Zürcher Hochschule
für Angewandte Wissenschaften

School of
Management and Law

EQUIS
since 2023

AACSB
since 2015

Retrieval Augmented Generation
RAG Improvements

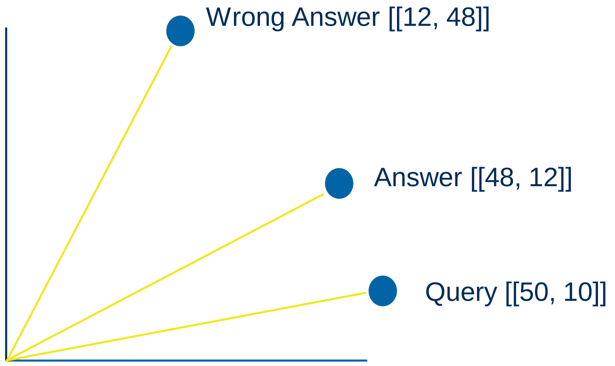
Building Competence. Crossing Borders.

Jasmin Heierli
jasmin.heierli@zhaw.ch, 20 March 2024

Cosine Similarity

Let‘s close the circle and go back to our original Problem:

We can now calculate the cosine similarity between two vectors 🥳



$$cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\Sigma_{i=1}^{n} A_i \cdot B_i}{\sqrt{\Sigma_{i=1}^{n} A_i^2}\cdot\sqrt{\Sigma_{i=1}^{n} B_i^2}}$$

$$\mathbf{A}\cdot\mathbf{B} = 50\times 48 + 10\times 12 = 2400 + 120 = 2520$$

$$\|\mathbf{B}\| = \sqrt{48^2+12^2} = \sqrt{2304+144} = \sqrt{2448} \approx 49.5$$

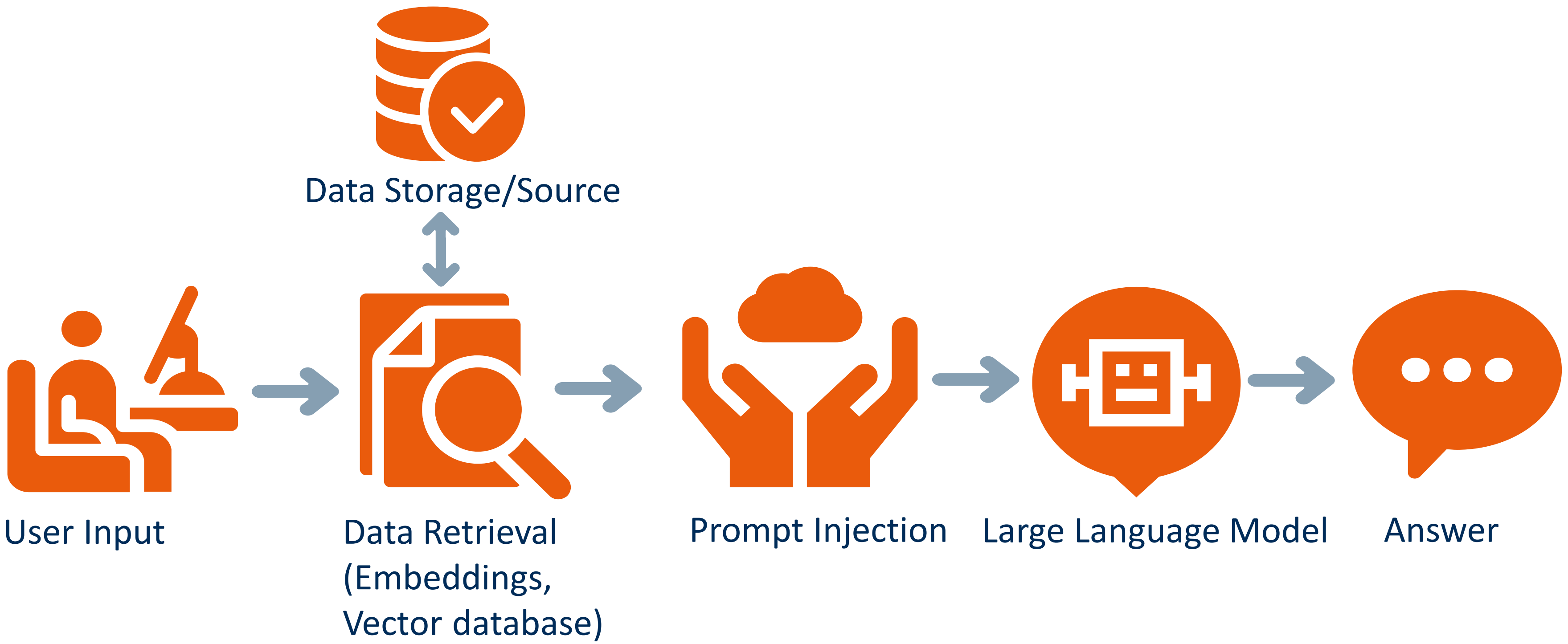
$$\|\mathbf{A}\| = \sqrt{50^2+10^2} = \sqrt{2500+100} = \sqrt{2600} \approx 51.0$$

