Pune AI Community
2,662 followers
18h • Edited • 🌐

Follow ···

🤯 Retrieval Augmented Generation (RAG) gives the power to use your documents with linguistic prowess of a Large Language Model. But once you move past the basics, you realize there are dozens of strategies, each with trade-offs.

At a high level, RAG has two phases. First is data preparation. We take documents, split them into meaningful chunks, embed them, and store them in vector databases or sometimes knowledge graphs. Second is the retrieval and generation phase, where a user question is embedded, matched against stored knowledge, and passed to an LLM as context so it can answer accurately.

The real challenge starts when you try to optimize this flow. Simple chunking and search rarely scale. That is why advanced strategies matter.

➡️ Re-ranking helps reduce noise by pulling many results first, then narrowing them down using a specialized model.

➡️ Agentic RAG lets the AI decide how to search, whether by semantic similarity or full document reads.

➡️ Knowledge graphs shine when relationships matter more than raw text similarity.

➡️ Other techniques push things further. Contextual retrieval enriches each chunk with document-level meaning.

➡️ Query expansion and multi-query RAG increase recall by asking better or multiple questions.

➡️ Context-aware and late chunking improve how documents are split, preserving structure and meaning.

➡️ Hierarchical RAG balances precision with broader context.

➡️ Self-reflective RAG adds a feedback loop. Fine-tuned embeddings adapt similarity to your domain, even sentiment.

Here is the key takeaway. The best systems rarely rely on one strategy. In practice, combining three to five approaches delivers the best results 🚀 Re-ranking, agentic RAG, and context-aware chunking are a strong starting point.

Now the uncomfortable question. Are we over-engineering RAG systems instead of fixing poor data quality at the source? 🫣

#ArtificialIntelligence #RAG #AIEngineering #LLMOps #GenAI #CommunityLearning 🤖📚