

THE 23RD INTERNATIONAL SEMANTIC WEB CONFERENCE

November 11, 2024 – November 15, 2024

Live! Casino & Hotel Maryland

Slot 1: Introduction to RAG systems with LLMs and KGs

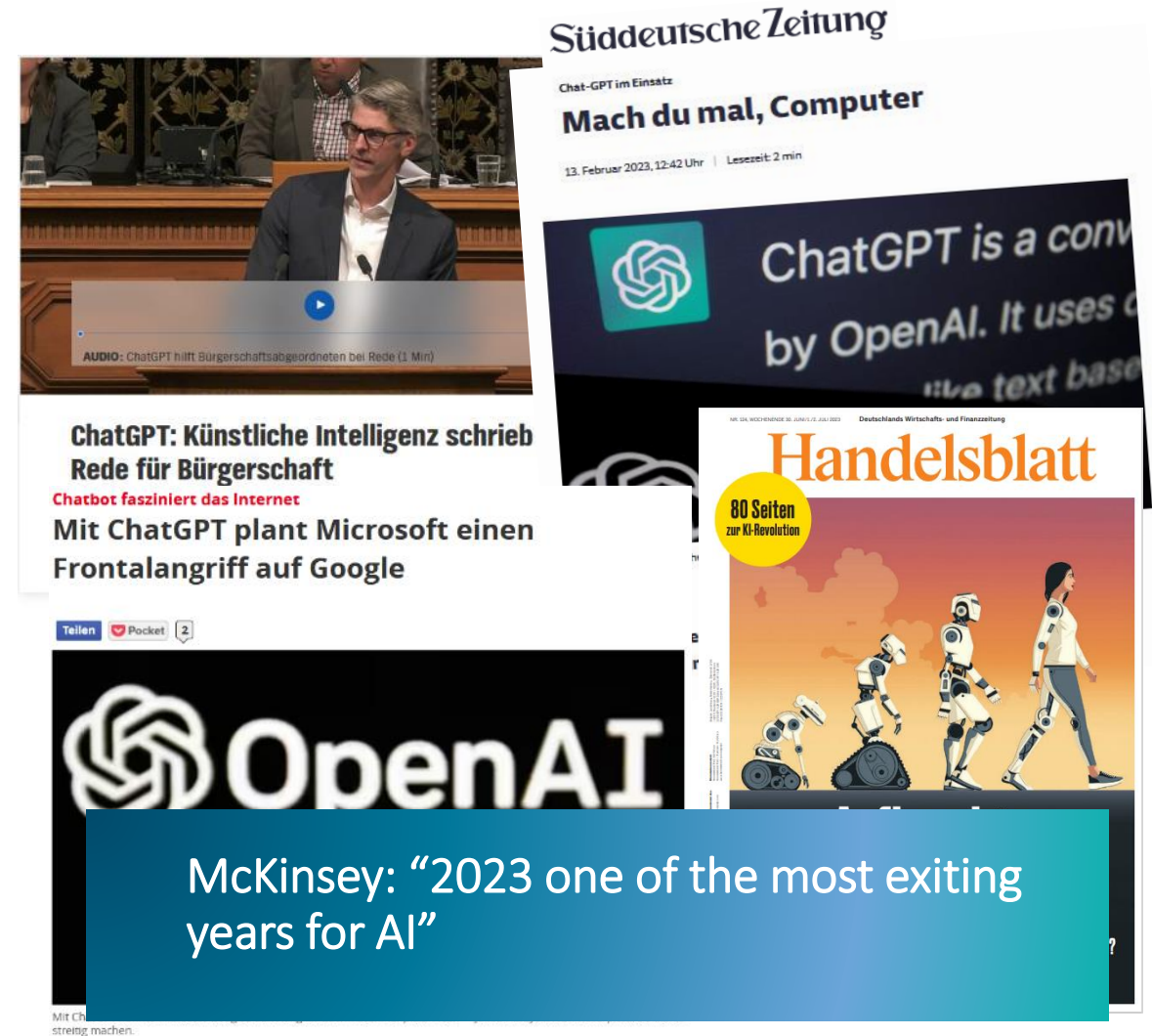
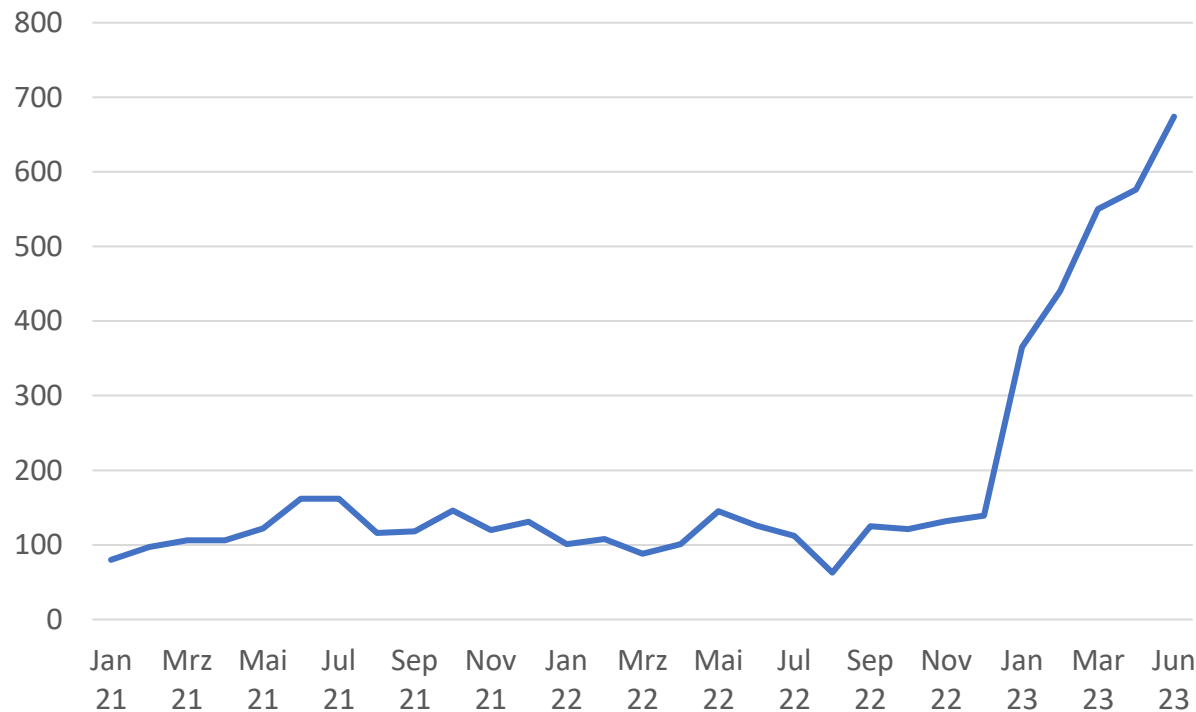
ISWC 2024

