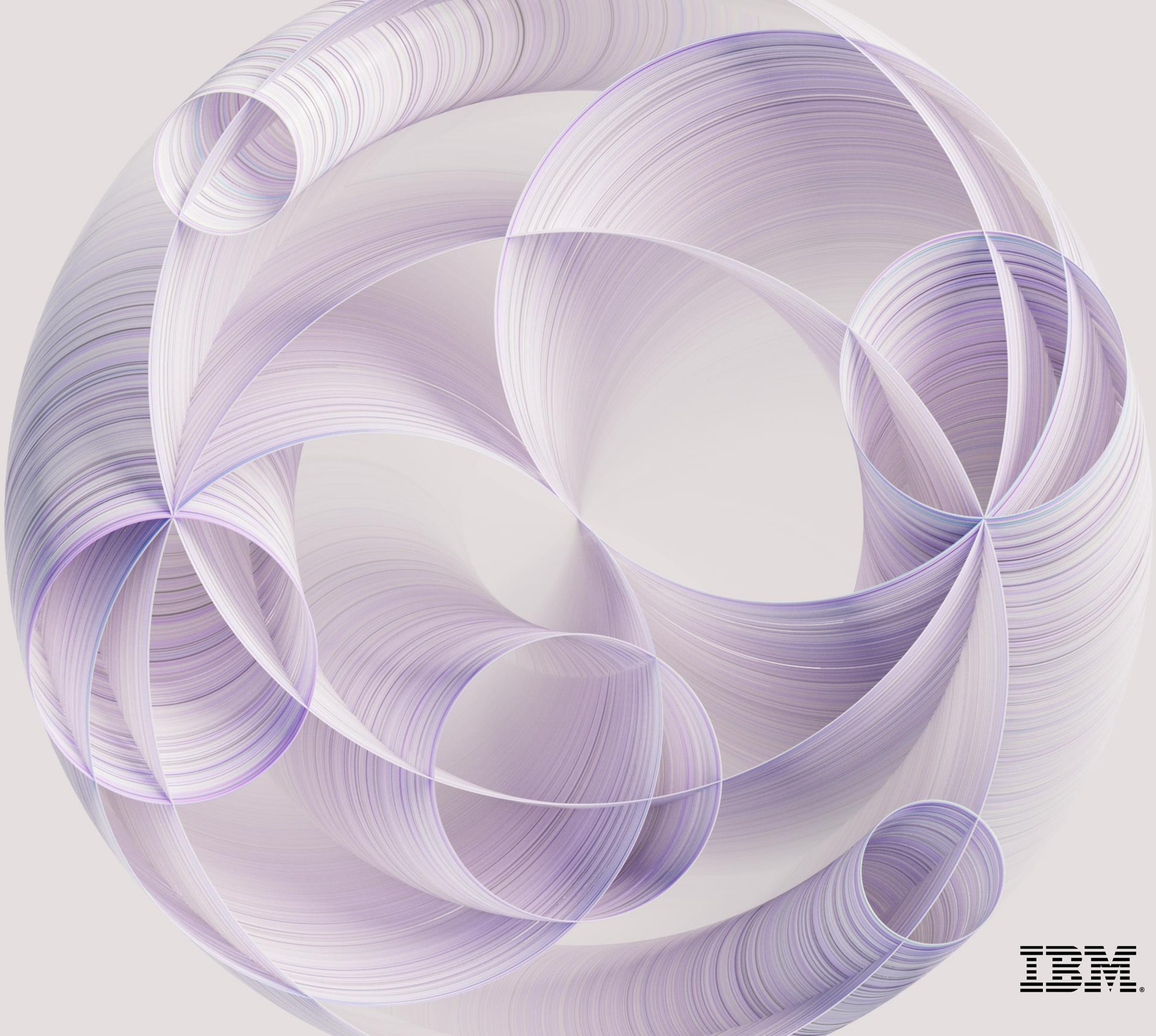
Watsonx.ai

Introduction to Retrieval Augmented Generation (RAG)



Felix Lee
Principal, Learning Content Development, AI and Data
felix@ca.ibm.com

Seller guidance and legal disclaimer

IBM and Business Partner
Internal Use Only

Slides in this presentation marked as **"IBM and Business Partner Internal Use Only"** are for IBM and Business Partner use and should not be shared with clients or anyone else outside of IBM or the Business Partners' company.

The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth, or other results.

All client examples described are presented as illustrations of how those clients have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by client.

