Optimizing retrieval augmented generation with graph language model

Aga Kopytko
Founder & CTO @Smabbler
akopytko@smabbler.com

10 years experience in NLP (natural language processing) and graph technologies.

Research areas: natural language algorithms, semantics, symbolism, graph compositionality, lifelong learning, micro-agents.

Fan of planes, cars, diving, climbing and zombie movies.

Key talking points

1. RAG (retrieval-augmented generation) in a nutshell
2. Key RAG challenges
3. Benefits of using graph technologies
4. A novel graph architecture with built-in NLP
5. Simplified Graph RAG pipeline
6. From prototyping to production deployment in hours
7. Q&A

RAG in a nutshell

RAG

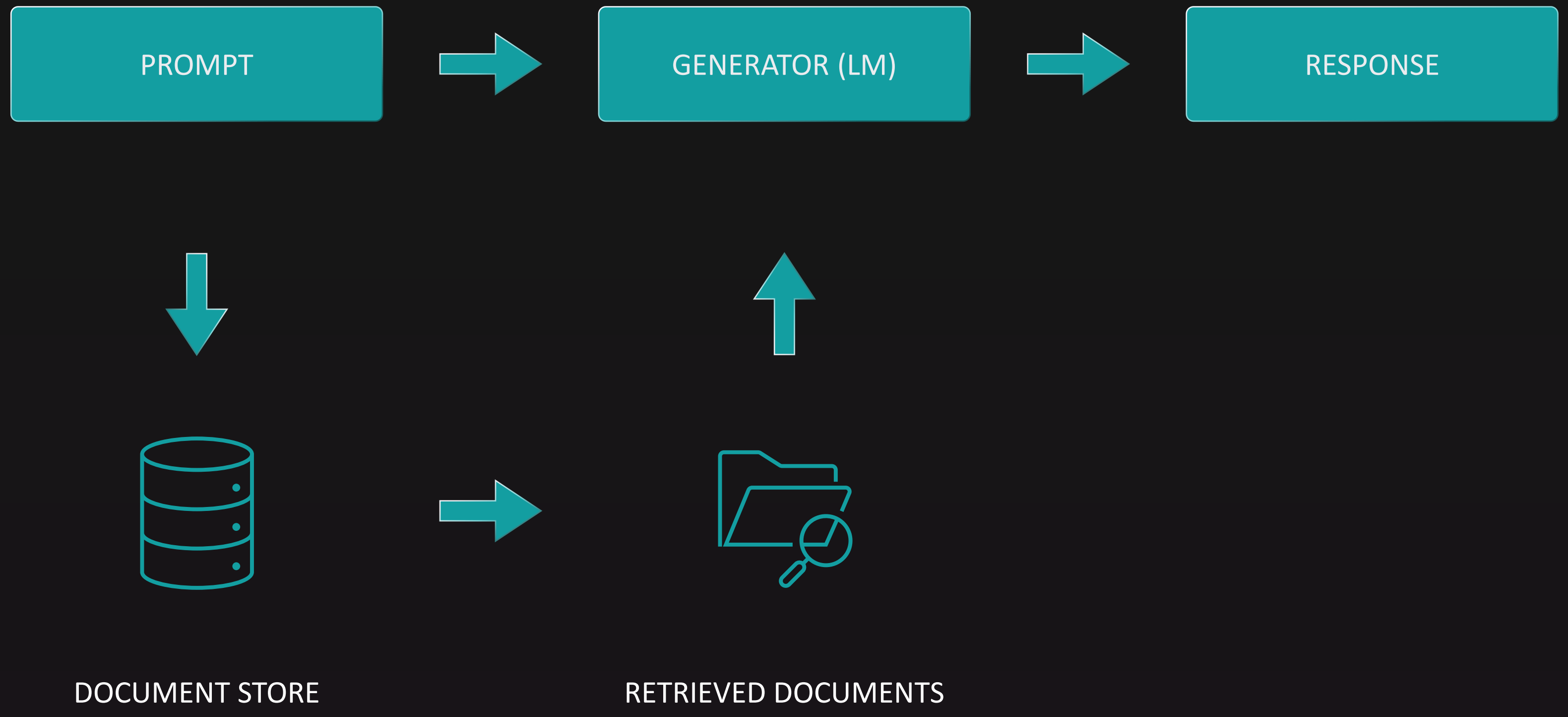
- extends LLM knowledge with additional data source.
- combines elements of both retrieval-based and generative models.
- allows for more reliable and personalized generative output.

LLM output WITHOUT RAG

- LLM creates a query response based on information it was trained on

LLM output WITH RAG

- a retrieval component is added between a query and an LLM response to firstly pull information from a new data source


PROMPT
GENERATOR (LM)
RESPONSE
DOCUMENT STORE
RETRIEVED DOCUMENTS

Key RAG challenges

3 key RAG challenges

Creating **External Data***

Converting data into numerical representations / relations and storing converted data in a vector database / graph database

The choice of how to create the external data has implications for further steps - including the accuracy and complexity of the RAG pipeline

Maintaining **Quality Retrieval**

E.g., Semantic search, Vector (similarities) search, Keywords matching

Low retrieval quality reduces the accuracy and relevance of generative outputs

Handling **Integration Complexity**

RAG systems have several interconnected components, e.g., retrievers, rankers, and generators.

Multitool pipeline can make it challenging to maintain flexibility, scalability, and continuous improvement

*External data – any new data outside of the LLM's original training data set

Baseline (vector) RAG vs. Graph RAG.

Embeddings (vector RAG)

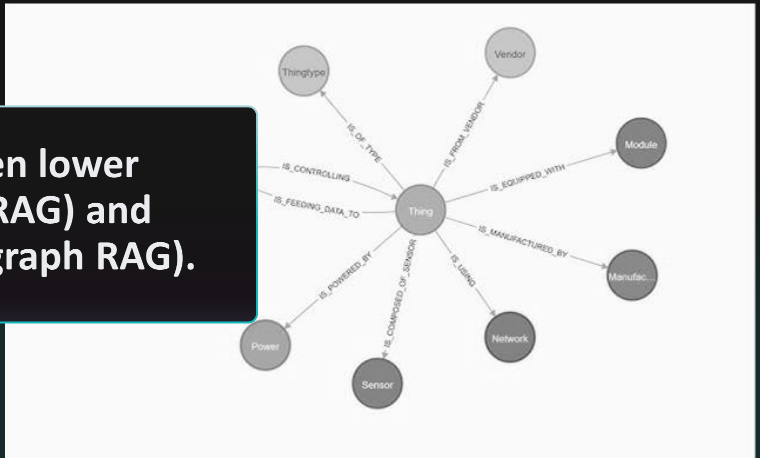
```
embeddings_index.get("shop")

array([ 3.0426e-01, -1.4191e-01, -7.9738e-01, -3.5484e-01,  3.0333e-01,
        4.3690e-01, -9.8706e-02,  6.9080e-01,  6.9362e-01,  1.8528e-01,
        1.0648e-01, -4.5209e-01,  8.7568e-01,  1.1414
        6.0731e-01,  2.7596e-01,  2.3698e-01, -7.1692
        4.3669e-01,  4.1931e-01,  2.1568e-01, -1.2316
       -9.0922e-02, -3.8767e-01, -7.0817e-01, -2.4242
       -3.8969e-01,  5.2464e-01,  2.1317e-01,  8.8327
        6.7755e-01, -3.3464e-01, -6.1269e-01,  8.2305
        8.5966e-01, -4.6323e-01, -1.3172e-02, -8.1801
        1.7025e-01, -6.3946e-01,  4.8516e-01,  6.1706
       -1.7953e-01,  4.8890e-03, -4.7809e-01,  5.8311
       -1.7160e+00, -1.3190e+00,  9.0167e-02,  1.3612
        2.1325e-01,  1.5207e-01,  2.9252e-01,  5.7116
       -1.4311e-01,  1.2564e+00, -1.6377e-01,  6.9895
       -4.3554e-01,  4.1309e-01, -1.4767e-01, -4.0058e-01,  2.1931e-01,
        1.9361e-01,  6.5205e-01, -2.0986e-01, -5.8788e-01, -1.4051e-01,
        1.2399e-01, -8.9099e-03, -1.5384e-01, -4.6232e-02, -6.4600e-01,
       -3.1246e-01, -1.4165e-01, -7.6865e-01, -2.7654e-01, -7.6462e-03,
        6.9244e-01,  3.6744e-01,  1.0840e+00, -2.4375e-01, -8.9562e-01,
       -2.3390e-01,  1.4788e-01,  1.3795e-01,  1.2635e+00,  1.0817e-01],
      dtype=float32)
```

Opaque. Struggles with accuracy and efficiency.