Part 1: Introduction

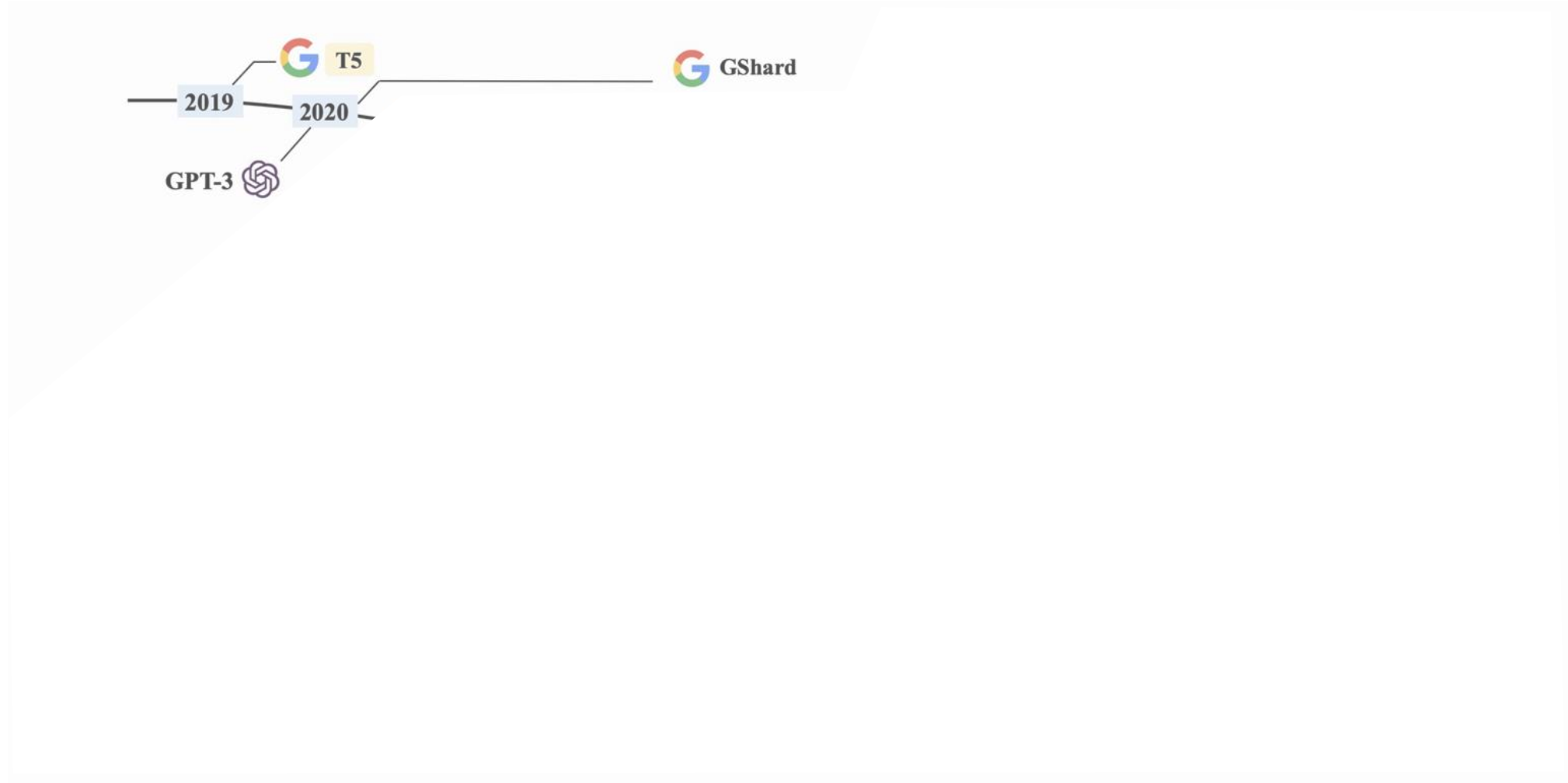
Large Language Models, Retrieval Augmented Generation (RAG), and Graph RAG

Artificial intelligence is the number one topic of conversation

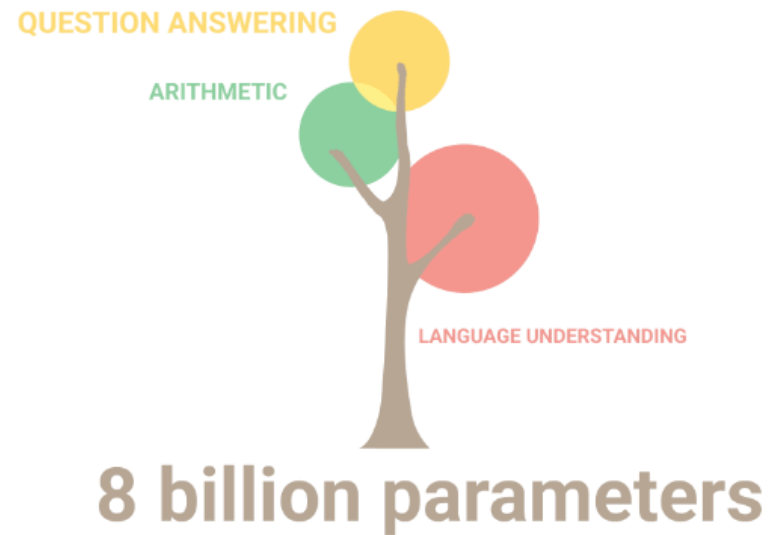
Media Coverage of AI



Large language models (LLMs) are taking the world by storm



The bigger the LLM, the more capabilities arise



The bigger the LLM, the more capabilities arise

Question answering

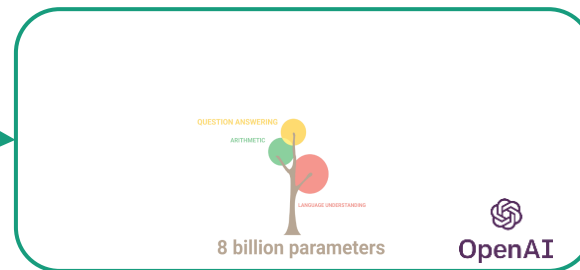
What is the capital of Germany?

Translation

Translate "Welcome to the workshop" into Spanish

Code generation

Write a Python function to calculate the Pythagoras theorem

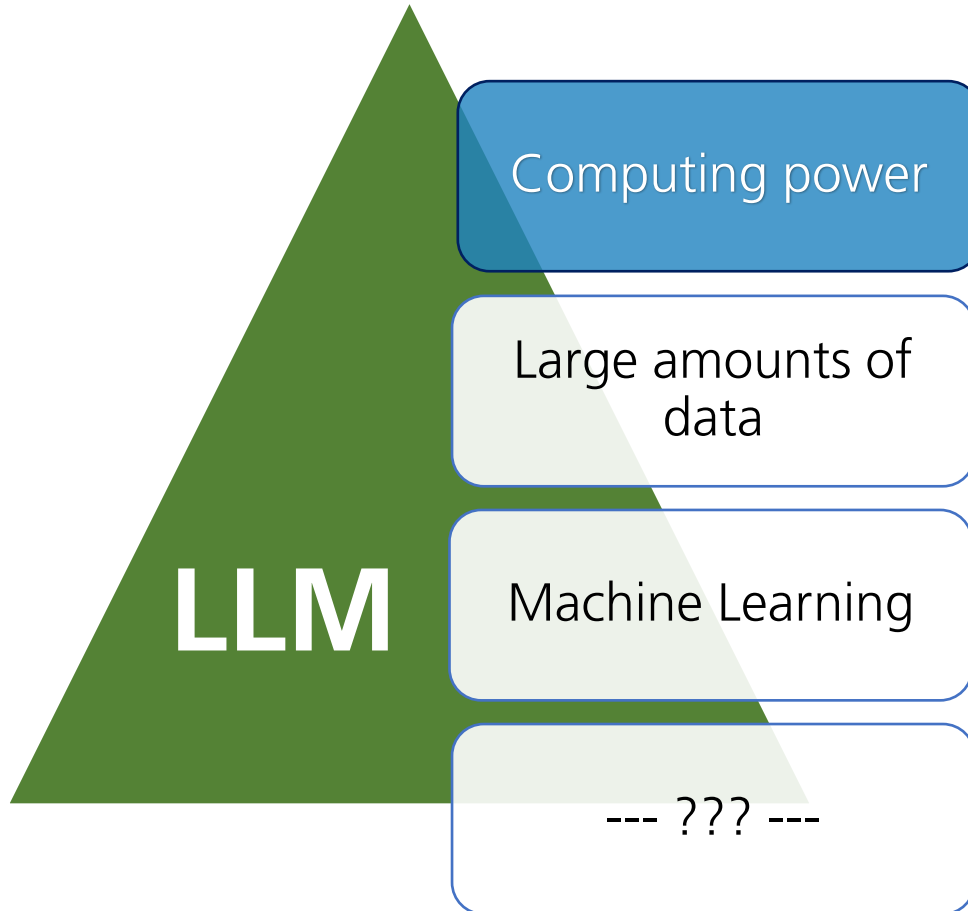


The capital of Germany is Berlin.

