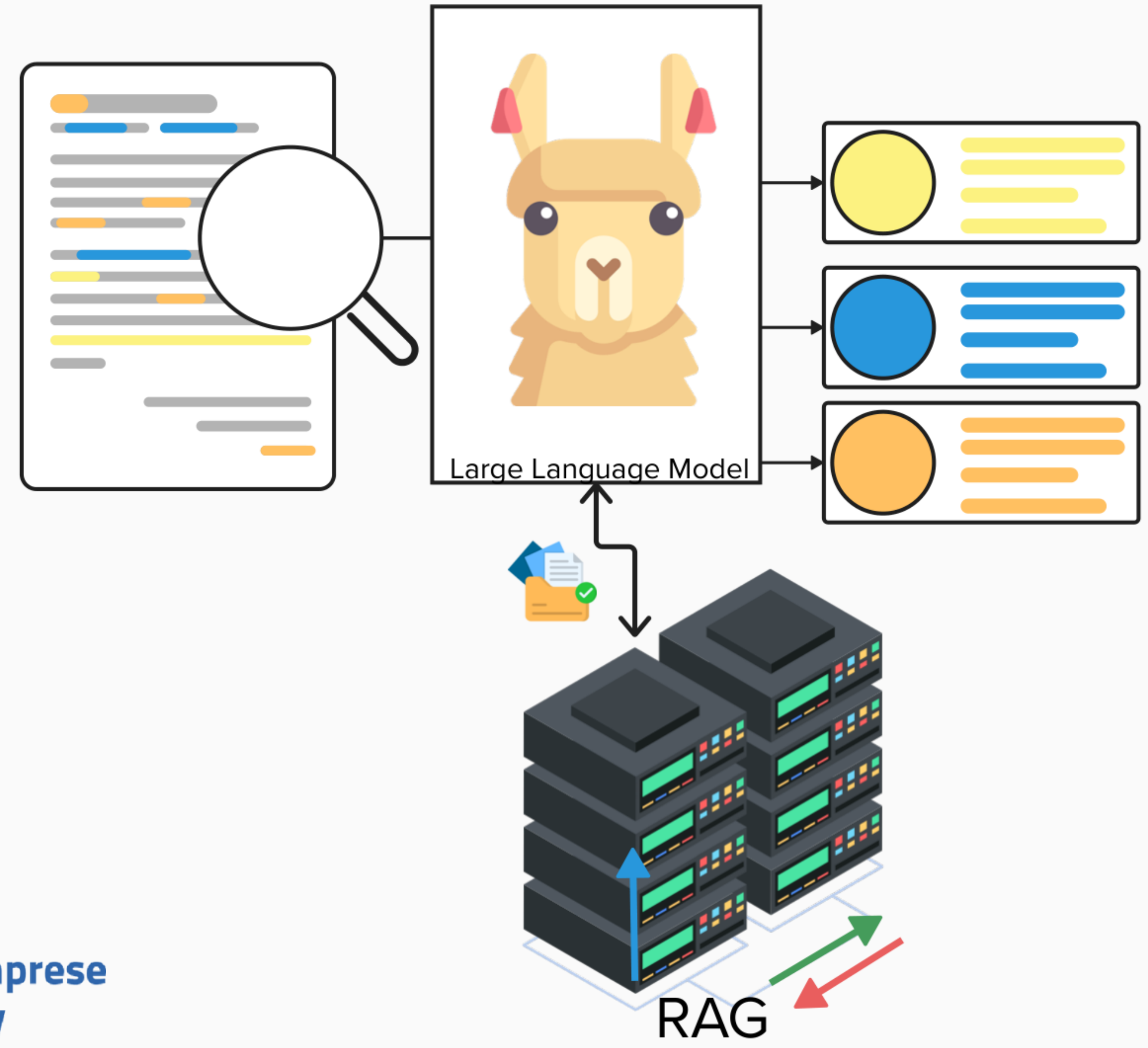
Retrieval Augmented Generation



DLAB CTE·CAGLIARI
COMUNE DI CAGLIARI
FSC Fondo per lo Sviluppo e la Coesione
Ministero delle Imprese e del Made in Italy

Lesson Overview

Key Topics Covered:

1. **Introduction**
    - LLM Limitations
    - What is RAG
    - Why RAG is needed
2. **Semantic Search**
    - Dense Retrieval
    - Reranking Methods
    - Document Embeddings
3. **RAG Implemetation**
    - Document Chunking Strategies
      Embedding Generation
    - Vector Storage
    - RAG Workflow
4. **Practical Workshop**

1. LLMS's Limitations

Outdated Information

Models only have information up to their training cutoff date. New data requires retraining, which is resource-intensive.

Domain Specificity

Generic responses may lack accuracy for specialized domains, requiring careful knowledge base curation.

Hallucinations

May generate plausible but incorrect information, particularly challenging when combining retrieved and generated content.

Source Attribution

Difficulty in tracking and citing original sources, raising concerns about information credibility and attribution.

Knowledge Gap Response

Instead of acknowledging uncertainty, may generate incorrect answers when information is missing from its knowledge base.

Long Training Time

Updating models with new information requires extensive computational resources and time, making rapid knowledge updates challenging.

1.1 What is RAG

Retrieval-Augmented Generation combines LLMs with external knowledge bases

Retrieval Augmented Generation is a branch of AI that combines information retrieval and text generation. This approach enables AI models to retrieve relevant knowledge from external sources and incorporate it into their generated responses, improving accuracy and contextual relevance.

