Retrieval-Augmented Generation: Best Practices

Retrieval-Augmented Generation (RAG) combines retrieval systems with generative language models for accurate, grounded responses.

1. Document Ingestion Pipeline

Effective RAG systems begin with robust document processing. PDFs, web pages, and structured data must be parsed, cleaned, and chunked. Chunk size significantly impacts retrieval quality.

2. Embedding Strategies

Modern embedding models like text-embedding-3-small produce dense vector representations capturing semantic meaning far better than TF-IDF. Hybrid approaches combining dense and sparse retrieval often outperform either alone.

3. Vector Databases

ChromaDB, Pinecone, Weaviate, and Qdrant are popular choices. Key considerations: indexing speed, query latency, filtering, metadata support, and scalability.

4. Hybrid Search

Combining semantic search with keyword search (BM25) provides robust retrieval. Reciprocal Rank Fusion (RRF) merges results from different methods without score normalization.

5. Re-ranking

A re-ranking step after initial retrieval improves quality. Cross-encoder models score query-document pairs more accurately than bi-encoder similarity.

6. Prompt Engineering for RAG

Key elements: clear system instructions, formatted context with source attribution, instructions to only use provided context, and handling insufficient context cases.

7. Evaluation Metrics

Evaluate on: retrieval relevance, answer correctness, faithfulness, and latency. Tools like RAGAS provide automated evaluation.

8. Production Considerations

Production RAG requires: rate limiting, caching, monitoring retrieval quality, handling document updates, and managing embedding costs.