AI BASELINE

webinar • may 20th 2026

RELIABLE RETRIEVAL FOR AI AGENTS

"Where is the wisdom we have lost in knowledge? Where is the knowledge we have lost in information?"

— T.S. Eliot
The Rock, 1934

AI Factory Austria AI:AT

AI FACTORY AUSTRIA

together with

AI Baseline

Shaping the Future of AI

WHO WE ARE

XENIA

CEO · XENIA G

I led product development of the world's largest biomedical knowledge graph and information retrieval system at a drug discovery unicorn (BenevolentAI). Later, while building a multi-agent system for investigations to identify disinformation campaigns, it became obvious that existing search and retrieval methods are inaccurate, break down at scale, and are not ready to support agentic operations.

ALEX

CTO · ALEX C

My work in academia focused on how information relates and can be represented across space. I then went on to build retrieval infrastructure serving 4,000+ users in regulated sectors like finance and government — where I kept hitting the same walls with retrieval precision. AI Baseline is the back-end intelligence layer that enables fleets of AI agents to run knowledge workflows current solutions can't support.

AGENDA

01 | **Industry landscape & failure modes**

What teams are building today and what is breaking.

02 | **Deep Dive into existing approaches**

BM25, Hybrid, Agentic Search, Reranking and Graph Retrieval.

03 | **How to choose what to use?**

Persistent knowledge, provenance-aware systems, moving away from query-time compute.

TELL US ABOUT YOU

What is your role?

drop in the chat

Analyst | Data/AI | AI Engineer | CTO | a Wild Card


DAYS WITHOUT
0
0
A "RAG IS DEAD" POST

HOW IS YOUR RETRIEVAL DOING?

One word — where does retrieval/search break for you?

drop in the chat

accuracy | speed | hallucinations | cost | something else..

WHY RETRIEVAL BREAKS

01 **CROWDING.** Millions of documents make search spaces too dense. Everything looks “similar,” drowning the AI in high-scoring noise.

02 **CHRONOLOGICAL BLINDNESS.** Vectors don’t understand time — outdated documents get retrieved alongside current ones, forcing the LLM to guess the truth.

03 **SEMANTIC COLLISION.** Enterprise jargon overlaps. Without structural connections, the AI merges unrelated concepts that share the same name.

04 **CONTEXT SATURATION.** Brute-forcing the context window with massive text overwhelms the model and degrades its reasoning.

05 **CASCADING HALLUCINATIONS.** Fed crowded, contradictory, saturated text, the model doesn’t just fail — it confidently invents facts.

06 **GOVERNANCE FAILURE.** Naive search ignores access controls (RBAC), blindly retrieving restricted docs — a security and data-leak risk.

07 **RUNAWAY ECONOMICS.** Massive chunks and unconstrained agentic loops consume tokens exponentially, turning budgets into black holes.

BRIEF HISTORY OF SEARCH

1994

BUILT FOR HUMANS

BM25

Keyword ranking lands at TREC-3. Becomes the default search algorithm for ~20 years.

2013

BUILT FOR MEANING

WORD2VEC

Words become vectors. “Meaning as coordinates” turns semantic search practical.

2020

THE PIVOT

RAG & DPR

Dense retrieval fuses with generation. Retrieval becomes core LLM infrastructure.

2023

BUILT FOR MACHINES

HYBRID, GRAPH & AGENTIC

Agents plan, search and loop. The reader is no longer human.

2026

THE BREAK

RETRIEVAL IS THE BOTTLENECK

A human-era stack buckles under machine speed and scale. Time to rebuild the layer.

For 50 years we built search for humans — **keywords**, then **meaning**. Then the reader stopped being human: **agentic AI** reads at machine speed, and the human-era stack breaks beneath it.

CONTEXT

THE REFRAME

An LLM is not a thinking brain.
It is a statistical model that solves a calculation based entirely on what you put in its context window.

THE IMPLICATION

To get accurate output, you must isolate and input the right variables.
What we'll call "high-quality context" — the discipline of deciding what each slot contains.

THE CONTEXT FORMULA fig.01

Every word you provide acts as a variable in the model's equation.

OUTPUT = f (*context*)

↓ context is composed of ↓

SYS	**System instructions**	how the model should behave
RAG	**Retrieved data**	documents fetched at query time
HIST	**Chat history**	prior turns in the conversation
USER	**User query**	what's being asked right now

