



41st IEEE International Conference on Data Engineering

— HONG KONG SAR, CHINA | MAY 19 – 23, 2025 —



RAG Meets LLM: Towards Retrieval-Augmented Large Language Models

Website: <https://shorturl.at/j5lGX>

Survey: <https://arxiv.org/pdf/2405.06211.pdf>

Pangjing Wu, Yujuan Ding, Liangbo Ning, Shijie Wang,

Wenqi Fan, and Qing Li

The Hong Kong Polytechnic University

May 20th (Day 2), 14:00-17:00

ICDE 2025, Hong Kong SAR





Tutorial Outline

- **Part 1: Introduction of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)**
- **Part 2: Architecture of RA-LLMs and Main Modules (Dr. Yujuan Ding)**
- **Part 3: Data Management for RA-LLMs (Pangjing Wu)**
- **Part 4: Learning Approach of RA-LLMs (Liangbo Ning)**
- **Part 5: Applications of RA-LLMs (Shijie Wang)**
- **Part 6: Challenges and Future Directions of RA-LLMs**

Website of this tutorial
Check out the slides and more information!

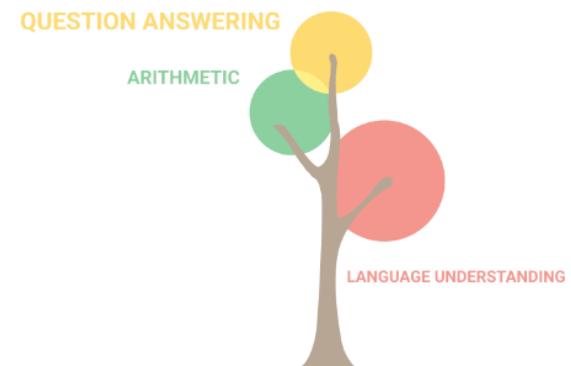


Website

Large Language Models (LLMs)

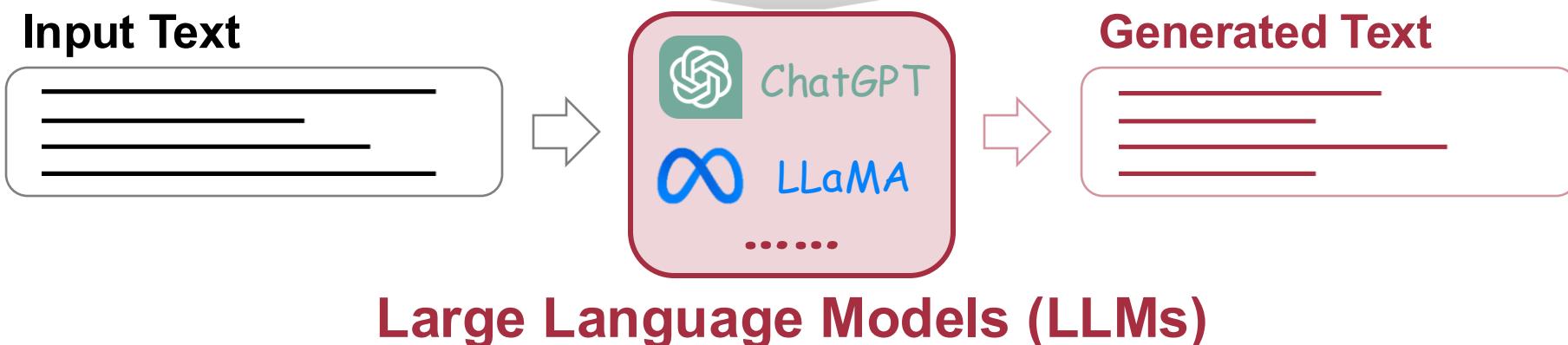
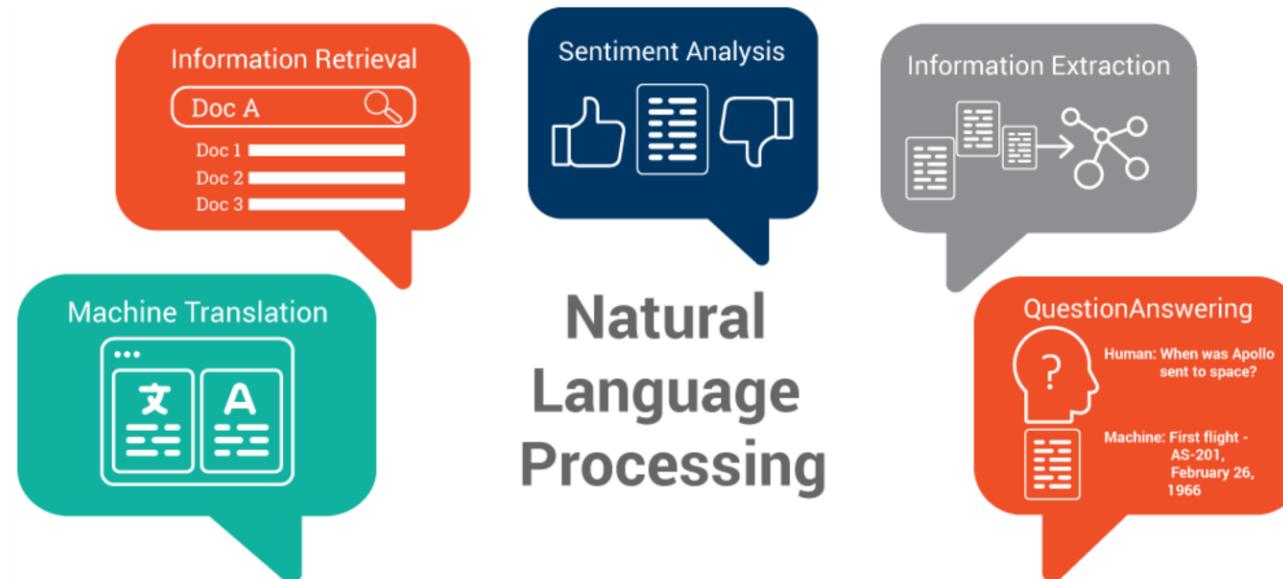


.....

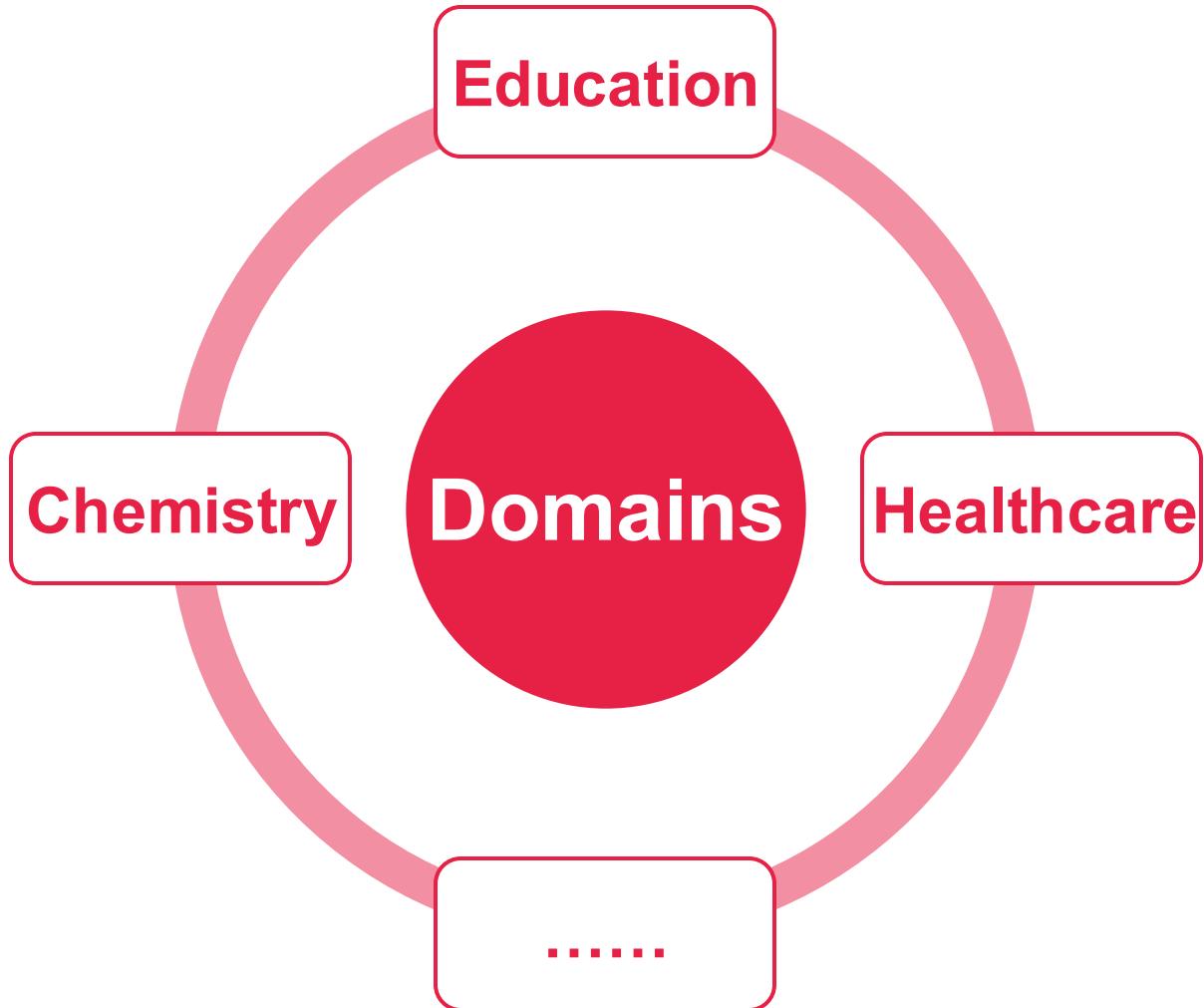


8 billion parameters

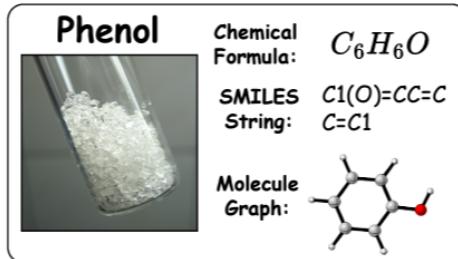
Capability of Large Language Models (LLMs)



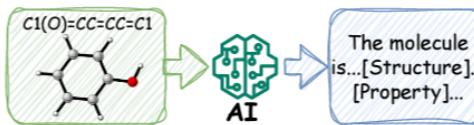
LLMs in Downstream Domains



❑ Molecule discovery, etc.



(a) Molecule Representations.



(b) Molecule Captioning.



ChatGPT

(a) Molecule Captioning

Please show me a description of this molecule:
 $C1=CC=C(C=C1)OC2=CC=CC=C2$

The molecule is an aromatic ether in which the oxygen is attached to two phenyl substituents. It has been found in muscat grapes and vanilla. It has a role as a plant metabolite.

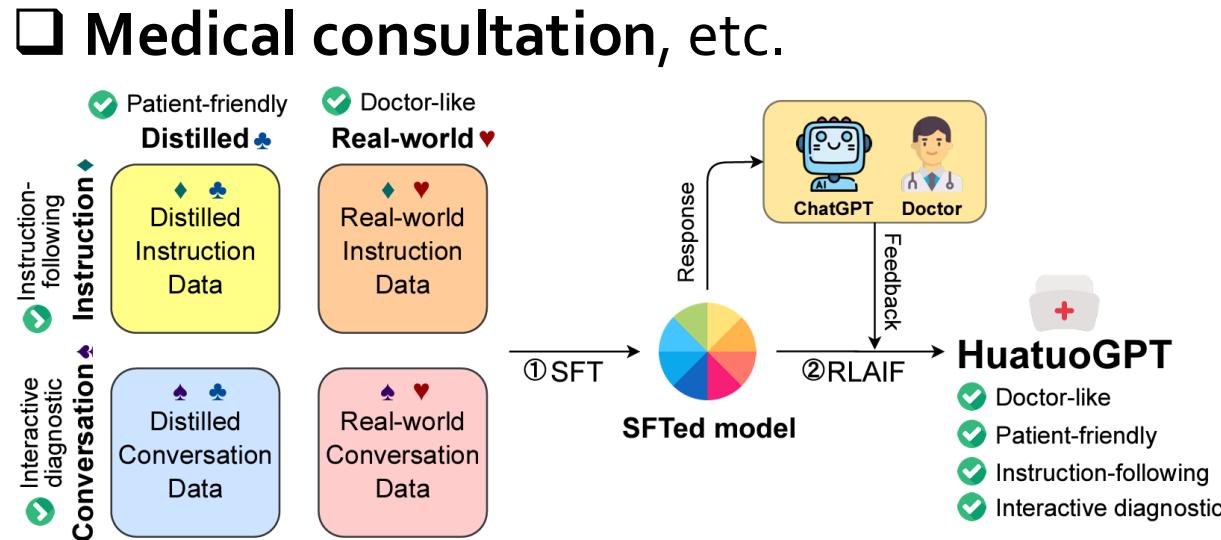
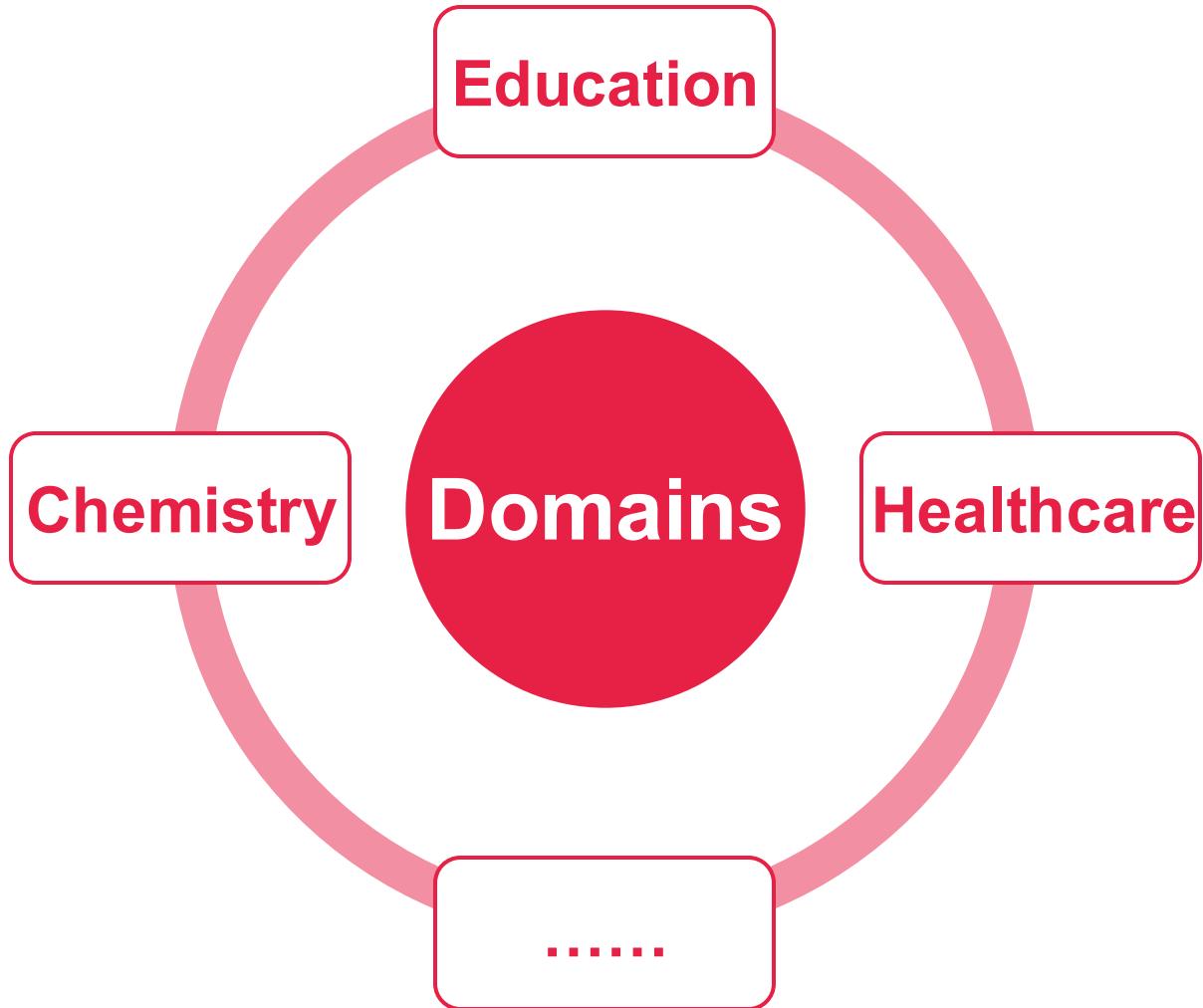
(b) Text-based Molecule Generation

Help me generate a molecule based on the given description:

"The molecule is a quinolinemonocarboxylate that is the conjugate base of xanthurenic acid, obtained by deprotonation of the carboxy group. It has a role as an animal metabolite. It is a conjugate base of a xanthurenic acid."

$C1=CC2=C(C(=C1)[O-])NC(=CC2=O)C(=O)O$

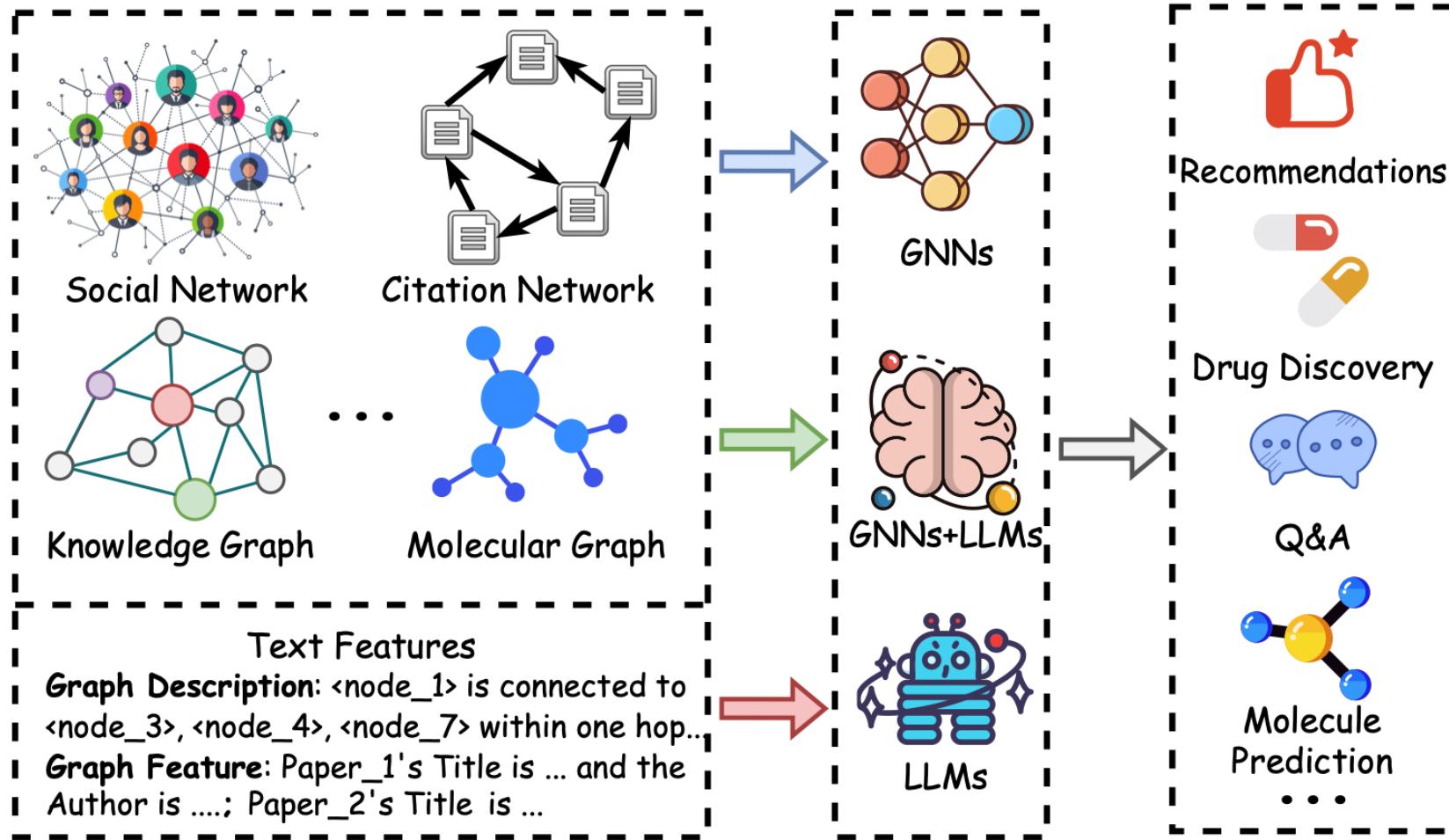
LLMs in Downstream Domains



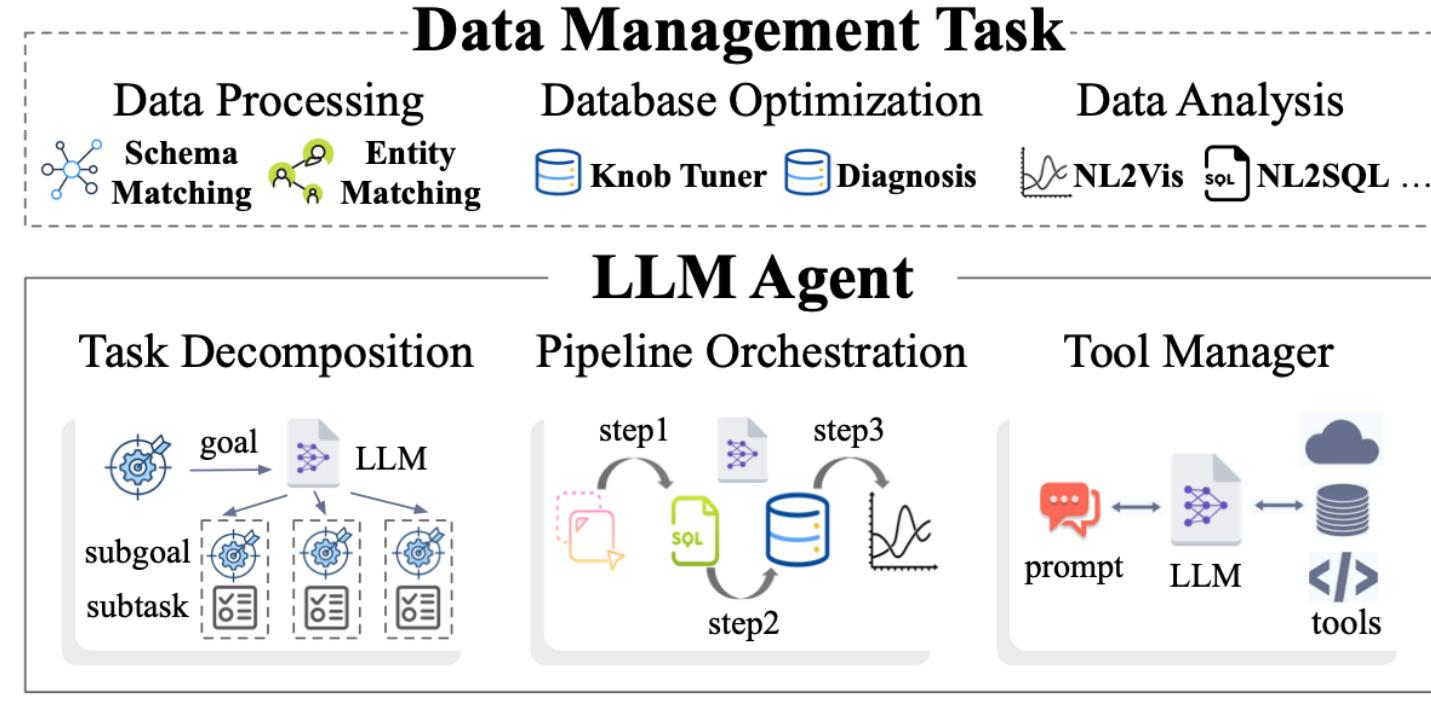
Curriculum & Teaching, etc.



LLMs on Graph-structured Data



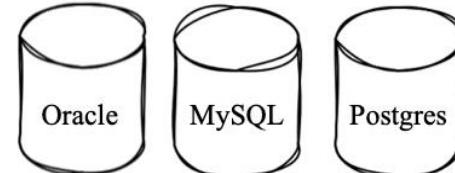
LLMs in Data Engineering Applications



Vector Database



Database Connector



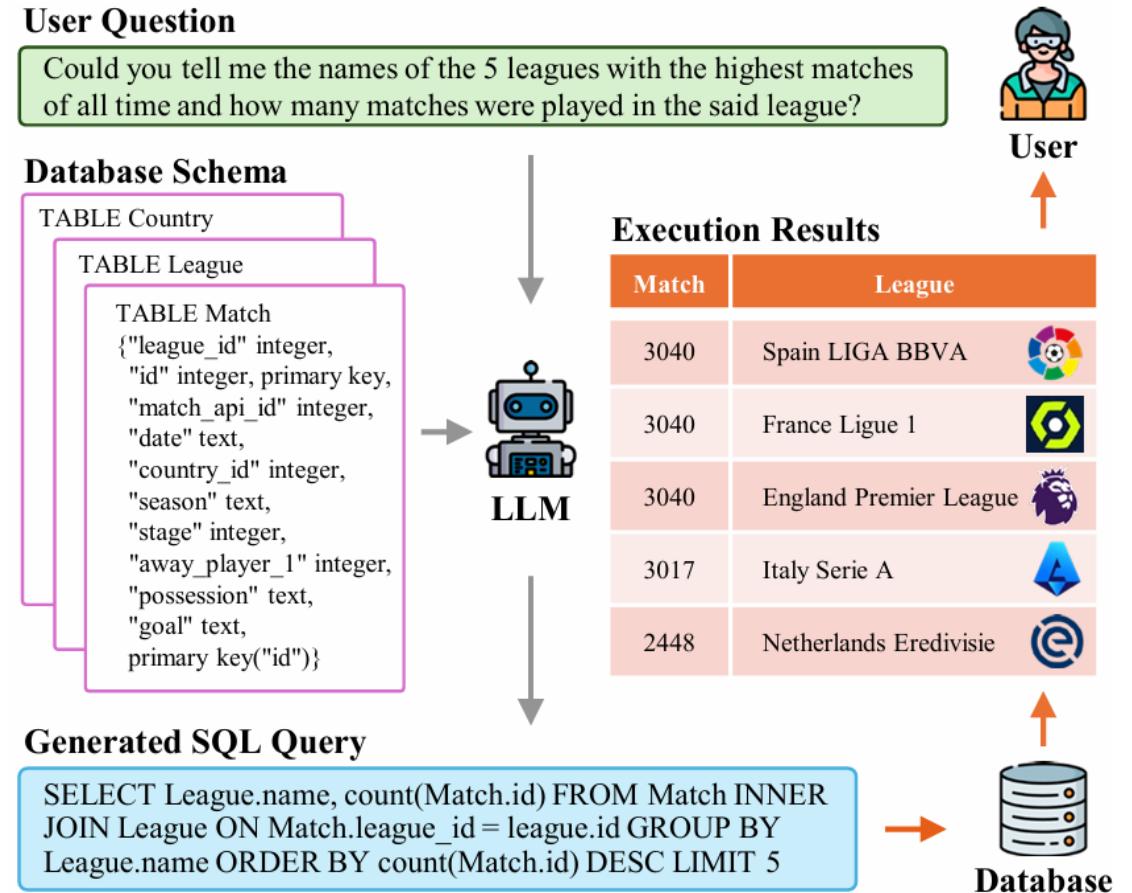
LLM Hub



LLMs in Data Engineering Applications

Text-to-SQL by LLMs

- LLMs enable natural language querying **without SQL skills**.
- LLMs handle **complex questions** with accurate SQL translation.
- LLMs improve **query efficiency** and reduce manual errors.



Challenges and Risks of LLMs

□ Hallucination

the generation of inaccurate, nonsensical, or detached text, posing potential risks and challenges for organizations utilizing these models.



□ Privacy

Various risks to data privacy and security exist at different stages of LLMs, which becomes particularly acute in light of incidents where sensitive internal data was exposed to LLMs.



□ Domain-specific knowledge & expertise

LLMs cannot perform well in many domain-specific fields like medicine, law, finance and more because of the lack of domain-specific knowledge and expertise.



□ Inconsistency

Sometimes they nail the answer to questions, other times they regurgitate random facts from their training data.

LLM challenges in Vertical Domains

□ Domain of Law

OXFORD

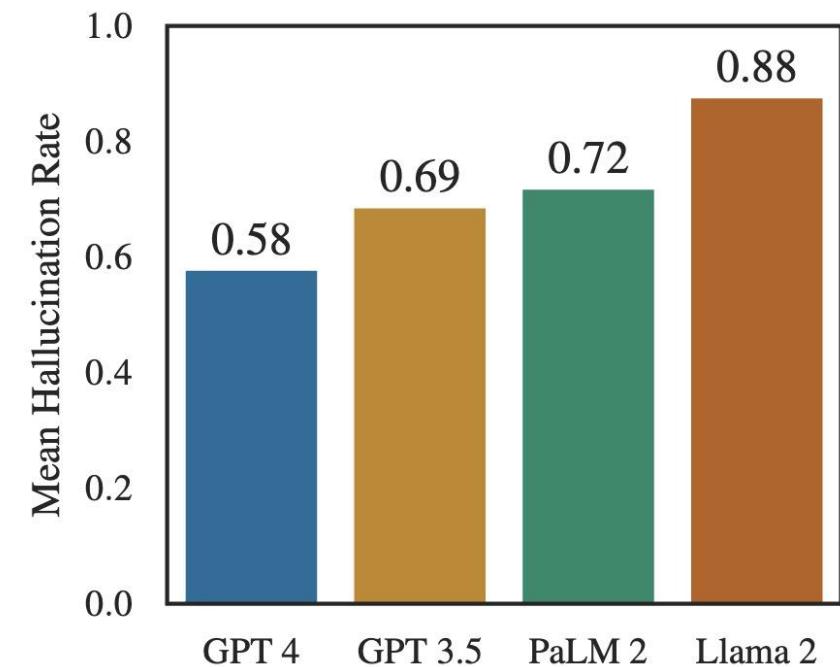
Journal of Legal Analysis, 2024, 16, 64–93
<https://doi.org/10.1093/jla/laae003>
Advance access publication 26 June 2024
Article

Large Legal Fictions: Profiling Legal Hallucinations in Large Language Models

Matthew Dahl[†], Varun Magesh[†], Mirac Suzgun[‡], and Daniel E. Ho[§]

In a new study by Stanford RegLab and Institute for Human-Centered AI researchers, it is demonstrated that legal hallucinations are pervasive and disturbing: hallucination rates range from 69% to 88% in response to specific legal queries for state-of-the-art language models.

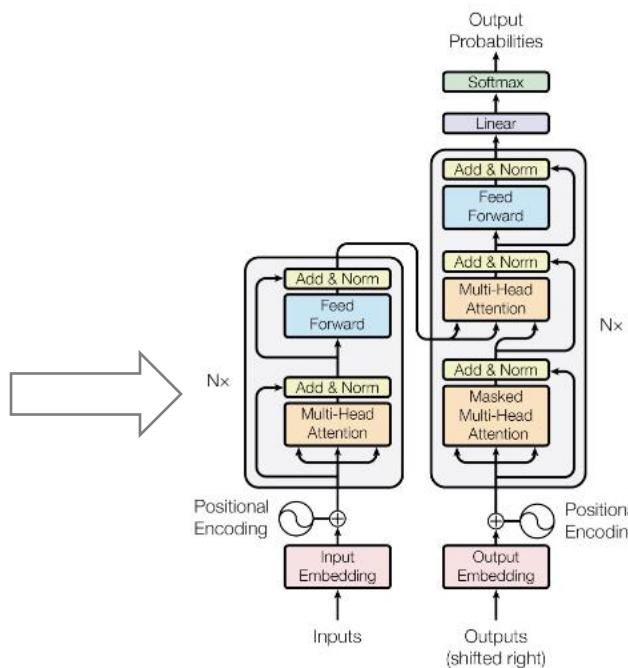
Hallucinations are common across all LLMs when they are asked a direct, verifiable question about a federal court case



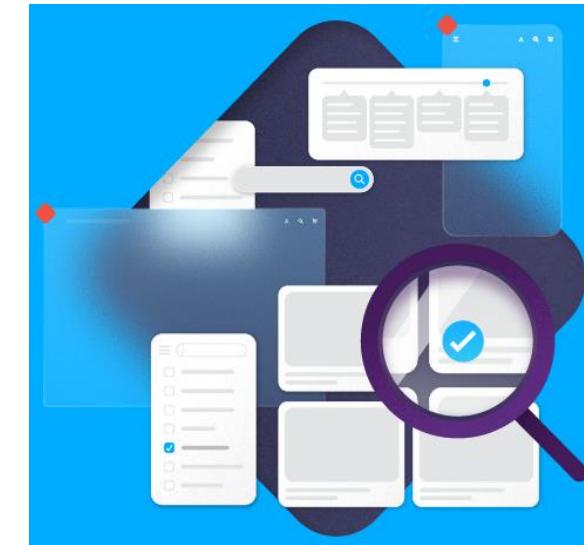
Why Large Language Models Work Well?

- Big Model + Big Training Data

Storing knowledge in the parametric model !



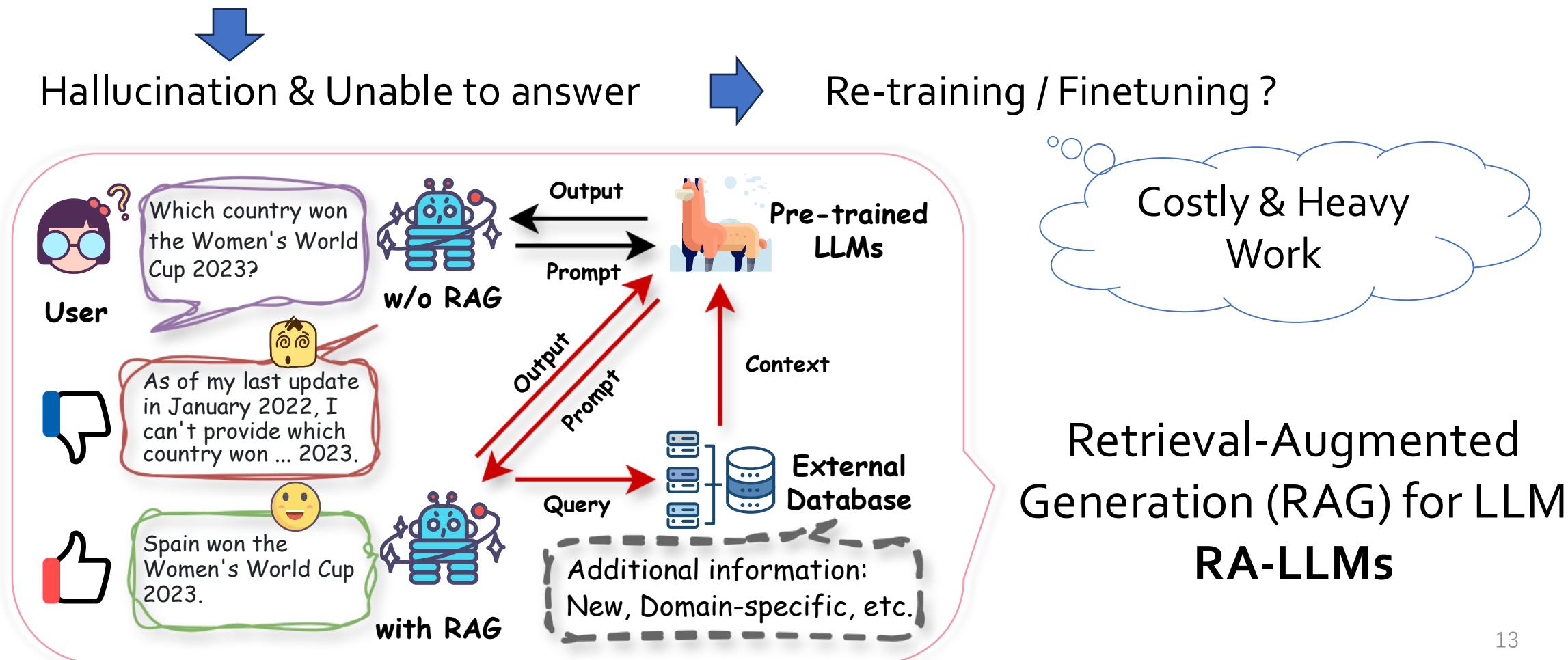
Storing knowledge in the non-parametric model?



