Using a Local Language Model together with Retrieval-Augmented Generation (RAG) for answering questions on custom data

By Alexander Pintsuk

Table of contents

1. Problem statement

- Language Models do not have all knowledge
- Letting LM's learn new facts is computationally expensive and they even hallucinate
- Offloading to the cloud raises Privacy concerns
- LM's struggle with more complex tasks

Solution:

- Fully locally Agent, with the ability to retrieve information and reason from it

2. Introduction

Introduction – Prompt techniques

- Zero-shot prompting

Prompt:
Classify the text into neutral, negative or positive. Text: I think the vacation is okay.
Sentiment:

Response:
Neutral

Introduction – Prompt techniques

- Zero-shot prompting
- Few-shot

Prompt:
This is awesome! // Negative
This is bad! // Positive
What a horrible show! //

Response:
Negative

Introduction – Prompt techniques

- Zero-shot prompting
- Few-shot
- CoT - Chain-of-Thought [1]

Prompt:
John has 10 apples. He gives away 4 and then receives 5 more. How many apples does he have?
Reasoning:
John starts with 10 apples.
He gives away 4, so 10 - 4 = 6.
He then receives 5 more apples, so 6 + 5 = 11. Final Answer: 11

John has 9 apples. He gives away 4 and then receives 5 more. How many apples does he have?
Response:
….

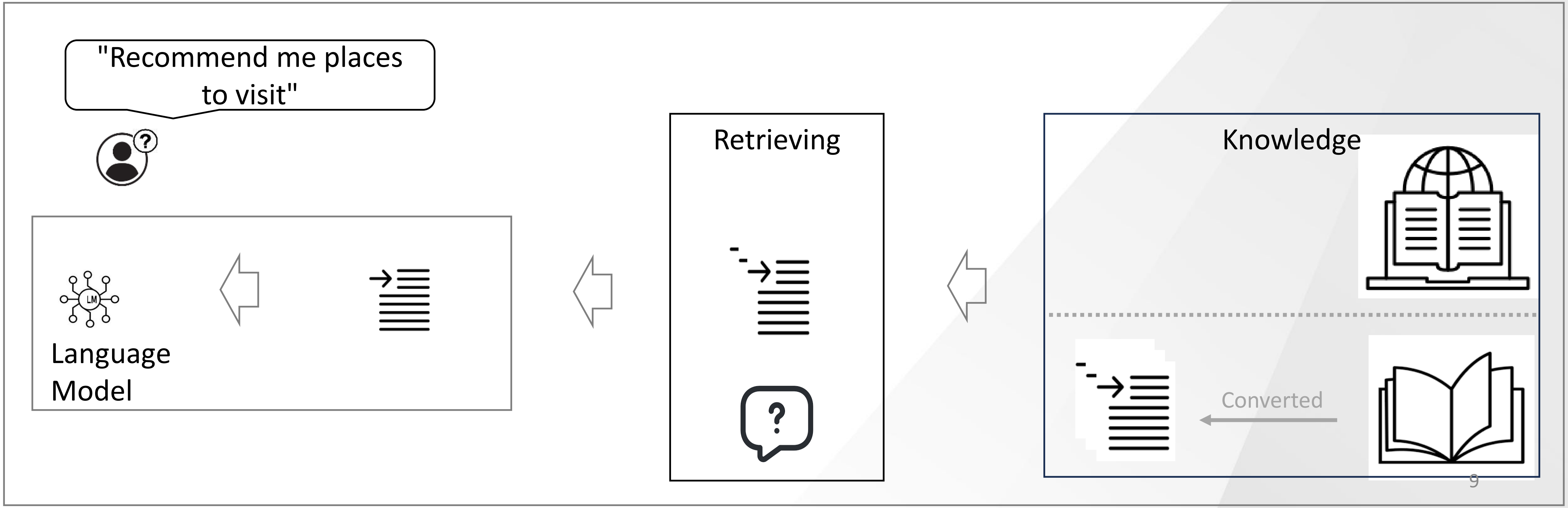
Introduction – Prompt techniques

- Zero-shot prompting
- Few-shot
- CoT - Chain-of-Thought [1]
- ReAct – Reasoning and Act [2]

Prompt:
Answer the following questions as best you can. You have access to the following tools:
{tools}
Use the following format:
Question: the input question you must answer
Thought: you should always think about what to do
Action: the action to take, should be one of [{tool_names}]
Action Input: the input to the action
Observation: the result of the action
... (this Thought/Action/Action Input/Observation can repeat N times)
Thought: I now know the final answer
Final Answer: the final answer to the original input question

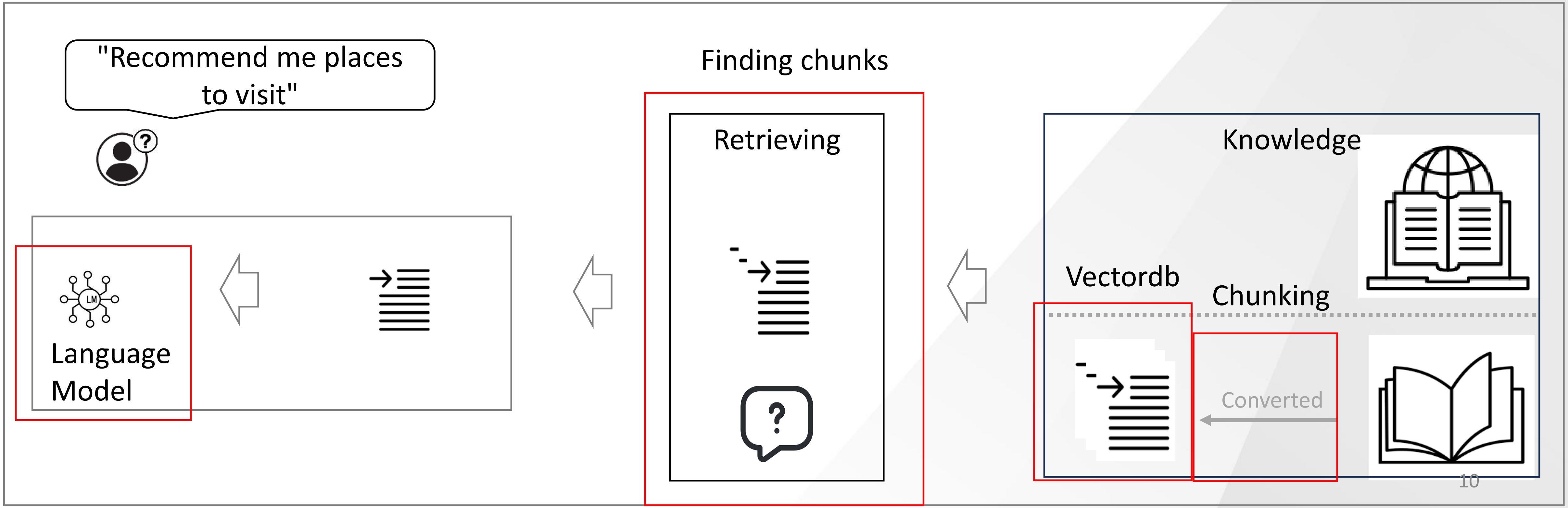
Introduction – Retrieval Augmented Generation - RAG