"Welcome to the workshop" translates to "Bienvenidos al taller" in Spanish.

```
def pythagoras_theorem( a, b):  
    c_squared = a**2 + b**2  
    c = math.sqrt(c_squared)  
    return c
```

The core elements of an LLM



GPT-3 175B model **requires O(1,000–10,000) V100 GPUs**

Costs O(1–10) \$M for a single training run

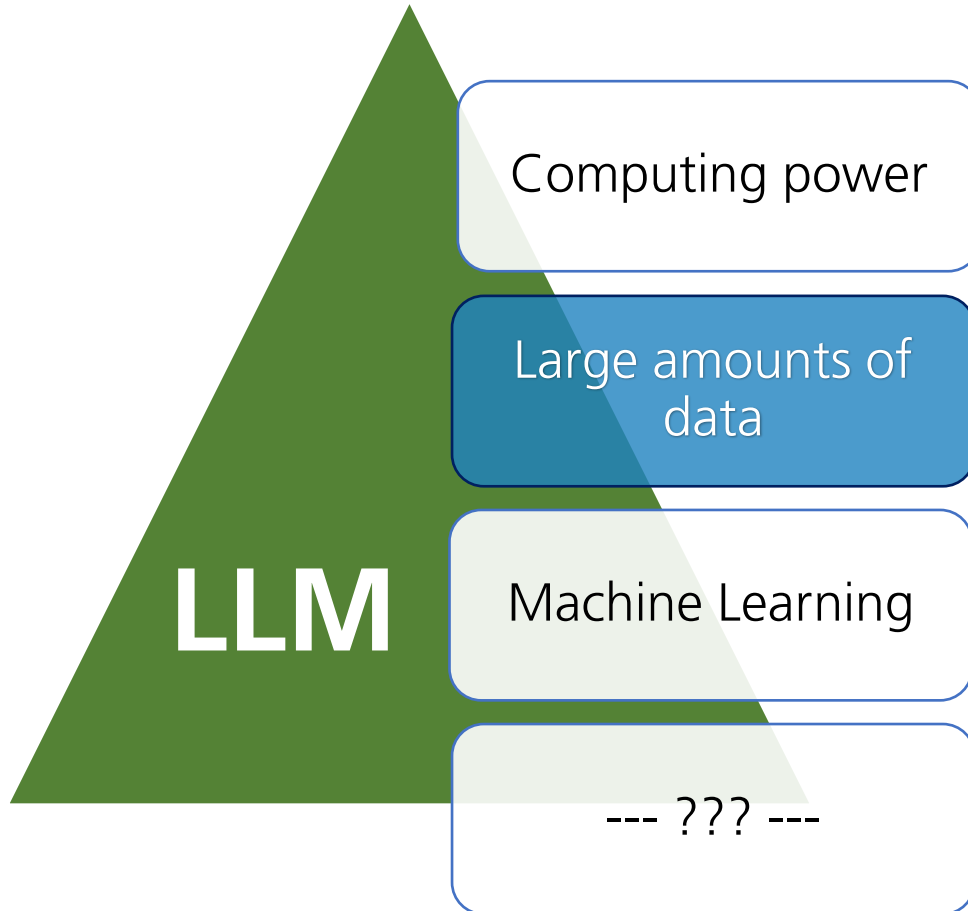
O(1) month of training

““The supercomputer built for OpenAI is a single system with more than 285,000 CPU cores, 10,000 GPUs, and 400 Gigabits per second of network connectivity for each GPU server,’ the companies explained in a blog post.”²

Sources:

1. <https://news.developer.nvidia.com/openai-presents-gpt-3-a-175-billion-parameters-language-model/>
2. <https://news.microsoft.com/source/features/innovation/openai-azure-supercomputer/>

The core elements of an LLM

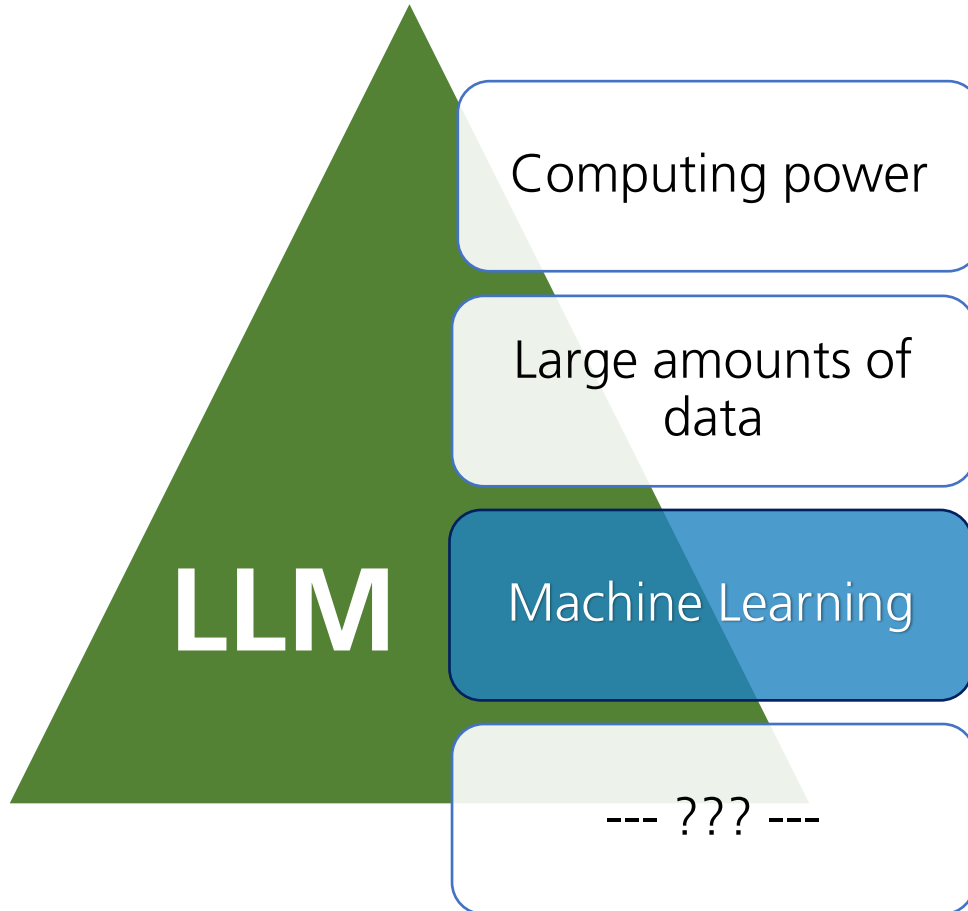


GPT-3 was trained on **570 GB of text**

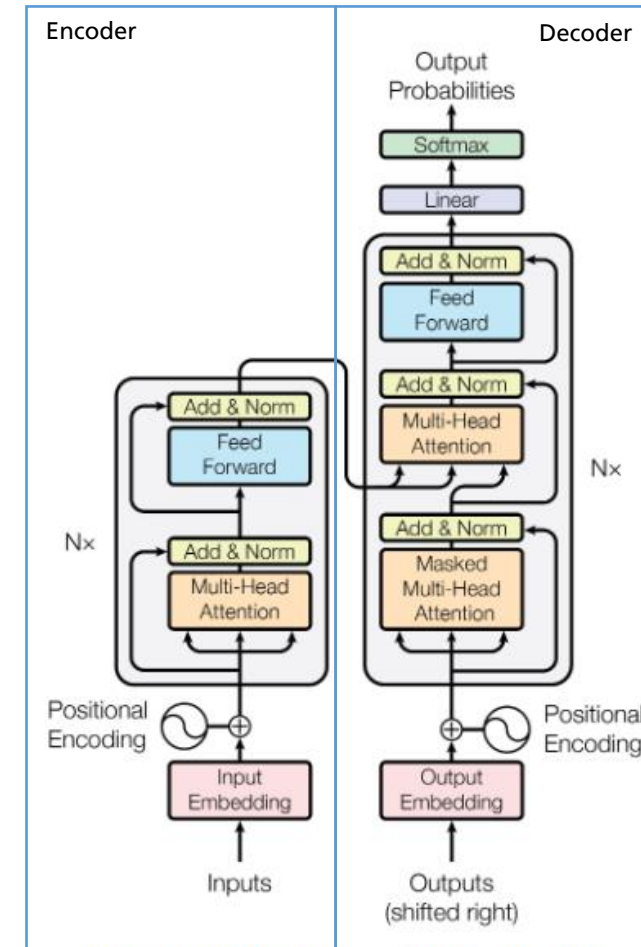
- Number of documents per language:

English	235.987.420
German	3.014.597
French	2.568.341
- Paper shows that the quality of the model depends on the size of the model, the amount and quality of data, and the number of training runs
- The texts contain a total of approx. **500 billion tokens** (note: 1 token != 1 word)

The core elements of an LLM



- BERT models:**
Prediction of masked tokens
- Powerful Embeddings



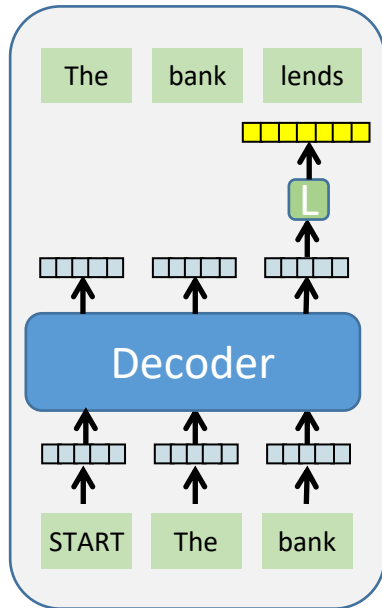
- GPT models:**
Prediction of the next token
- In-Context Learning via Prompting

Figure 1: The Transformer - model architecture.

Quelle: [Vaswani et al.](#) – Attention is all you need (2017)




Adaption via Prompting

GPT language models respond to instructions



- A GPT language model is given a **starting text**
 - It generates a continuation that is syntactically correct and plausible in terms of content
 - If the starting text contains a **direct instruction**, the corresponding response is generated
 - Self-attention only on **previous** tokens
- The GPT language model can be taught how to solve a task using one or more **examples**.
- Through **fine-tuning** (e.g., RLHF) the LLM can be trained to
 - avoid **abusive** statements
 - fulfil any request **correctly** and accurately
 - respond to the dialogue partner and make the dialogue **interesting**

Explain to an 8 year old child how a large language model works






A large language model is like a super smart robot that can understand and write words. Here's how it works:

1. **Learning from Books:** Imagine if the robot read millions of books, stories, and articles. It learns all the different ways people use words.
2. **Patterns:** While reading, the robot notices patterns. For example, it learns that "The cat is on the" is often followed by "mat" or "roof."
3. **Guessing:** When you ask the robot a question or give it a sentence, it tries to guess the best words to use based on what it learned.
4. **Talking Back:** Finally, it puts those words together to give you a response that makes sense!

So, it's like a really good friend who has read a lot and can help you with your questions or stories!

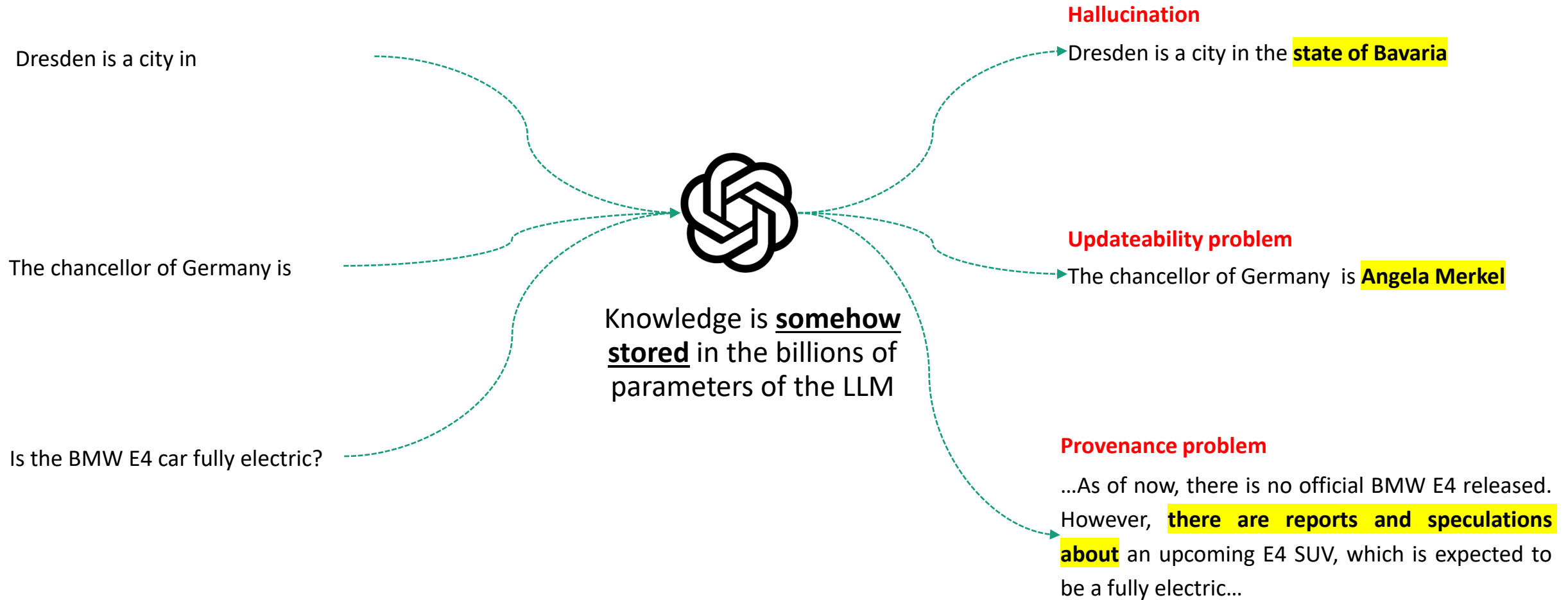
Chat: GPT 4o Mini ▾ Temp: 0.7 ▾ Stream ↺ ⚙️ ↶ ⬇️ 🗑️ 🌐

Tweet: "I hate when my handy runs out of battery" - Sentiment: negative. ### Tweet: "Today is everything wonderful" - Sentiment: positive. ### Tweet: "The new Taylor Swift music video is amazing" - Sentiment:



positive

Limitations of Large Language Models

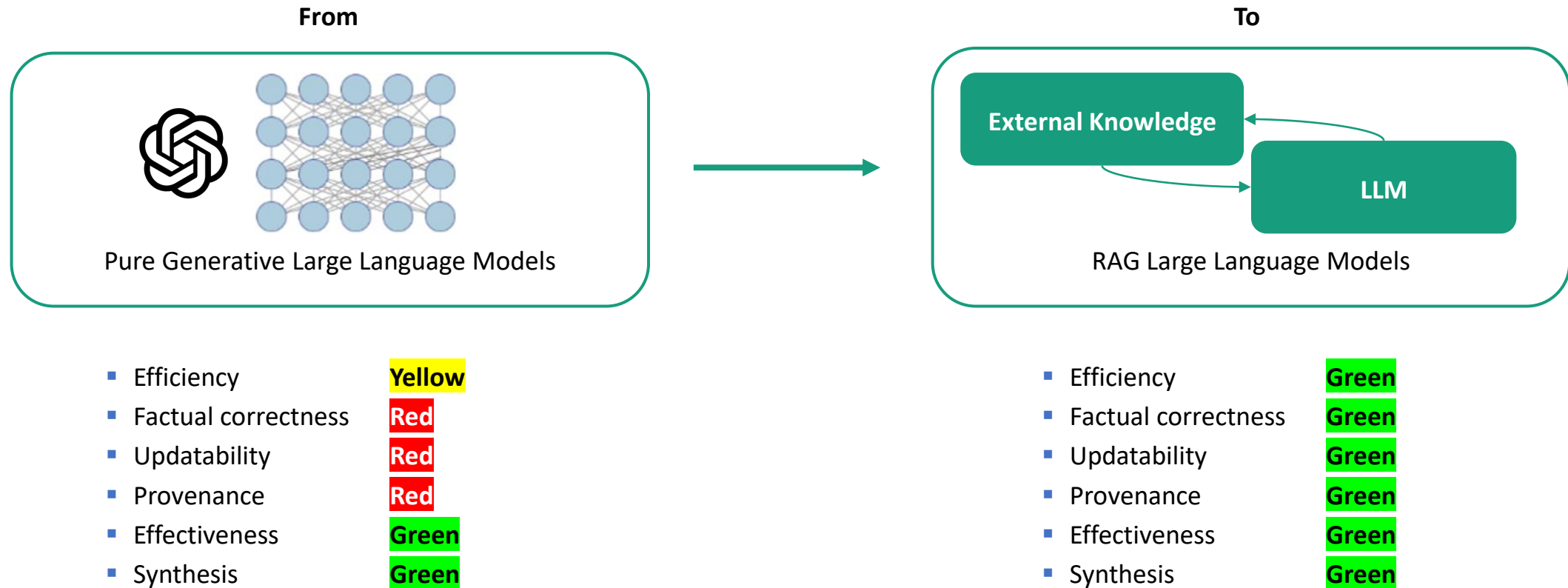


Retrieval Augmented Generation

To the rescue

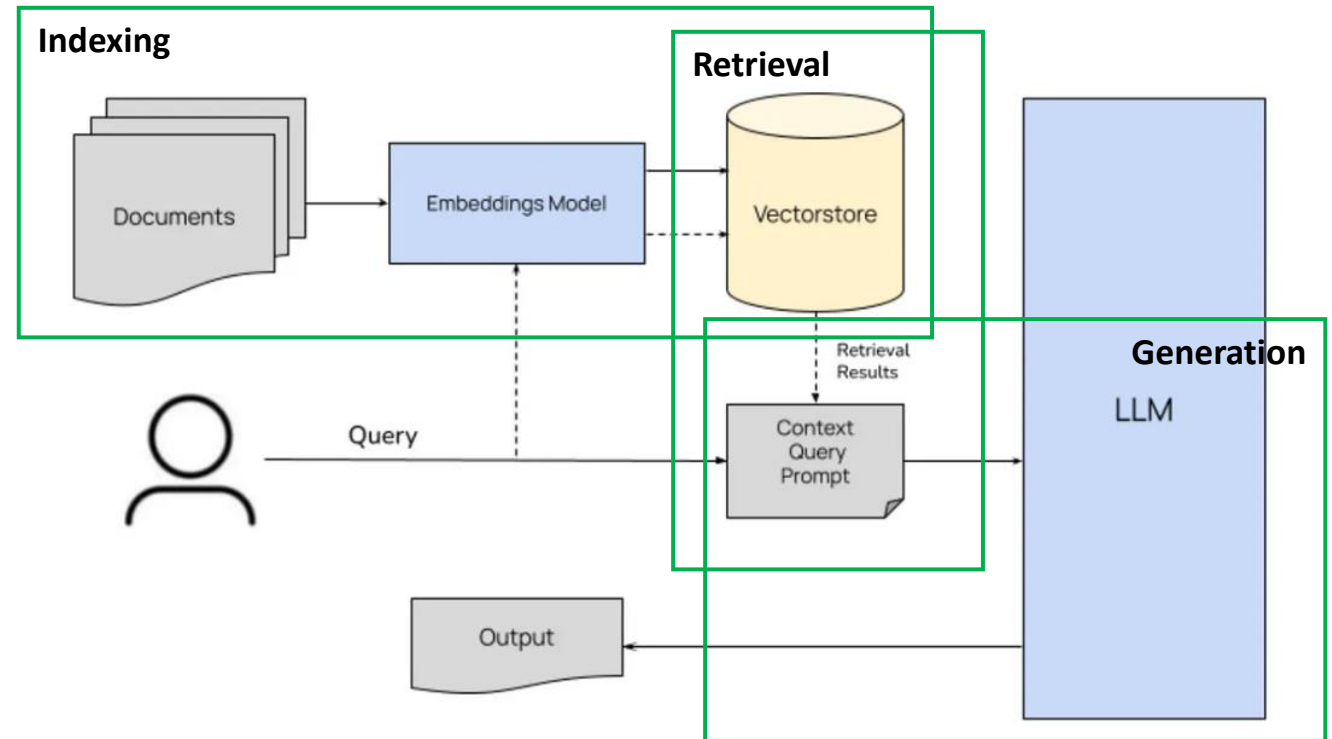
Retrieval Augmented Generation

Separate Generative AI into two components, i.e., Knowledge Store and Linguistic Capabilities



Retrieval Augmented Generation (Vector RAG)

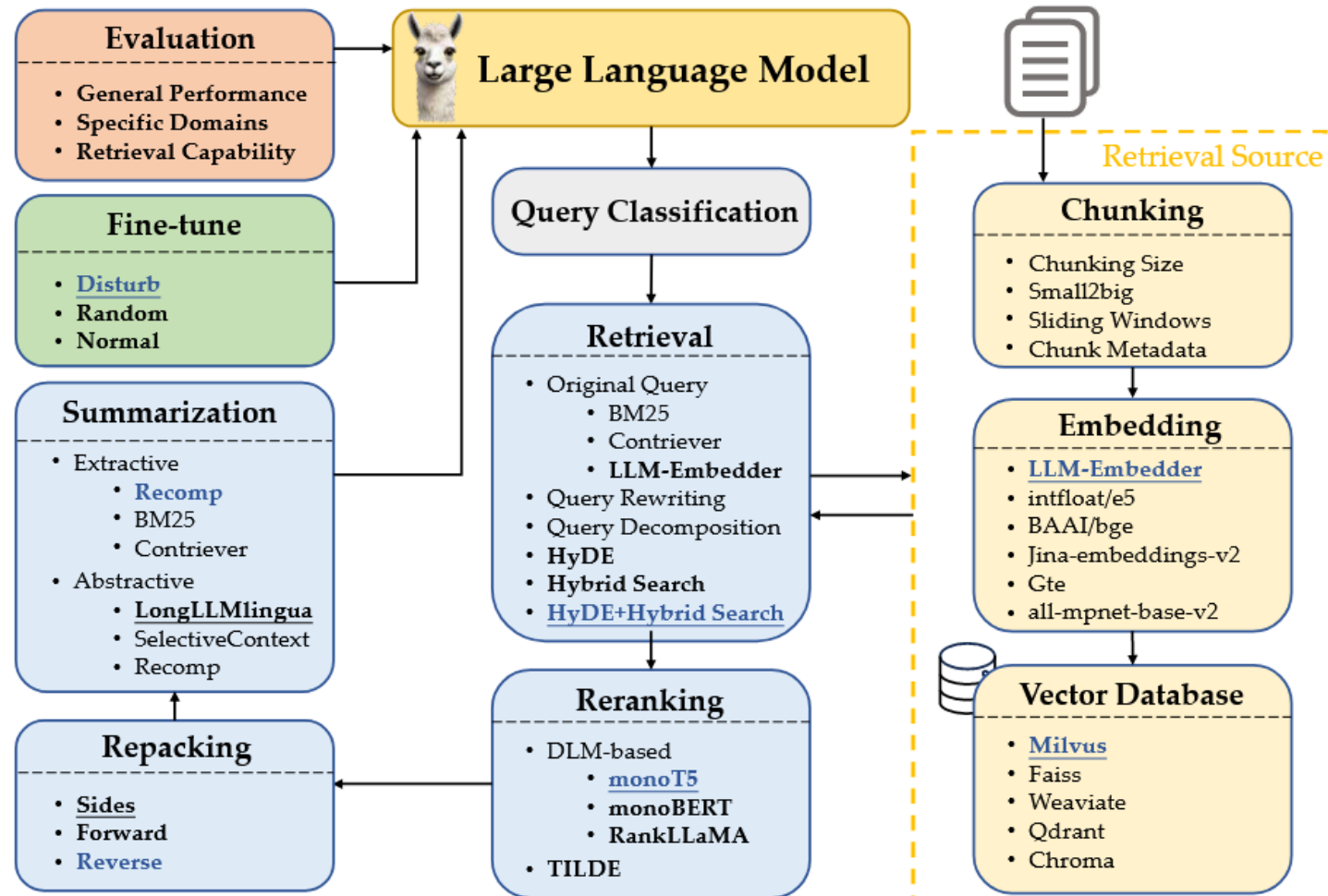
- Factual grounding
- Mitigating hallucinations
- Parser and Chunking impact the quality
- Dependent on quality and accuracy of the retrieval
- Latency concerns with vast datasets