CONTEXT

The Trap of the Data Dump

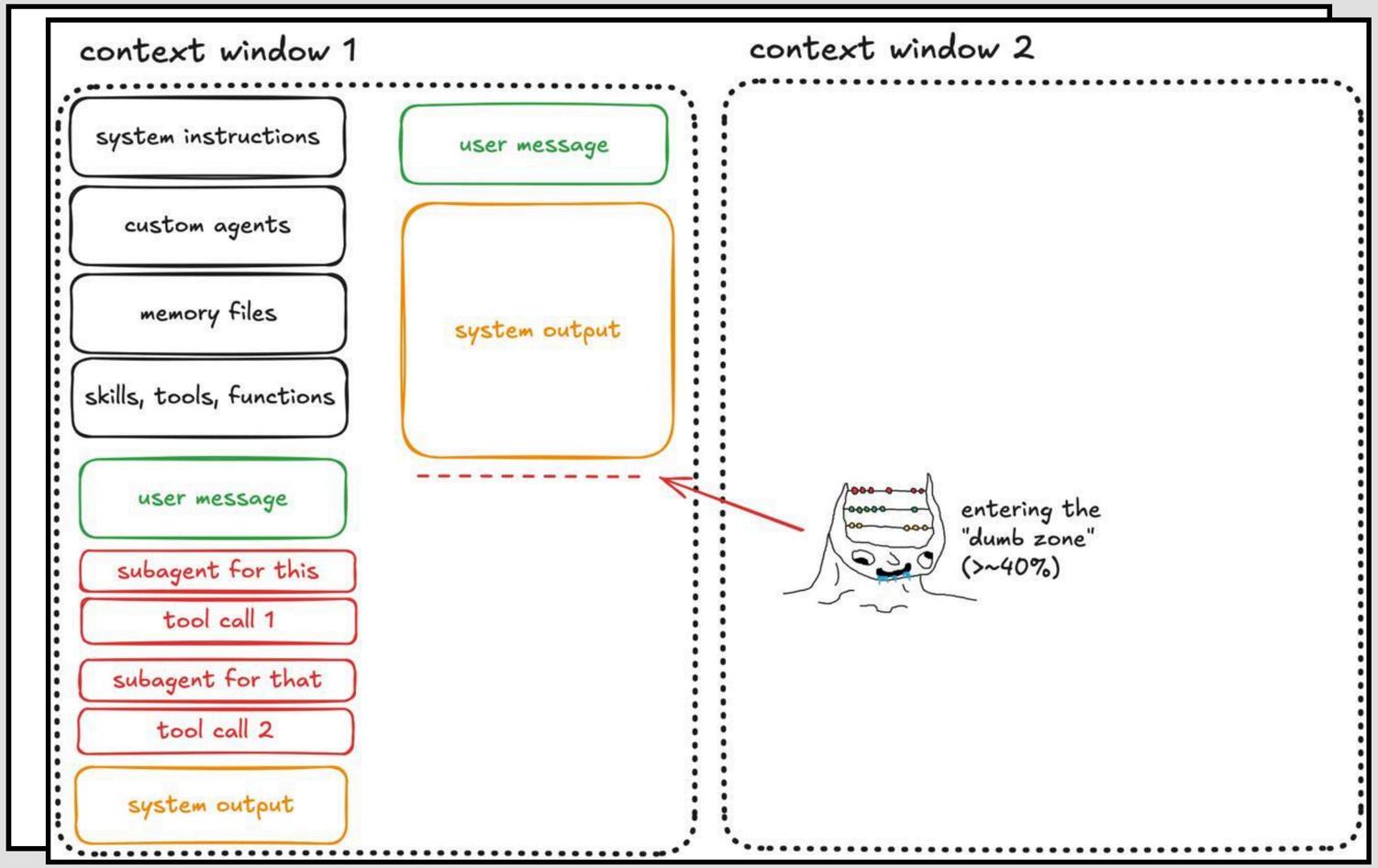
Large > 1 Million token context windows, tempt to drop a lot of context into an context window.

Structure is a Multiplier:

A clean, well-organized context window allows you to achieve expert-level results even with smaller models.

LLM are quite tolerant but:

Garbage In, Garbage out



VECTORS (OR EMBEDDINGS)

How LLMs "Understand" Language: A Primer on Vectors

01
TRANSLATION

Meaning as coordinates.
Any word's conceptual meaning is translated into a specific numerical coordinate — a vector — in high-dimensional space.

02
COMPARISON

Proximity is similarity.
Distance between coordinates measures meaning. Concepts with similar meanings sit physically close in vector space.

03
ARITHMETIC

Math with words.
Because relationships are geometric, you can literally do arithmetic on meaning.
King - Man + Woman = Queen.

04
COMPRESSION

Information density.
A single vector can hold one word — or compress the meaning of a whole sentence or document into one coordinate.

WHY THIS MATTERS

Every retrieval method we'll discuss next is built on these four properties. Understand vectors, and you understand why each method works the way it does.

VECTOR SEARCH

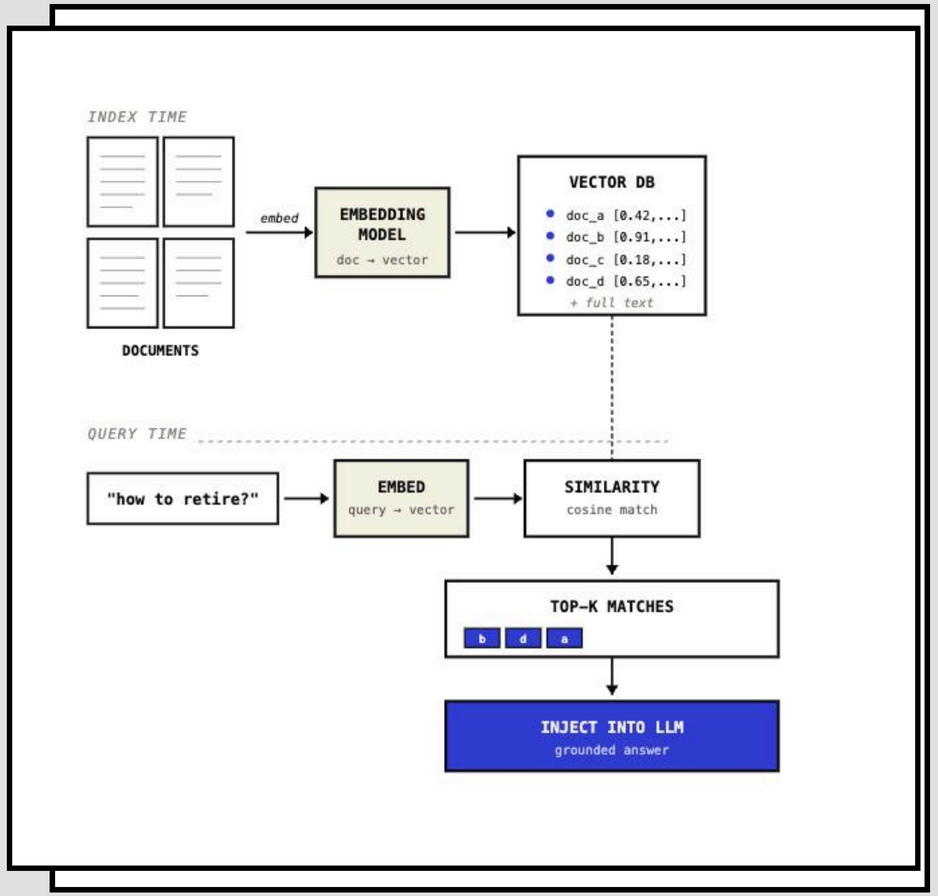
Semantic retrieval over dense vector embeddings