Information Retrieval (IR)

Retrieval-Augmented Large Language Models (RA-LLMs)

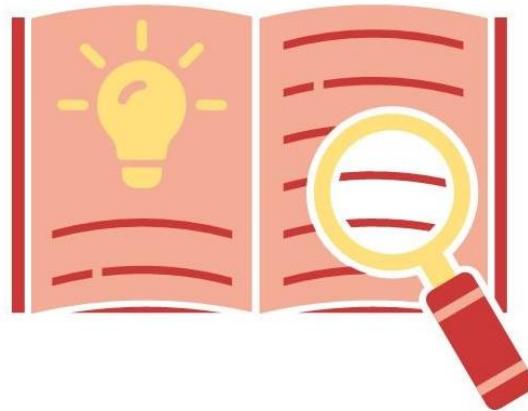
- LLMs **cannot memorize all** (particularly long-tail) knowledge in their parameters
- Lack of **domain-specific knowledge, updated information**, etc



Integrating Information Retrieval in Generation: RA-LLM

External Knowledge Base

- High quality
- Specialized
- Limited scale
- Easy-updated



Information / Knowledge
retrieval

Training Data Base

- Low quality
- General
- Massive
- Hard to update



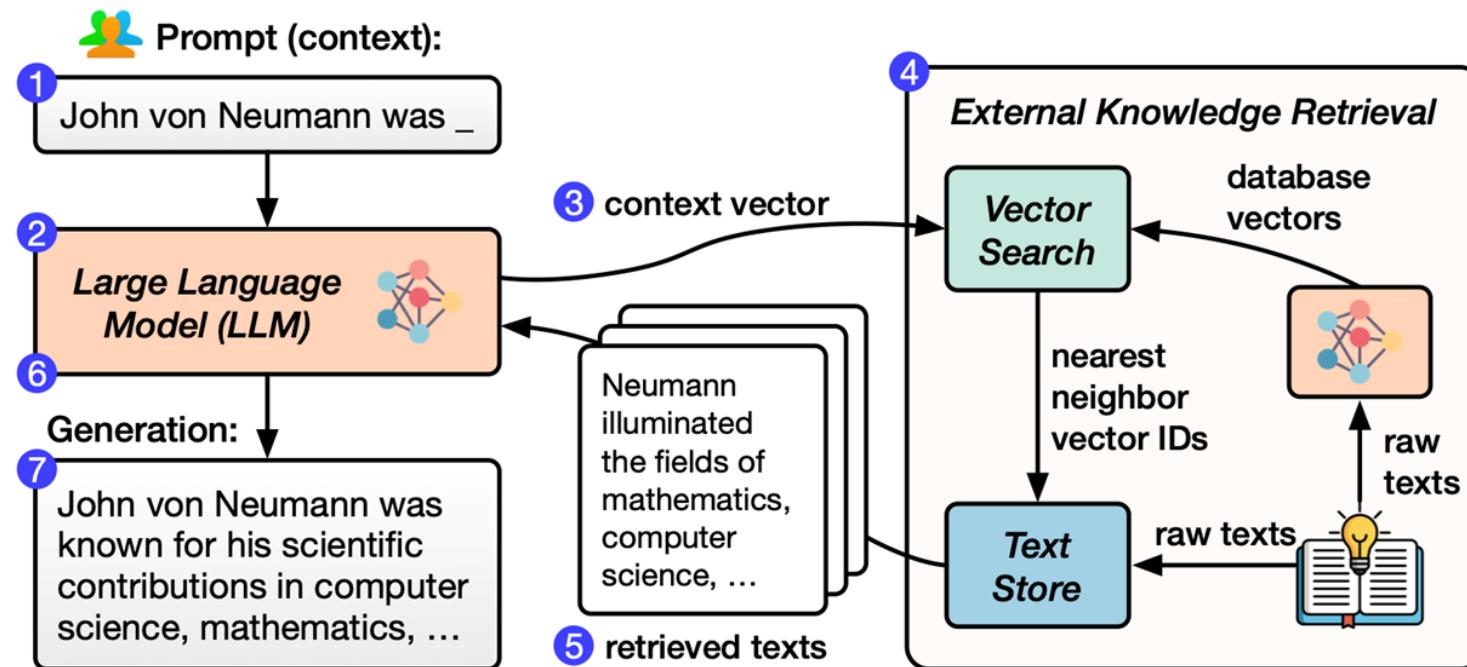
Content generation
Close-book exam
(Hard mode, have to remember everything)



Retrieval augmented generation
Open-book exam
(Easy mode, allow to search in reference)

Data Engineering-Powered RAG

- **HNSW** on hardware reduces memory bottlenecks in search.
- **Disaggregated DB design** scales retrieval and generation independently.
- **FPGA-accelerated ANN** boosts retrieval speed and throughput.



Integrating Data Management in Generation

- Document Management
- Embedding
 - Fast indexing
 - Optimizing Storage
 - Query Optimization



Documents



Vector DB

*Create outlines of
documents, and quickly find
knowledge)*



Content generation



Efficient and accurate
retrieval augmented
generation

A Comprehensive Survey Paper

A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models

Wenqi Fan

wenqifan03@gmail.com

The Hong Kong Polytechnic University, HK SAR

Yujuan Ding*

dingyujuan385@gmail.com

The Hong Kong Polytechnic University, HK SAR

Liangbo Ning

BigLemon1123@gmail.com

The Hong Kong Polytechnic University, HK SAR

Shijie Wang

shijie.wang@connect.polyu.hk

The Hong Kong Polytechnic University, HK SAR

Hengyun Li

neilhengyun.li@polyu.edu.hk

The Hong Kong Polytechnic University, HK SAR

Dawei Yin

yindawei@acm.org

Baidu Inc, China

Tat-Seng Chua

dcscts@nus.edu.sg

National University of Singapore,
Singapore

Qing Li

csqli@comp.polyu.edu.hk

The Hong Kong Polytechnic University, HK SAR

Accepted by KDD'2024
<https://arxiv.org/pdf/2405.06211.pdf>



Recruitment

- Our research group are actively recruiting self-motivated **postdoc**, **Ph.D. students**, and **research assistants**, etc. **visiting scholars, interns, and self-funded students** are also welcome. Send me an email if you are interested.
- ❖ Research areas: machine learning (ML), data mining (DM), artificial intelligence (AI), deep learning (DNNs), large language models (LLMs), graph neural networks (GNNs), computer vision (CV), natural language processing (NLP), etc.
- ❖ Position details:
<https://wenqifano3.github.io/openings.html>





Tutorial Outline

- **Part 1: Introduction of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)**
- **Part 2: Architecture of RA-LLMs and Main Modules (Dr. Yujuan Ding)**
- **Part 3: Data Management for RA-LLMs (Pangjing Wu)**
- **Part 4: Learning Approach of RA-LLMs (Liangbo Ning)**
- **Part 5: Applications of RA-LLMs (Shijie Wang)**
- **Part 6: Challenges and Future Directions of RA-LLMs**

Website of this tutorial
Check out the slides and more information!



PART 2: Architecture of RA-LLMs and Main Modules

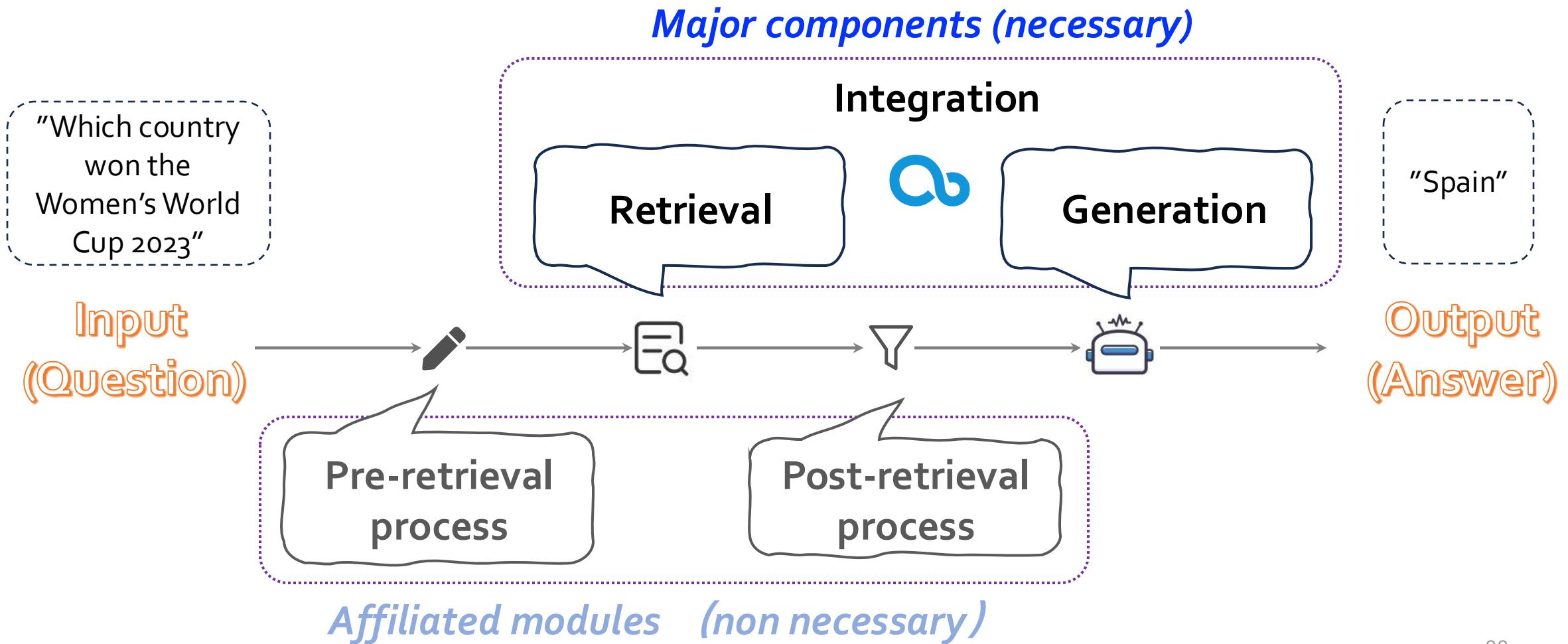


Presenter
Dr. Yujuan DING
HK PolyU

- **RA-LLM architecture overview**
- **Retriever in RA-LLMs**
- **Retrieval results integration**
- **Pre/Post-retrieval techniques**
- **Special RA-LLM paradigms**

RA-LLM Architecture: Standard Pipeline

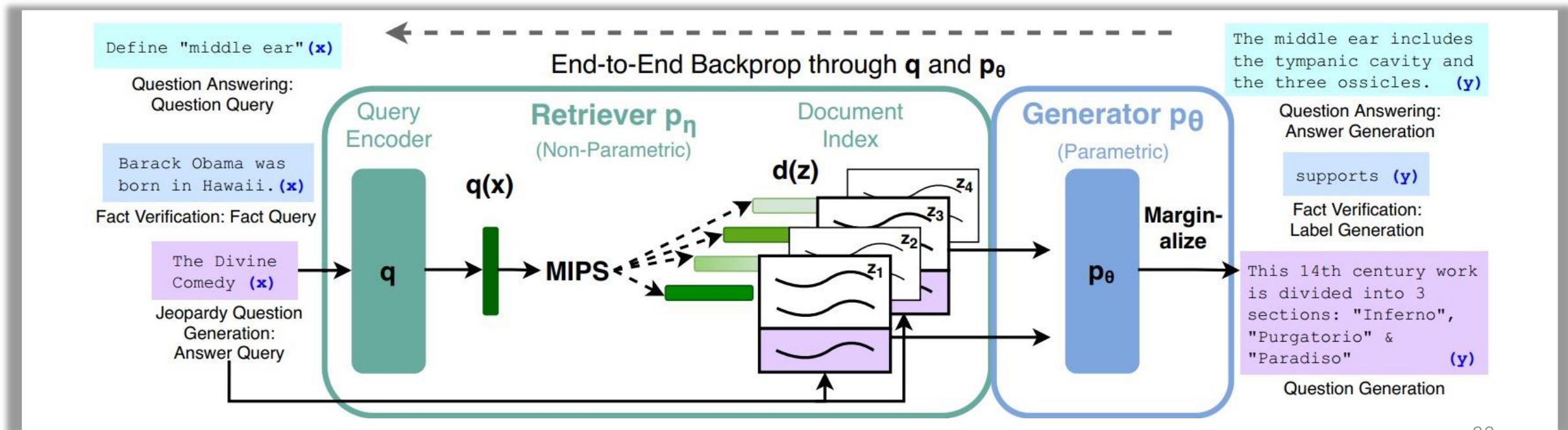
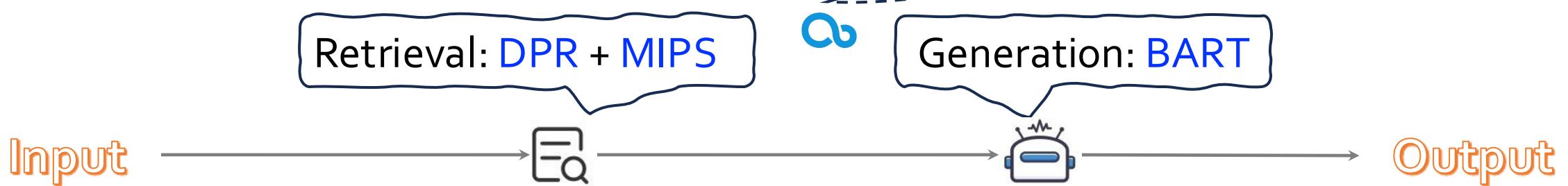
- Technical component illustration in a RA-LLM for the Q&A task



A Simple Retrieval-Augmented Generation Model

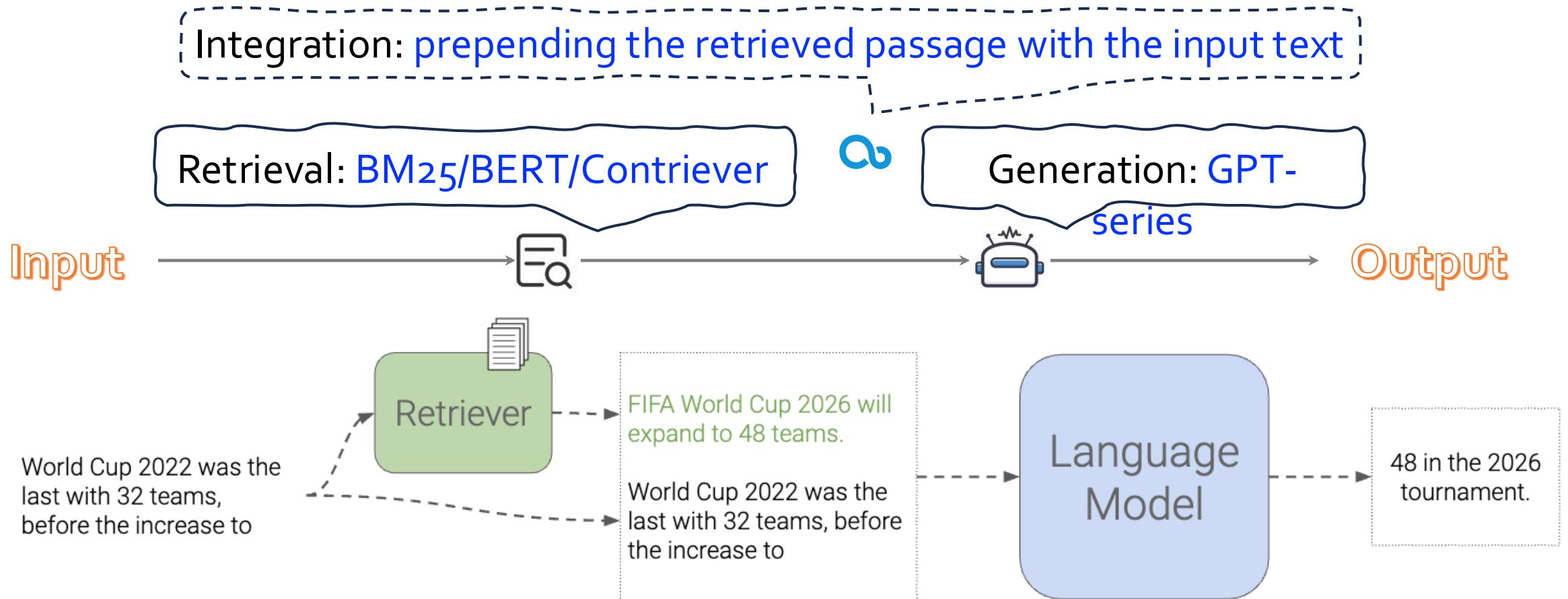
❑ RAG

Integration: concatenating each retrieved passage with the question



A Simple Retrieval-Augmented Generation Model

□ In-Context RALM



PART 2: Architecture of RA-LLMs and Main Modules



Slides

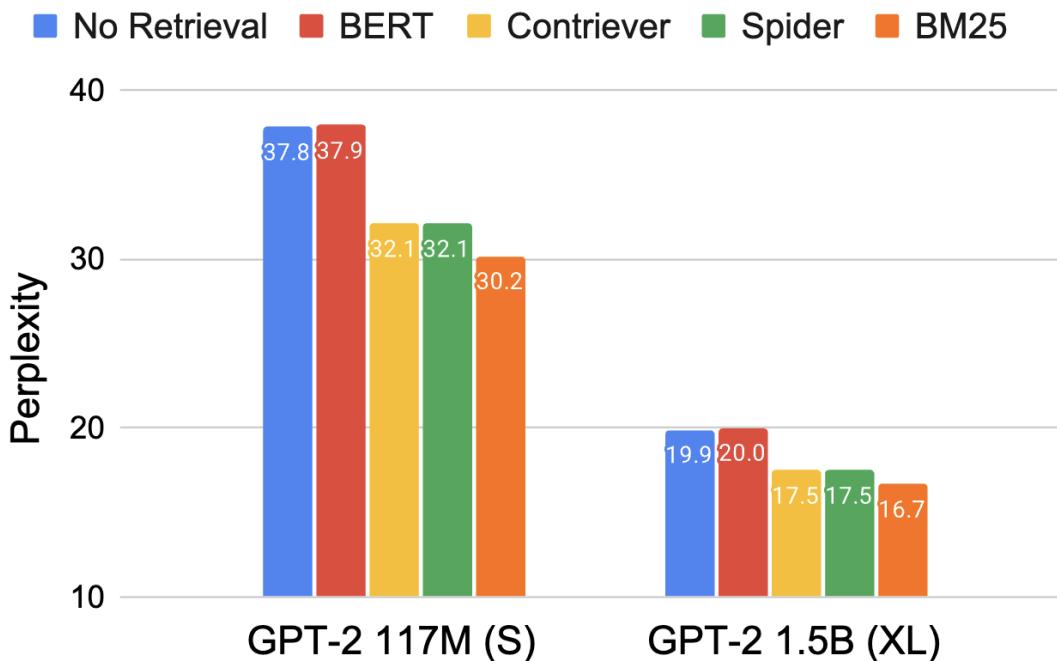


Website of this tutorial

- RA-LLM architecture overview
- Retriever in RA-LLMs
- Retrieval results integration
- Pre/Post-retrieval techniques
- Special RA-LLM paradigms

RA-LLM Architecture: Retriever Types

- Different types of retriever deliver different generation performance



Relevance
measurement

Retriever
learning

Sparse

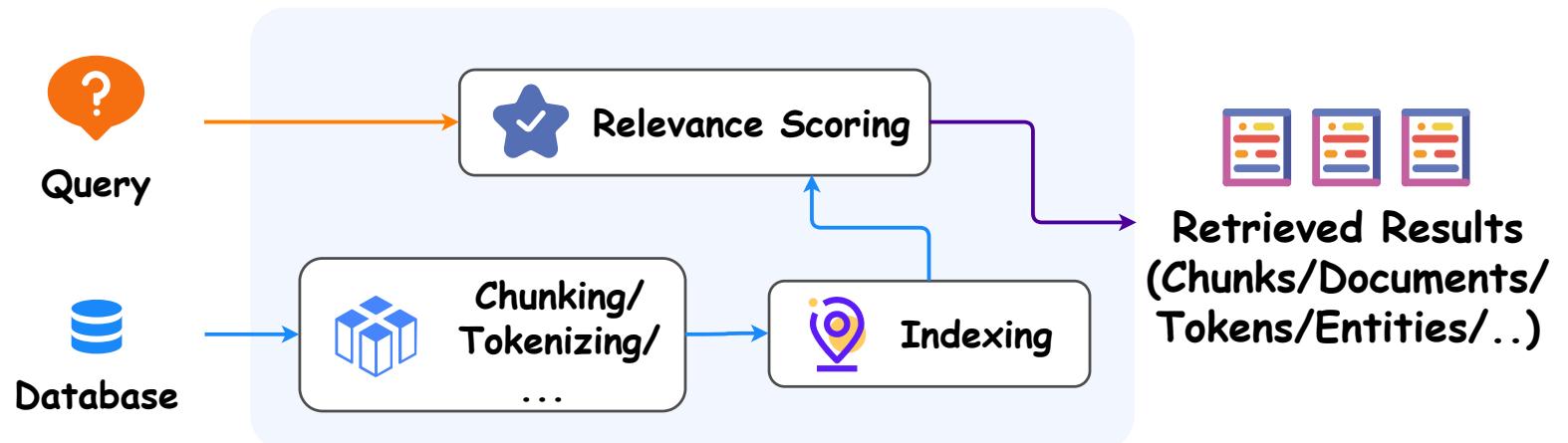
Task-specific
pre-trained

Dense

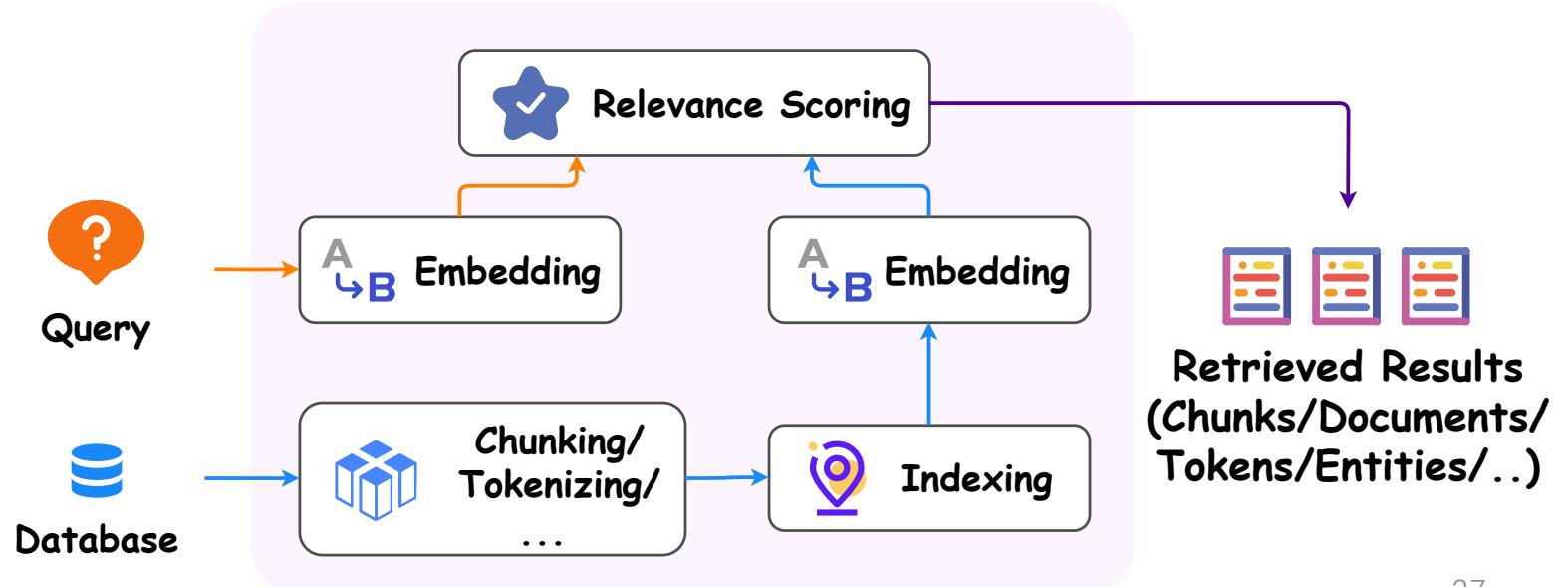
General-purpose
pre-trained

Dense v.s. Sparse Retrievers

Sparse Retrievers (SR)



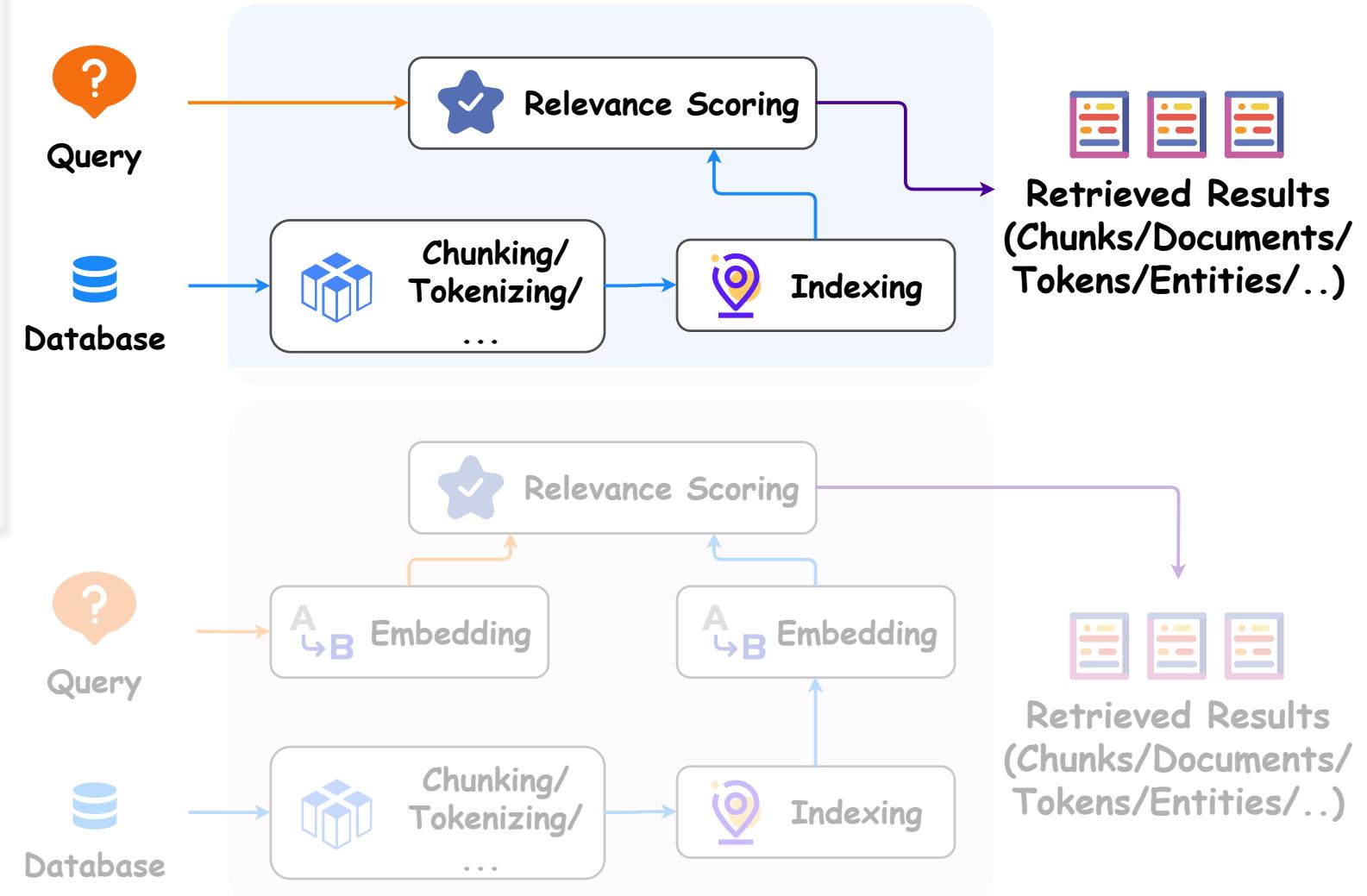
Dense Retrievers (DR)



Dense v.s. Sparse Retrievers

Sparse Retrievers (SR)

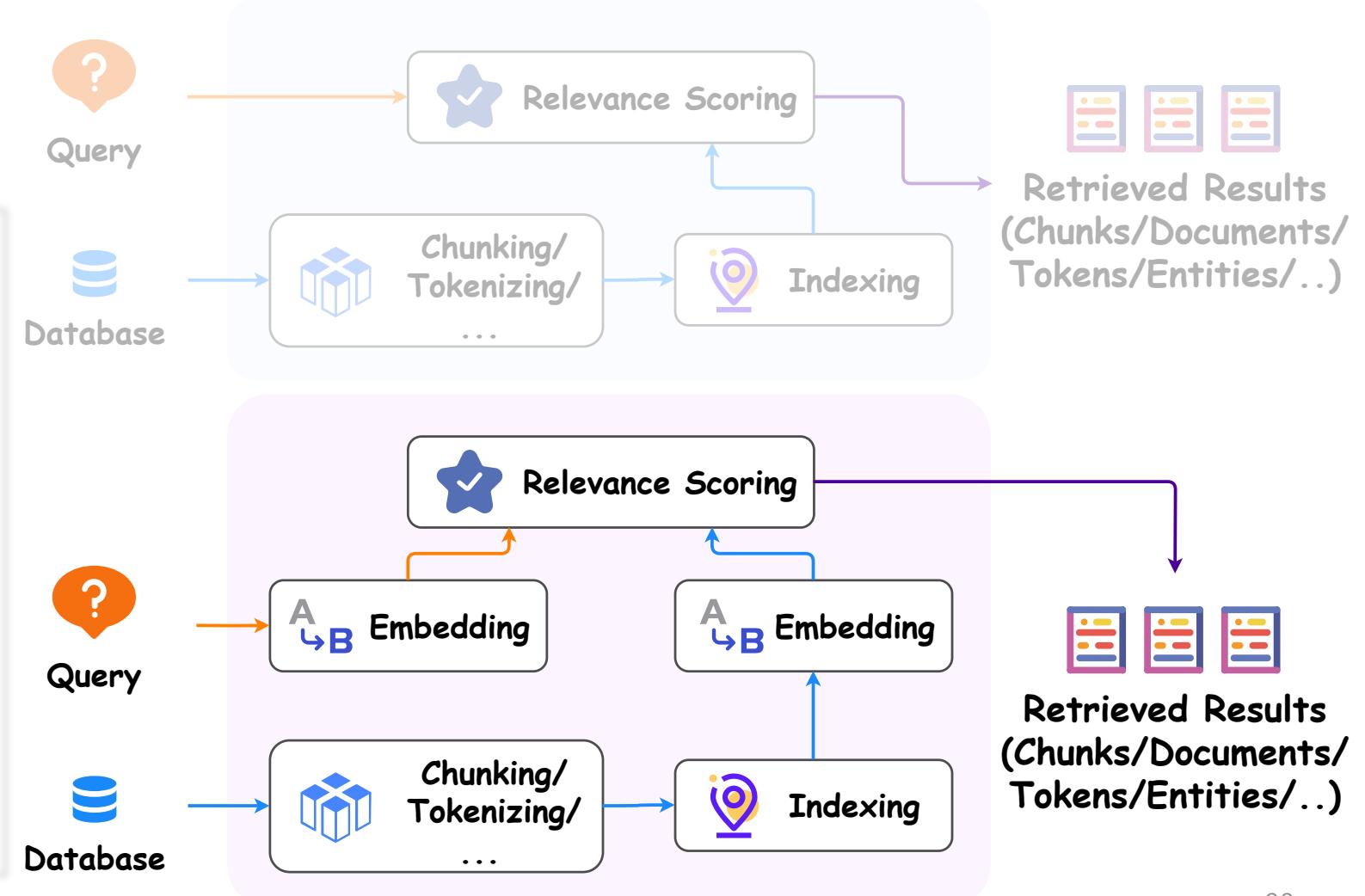
- Feasible to apply
- High efficiency
- Fine performance
- Example: TF-IDF, BM25



Dense v.s. Sparse Retrievers

Dense Retrievers (DR)

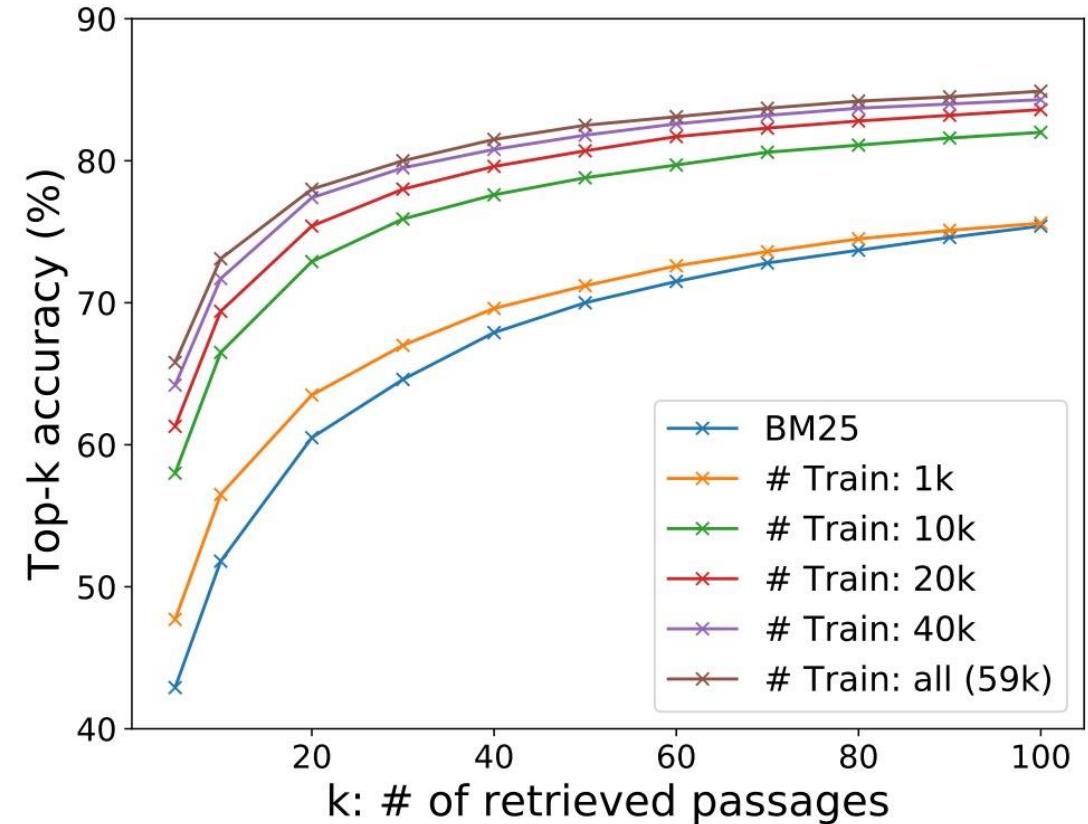
- Allowing fine-tuning
- Better adaptation
- Customizable for more retrieval goals
- Example: DPR, Contriever



Task-Specific Pre-trained Retriever (Supervised)

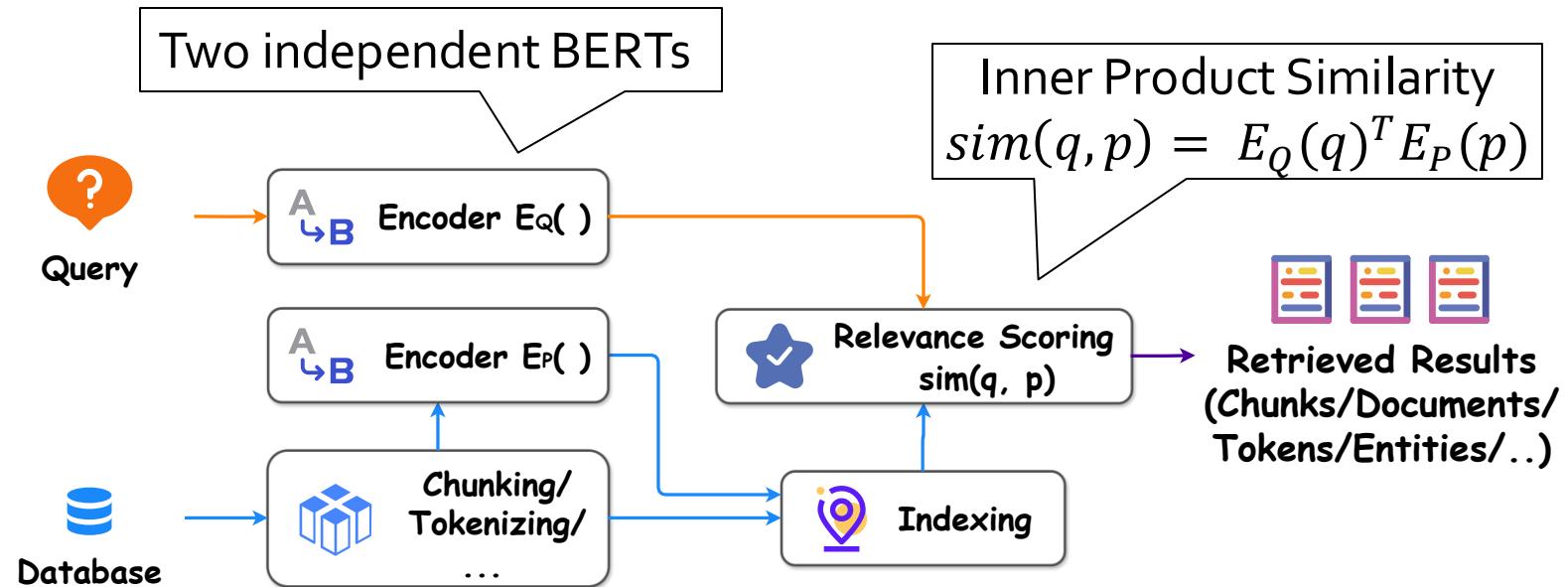
□ Dense Passage Retriever (DPR): Pretrained for Question Answering (QA)

- ❖ It is generally believed that learning a good dense vector representation needs a large number of labeled pairs of question and contexts.
- ❖ Dense retrieval methods have never been shown to outperform TF-IDF/BM25 for open-domain QA before ORQA (Lee et al., 2019), which is unfortunately computationally intensive for pre-training and not specifically trained on Q&A data



Task-Specific Pre-trained Retriever (Supervised)

- **Dense Passage Retriever (DPR):** Pretrained for Question Answering (QA)



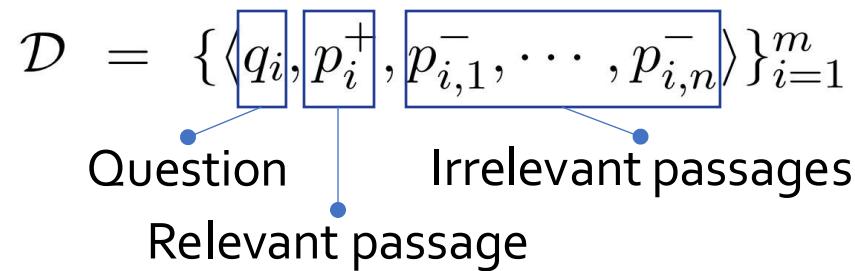
Task-Specific Pre-trained Retriever (Supervised)

□ Dense Passage Retriever (DPR): Pretrained for Question Answering (QA)

- Learning Objective

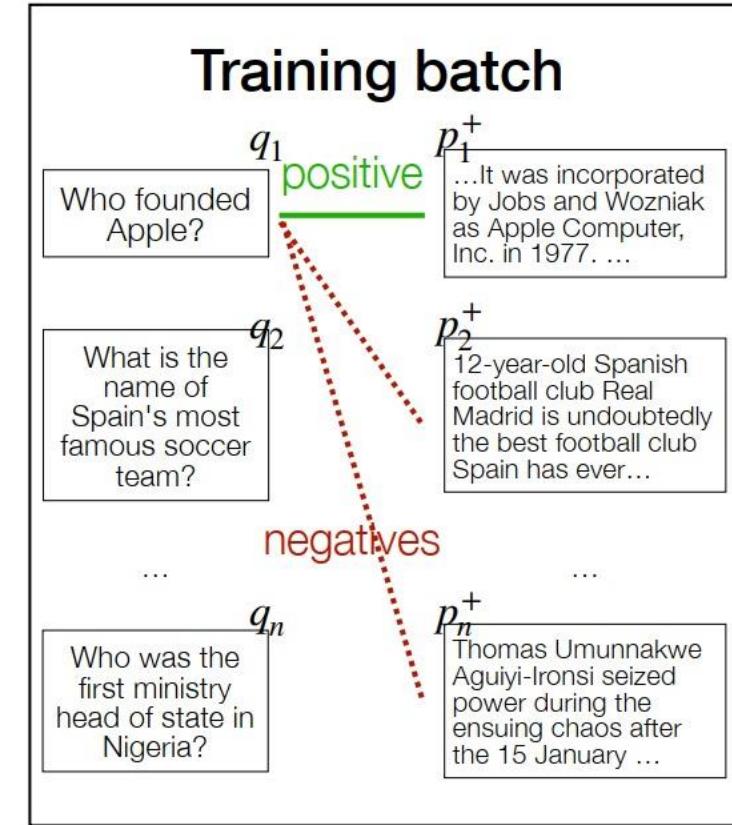
$$L(q_i, p_i^+, p_{i,1}^-, \dots, p_{i,n}^-) = -\log \frac{e^{\text{sim}(q_i, p_i^+)}}{e^{\text{sim}(q_i, p_i^+)} + \sum_{j=1}^n e^{\text{sim}(q_i, p_{i,j}^-)}}$$

- Training data: Question-Passage Sets



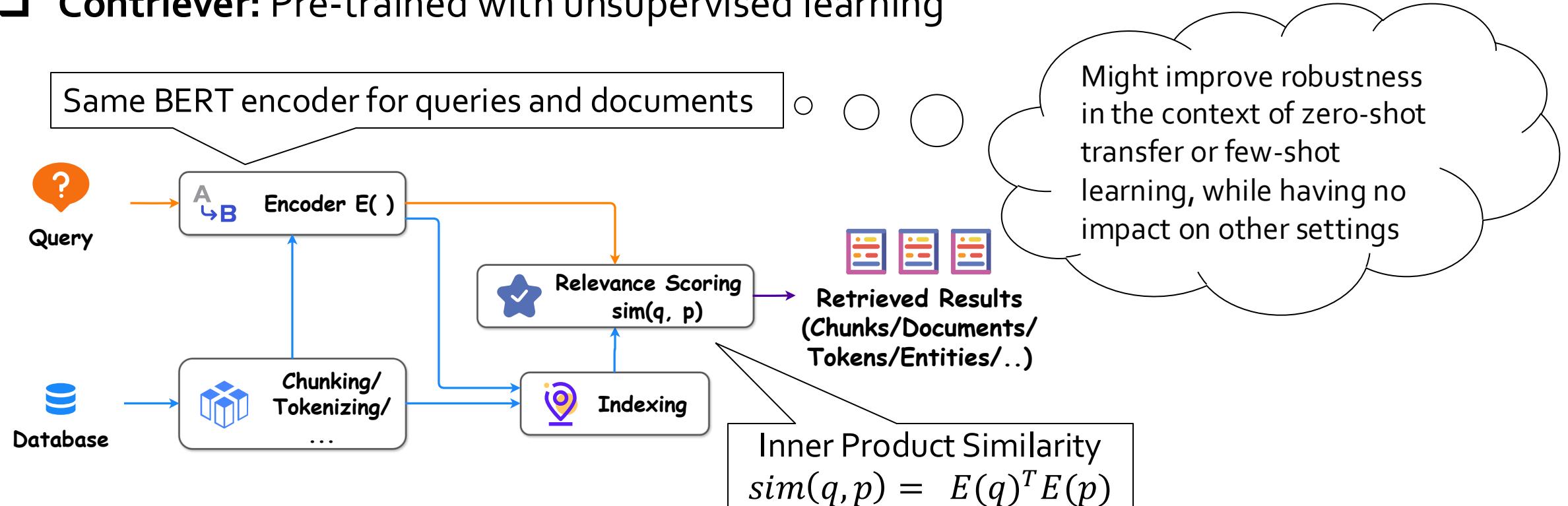
Negative sample selection?