Choosing between lower accuracy (vector RAG) and higher complexity (graph RAG).

Relations (graph RAG)



Transparency, but also higher complexity.
Monthly database cost can exceed 10,000 EUR.

It’s easy to start but building RAG is difficult.
Current tools are not designed to be simple and / or optimized.

Growing complexity

Compromises between lower accuracy (opaque vector RAG) and higher complexity (more transparent graph RAG).

Complex and difficult retrieval.

Increasing costs

Building RAG takes even 100 days.

Only the monthly database cost can exceed 10,000 EUR.

Projects fail

Even 80% of projects do not reach production.

Benefits of using graph technologies

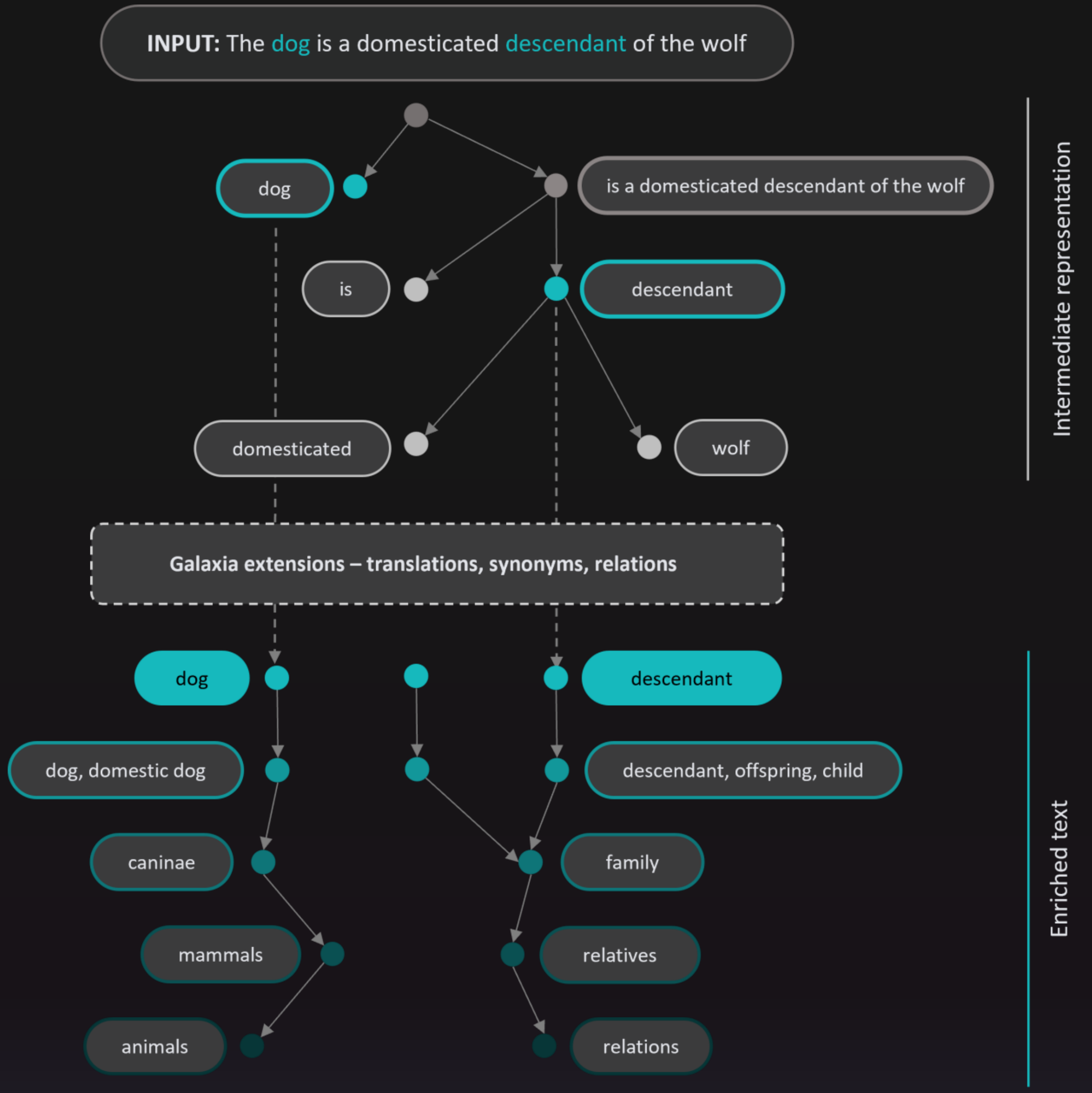
- Explainability
- More contextually relevant answers
- Integration of information from multiple sources (graph relations)

A novel graph architecture with built-in NLP

LLM independent graph language model (Galaxia)

- Symbolism
- Semantics
- Compositionality

Galaxia augments raw text by injecting relations and context into data



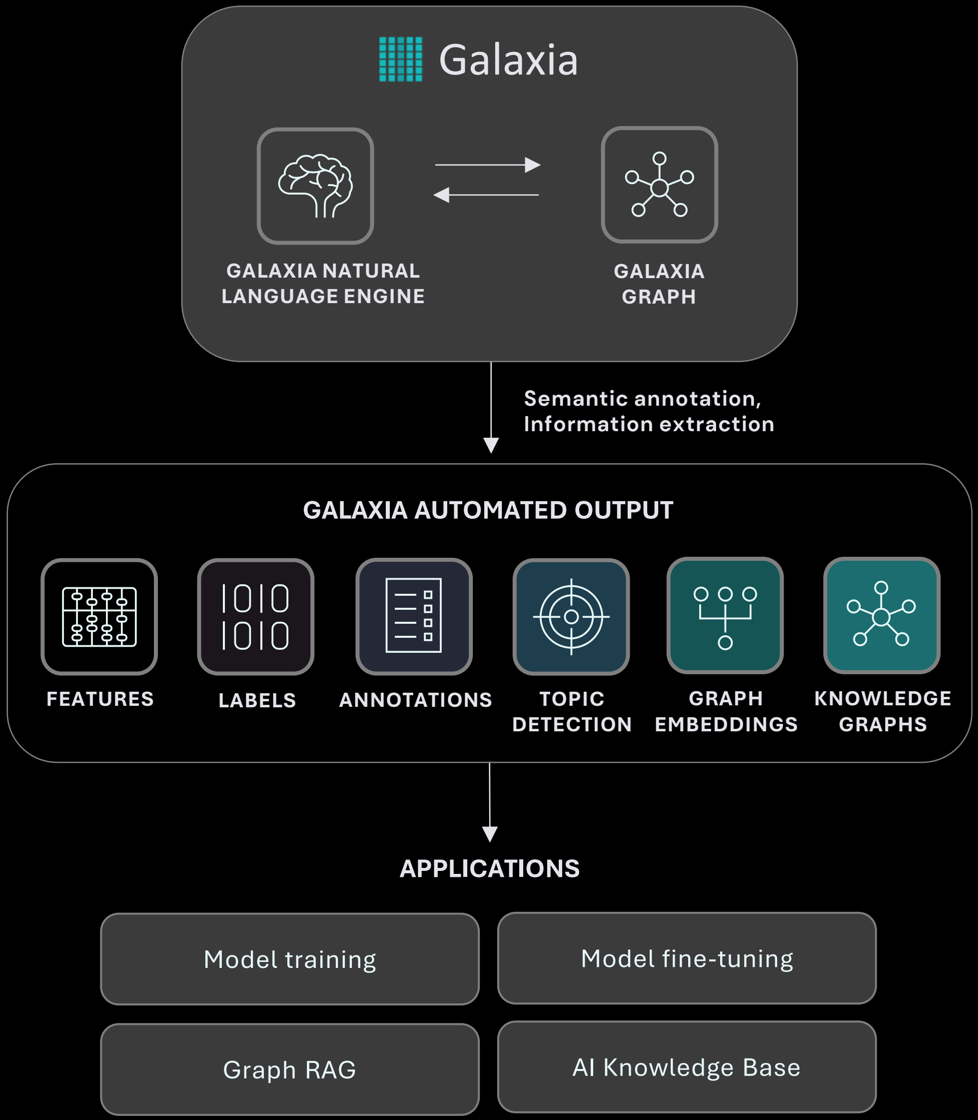
Sentence parsing - forming intermediate representation:

- Tokenization
- Lemmatization
- Part of Speech (POS) tagging
- Dependency parsing

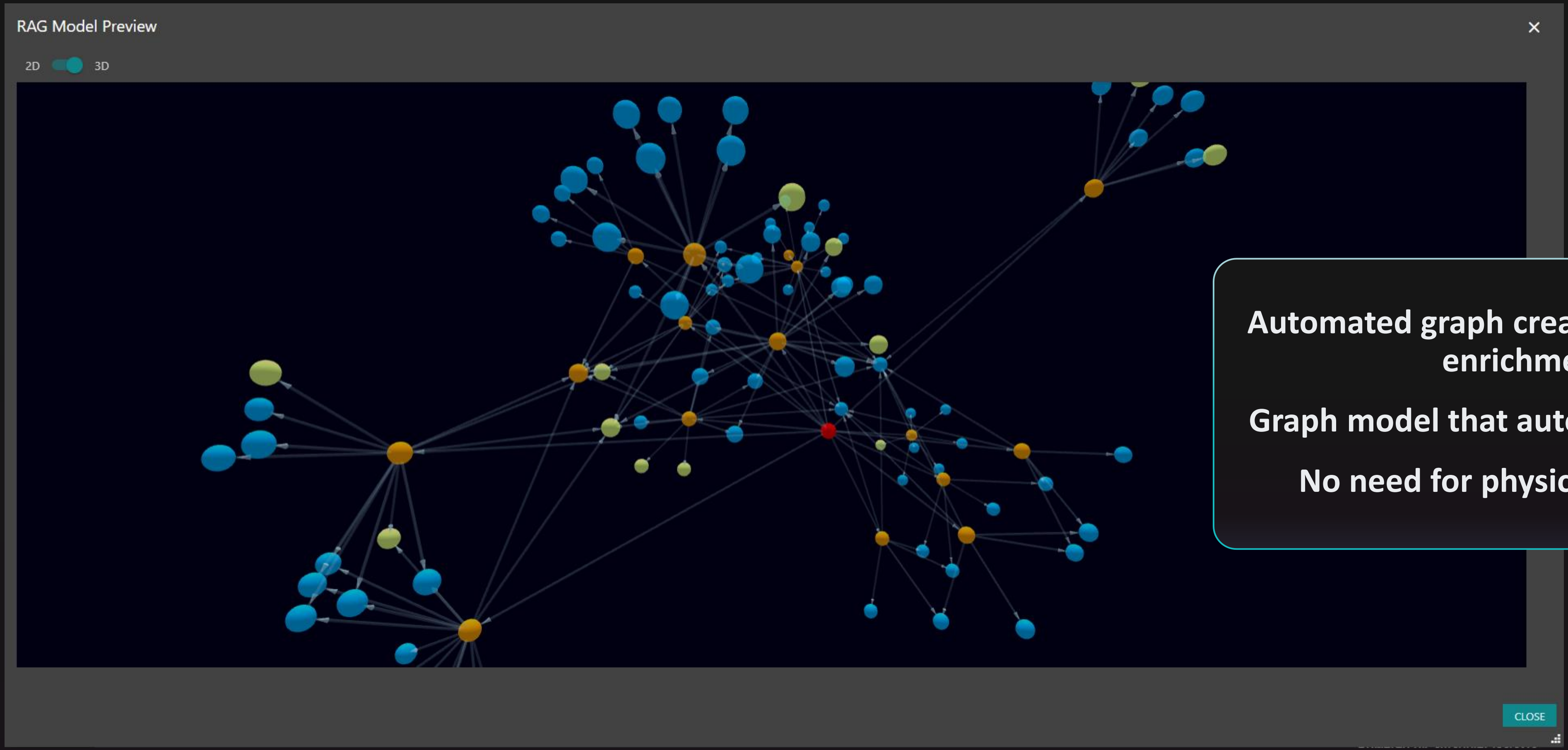
Language graph - enriching text with:

- Semantics
- Similarities
- Definitions
- Context and relations

Continuous communication: language engine and graph structure / knowledge base



In-memory graph automates retrieval and assures transparency.



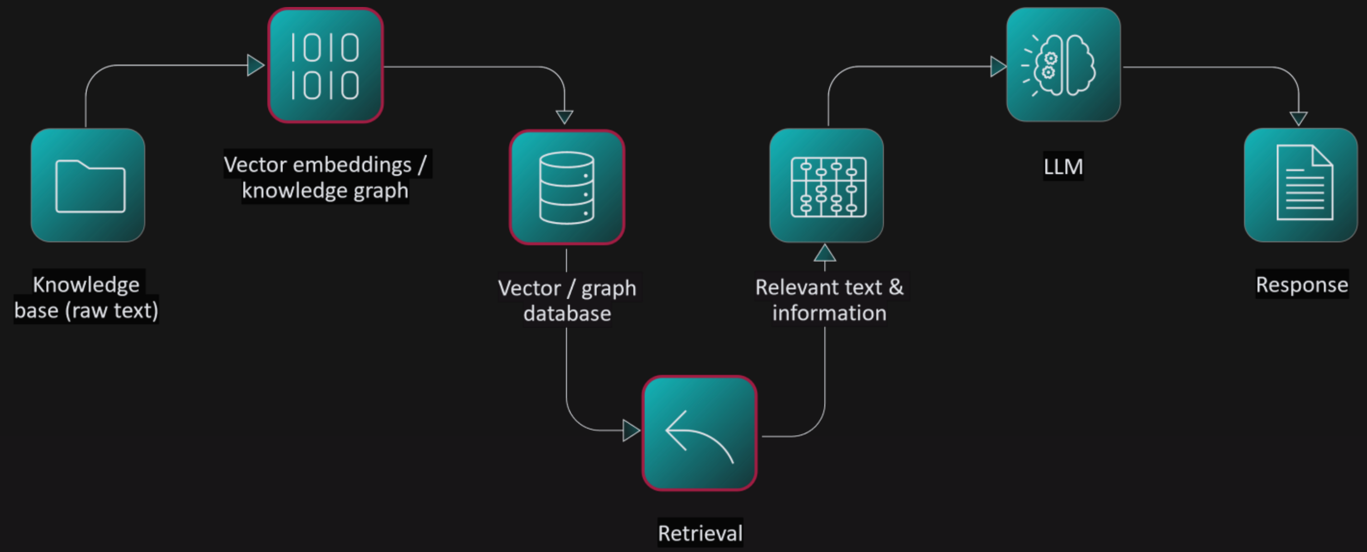
Automated graph creation and context enrichment.

Graph model that automates retrieval.

