“RETRIEVAL AUGMENTED GENERATION”

DATE
09/19/2023

Combined power of large pre-trained language model with external retrieval or search mechanism

AI framework for retrieving facts to ground LLMs on the most accurate information and provide “source” and “data updating” in real time.

PRESENTED BY
Manav Gupta

TABLE OF CONTENTS

01 UNDERSTANDING LIKE A 5-YEAR-OLD.

02 HOW IT WORKS

06 INFORMATION RETRIEVAL

07 EVALUATION

08 RESEARCH PAPERS

09 RETRIEVER: DENSE PASSAGE RETRIEVER

10 USES

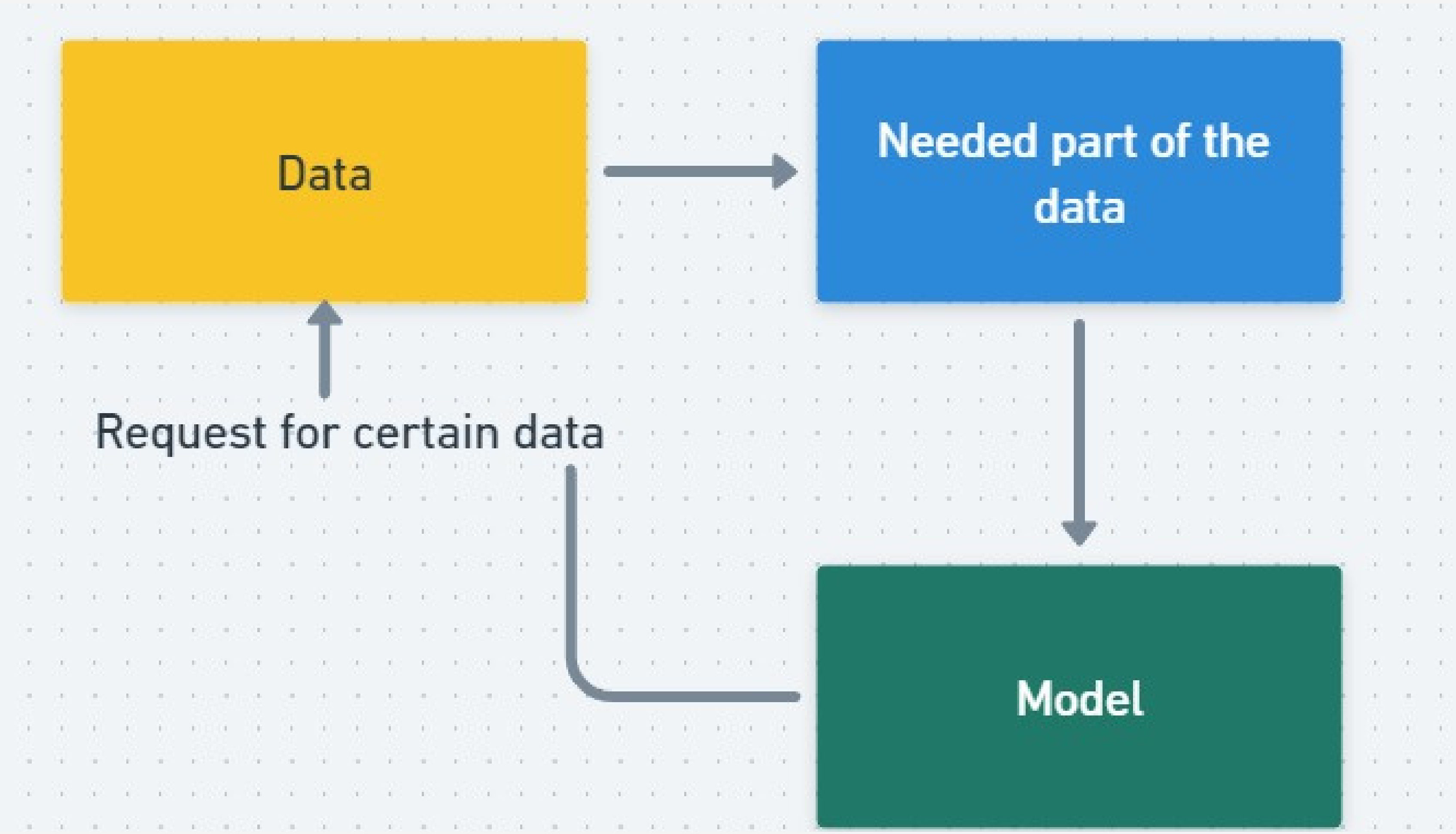
01

I AM A 5 YEAR OLD.

Retrieval:

The model queries an external database (Like a search engine)

- It's like I have a open book exam.
- Precise and accurate knowledge access mechanism
- Trivial to update at test time.



But often limited applicability

- Need retrieval supervison
- “Heuristic” - based retrieval (eg: TF-IDF)
- Need some taslk specific ways to integrate

.

Tech we use: Dense Passage Retrieval
Convert both document and query into embeddings

Augmentation:

It is a system for data updating.

- It automatically updates data in real-time instead of updating each time.
- Using this, the model is less likely to hallucinate.
- It decreases the leaking of data.

Positive Behavior:

- The model knows when to say "I don't know."
- Model doesn't return made-up answers.

Generation:

This is the model which takes retrieved document as content to generate a detailed and coherent response. We use sequential-to-sequential model like BART, GPT2.

- Capture world knowledge and parameters
- Strong results on loads of tasks
- Applicable for almost everything