Knowledge Base
External documents, data, and information stored in vector databases

Retrieval
Semantic search finds relevant information based on user queries

Generation
LLM generates responses using retrieved context and user input

Real-time
Access to up-to-date information without model retraining

1.2 Why RAG

LLMs are designed to generate responses, which can lead to fabricated answers when facing uncertainty. This phenomenon, known as hallucination, poses significant risks in applications where accuracy is crucial.

Enterprise Search

Access and query internal documents, policies, and procedures accurately

⚠ Critical business decisions based on outdated or incorrect information

Customer Support

Real-time access to product documentation and support history

⚠ Providing incorrect solutions to customer issues

Educational Systems

Personalized learning with accurate course materials and resources

⚠ Spreading misinformation to students

Legal Assistance

Access to case law and legal documentation

⚠ Incorrect legal interpretations or advice

Medical Knowledge

Access to latest research and medical protocols

⚠ Critical errors in medical information

Research Tools

Scientific literature analysis and citation

⚠ Fabricated research findings

RAG as a Solution

By grounding responses in verified external knowledge, RAG ensures accuracy and reliability in critical applications, significantly reducing the risk of hallucinations while maintaining the generative capabilities of LLMs.

1.3 Semantic Search Approaches

Dense Retrieval

Vector Space
Converts text into numerical embeddings

1

Similarity Search
Finds nearest neighbors in embedding space

2

Direct Results
Returns most similar documents

3

Reranking

Initial Search
Takes preliminary search results

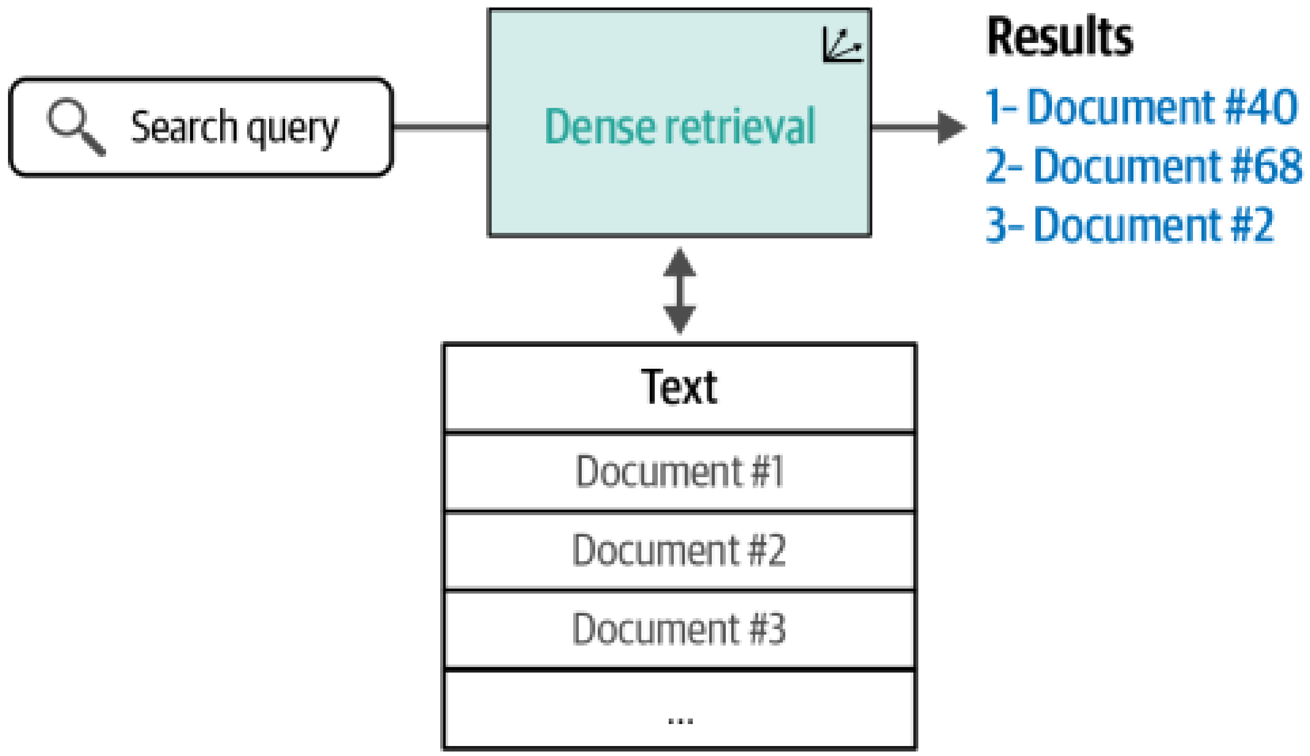
1

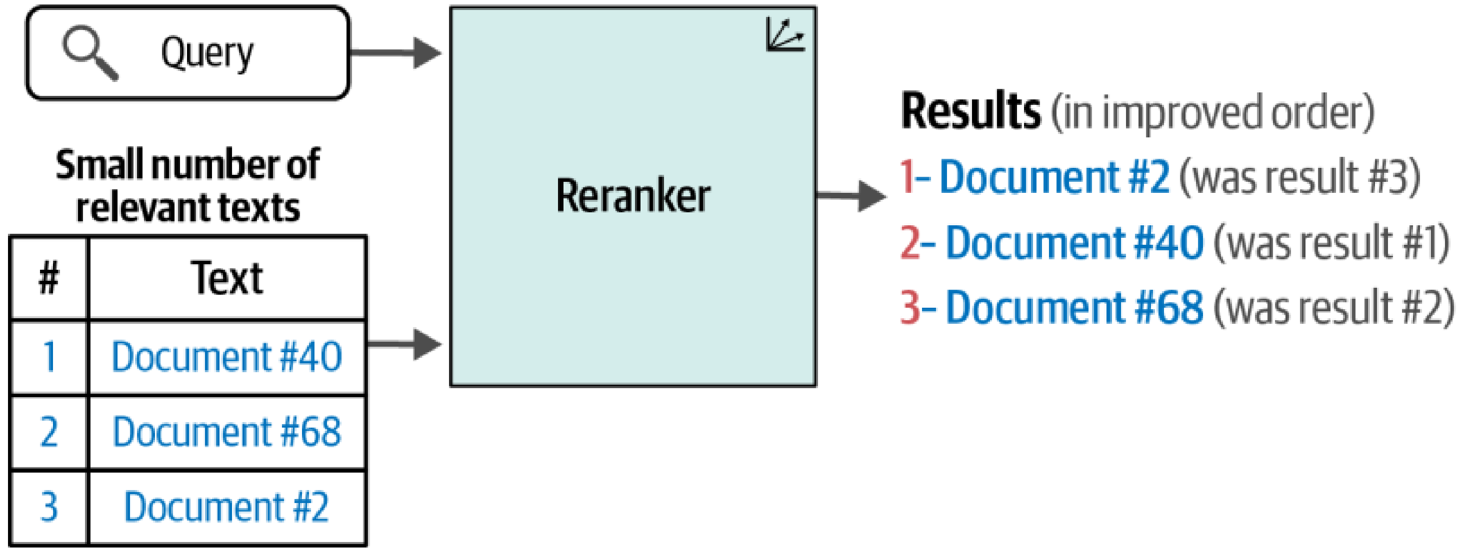
Relevance Scoring
Evaluates result quality against query

2

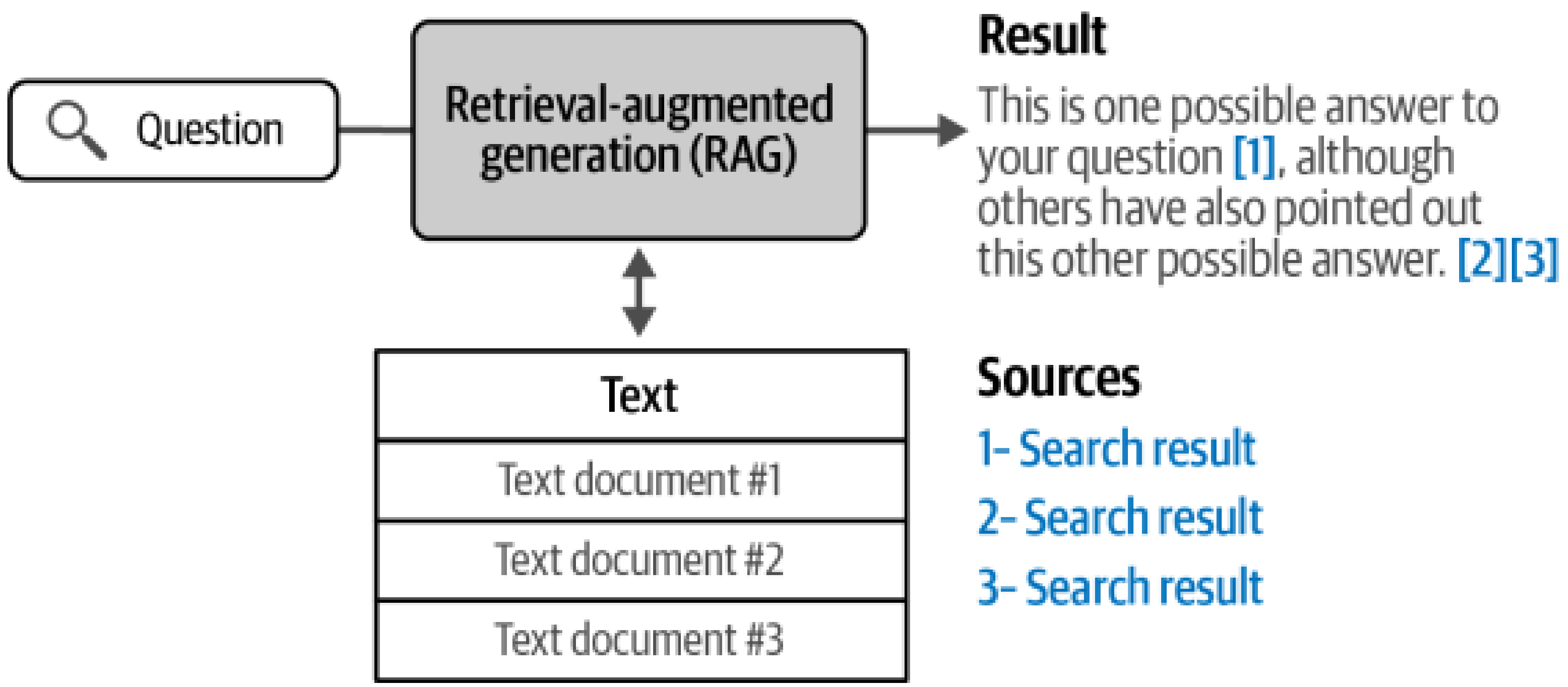
Result Reordering
Optimizes final result order