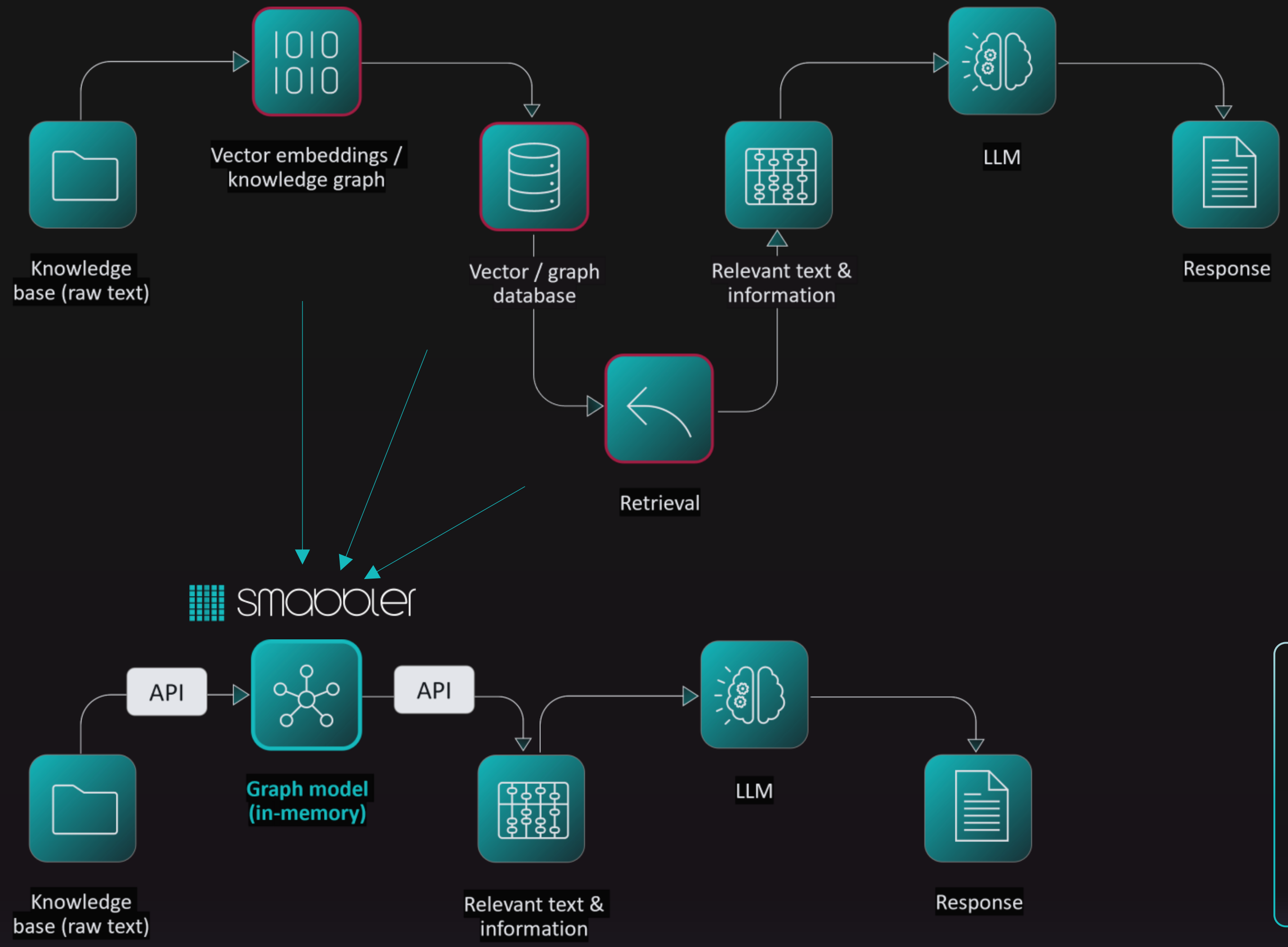
No need for physical databases.

Simplified Graph RAG pipeline

Multi-tool 'Classic' pipeline

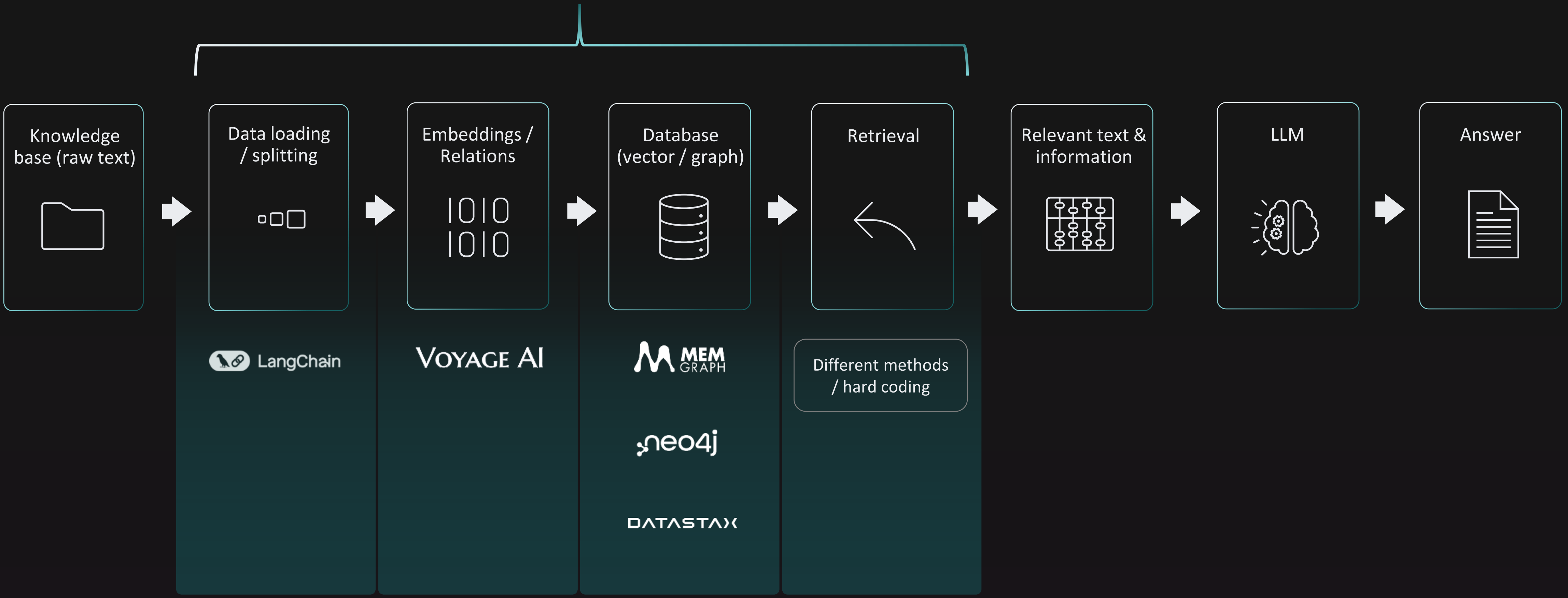


Optimized ‘one-stop’ pipeline



- One stop API Graph RAG.
- No need for databases.
- Cost and energy efficient (runs on CPUs).

Smabbler API Graph RAG - cutting down time from months to hours



+ different frameworks / suites of tools

nuclia vectara glean

1. Parsing

Sentence / text parsing to form intermediate representation - tokenization, lemmatization, part of speech (POS) tagging, dependency parsing. The text is represented as a graph structure.

2. Enrichment

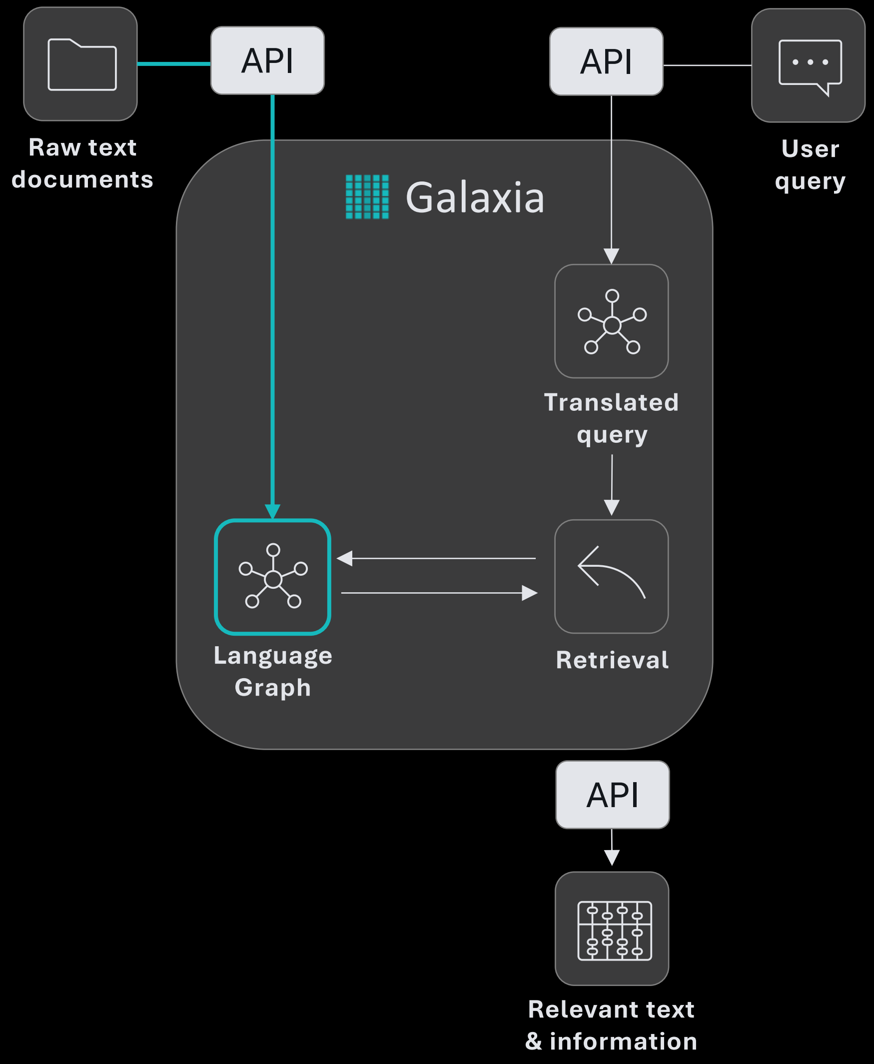
Enriching text with semantics, similarities, definitions, context and relations. The enriched text is represented as a highly interconnected graph structure, creating a user's Language Graph.