HOW IT WORKS

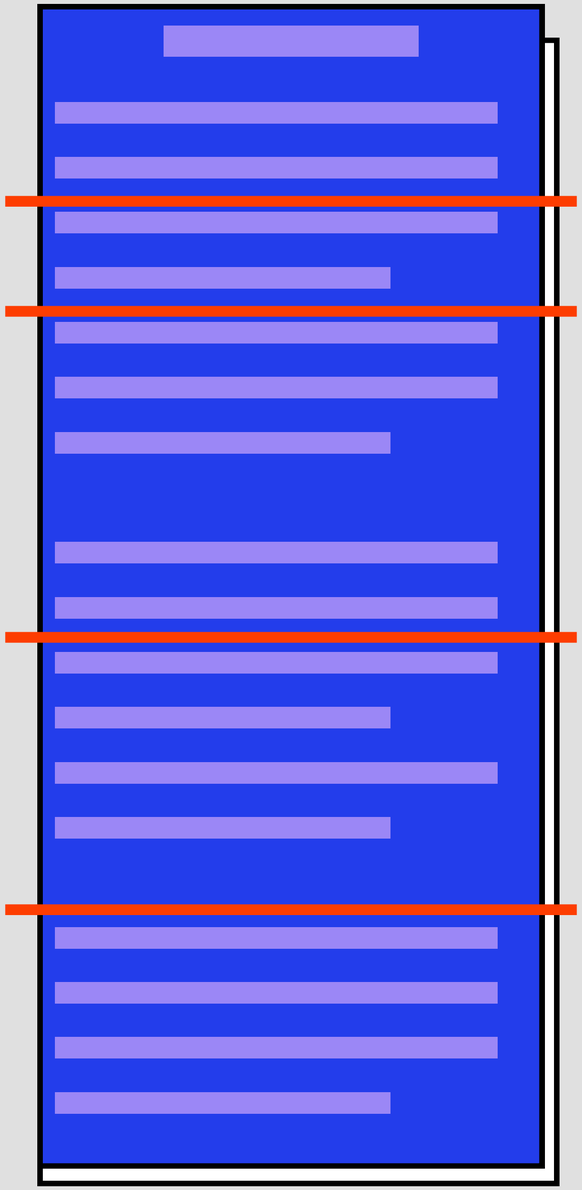
Each document is represented as a single semantic fingerprint — its vector. At query time, the user's question becomes a vector too, and the system returns documents whose vectors sit closest in semantic space.

FOUR STEPS

01 Semantic indexing.
Each document is embedded as a vector — its semantic fingerprint.
02 Vector database.
Vectors + full text stored in a specialized index.
03 Run-time query.
User's question becomes a vector and is matched against the index.
04 Context injection.
Top-matching documents are passed into the LLM's context.



CHUNKING



Strategy	How It Works	Pros	Cons	Best For
Fixed-Size	Slices text at strict token counts	Fast, simple, cheap	Ignores context	Quick prototypes, simple data.
Recursive	Splits text using punctuation	Keeps paragraphs and sentences intact.	Rigid boundaries, awkward splits for large elements	General Text
Semantic	Using embeddings to split text at topic shift	Captures complete ideas perfectly; high relevance.	Slow and expensive	Deep RAG applications, complex semantic search.
Document-Based	Using Structural document markers (Headings, html tags ...)	Matches the intended layout	requires clean and well formatted files	Markdown, FAQs, structured PDFs*, code.
Agentic	LLM reads the text and splits it	Unmatched flexibility	Even slower and more expensive	High-stakes RAG, messy/highly unstructured data.

They all have a specificity vs context tradeoff
All chunking methods risk that a chunk loses the bigger context

BM25

While vectors are incredible at finding similar concepts, they sometimes fail at finding exact terms. That's where BM25 comes in.

Literal matching

A search for the specific error code ERR_709_XYZ — vector search might return documents about "general system failures." BM25 matches the string symbol by symbol.