- Process of combining information retrieval with language models [6]
- Information retrieval includes local docs and the web

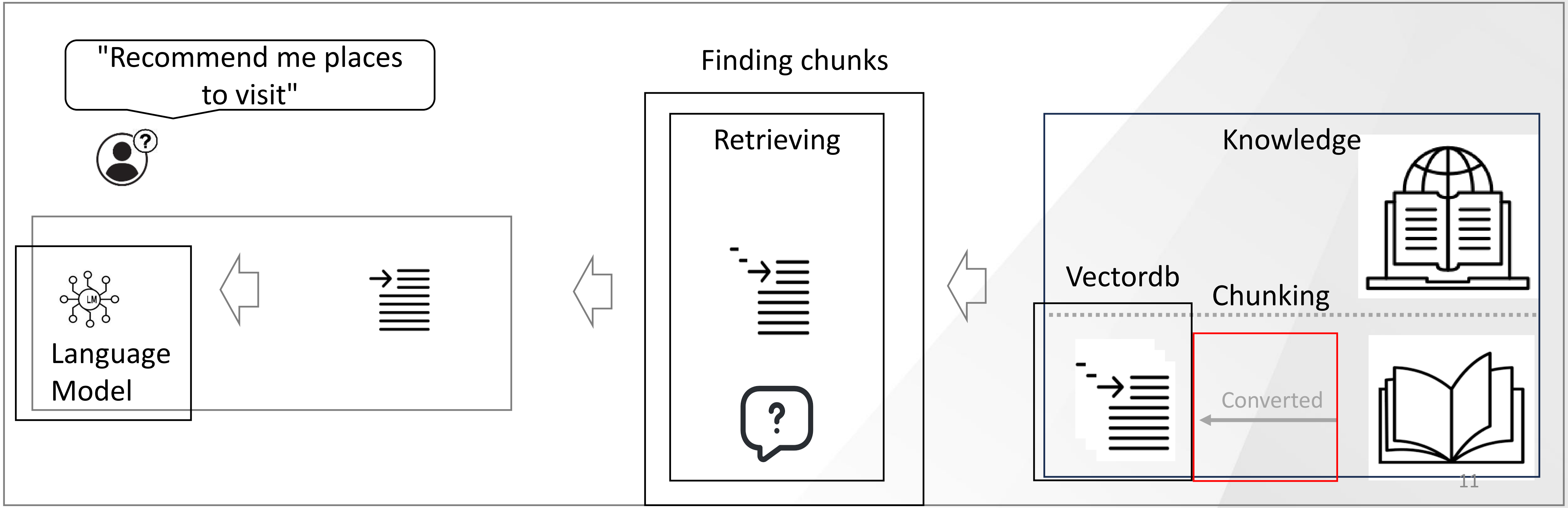


Introduction – Retrieval Augmented Generation - RAG

- Process of combining information retrieval with language models [6]
- Information retrieval includes local docs and the web

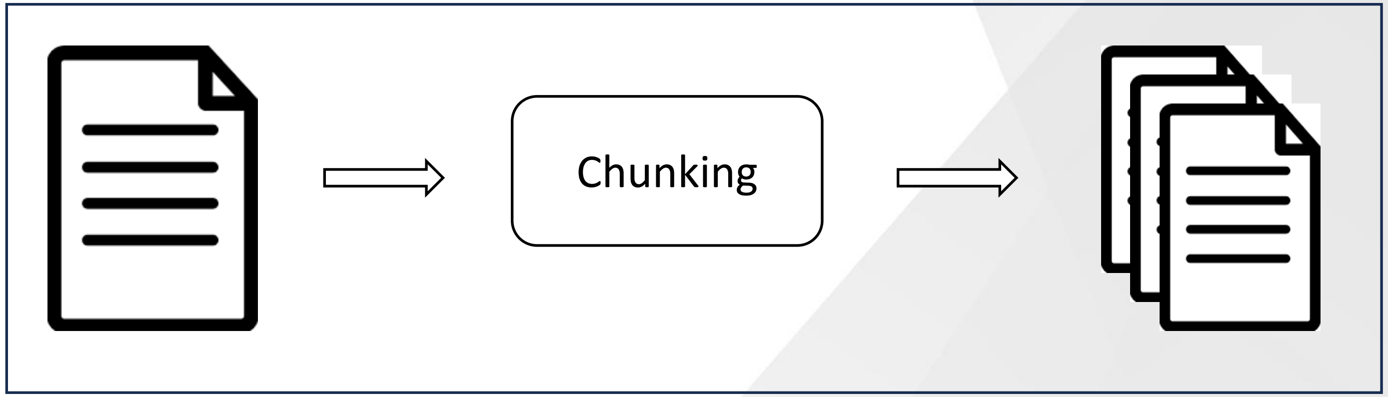


Chunking


"Recommend me places to visit"
Language Model
Finding chunks
Retrieving
Knowledge
Vectordb
Chunking
Converted

Chunking

- Fixed-Size Character Splitting [10]
- Recursive Chunking [10]
- Semantic Chunking [10]
- Overlap? Context?



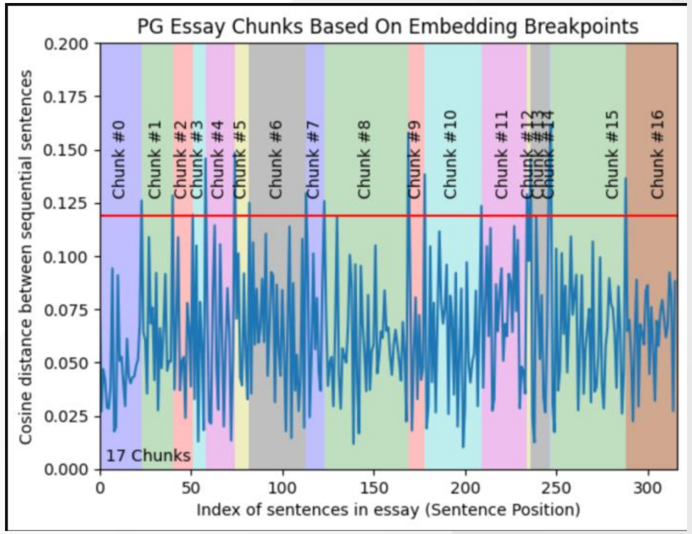
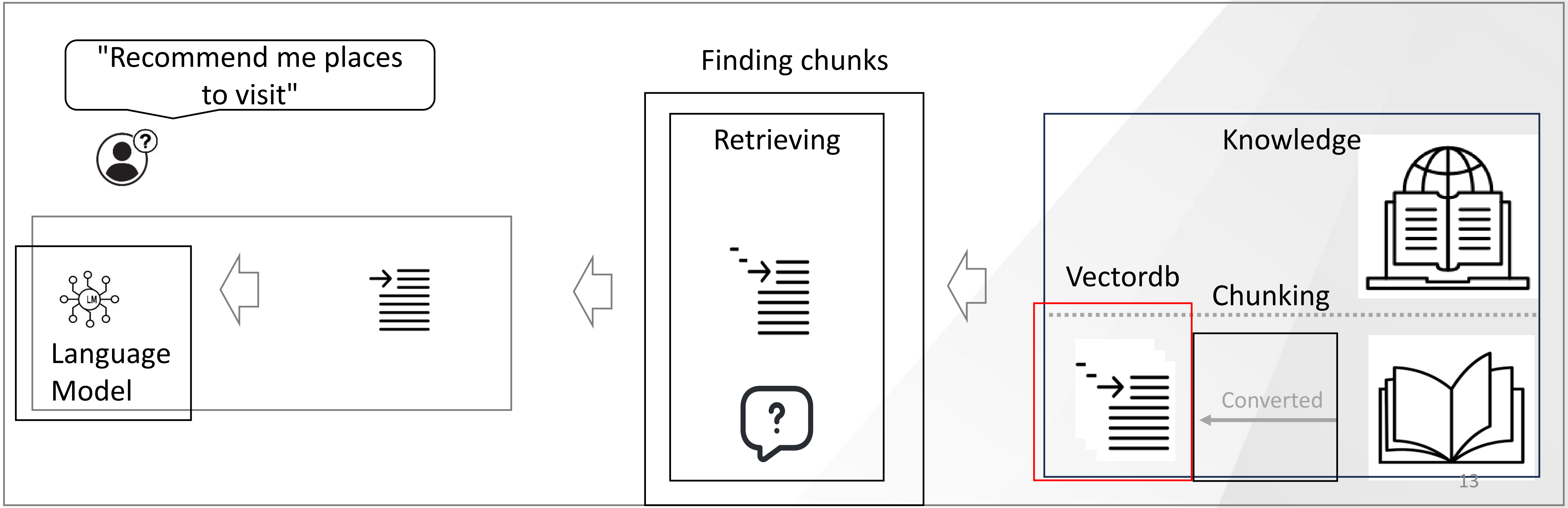