3



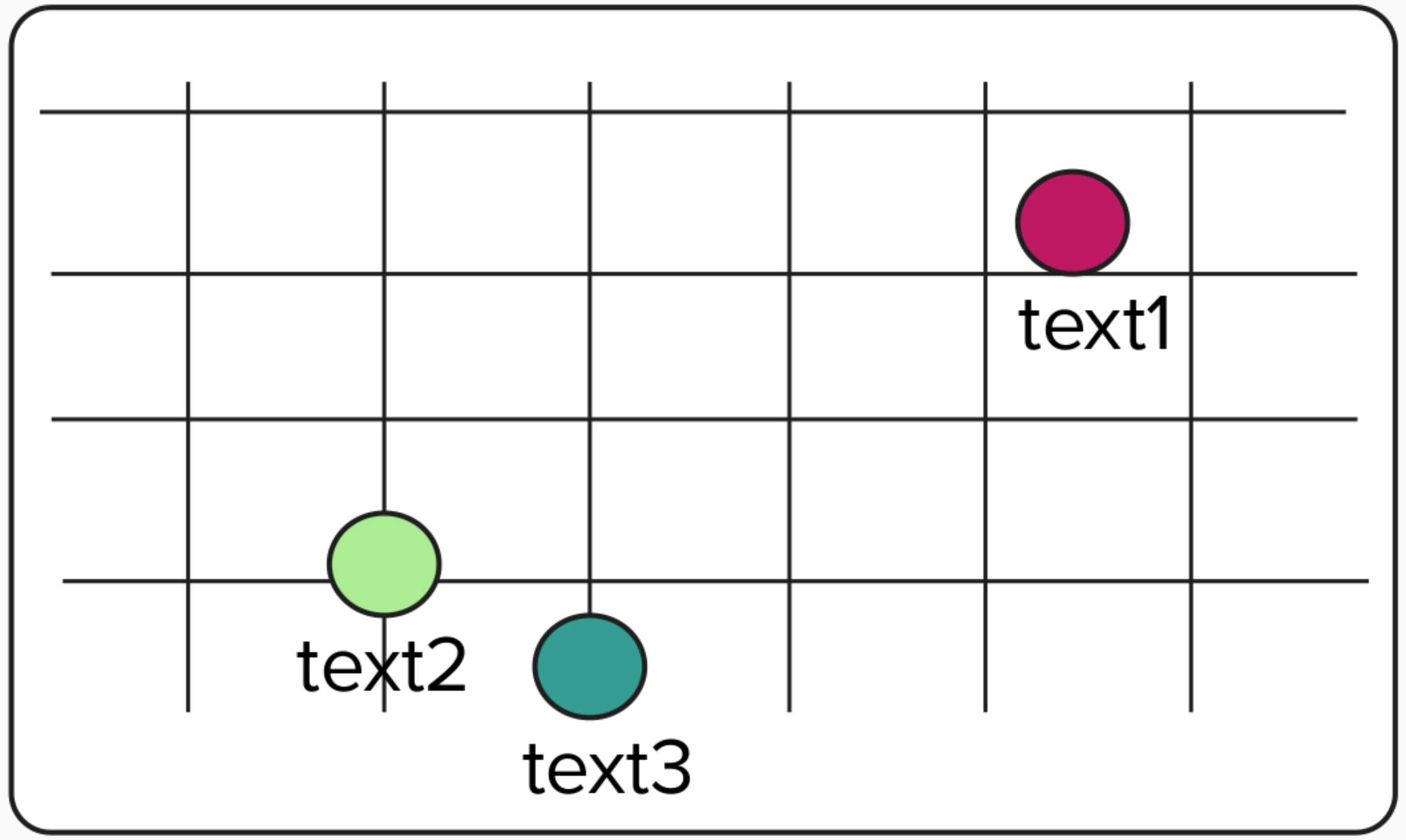


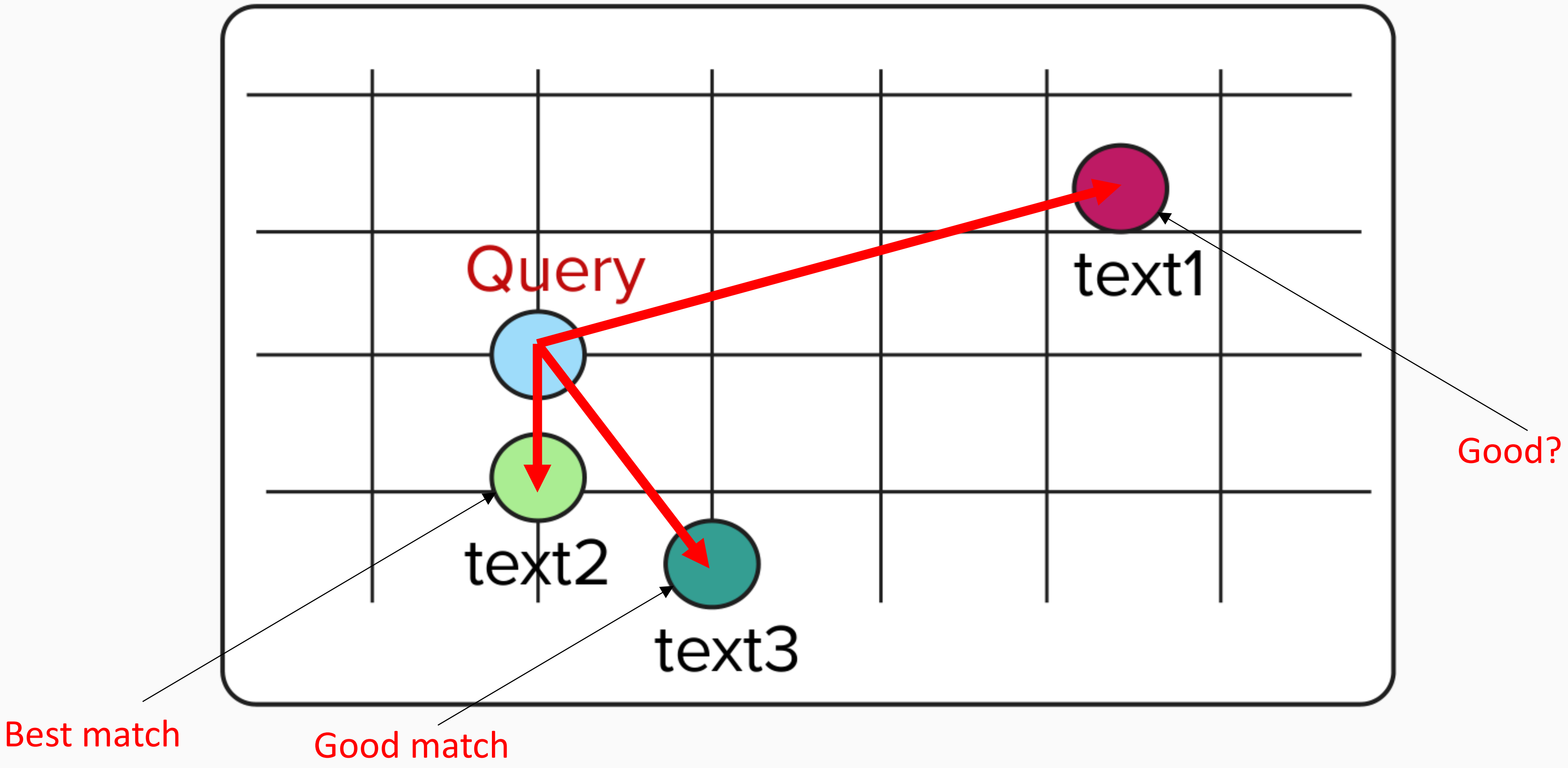
From Hands-On Large Language Models: Language Understanding and Generation.