Term frequency

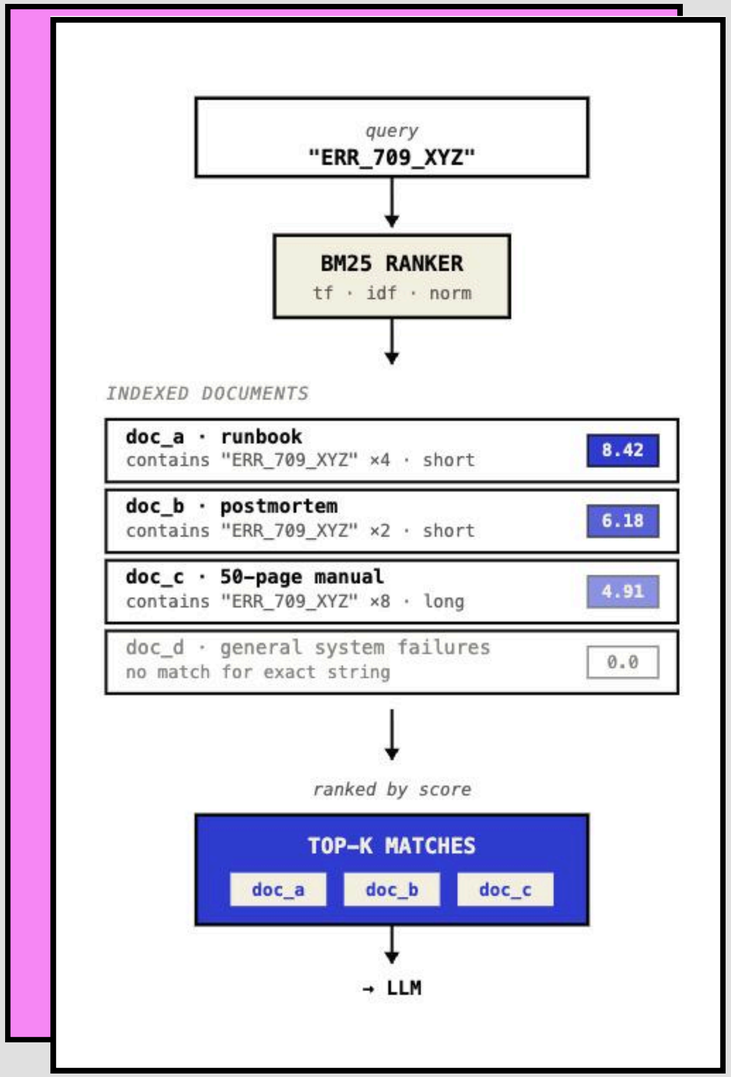
The more times your search word appears in a document, the higher that document scores — with an upper limit so saturation doesn't dominate.

Rarity is rewarded

BM25 assigns higher weight to rare words and effectively ignores common ones. "The" counts for almost nothing; "401(k)" counts a lot.

Document length normalization

A 50-page manual shouldn't win just because it has more words. Short but highly relevant documents can still rank first.



HYBRID

Combining sparse (BM25) and dense (Vector) retrieval

THE COMBINED APPROACH

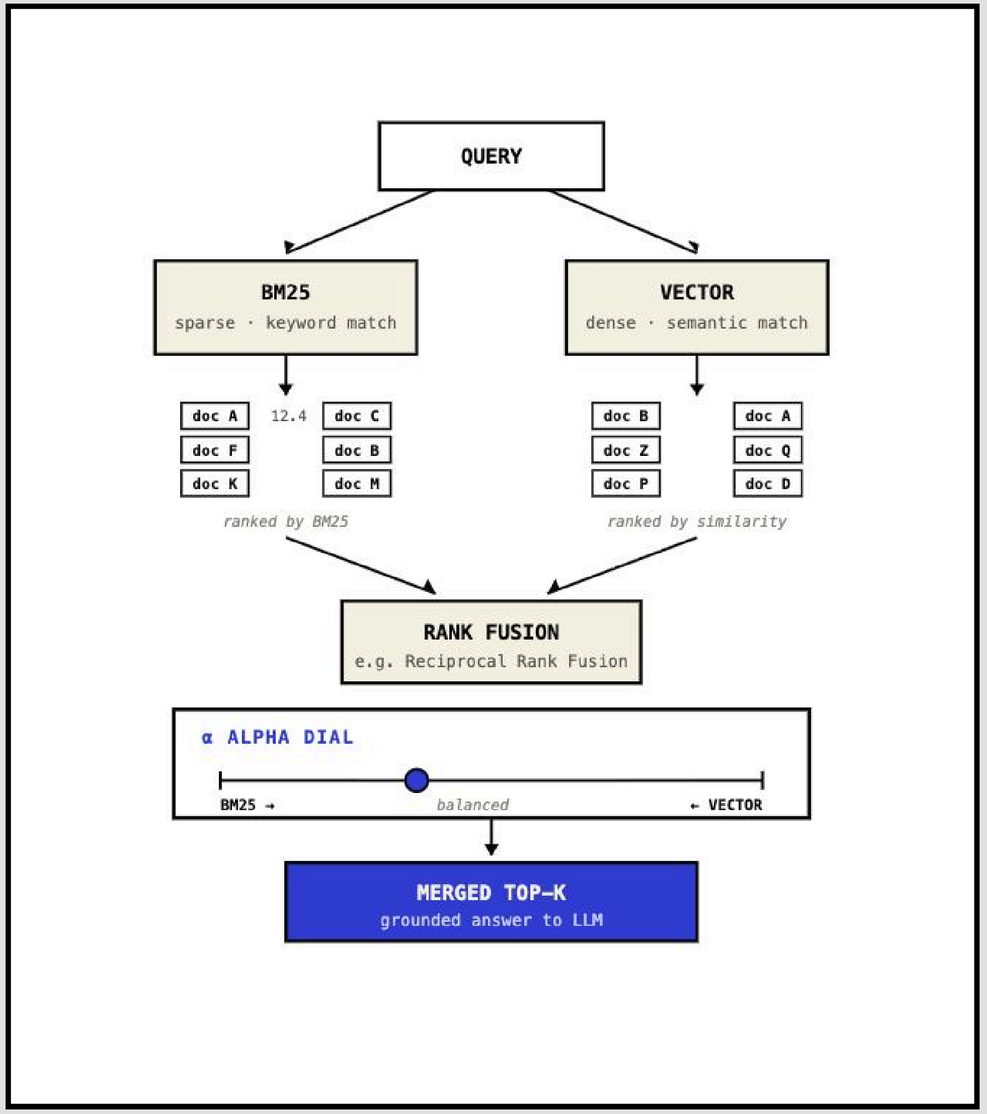
Hybrid systems run both searches simultaneously — keyword matching and semantic similarity, in parallel.