Figure 1. Semantic Chunking

Embeddings [⋮]



"Recommend me places to visit"
Finding chunks
Retrieving
Knowledge
Vectordb
Chunking
Converted
LM
Language Model

Embeddings

- Represent data as a vector
- Word Embeddings (Word2Vec) [7]
- Sentence Embeddings [8]
- Document Embeddings [9]

$$\begin{bmatrix} 0.24 \\ 0.94 \\ 0.12 \end{bmatrix}$$

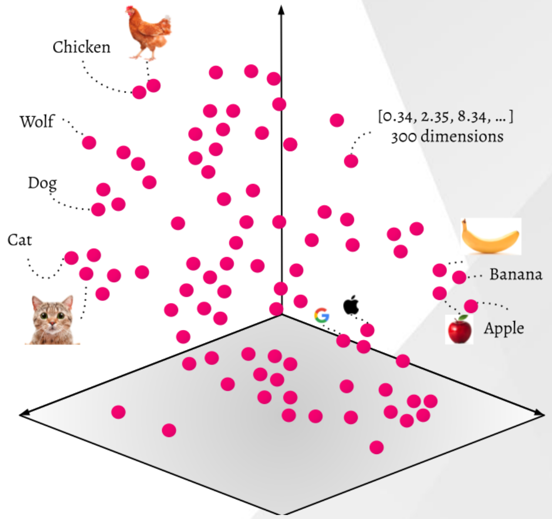


Figure 2. Embedding space

King - Man + Women = Queen

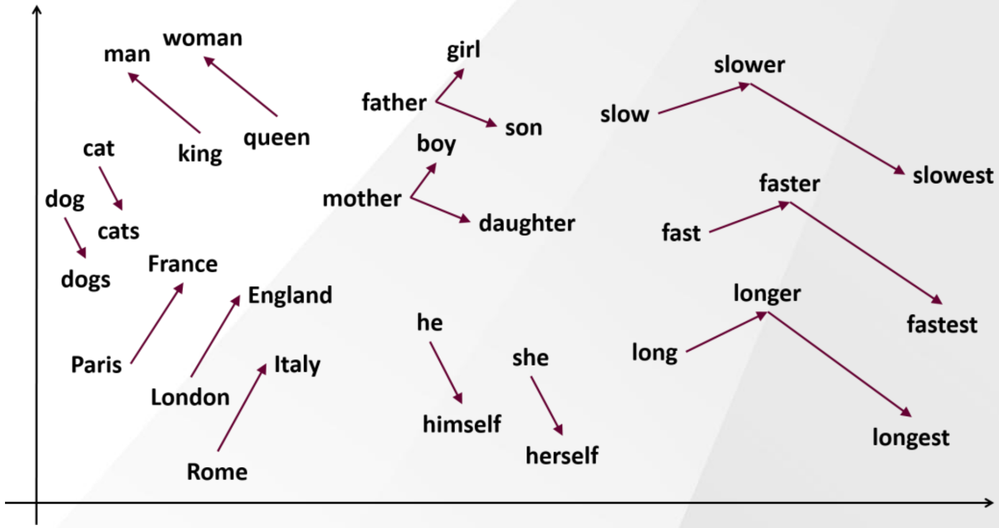


Figure 3. Vector arithmetic

Vector Databases

- Stores all embeddings
- Leverages the power of semantics
- Offers efficient search via indexing
- Best of both worlds: Hybrid Search