But,

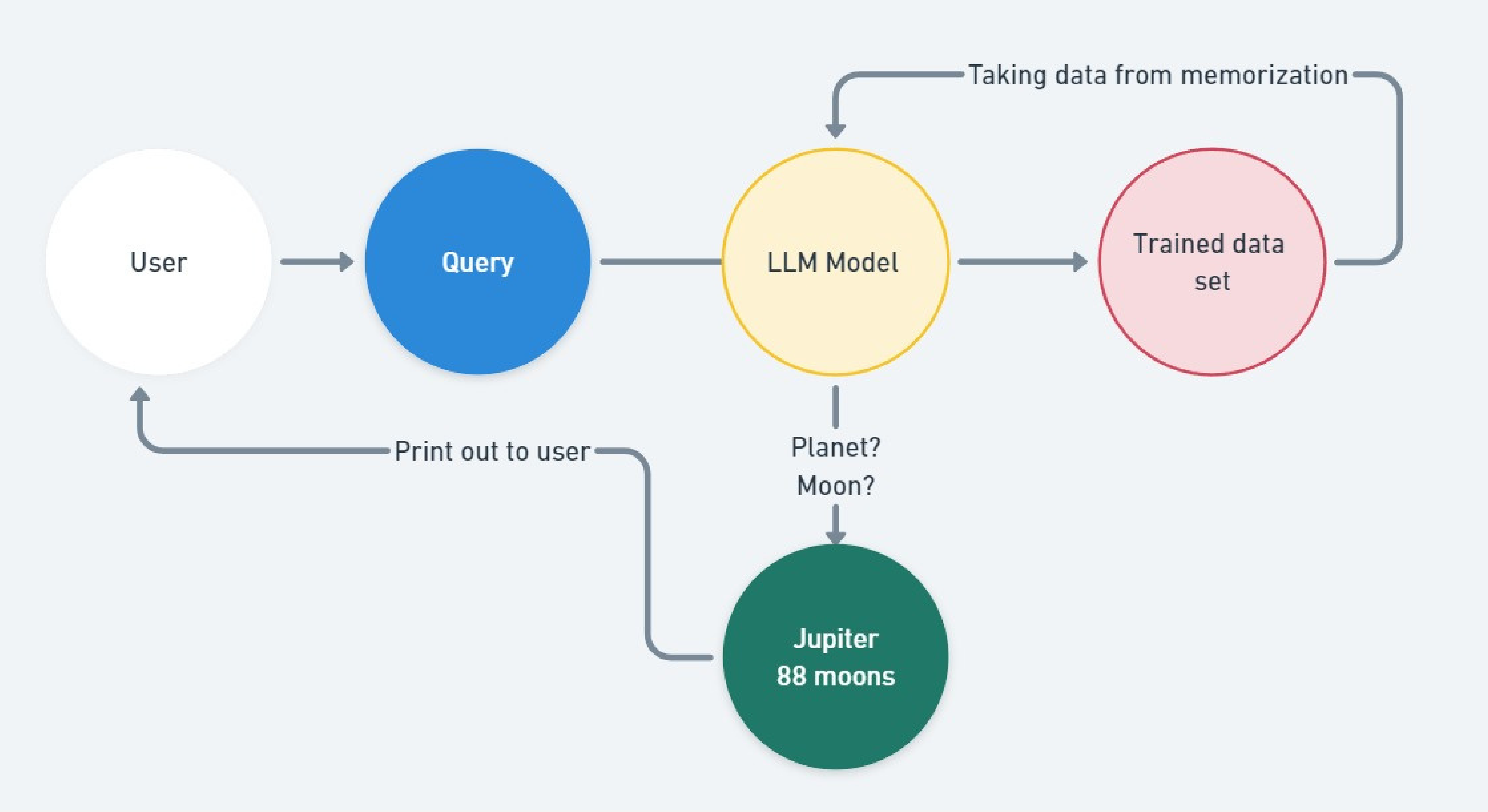
- Hallucinate
- Struggle to access and apply knowledge
- Difficult to update

02 LET'S UNDERSTAND HOW IT WORKS

OLD SCHOOL

LLM that generates text in response to given prompt.

It uses memorization of the data on which it is trained on.



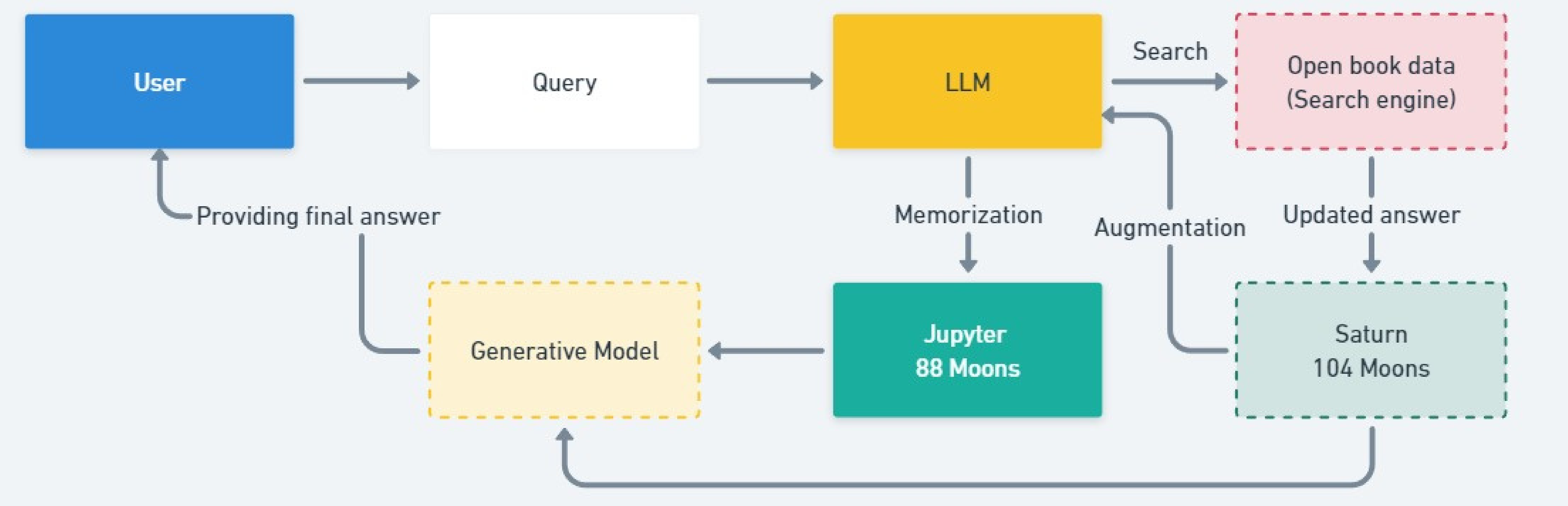
Q: "Which planet has the most number of moons?"

A: Jupiter; it has 88 moons.

NEW SCHOOL

Issue it is solving:

- Source of information : It is providing the source or text corpus
- Data out of date: Provide real time updating in data



While using RAG, it first goes and retrieves relevant content from an open-book search system, then combines it with user problem and then generates answers.

Information Retrieval

Searching for relevant piece of information from a large amount of data to respond to the query.