Content

- Common generative AI issues
- Retrieval augmented generation (RAG) overview
- Phases of RAG
  - Data preparation
  - Data retrieval
  - Answer generation with a large language model
- Watson Discovery vs vector database
- RAG value proposition

Two common issues with large language models

Lack of information source

“The bank offers 2.5% interest on accounts with a balance over $20,000.00.”

This sounds great – but where did the information come from?

How can a user verify that this is true?

Where is it documented?

Outdated information

“Who is the highest-scoring player in the NBA?”

The Llam2-70b model returns:

”Kareem Abdul-Jabbar is the highest-scoring player in the NBA”.

This is an outdated answer as Lebron James broke that record in 2023.

This means that llama2-70b was trained on pre-2023 data.

Retrieval augmented generation (RAG)

RAG addresses these issues:

- Where did the LLM get its answer?
- Is the answer based on updated material?

RAG does this by:

- Working with “external data” (data not used for training the LLM):
  - Source of answers? From curated, validated, and accurate data
  - Currency of data? As current as the source
- NO model retraining required

A “human interaction” analogy of RAG is providing an update document to a person and asking them to answer question based on the information in the document.

Retrieval Augmented generation (RAG)

An AI framework for improving the quality of LLM-generated responses

Grounds a model on additional sources of knowledge to supplement its internal representation of information

RAG involves three basic steps:

1. Search for relevant content in your knowledge base
2. Pull the most relevant content into your model prompt as context
3. Send the combined prompt text to the model to generate output

---

Significantly elevates level of trust:

- Ensures that the model has access to the most current and reliable facts
- System becomes "business-aware"
- Sources are known, ensuring output can be checked for accuracy
- Less likely to make-up a factually inaccurate responses, with ability to say, "I don't know."

Retrieval Augmented Generation components

Knowledge Base

Can be any collection of information containing artifacts such as:

- Internal procedural wiki pages
- Files in GitHub (various formats)
- Messages in a collaboration tool
- Topics in product documentation
- PDF files
- Customer support tickets
- more

Retriever

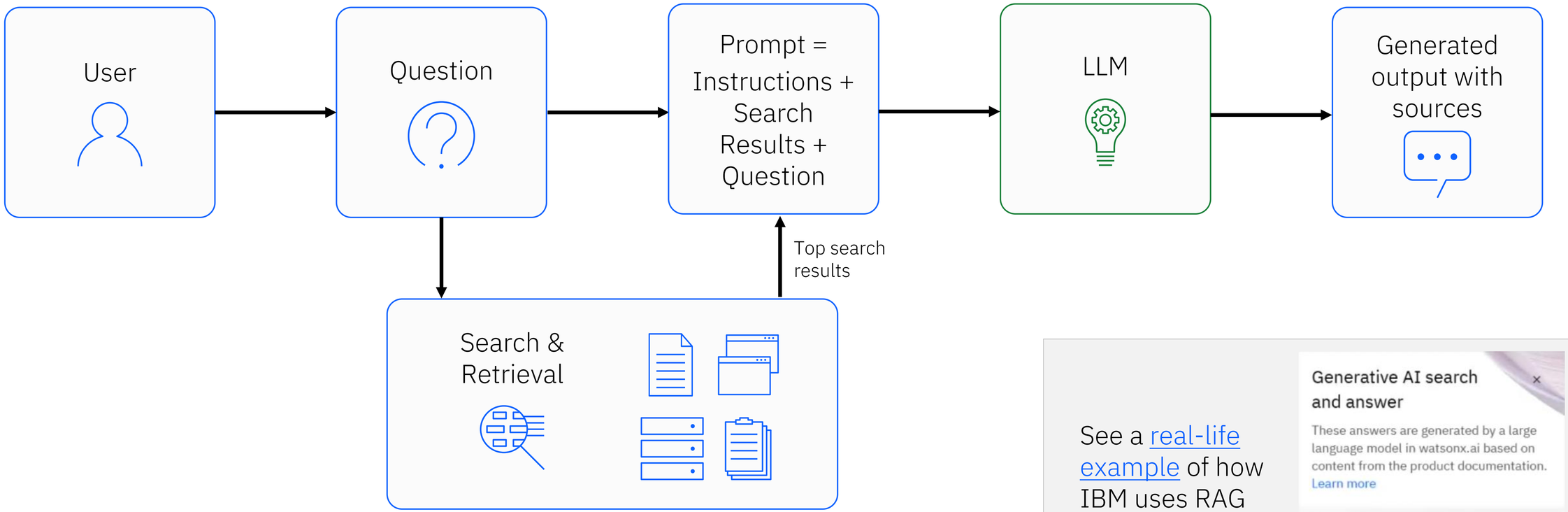
Can be any combination of search and content tools that reliably return relevant content from a knowledge base (or bases):

