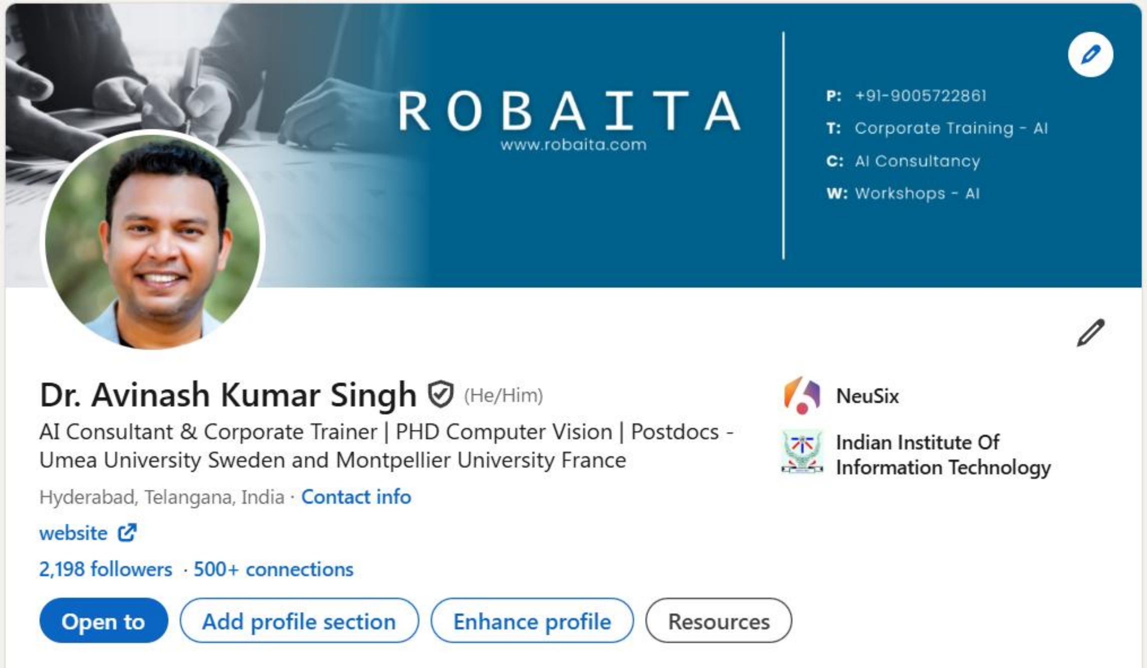
Retrieval Augmented Generation (RAG)

Dr. Avinash Kumar Singh

AI Consultant and Coach, Robaita

Dr. Avinash Kumar Singh

- **Possess** 15+ years of **hands-on expertise** in Machine Learning, Computer Vision, NLP, IoT, Robotics, and Generative AI.
- **Founded** Robaita—an initiative **empowering** individuals and organizations to **build, educate, and implement** AI solutions.
- **Earned** a Ph.D. in Human-Robot Interaction from IIIT Allahabad in 2016.
- **Received** postdoctoral fellowships at Umeå University, Sweden (2020) and Montpellier University, France (2021).
- **Authored** 30+ research papers in **high-impact** SCI journals and international conferences.
- Unlearning, learning, making mistakes …



https://www.linkedin.com/in/dr-avinash-kumar-singh-2a570a31/

Things to be discussed

- What is RAG
- Application of RAG
- Different Types of RAG
- How to build a RAG system - Demo
- Configurations and Challenges
- Questions & Answers

What is RAG

Retrieval Augmented Generation

- **Retrieval:** The system searches a knowledge base using a query.

  Example: Searching a vector DB for top chunks from resumes that matches the query.

- **Augmented:** The retrieved content is added to the prompt, enriching the LLM with real-time, external knowledge.

- **Generation:** The LLM uses the augmented context to generate a coherent, relevant, and factual response.

Retrieval Augmented Generation

Could you please tell me your knowledge cutoff date?

My knowledge is up to date as of **June 2024**. Let me know if you need information verified or updated beyond that—I'm happy to help search for the latest!

- **Proposed by:** Facebook AI Research (FAIR)
- **Introduced in:** 2020
- **Key Paper:** Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks [1]
- **Authors:** Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, Douwe Kiela
- **Motivation:** To overcome the limitations of closed-book language models in knowledge-intensive tasks by integrating a retriever component.

[1] NeurIPS 2020, https://arxiv.org/abs/2005.11401

Retrieval Augmented Generation

Why it is useful?

- ❑Keeps models up-to-date without retraining
- ❑Cost-effective and scalable
- ❑Ensures traceability and factual correctness