Relevance : The how well the document match the query

1. Boolean retrival
2. Vector space model
3. TF-IDF

1. Boolean Retrival

Boolean retrieval is a technique used in information retrieval systems to retrieve documents that match a user's query based on Boolean logic (using logical operators such as AND, OR, and NOT)

1.Query Formulation:

- User creates a query using Boolean operators (AND, OR, NOT) to define search conditions (e.g., "cats AND dogs NOT allergies").

2. Tokenization and Preprocessing:

- Query is broken into keywords (e.g., "cats", "dogs", "allergies").
- Unnecessary words like "and", "or", "not" are removed.

3. Boolean Query Evaluation:

- Boolean operations (AND, OR, NOT) are applied to the query terms.
- AND: Retrieve documents with all specified terms.
- OR: Retrieve documents with any specified term.
- NOT: Exclude documents with the specified term.

4. Retrieval and Ranking:

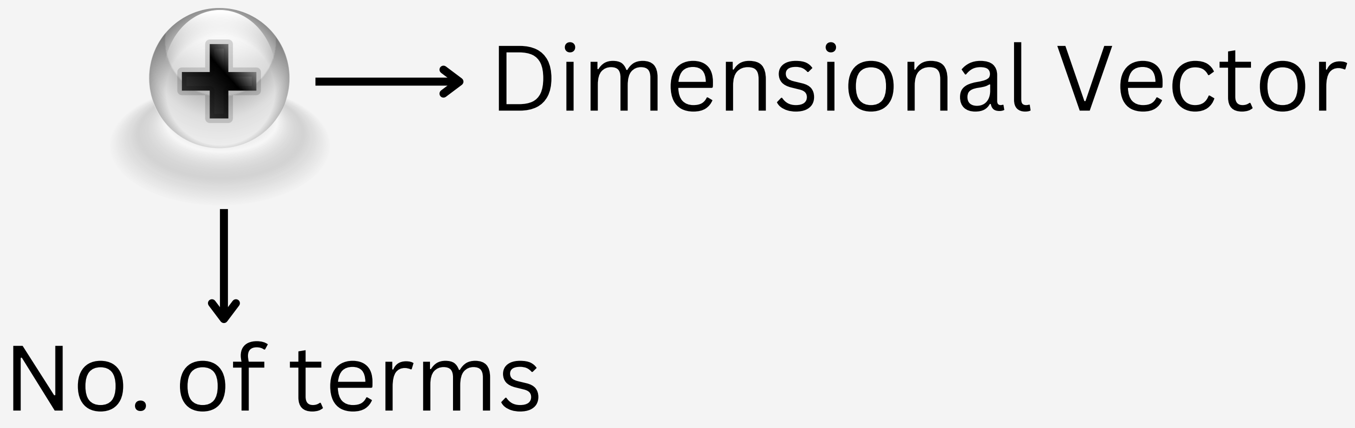
- Retrieve documents that match the Boolean expression.
- Rank the retrieved documents based on relevance to the query.

Merits :

- Result are very predictable and easy to explain tothe user.
- High efficiency

Demerits:

- Equivalent relevance
- Entirely dependent on on user query



Dimensional Vector

No. of terms

2. Vector Space Model

Di=(Di1+Di2+....Dit)

Dj=Weightage

Vector space information retrieval is a method used in information retrieval to represent documents as vectors in a high-dimensional space. Each dimension in this space represents a term or a word. The vector for a document is then constructed by counting the frequency of each term in the document.

	TERM 1	TERM 2	TERM 3
DOC 1	--	--	--
DOC 1	--	--	--
DOC 1	--	--	--

TF-IDF, short for Term Frequency-Inverse Document Frequency

TF-IDF, short for Term Frequency-Inverse Document Frequency, is a numerical statistic that reflects the importance of a term in a document corpus. It assigns a weight to each term based on how often it appears in a document and how rare it is in the entire corpus. This approach is commonly used in natural language processing and information retrieval to identify the most relevant documents for a given query.

TF-IDF for a word in a document is calculated by multiplying two different metrics:

- The term frequency of a word in a document (tf)
- The inverse document frequency(idf) of the word across a set of documents. The closer it is to 0, the more common a word is.

So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach to 1

(tf* idf) results in the TF-IDF score of a word in a document. The higher the score, the more relevant that word is in that particular document.

Evaluation

1. Precision

$$\text{Precision} = \frac{\text{relevant retrieval}}{\text{Total retrieval}}$$

2. Recall

$$\text{Recall} = \frac{\text{No. of relevant retrieval}}{\text{Total no. of relevant}}$$

F-measure

Harmonic mean of precision and recall

$$F = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} \times \text{Recall}}$$

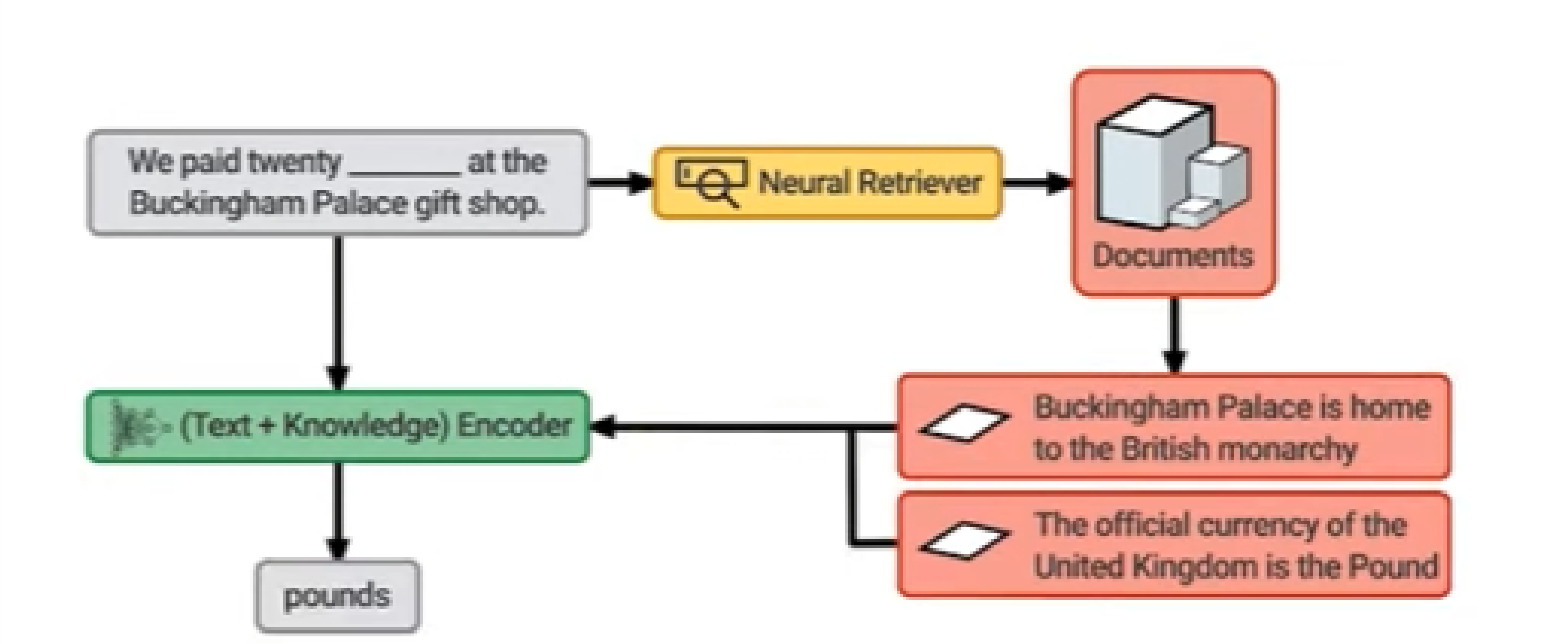
MAP (Mean Average Precision) is the average of precision value of all point of document.

