Refining the Pipeline

From Naive Retrieval to Contextual Understanding

Iverina Ivanova & Gemini 3 Pro

Goethe University

January 19th, 2026

Recap: The Standard RAG Pipeline

Last week, we learnt how to build a *Simple RAG*:

Load PDF $\rightarrow$ **Split** into chunks $\rightarrow$ **Embed** $\rightarrow$ **Retrieve** $\rightarrow$ **Generate**

Recap: The Standard RAG Pipeline

The Reality Check: While this works for simple FAQs, it often fails with long and complex texts.

The *Naive* RAG Failures

- It retrieves the wrong chunk.
- It misses the answer entirely.
- It gets confused by complex document layouts.

Today, we'll look at key challenges and outline concrete solutions to refine our RAG pipeline – Contextual RAG.

Challenge 1: Messy Input Data

The Input: A beautifully formatted PDF (columns, sidebars, tables).
The Machine's View: A chaotic stream of text.

Why this breaks RAG:

- Computers usually read files line-by-line, left-to-right. An issue occurs when a sentence spans across two columns.
- **Tables:** If you turn a table into a flat list of words, the relationship between **Row 1** and **Column 3** is lost.

Solution: Intelligent Document Processing (IDP).

- Tools like *Unstructured.io* use computer vision to separate the layout visually (just like your eyes do), before it starts reading the words.

Challenge 2: Advanced Chunking

How big should a chunk be?

Factors determining your strategy:

1. **The nature of the data:** Long texts (e.g., novels) or short texts (e.g., FAQ); complex (e.g., academic papers) or simple (e.g., recipes)?
2. **The Embedding Model:** How many tokens can it process at once?
3. **The User Query:** Broad summary questions or fact-finding questions?

Strategies beyond Fixed Size

- **Recursive Character Split:** Split by paragraphs first. If too big, split by sentences.
- **Semantic Chunking:** Don't split by character count. Split when the *topic* changes.

Chunk Size Selection for RAG Systems

Chunk Size	**Best For**	**Limitations**
Small **200–400 chars**	✓ QA pairs ✓ Definitions ✓ Specific facts	× Complex explanations × Narratives × Contextual depth
Medium **600–1000 chars**	✓ Paragraphs ✓ Technical docs ✓ Balanced retrieval	× Very specific queries × Very broad queries
Large **1500–2000 chars**	✓ Full context ✓ Summarization ✓ Complex topics	× Retrieval precision × Token limits × Computational cost

Key Principle

Chunk size should balance precision vs. context based on the content type and query complexity.

Challenge 3: Is Cosine Similarity Enough?

Imagine we split a financial report. We get this chunk:

> *"The company reported a 5% decline in revenue compared to the previous quarter."*

The Issue:

- **Which company?** The chunk doesn't say.
- **Which quarter?** The chunk doesn't say.

Challenge 3: Is Cosine Similarity Enough?

Imagine if the **user's query** were: **How did Google perform in Q3 2024?**
Chunk:

> *"The company reported a 5% decline in revenue compared to the previous quarter."*

This chunk will be ignored by the retriever because...

- The **chunk vector** represents a specific company (definite article), but its **antecedent** (Google) is in a previous chunk. The reference is unresolved.
- The **query vector** points towards **Google**, but the chunk vector points towards **Unnamed Corporate Entity.**

Solution: Contextual RAG

Concept: We must give the chunk its identity back.

The Workflow:

1. Before embedding, we use an LLM to read the **whole document** and generate a short summary.
2. We **prepend** this summary to every single chunk.

The Enhanced Chunk

[Context: This is from the Google Q3 2024 Financial Report.]
"The company reported a 5% decline in revenue..."

Result: Now the chunk contains the keywords **Google** and **Q3**. The retriever finds it!

Additional Optimization: Hybrid Search

The Problem: Vector search captures *meaning*, but sometimes we need *exact* words (e.g., a specific date **12.01.2026** or a name **Turing**).

The Solution: Combine two strategies.

1. Keyword Search

(The Ctrl+F Method)
Matches exact words
Good for: Dates, Proper Nouns, Specific Terminology
Bad for: Synonyms

2. Vector Search

(The Meaning Method)
Matches concepts
Good for: Captures semantic relations; Understands synonyms
Bad for: Exact precision

Hybrid Search merges these two lists to get the best of both worlds.

Hybrid Search: The Two Judges Vote

Query: *"What is scrambling in syntax?"*
Step 1:

Judge 1: Semantic Search

Looks for: **Meaning / Concepts**

The Votes:

- **Doc 5 (Score 0.90):** $\star\star\star$
  "Discusses word movement rules." (High match!)
- **Doc 3 (Score 0.50):** $\star$
  "Mentions transformations." (Okay match)

Judge 2: Keyword Search

Looks for: **Exact Words**

The Votes:

- **Doc 12 (Score 0.80):** $\star\star\star$
  Contains "scrambling" 3 times.
- **Doc 8 (Score 0.70):** $\star\star$
  Contains "scrambling" 1 time.

Hybrid Search: The Final Verdict

Step 2: We average the scores (50/50 split).
We want documents/chunks that make **BOTH** judges happy.

Doc	Meaning Score	Keyword Score	Final Average	Result
Doc 5	0.90	0.60	**0.75**	**Winner**
	(Great Meaning)	*(Okay Keywords)*		*Balanced*
Doc 12	0.70	0.80	**0.75**	**Winner**
	(Good Meaning)	*(Great Keywords)*		*Balanced*
Doc 8	0.00	0.70	**0.35**	Rejected
	(Irrelevant Topic)	*(Has Keyword)*		*False Positive*

From Knowledge to Agency

We have mastered **Knowledge Access** (RAG).

RAG gives the model **spectacles** *to read your library.*

Next Horizon: Agency
What if the answer isn't in a PDF? What if the answer requires calculation?

- **Agents:** Giving the model *hands*.
- Instead of just retrieving text, the model can:
    - Execute Python code (to calculate statistics).
    - Search the web in real time (for today's weather).
    - Use an API (to book a room).

Meta-Analysis

This slide deck was developed in a multi-step process involving **Gemini 3 Pro**.

1. **Contextual Prompting:** Defined the specific learning goals for the current session (moving from "Simple RAG" to "Production Challenges" like messy data and retrieval failures).
2. **Analogy Generation** Translating engineering concepts into intuitive metaphors (e.g., Hybrid Search as *Two Judges*).
3. **Human Verification & Correction:**
4. **Visual Refinements**