$$\begin{bmatrix} 0.24 \\ 0.94 \\ 0.12 \end{bmatrix}$$

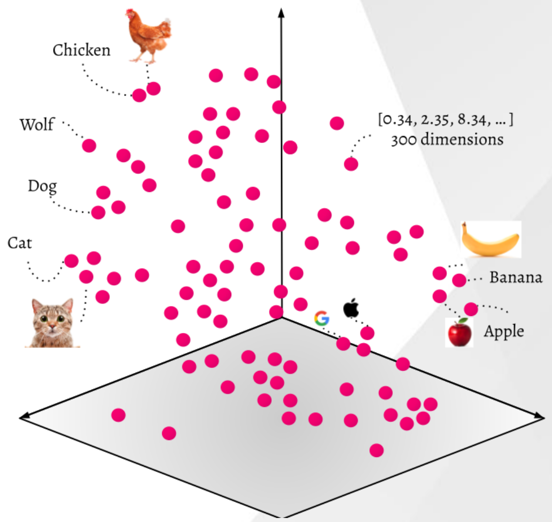
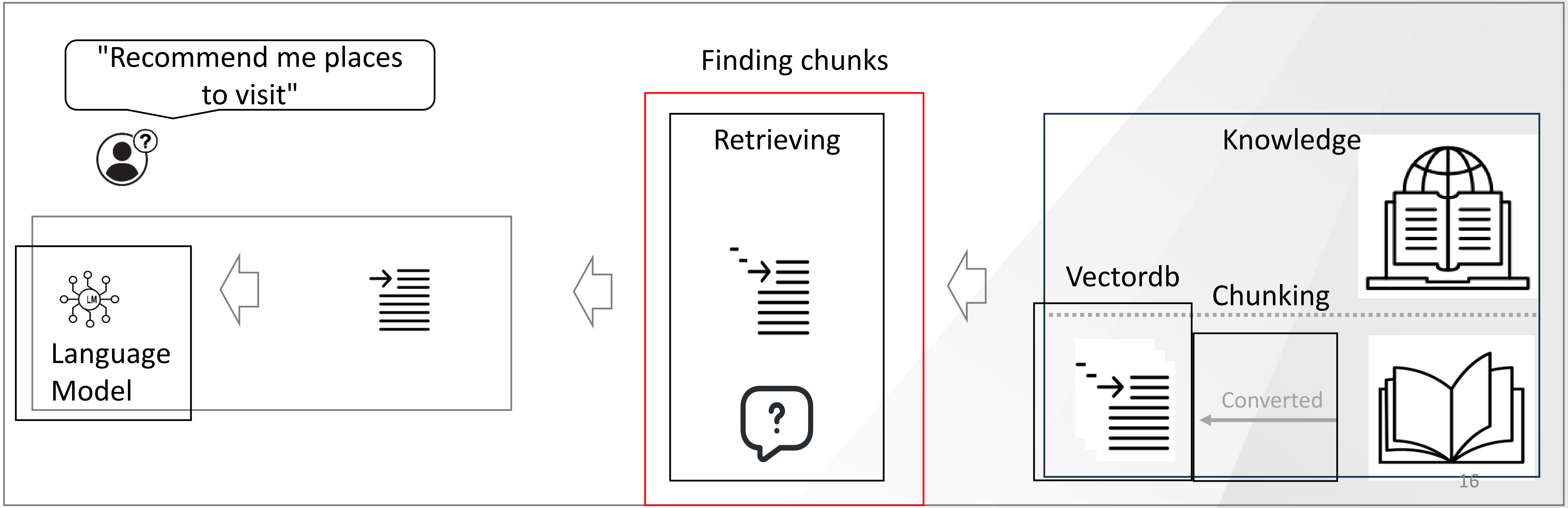


Figure 2. Embedding space

"Recommend me places to visit"

Search: "tourist attractions", as [0.94,0.43 , ..] and "places" keyword

Finding relevant chunks



Finding relevant chunks

- Keyword, vector search
- Rerank all results
- Give the best results as answer
- Bi-Encoder [11]
- Cross Encoder [12]
- Similarity -> Symmetric Encoder

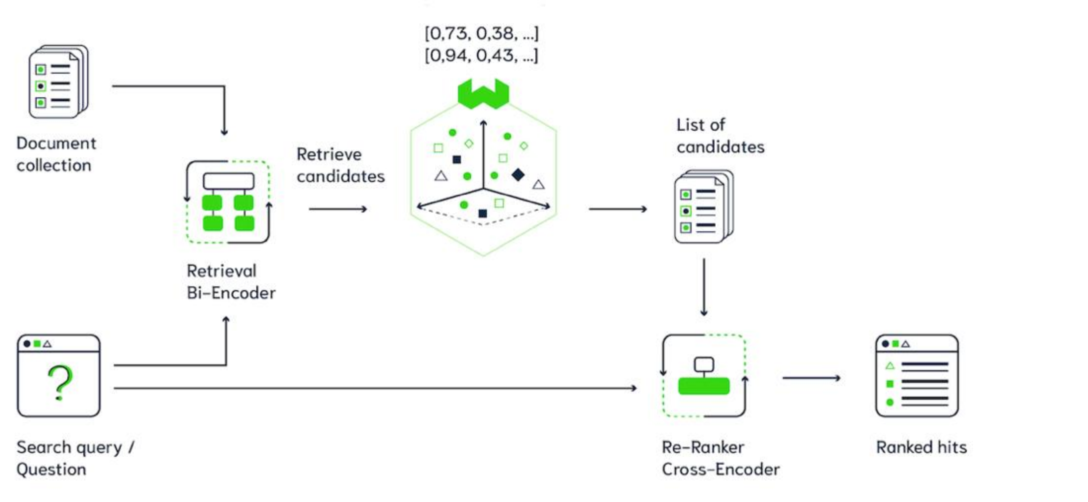


Figure 4. RAG pipeline

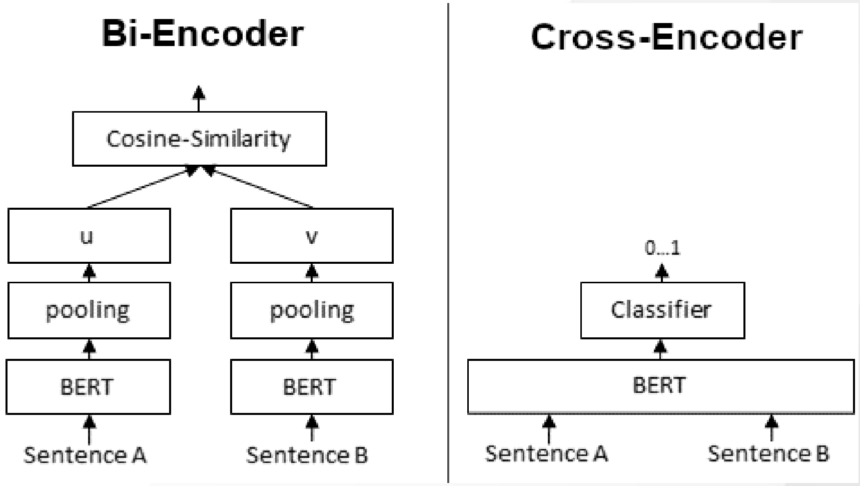
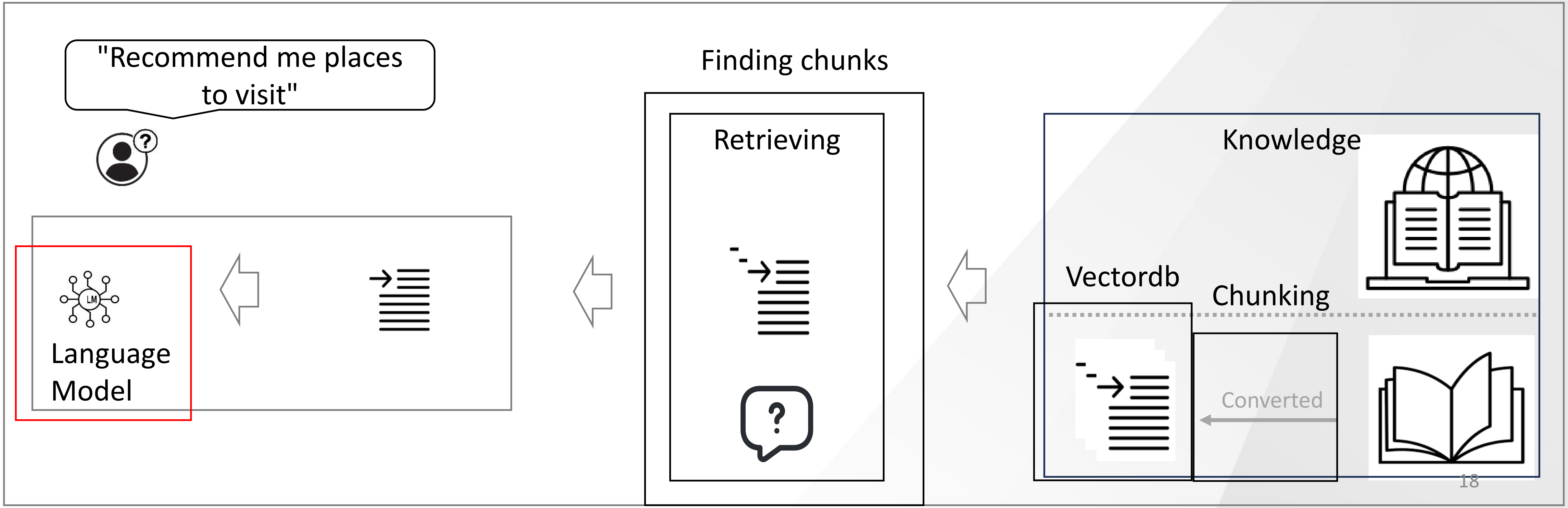