- Search tools like IBM Watson Discovery
- Search and content APIs like GitHub APIs
- Vector databases like Milvus

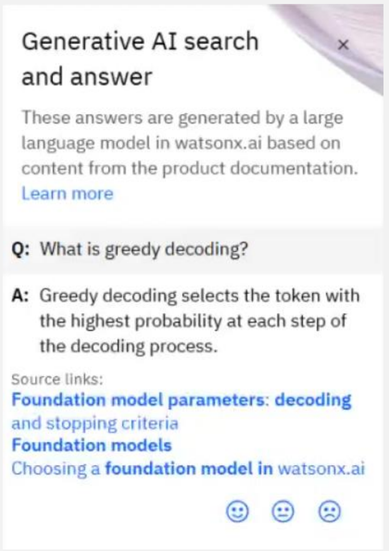
Generator

A generative LLM (such as those found in watsonx.ai) that suits your use case, prompt format, and content being pulled in for context

Typical RAG process



See a real-life example of how IBM uses RAG to answer user questions about watsonx.ai in the product documentation



Retrieval Augmented Generation – Phase 1 – data preparation

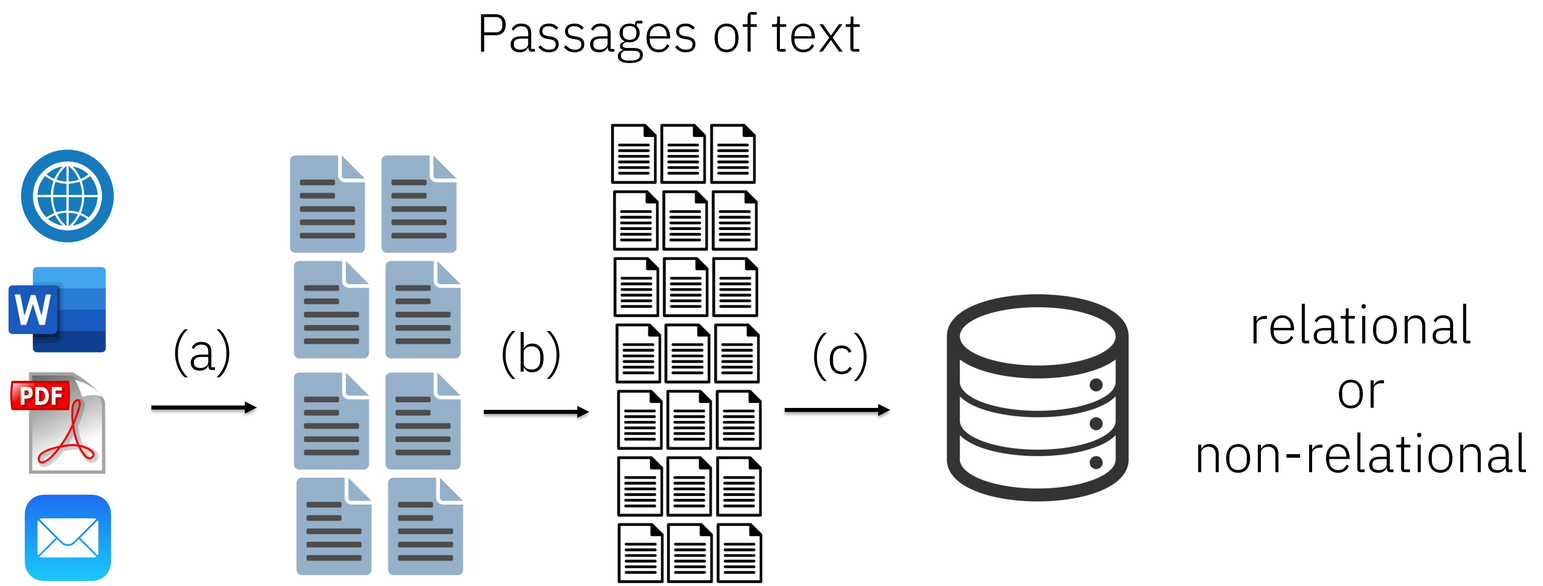
Validated, filtered, and curated data is stored. These can be files/data in Cloud Object Storage, Box, or other data repositories.

Clients can update them whenever necessary, ensuring that only the latest is used.