From Hands-On Large Language Models: Language Understanding and Generation.

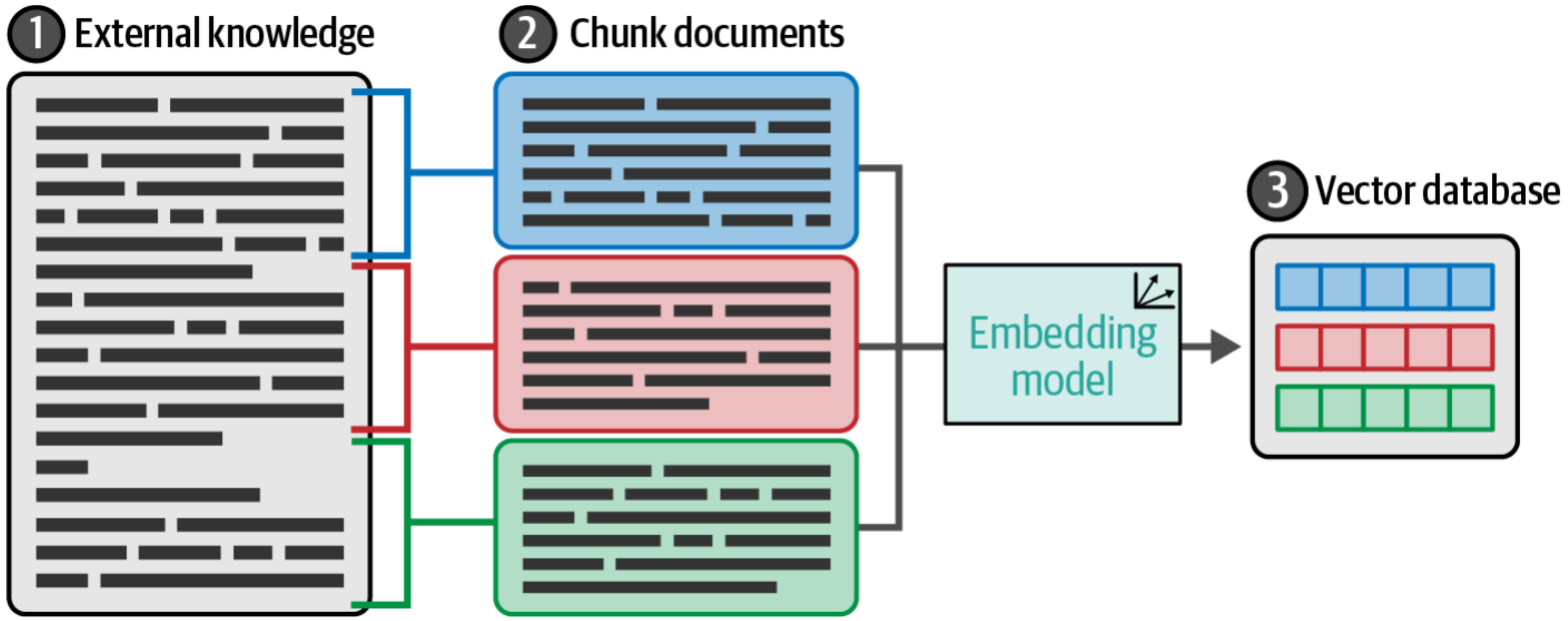
1.5 Dense Retrieval and Embeddings





1.6 How can we transform a document to embeddings

We know how to embed a word or a sentence



From Hands-On Large Language Models: Language Understanding and Generation.