Figure 5. Encoders

Language Model



Language Model – Transformer Architecture

- Uses "Self-attention": Weighting the significance of each part the input [3]
- Quadratic complexity
- Transformer: abstractive summarisation
- Encoder: classification, Q&A, extractive summarisation [4]
- Decoder: translation, generation [5]

	Life	is	short	eat	desert	first
Life	0.17	0.13	0.18	0.16	0.15	0.18
is	0.03	0.68	0.02	0.08	0.14	0.02
short	0.19	0.06	0.25	0.14	0.11	0.23
eat	0.15	0.21	0.14	0.16	0.17	0.14
desert	0.13	0.27	0.11	0.16	0.18	0.12
first	0.19	0.02	0.31	0.11	0.07	0.27

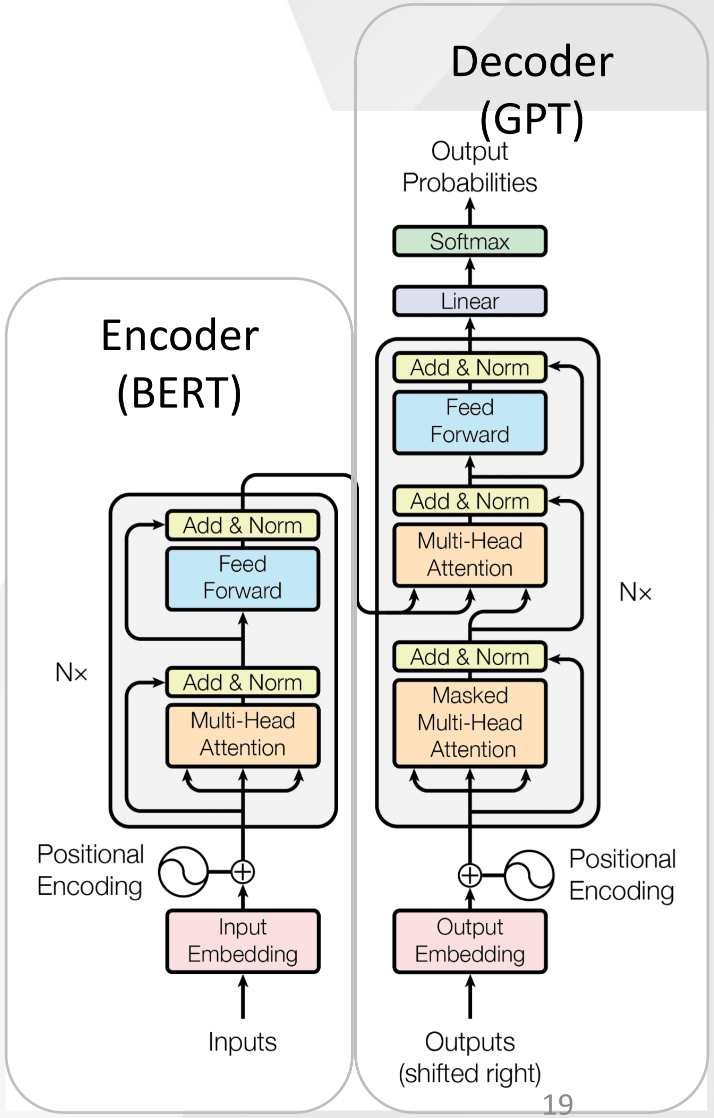


Figure 6. Transformer Architecture

3. Implementation

Implementation - Application Features

- Chat in a chat window (Foreign languages allowed)
- Upload Documents (english only)
- Conversation Memory
- "Debug view"
    - Agent state
    - Vector Database contents
- Settings