A typical RAG system [6]

Retrieval Augmented Generation (Vector RAG)

In practices RAG gets more complicated



Limitations of Vector RAG

- **Limited Contextual Understanding:** Struggling to understand the context of queries and the relationships between different pieces of information
- **Difficulty in Handling Global Queries:** Hard to answer global queries that require understanding information spread across an entire dataset
- **Inefficient for Complex Search and Content Management:** Handling multiple language variants, version control, and fine-grained content filtering can be challenging using traditional vector-based RAG
- **Chunking and Context Loss:** Breaking documents into documents can lead to the loss of important contextual information
- **Limited Reasoning Capabilities**
 - Multi-hop reasoning: Navigating multiple relationships to arrive at an answer
 - Abstractive reasoning: Answering with knowledge not explicitly present in the text requires deeper reasoning and understanding

What are the side effects of medications used to treat diseases like asthma?



Answering this question accurately requires identifying medications for asthma, finding similar diseases, and then identifying side effects of treatments for those diseases—a multi-step process difficult for conventional RAG models

Graph RAG

To the rescue

Graph RAG - Knowledge Representation, Vectors & Graphs

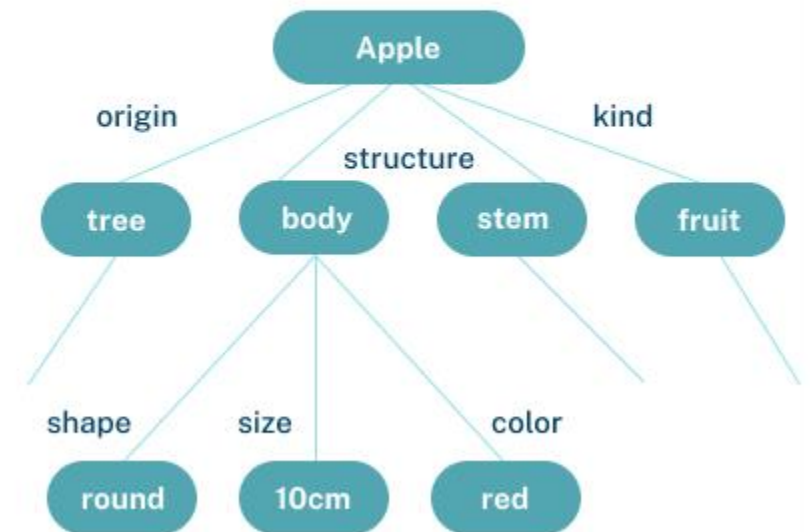
**Human View of
an Apple**



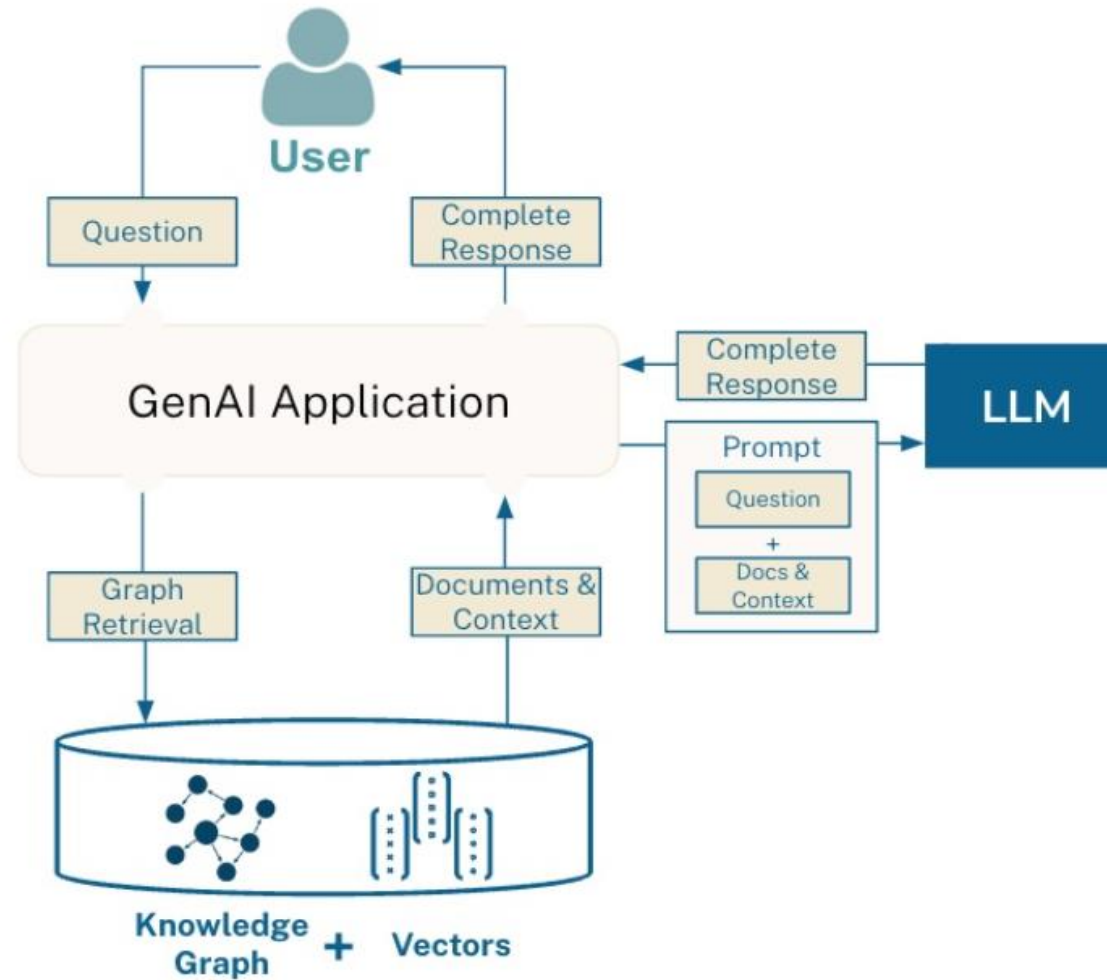
**Vector View of
an Apple**



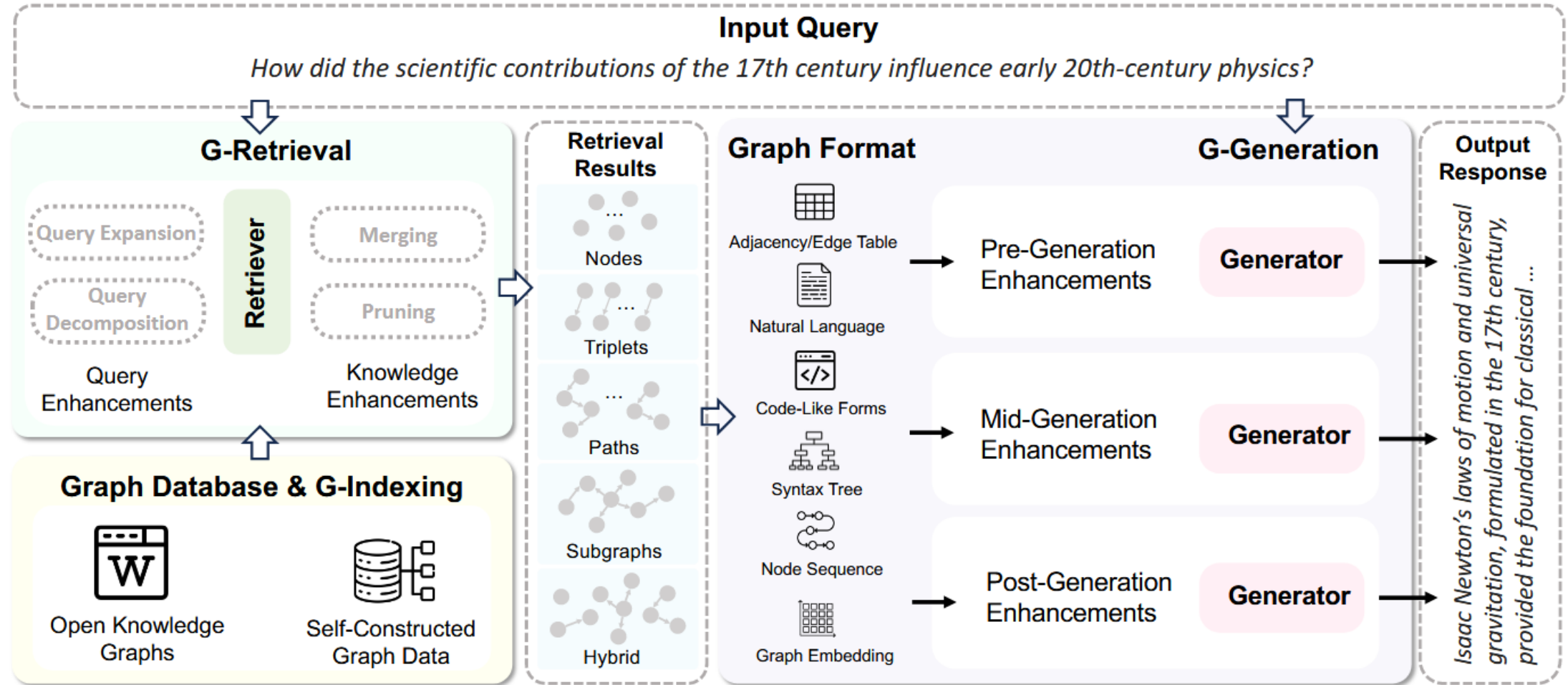