Research Papers

1. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks (Meta, 2020)
2. REALM (Cuu et al 2020)
3. LAMA (Petroni et al. 2019, Petroni et al. 2020)

REALM: Retrieval- Augmented Language Model Pre-Training

REALM is just the BERT based language model (LM) augmented with a retriever.

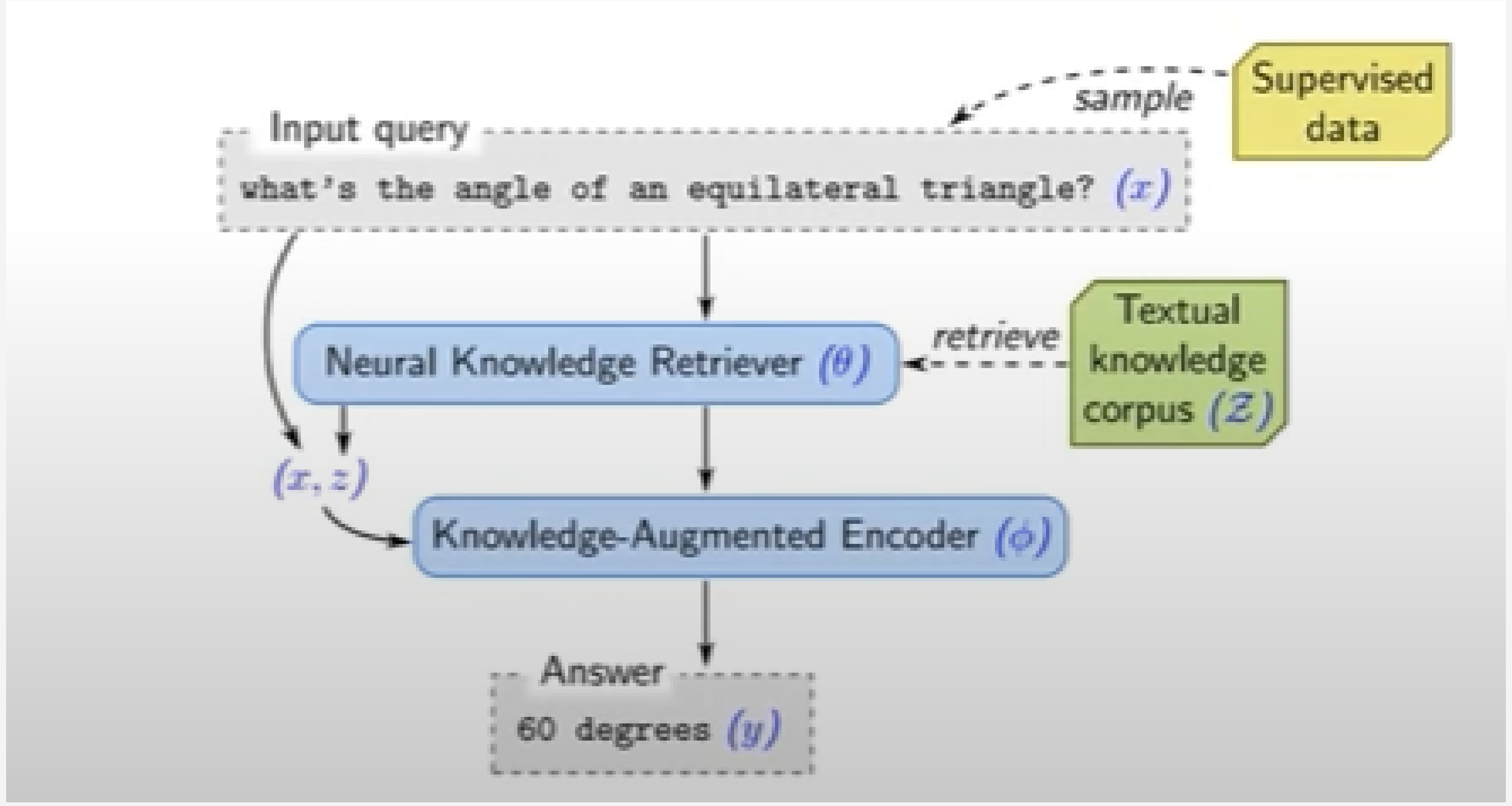

We paid twenty _____ at the Buckingham Palace gift shop.
Neural Retriever
Documents
(Text + Knowledge) Encoder
Buckingham Palace is home to the British monarchy
The official currency of the United Kingdom is the Pound
pounds

Pretraining

- MLM task: predict y given x and retrieved docs z.
- Mask salient spans (entities, dates)
- Allow for null document.
- Prohibit trivial retrievals.

Finetuning

- Open QA task: predict (start, end) spans given question and retrieved docs z.
- Need to find top k (k=5) matching documents efficiently and augment them to input for LM.