3. Storage

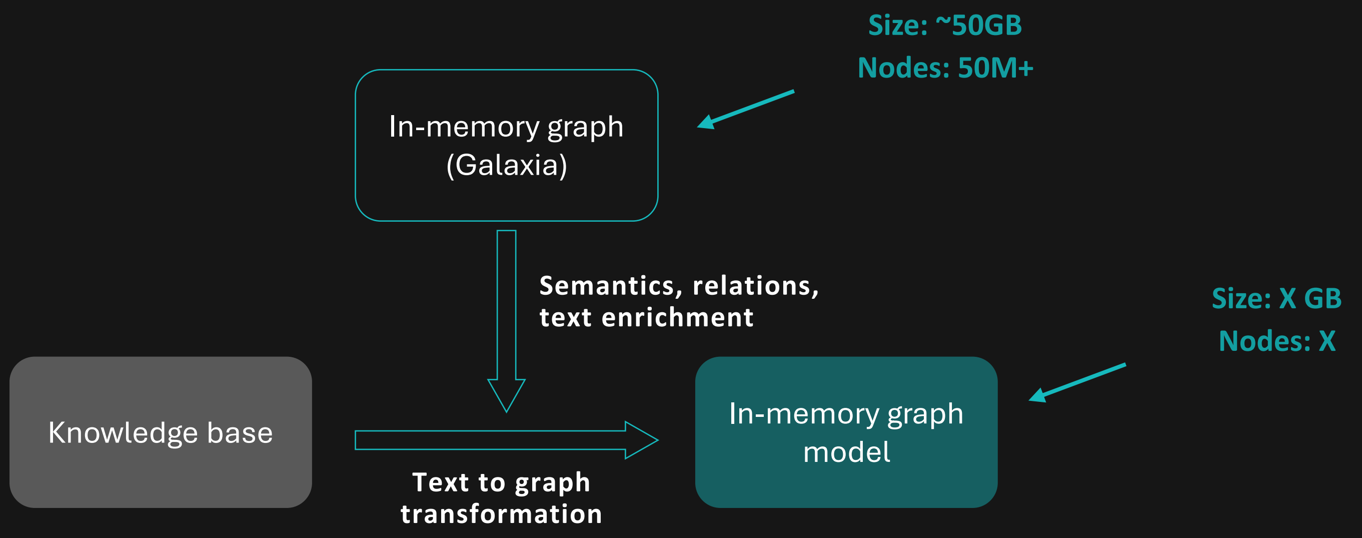
Language Graph (Graph Load) in the form of a file (JSON) is stored in the database in the user's account

4. Activation

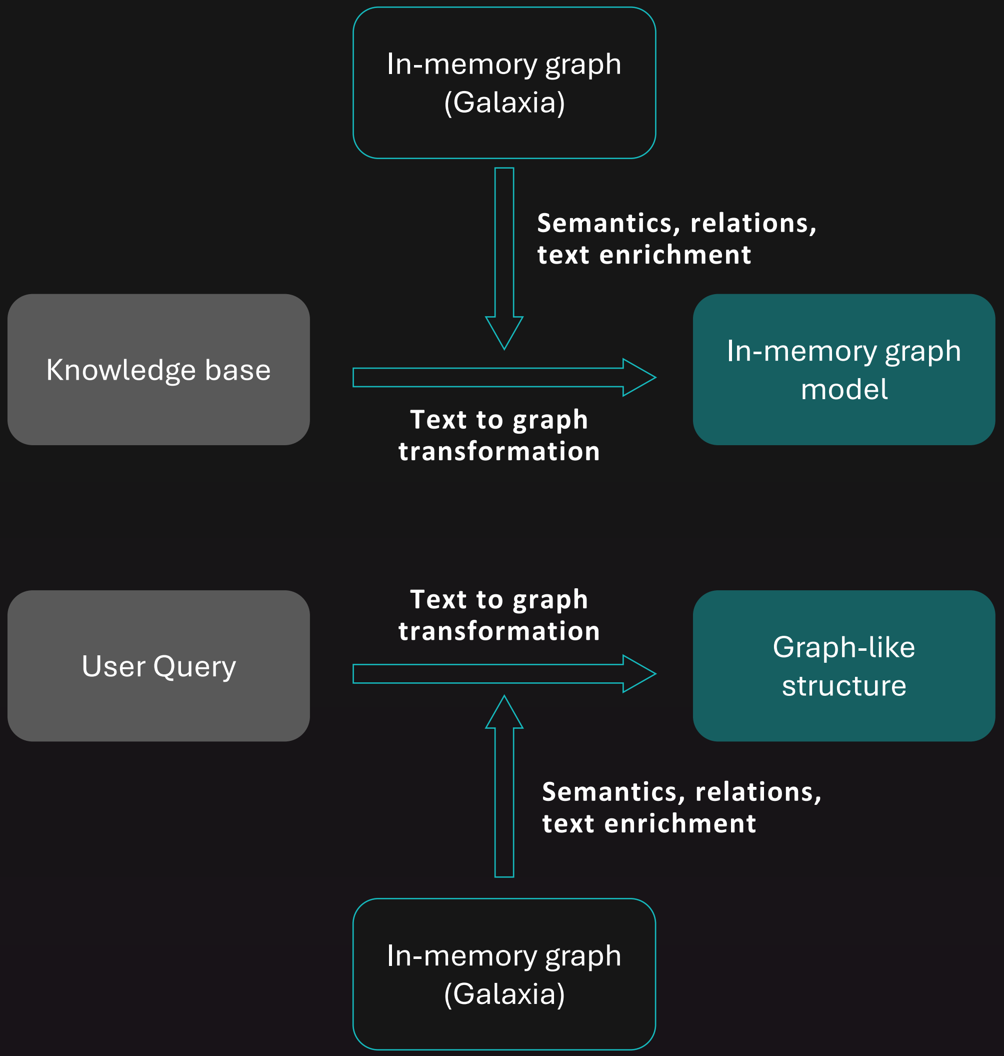
During the activation graph load connects to the Galaxia graph language model, to create a user Graph Model.