Data storage – the traditional way

Data ingestion

1. Original files to documents
2. Split documents into chunks
3. Store chunks to database



Data storage – using embedding and a vector database

Phase 1

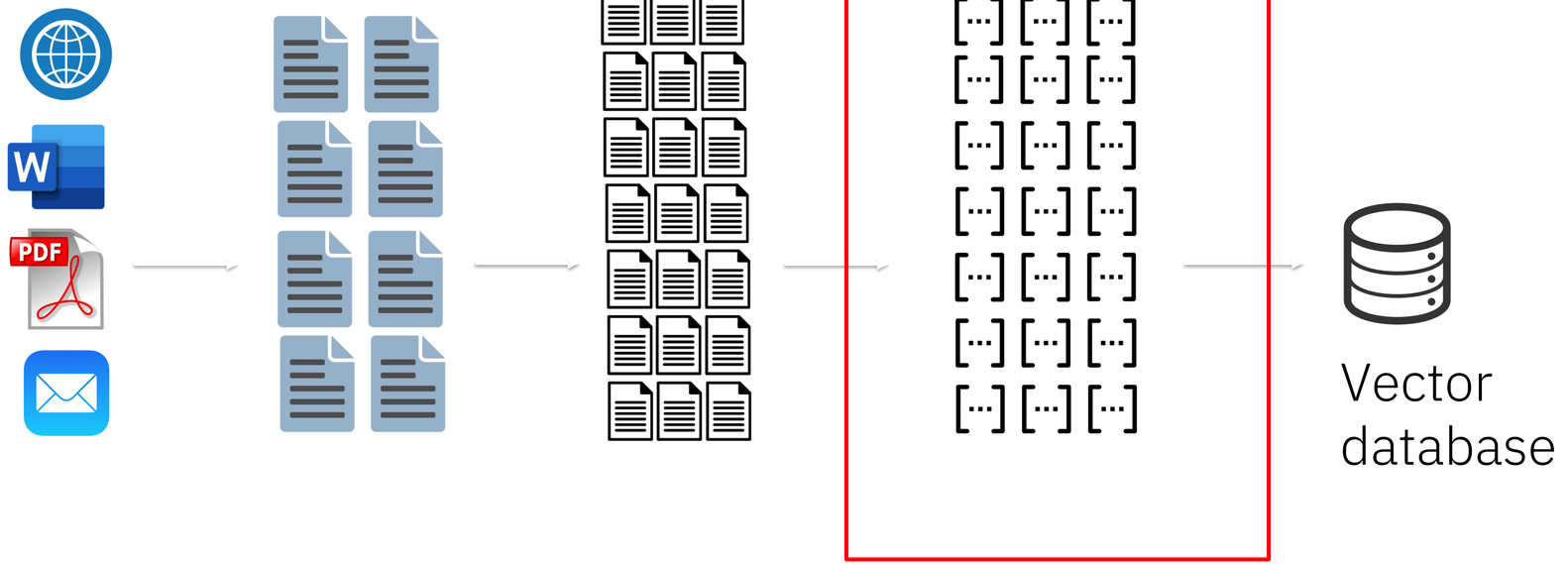
Ingest your data

(a) Original files to documents

(b) Documents to chunks

(c) Chunks to embeddings

(d) Embeddings to vector store



Semantic versus syntactic search

A user expresses themselves in their way, whereas the documents usually use “specialized” terms.

Examples:

Paid leave of absence
(IBM HR documents)

Corporate assets
(Bank’s code of ethics)

Revenue, profits, benefits
(10K form)