**Knowledge Graph
View of an Apple**



Graph RAG - A typical Graph RAG system



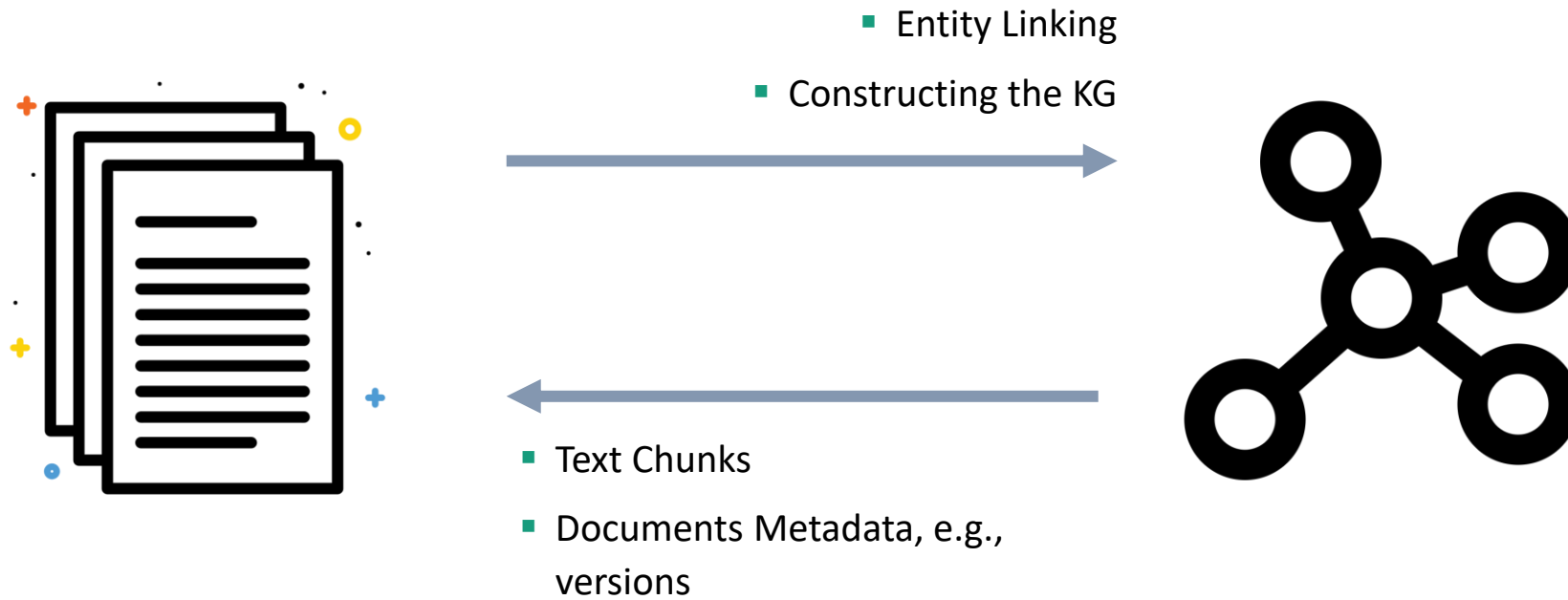
Graph RAG



Part 1: G-Indexing

Creating a Knowledge Graph with LLMs for RAG systems

Graph Indexing



Typical Pipeline



LlamaIndex approach - Knowledge Graph Index



- A specialized index designed to handle KG construction from unstructured text, enabling entity-based querying.
- **Builds a KG in several steps:**
 - Load Your Data: Using LlamaIndex's data connectors
 - Instantiate KnowledgeGraphIndex object. You can pass your loaded documents, specify the maximum number of triplets per chunk
 - Configure Storage Context: LlamaIndex provides simple in-memory storage or integration with external graph databases
 - Run the Indexer: This process will automatically extract relationships, create nodes, and build the knowledge graph.
- It allows **manual addition of triplets** to enhance or refine the automatically constructed KG

Building the Knowledge Graph

```
from llama_index.core import SimpleDirectoryReader, KnowledgeGraphIndex
from llama_index.core.graph_stores import SimpleGraphStore

from llama_index.llms.openai import OpenAI
from llama_index.core import Settings
from IPython.display import Markdown, display
```

INFO:numexpr.utils:NumExpr defaulting to 8 threads.

```
documents = SimpleDirectoryReader(
    "../../../../../examples/paul_graham_essay/data"
).load_data()
```

```
# define LLM
# NOTE: at the time of demo, text-davinci-002 did not have rate-limit errors
llm = OpenAI(temperature=0, model="text-davinci-002")
```