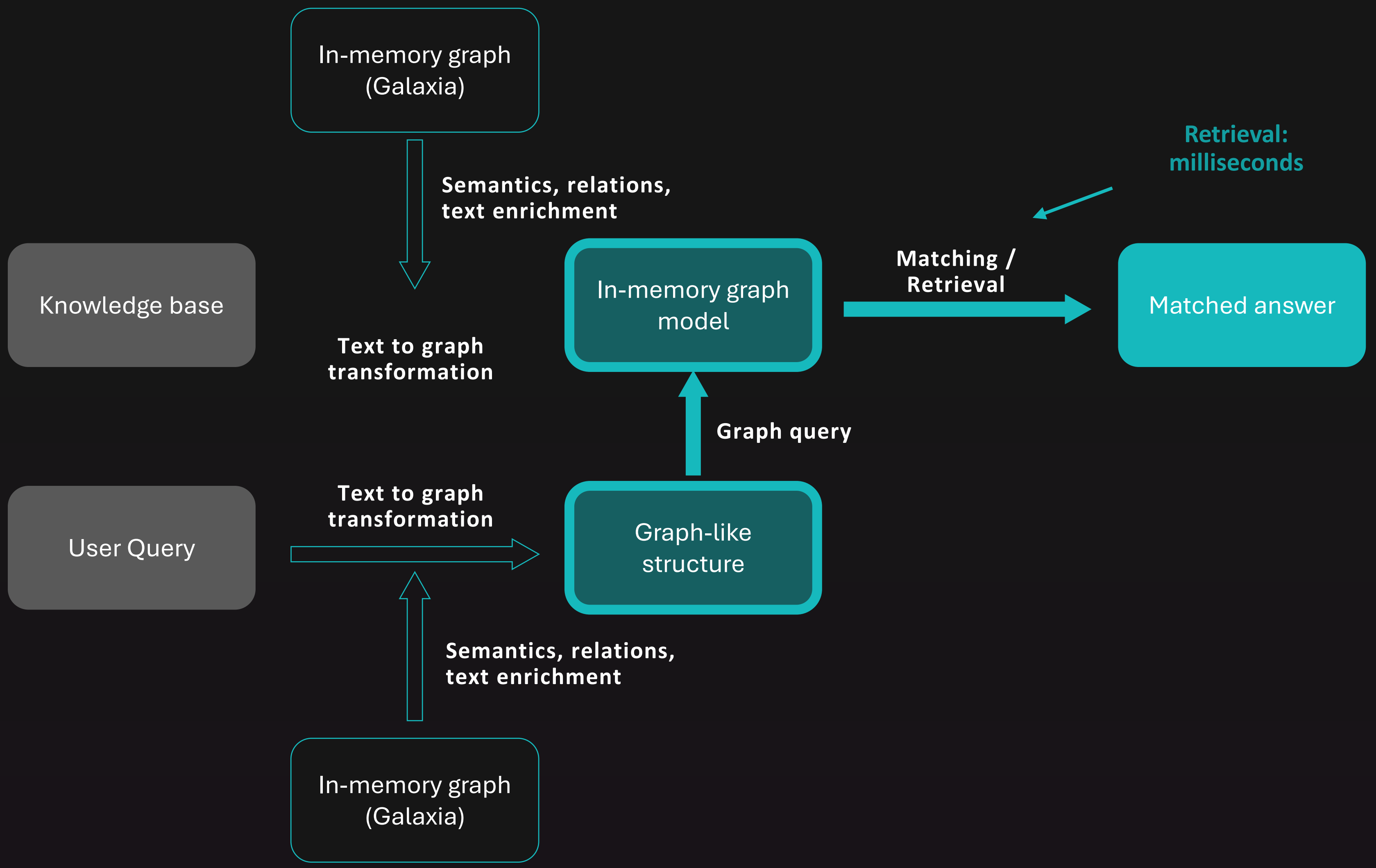
Knowledge base to graph transformation is done by Galaxia - a graph language model



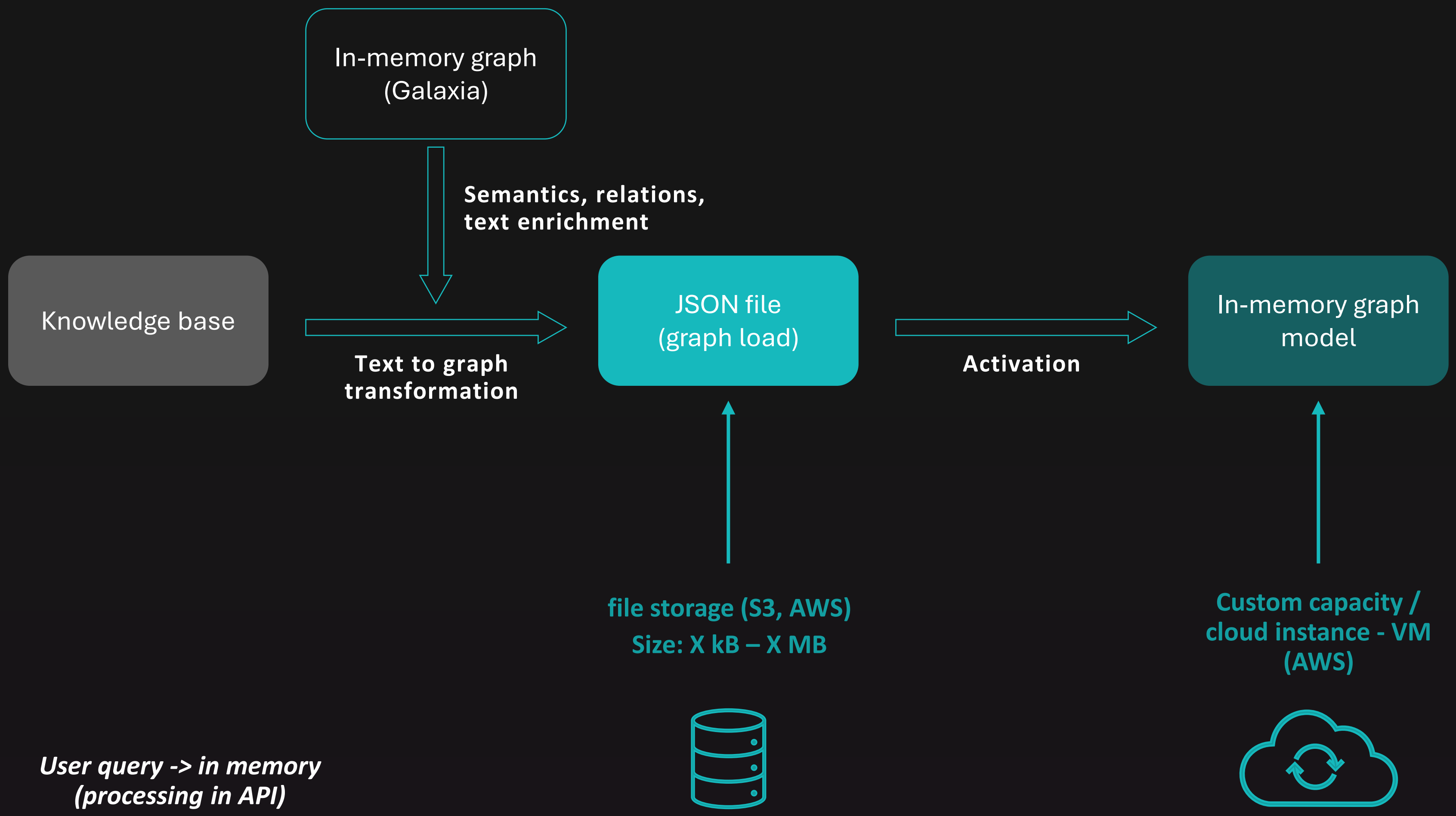
The user query is also transformed into a graph structure



