

Query	How to use LangChain for RAG
Documents	
D1	LangChain enable retrieval with FAISS
D2	LangChain can use Chroma and Pinecone too
D3	LangChain agents can call external tools

Document	**Similarity Score**	
D1, D2	0.90	Redundant
D1, D3	0.30	Diverse
D2, D3	0.40	

1. Pick Document based on highest similarity score

Document	**Similarity Score**
D1	0.95
D2	0.93
D3	0.80

2. Select second document using MMR

   Compare D2 and D3 based on

   a) Relevance to the query

   b) Non-Redundancy with selected document (D1)

$$MMR\,(D2)\ = \lambda\ *\ Sim\,(D2, q)\ -\ (1\ -\ \lambda)\ * max_{s\,\epsilon\,S}\, Sim\,(D2,\ s)$$

$$MMR\,(D2)\ = 0.7\ *\ 0.93\ -\ 0.3\ *\ 0.90\ = 0.381$$

$$MMR\,(D3)\ = \lambda\ *\ Sim\,(D3, q)\ -\ (1\ -\ \lambda)\ * max_{s\,\epsilon\,S}\, Sim\,(D3,\ s)$$

$$MMR\,(D3)\ = 0.7\ *\ 0.80\ -\ 0.3\ *\ 0.30\ = 0.47$$

D1 —----------> Relevance to the query

D3 —----------> MMR Score

When to use MMR

1. In RAG, to avoid feed the LLM redundant information

   Result is more richer, more useful input or context to LLM

2. Building Chatbots like FAQs or Search App or Document Browser
3. When Retriever already return may results
4. MMR + Hybrid Retriever (Dense + Sparse)

When not to use MMR

1. Extremely short window context, we may just want top-1 most relevant
2. Need precision only, not focusing on coverage
3. Documents are already diverse, no need to enforce diversity
4. Using Reranker with LLM, Redundancy is handled by post filtering