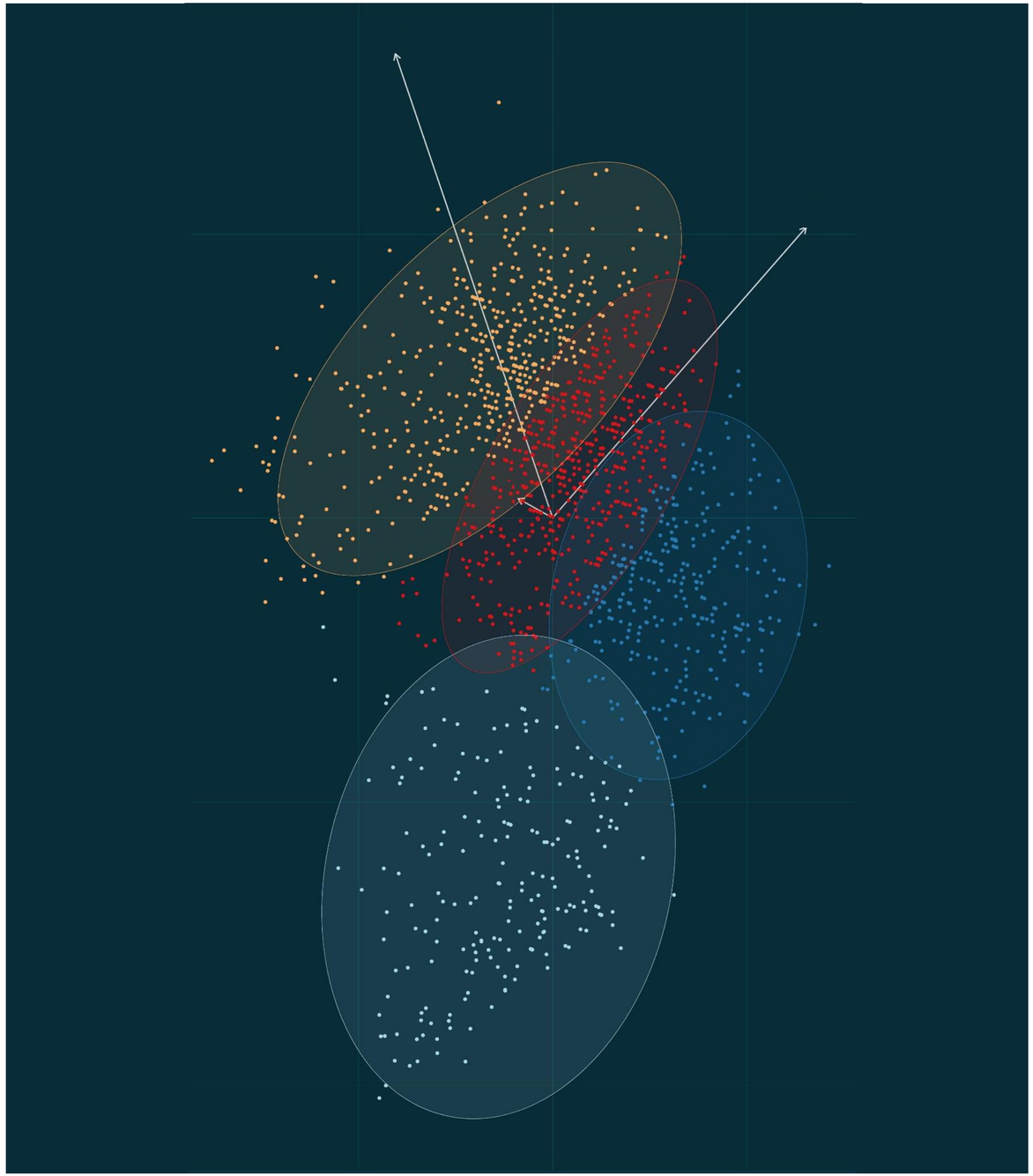
Day off

My MacBook

Money

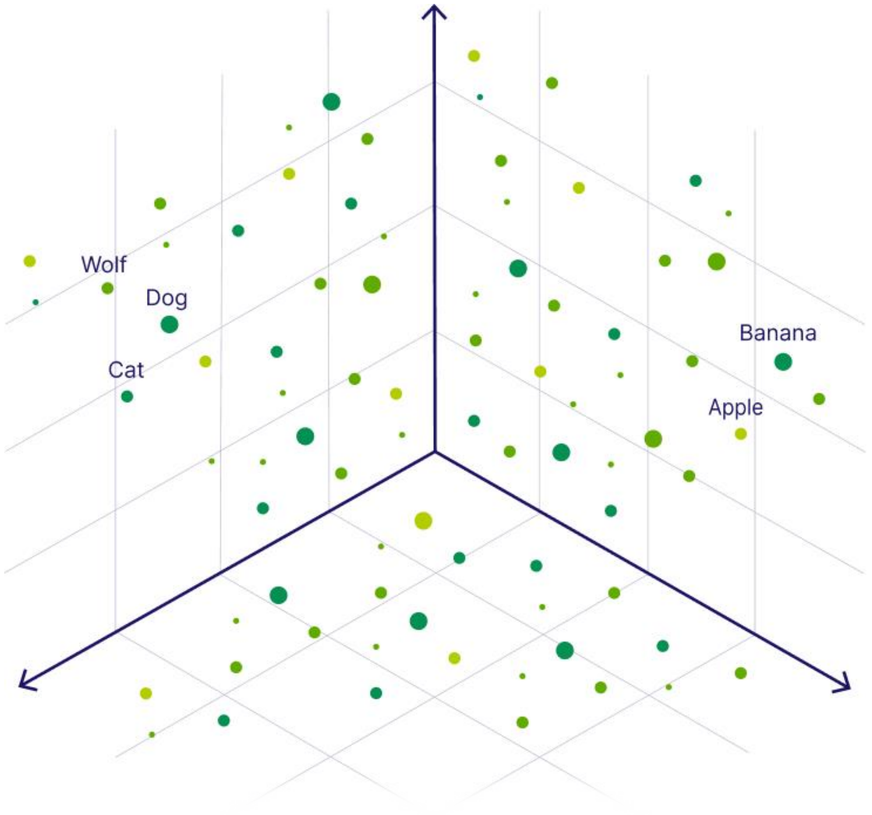
Vector databases

- Commonly used as the back-end knowledge base storage for RAG
- Designed to efficiently store, index, and retrieve vector embeddings (mathematical representations of data)