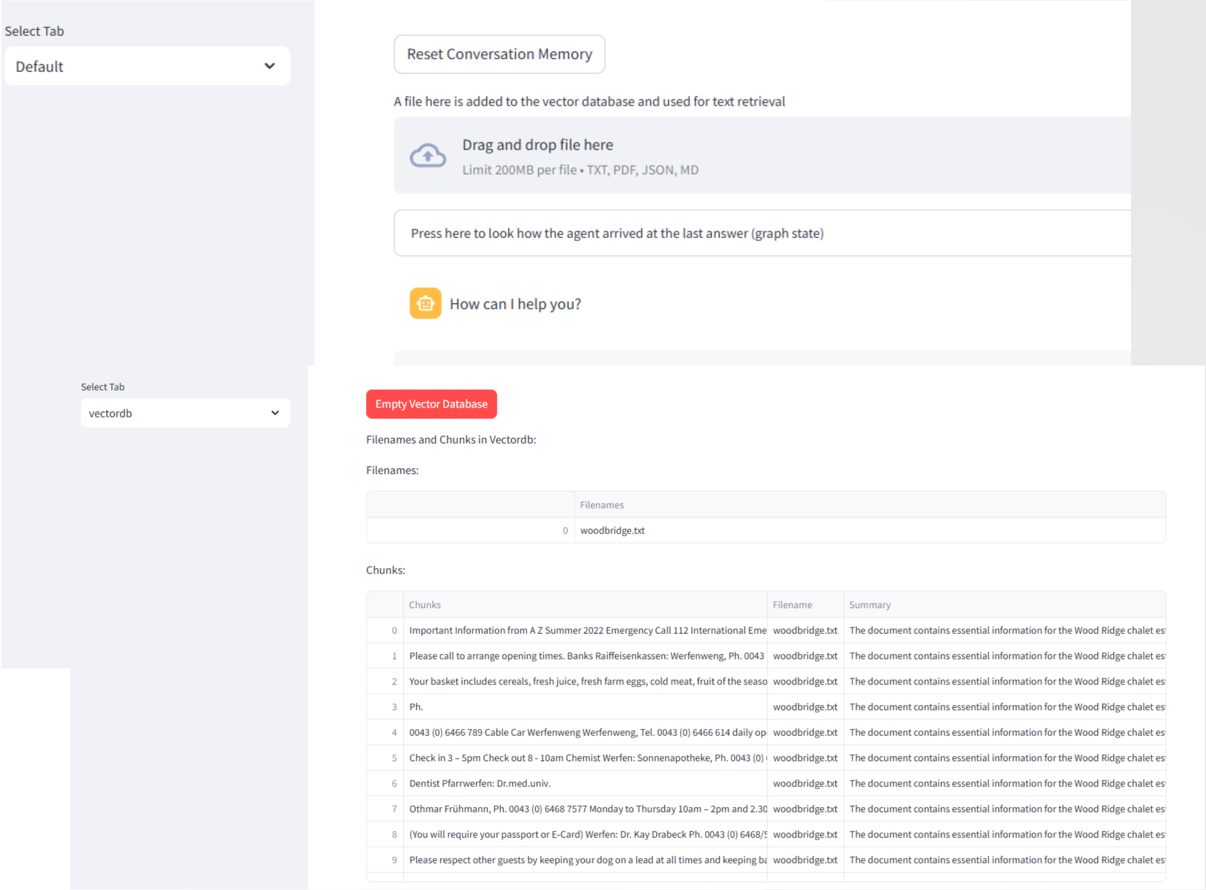
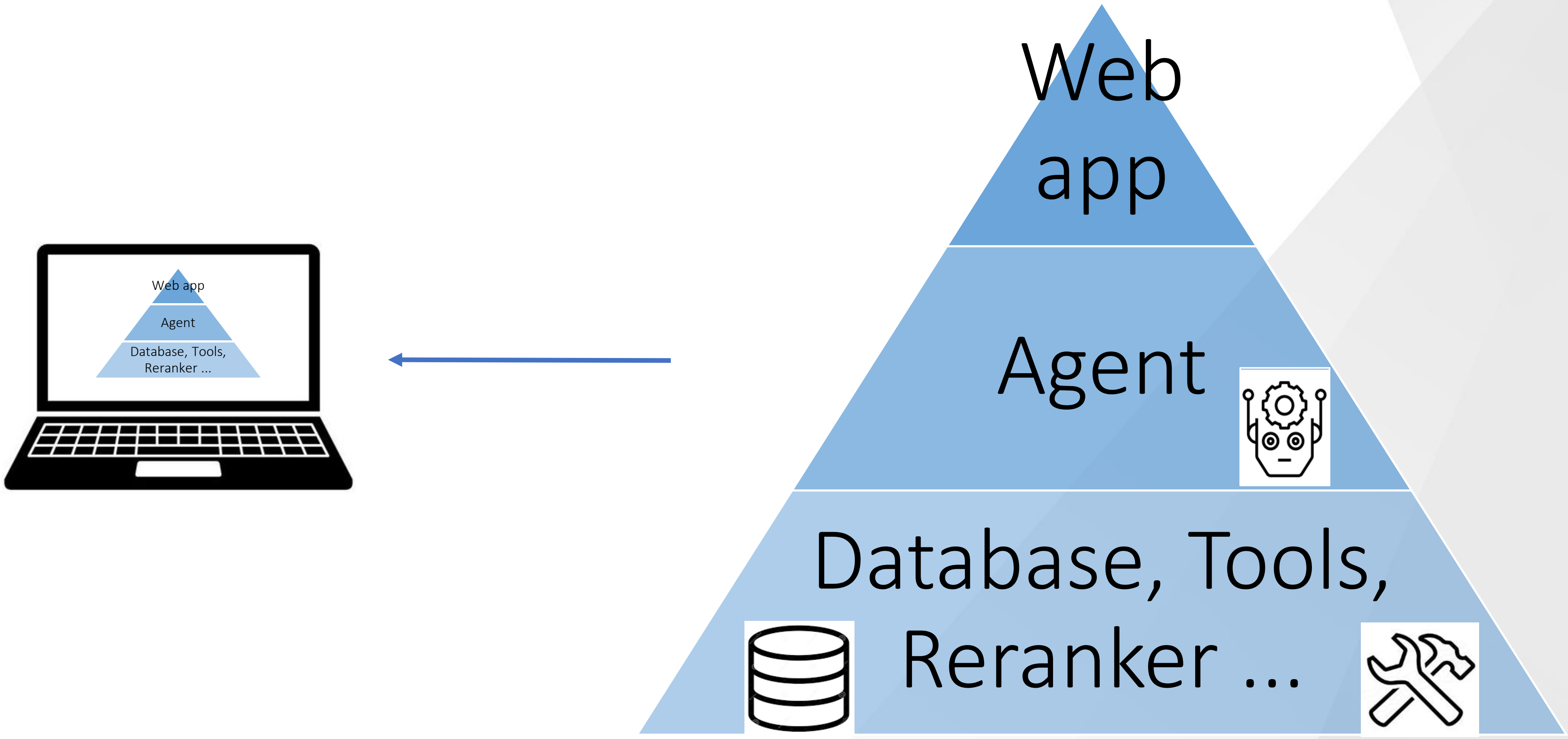


Figure 3. Application View

Implementation - Application components



Agent flow

Input: Conversation

1. Translate user message (optional)
2. Check for profanity (toggle)
3. Call tools (optional)
4. Translate back (optional)

Properties

- Uses "global" state to access/modify variables

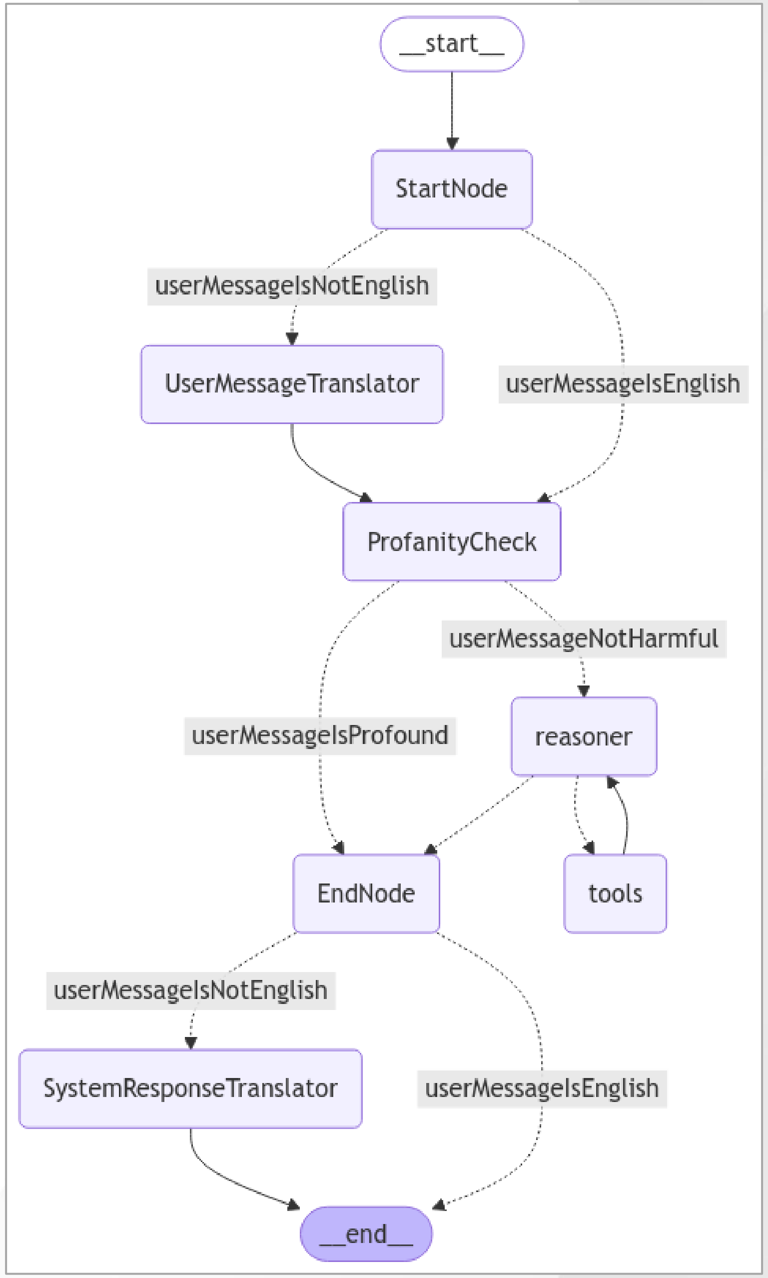
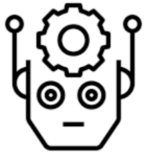