MERGING RESULTS

Both sets are not just dumped together. A ranking algorithm — typically Reciprocal Rank Fusion — mathematically blends the scores from both methods.

THE ALPHA DIAL

Most hybrid systems let you adjust the weighting (often called α). Dial closer to BM25 for technical, heavily-coded datasets — closer to Vector for conversational, knowledge-based queries.



A RAG STACK

Four stages. Three of them happen before a query ever arrives

OFFLINE / ASYNC *prepare the index*

01 **Chunking** ✂️
Split source documents into retrievable units

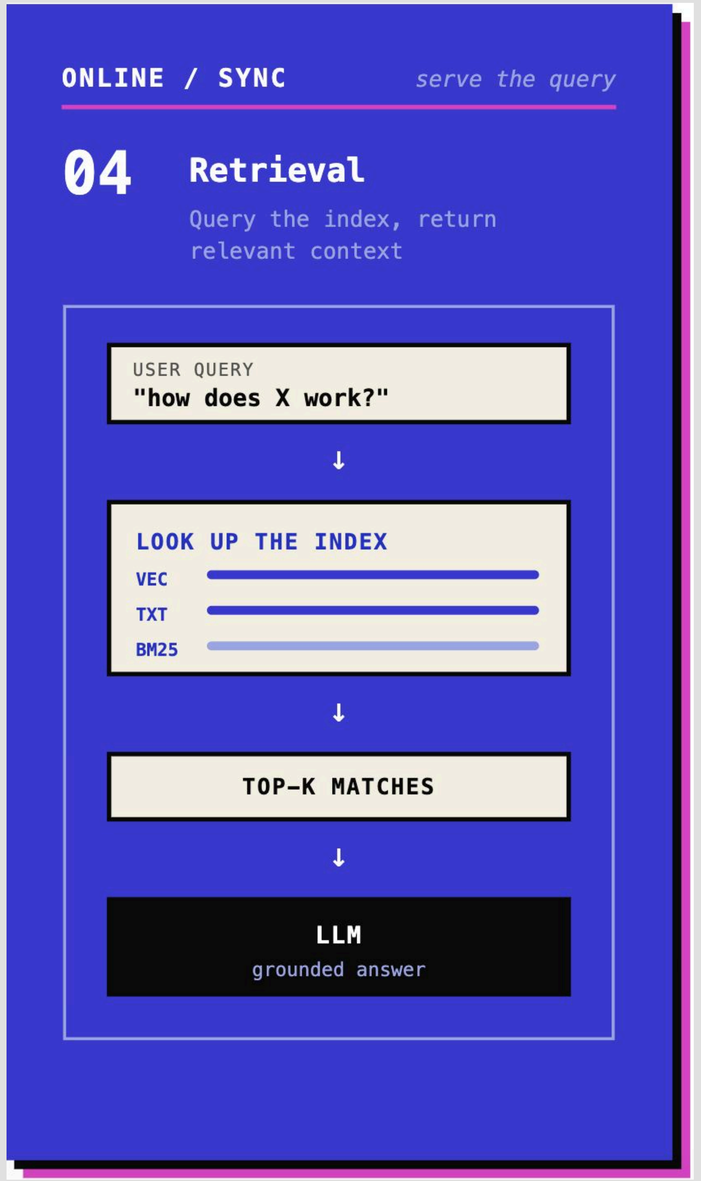
02 **Ingestion**
Embed, enrich, and structure each chunk

03 **Storage**
Indexed for fast lookup — typically more than one store

VEC	vector index — semantic similarity
TXT	raw chunks — with IDs
BM25	keyword — inverted Index

ONLINE / SYNC *serve the query*

04 **Retrieval**
Query the index, return relevant context



LATE CHUNKING & RERANKING

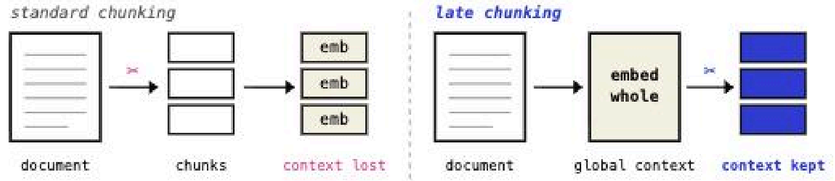
Two fixes that meaningfully improve standard RAG

01 LATE CHUNKING

solving the context-loss problem

HOW IT WORKS

Pass the entire document through the embedding model first, so every token understands global context. Only then cut into chunks.