- Training with in-batch negatives



General-Purpose Pre-trained Retriever (Unsupervised)

□ Contriever: Pre-trained with unsupervised learning



Contrastive learning with unaligned documents

$$\mathcal{L}(q, k_+) = - \frac{\exp(s(q, k_+)/\tau)}{\exp(s(q, k_+)/\tau) + \sum_{i=1}^K \exp(s(q, k_i)/\tau)}$$

DPR & Contriever Performance on OpenQA Tasks

End-to-end QA (Exact Match) Accuracy

Training	Model	NQ	TriviaQA	WQ	TREC	SQuAD
Single	BM25+BERT (Lee et al., 2019)	26.5	47.1	17.7	21.3	33.2
Single	ORQA (Lee et al., 2019)	33.3	45.0	36.4	30.1	20.2
Single	HardEM (Min et al., 2019a)	28.1	50.9	-	-	-
Single	GraphRetriever (Min et al., 2019b)	34.5	56.0	36.4	-	-
Single	PathRetriever (Asai et al., 2020)	32.6	-	-	-	56.5
Single	REALMWiki (Guu et al., 2020)	39.2	-	40.2	46.8	-
Single	REALMNews (Guu et al., 2020)	40.4	-	40.7	42.9	-
Single	BM25	32.6	52.4	29.9	24.9	38.1
	DPR	41.5	56.8	34.6	25.9	29.8
	BM25+DPR	39.0	57.0	35.2	28.0	36.7
Multi	DPR	41.5	56.8	42.4	49.4	24.1
	BM25+DPR	38.8	57.9	41.1	50.6	35.8

Both better than
the sparse retriever!

Both widely applied in
RAG and RA-LLMs



	NaturalQuestions			TriviaQA		
	R@5	R@20	R@100	R@5	R@20	R@100
Inverse Cloze Task (Sachan et al., 2021)	32.3	50.9	66.8	40.2	57.5	73.6
Masked salient spans (Sachan et al., 2021)	41.7	59.8	74.9	53.3	68.2	79.4
BM25 (Ma et al., 2021)	-	62.9	78.3	-	76.4	83.2
Contriever	47.8	67.8	82.1	59.4	74.2	83.2

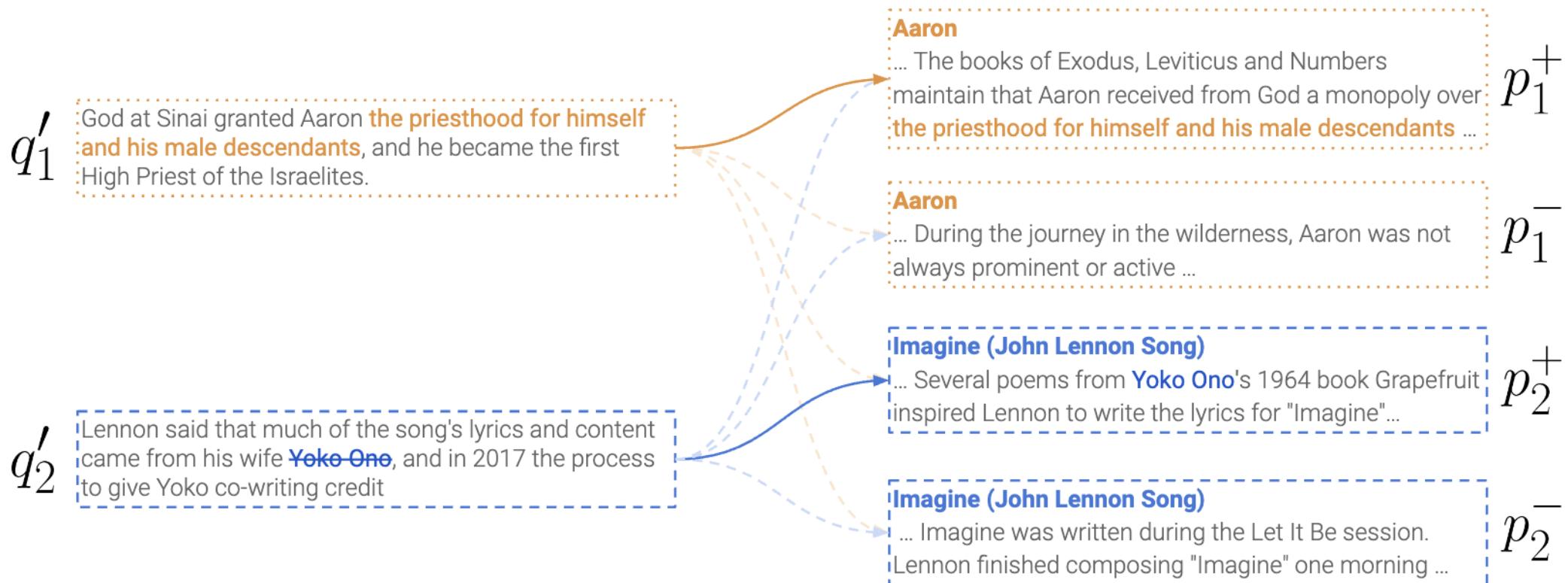
supervised model: DPR (Karpukhin et al., 2020)

-	78.4	85.4	-	79.4	85.0
---	------	------	---	------	------

Task-Specific Pre-trained Retriever (Unsupervised)

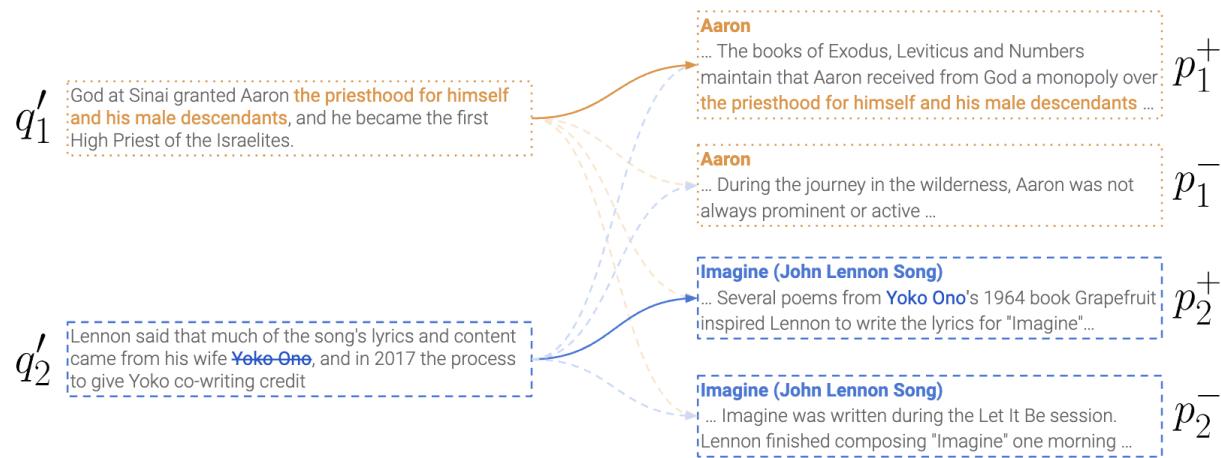
□ Spider (Span-based unsupervised dense retriever)

- ❖ **Recurring Span Retrieval:** It is based on the notion of recurring spans within a document: given two paragraphs with the same recurring span, we construct a query from one of the paragraphs, while the other is taken as the target for retrieval

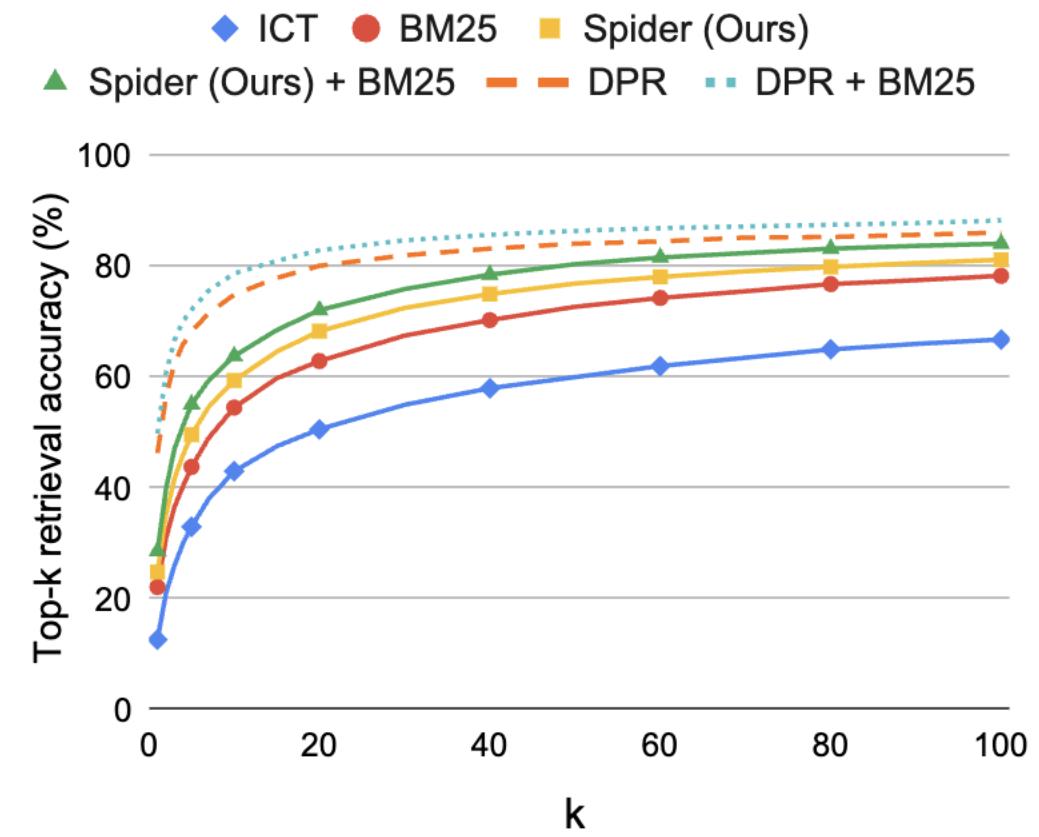


Task-Specific Pre-trained Retriever (Unsupervised)

□ Learning and results of Spider

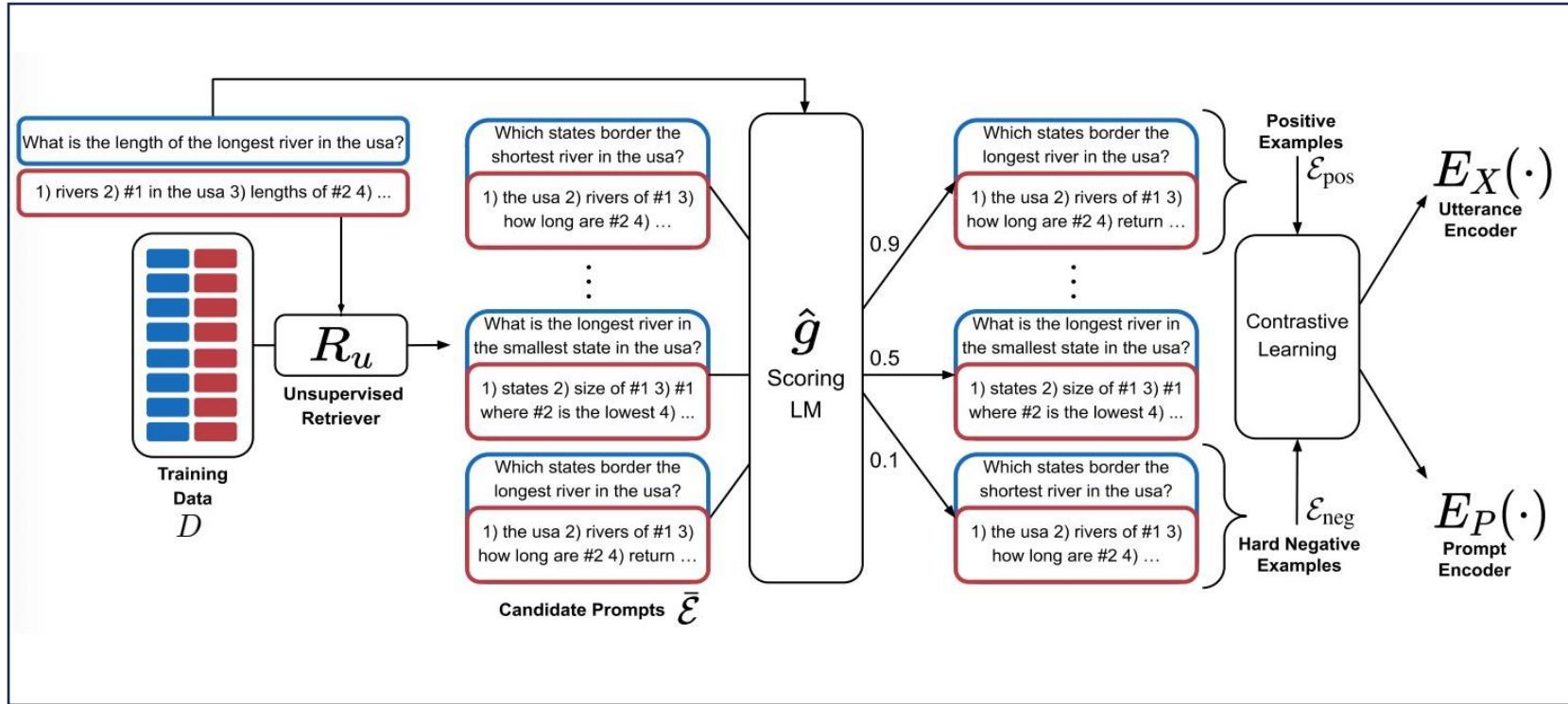


$$-\log \frac{\exp(s(q'_i, p_i^+))}{\sum_{j=1}^m (\exp(s(q'_i, p_j^+)) + \exp(s(q'_i, p_j^-)))}$$

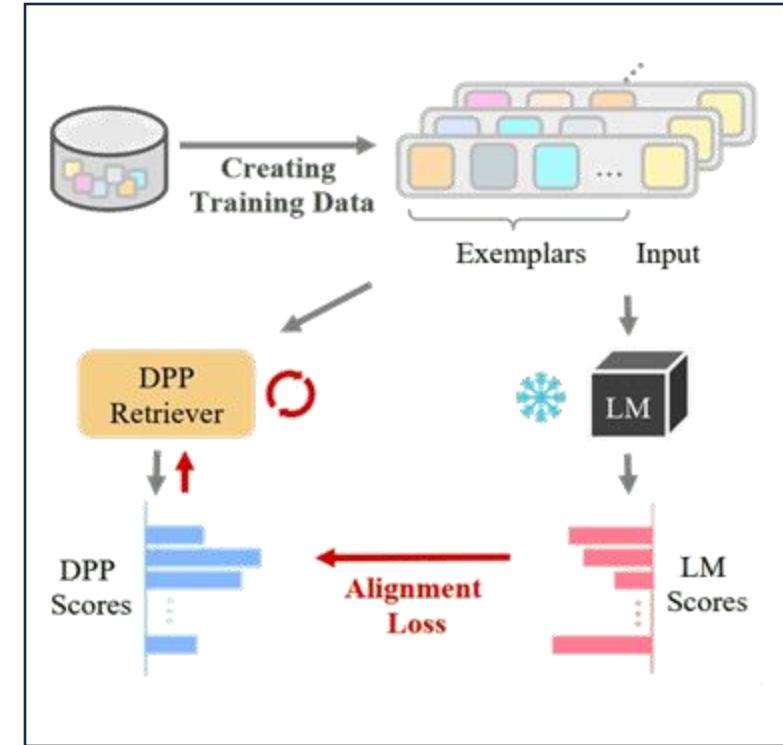


Retrievers for In-Context Learning of LLMs

□ Prompt Retriever



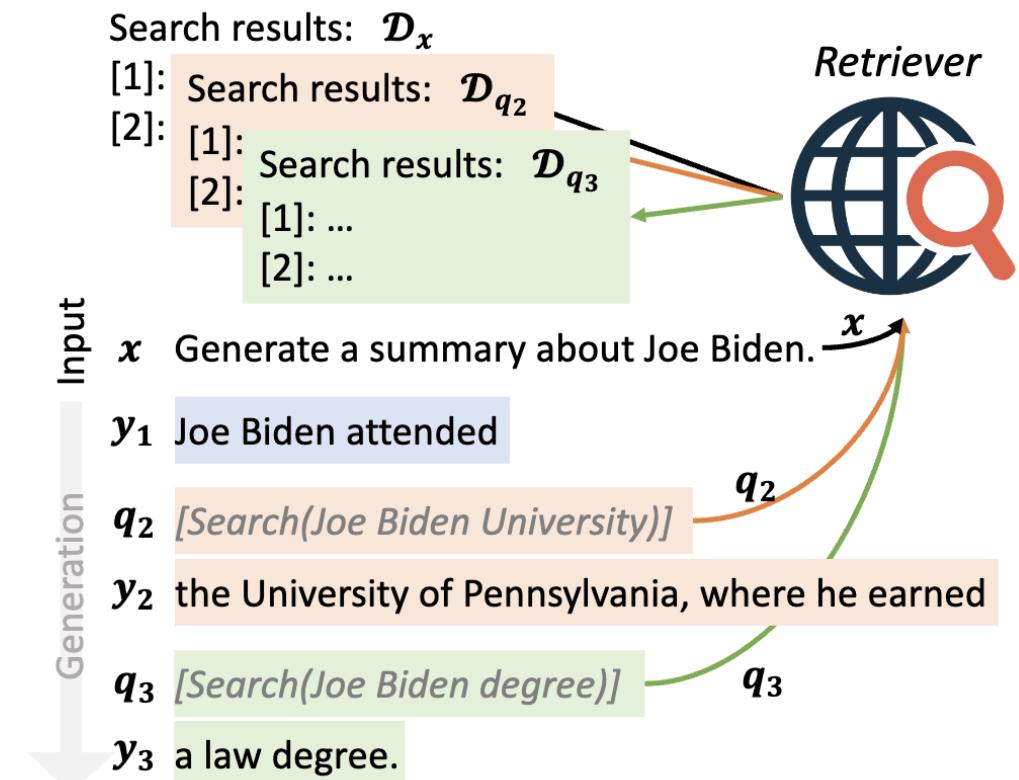
□ Examplar Retriever (CEIL)



Search Engine as Retrievers

❑ Traditional retrieval methods

- ❖ May be difficult to update to real-time web documents
- ❖ May be a limit to the number of documents storable in the pre-defined database
- ❖ Will not take advantage of the high quality ranking that has been finely tuned in Internet Search engines over decades of use



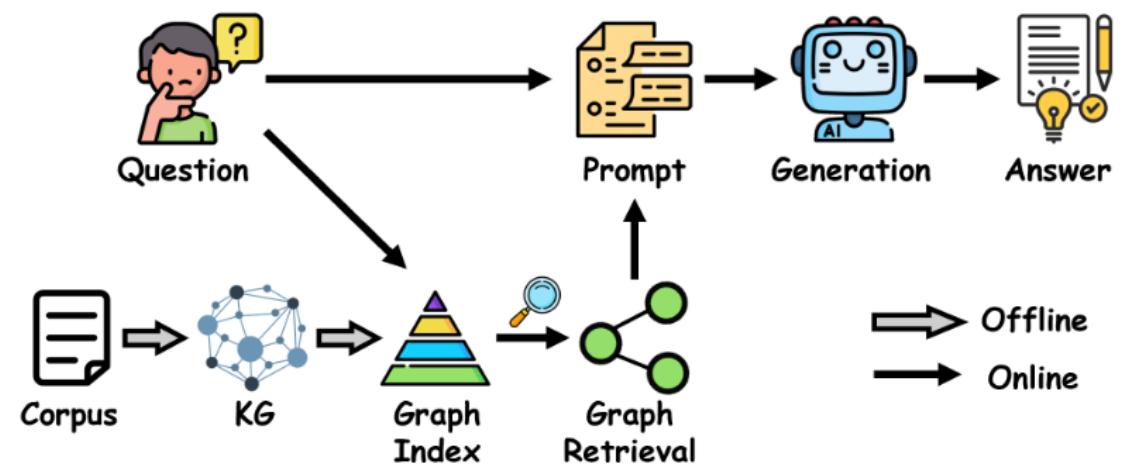
Knowledge Graph (KG) for Retrieval

❑ Text-based retrieval

- ☒ Measuring only **semantic similarity** between two texts
- ☒ Overlooking knowledge-level association
- ☒ Cannot capture structured relationship

e.g., *Global Financial Crisis and The 2008 Recession*

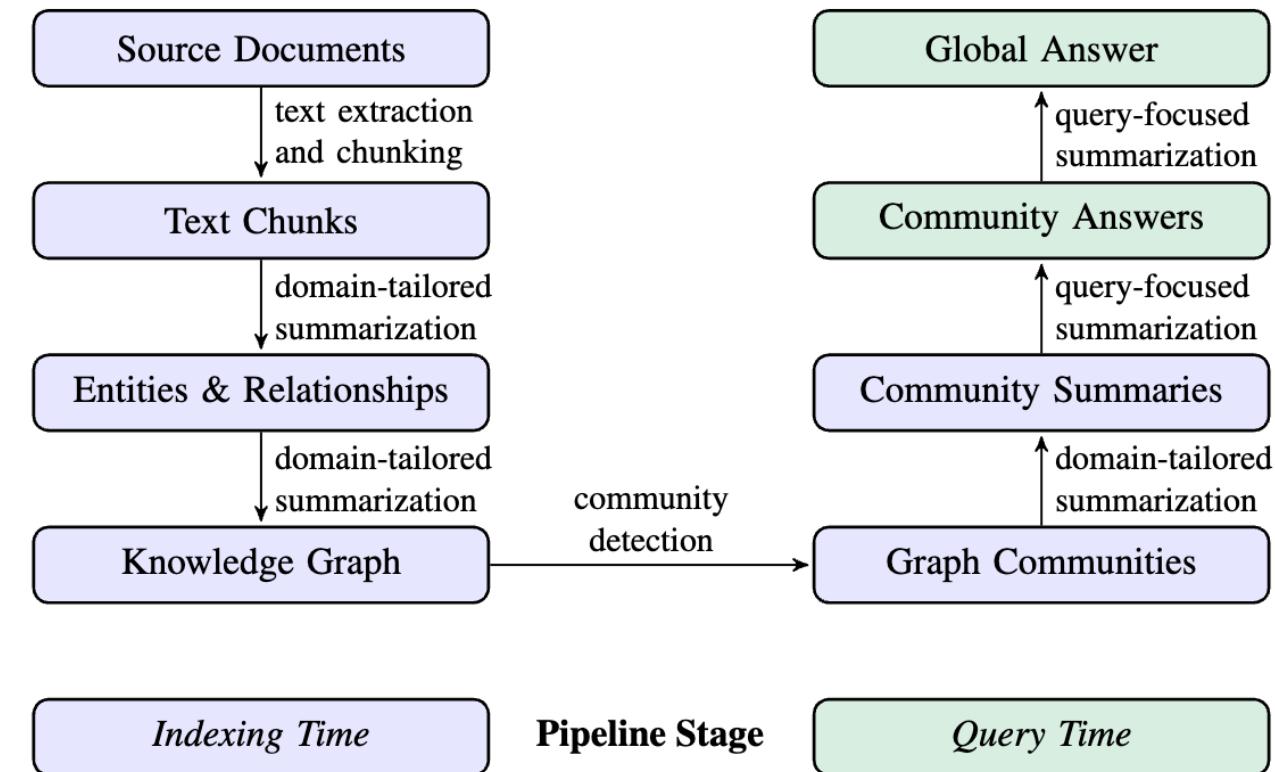
*Harry Potter and Fantastic Beasts which are both written
by J.K. Rowling*



Knowledge Graph (KG) for Retrieval

- **GraphRAG:** aggregates nodes into communities, and produces a community report to capture global information

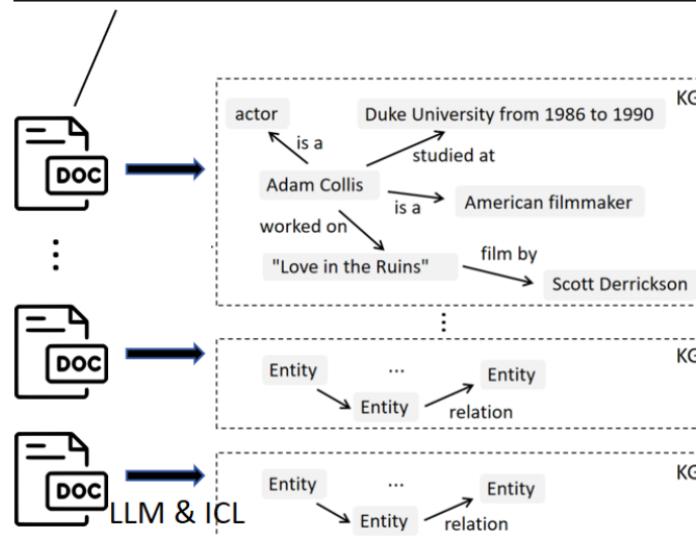
- ☒ RAG fails on global questions directed at an entire text corpus
 - e.g., *What are the main themes in the dataset?*



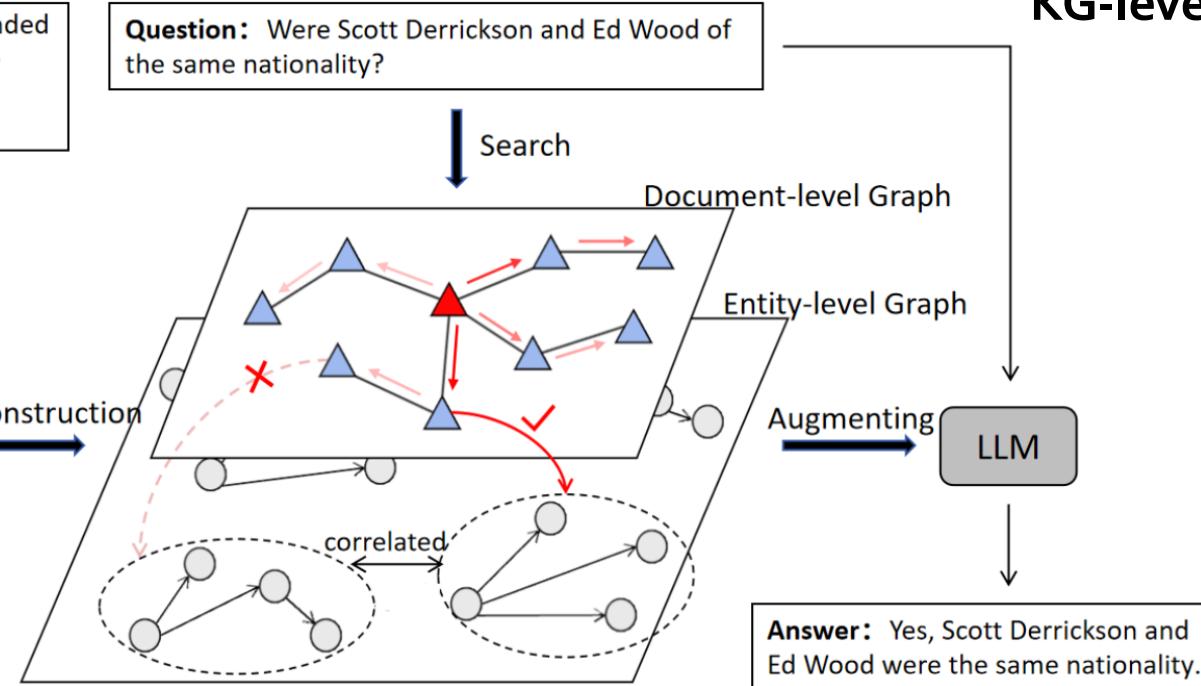
Knowledge Graph (KG) for Retrieval

❑ KG-Retriever with Hierarchical Index Graph (HIG)

Text: Adam Collis is an American filmmaker and actor. He attended the Duke University from 1986 to 1990. Collis first work was the assistant director for the Scott Derrickson's short "Love in the Ruins" (1995).



Question: Were Scott Derrickson and Ed Wood of the same nationality?