Figure 3. Agent graph

Agent flow 

Input: Conversation

1. Translate user message (optional)
2. Check for profanity (toggle)
3. Call tools (optional)
4. Translate back (optional)

Properties

- Uses "global" state to access/modify variables

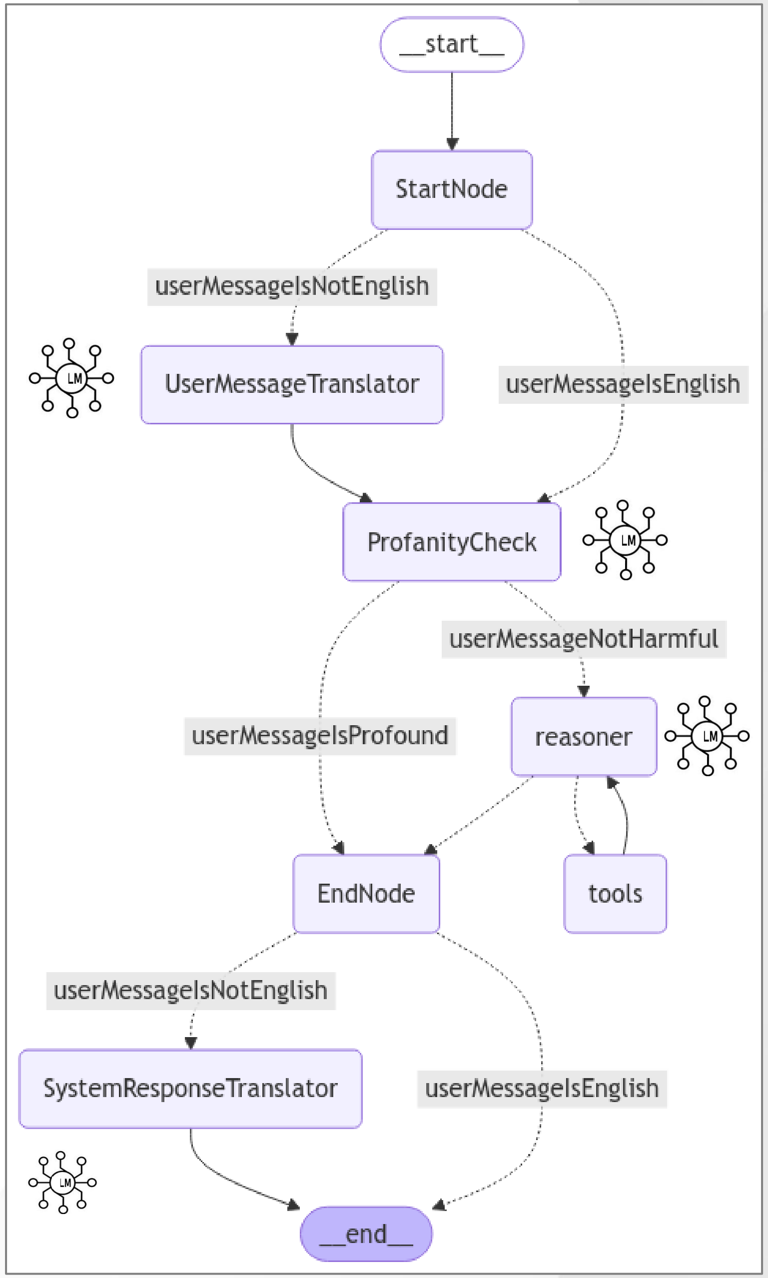
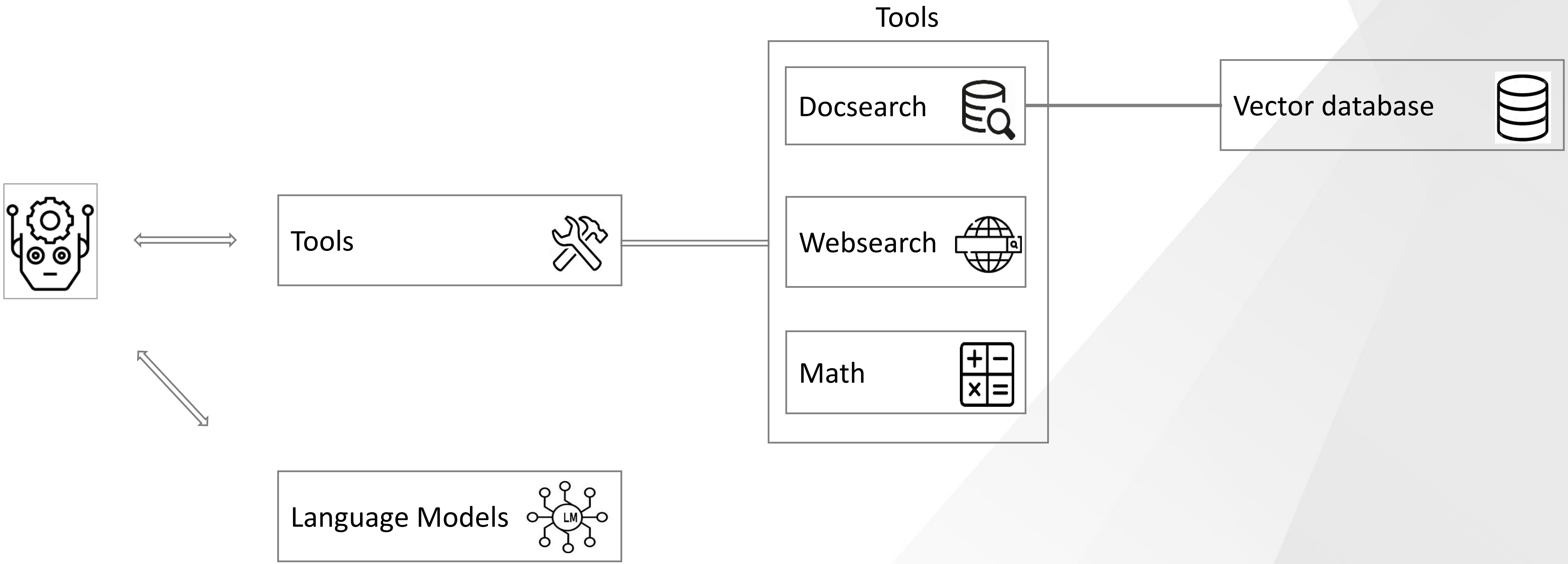


Figure 3. Agent graph

Agent components



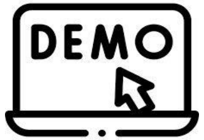
Specifications

- Reasoning Model: Llama3.1:8b
- Translation Model: Aya:8b
- Document Embedding Model: BAAI/bge-small-en-v1.5
- Semantic Chunker Model: BAAI/bge-small-en-v1.5
- Reranker Model: ms-marco-MiniLM-L-12-v2
- Language Detector: papluca/xlm-roberta-base-language-detection

Sysprompt: "You are a helpful assistant with access to tools. You can search for relevant information using the provided tools and perform arithmetic calculations.
For each question, determine if you can answer the question directly based on your general knowledge, or If necessary Use the `Search_in_document` tool to find the necessary information within the available documents.