Supervised data
sample
Input query
what's the angle of an equilateral triangle? (x)
Neural Knowledge Retriever (θ)
retrieve
Textual knowledge corpus $(\mathcal{Z})$
(x, z)
Knowledge-Augmented Encoder (ϕ)
Answer
60 degrees (y)

Indexing and Results

- Maximum Inner Product Search (MIPS)
- Similarity between question and knowledge corpus documents.
- Pre-compute doc embeddings.
- Refresh periodically (every 500 training steps)
- Initialize embeddings carefully.

Summary

- REALM is LM + knowledge retriever.
- Retriever is supported by knowledge corpus and MIPS.
- Showed great results for OpenQA task.
- Followed by many other studies: RAG, RETRO, ATLAS.

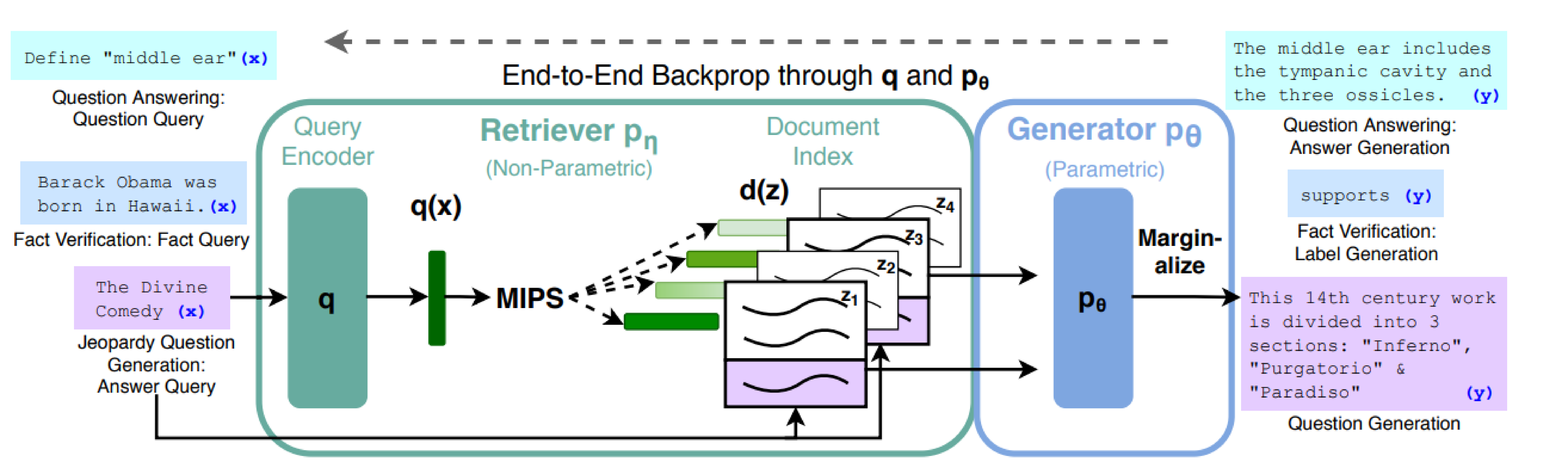
Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks (Meta, 2020)

- Jointly learn to retrieve and generate end2end.
- Latent retrieval - no labels needed for retrieved docs
- General recipe for any seq2seq task

Needs 3 things:

- A (pretrained) generator model P(yl...) e.g. **BART**, GPT2, T5
- A (pretrained) retriever model P(z|) e.g. **DPR**, ICT
- An indexed KB of text documents Z e.g. Wikipedia, CommonCrawl, tweets, ++

RAG models combine **parametric and non-parametric** memory and work well for **knowledge intensive tasks**

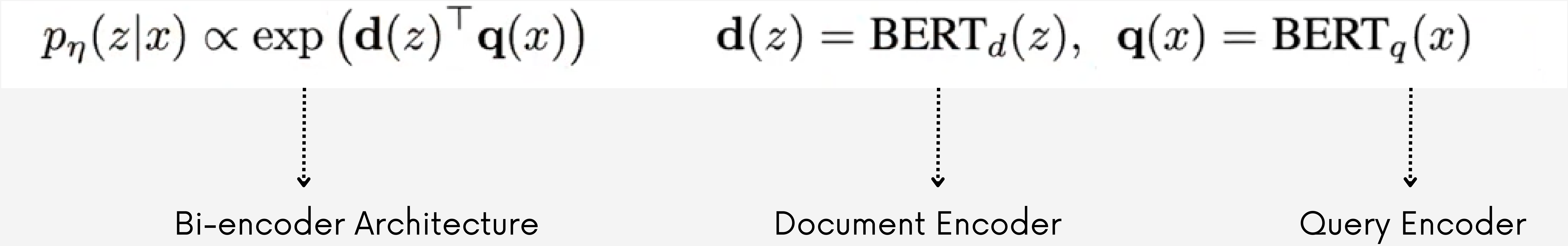

Define "middle ear" (x)
Question Answering: Question Query
Barack Obama was born in Hawaii. (x)
Fact Verification: Fact Query
The Divine Comedy (x)
Jeopardy Question Generation: Answer Query
End-to-End Backprop through q and pθ
Query Encoder
Retriever pη
(Non-Parametric)
Document Index
q
q(x)
MIPS
d(z)
z4
z3
z2
z1
Generator pθ
(Parametric)
pθ
Margin-alize
The middle ear includes the tympanic cavity and the three ossicles. (y)
Question Answering: Answer Generation
supports (y)
Fact Verification: Label Generation
This 14th century work is divided into 3 sections: "Inferno", "Purgatorio" & "Paradiso" (y)
Question Generation

- Follow-up of REALM for NLP
- Parametric pre-trained Seq2Seq BART-Large combined with non-parametric retriever
- Retriever: Query Encoder + Document Index approximation
- BERT-base encoders

- RAG-Sequence: uses same document to predict each target token. Generator produces the output sequence probability for each document, which are then marginalized.

- RAG-Token: predict each target token based on a different document. Generator can choose content from several documents when producing an answer.