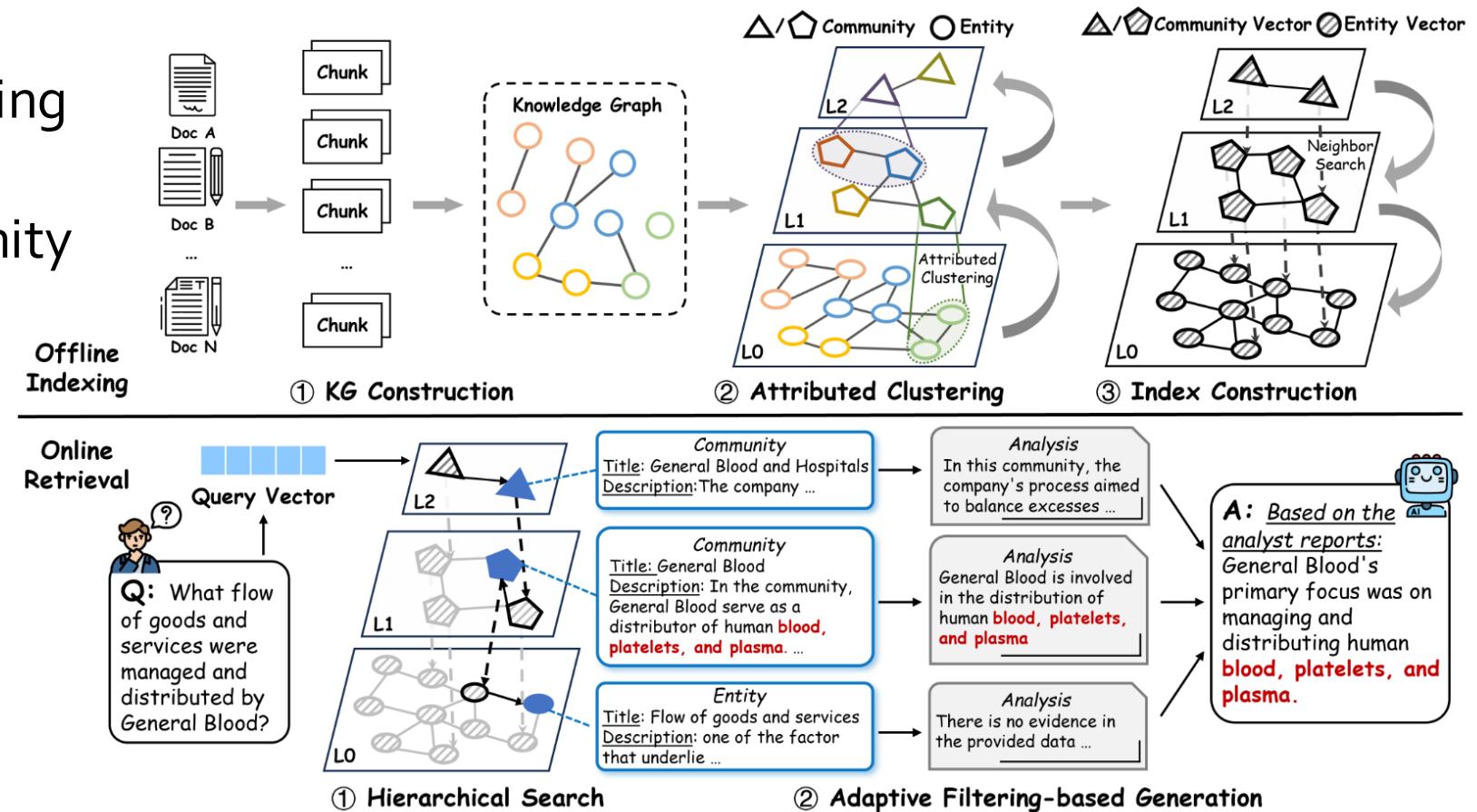
Document-level Retrieval
+
KG-level Retrieval

Index Construction ————— Retrieval strategies ————— Response Generation

Knowledge Graph (KG) for Retrieval

- ❑ ArchRAG (Attributed Community-based Hierarchical RAG): augmenting the question using attributed communities

- ❑ Hierarchical indexing
- ❑ Attribute community
- ❑ Efficient HNSW



PART 2: Architecture of RA-LLMs and Main Modules



Slides



Website of this tutorial

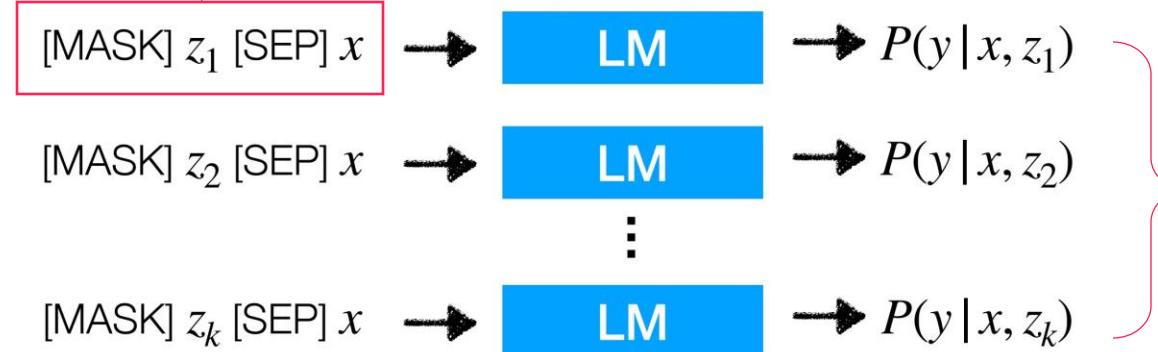
- RA-LLM architecture overview
- Retriever in RA-LLMs
- Retrieval results integration
- Pre/Post-retrieval techniques
- Special RA-LLM paradigms

Retrieved Results Integration: Input-layer Integration

□ REALM



Integrating the
retrieved passage
 z and x the
original input



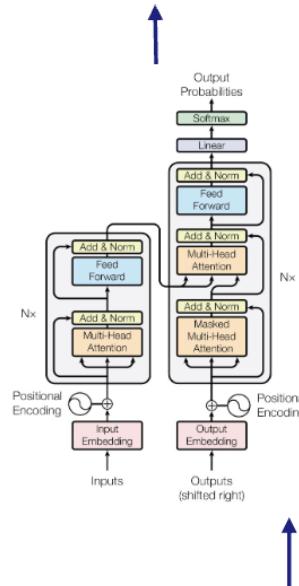
Weighted aggregating
the prediction results
based on all retrieved
passages

$$\sum_{z \in \mathcal{D}} P(z | x) P(y | x, z)$$

Retrieval-Augmented Generator

Typical encoder: $p(y|x)$

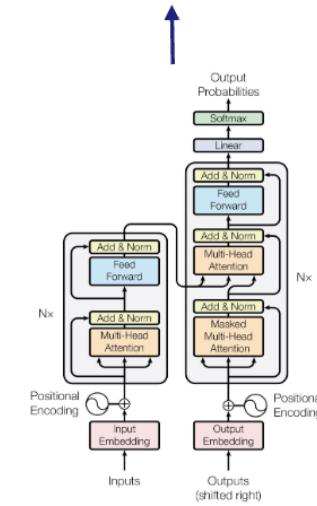
$y = \text{pounds}$



$x: \text{we paid } 20 _ \text{ at the Buckingham Palace gift shop}$

Knowledge-augmented encoder: $p(y|x, z)$

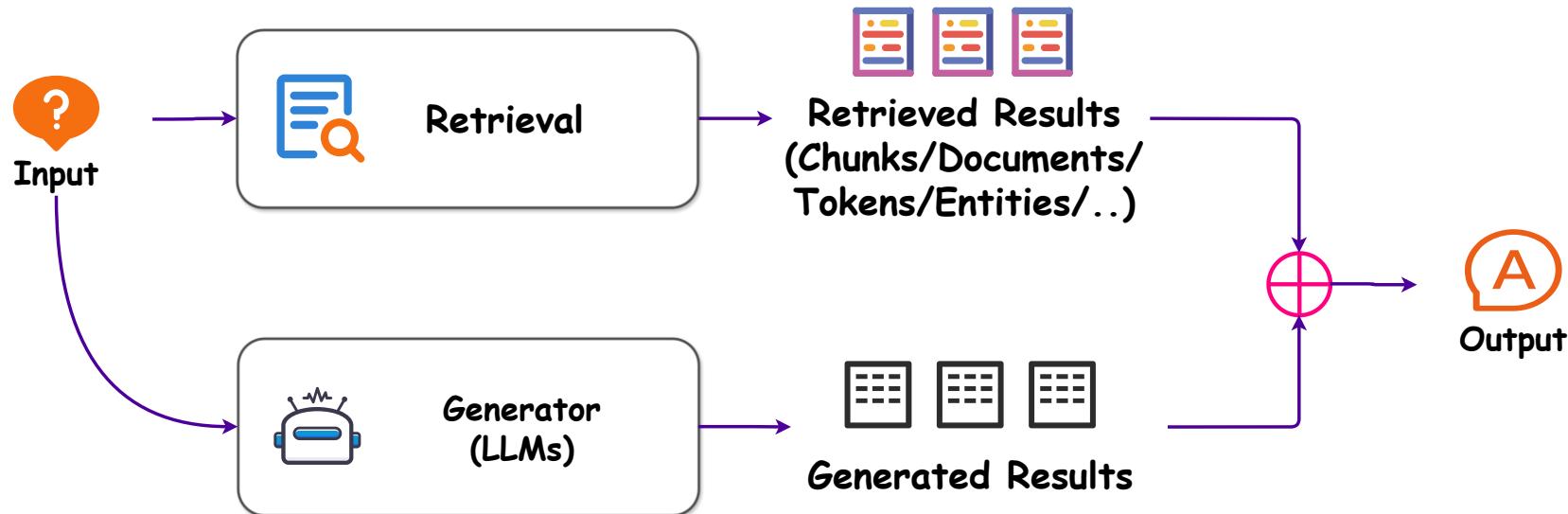
$y = \text{pounds}$



$x: \text{we paid } 20 _ \text{ at the Buckingham Palace gift shop}$

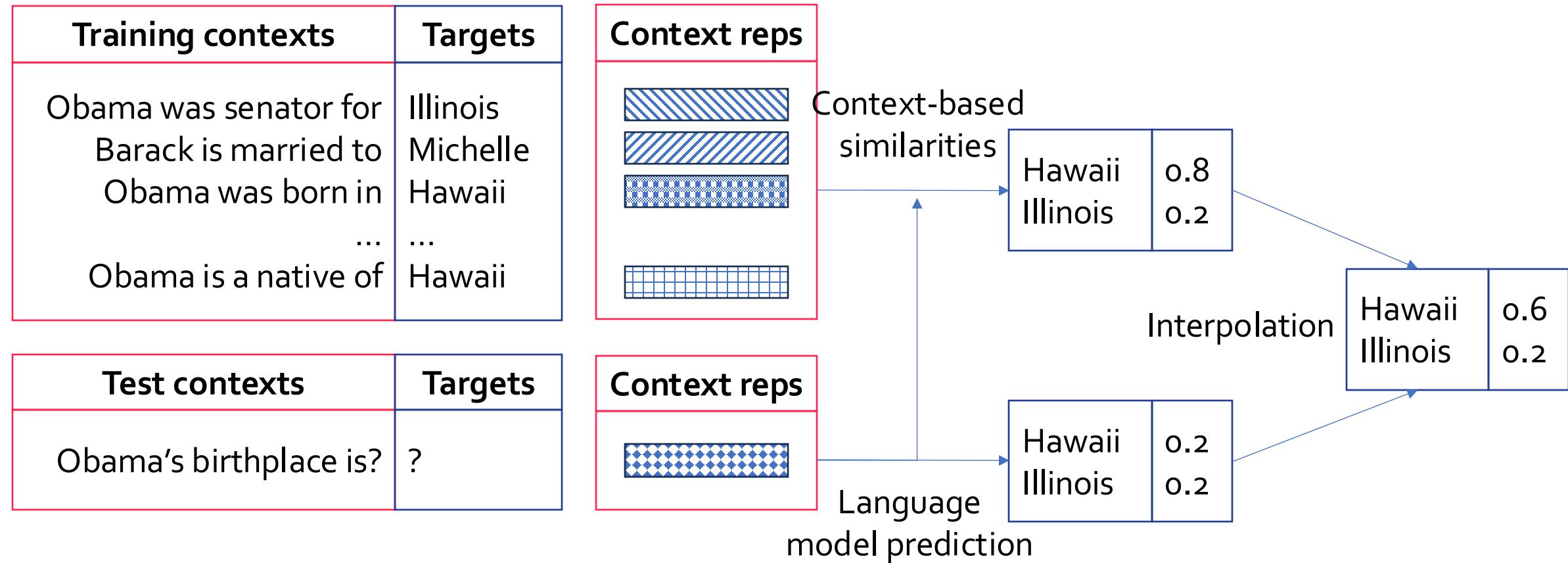
$z: \text{Buckingham Palace is home to the British monarchy}$
explicit knowledge

Retrieved Results Integration: Output-layer integration

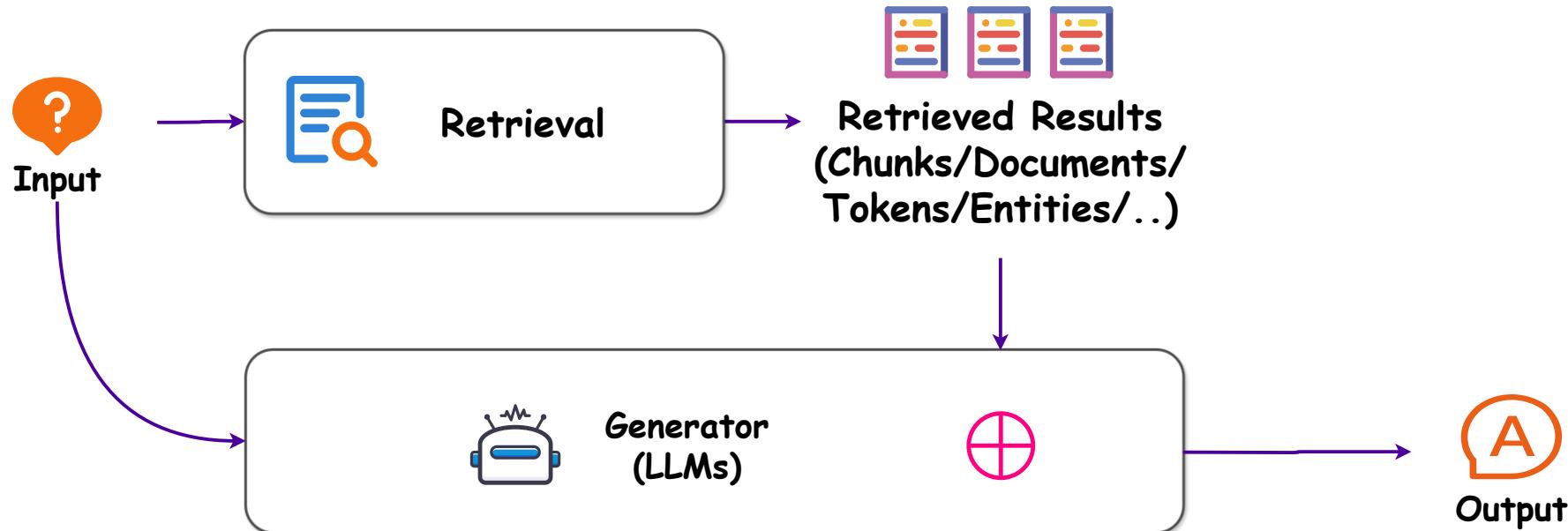


RA-LLM Architecture: Output-layer Integration

- ❑ **kNN-LM:** Combining retrieved probabilities and predicted ones in generation

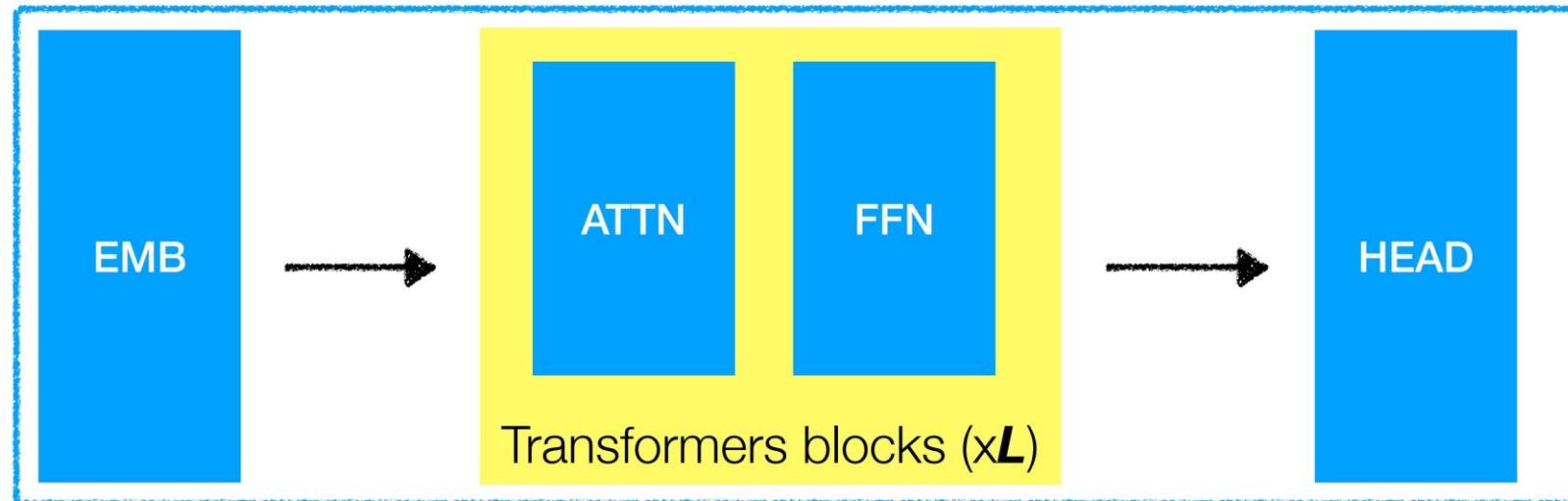


Retrieved Results Integration: Intermediate-layer Integration



Retrieved Results Integration: Intermediate-layer Integration

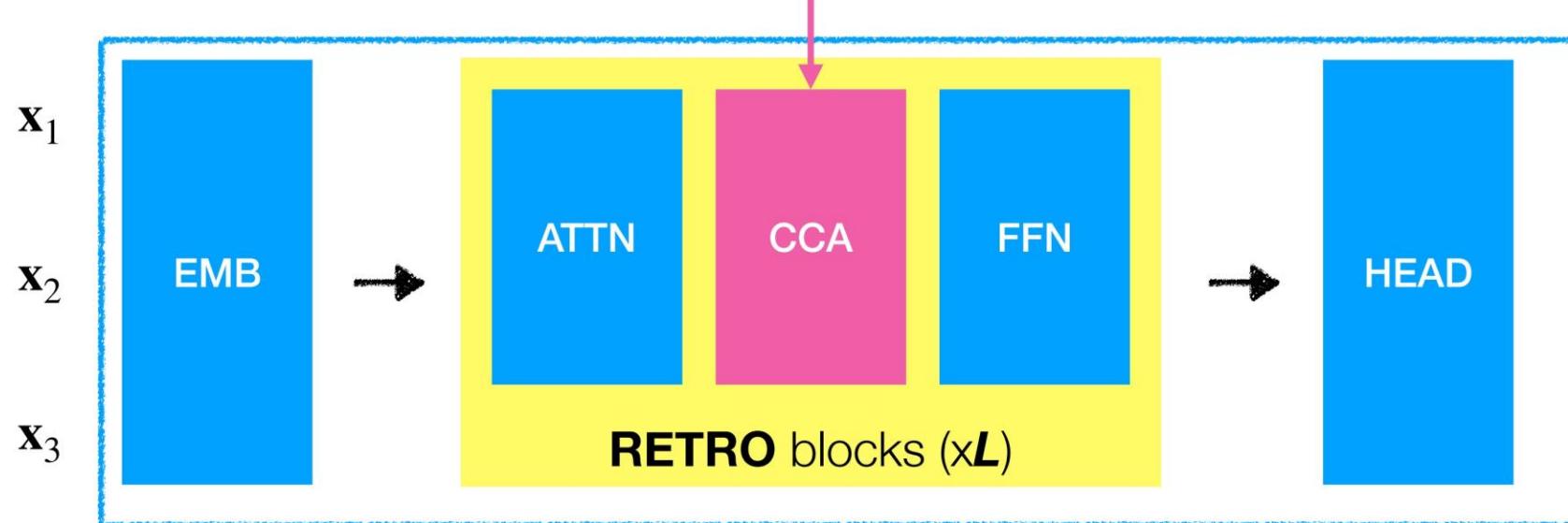
Regular Decoder



Retrieved Results Integration: Intermediate-layer Integration

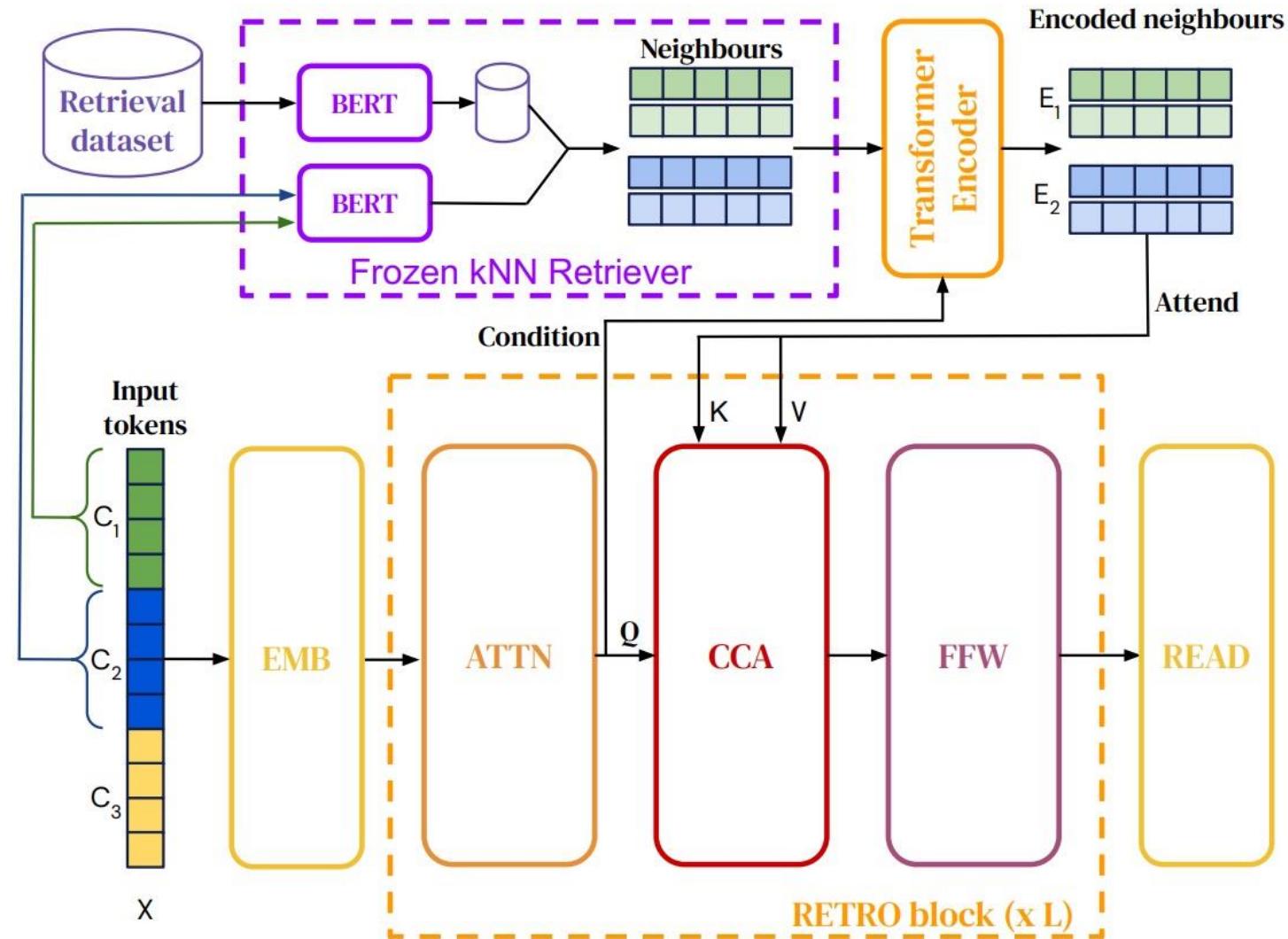
Decoder to incorporate retrieved results
(RETRO)

With retrieved results $\rightarrow \mathbf{E}_1 \ \mathbf{E}_2 \ \mathbf{E}_3$

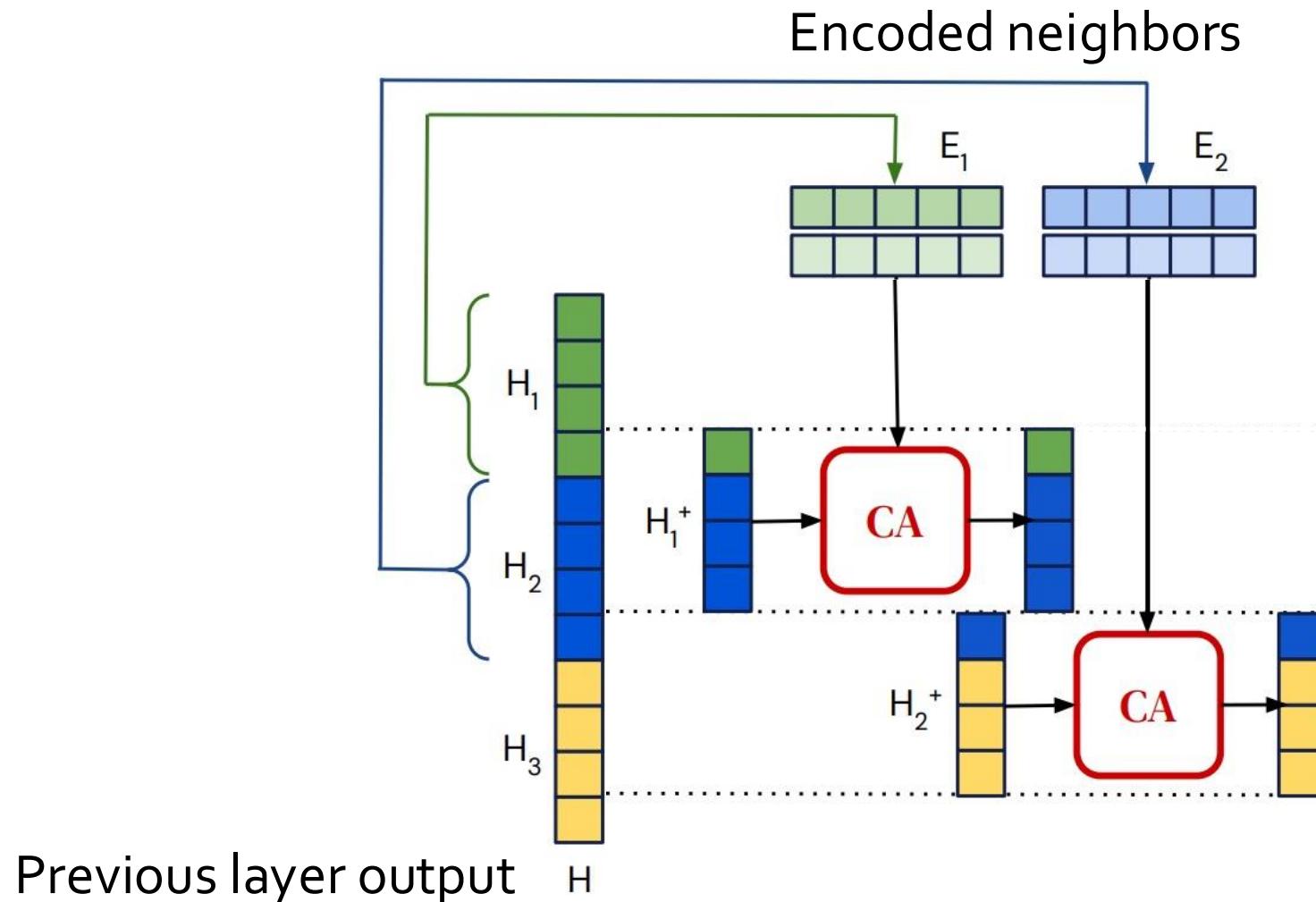


Chunked Cross Attention (CCA)

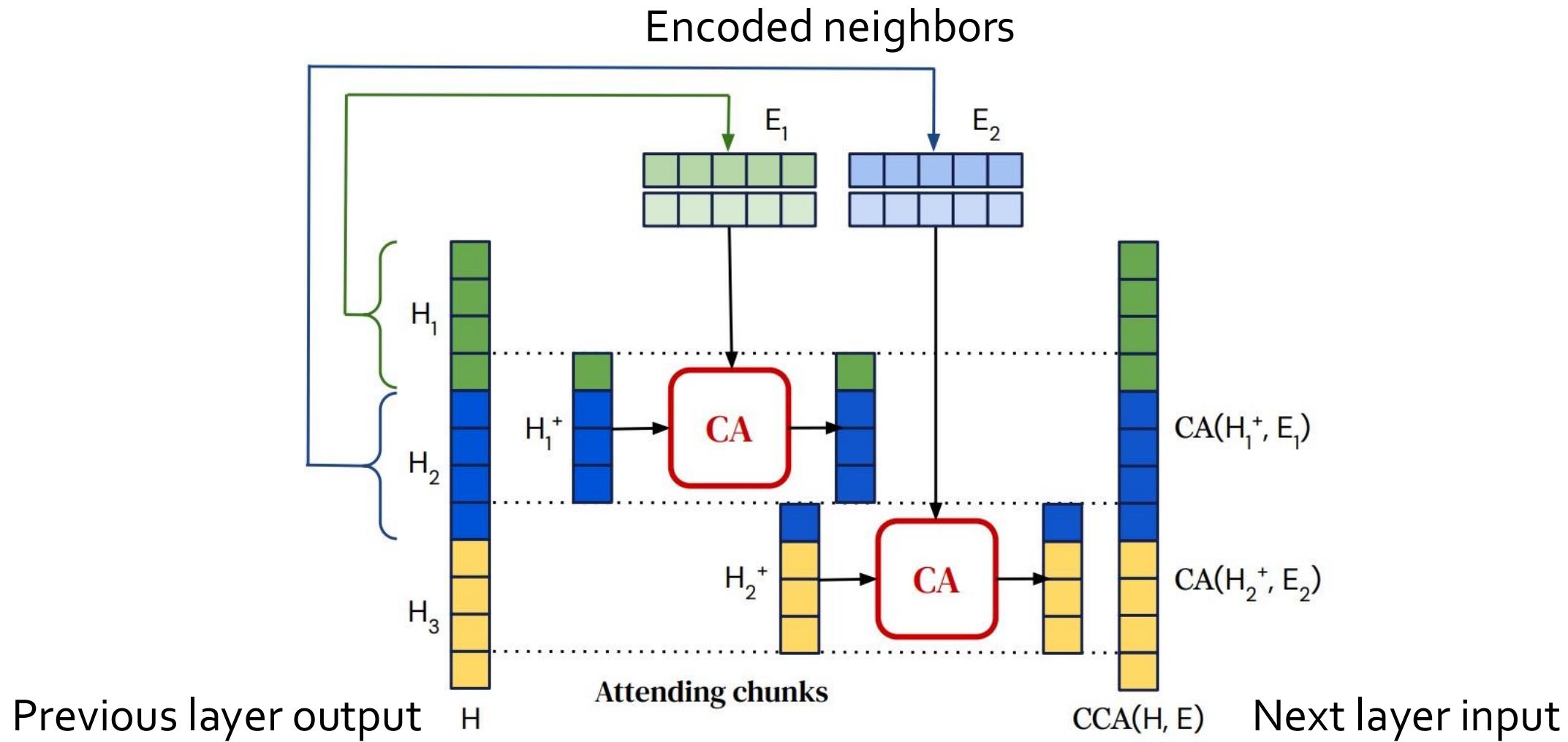
Retrieved Results Integration: Intermediate-layer Integration



Retrieved Results Integration: Intermediate-layer Integration



Retrieved Results Integration: Intermediate-layer Integration



PART 2: Architecture of RA-LLMs and Main Modules



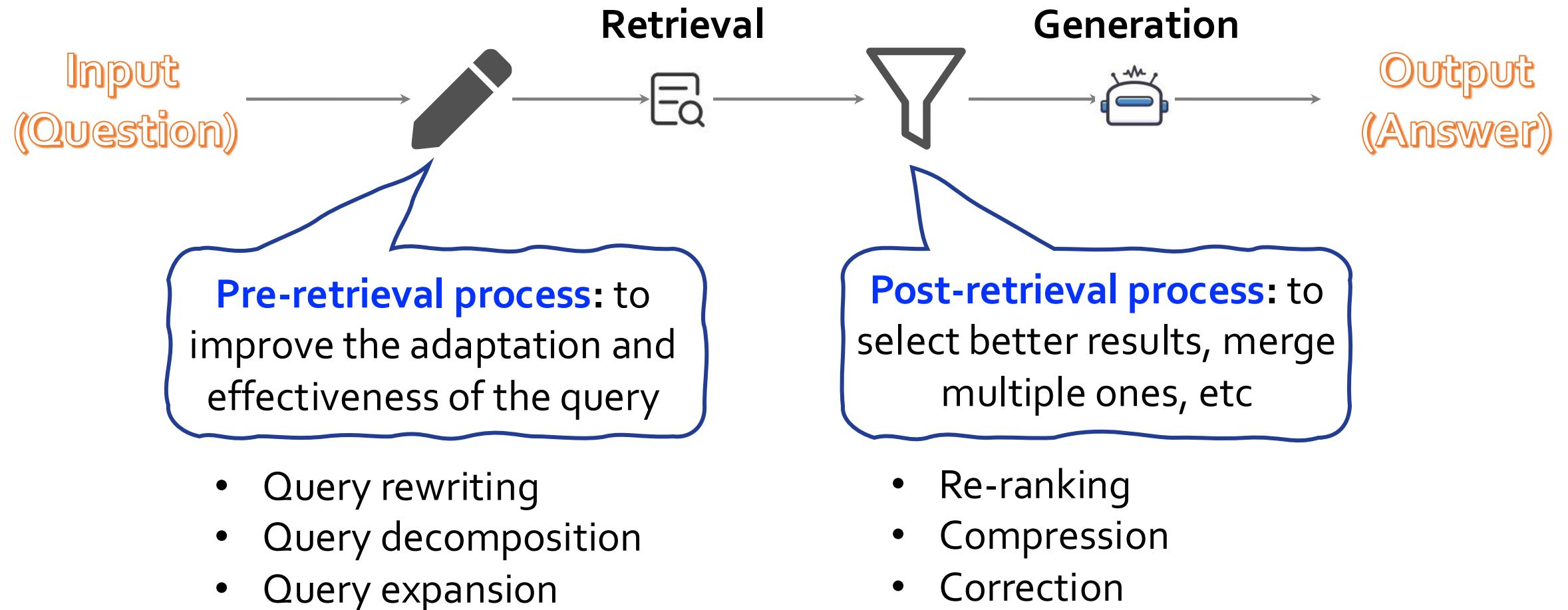
Slides



Website of this tutorial

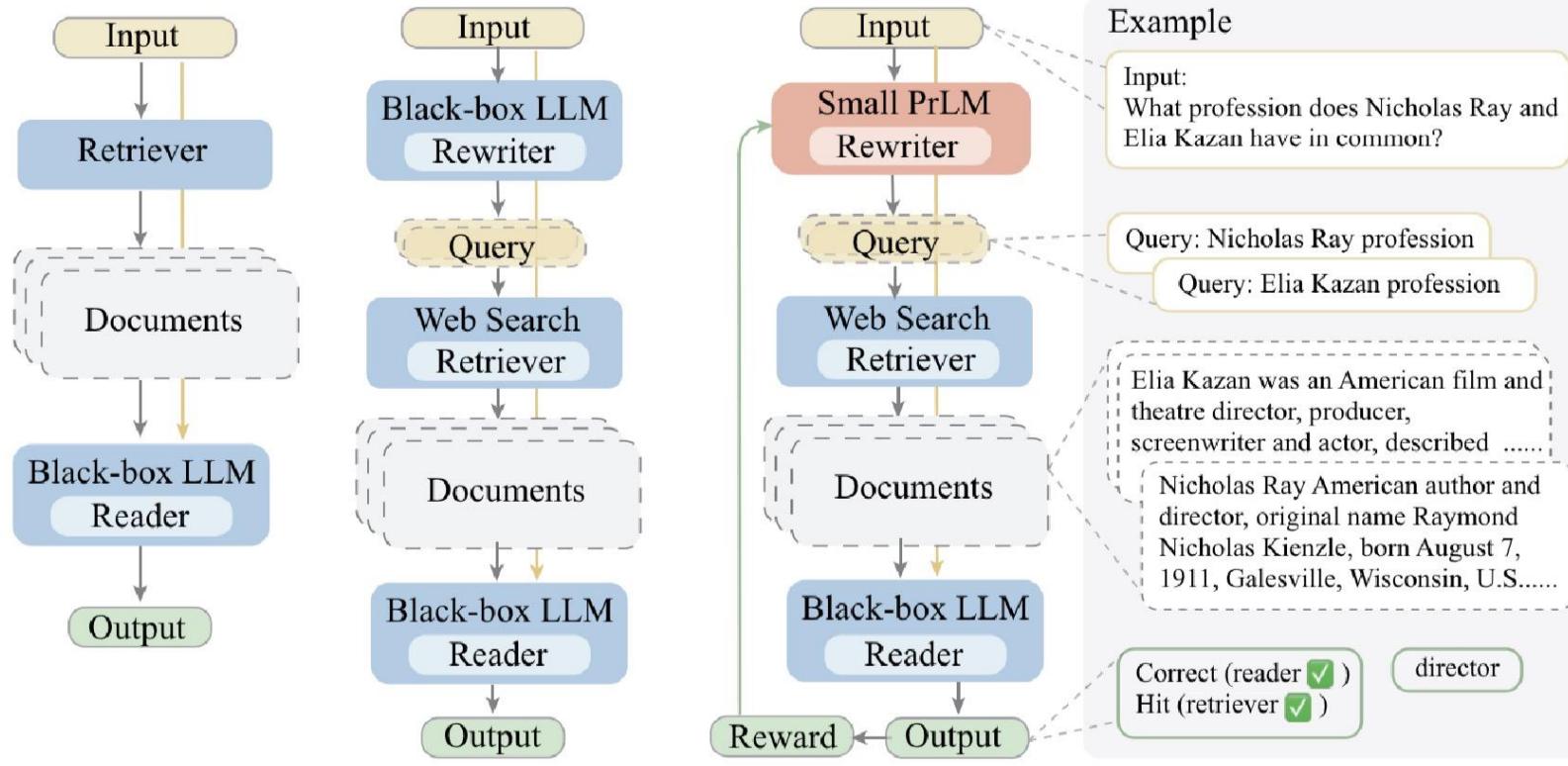
- RA-LLM architecture overview
- Retriever in RA-LLMs
- Retrieval results integration
- Pre/Post-retrieval techniques
- Special RA-LLM paradigms

Pre/Post-Retrieval Techniques



Pre-Retrieval Techniques

□ Query Rewriting: to improve the adaptation of the query

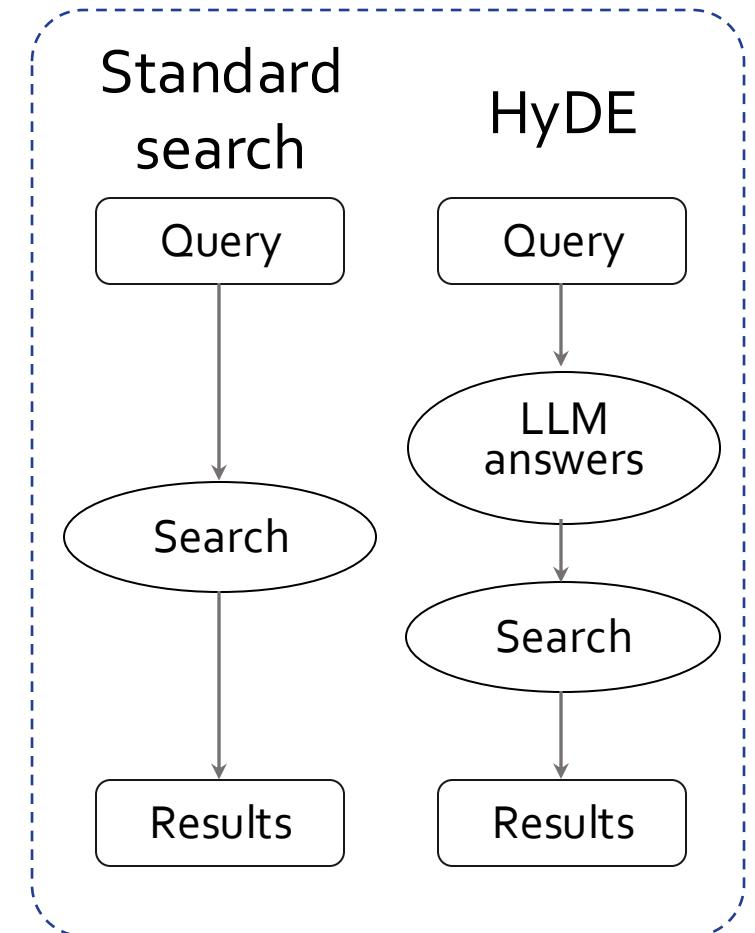
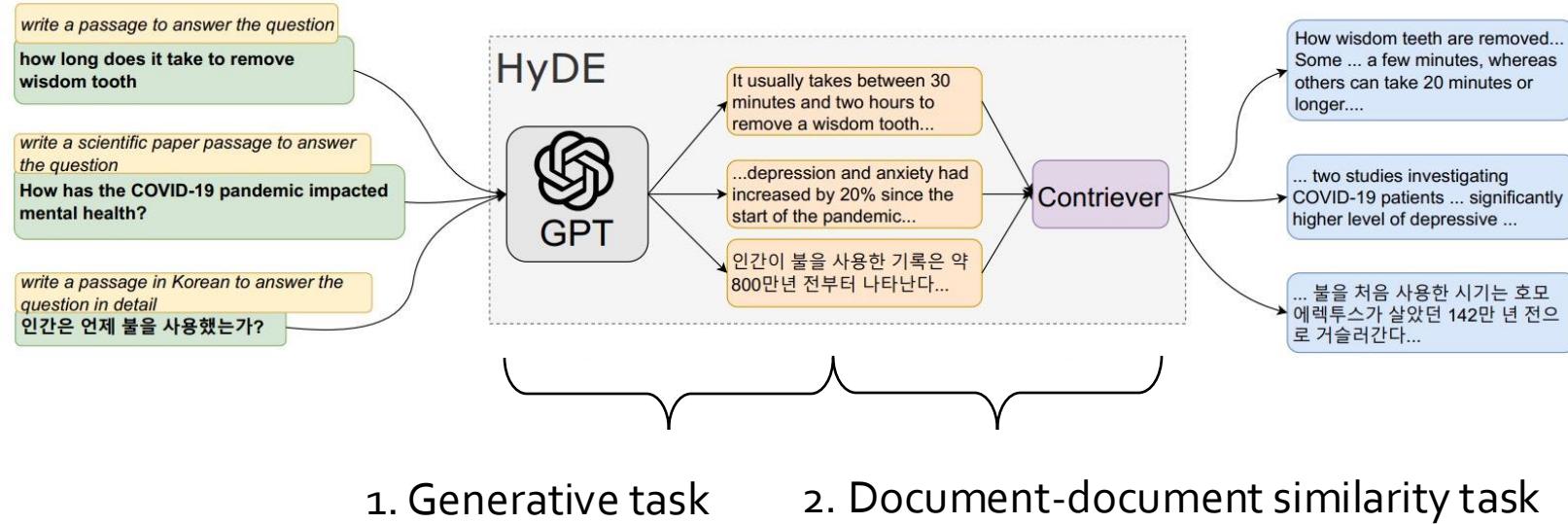


Model	EM	F ₁
<i>HotpotQA</i>		
Direct	32.36	43.05
Retrieve-then-read	30.47	41.34
LLM rewriter	32.80	43.85
Trainable rewriter	34.38	45.97
<i>AmbigNQ</i>		
Direct	42.10	53.05
Retrieve-then-read	45.80	58.50
LLM rewriter	46.40	58.74
Trainable rewriter	47.80	60.71
<i>PopQA</i>		
Direct	41.94	44.61
Retrieve-then-read	43.20	47.53
LLM rewriter	46.00	49.74
Trainable rewriter	45.72	49.51

Works on different QA settings

Pre-Retrieval Techniques

□ HyDE: Hypothetical Document Embeddings



Pre-Retrieval Techniques

□ Query Expansion

LLM Prompts

Write a passage that answers the given query:

Query: what state is this zip code 85282

Passage: Welcome to TEMPE, AZ 85282.
85282 is a rural zip code in Tempe, Arizona.
The population is primarily white...