- Very good at finding things that are similar to each other (similarity search)
- Examples of vector databases include Milvus, Chroma, Pinecone, +++



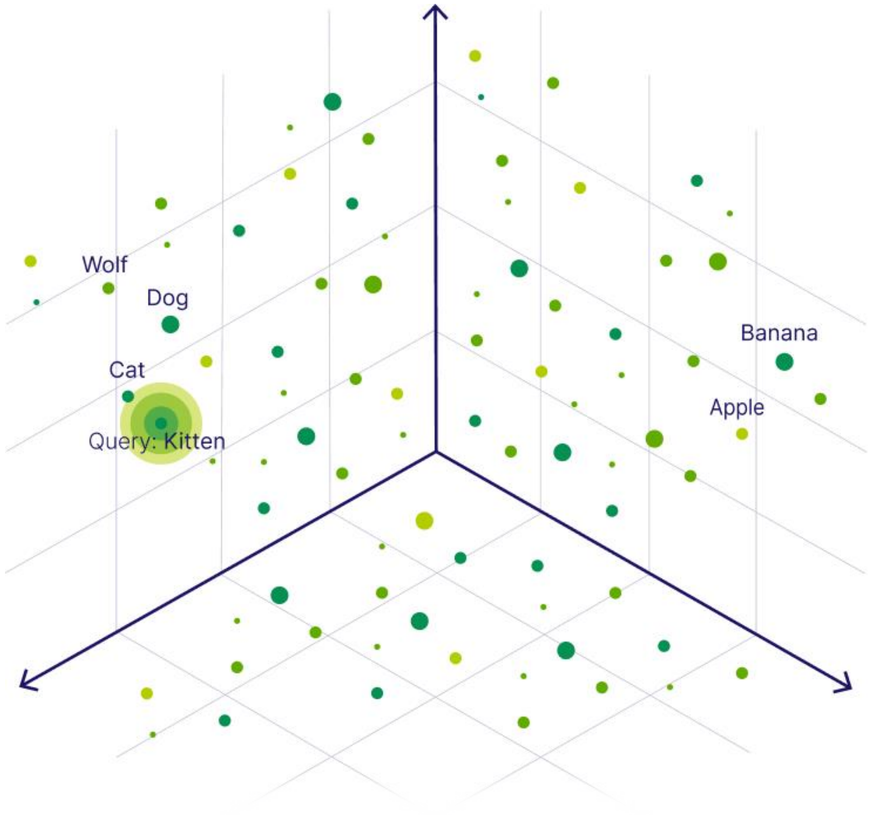
Similarity search

Example Word Vectors



What is "Kitten" similar to?

