INO AI LAB

[AI EDUCATION • COURSE MATERIAL]

Building RAG Systems That Work

Retrieval-augmented generation in production

LEVEL	DURATION	LESSONS
Advanced	5 hours	9

[00]

Table of Contents

[LESSON 01]

Why Most RAG Demos Fail in Production

The classic RAG demo — embed your docs, retrieve top-k, paste into the prompt, ship — works on 100-doc demos and breaks on 100,000-doc production corpora. Recall drops, irrelevant chunks pollute context, costs balloon, and users lose trust. The 'easy RAG' tutorials of 2023 are wrong for 2026 production at scale. Real systems require thoughtful chunking, hybrid search, reranking, and grounded answer generation.

Start by measuring what you have. Build a labeled evaluation set of 50–200 question-answer pairs with ground-truth source documents. Measure retrieval recall@k and answer faithfulness on this set before touching the pipeline. Without this baseline, every 'improvement' is a guess. With it, you can ship changes that you know move the needle and roll back the ones that don't.

// KEY TAKEAWAYS

- Easy RAG tutorials don't scale.
- Build a labeled eval set before tuning.
- Measure retrieval recall AND answer faithfulness.

[LESSON 02]

Chunking Strategies That Survive Real Documents

Naive fixed-size chunking (e.g., 512 tokens) destroys semantic boundaries — half-sentences, split tables, orphaned headers. Use semantic chunking that respects document structure: chunk per heading, per paragraph, or per logical unit (function, contract clause, FAQ entry). For long technical docs, add the parent heading hierarchy to each chunk's metadata so retrieval has structural context.

For very large documents, use hierarchical chunking: small chunks for retrieval, larger parent chunks for context. Embed the small chunks, retrieve them, then fetch their parents for the LLM. This dramatically improves answer quality on complex questions while keeping embedding costs low. LlamaIndex and LangChain both implement this pattern as 'parent document retriever' — well worth the few hours to set up.

// KEY TAKEAWAYS

- Respect document structure, not fixed sizes.
- Add heading hierarchy as metadata.
- Use parent-document retrieval for big docs.

[LESSON 03]

Hybrid Search and Reranking

Pure vector search misses keyword matches; pure keyword search misses paraphrases. Hybrid search (BM25 + vector, fused via Reciprocal Rank Fusion) consistently beats either alone by 15–30% on real workloads. Most modern vector DBs (Qdrant, Weaviate, Vespa, pgvector with extensions) support hybrid natively in 2026. Turn it on by default and tune fusion weights per use case.

Add a reranker as the final step. Take the top 50 hybrid results, rerank with a cross-encoder (Cohere Rerank 3, Voyage rerank-2, BGE reranker), and keep the top 5–8 for the LLM. Rerankers add 50–200ms and dramatically improve precision. The full stack — chunking → hybrid search → reranking → grounded generation — is the 2026 baseline. Anything simpler underperforms in production.

// KEY TAKEAWAYS

- Hybrid search beats vector-only by 15–30%.
- Always add a reranker for final precision.
- Full stack: chunk → hybrid → rerank → generate.

[LESSON 04]

Grounded Answers, Citations, and Hallucination Control

Force the model to cite. Every claim in the answer should include a chunk ID or document reference. Build a post-processor that checks every citation actually appears in retrieved context — claims without grounding get flagged or stripped. This single discipline kills 80% of RAG hallucinations and makes the system trustworthy enough for legal, medical, and financial use.

Add a 'don't know' path. Instruct the model: 'If the context doesn't contain a clear answer, say so. Do not guess.' Pair with an LLM-as-judge that scores answer faithfulness against retrieved chunks. Anything below your threshold returns 'insufficient information' instead of a hallucinated answer. Users tolerate 'I don't know' far better than confident wrong answers — design for that reality from day one.

// KEY TAKEAWAYS

- Citations + verification kill 80% of hallucinations.
- Build an explicit 'don't know' path.
- Use LLM-as-judge to score faithfulness.

[NEXT]

Keep Going

You've completed this course material. The real learning starts when you apply what you've read. Pick one idea from this PDF and run a small experiment this week. Document what worked and what didn't. Share your findings with the community.

Explore more free courses, daily AI tips, and curated tools at:

innovationailab.com

Have feedback or want to suggest a topic? We read every message.

hello@innovationailab.com

— Innovation AI Lab Team —

// part of LumiLife Tech