...

Query: when was pokemon green released

Passage:

New query = original query + generated documents

$$q^+ = \text{concat}(q, [\text{SEP}], d')$$

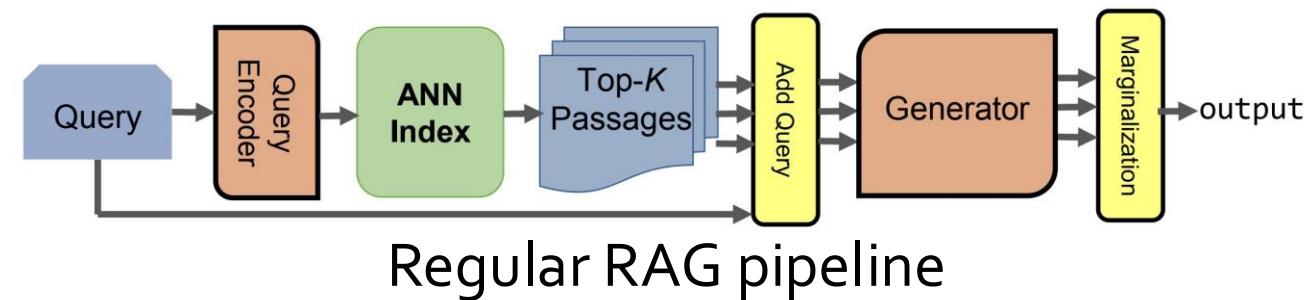
Method	Fine-tuning	MS MARCO dev			TREC DL 19	
		MRR@10	R@50	R@1k	nDCG@10	
Sparse retrieval						
BM25	✗	18.4	58.5	85.7	51.2*	
+ query2doc	✗	21.4 ^{+3.0}	65.3 ^{+6.8}	91.8 ^{+6.1}	66.2^{+15.0}	
BM25 + RM3	✗	15.8	56.7	86.4	52.2	
docT5query (Nogueira and Lin)	✓	27.7	75.6	94.7	64.2	
Dense retrieval w/o distillation						
ANCE (Xiong et al., 2021)	✓	33.0	-	95.9	64.5	
HyDE (Gao et al., 2022)	✗	-	-	-	61.3	
DPR _{bert-base} (our impl.)	✓	33.7	80.5	95.9	64.7	
+ query2doc	✓	35.1^{+1.4}	82.6^{+2.1}	97.2^{+1.3}	68.7^{+4.0}	

Works for both sparse and dense retrievers

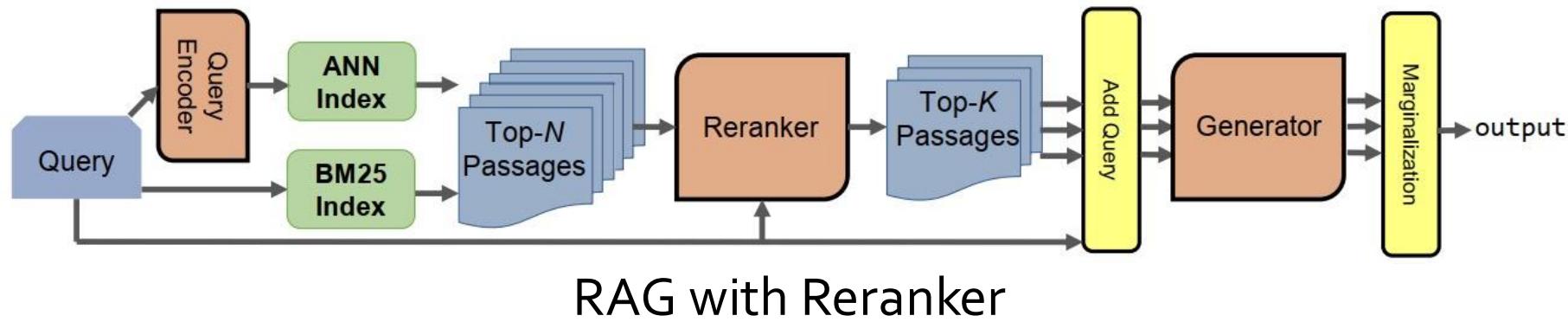
Post-Retrieval Techniques

□ Retrieved Result Rerank (Re2G)

- ❖ Results from initial retrieval can be greatly improved through the use of a reranker
- ❖ Reranker allows merging retrieval results from sources with incomparable scores, e.g., BM25 and neural initial retrieval



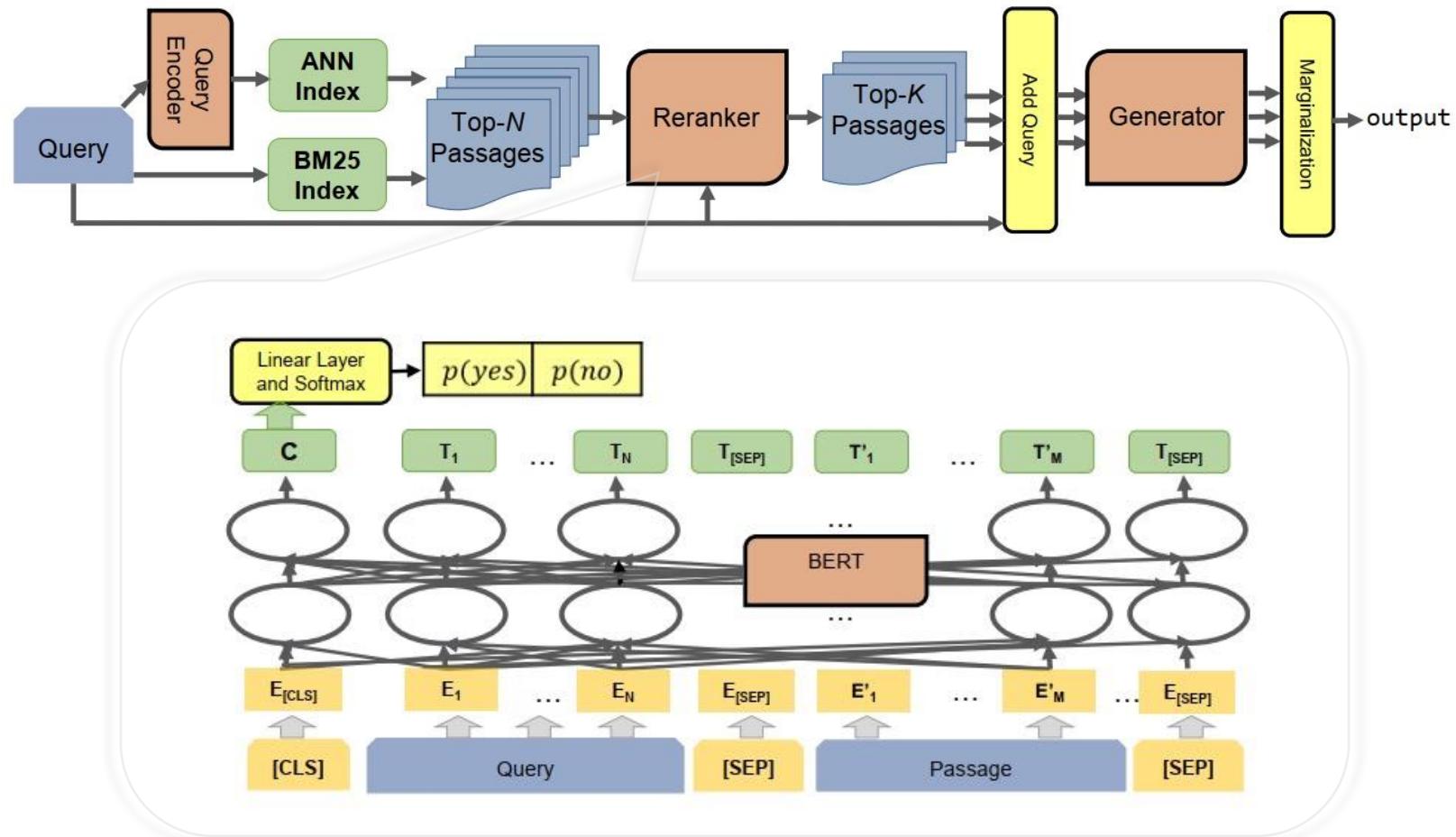
Regular RAG pipeline



RAG with Reranker

Retrieved Result Rerank (Re2G) Model

□ Reranker: interaction model based on the sequence-pair classification



Retrieved Result Rerank (Re2G) Performance

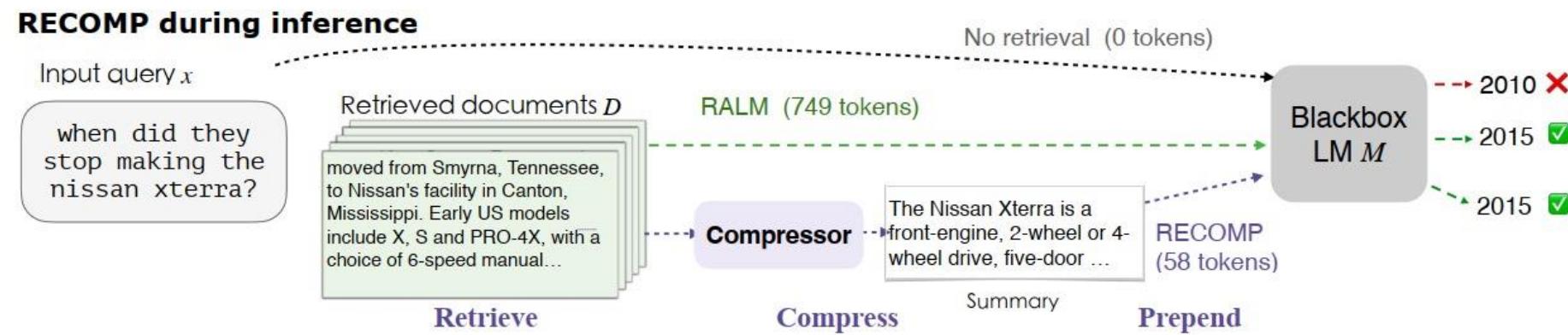
	T-REx		NQ		TriviaQA		FEVER		WoW	
	R-Prec	R@5								
BM25	46.88	69.59	24.99	42.57	26.48	45.57	42.73	70.48	27.44	45.74
DPR Stage 1	49.02	63.34	56.64	64.38	60.12	64.04	75.49	84.66	34.74	60.22
KGI ₀ DPR	65.02	75.52	64.65	69.60	60.55	63.65	80.34	86.53	48.04	71.02
Re ² G DPR	67.16	76.42	65.88	70.90	62.33	65.72	84.13	87.90	47.09	69.88
KGI ₀ DPR+BM25	60.48	80.06	36.91	66.94	40.81	64.79	65.95	90.34	35.63	68.47
Reranker Stage 1	81.22	87.00	70.78	73.05	71.80	71.98	87.71	92.43	55.50	74.98
Re ² G Reranker	81.24	88.58	70.92	74.79	60.37	70.61	90.06	92.91	57.89	74.62

Significantly outperforms pipelines without the *Rerank* stage

Post-Retrieval Techniques

❑ Retrieved Result Compression

- ❖ To reduce the computational costs and also relieve the burden of LMs to identify relevant information in long retrieved documents.



❑ Compressor Learning Objectives

- ❖ Concise
- ❖ Effective
- ❖ Faithful

Retrieved Result Compression Performance

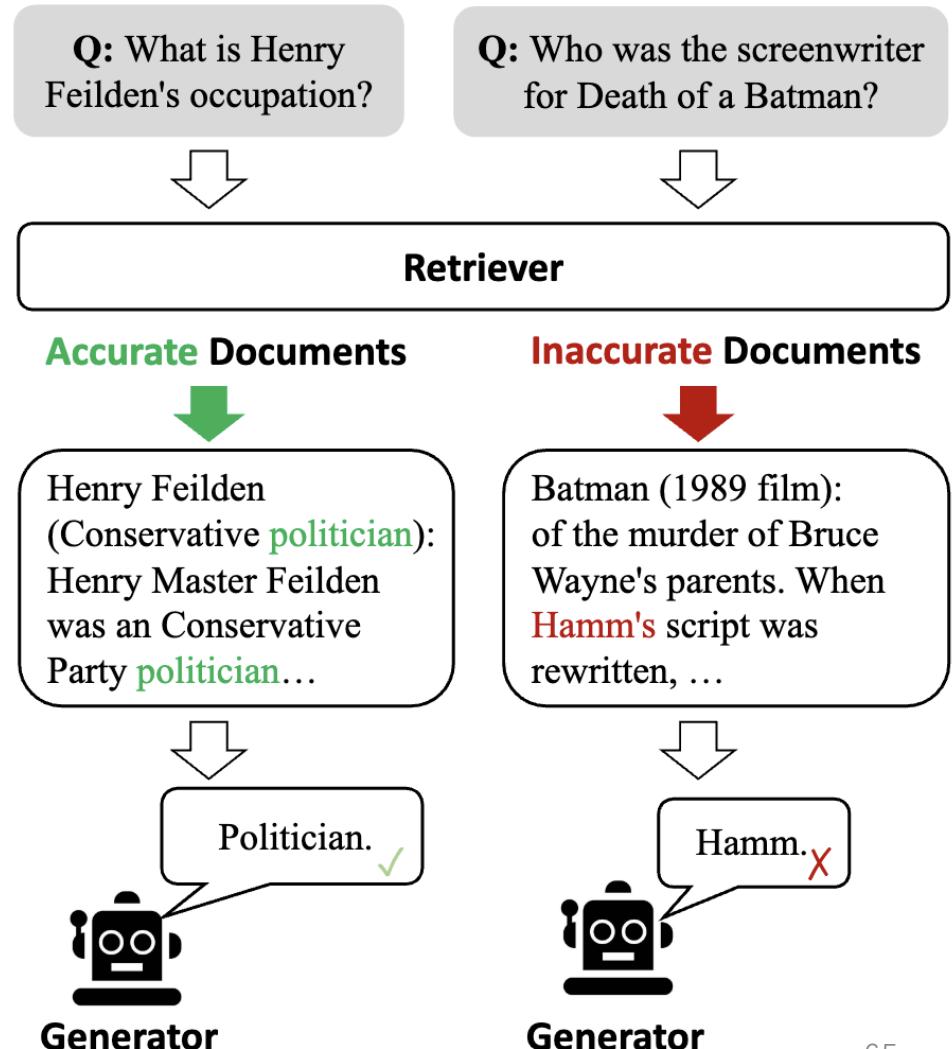
□ QA tasks

In-Context evidence	# tok	NQ		TQA		HotpotQA			
		EM	F1	# tok	EM	F1	# tok	EM	F1
-	0	21.99	29.38	0	49.33	54.85	0	17.80	26.10
<i>RALM without compression</i>									
Top 1 documents	132	33.07	41.45	136	57.84	64.94	138	28.80	40.58
Top 5 documents	660	39.39	48.28	677	62.37	70.09	684	32.80	43.90
<i>Phrase/token level compression</i>									
Top 5 documents (NE)	338	23.60	31.02	128	54.96	61.19	157	22.20	31.89
Top 5 documents (BoW)	450	28.48	36.84	259	58.16	65.15	255	25.60	36.00
<i>Extractive compression of top 5 documents</i>									
Oracle	34	60.22	64.25	32	79.29	82.06	70	41.80	51.07
Random	32	23.27	31.09	31	50.18	56.24	61	21.00	29.86
BM25	36	25.82	33.63	37	54.67	61.19	74	26.80	38.02
DPR	39	34.32	43.38	41	56.58	62.96	78	27.40	38.15
Contriever	36	30.06	31.92	40	53.67	60.01	78	28.60	39.48
Ours	37	36.57	44.22	38	58.99	65.26	75	30.40	40.14

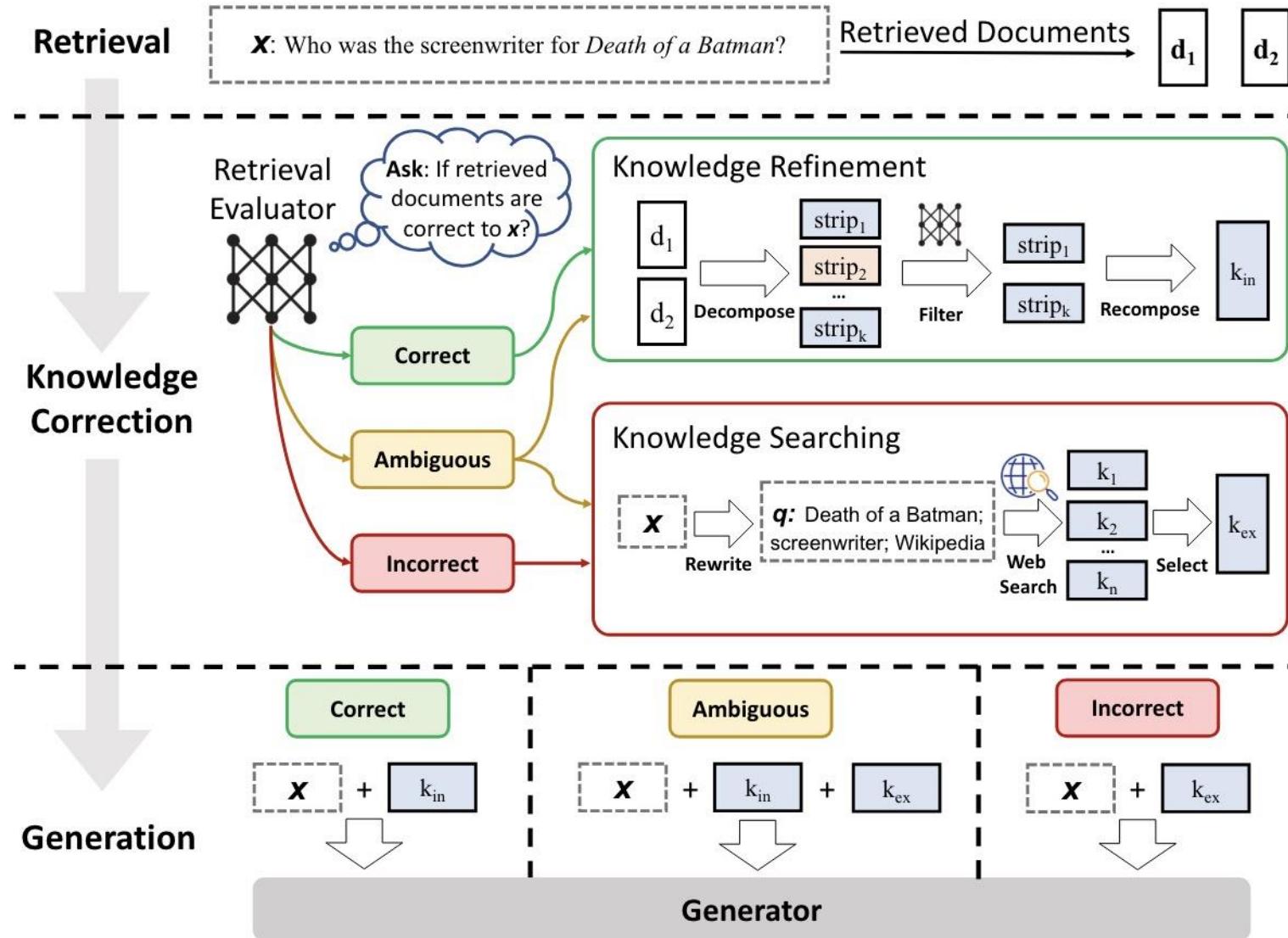
Outperforms representative sparse and dense retrievers

Post-Retrieval Techniques: Corrective RAG

- ❑ Corrective Retrieval Augmented Generation (CRAG)
 - ❖ Although retrieval-augmented generation (RAG) is a practicable complement to LLMs, it relies heavily on the **relevance of retrieved documents**
 - ❖ A lightweight **retrieval evaluator** is designed to assess the overall quality of retrieved documents for a query, returning a **confidence degree** based on which different knowledge retrieval actions can be triggered

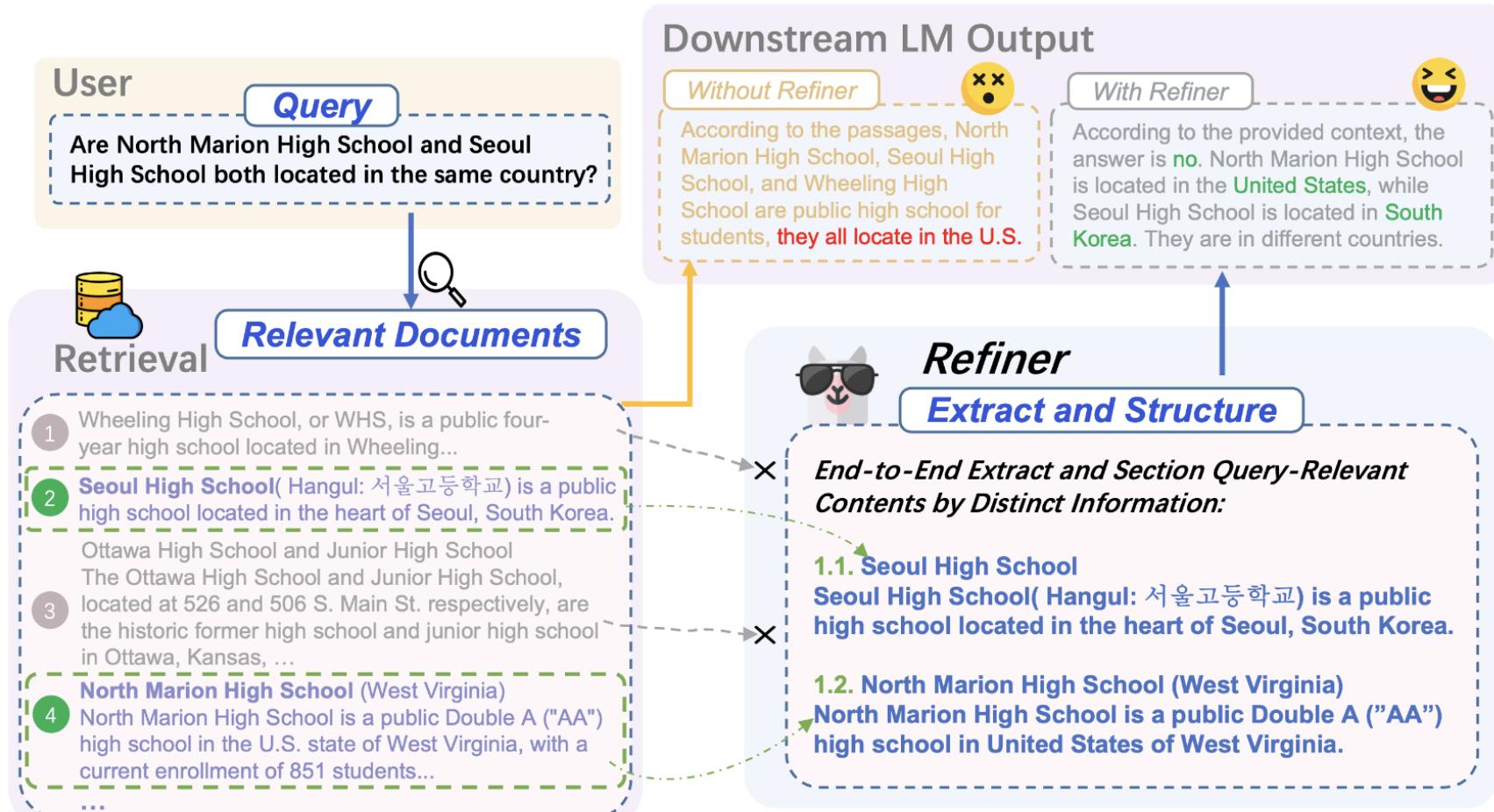


Post-Retrieval Techniques: Corrective RAG



Post-Retrieval Techniques: Refiner

- **Refiner:** leveraging a single decoder-only LLM to adaptively extract query relevant contents verbatim along with the necessary context



PART 2: Architecture of RA-LLMs and Main Modules



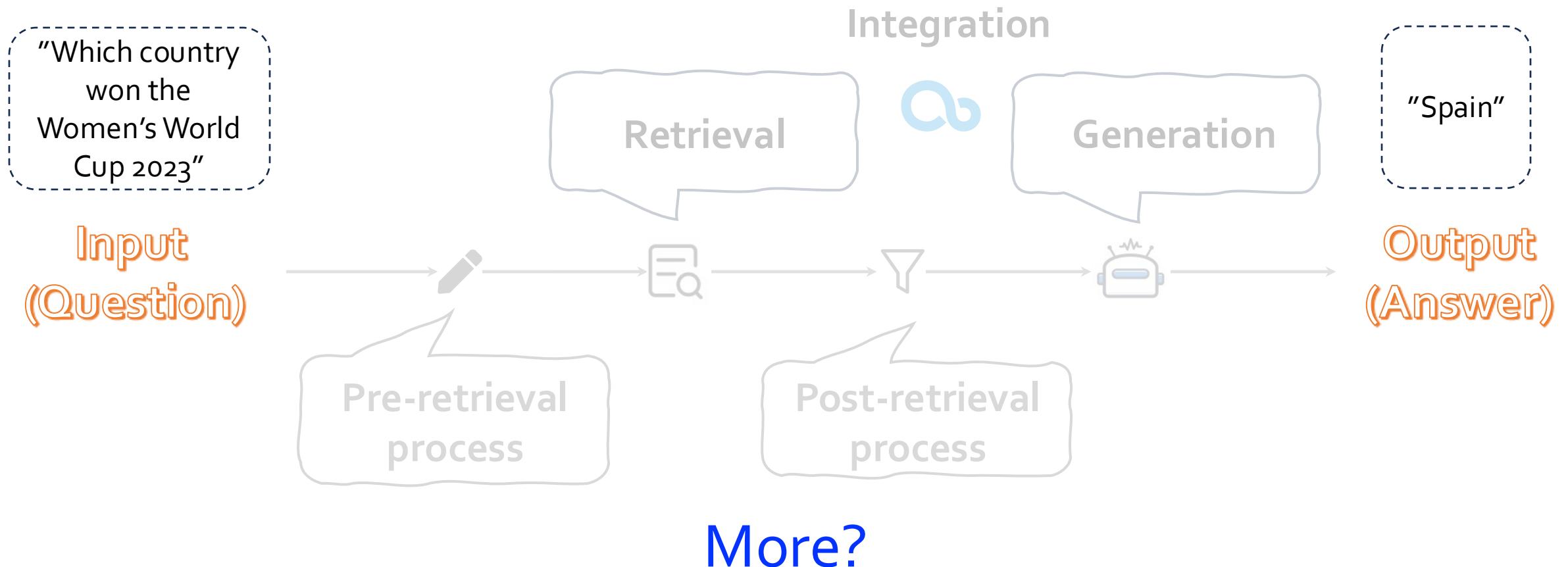
Slides



Website of this tutorial

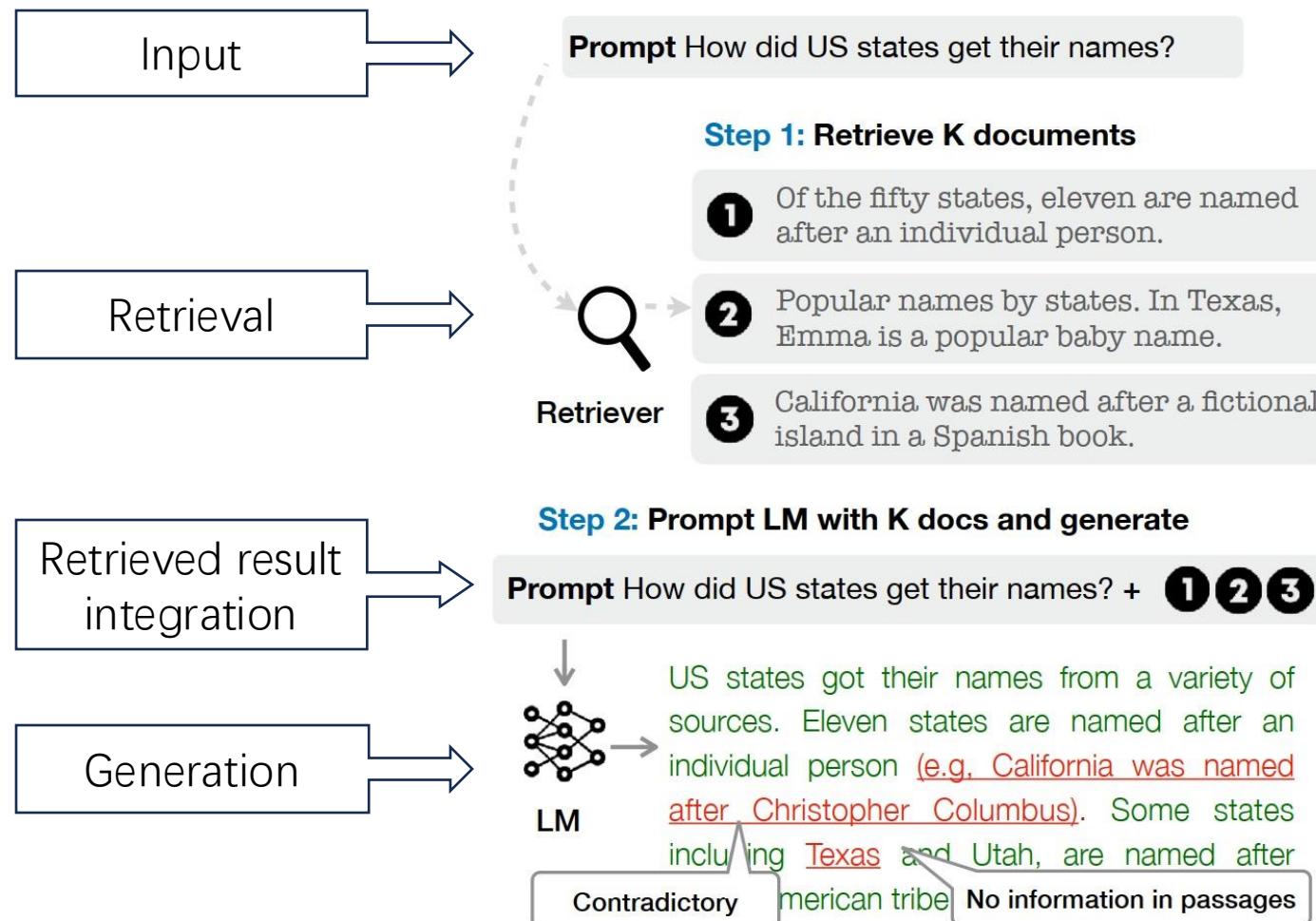
- RA-LLM architecture overview
- Retriever in RA-LLMs
- Retrieval results integration
- Pre/Post-retrieval techniques
- Special RA-LLM paradigms

Beyond Standard Pipelines and Components?

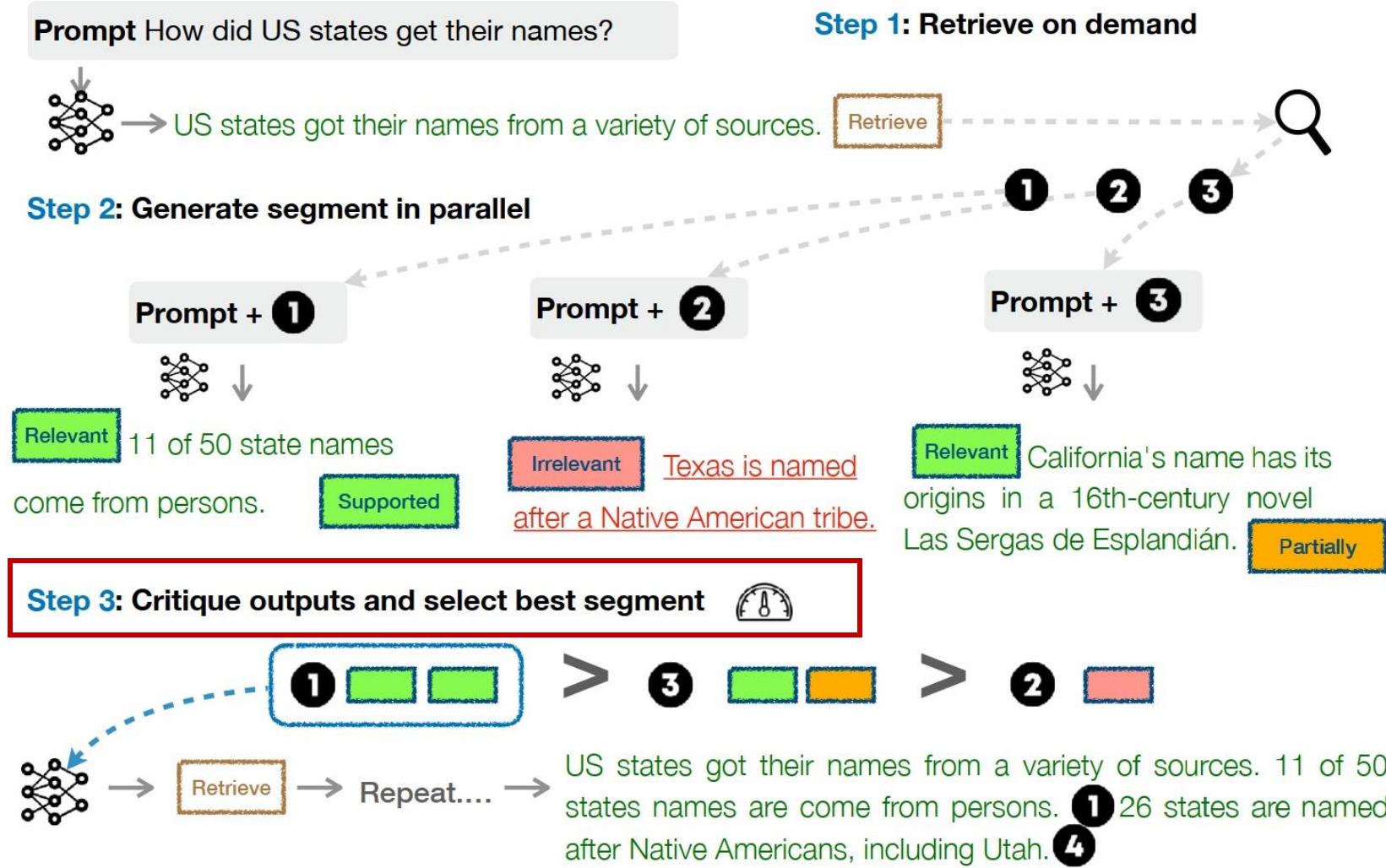


Special RAG Pipeline: Self-Reflective RAG (SELF-RAG)

□ General Retrieval-Augmented Generation (RAG)

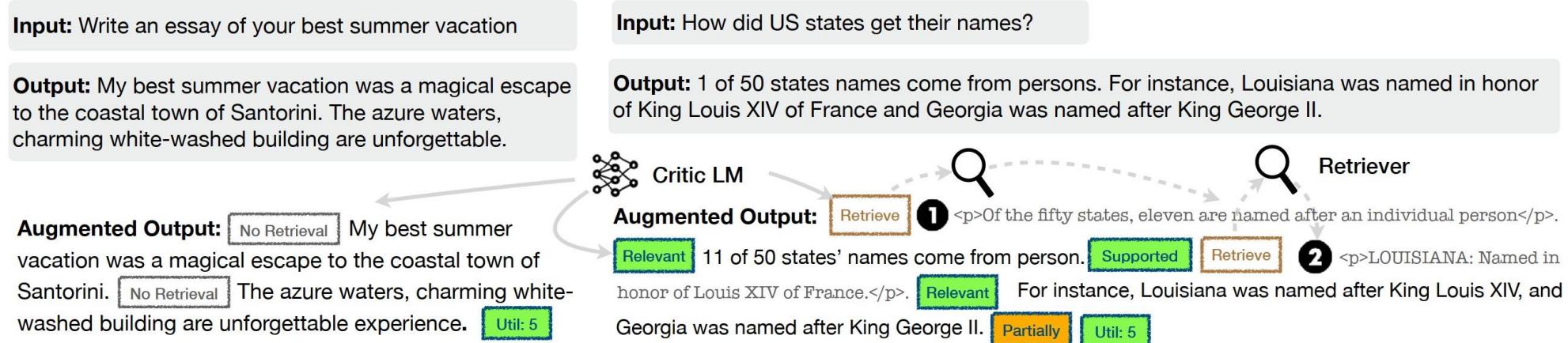


SELF-RAG Overview



Key Technical Design in SELF-RAG

❑ Critic Model Training



❑ Four types of reflection tokens used in SELF-RAG

Type	Input	Output	Definitions
Retrieve	$x / x, y$	{yes, no, continue}	Decides when to retrieve with \mathcal{R}
ISREL	x, d	{relevant, irrelevant}	d provides useful information to solve x .
IsSUP	x, d, y	{fully supported, partially supported, no support}	All of the verification-worthy statement in y is supported by d .
ISUSE	x, y	{5, 4, 3, 2, 1}	y is a useful response to x .