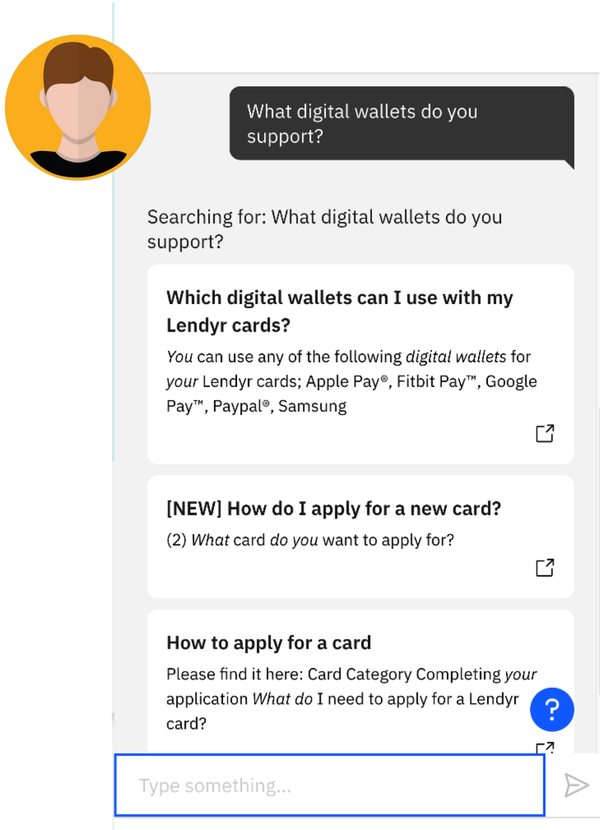
Finding Similarities



Common similarity measures: Cosine similarity, Euclidean distance, dot product

Source: https://weaviate.io/blog/what-is-a-vector-database

Retrieval Augmented Generation - Phase 2 – data retrieval



User's question

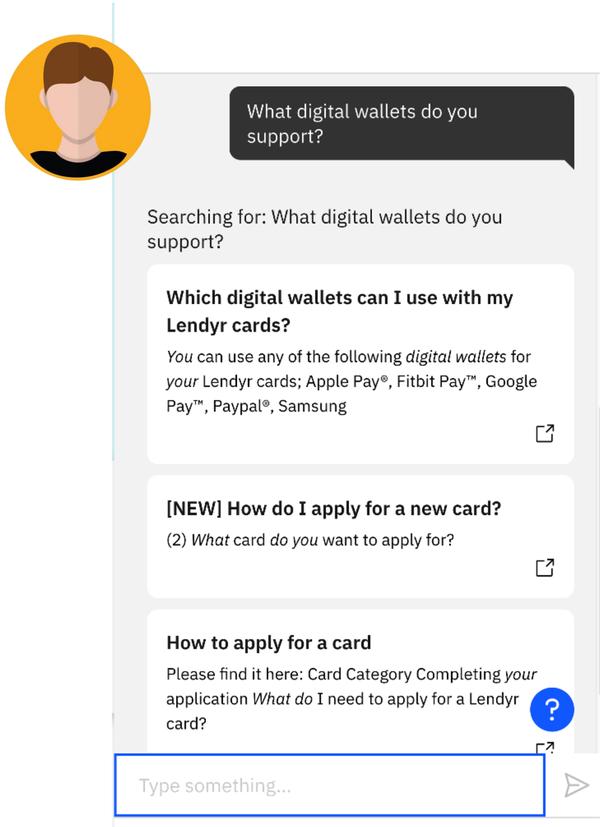
Few paragraphs related to the question

Can be data/information cataloged
Watson Discovery, or a vector database

Read the following paragraphs and answer the question:
[Paragraph 1]
[Paragraph 2]
[Paragraph 3]
Question: [User's question]
Answer:

watsonx.ai

Retrieval Augmented Generation - phase 3 – completion



User's question

Few paragraphs related to the question

Can be data/information cataloged in Watson Discovery, or a vector database

Read the following paragraphs and answer the question:
[Paragraph 1]
[Paragraph 2]
[Paragraph 3]
Question: [User's question]
Answer:

watsonx.ai

Retrieval augmented generation – 2 scenarios

1

User
What they do:
Upload new documents

2

Watson Discovery
What it does: Extract all textual content of the uploaded files

3

Conversion
What it does: Pull the last uploaded content, convert it into embeddings and save it to a vector store

Vector store (Example, Mulvis)
What it does: Store all the document's contents in the form of embeddings

4

User
What they do:
Ask a question

5

Conversational application
What it does: Build the conversation history and send it to a web service through a custom extension