Retrieval Augmented Generation

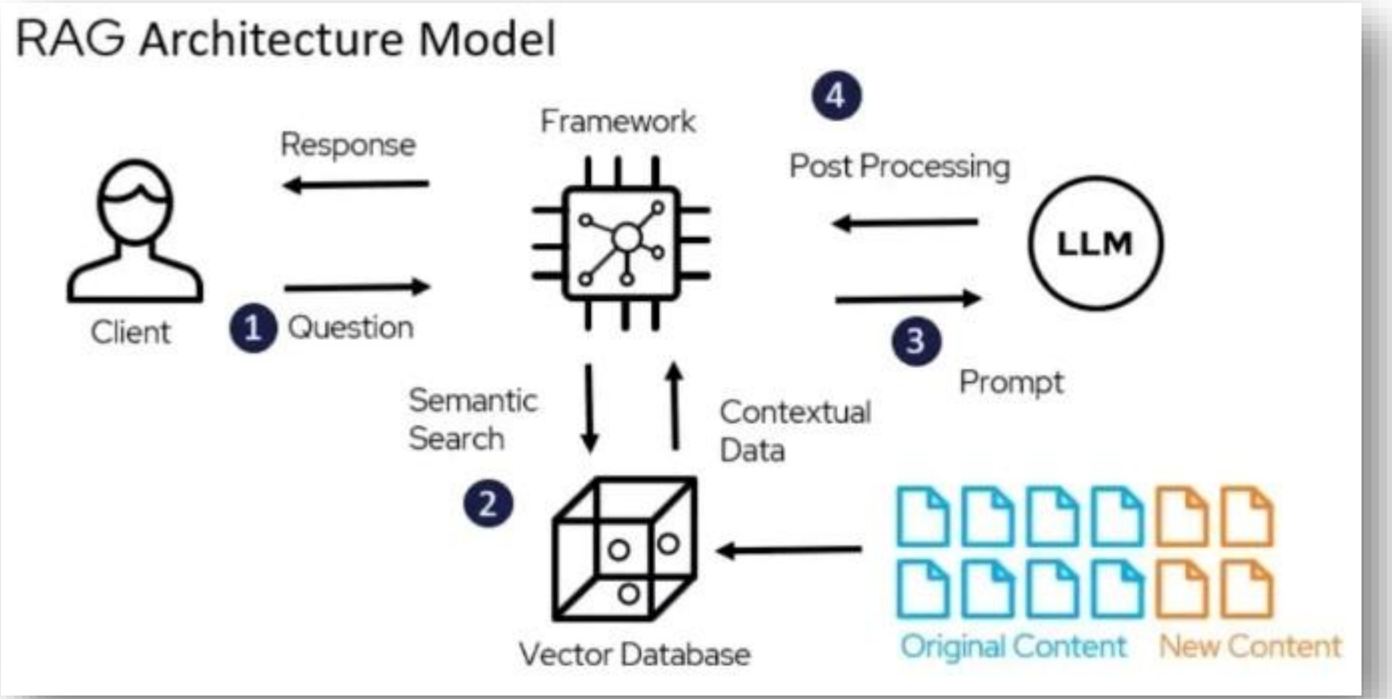
Applications

- ❑Enterprise Knowledge Assistants
  - ❑Internal document Q&A over policies, manuals, SOPs.
- ❑Legal Document Review
  - ❑Cases retrieval and summary generation from legal archives.
- ❑Healthcare Support
  - ❑Medical chatbot retrieving treatment guidelines and summarizing research.
- ❑Education and Research
  - ❑Academic assistant answering syllabus-based questions with citations.
- ❑E-commerce Search & Support
  - ❑Product search, reviews, and spec-based query response.

Retrieval Augmented Generation

Architecture

- ❑Input Query → Retriever (Vector DB) → Top-k Chunks → LLM → Output
- ❑Use of embeddings, similarity search, and context injection



Retrieval Augmented Generation Architecture

Training

Data Ingestion
Uploading documents in the system to be used for model training

Data Chunking
Split the training documents into small chunks so that we can have better information representation

Data Embedding
Create embedding of each chunk (representing the information in vectors)

Embedding Storage
Store embedding of each chunk into the vector databases for the faster and efficient retrieval

Testing

Find the most relevant matches

Query

Query Embedding
Get the embedding of your query. A vector

Top-K Results
Retrieve top k results (chunks) from the database based on similarity

LLM
Provide Query + Prompt + Results to generate response

Retrieval Augmented Generation
Key Components

- ❑Retriever: FAISS, Chroma, Pinecone
- ❑Embeddings: text-embedding-3-small (openai), intfloat/e5-base-v2 (hugging face), nomic-embed-text (Nomic AI)
- ❑Vector DB: Stores indexed chunks
- ❑Generator: GPT, llama, Deepseek, Mistral

Retrieval Augmented Generation

Chunking Strategies

- ❑RecursiveCharacterTextSplitter (LangChain)
- ❑Section-wise and semantic chunking
- ❑Ideal chunk size: 300-500 tokens

Retrieval Augmented Generation
Configuration: Chunking Strategies

- RecursiveCharacterTextSplitter (LangChain)
- Section-wise and semantic chunking
- Ideal chunk size: 300-500 tokens
- Text embedding
- Similarity parameters (top k)

Retrieval Augmented Generation

Evaluation & Accuracy

- ❑Metrics: Recall, Answer accuracy, Hallucination rate
- ❑Human feedback loop for refinement
- ❑Use of citations and confidence scores

Retrieval Augmented Generation

Tools

- ❑LangChain, LlamaIndex, Google Agent Builder
- ❑OpenAI, Hugging Face, SentenceTransformers
- ❑FAISS, Chroma, Pinecone
- ❑Additional Tools: Ollama, OpenWebUI

Retrieval Augmented Generation Types

- **<u>Standard RAG</u>:** Retrieve top-k relevant chunks from a vector DB and pass them as context to the LLM.
  - Example: Chatbot answering product-related queries from a PDF knowledge base.
- **Memory-Augmented RAG:** Incorporates past dialogue history into retrieval to maintain continuity.
  - Example: Customer support bot that remembers previous customer interactions.
- **Tool-Augmented RAG:** Combines RAG with function calling or external tool execution.
  - Example: AI assistant that retrieves documents and schedules meetings based on retrieved context.
- **Multimodal RAG:** Retrieves from multiple data types (text, image, audio) before generation.
  - Example: Customer support AI that fetches images of scanned bills and summarizes the findings.
- **Path-RAG:** Adds reasoning chains to retrieval, improving multi-hop or cause-effect queries.
  - Example: Academic assistant answering “What were the impacts of the 2008 crisis on Indian banking?”
- **<u>Light RAG</u>:** Minimalist RAG setup with a smaller retriever or rule-based fallback.
  - Example: FAQ bots using local keyword search before calling an LLM.

Thanks for your time