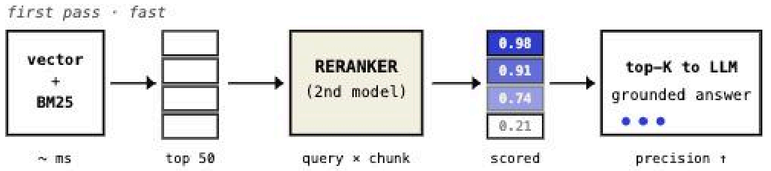
The result:
Chunks that still remember what the rest of the document was talking about.

02 RERANKING

the quality filter

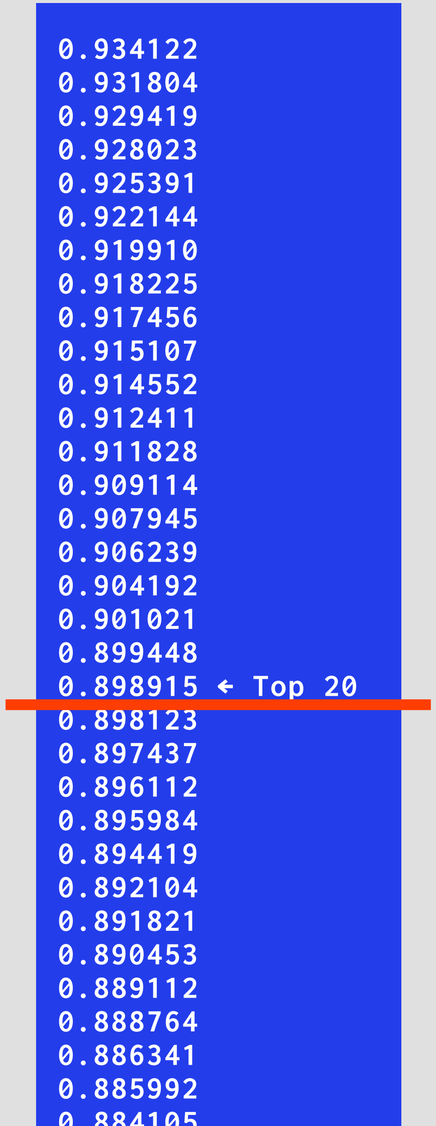
HOW IT WORKS

Two-stage retrieval. Fast methods grab a broad candidate set; a second model scores each candidate against the query for true relevance.



HOW HUMANS DO RESEARCH

How the Machine picks documents (Chunks)



The Iteration Gap

Humans don't read 20 paragraphs and stop. They read, learn something, then ask a new question. Standard retrieval misses this iterative loop entirely.

The Missing Link

If no single document mentions A and B together, they all just describe A to X and X to B, vector search fails. It can't connect the dots across separate documents to build a bridge that was never written down.

The Knowledge Management Trap

If the system can't infer connections, you're forced to pre-map them by hand. How much time and money does it cost to manually make an entire corporate knowledge base "AI-readable"?

AND WE HAVEN'T EVEN ASKED →

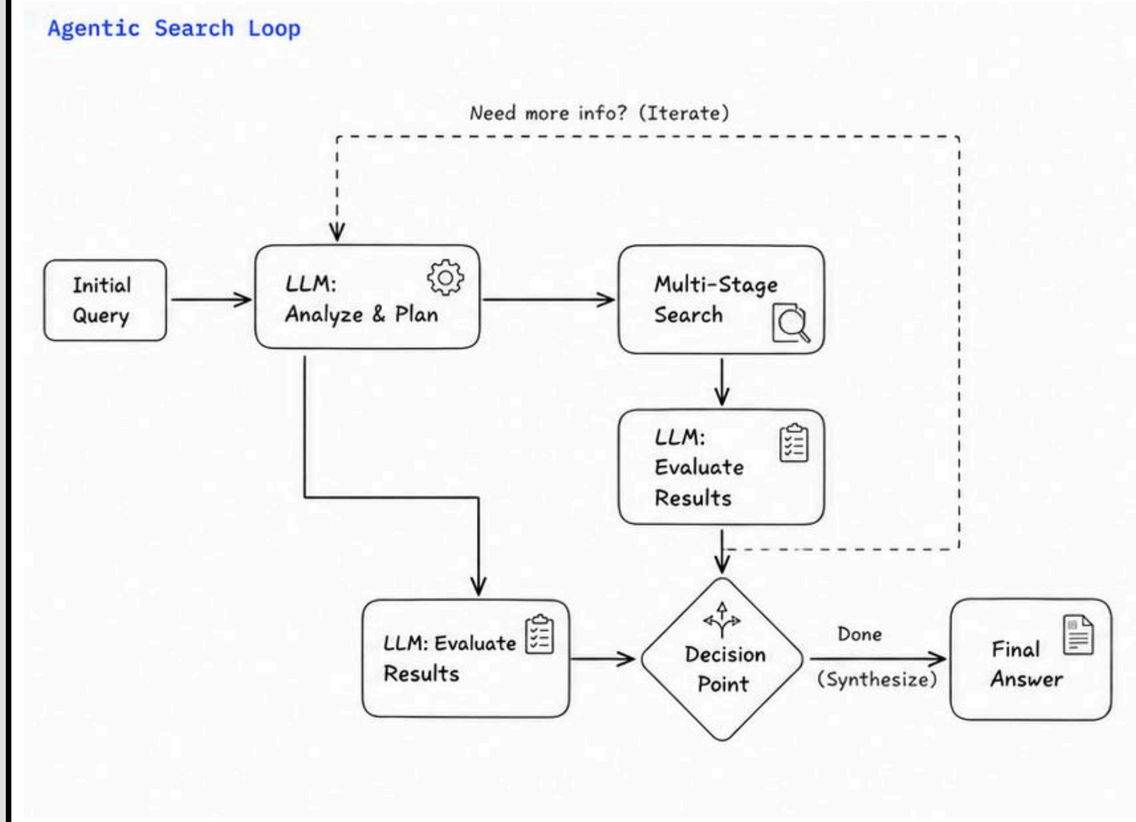
- contradcitions?
- timeline problems?
- big-picture blindspots?