$$cosine\ similarity = \frac{2520}{51.0\times 49.5} \approx 0.998$$

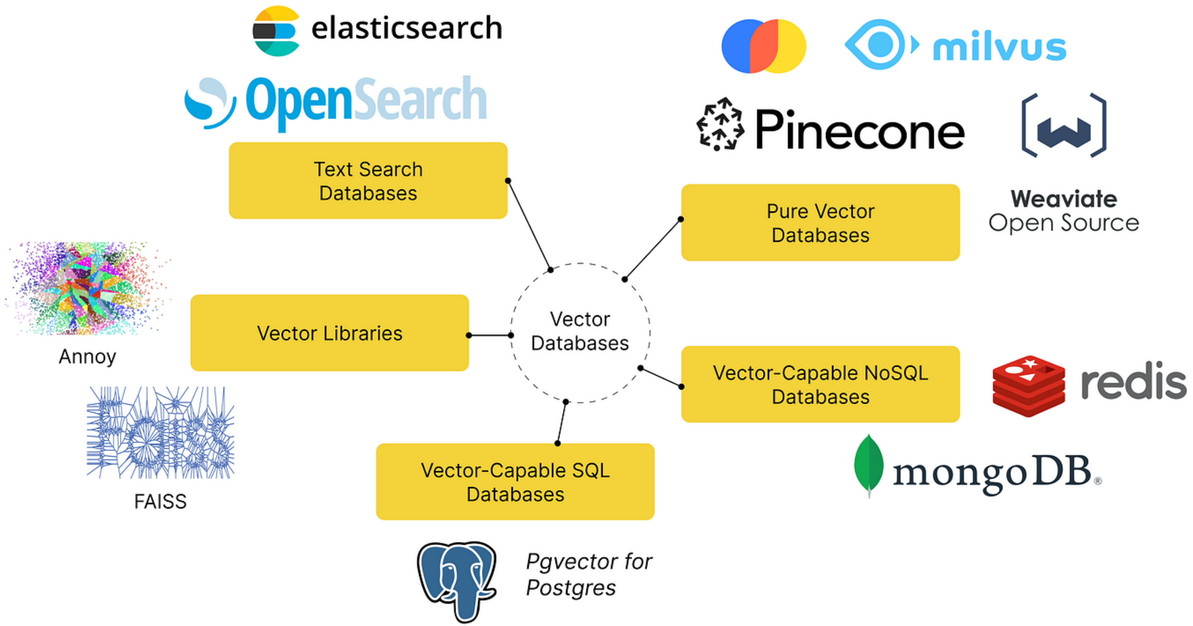
]]

$$cos(2.73°) \approx 0.998$$

RAG Architecture



Vector Stores



https://towardsdatascience.com/all-you-need-to-know-about-vector-databases-and-how-to-use-them-to-augment-your-llm-apps-596f39adfedb

Vector Store vs. „Traditional“ Database

Vector Store	Traditional Database
Indexed unstructured data	Structured data in tables
Results based on similarity	Results based on exact keyword match
Store text/images and their embeddings	Store in database scheme pre-defined attributes
Index groups similar documents	Table groups related attributes

https://towardsdatascience.com/explaining-vector-databases-in-3-levels-of-difficulty-fc392e48ab78

Vector Stores Indexing

- Problem statement: Calculating the similarity between your vector and every database entry is expensive and time-consuming
- Vectors are **indexed**: they are organized to enable an efficient data retrieval.
- Solution: approximate nearest neighbor (ANN) approach
  - Pre-calculate distances between vector embeddings and organise and store similar vectors close to each other
  - Vectors that are similar to each other are stored close to each other in clusters or a graph

We trade in accuracy for speed.

