Retrieval-Augmented Generation (RAG)

From Static Models to Knowledge-Aware AI Systems

Conceptual Overview
Machine Learning / Applied AI

Why Are We Talking About RAG?

Traditional ML / LLM Problems

- Models **hallucinate**
- Knowledge is **static**
- No awareness of **your documents**
- Cannot cite sources
- Unsafe for legal, medical, policy use

“Sounds confident” ≠ “Correct”

Real-World Question

How do you build an AI system that:

- Answers **only from trusted documents**
- Does **not make up facts**
- Can be **updated without retraining**
- Works for **legal, policy, healthcare** domains?

The Idea Behind RAG

RAG = Search first, then generate

Instead of asking an LLM to *remember everything*,
we let it:

1. **Find relevant information**
2. **Answer using that information only**

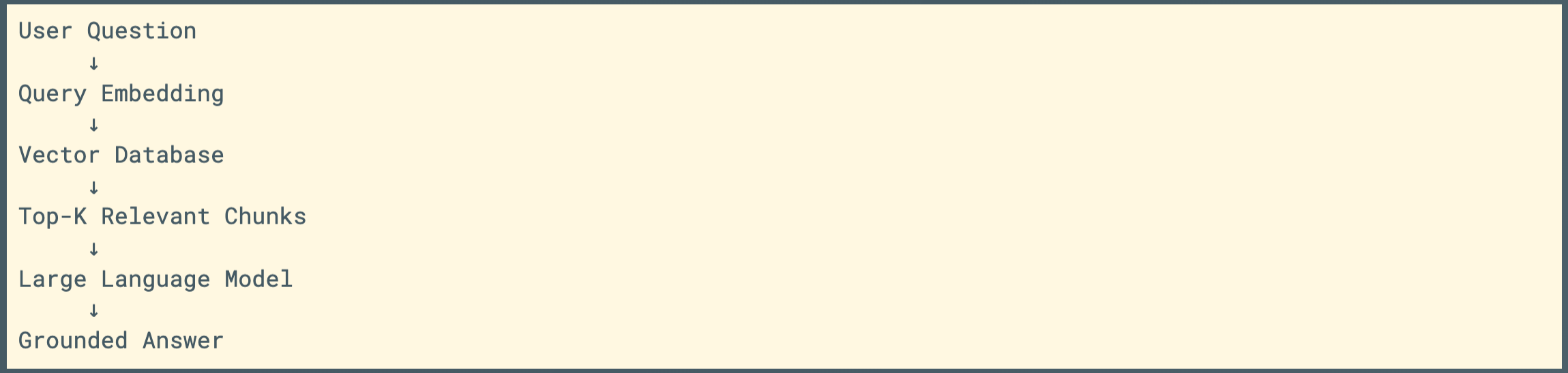
What Is RAG?

Definition

Retrieval-Augmented Generation (RAG) is an architecture that:

- Retrieves relevant documents from a knowledge base
- Injects them into the prompt
- Generates an answer grounded in retrieved data

High-Level RAG Architecture



Key Components of a RAG System

Component	Purpose
Documents	Source of truth
Chunking	Break large text
Embeddings	Convert text → vectors
Vector Database	Similarity search
LLM	Generate final answer

Role of Data in RAG

What Type of Data Works Best?

✅ Policies
✅ Laws & regulations
✅ Manuals & handbooks
✅ FAQs
✅ Research documents

❌ Live transactional data
❌ Rapidly changing numeric data
❌ Raw tables without structure

Why Chunking Is Necessary

LLMs and vector models:

- Cannot process very large documents
- Need **small, meaningful pieces**

Chunking helps:

- Preserve context
- Improve retrieval accuracy
- Reduce hallucinations

What Are Embeddings?

Intuition (No Math)

- Embeddings convert text into **numbers**
- Similar meaning → similar vectors
- Enables **semantic search**

Example:

- “Citizenship Act” ≈ “Nationality Law”
- “Weather” ≠ “Legal policy”

Why Vector Databases?

Traditional databases:

- Match exact keywords

Vector databases:

- Match **meaning**

Example:

- Query: *“Who qualifies for citizenship?”*
- Retrieved text:
  “Eligibility criteria for nationality…”

Where the LLM Fits In

Important rule:

The LLM **does NOT know your documents**

It only:

- Reads retrieved chunks
- Combines them into natural language
- Follows strict instructions

LLM = **Writer**, not **source of truth**

Grounded (RAG) vs Ungrounded AI

Ungrounded LLM	RAG System
Hallucinates	Uses documents
No citations	Source-based
Static	Easily updated
Risky	Auditable

Example Use Cases

- Legal assistant
- Policy Q&A system
- University handbook bot
- Healthcare knowledge assistant
- Company internal documentation search

When NOT to Use RAG

If you need:

- Real-time calculations
- Numerical forecasting
- Image generation
- Transaction processing

RAG is for **knowledge retrieval**, not reasoning alone.

Key Takeaway

RAG is a **knowledge-grounded AI system** which can "act" like chatbots

It combines:

- ML
- Information Retrieval
- Databases
- APIs
- Prompt design

What’s Coming Next

- Data preparation
- Chunking strategies
- Embeddings
- Vector databases
- Full RAG implementation
- FastAPI service

Next class:

Data & Chunking – where RAG succeeds or fails