SELF-RAG Algorithm

Algorithm 1 SELF-RAG Inference

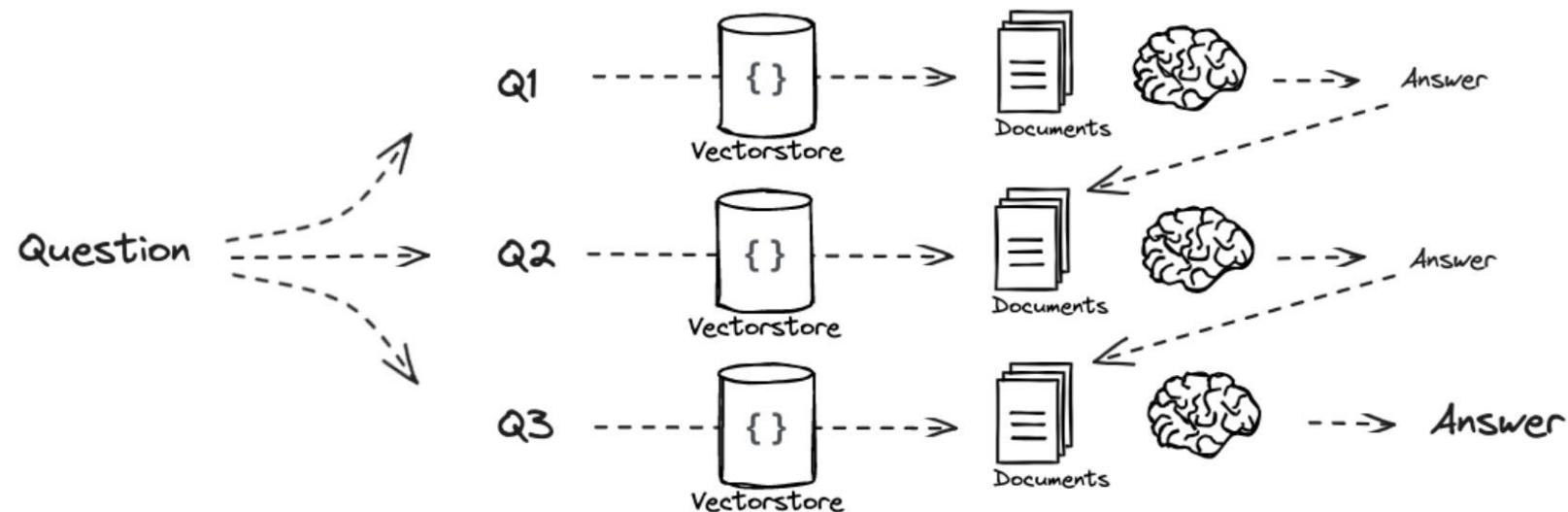
Require: Generator LM \mathcal{M} , Retriever \mathcal{R} , Large-scale passage collections $\{d_1, \dots, d_N\}$

- 1: **Input:** input prompt x and preceding generation $y_{<t}$, **Output:** next output segment y_t
 - 2: \mathcal{M} predicts **Retrieve** given $(x, y_{<t})$
 - 3: **if** **Retrieve** == Yes **then**
 - 4: Retrieve relevant text passages \mathbf{D} using \mathcal{R} given (x, y_{t-1}) ▷ **Retrieve**
 - 5: \mathcal{M} predicts **ISREL** given x, d and y_t given $x, d, y_{<t}$ for each $d \in \mathbf{D}$ ▷ **Generate**
 - 6: \mathcal{M} predicts **ISSUP** and **ISUSE** given x, y_t, d for each $d \in \mathbf{D}$ ▷ **Critique**
 - 7: Rank y_t based on **ISREL**, **ISSUP**, **ISUSE** ▷ Detailed in Section 3.3
 - 8: **else if** **Retrieve** == No **then**
 - 9: \mathcal{M}_{gen} predicts y_t given x ▷ **Generate**
 - 10: \mathcal{M}_{gen} predicts **ISUSE** given x, y_t ▷ **Critique**
-

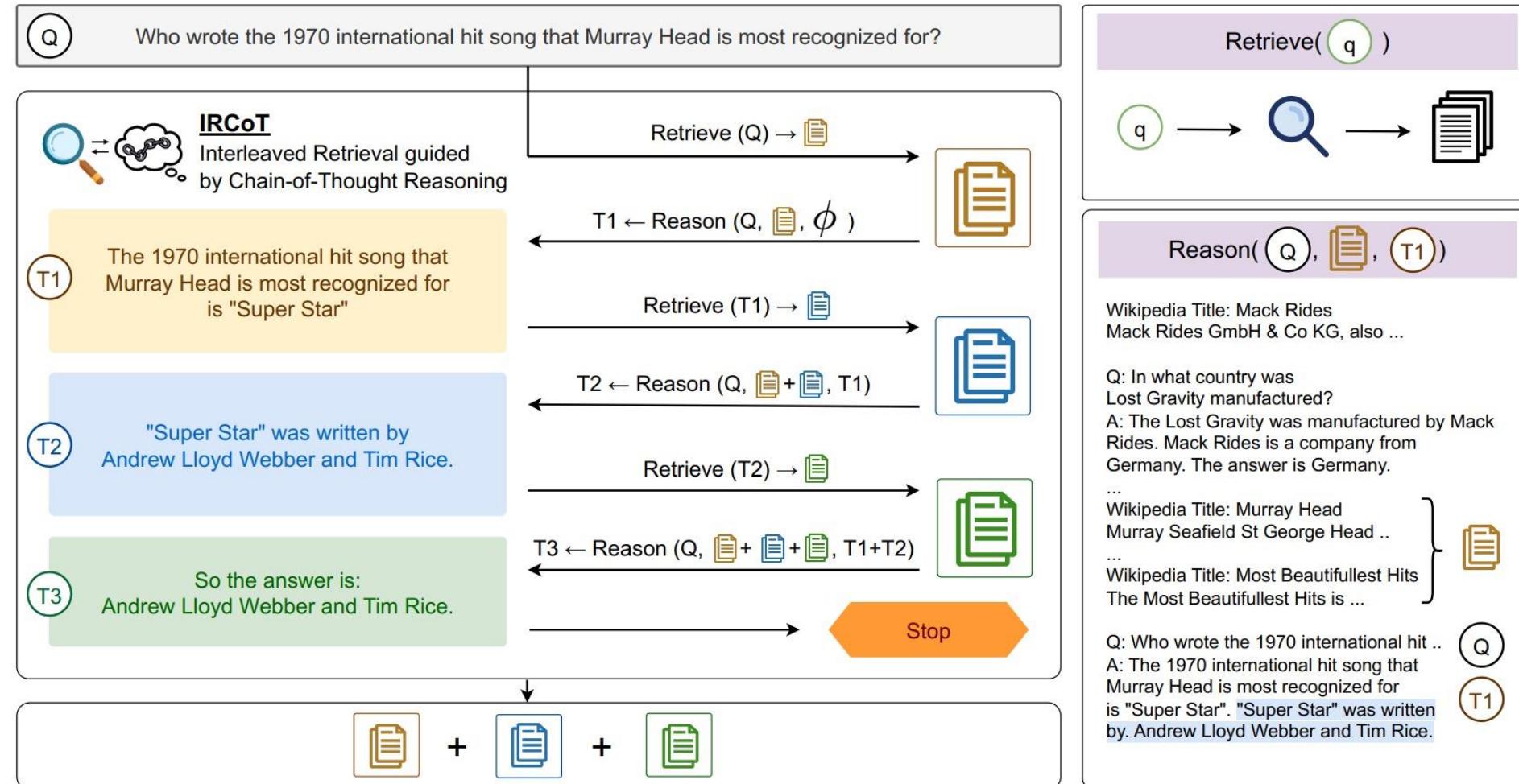
Special RAG Pipeline: Recursively Answer

❑ Chain-of-Thought + RAG

- ❖ One-step retrieve-and-read approach is insufficient for multi-step QA
- ❖ What to retrieve depends on what has already been derived, which in turn may depend on what was previously retrieved

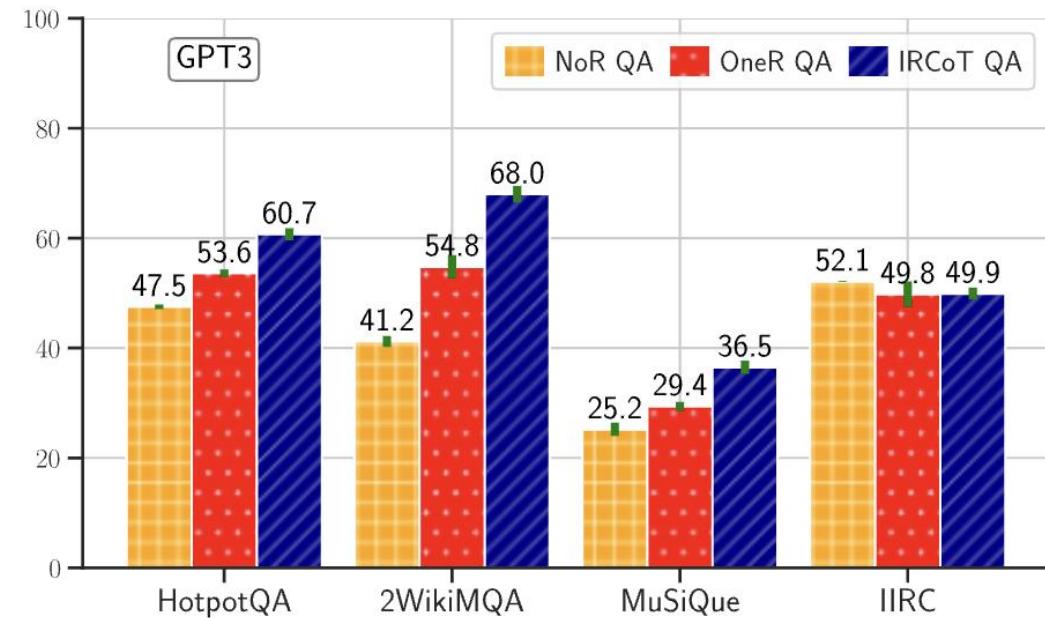
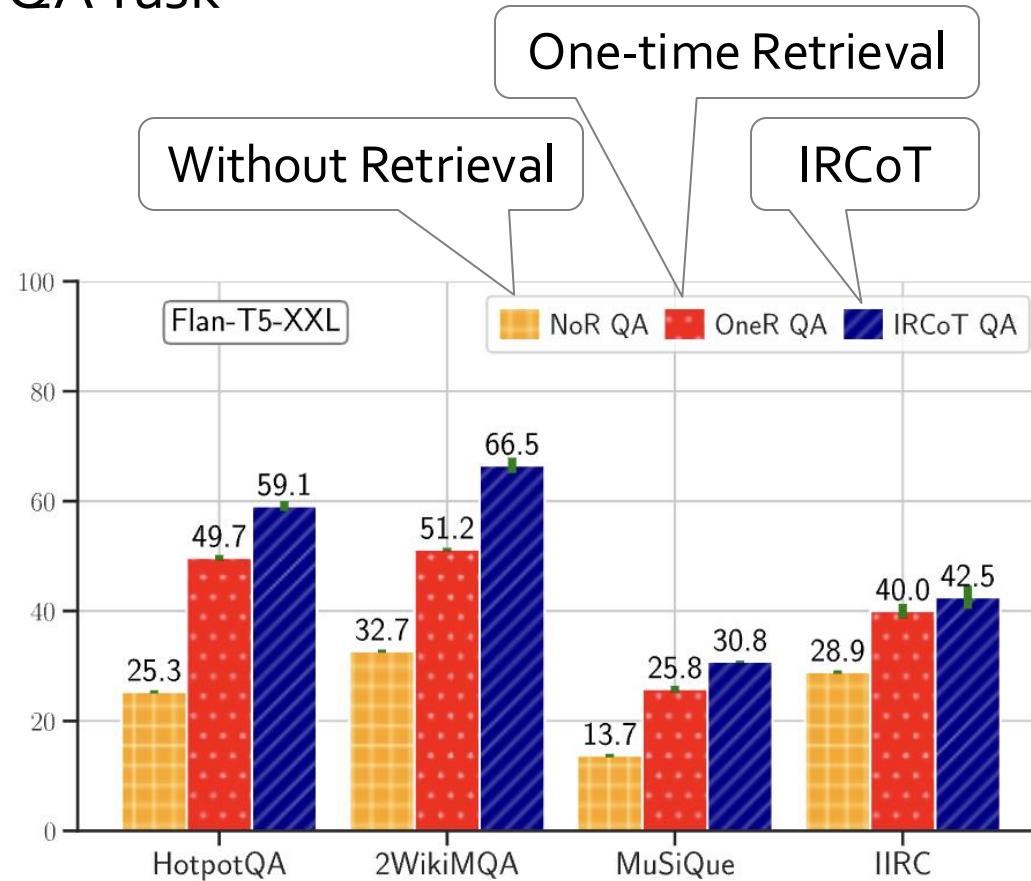


Interleaved Retrieval guided by Chain-of-Thought (IRCoT)



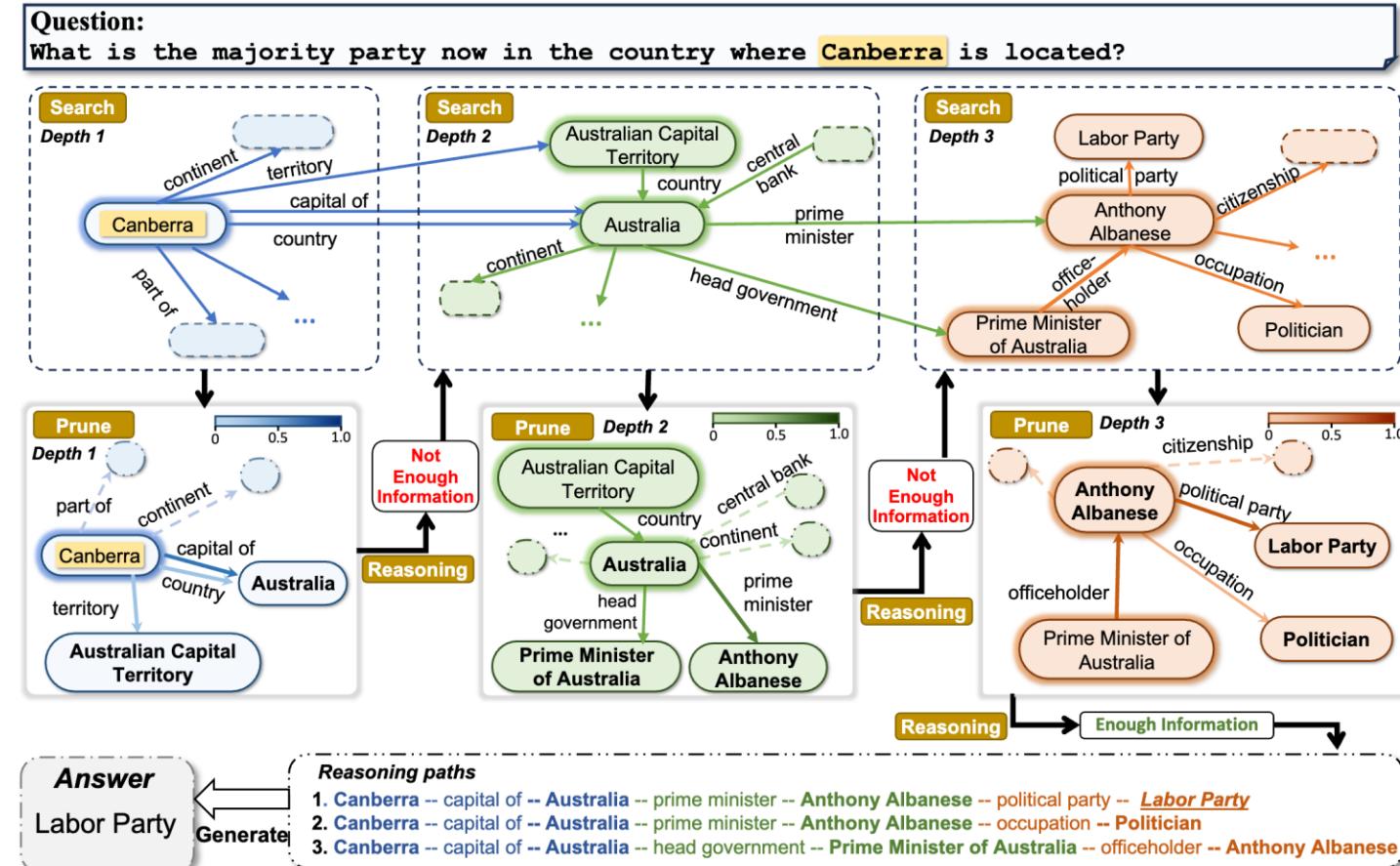
IRCoT Performance

❑ QA Task



Iterative Retrieval with KG

□ Think-on-Graph





Tutorial Outline

- **Part 1: Introduction** of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)
- **Part 2: Architecture** of RA-LLMs and **Main Modules** (Dr. Yujuan Ding)
- **Part 3: Data Management for RA-LLMs (Pangjing Wu)**
- **Part 4: Learning Approach of RA-LLMs (Liangbo Ning)**
- **Part 4: Applications of RA-LLMs (Shijie Wang)**
- **Part 5: Challenges and Future Directions** of RA-LLMs (Dr. Wenqi Fan)

Website of this tutorial
Check out the slides and more information!



Part 3: RA-LLM Data Management



Presenter
Pangjing Wu
HK PolyU

- **Vector Generation**
- **Indexing**
- **Query Processing**
- **RAG-Oriented VDB Pipeline**

Part 3: RA-LLM Data Management



Website of this tutorial

- **Vector Generation**
- Indexing
- Query Processing
- RAG-Oriented VDB Pipeline

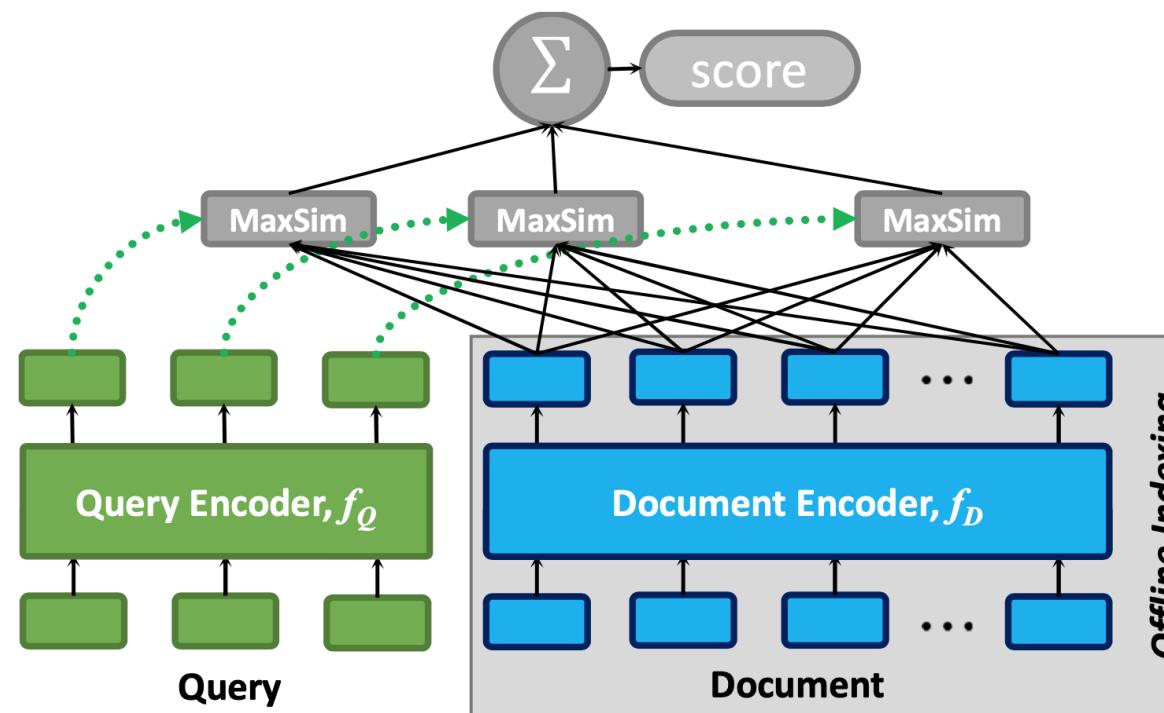
RA-LLM Learning: Vector Generation

- Vector generation transforms **documents** into **vectors**.
 - **Dense embedding:** using **low-dimensional dense vectors** capturing semantics.
 - **Sparse embedding:** using **high-dimensional vectors with mostly zero entries**.

RA-LLM Learning: Vector Generation

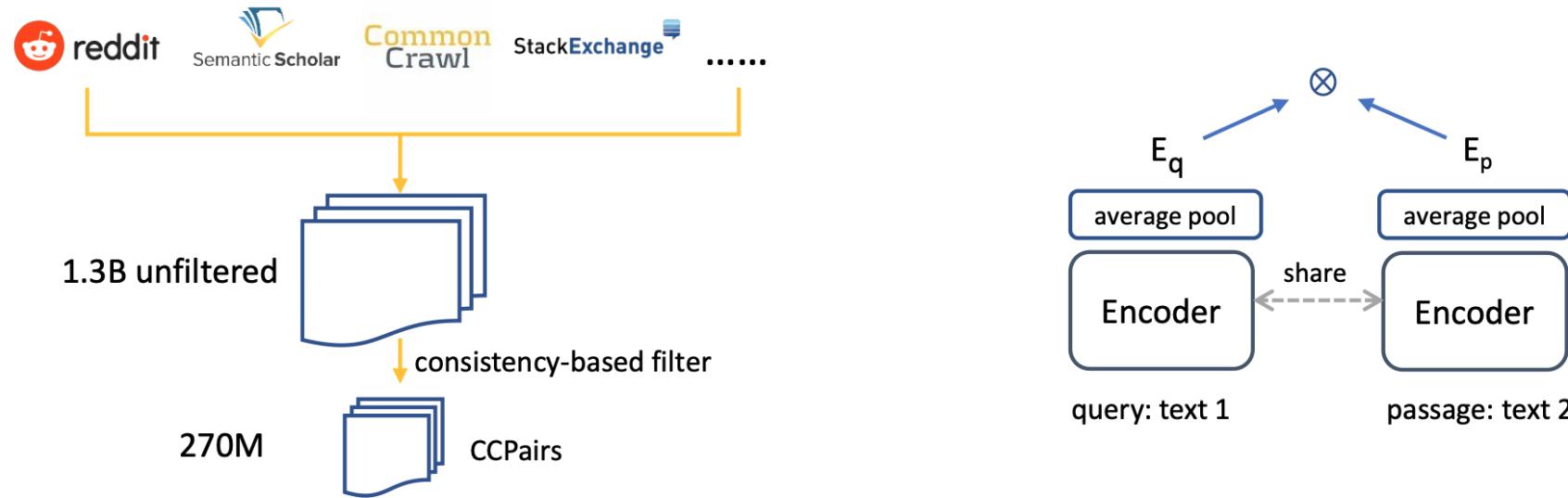
- **CoBERT**

- Generates **query and document vectors** separately using BERT, with **MaxSim** for interaction scoring.



RA-LLM Learning: Vector Generation

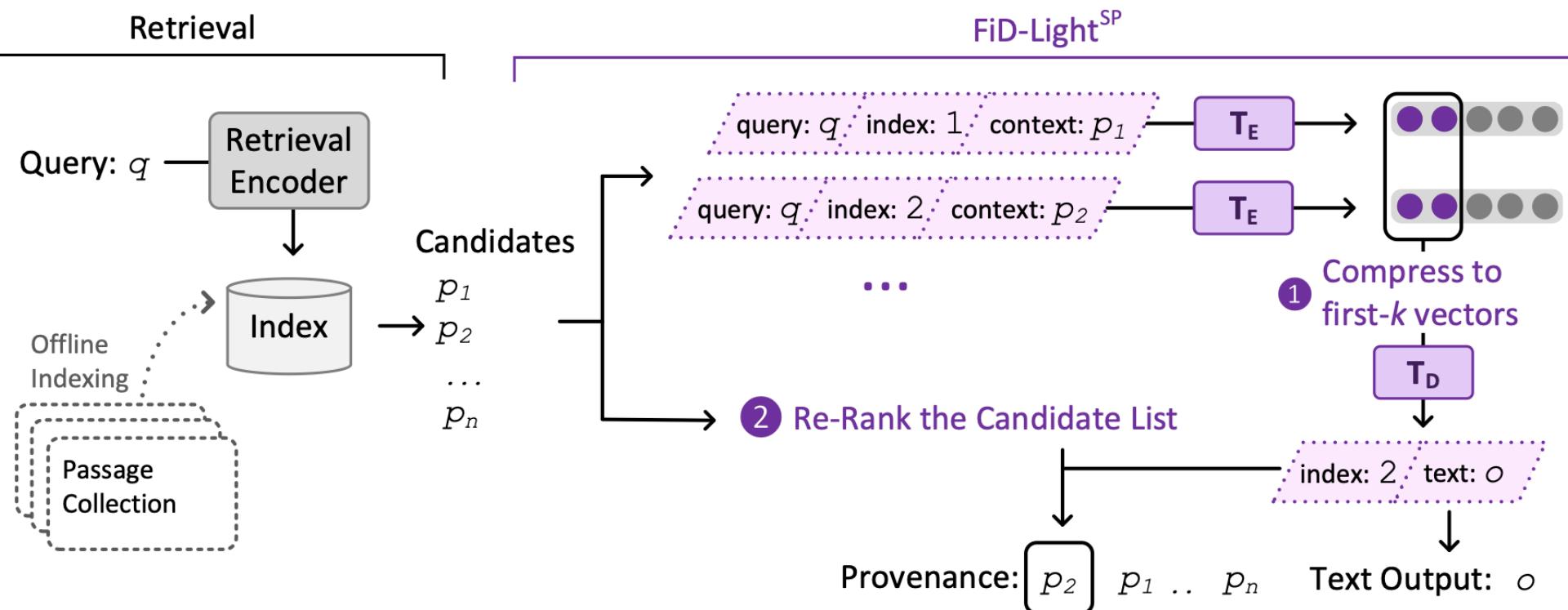
- E5
 - Uses a shared encoder with **contrastive loss** to generate consistent text embeddings.



$$\min L_{\text{cont}} = -\frac{1}{n} \sum_i \log \frac{e^{s_{\theta}(q_i, p_i)}}{e^{s_{\theta}(q_i, p_i)} + \sum_j e^{s_{\theta}(q_i, p_{ij}^-)}}$$

RA-LLM Learning: Vector Generation

- **FiD-Light**
 - Compresses encoder outputs into the first- k vectors for efficient ranking.

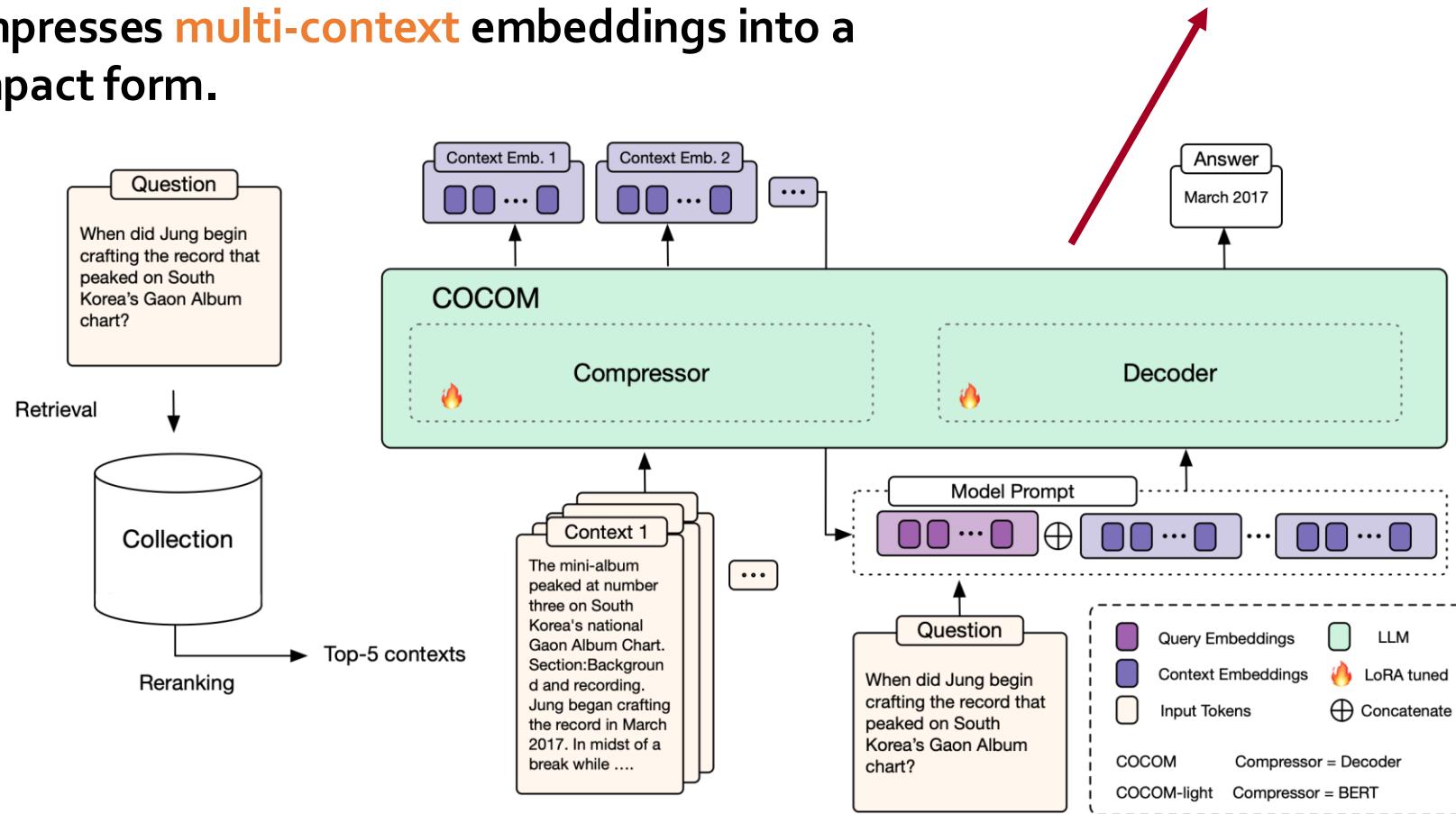


RA-LLM Learning: Vector Generation

- **COCOM**

- Compresses multi-context embeddings into a compact form.

$$\mathcal{L}(\theta_{LLM}, \phi_{comp}) = - \sum_{x_t \in \mathcal{X}_B} \log P_{\theta_{LLM}}(x_t | \phi_{comp}(\mathcal{X}_A), x_1, \dots, x_{t-1})$$

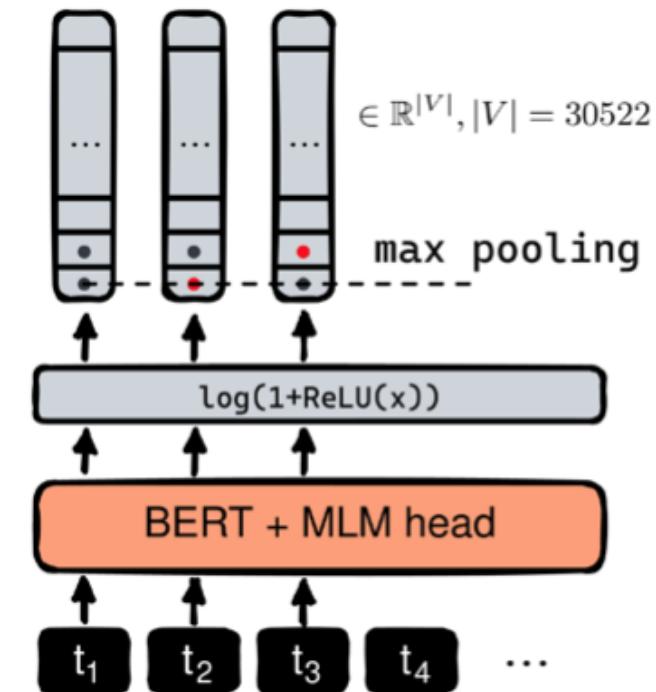
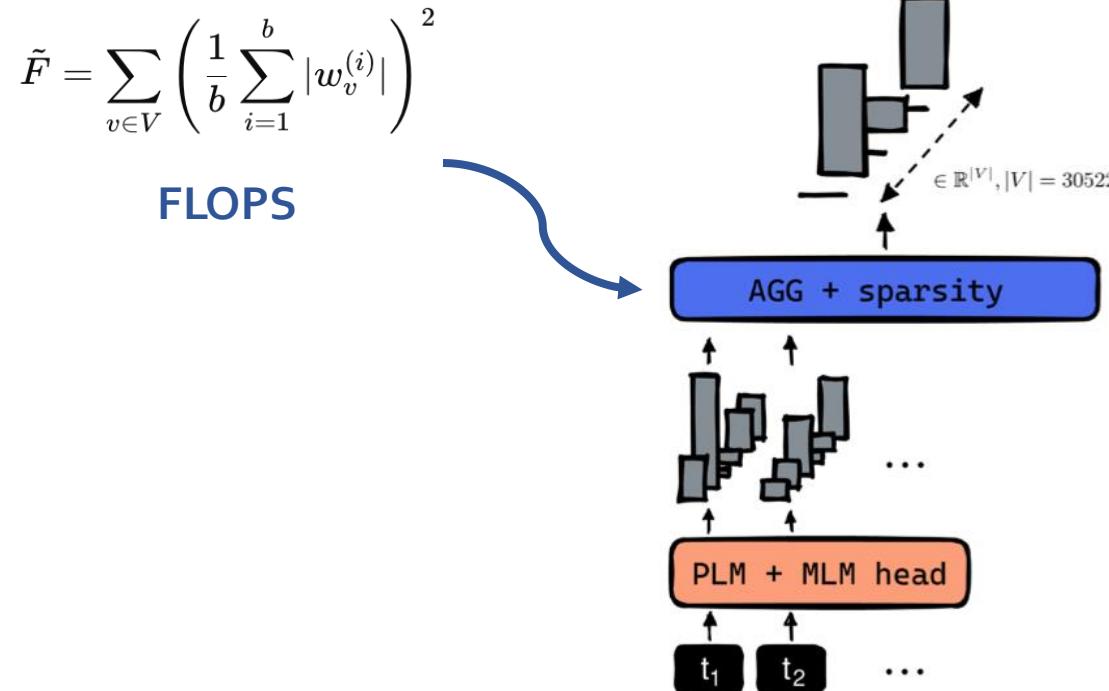


RA-LLM Learning: Vector Generation

- Vector generation transforms **documents** into **vectors**.
 - Dense embedding: using **low-dimensional dense vectors** capturing semantics.
 - Sparse embedding: using **high-dimensional vectors with mostly zero entries**.