If you do not get an answer from the 'Search_in_document' tool Message or get an error, use the websearch tool, but the websearch tool should have lower priority.

Sysprompt: You are a professional translator. You must only translate the given human message into English. Even if the user writes a question, you have to translate the question to english and you are NOT allowed to respond to the question. Provide only the translated text without any additional information, comments, or explanations.
Examples: Input: "Bonjour, comment ça va ?"
Output: "Hello, how are you?" Input: "¿Dónde está la biblioteca?

4. Demo 

https://localhost

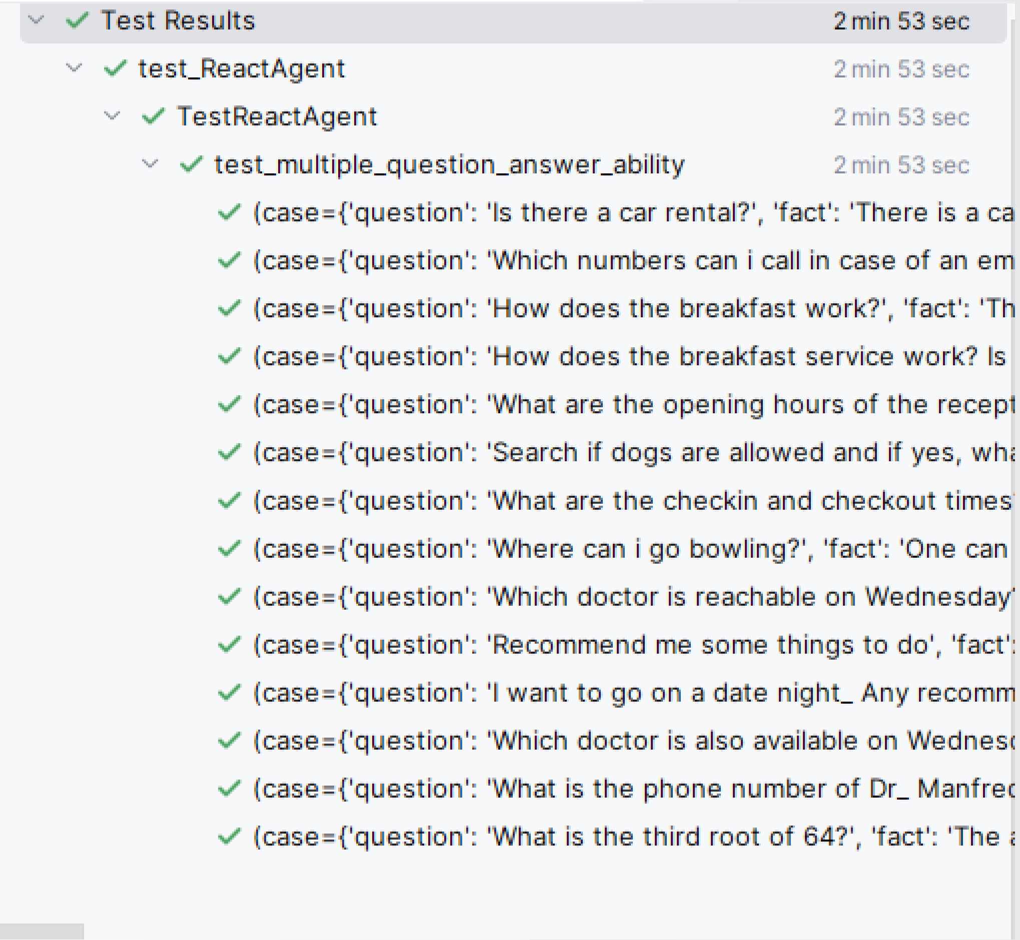
5. Evaluation

5. Evaluation

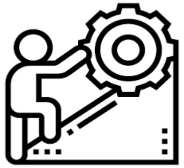
- 2 main evaluations
  - Simple Q&A
  - Non english Q&A

Method

- Let a language model decide if the agent's response is correct given a true fact
- Logfile with
  - Questions
  - Facts
  - Snippets
  - Provided Contexts



6. Summary

6. Summary 

- Local Agents still have room for improvement
- Local Language Models get more powerful over time
- Tradeoff: Performance and Quality
- Dependence on Libraries
- Intransparent abstractions
- Possible Strategies and Parameters

Literature

[1] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia,F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large anguage models, 2023.

[2] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. React: Synergizing reasoning and acting in language models, 2023.

[3] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2023.

[4] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[5] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training.

[6] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks. In Advances in Neural Information Processing Systems (2020), vol. 33, pp. 9459–9474.

[7] Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. From word embeddings to document distances. In Proceedings of the 32nd International Conference on Machine Learning (Lille, France, 07–09 Jul 2015), F. Bach and D. Blei, Eds., vol. 37 of Proceedings of Machine Learning Research, PMLR, pp. 957–966.

[8] Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y., Strope, B., and Kurzweil, R. Universal sentence encoder. CoRR abs/1803.11175 (2018).

[9] Sinoara, R. A., Camacho-Collados, J., Rossi, R. G., Navigli, R., and Rezende, S. O. Knowledge-enhanced document embeddings for text classification. Knowledge-Based Systems 163 (2019), 955–971.

[10] Kshirsagar, A. Enhancing rag performance through chunking and text splitting techniques. International Journal of Scientific Research in Computer Science, Engineering and Information Technology 10, 5 (2024), 151– 158.

[11] Reimers, N., and Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[12] Tran, T. Q., Kang, M., and Kim, D. Rerankmatch: Semi-supervised learning with semantics-oriented similarity representation, 2021.

Figures

- Figure 1: https://ai.gopubby.com/unleashing-the-power-of-semantic-chunking-a-journey-with-llamaindex-767e3499ca73
- Figure 2: https://odsc.com/blog/getting-started-with-vector-based-search/
- Figure 3: https://steemit.com/programming/@oddpotato/word2vec-introduction
- Figure 4: https://weaviate.io/blog/cross-encoders-as-reranker
- Figure 5: https://www.sbert.net/examples/applications/cross-encoder/README.html
- Figure 6: [3]

Thank you for your Attention!