ANN Algorithms

Overview Only:

- Clustering-based index
  - FAISS
- Proximity graph-based index
  - HNSW
- Tree-based index
  - ANNOY
- Hash-based index
  - LSH
- Compression-based index
  - PQ

https://weaviate.io/blog/why-is-vector-search-so-fast

Problem: Our retrieved documents do not seem relevant enough

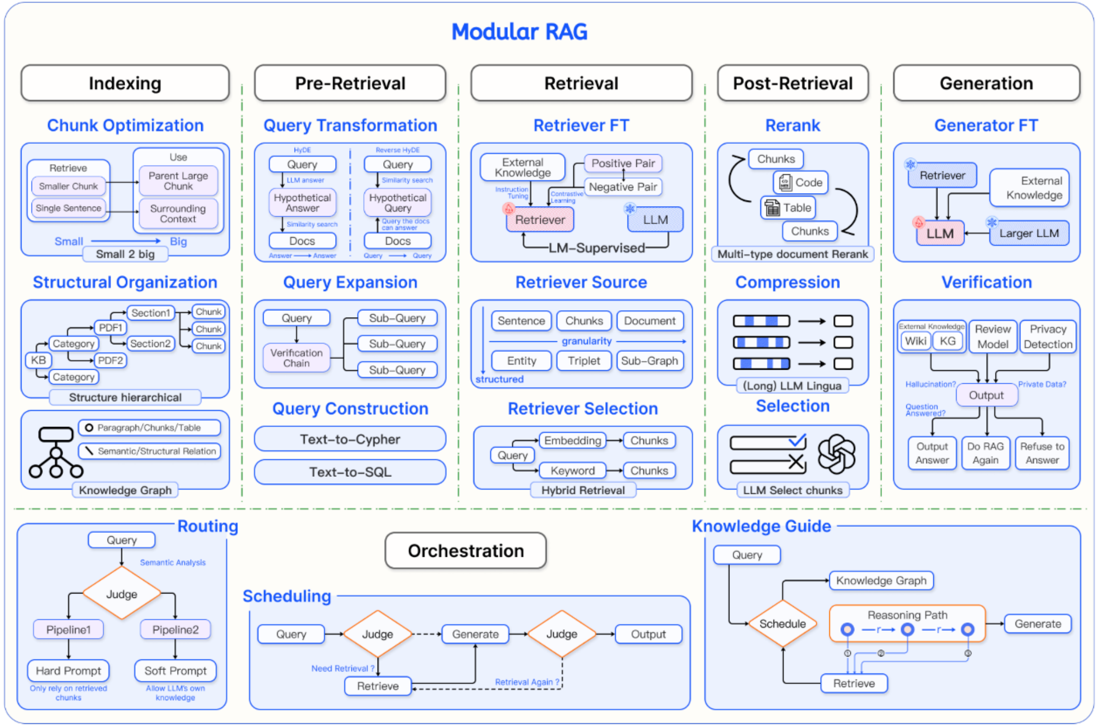
Given cosine similarity for the retriever

Give our database about air travel

Given our original query

What were the first words spoken after the landing on the moon? → **The first landing on the moon was on** 21 July 1969 according to a spokesperson…