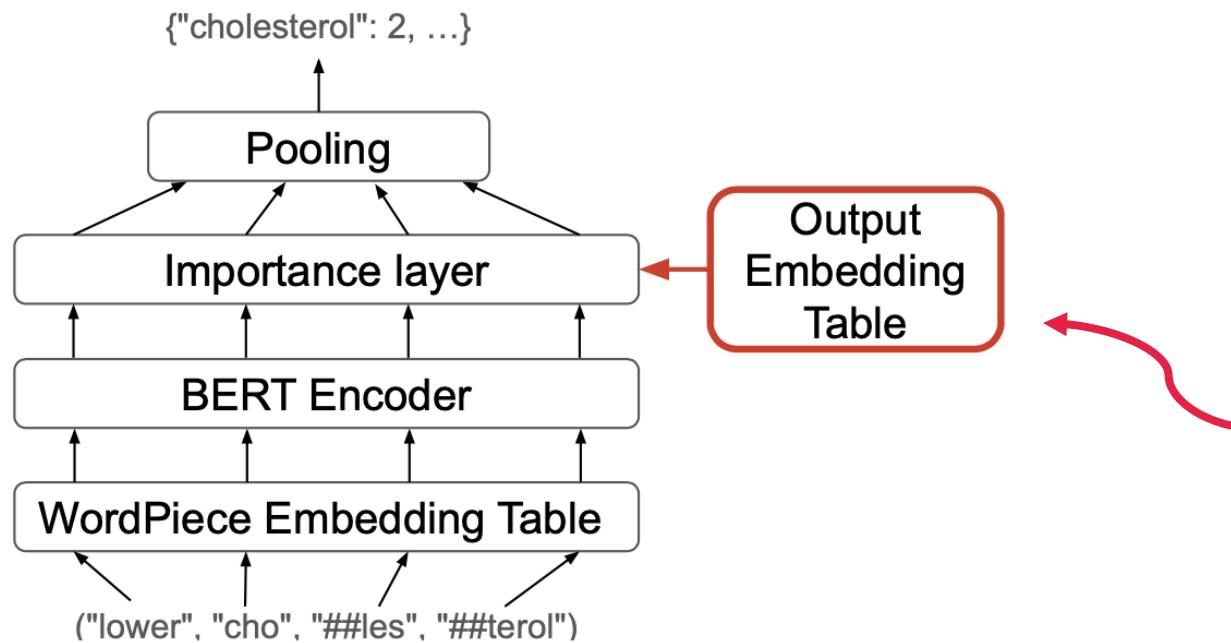
RA-LLM Learning: Vector Generation

- Efficient-SPLADE
 - Uses **log-saturated term weighting** with **FLOPS regularization**.



RA-LLM Learning: Vector Generation

- **Expanded-SPLADE**
 - Expands vocabularies by **reprojection**.



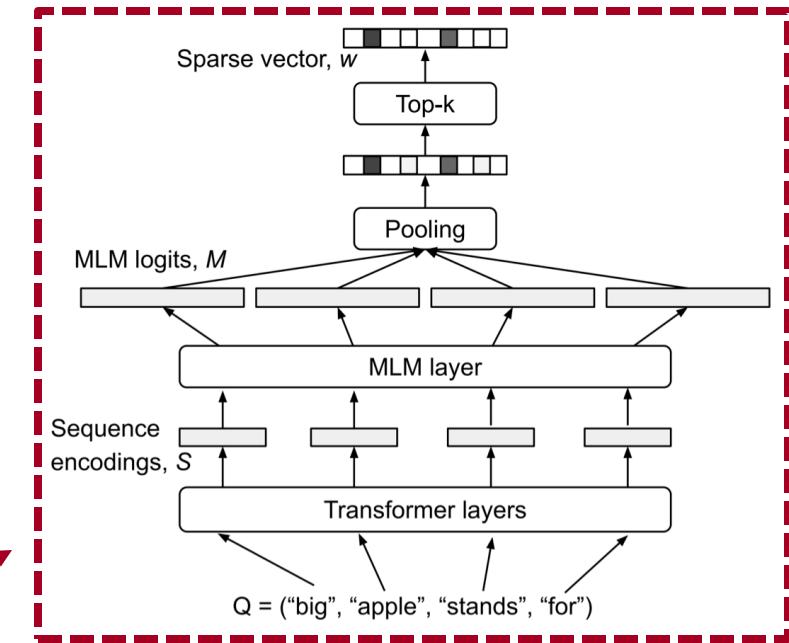
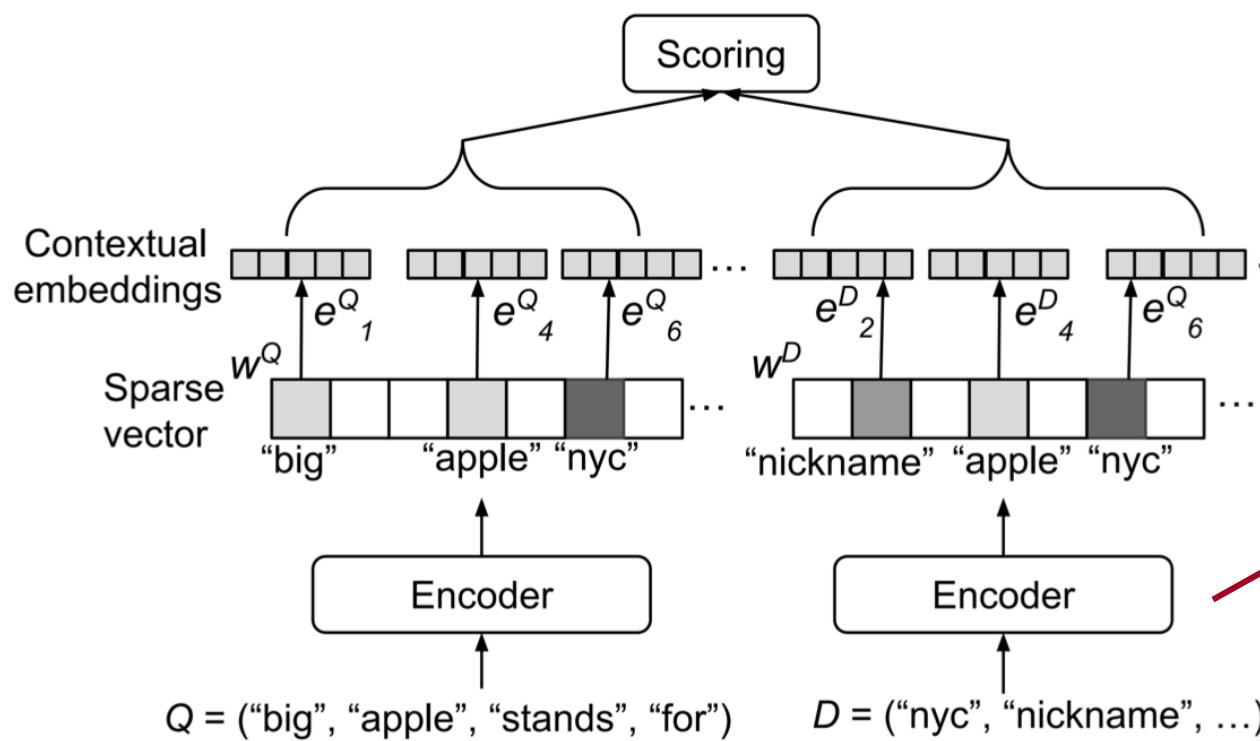
Algorithm 2: Expanded-SPLADE

Inputs: Text t , BERT WordPiece vocab V , Output vocab U
Output: $w^{(t)} \in \mathbb{R}^U$

- 1 $(t_1, t_2, \dots, t_N) \leftarrow \text{tokenize } t \text{ in } V$
- 2 $(h_1, h_2, \dots, h_N) \leftarrow \text{BERT}(t_1, t_2, \dots, t_N)$
- 3 **foreach** $1 \leq i \leq N, j \in U$ **do**
 - 4 $w_{ij}^{(t)} = \text{transform}(h_i)^T E'_j + b'_j$
 /* $E'_j \in \mathbb{R}^H$ is a new learned embedding for j */
- 5 **foreach** $j \in U$ **do**
 - 6 $w_j^{(t)} = \max_{1 \leq i \leq N} \log(1 + \text{ReLU}(w_{ij}^{(t)}))$
- 7 **return** $\langle w_j^{(t)} : j \in U \rangle$

RA-LLM Learning: Vector Generation

- **SparseEmbed**
 - Using top-k MLM logits and lightweight contextual attention pooling.



RA-LLM Learning: Vector Generation

- **Dense embedding**

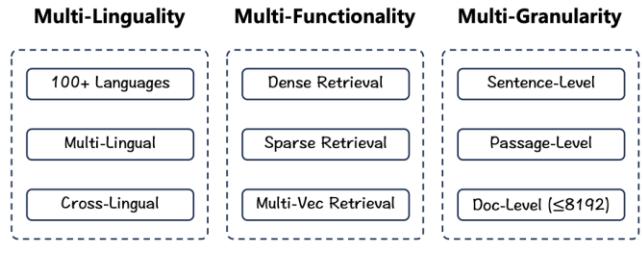
- ✓ Strong performance in semantic search
 - ✗ May retrieve irrelevant documents

- **Sparse embedding**

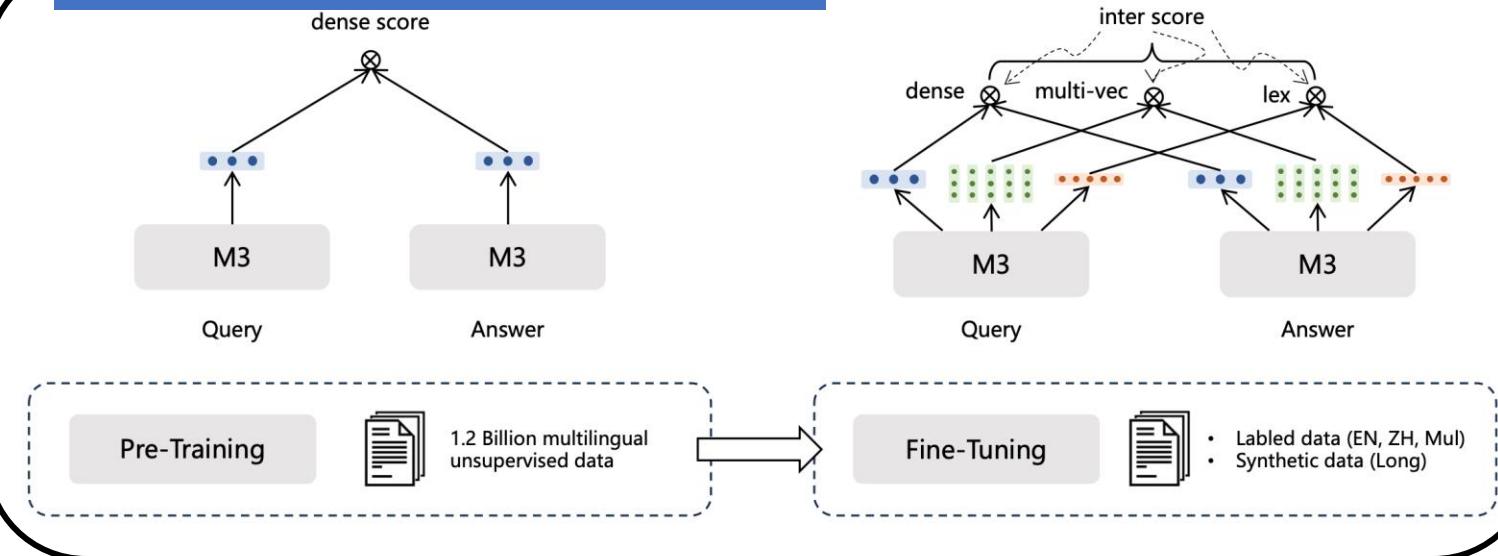
- ✓ Effective for exact match retrieval
 - ✗ High-dimensional

RA-LLM Learning: Vector Generation

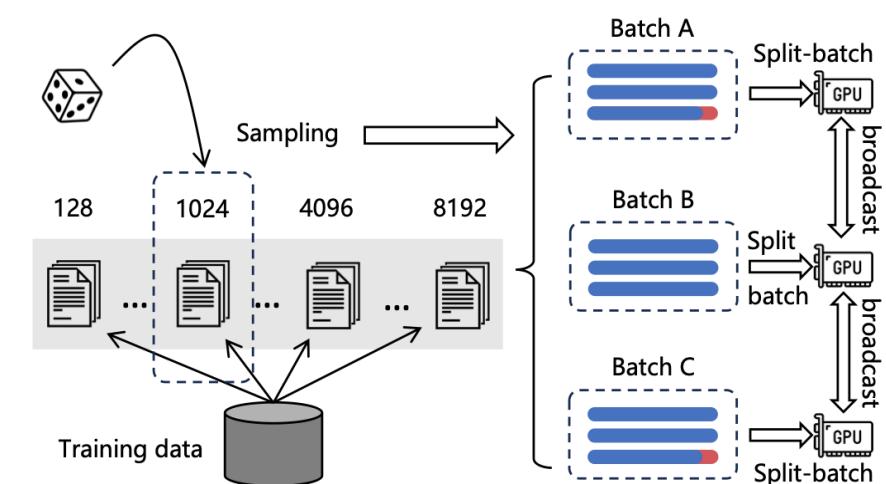
- **M3-Embedding**
 - **Simultaneously** accomplish the three common retrievals:
 - dense retrieval,
 - multi-vector retrieval,
 - and sparse retrieval.



Multi-stage Training



Efficient Batching



Part 3: RA-LLM Data Management



Website of this tutorial

- Vector Generation
- Indexing
- Query Processing
- RAG-Oriented VDB Pipeline

Part 3: RA-LLM Data Management



Website of this tutorial

- Vector Generation
- Indexing
 - partitioning
- Query Processing
- RAG-Oriented VDB Pipeline

RA-LLM Learning: Indexing

- Vector indexes **speed up** queries by **minimizing the number of comparisons**. This is achieved by **partitioning** data so that only a small subset is compared.
 - **Tables** divide data into buckets containing similar vectors.
 - **Trees** are a nesting of tables.
 - **Graphs** connect similar vectors with virtual edges that can then be traversed.

RA-LLM Learning: Indexing

- SPANN

- Learning-based **hash indexing (table)** for fast search.
- Vectors X are hashed using $h(x)$, assigned to N posting lists $\{X_1, \dots, X_N\}$.

Balanced Posting List Sizes

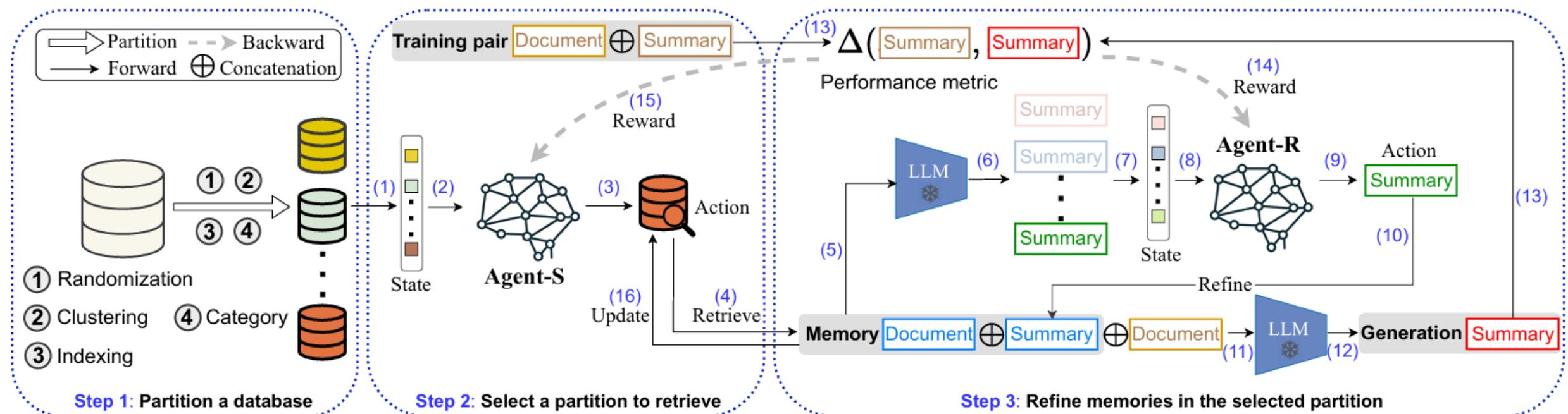
$$\min_{\mathbf{H}, \mathbf{C}} \|\mathbf{X} - \mathbf{HC}\|_F^2 + \lambda \sum_{i=1}^N \left(\sum_{l=1}^{|\mathbf{X}|} h_{li} - |\mathbf{X}|/N \right)^2, \text{ s.t. } \sum_{i=1}^N h_{li} = 1.$$

Adaptive Hashing

$$\begin{aligned} \mathbf{x} \in \mathbf{X}_{ij} &\iff \text{Dist}(\mathbf{x}, \mathbf{c}_{ij}) \leq (1 + \epsilon_1) \times \text{Dist}(\mathbf{x}, \mathbf{c}_{i1}), \\ &\quad \text{Dist}(\mathbf{x}, \mathbf{c}_{i1}) \leq \text{Dist}(\mathbf{x}, \mathbf{c}_{i2}) \leq \dots \leq \text{Dist}(\mathbf{x}, \mathbf{c}_{iK}) \end{aligned}$$

RA-LLM Learning: Indexing

- M-RAG
 - Data partitioning with RL agents to optimize data selection and refinement.



RA-LLM Learning: Indexing

- Vector indexes **speed up** queries by **minimizing the number of comparisons**. This is achieved by **partitioning** data so that only a small subset is compared.
 - **Tables** divide data into buckets containing similar vectors.
 - **Trees** are a nesting of tables.
 - **Graphs** connect similar vectors with virtual edges that can then be traversed.

RA-LLM Learning: Indexing

- **HST+**

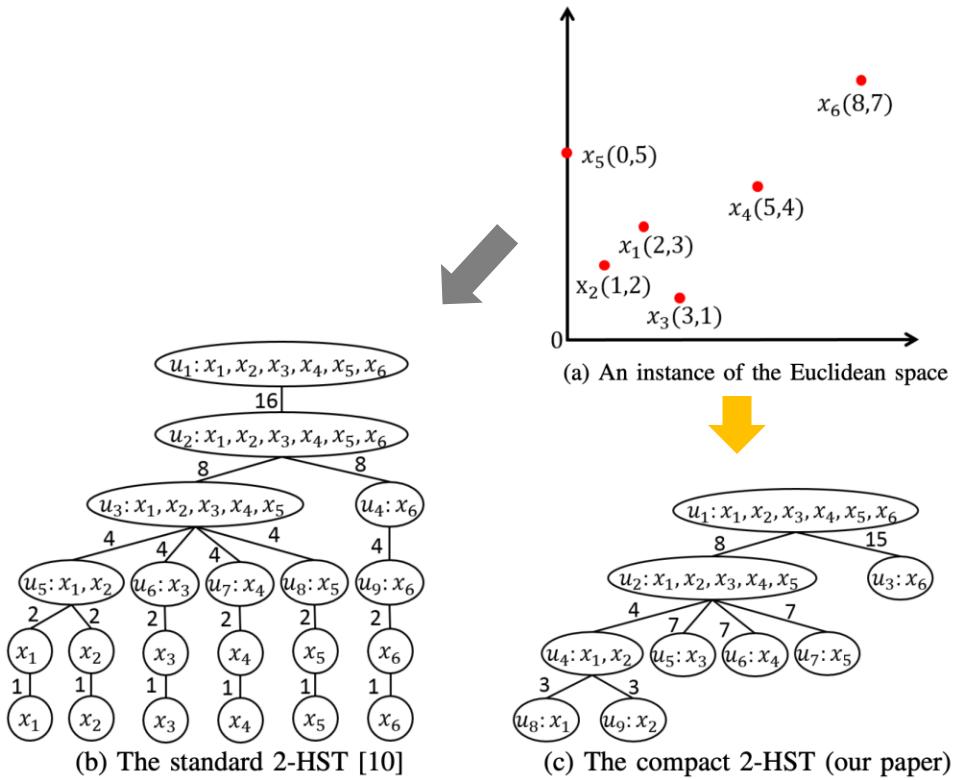
Algorithm 2: Our construction method HST+DPO

```

1 Randomly sample a permutation  $\pi$  and  $\beta \in [1/k, 1]$ ;
2  $\Delta \leftarrow \max_{y \in \mathcal{V}} \mathcal{D}(\pi[1], y)$ ,  $\mathcal{H} \leftarrow \lceil \log_k (\Delta + 1) \rceil + 1$ ;
3 Root  $rt \leftarrow \mathcal{V}$ , radius  $r_1 \leftarrow k^{\mathcal{H}} \cdot \beta$ , node set  $U_1 \leftarrow \{rt\}$ ;
4  $\text{cen}[1 \dots n] \leftarrow \{1, 1, \dots, 1\}$ ; // Initialization
5 for level  $i \leftarrow 2$  to  $\mathcal{H} + 1$  do
6   Radius  $r_i \leftarrow r_{i-1}/k$ , the set of nodes  $U_i \leftarrow \emptyset$ ;
7   foreach point  $x$  in  $\mathcal{V}$  do // Improved DP
8     Update  $\text{cen}[x]$  by Equation (6);
9   foreach node  $u$  in  $U_{i-1}$  do
10    vec  $\leftarrow$  vec  $\cup$   $\{(\text{cen}[x], x, u) \mid \text{point } x \in u\}$ ;
11   Counting-sort vec by the cen in each triple;
12   foreach continuous interval in vec such that each
13     point  $x$  was contained in the same node  $u$  do
14     A node  $v \leftarrow$  the points of this interval;
15     Add  $v$  to the children of  $u$  with weight
16      $k^{\mathcal{H}-i+2} \cdot \beta$  and add  $v$  to the set of nodes  $U_i$ ;
17   foreach redundant  $u \in U_{i-1}$  do // Compress
18      $w_0 \leftarrow$  edge weight between  $u$  and  $\text{par}(u)$ ;
19     In the children of  $\text{par}(u)$ , replace  $u$  with  $u$ 's
20     child node with edge weight  $w_0 + k^{\mathcal{H}-i+2} \cdot \beta$ ;
21 return  $T$  is created by the nodes in  $U_1, \dots, U_{\mathcal{H}+2}$ ;

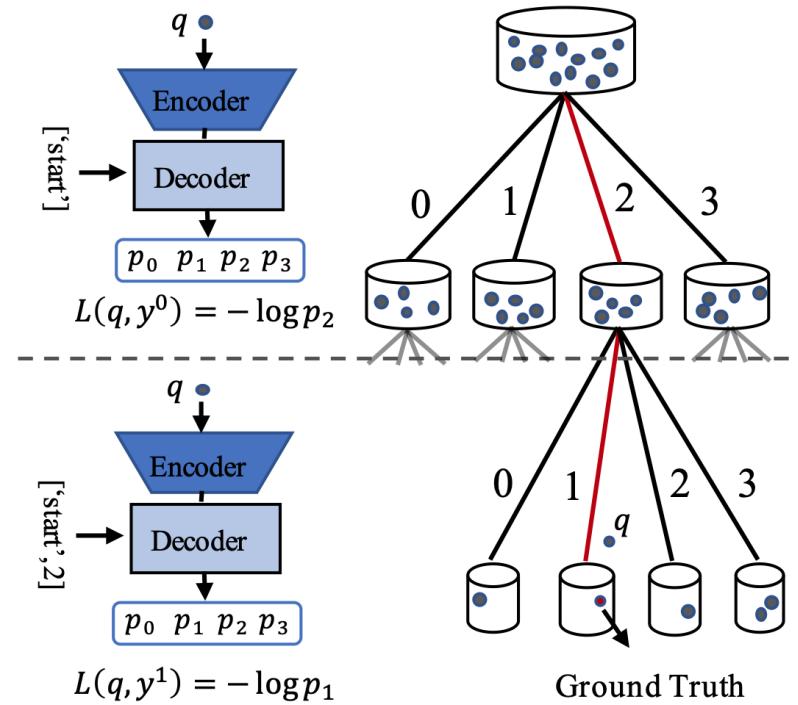
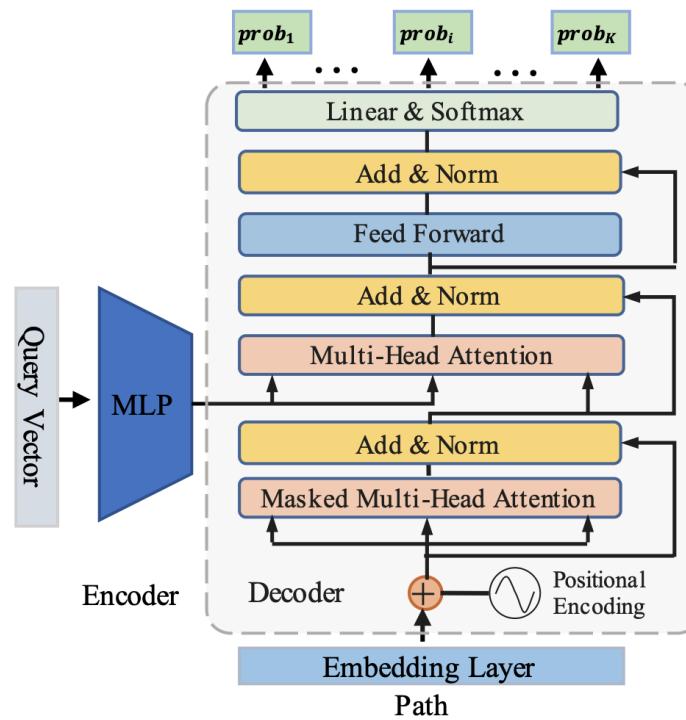
```

$$\begin{aligned} \text{cen}[x][i] = & \\ \begin{cases} \text{cen}[x][i-1], & \text{if } \mathcal{D}(x, \pi[\text{cen}[x][i-1]]) \leq r_i \\ \text{minimum } j > \text{cen}[x][i-1] \text{ s.t. } \mathcal{D}(x, \pi[j]) \leq r_i, & \text{otherwise} \end{cases} \end{aligned} \quad (6)$$



RA-LLM Learning: Indexing

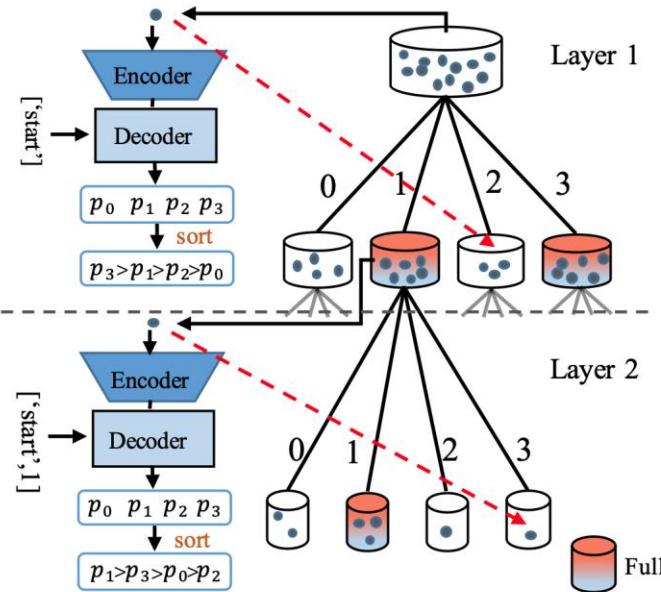
- BATLearn
 - Utilizes a **learnable balanced K-ary tree** with **sequence-to-sequence routing**



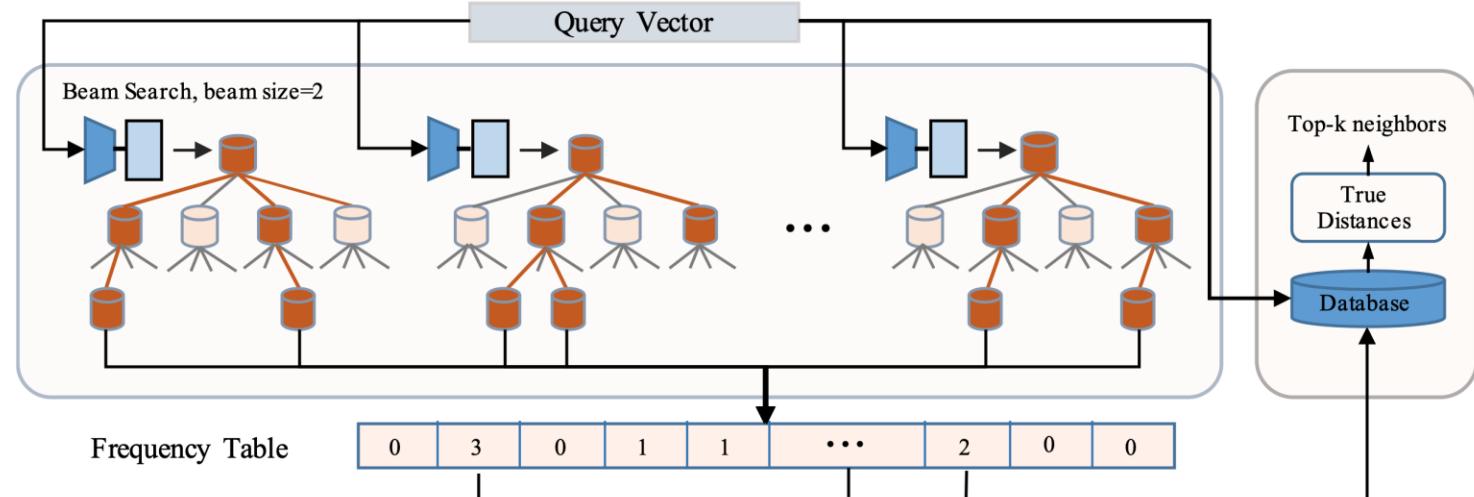
RA-LLM Learning: Indexing

- **BATLearn**

Learning



Inference

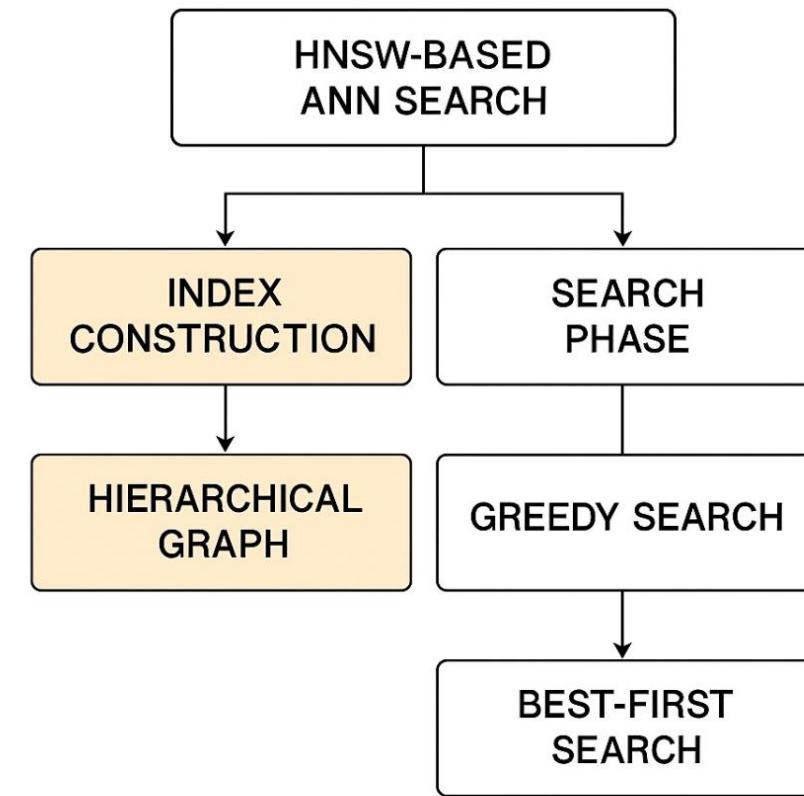
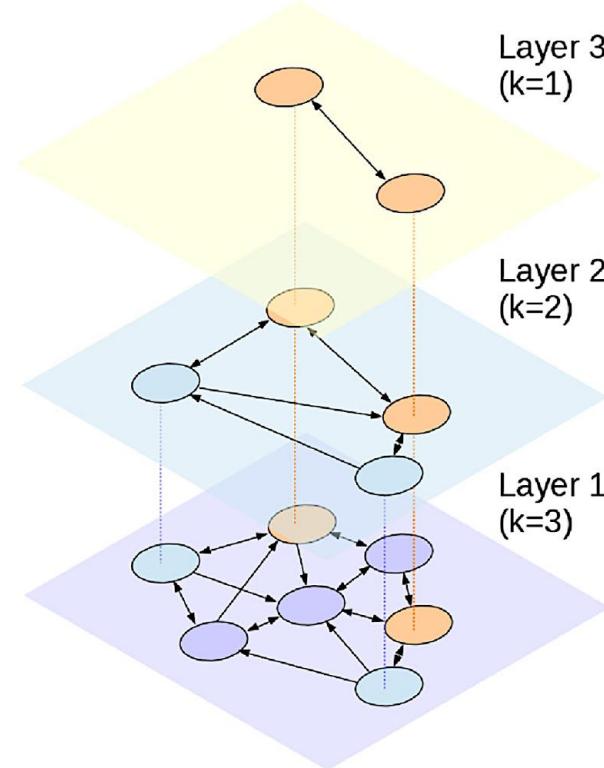


RA-LLM Learning: Indexing

- Vector indexes **speed up** queries by **minimizing the number of comparisons**. This is achieved by **partitioning** data so that only a small subset is compared.
 - **Tables** divide data into buckets containing similar vectors.
 - **Trees** are a nesting of tables.
 - **Graphs** connect similar vectors with virtual edges that can then be traversed.