AGENTIC SEARCH

Iterative retrieval using an LLM agent

HOW IT WORKS

An LLM agent plans what it needs to know, searches iteratively, evaluates results, and decides what to do next. It can reformulate queries, follow references, and chain multiple retrieval steps to solve complex information needs.



STRENGTHS

- Handles complex, multi-hop questions
- Can follow leads across documents
- Adapts queries based on new information

LIMITATIONS

- Higher latency and compute cost
- Errors and hallucinations can compound
- Quality depends on the agent's reasoning

KNOWLEDGE GRAPHS

THE BASICS

Most KG follow an RDF grammar composed of Entities and Edges:

[Leonardo da Vinci] --(painted)⟶ [Mona Lisa]
--(displayed at)⟶ [Louvre Museum]

INGESTION WORKFLOW

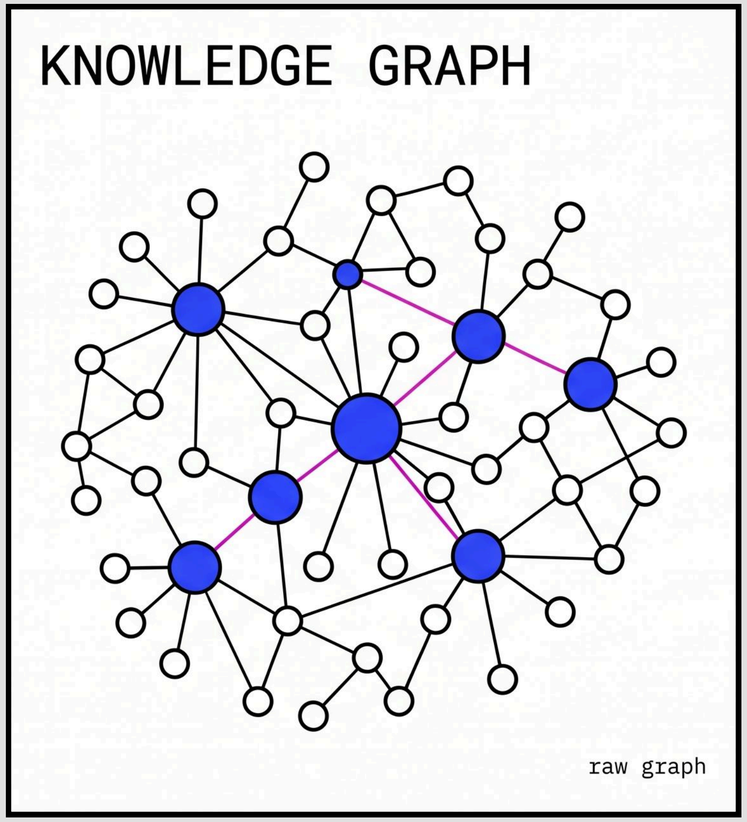
DOCs ⟶ Chunks ⟶ LLM Extraction ⟶ GraphDB

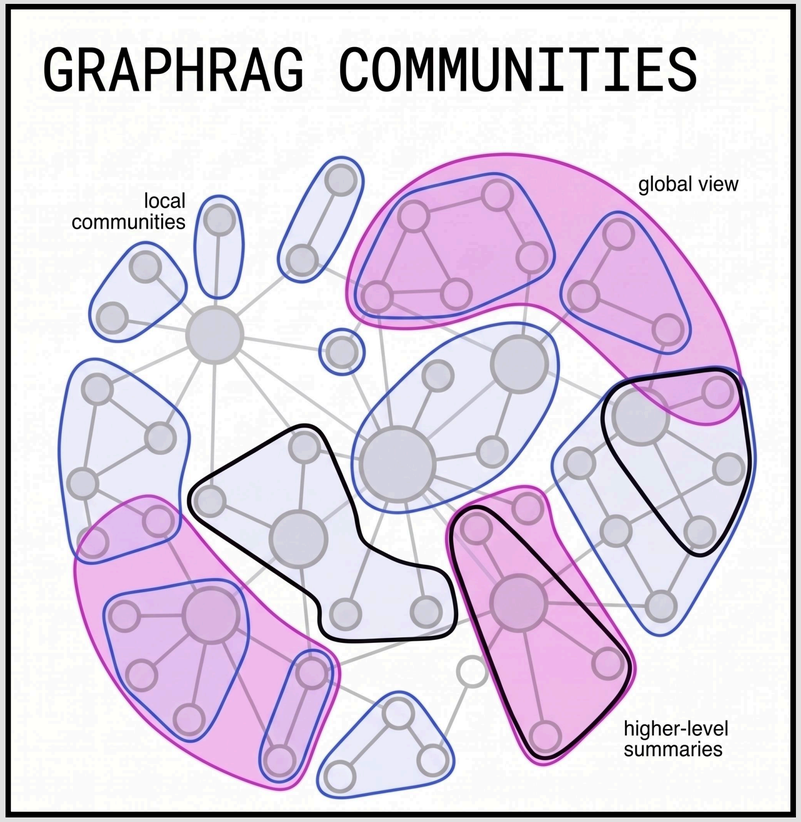
THIS ENABLES

1. Multi-Hop Reasoning
2. Cross-Document Synthesis
3. The "Global" Overview

LIMITATIONS

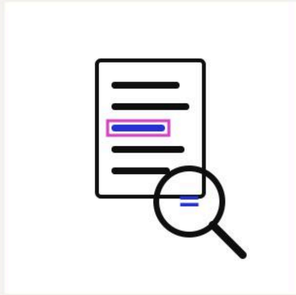
- higher upfront costs
- schema drifts and becomes brittle
- hard to govern (RBAC, Data Governance, ...)
- hard to update and information becomes stale