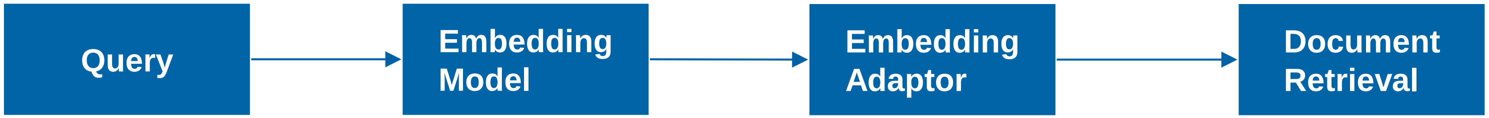
Potential Improvements



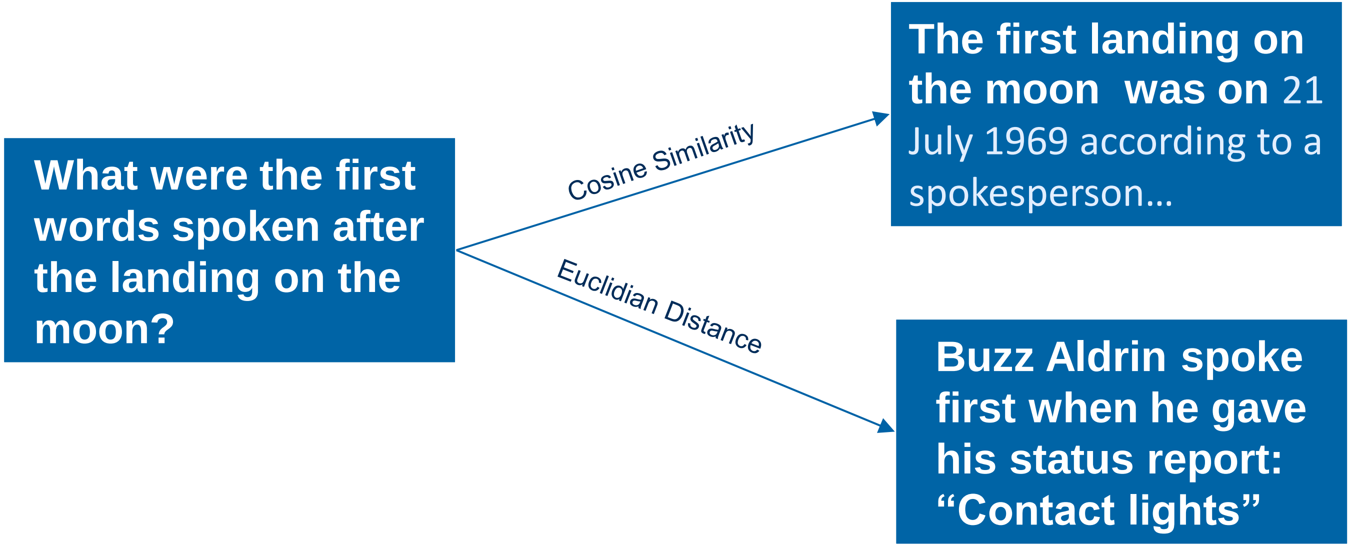
https://towardsdatascience.com/implementing-modular-rag-with-haystack-and-hypster-d2f0ecc88b8f/

Embedding Adaptors

- Goal: transform query embeddings that they align better with the target task's feature space
- Adapter: an alternative to fine-tune an entire pre-trained model, which is implemented as small feed-forward neural networks between layers of pre-trained models
- Advantage: can sustainably improve results
- Disadvantage: requires user feedback/training data for relevant documents



Choice of Similarity Metric



https://medium.com/@stepkurniawan/comparing-similarity-searches-distance-metrics-in-vector-stores-rag-model-f0b3f7532d6f

Hybrid Search

What were the first words spoken after the landing on the moon?

Vector Similarity →

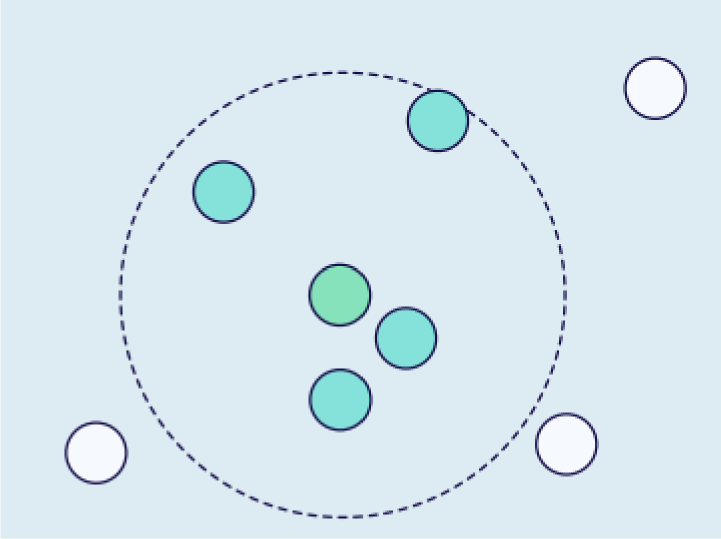
The first landing on the moon was on 21 July 1969 according to a spokesperson…

What were the first words spoken after the landing on the moon?