External Knowledge

Long documents containing knowledge in various formats (manuals, articles, documentation)

Raw source material that needs to be processed for efficient retrieval

Embedding Generation

Converting text chunks into numerical vectors

Using embedding models to create dense vector representations of text

Document Chunking

Breaking down large documents into smaller, manageable pieces

Chunks maintain semantic coherence while being small enough for effective embedding

Vector Storage

Storing embeddings in specialized vector databases

Enables efficient similarity search and retrieval

1.7 Chunking Strategies

Document Level

One vector per entire document

Key Points:

- Preserves full context
- Less storage efficient
- May miss specific details
- Better for short documents

Fixed-Size Chunks

Document split into multiple chunks

Key Points:

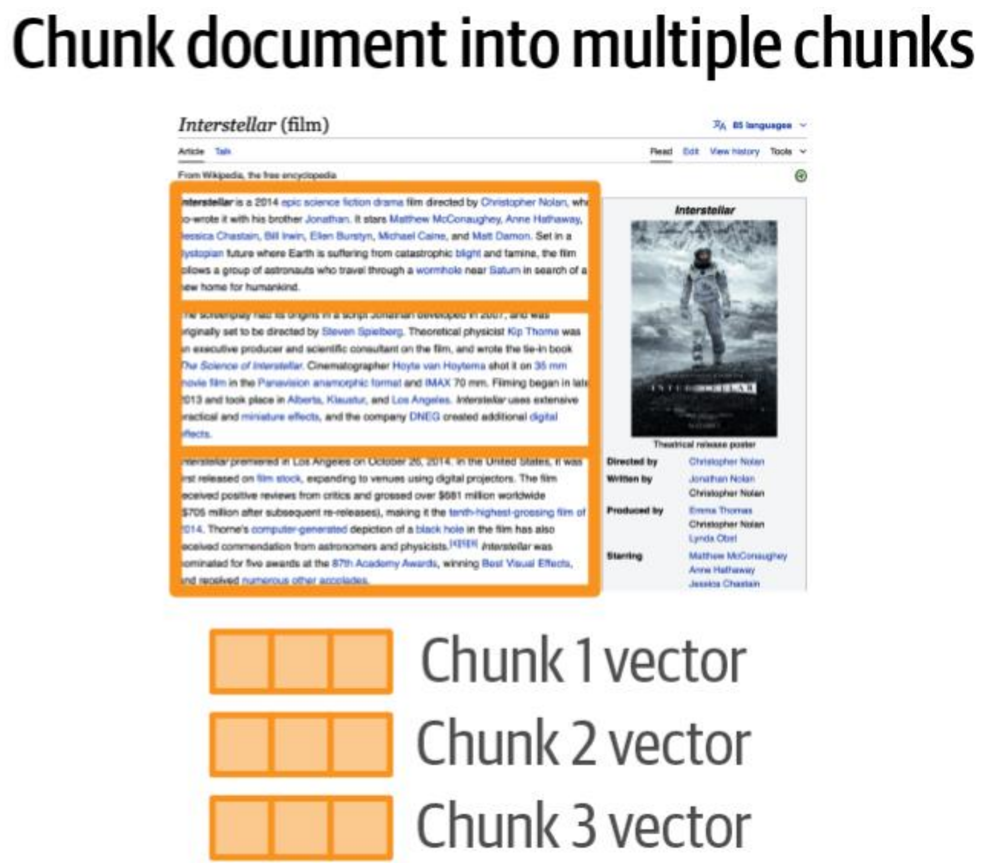
- Better for long documents
- More granular retrieval
- Multiple vectors per document
- Needs careful size selection