EXISTING METHODS RECAP

01
Keyword match

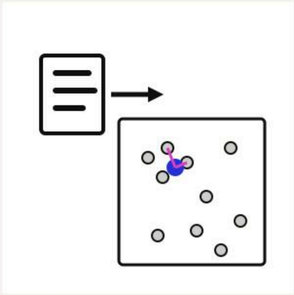


BM25

keyword search

Exact term matching using statistical ranking.

02
Semantic meaning

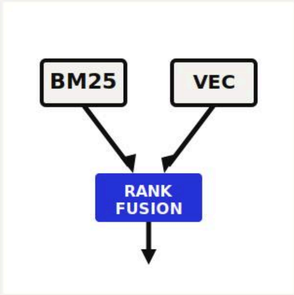


VECTOR

semantic search

Meaning-based retrieval using embeddings.

03
Best of both

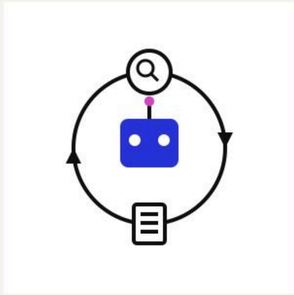


HYBRID

BM25 + vector

Combines lexical and semantic, then reranks.

04
Iterative reasoning

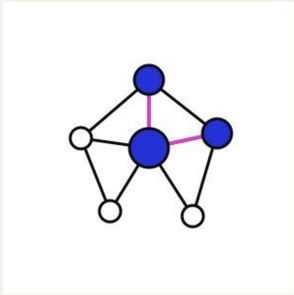


AGENTIC

iterative search

LLM agent searches in loops, follows leads.

05
Connected knowledge



GRAPH

graph-based

Traverses entities and relationships.

NEW APPROACH

RAG

RAG retrieves chunks. Useful, but is weak on relationships and multi-hop evidence.

AGENTIC SEARCH

Agentic search retrieves through loops. Flexible, but slow, costly, and inconsistent at scale and risk hallucinations.

KNOWLEDGE GRAPHS

Knowledge graphs store relationships. Powerful, but often brittle, stale, lack provenance, hard to govern, and human-first.

AI BASELINE

- Fixes **knowledge graphs** by making the graph retrieval-native, updateable, and provenance-aware.
- Fixes **agentic search** by moving repeated context work into a reusable knowledge layer.
- Fixes **RAG** by retrieving connected evidence, not just similar chunks.
- The result is high-quality context for models, with lower repeated work and stronger auditability.

It is a provenance-aware knowledge engine that acts as a governable and updateable knowledge layer for agentic workloads.

WHAT ORGANIZATIONS SHOULD EVALUATE

Three pillars before you trust retrieval in production

01
TRUST, SECURITY & GOVERNANCE

Can’t deploy AI if it compromises data or breaks compliance.

- **Access controls.** Strictly enforce existing permissions — no backdoor to restricted data.
- **Data governance.** Clear rules for the data lifecycle: what’s indexed, residency, ownership.
- **Provenance & audit.** The AI must “show its work” with exact citations; full query-log audit.

02
ENGINEERING RESILIENCE

A system built for 5 users must not collapse under 5,000.

- **Deterministic grounding.** Anchor answers to retrieved data — eliminate guesswork and hallucination.
- **Latency expectations.** Search is milliseconds; agentic loops run minutes. Deliver consistent performance.
- **Operational scalability.** Data is living: automated, zero-downtime pipelines to delete, edit, re-index.

03
ECONOMICS & COST CONTROL

Infrastructure scales dynamically; budgets do not.

- **Cost predictability.** Forecast vector-DB compute and LLM token costs against real traffic.
- **Cost control.** Hard guardrails — token caps, rate limits, cheap-vs-expensive model routing.
- **Agentic guardrails.** Open-ended agent loops can burn millions of tokens; constrain them.

Q&A

Now's the time — ask us anything about retrieval, agents, or what we covered. No question too small, no edge case too weird.

CONTACT

AI FACTORY AUSTRIA

Training & Skills Development

AI Factory Austria AI:AT
Karl-Farkas-Gasse 22
1030 Vienna, Austria

training@ai-at.eu info@ai-at.eu
ai-at.eu
@ai-factory-austria

FUNDED BY

under discussion with

AI Factory Austria AI:AT has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101253078. The JU receives support from the Horizon Europe Programm of the European Union and Austria (BMIMI / FFG).

THANK YOU !

If today's session sparked something — whether you're building agents that run research tasks or retrieval for your own knowledge base — there's more on how we approach reliable retrieval at scale over at AI Baseline.

STAY IN TOUCH

- Follow us:@ai-baseline
- Write us team@ai-baseline.com
- Talk to the founders
- Learn more:ai-baseline.com

AI BASELINE