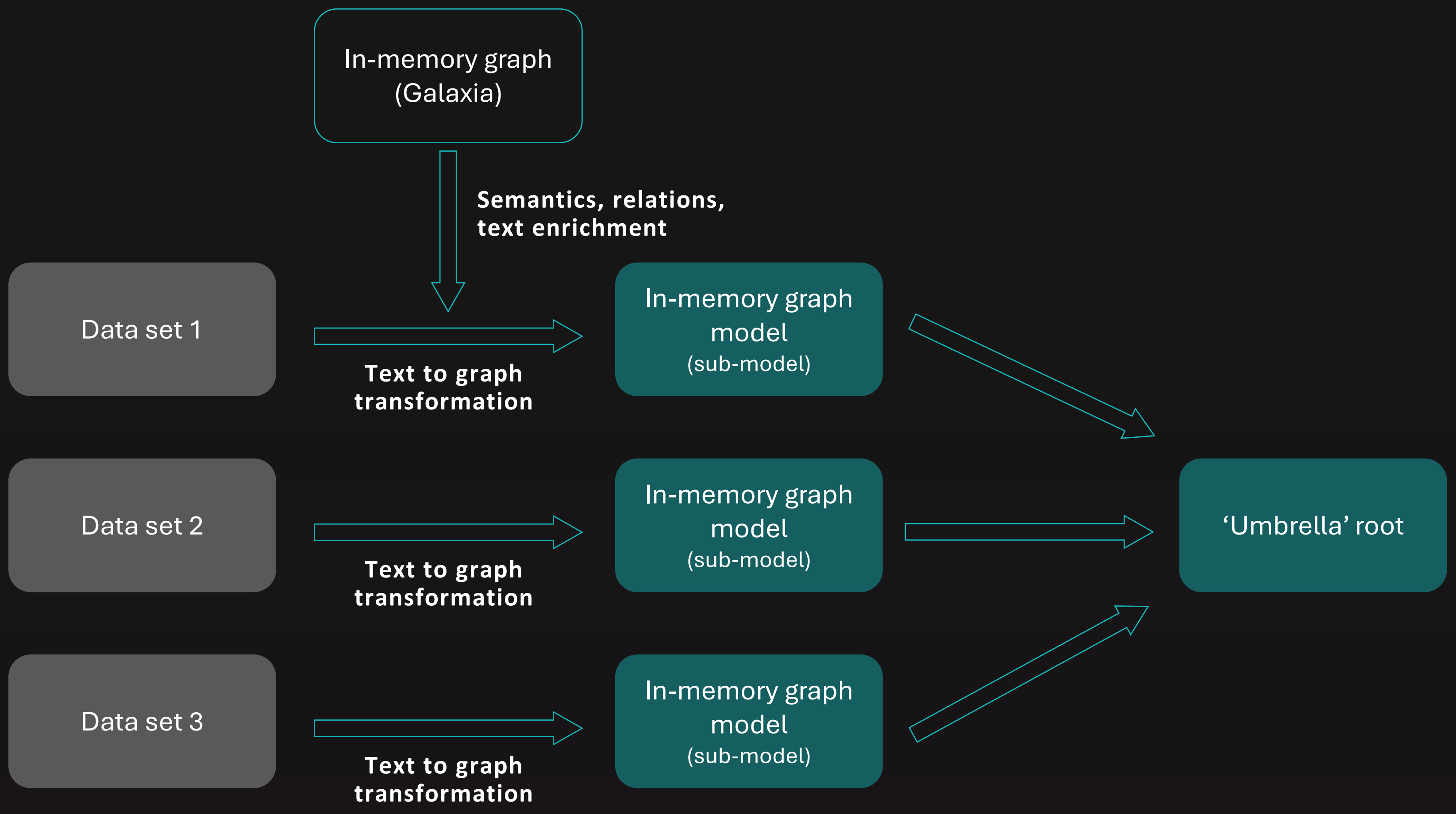
The graph query structure is matched with the in-memory graph model for the knowledge base



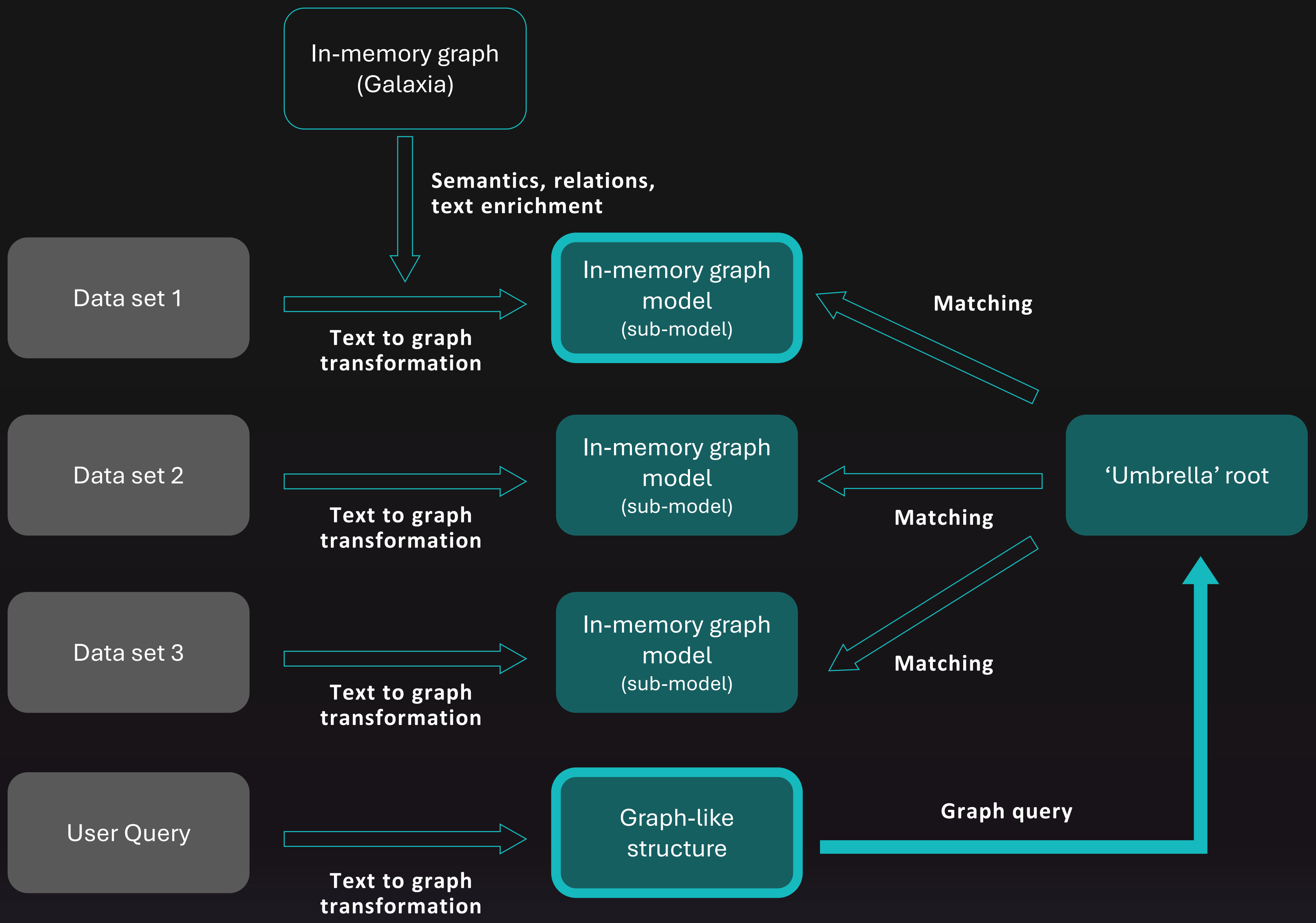
The created non-active graph model for the knowledge base is stored in the JSON format