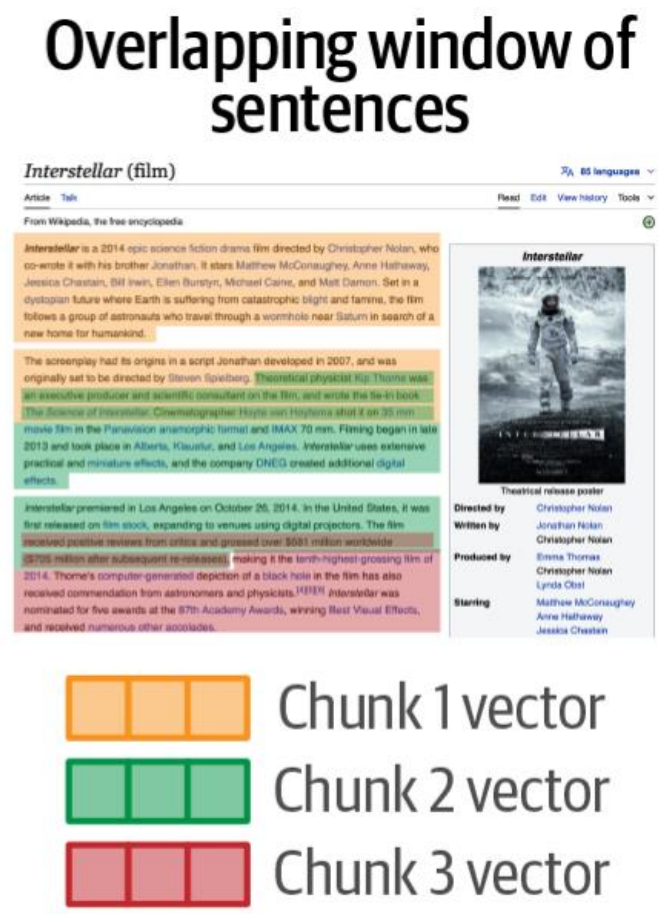
Token-Based Splitting

Split based on token count

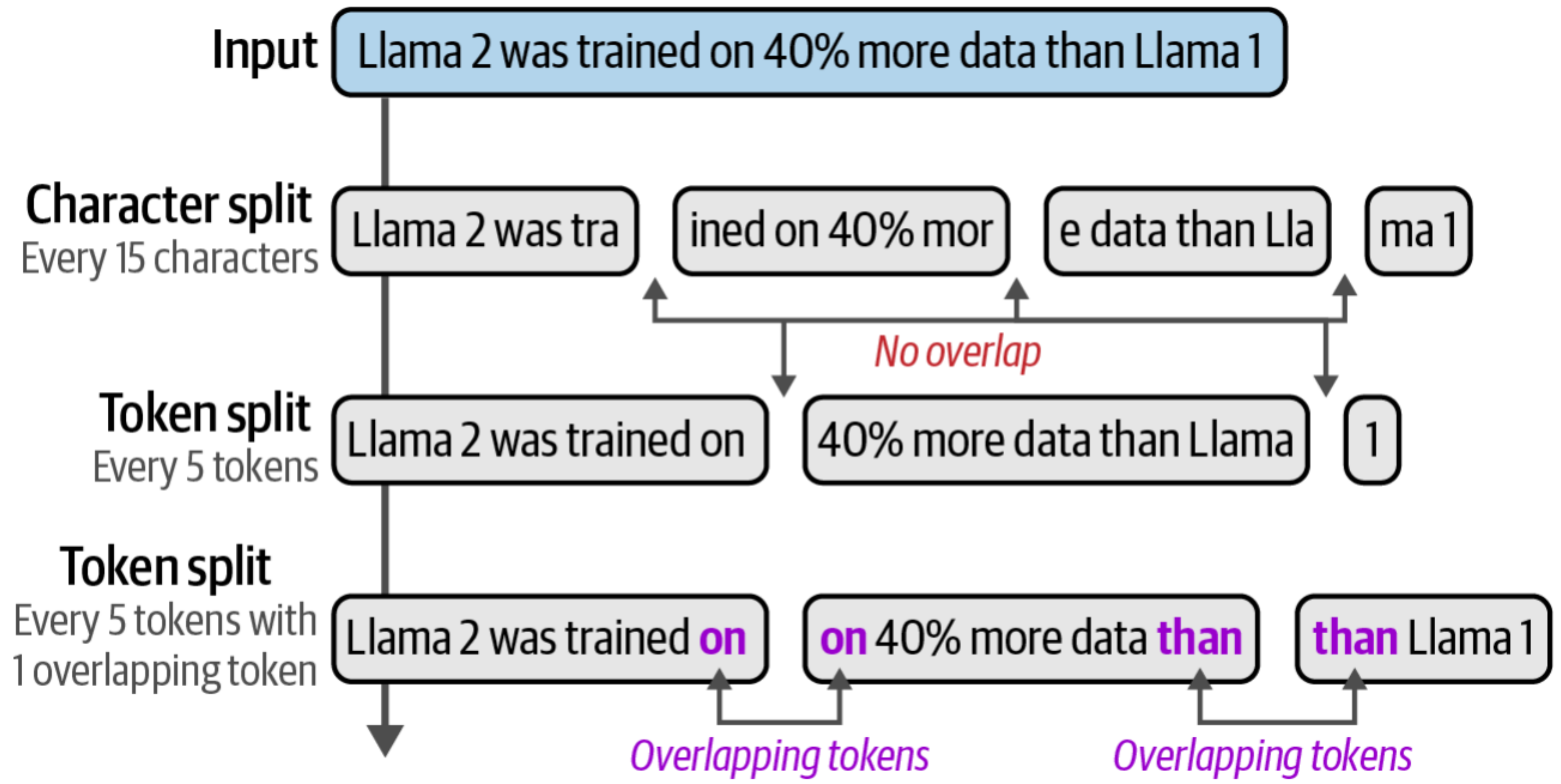
Key Points:

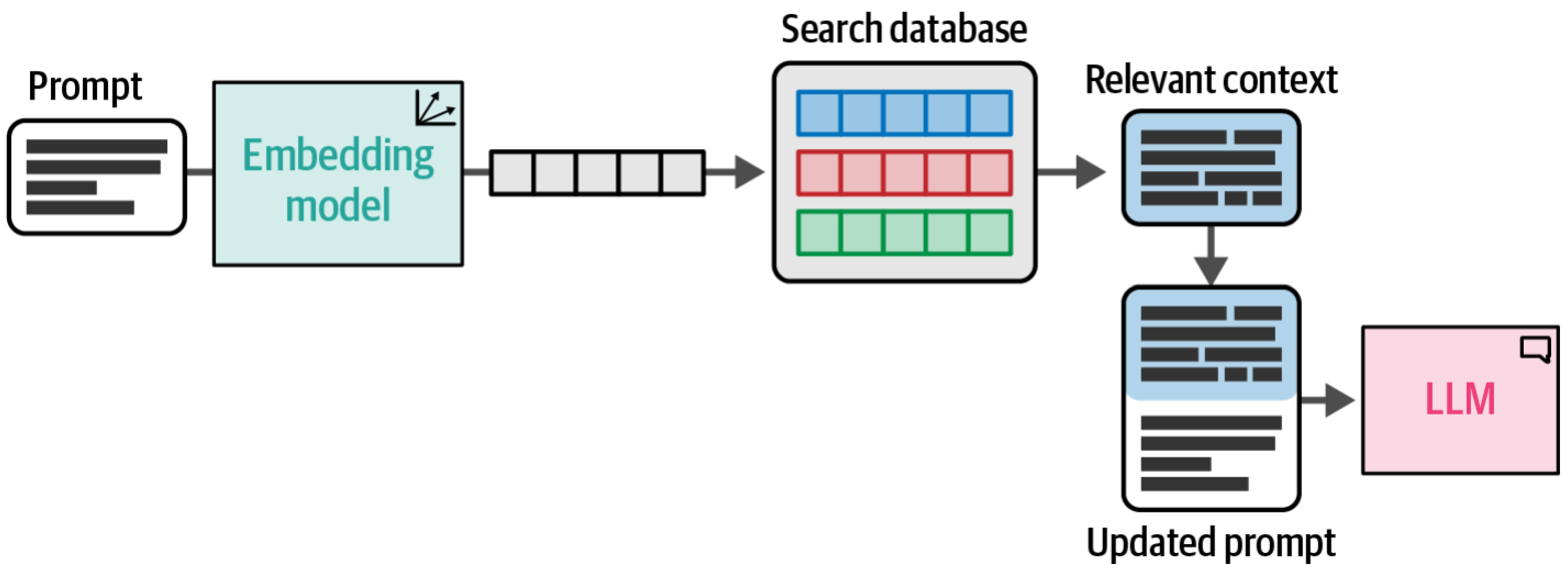
- Precise control over context window
- Can handle overlapping
- Maintains semantic units
- Common in modern LLMs