Key Words: First words spoken →

Buzz Aldrin spoke first when he gave his status report: “Contact lights”

Distance Thresholding



- Classic idea behind top k retrieval
- Decide on a max. acceptable distance between query and chunk vector
- Threshold can be used exclusively or combined with top k retrieval
- Idea: Find maximum possible proximity with predetermined question and answer pairs and maximum negative question and answer pairs with unrelated examples to find a good threshold

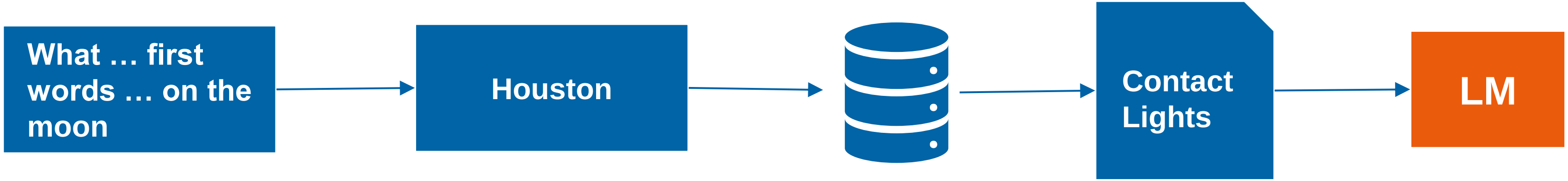
Structure of Base Prompt

- Think about where to inject the retrieved information
- Think about the structure of the injected information
  - Consider using simplified XML tags
  - Information at the beginning and at the end of a prompt is more likely to be considered

Answer the following question: {question}
Base your answer solely on the information given:
<information> {information} </information>

Query Transformation: Query Expansion

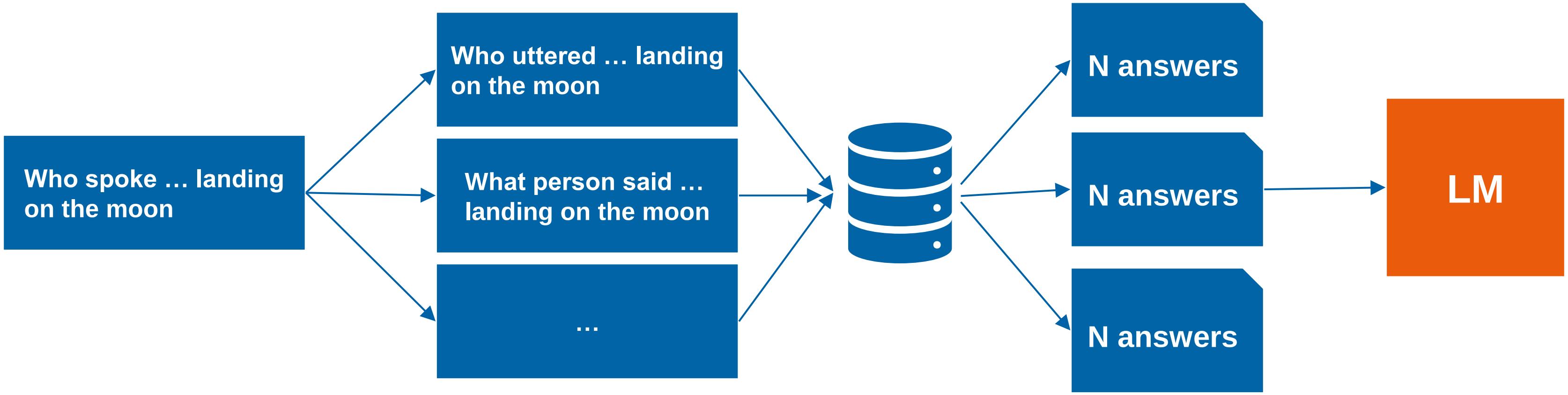
- Expand the query with a hypothetical answer for the question
- Does not really matter whether the answer is correct or not
- Based on the hypothetical answer the result is expected to be better
- The original query may or may not be sent too



https://towardsdatascience.com/3-advanced-document-retrieval-techniques-to-improve-rag-systems-0703a2375e1c