Retriever: Dense Passage Retriever



$$p_\eta(z|x) \propto \exp\left(\mathbf{d}(z)^\top \mathbf{q}(x)\right) \qquad \mathbf{d}(z) = \mathrm{BERT}_d(z), \quad \mathbf{q}(x) = \mathrm{BERT}_q(x)$$

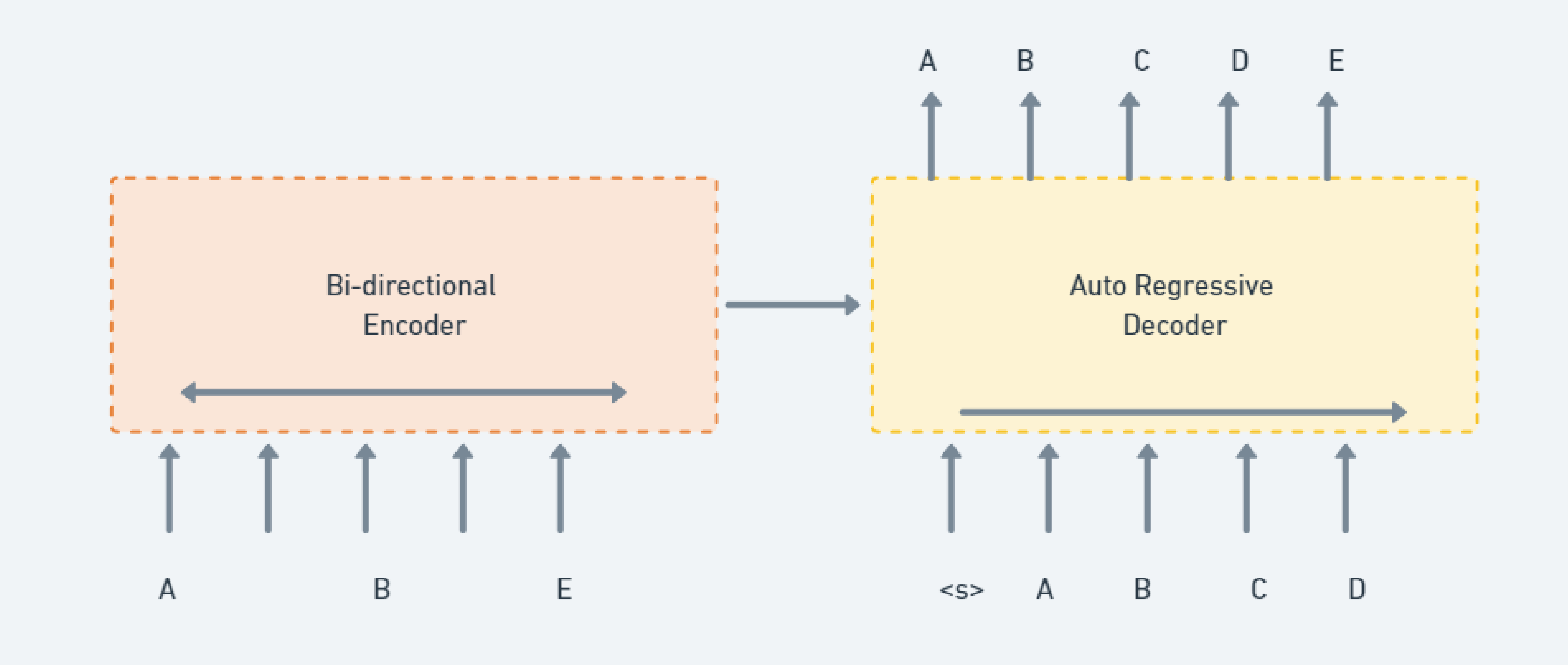
Bi-encoder Architecture

Document Encoder

Query Encoder

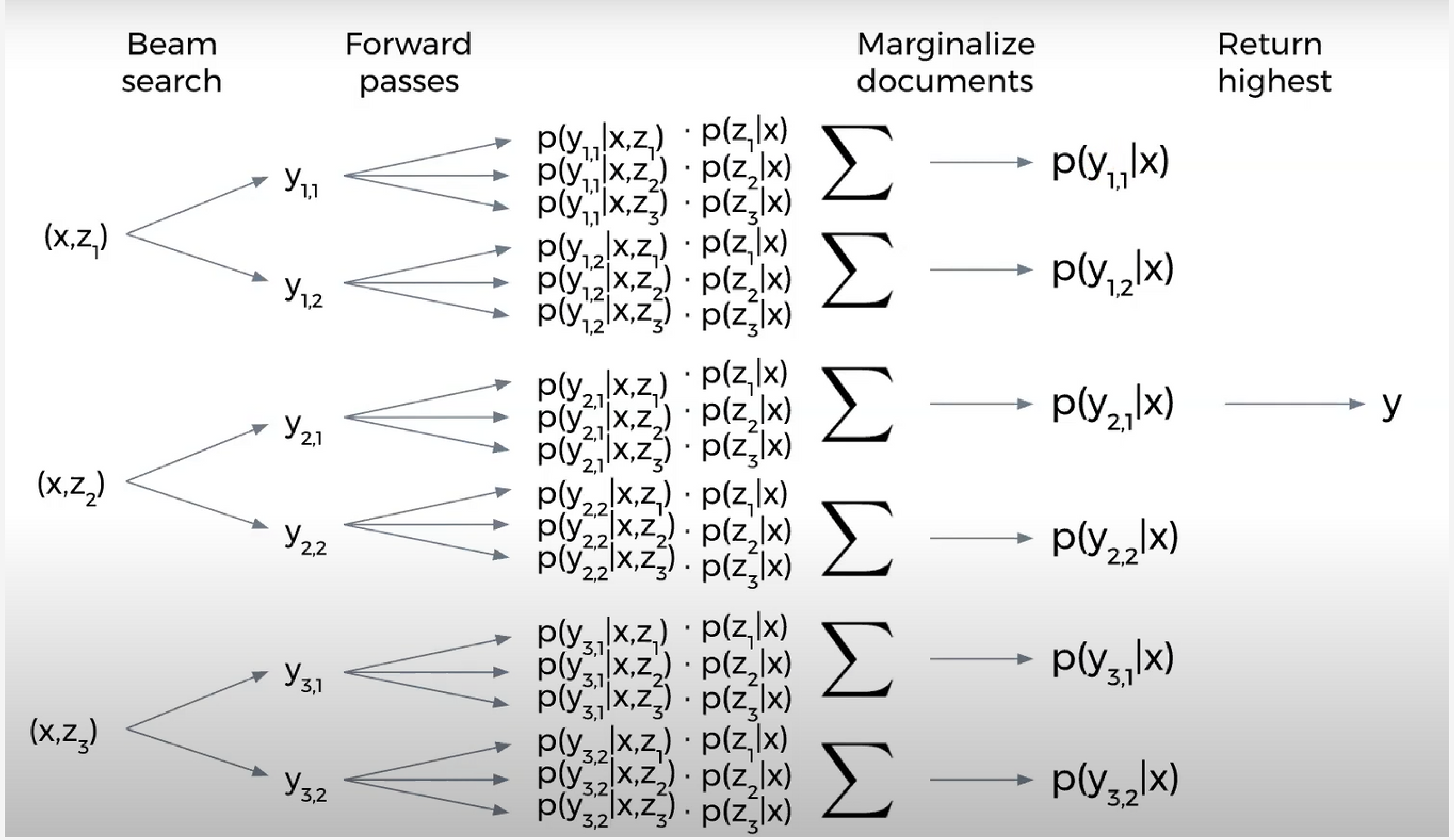
1. Get a pretrained Bi-Encoder
2. Encode Wikipedia Documents Once with Document Encoder
3. Finetune Query Encoder end-to-end with RAG