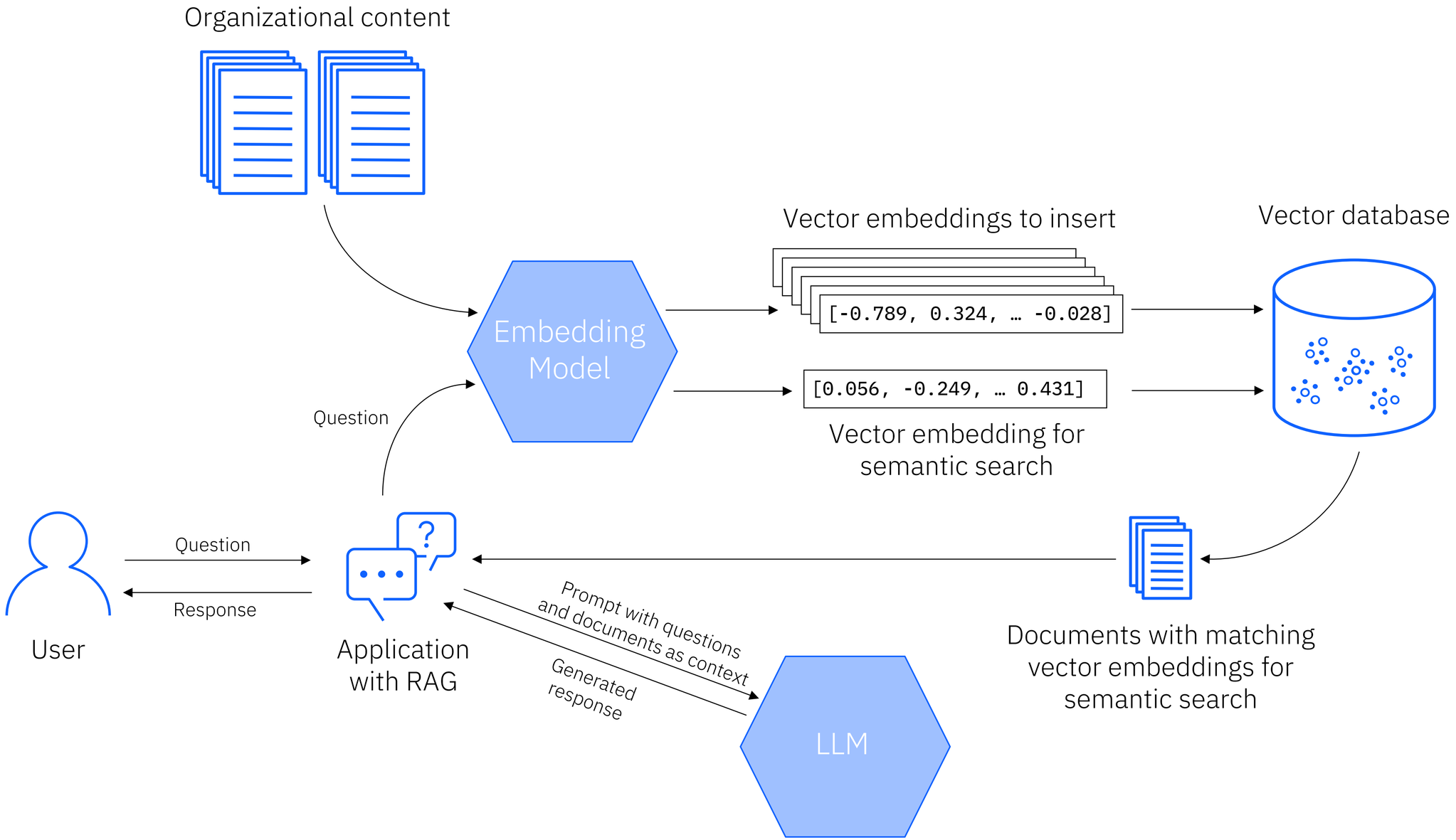
6

Web service
What it does: Query the vector store-based or other Watson Discovery on the conversation history, retrieve the most appropriate snippets of texts, and use an LLM hosted on watsonx.ai to formulate the response to the user

7

watsonx.ai
What it does: Formulate a user-friendly response

Retrieval Augmented generation (RAG) with a vector database



Watson Discovery versus a vector database in a PoX

Watson Discovery

- Easy to set up
  - Put documents in COS
  - Create a collection for the document
- Smaller set of documents
- Not as efficient if the collection is big
- Sufficient to illustrate the RAG concept

- Implementing RAG use case

Vector database

- Harder to set up
  - Embedding function
  - Vector database
- A large set of documents
- More efficient if the collection is big
- Consider this if the client insists on testing with a high volume of info
- Potential integration with watsonx.data (via Milvus)

Retrieval augmented generation - applications

Summarizing large document

Problem:
Large corpus, potential loss of info and expensive process

Solution:
Leverage RAG for questioning and collect results for summarization

API search agent

Problem:
Querying and processing large corpus of YAML files to feed the LLM

Solution:
Query processed YAML file for Q&A using RAG

Whisper bot

Problem:
Automatic RAG search from the user's historical chat conversations

Solution:
Real-time processing of the user's chat history and utilizing RAG to get the relevant info for the next steps

Retrieval augmented generation – value proposition in PoX

Business relevance

- Clients can specify what documents are searched for answers to ensure that the LLM will respond using trusted and relevant business information

Source of truth

- A client's RAG implementation can easily include the sources of information
- Easy for a human to verify the accuracy

Information currency

- Clients can be sure that the most current information is used to generate the LLM's response

Minimum hallucination

- Many clients' top concern
- One of the best ways to minimize hallucinations (but does not eliminate hallucinations)