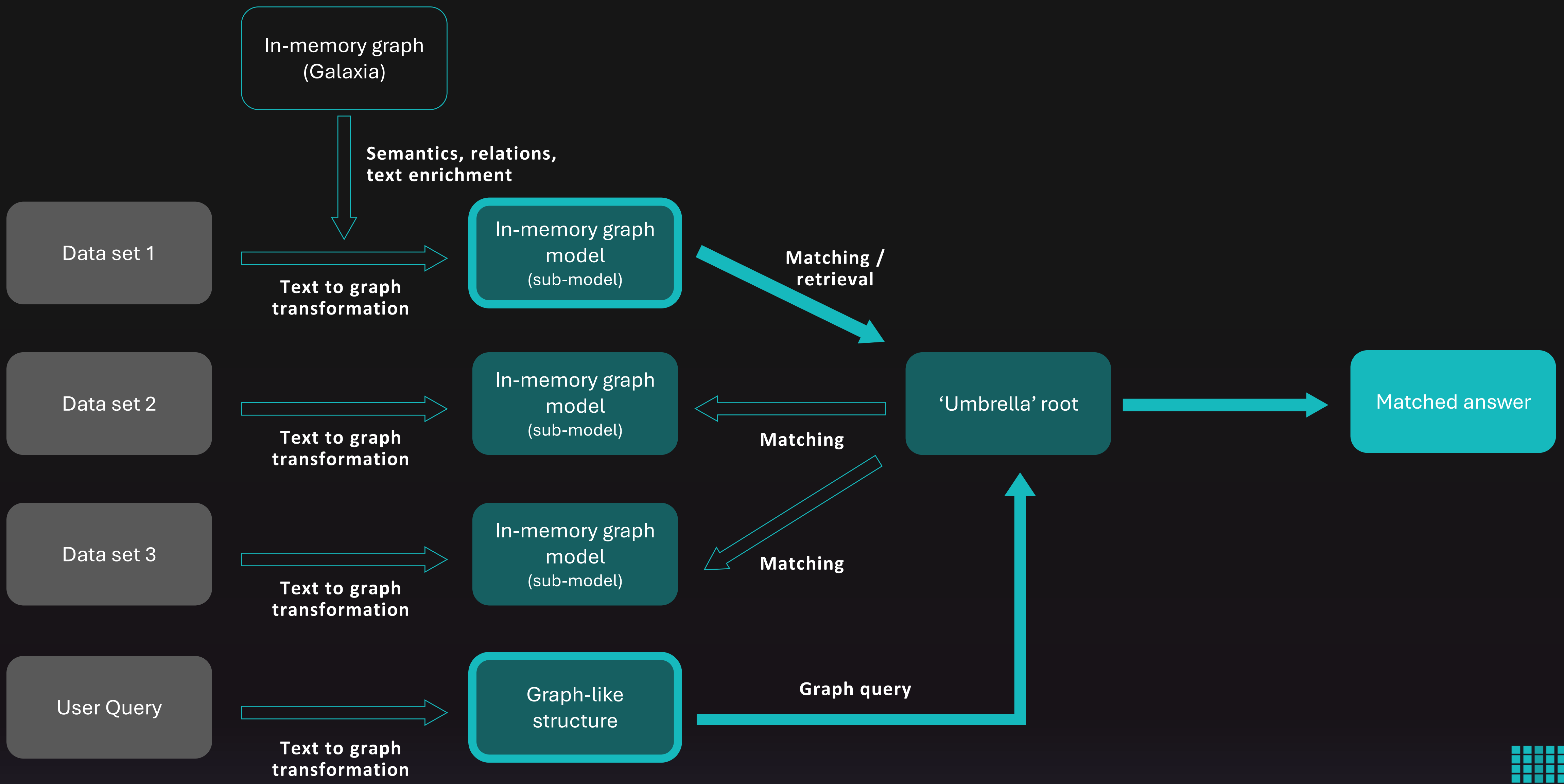
Size & retrieval optimization: Optimizing the size of the graph model by dividing the database into smaller parts



Size & retrieval optimization: The user query then passes through the 'umbrella' root (a cluster of smaller sub-models)



Size & retrieval optimization: During retrieval, the appropriate sub-model returns the result



From prototyping to production deployment in hours

Home

Account

Analysis

Docs

API

Q-Lab

query-user user developers@smabbler.com LOG OUT

smabbler

Welcome to Smabbler Portal

Analysis

Insert your own text, choose a random quote or upload your CSV file to analyze.

Query

Check our query wizard and build your own query.

DOCs

Flick through the API docs to find out more.

Powered by Smabbler Galaxia

Thx!

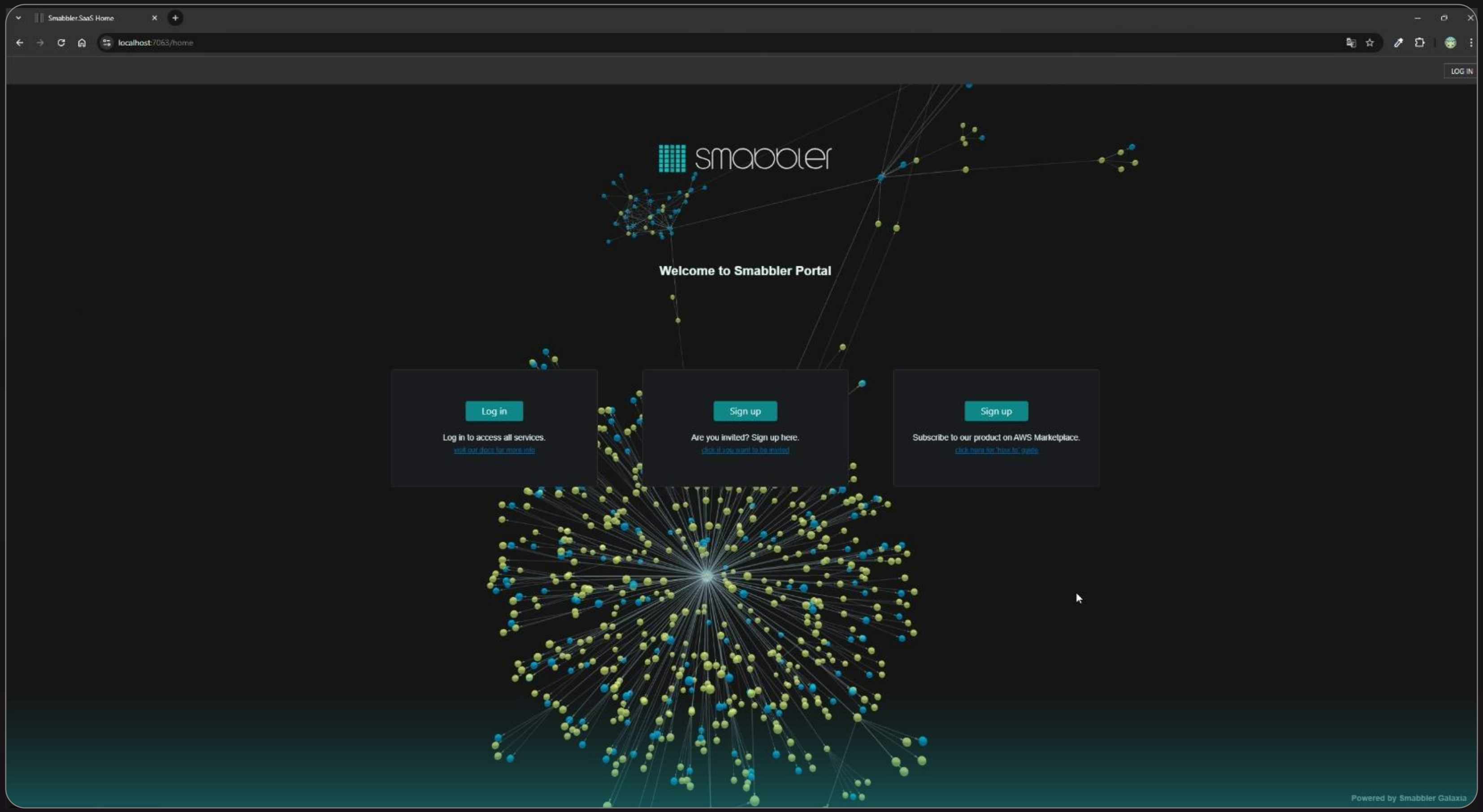
We create a new market standard for RAG, standing out with a combination of speed, accuracy, and cost efficiency.

	Smabbler	Other approaches
Use complexity	**A few clicks**	Multi-tool pipeline
KG[1] building	**1-click**	Hard coding
KG building speed	**Minutes to hours**	Weeks
Database	**No (in-memory graph)**	Graph / Vector DB
Retrieval	**Graph model**	Database
Processing	**CPUs**	CPUs / GPUs

[1]Knowledge Graph

Demo

You first RAG in minutes



Scan or **click** for demo

Narodowe Centrum Badań i Rozwoju

Smabbler - oparta o autorską technologię mapowania wiedzy na grafie uniwersalna platforma automatyzująca klasyfikowanie komunikacji

Wartość dofinansowania: 3 889 817,89 zł
Całkowity koszt projektu: 6 318 369,88 zł

W ramach projektu opracowane zostanie narzędzie automatyzujące proces klasyfikowania zgłoszeń/zapytań otrzymywanych w ramach obsługi klienta. Platforma umożliwi całkowitą automatyzację procesu niezależnie od skali, stopnia złożoności oraz kontekstu (treści) otrzymywanych zapytań.

Projekt współfinansowany przez Narodowe Centrum Badań i Rozwoju w ramach Strategicznego Programu Badań Naukowych i Prac Rozwojowych INFOSTRATEG