Generator : BART



- Pretrained Seq2seq model
- RAG Simply concatenates Latent Document z to Input x

Generator


Beam search
Forward passes
Marginalize documents
Return highest
(x,z_1)
(x,z_2)
(x,z_3)
$y_{1,1}$
$y_{1,2}$
$y_{2,1}$
$y_{2,2}$
$y_{3,1}$
$y_{3,2}$
$p(y_{1,1}|x,z_1) \cdot p(z_1|x)$
$p(y_{1,1}|x,z_2) \cdot p(z_2|x)$
$p(y_{1,1}|x,z_3) \cdot p(z_3|x)$
$p(y_{1,2}|x,z_1) \cdot p(z_1|x)$
$p(y_{1,2}|x,z_2) \cdot p(z_2|x)$
$p(y_{1,2}|x,z_3) \cdot p(z_3|x)$
$p(y_{2,1}|x,z_1) \cdot p(z_1|x)$
$p(y_{2,1}|x,z_2) \cdot p(z_2|x)$
$p(y_{2,1}|x,z_3) \cdot p(z_3|x)$
$p(y_{2,2}|x,z_1) \cdot p(z_1|x)$
$p(y_{2,2}|x,z_2) \cdot p(z_2|x)$
$p(y_{2,2}|x,z_3) \cdot p(z_3|x)$
$p(y_{3,1}|x,z_1) \cdot p(z_1|x)$
$p(y_{3,1}|x,z_2) \cdot p(z_2|x)$
$p(y_{3,1}|x,z_3) \cdot p(z_3|x)$
$p(y_{3,2}|x,z_1) \cdot p(z_1|x)$
$p(y_{3,2}|x,z_2) \cdot p(z_2|x)$
$p(y_{3,2}|x,z_3) \cdot p(z_3|x)$
$\sum$
$p(y_{1,1}|x)$
$p(y_{1,2}|x)$
$p(y_{2,1}|x)$
$p(y_{2,2}|x)$
$p(y_{3,1}|x)$
$p(y_{3,2}|x)$
y

Open-domain QA :
Natural Questions, TriviaQA, WebQuestions, CuratedTREC

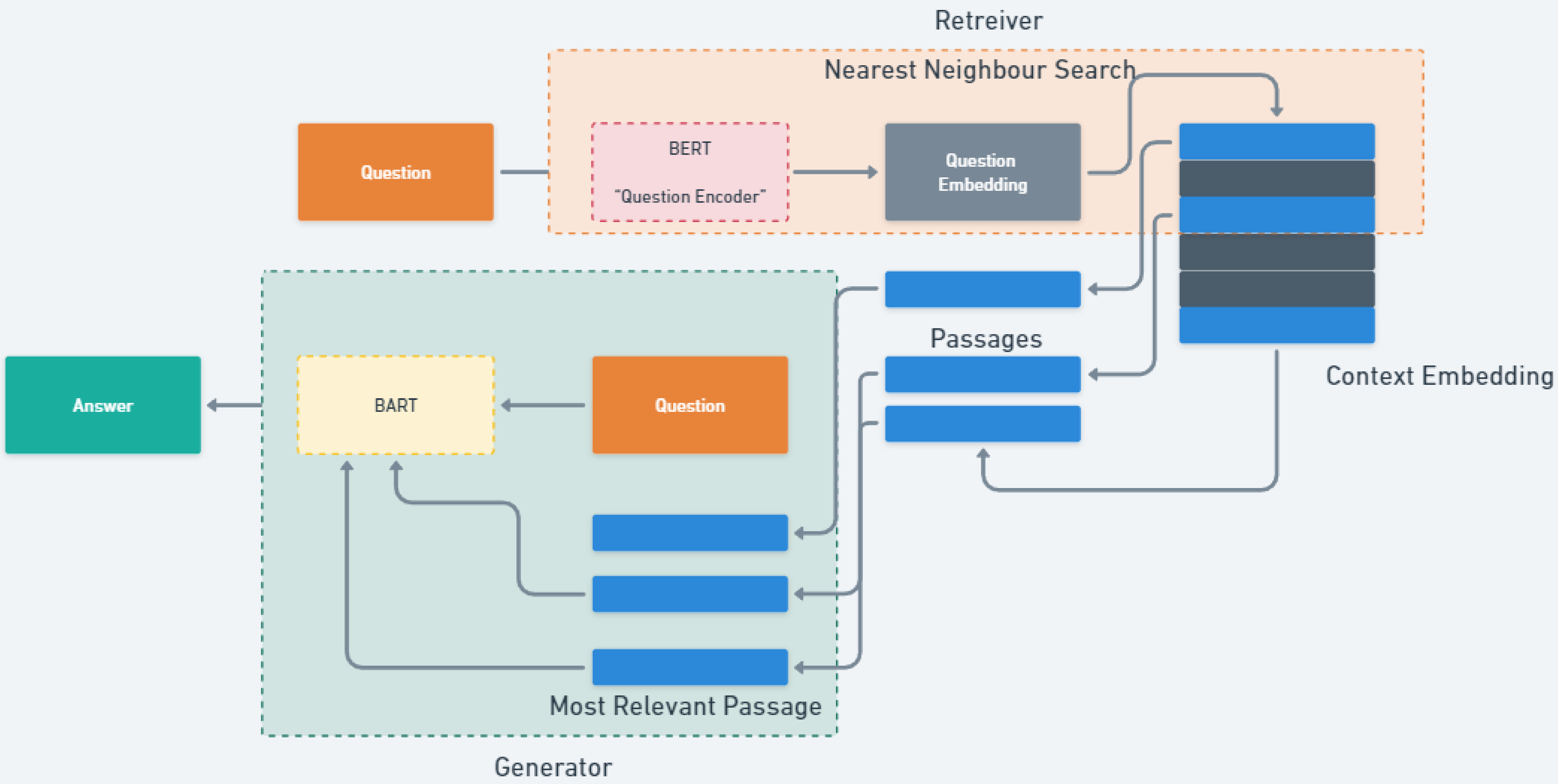
Abstractive open-domain QA:
“Open" MS MARCO