Query Expansion: Query Augmentation / Rewrite

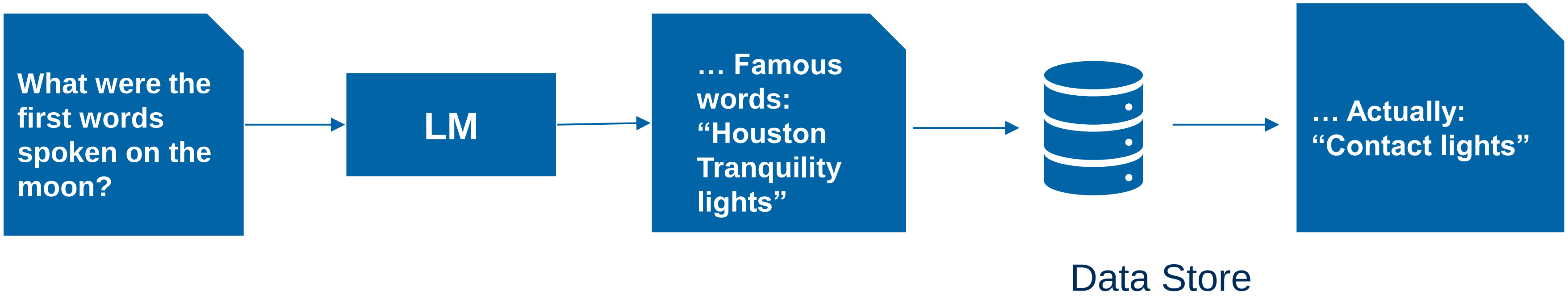
- Ask an LM to rewrite the given query → humans sometimes phrase their questions badly
- Ask the LM to generate queries that are related to the query, ask like an expert would ask, a 5 year old…
- Also works given just a keyword, depending on the prompt for the LM



https://towardsdatascience.com/9-effective-techniques-to-boost-retrieval-augmented-generation-rag-systems-210ace375049

Query Transformation: HyDe

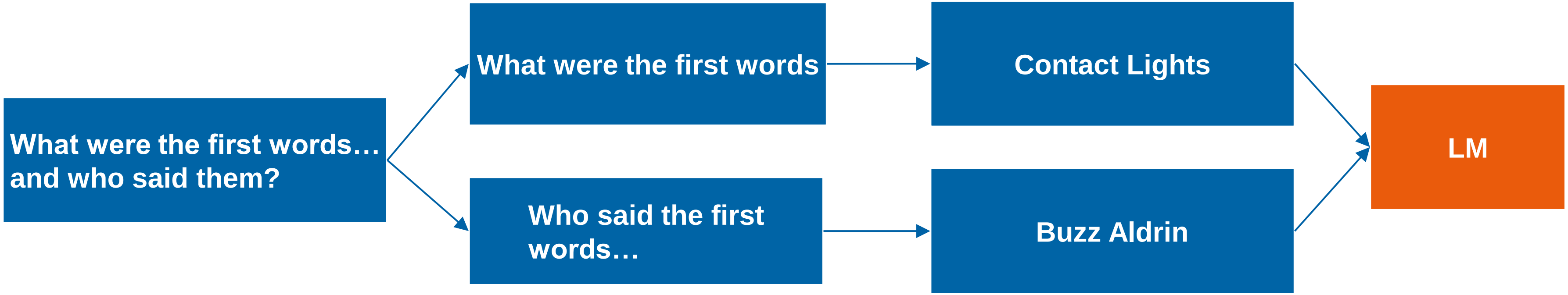
- HyDe: Hypthetical Document Embeddings
- We prompt the LM to generate/hallucinate a paragraph that would answer our question
- We send the hypothetical document to our retriever



https://towardsdatascience.com/3-advanced-document-retrieval-techniques-to-improve-rag-systems-0703a2375e1c