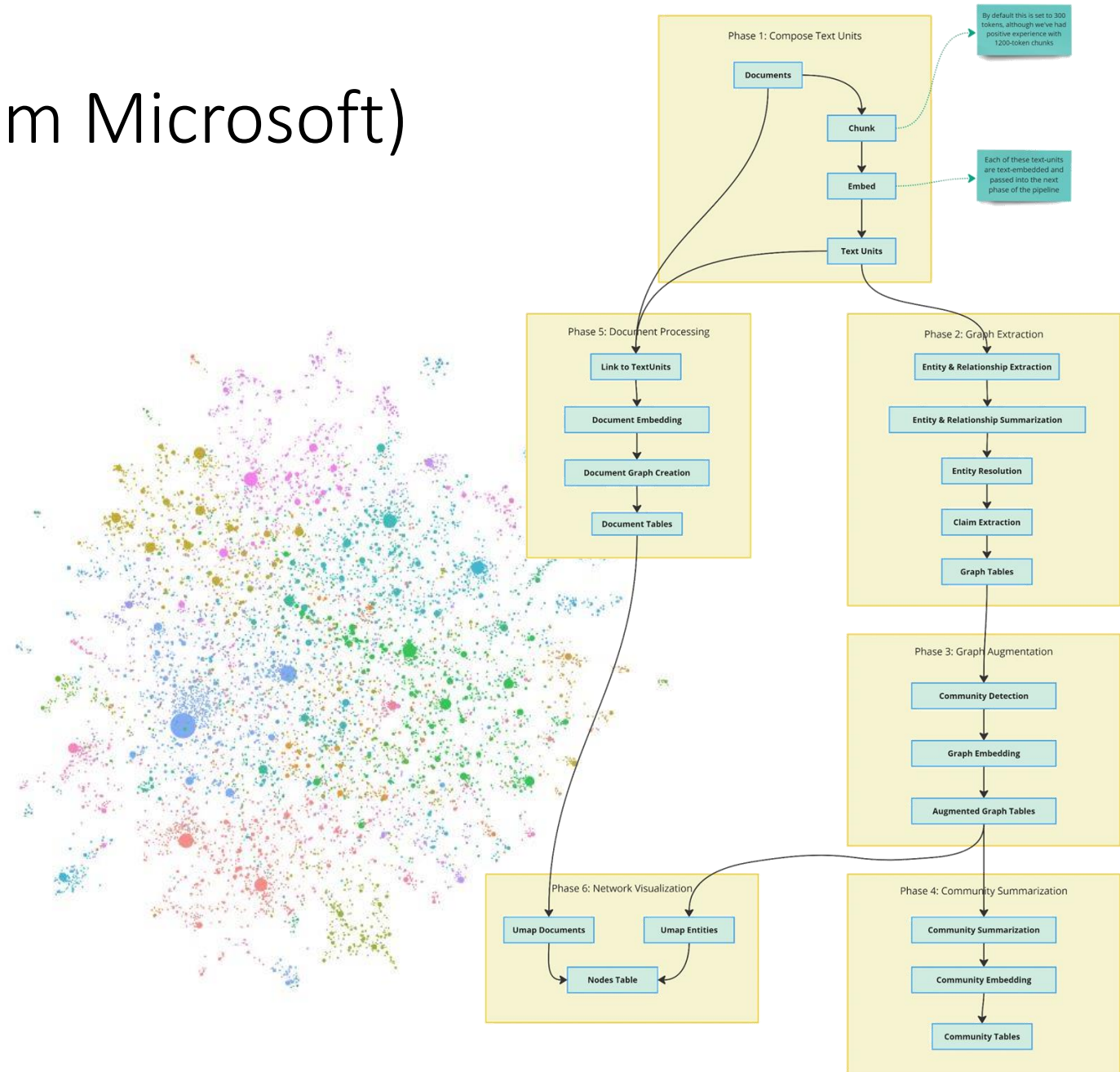
```
from llama_index.indices.property_graph import SchemaLLMPPathExtractor

entities = Literal["PERSON", "PLACE", "THING"]
relations = Literal["PART_OF", "HAS", "IS_A"]
schema = {
    "PERSON": ["PART_OF", "HAS", "IS_A"],
    "PLACE": ["PART_OF", "HAS"],
    "THING": ["IS_A"],
}

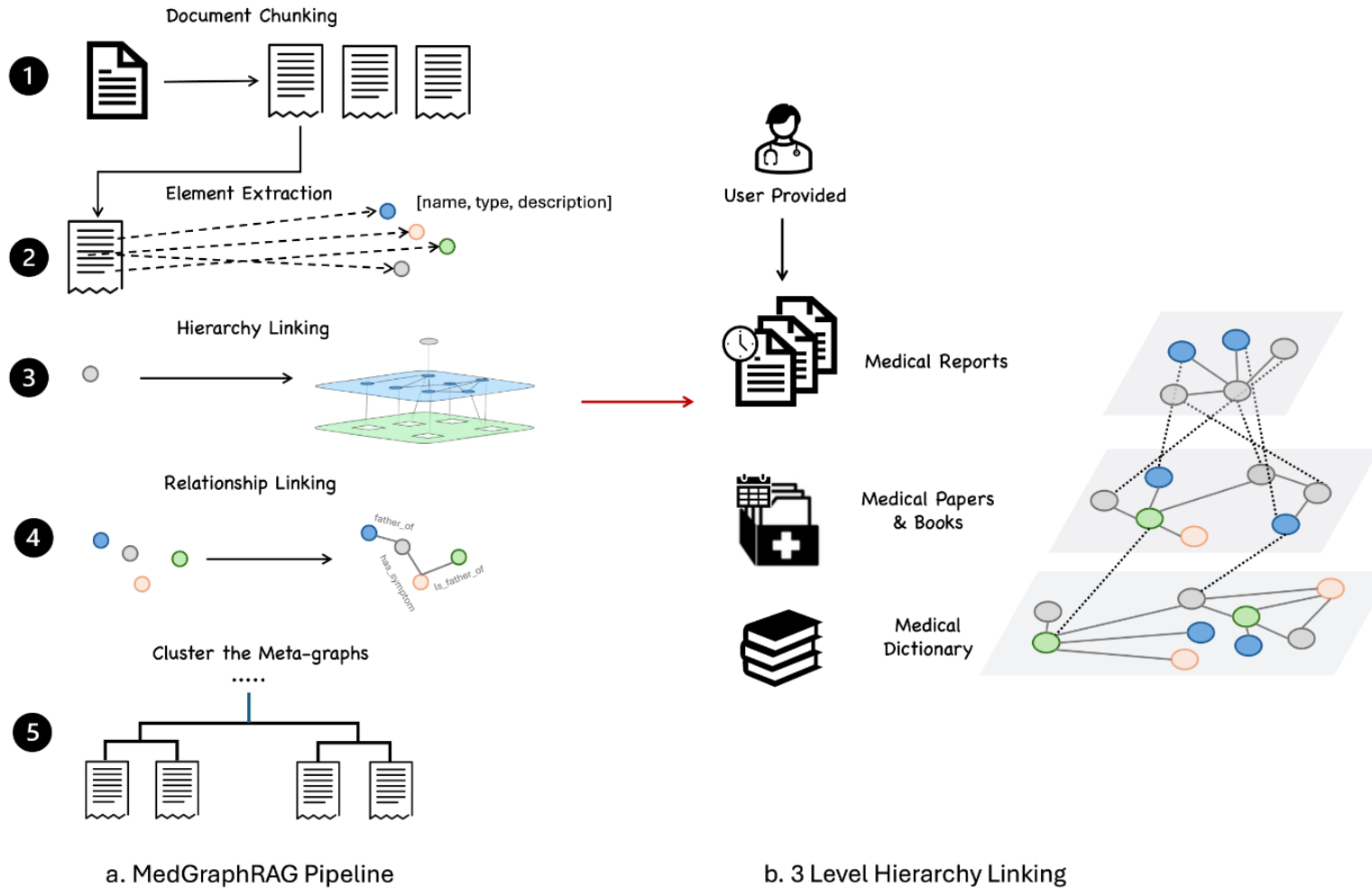
kg_extractor = SchemaLLMPPathExtractor(
    llm=llm,
    possible_entities=entities,
    possible_relations=relations,
    kg_validation_schema=schema,
    strict=True, # if false, allows values outside of spec
)
```


GraphRAG approach (from Microsoft)

More coming in session 3



Layered Knowledge Graph



Hands-On 1

Building a Knowledge Graph out of my repository of scientific papers

End of session 1