Question Generation: Jeopardy questions

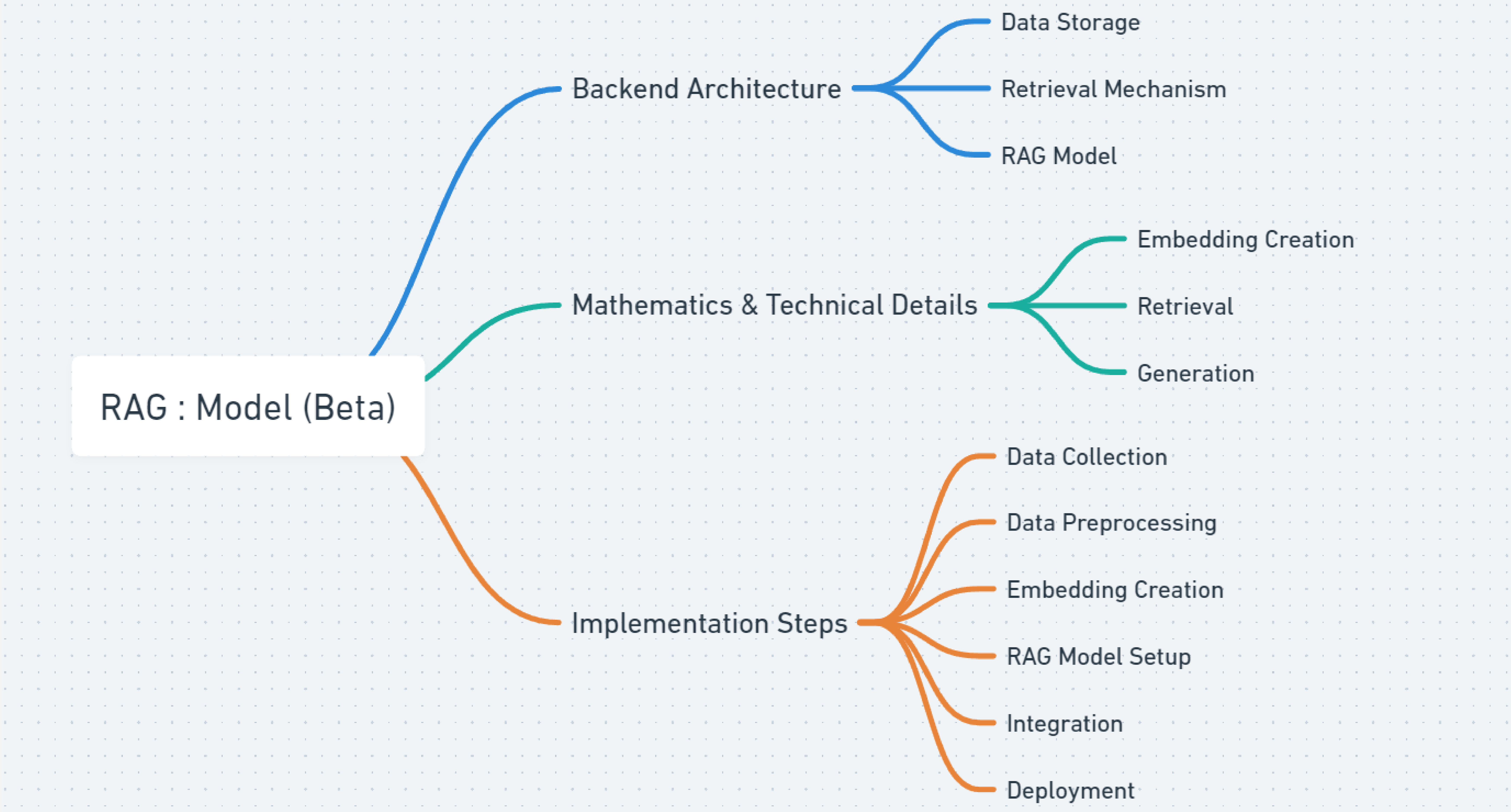
Fact Verification: FEVER

Experiments
RAG can be applied to any task with input and output sequences.
Focus on tasks with a clear need for precisely accessing knowledge

Full process: for chat bot



Our Product



- RAG : Model (Beta)
  - Backend Architecture
    - Data Storage
    - Retrieval Mechanism
    - RAG Model
  - Mathematics & Technical Details
    - Embedding Creation
    - Retrieval
    - Generation
  - Implementation Steps
    - Data Collection
    - Data Preprocessing
    - Embedding Creation
    - RAG Model Setup
    - Integration
    - Deployment

Uses of RAG

- CODE SUMMARIZATION VIA HYBRID GNN
- TEXT-TO-IMAGE GENERATOR
- Open-Domain Question Answering

RETRIEVAL-AUGMENTED GENERATION FOR CODE SUMMARIZATION VIA HYBRID GNN

1. Hybrid Approach: RAG combines the benefits of both retrieval-based and generation-based methods for code summarization.
2. Retrieving Similar Code: For each code sample, RAG retrieves the most similar code from a given retrieval database.

RETRIEVAL-AUGMENTED GENERATION FOR TEXT-TO-IMAGE GENERATOR

1. **Addressing Limitations:** RAG (Re-Imagen) addresses the limitations of existing models that struggle with generating images of uncommon or rare entities.
2. **External Multi-modal Knowledge Base:** Given a text prompt, Re-Imagen accesses an external multi-modal knowledge base to retrieve relevant (image, text) pairs, using them as references for image generation.
3. **Augmentation with Semantics and Visual Details:** The retrieval step augments Re-Imagen with high-level semantics and low-level visual details of the mentioned entities, enhancing the accuracy of generated images.