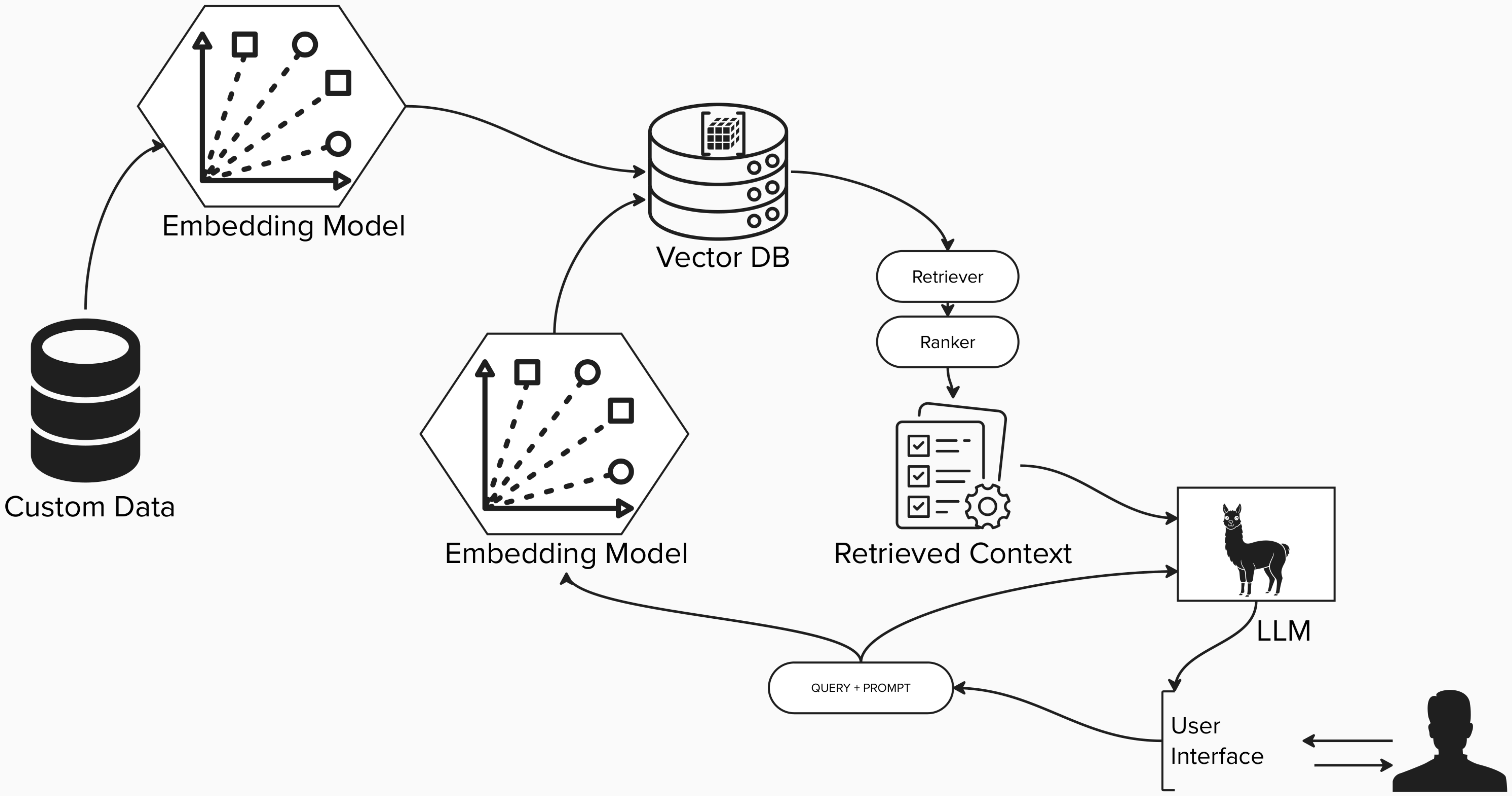
1.8 How RAG Works





From *Hands-On Large Language Models: Language Understanding and Generation*.

1.9 Summary


Embedding Model
Vector DB
Retriever
Ranker
Custom Data
Embedding Model
Retrieved Context
LLM
QUERY + PROMPT
User
Interface