

The need for retrieval in generative models



What's the Problem?

Generative models like GPT or T5 are trained on **huge static datasets**.

They learn to **generate** text from patterns in training data — but **they don't "know" anything beyond what they've seen**.

So they have 3 main **limitations**:

1. **Outdated Knowledge** – once trained, they can't access new facts or events.
 2. **Memory Constraints** – model parameters can't store every detail from training.
 3. **Hallucinations** – they might "make up" facts when unsure, because generation relies on learned probabilities, not direct retrieval.
-



The Core Need for Retrieval

To overcome these limitations, researchers introduced the idea of **retrieval in generative models** — that is, **augmenting generation with access to external knowledge** (like documents, databases, or the web).

This approach is known as:

| Retrieval-Augmented Generation (RAG)

The goal:

Combine **information retrieval** (from a knowledge source) + **text generation** (from the model).



How Retrieval Helps

Challenge	Without Retrieval	With Retrieval
Knowledge limitation	Model relies only on what it memorized	Model fetches relevant info from an external corpus
Outdated data	Model can't know recent events	Retrieval can include up-to-date documents
Factual accuracy	Often hallucinates details	Reduces hallucinations by grounding outputs in real text
Explainability	Hard to trace where info came from	Easier to show sources retrieved

Example

Without retrieval:

Prompt: "Who won the FIFA World Cup in 2022?"

Model (trained before 2022): "*I think Germany won.*" 

With retrieval:

The system retrieves a snippet from Wikipedia:

"Argentina won the 2022 FIFA World Cup after defeating France."

Then the generator outputs:

 "Argentina won the 2022 FIFA World Cup after defeating France in the final."

Here, retrieval provides *factual grounding* before generation.

How Retrieval-Augmented Generation (RAG) Works

1. Query Encoder

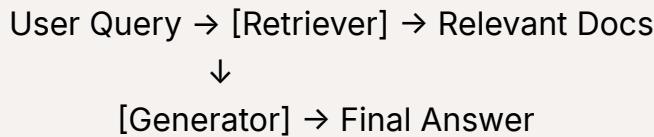
- Takes the user query and converts it into a vector (embedding).

2. Retriever

- Searches a document database (using embeddings similarity) for the **top-k relevant documents**.

3. Generator

- Feeds both the **query** and **retrieved documents** into a **generative model** (e.g., GPT, T5, BART) to produce a contextually accurate response.



Why Retrieval is Important (Summarized)

Benefit	Explanation
Grounded responses	Reduces hallucinations by basing answers on real data
Fresh knowledge	Keeps models up to date without full retraining
Transparency	Allows source attribution
Efficiency	Avoids cramming all facts into model weights
Adaptability	Model can specialize to new domains quickly by changing retrieval corpus

Example Systems

Model	Type	Key Idea
RAG (Facebook)	Encoder-decoder (BERT + BART)	Retrieves top documents and conditions generation
REALM (Google)	Pretraining with retrieval	Learns to retrieve during pretraining
RETRO (DeepMind)	Retrieval during training	Enhances transformer with retrieval memory
GPT + Vector DB (e.g., using LangChain)	Post-hoc retrieval	Combines LLM with external database like FAISS, Pinecone, or Chroma

Analogy

Think of a **retrieval-augmented model** as:

-  A smart student (the generator) who has learned language deeply
 -  but keeps a library (retriever) nearby to look up factual information when answering.
-

In Short

Retrieval in generative models bridges the gap between language fluency and factual grounding by allowing models to access external knowledge instead of relying solely on internal memory.