Query Transformations: Subquestions/Query Planning

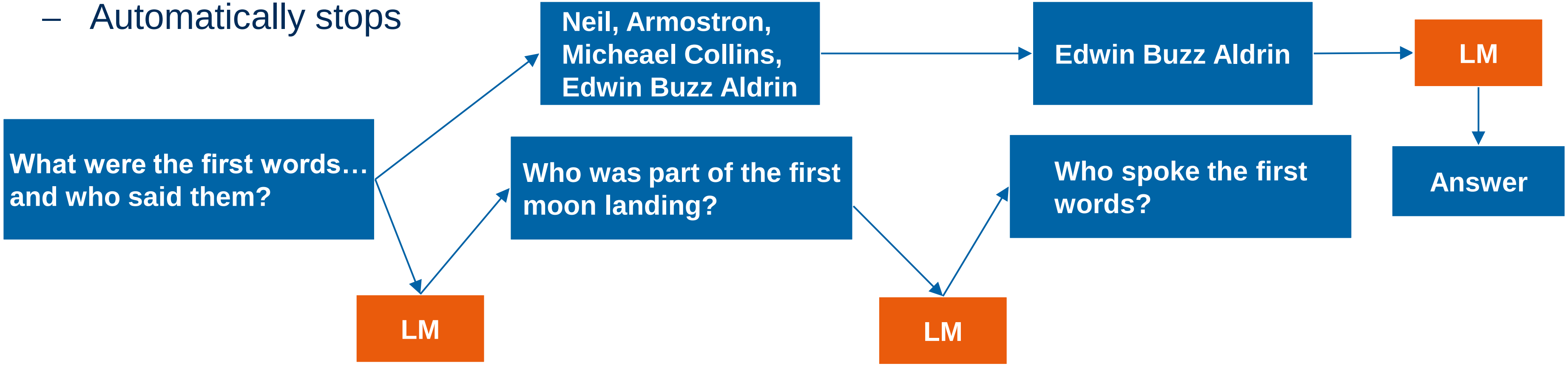
- Divide and Conquer approach to answer complex questions
- Analyze the original question and break them down into smaller subquestions
- Answers to subquestions are concatenated
- Concatenated answers will be used to generate the final question



https://towardsdatascience.com/advanced-query-transformations-to-improve-rag-11adca9b19d1

Query Transformation: Multi-Step Query Transformation

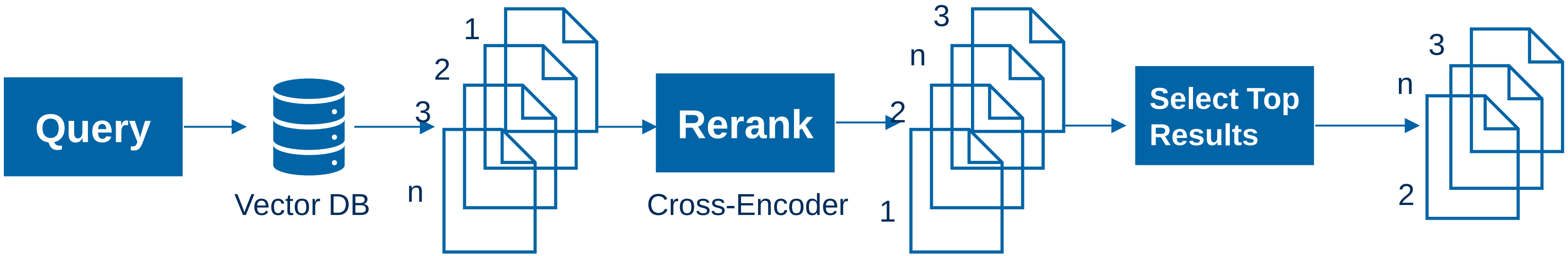
- Based on the self-ask / reasoning method
- LM is asked whether a follow-up question is required to answer the original query
- After each answer, the model will be asked whether it requires more subquestions
- Automatically stops



https://towardsdatascience.com/advanced-query-transformations-to-improve-rag-11adca9b19d1

Cross-encoder Reranking

- "Second opinion" on the relevancy of results
- Retrieve more documents than required
- Cross-encoder compares each result with the query
- Sorts the result in decreasing order



https://towardsdatascience.com/3-advanced-document-retrieval-techniques-to-improve-rag-systems-0703a2375e1c

End-to-end Evaluation of RAGs

- Generate a set of questions (min 30-50) for a subset of documents with an LM
  - Ideally, generate more questions than you need and remove duplicates or add more relevant examples
- Let your RAG application answer all of these questions and store the answer alongside the original question
- Let a high-quality LM (GPT-4 for example) evaluate whether the answer is relevant or not
- Manually double-check the judgments

```
"A sample from the documents is below.\n"
"---------------------\n"
"{context_str}\n"
"---------------------\n"
"Using the documentation sample,
"{query_str} generate more questions similar to the
example"
```

https://betterprogramming.pub/exploring-end-to-end-evaluation-of-rag-pipelines-e4c03221429

Thank you.