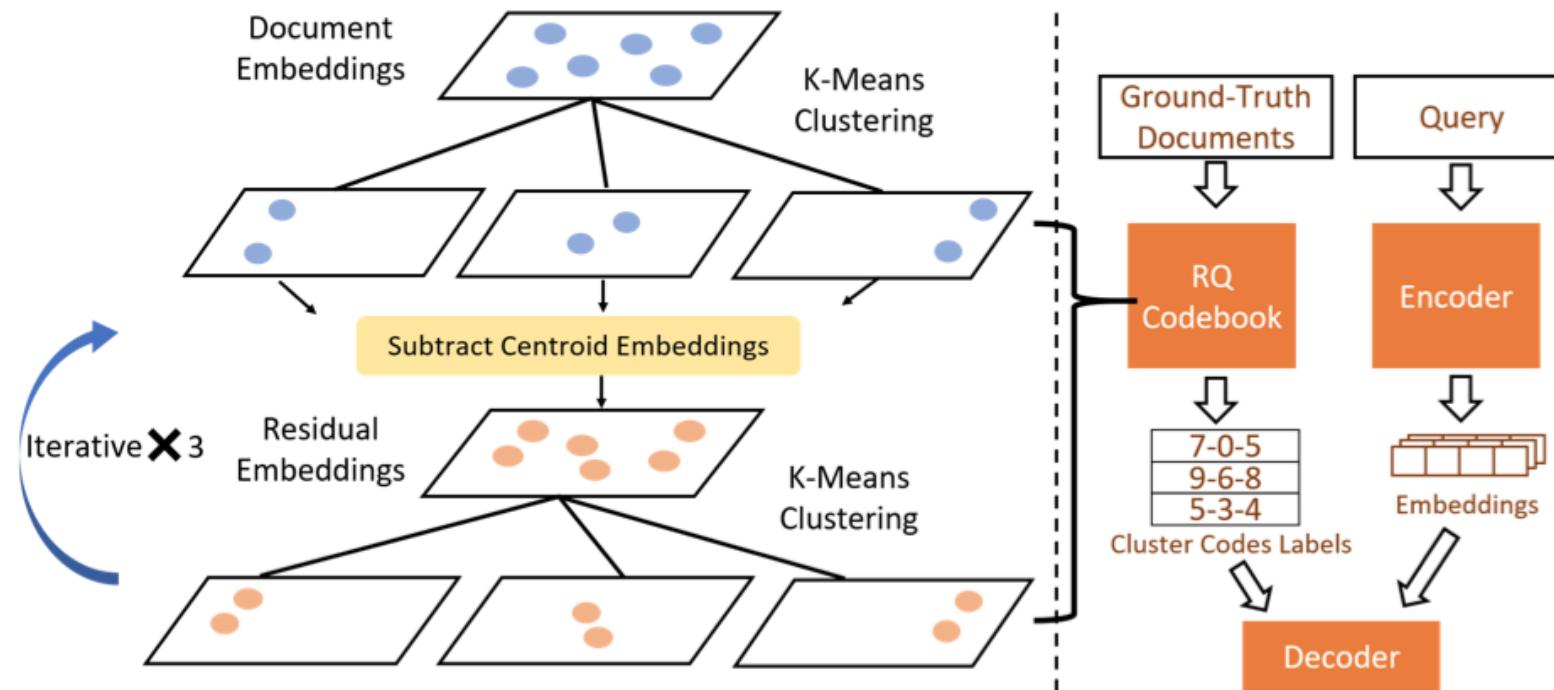
RA-LLM Learning: Indexing

- Hierarchical NSW
 - A **multi-layer proximity graph** with **probabilistic layer assignment**



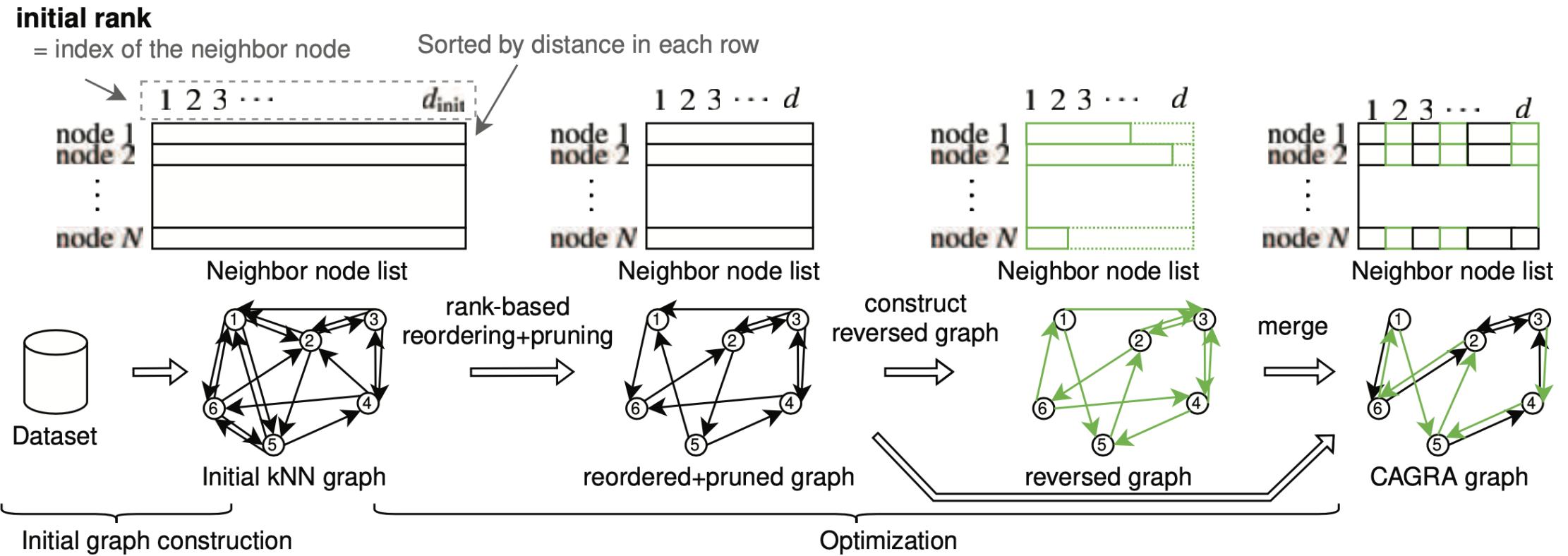
RA-LLM Learning: Indexing

- MEVI
 - Residual quantization with **hierarchical clustering**



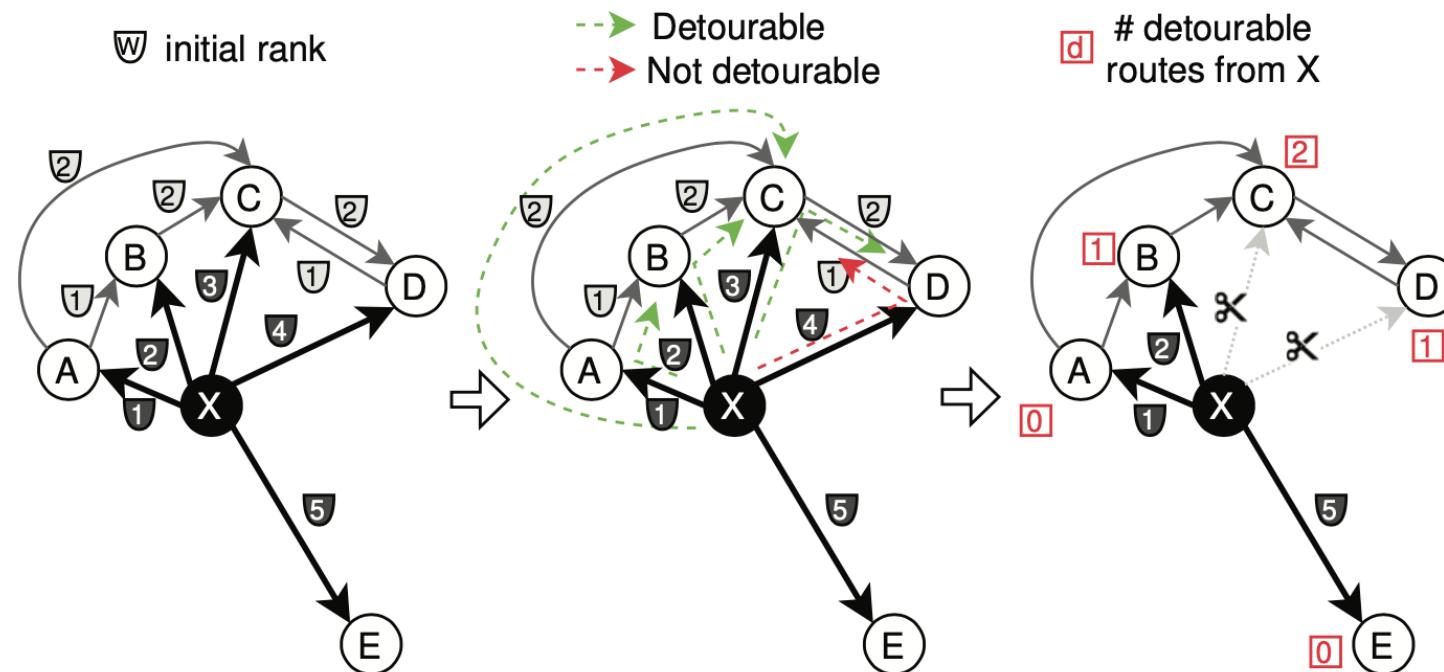
RA-LLM Learning: Indexing

- CAGRA
 - Rank-based edge reordering and reversed edge optimization



RA-LLM Learning: Indexing

- CAGRA



$$(e_{X \rightarrow Z}, e_{Z \rightarrow Y}) \text{ s.t. } \max(w_{X \rightarrow Z}, w_{Z \rightarrow Y}) < w_{X \rightarrow Y},$$

Part 3: RA-LLM Data Management

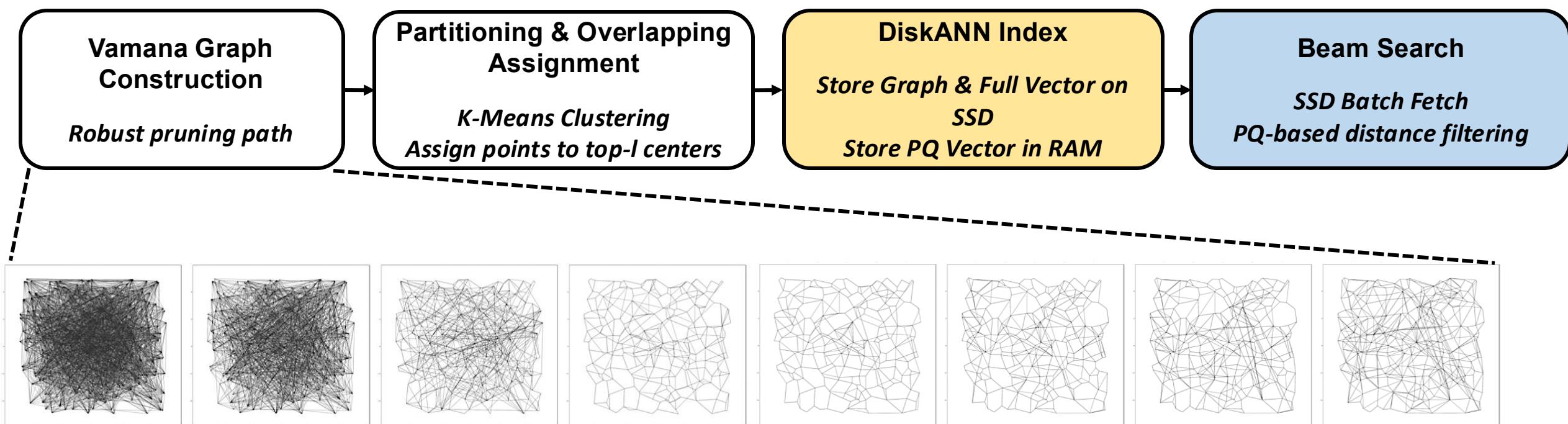


Website of this tutorial

- Vector Generation
- Indexing
- Storage
- Query Processing
- RAG-Oriented VDB Pipeline

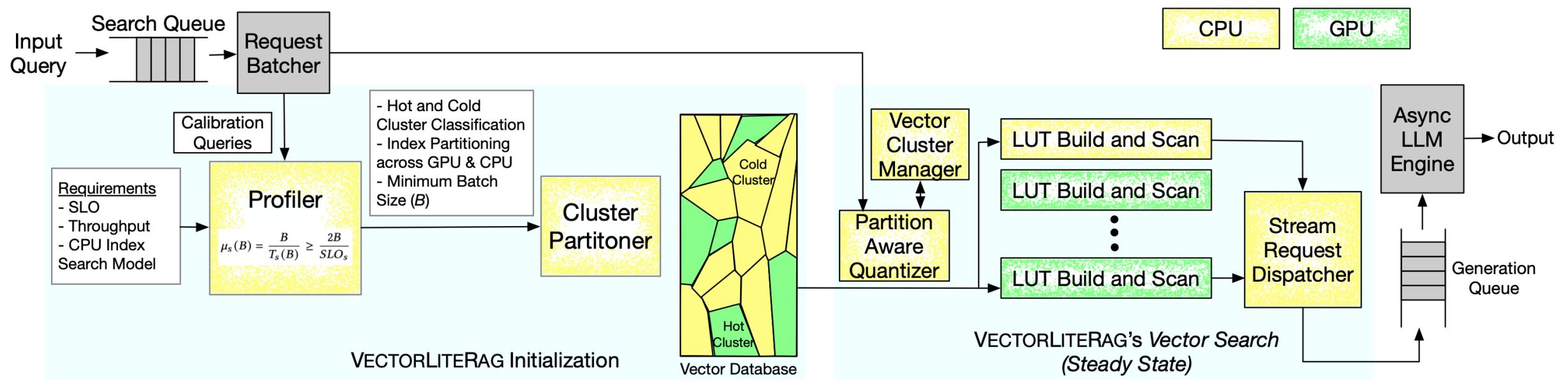
RA-LLM Learning: Storage

- DiskANN
 - Uses a **Disk-Memory mixed structure** to accelerate query search.



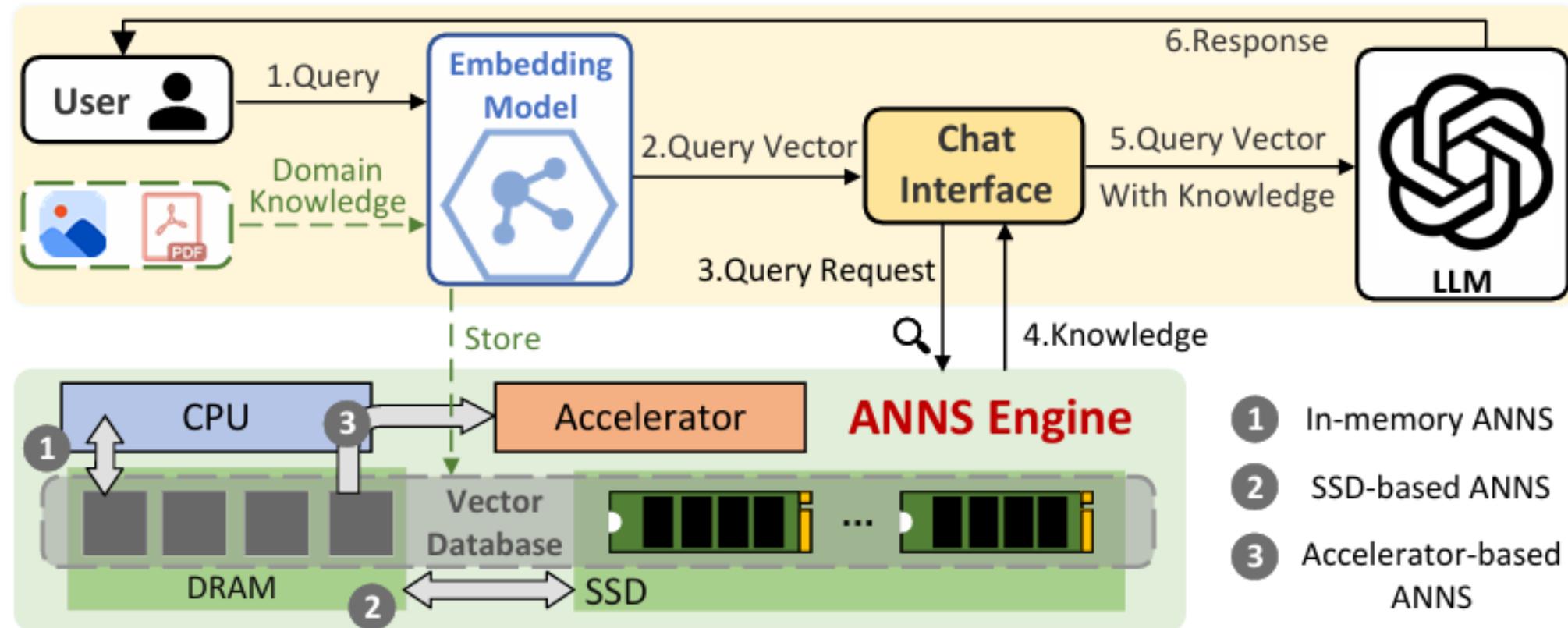
RA-LLM Learning: Storage

- **VectorLiteRAG**
 - Uses a **GPU-CPU mixed storage structure** to accelerate query search.



RA-LLM Learning: Storage

- Fusion-ANNS
 - A GPU-CPU-SSD optimized structure to accelerate query search.



Part 3: RA-LLM Data Management



Website of this tutorial

- Vector Generation
- Indexing
- **Query Processing**
- RAG-Oriented VDB Pipeline

Part 3: RA-LLM Data Management



Website of this tutorial

- Vector Generation
- Indexing
- **Query Processing**
 - **Similarity Calculation**
- RAG-Oriented VDB Pipeline

RA-LLM Learning: Query Processing

- **General Similarity Scores**

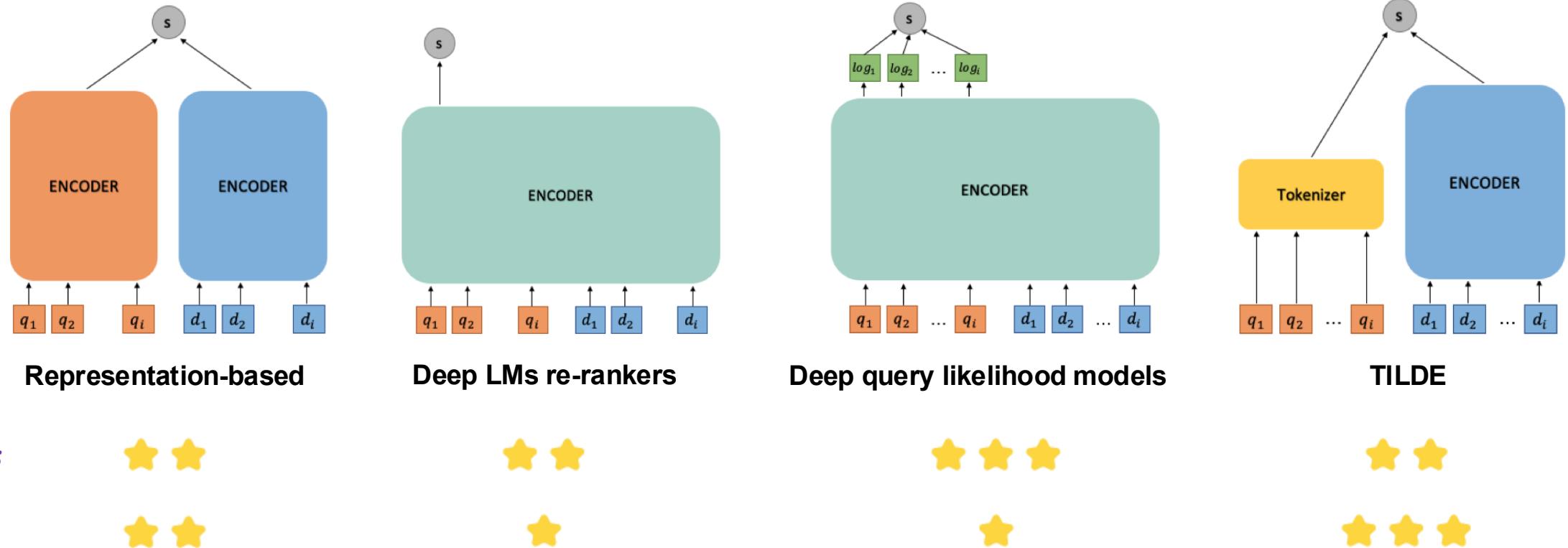
- The query processor mainly deals with how to specify the query criteria and how to execute search queries.

Table 1 Common similarity scores

Type	Score	Metric	Complexity	Range
Sim	Inner Prod	✗	$O(D)$	\mathbb{R}
	Cosine	✗	$O(D)$	$[-1, 1]$
Dist	Minkowski	✓	$O(D)$	\mathbb{R}^+
	Hamming	✓	$O(D)$	\mathbb{N}

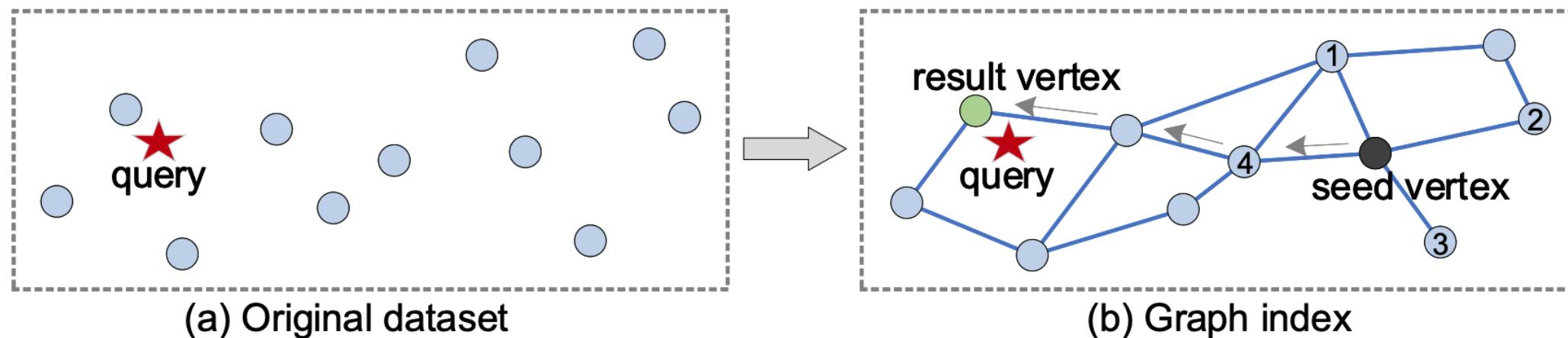
RA-LLM Learning: Query Processing

- Existing Query Processing Paradigm



RA-LLM Learning: Query Processing

- **Query Processing with Graph-Based ANN Search.**
 - Navigates the graph index from a **seed vertex** until the **closest result vertex** is identified.



Part 3: RA-LLM Data Management

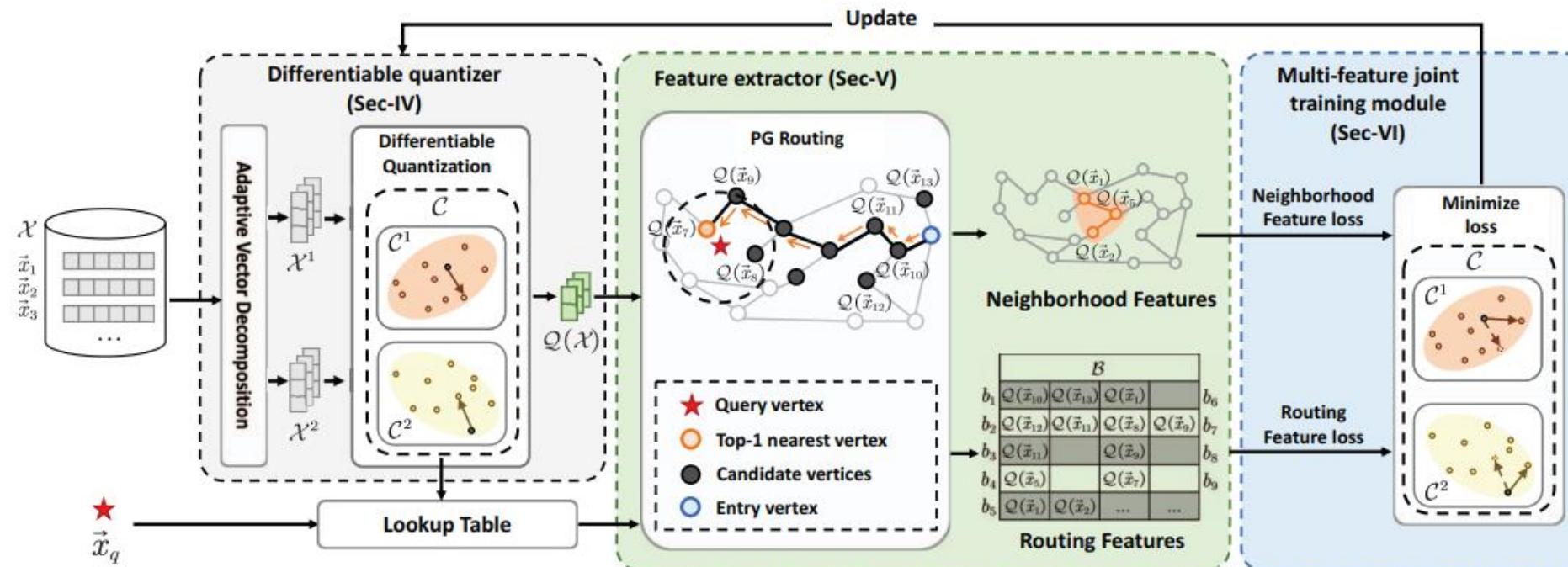


Website of this tutorial

- Vector Generation
- Indexing
- **Query Processing**
 - Acceleration
- RAG-Oriented VDB Pipeline

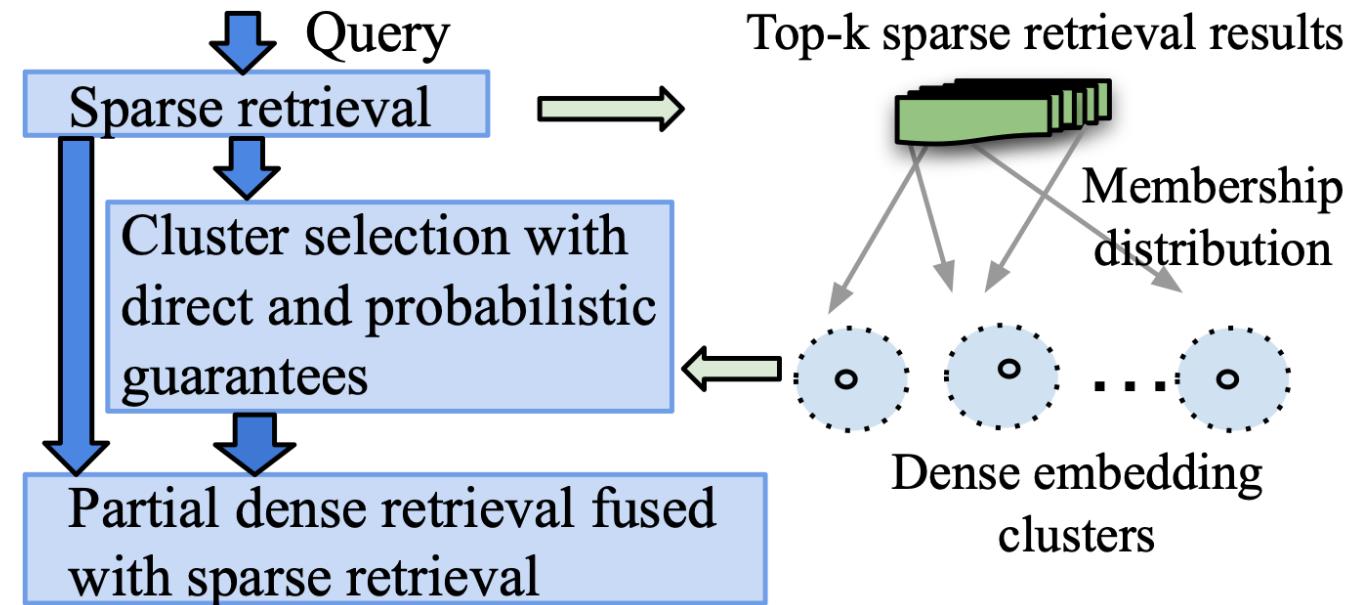
RA-LLM Learning: Query Processing

- Routing-guided learned Product Quantization (RPQ)
 - Integrates **routing-guided** learned product quantization with adaptive vector decomposition.



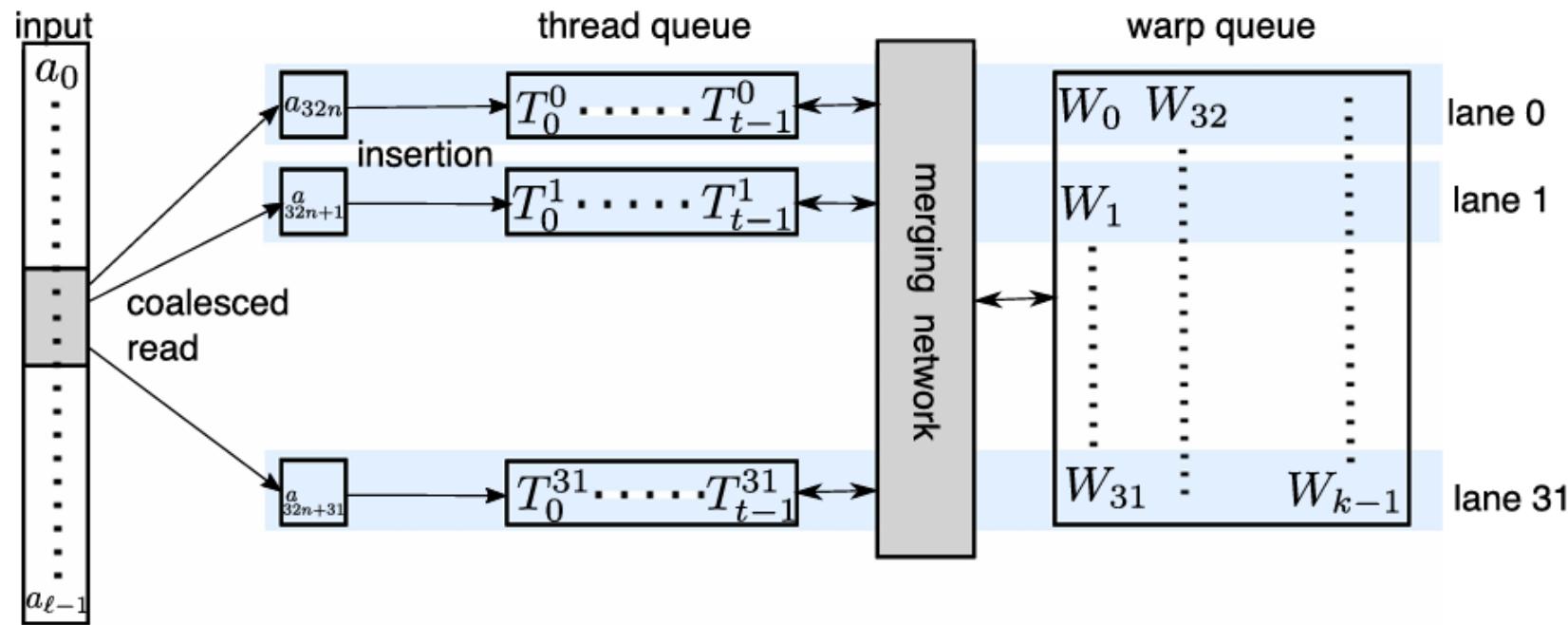
RA-LLM Learning: Query Processing

- CDFS
 - Combines **sparse retrieval** with cluster-guided **partial dense retrieval**



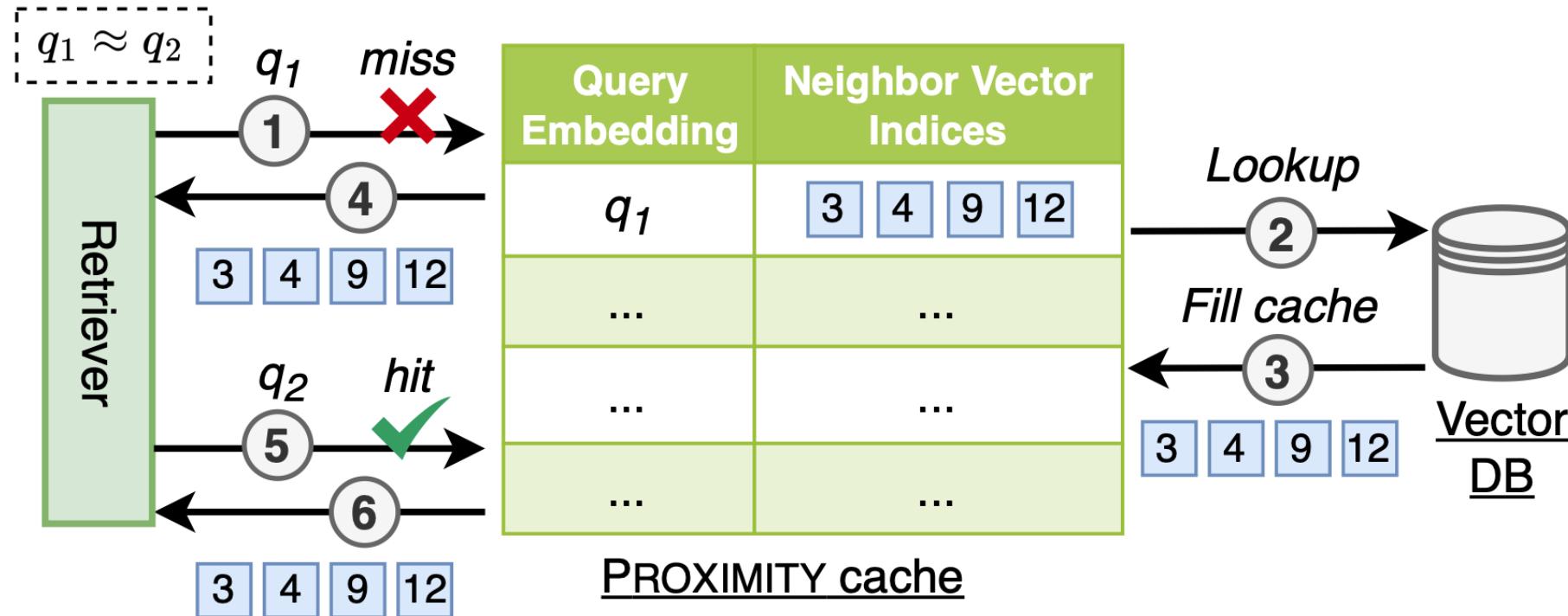
RA-LLM Learning: Query Processing

- WarpSelect
 - A **GPU-optimized** high-performance top-k selection algorithm.



RA-LLM Learning: Query Processing

- Proximity
 - Reusing similar query results without repeated database lookups.



Part 3: RA-LLM Data Management

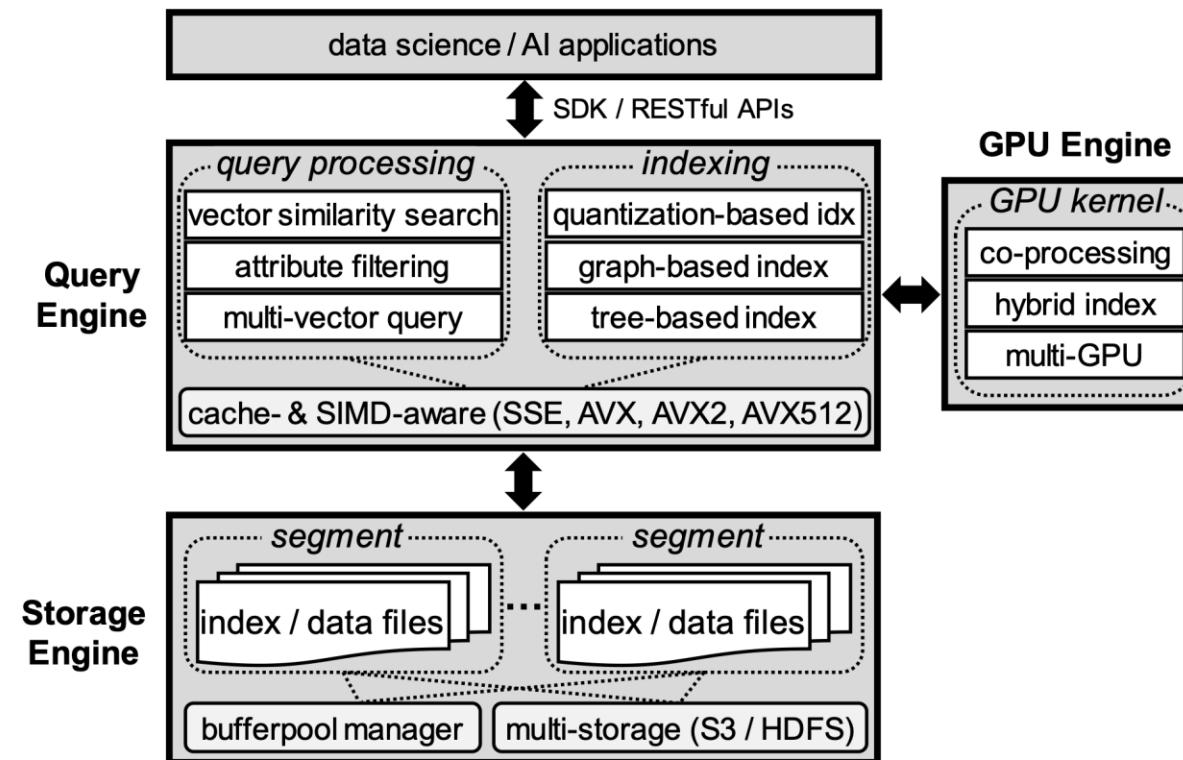


Website of this tutorial

- Vector Generation
- Indexing
- Query Processing
- **RAG-Oriented VDB Pipeline**

RA-LLM Learning: RAG-Oriented VDB Pipeline

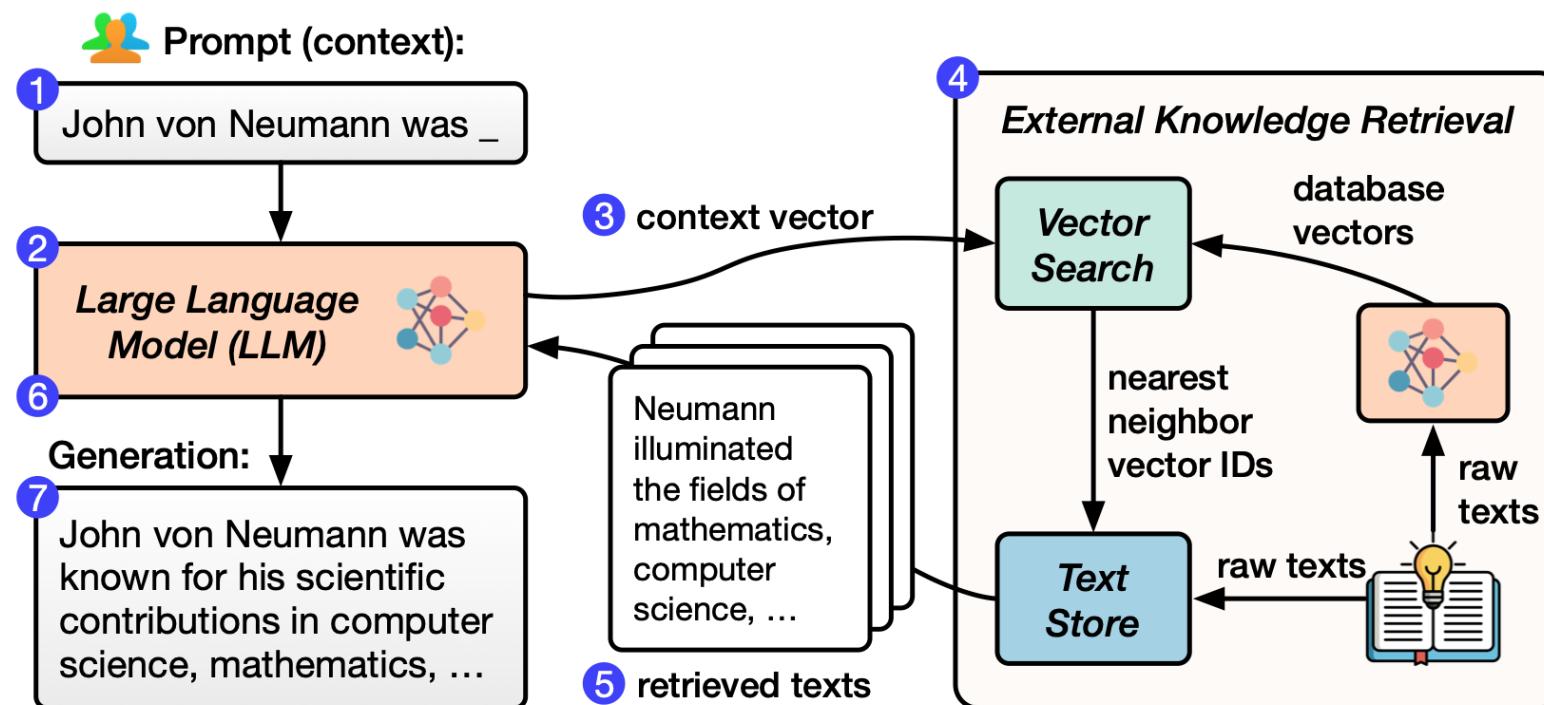
- **Milvus**
 - Leverages a hybrid GPU-CPU engine with SIMD-optimized query processing and multi-storage for scalability.



RA-LLM Learning: RAG-Oriented VDB Pipeline

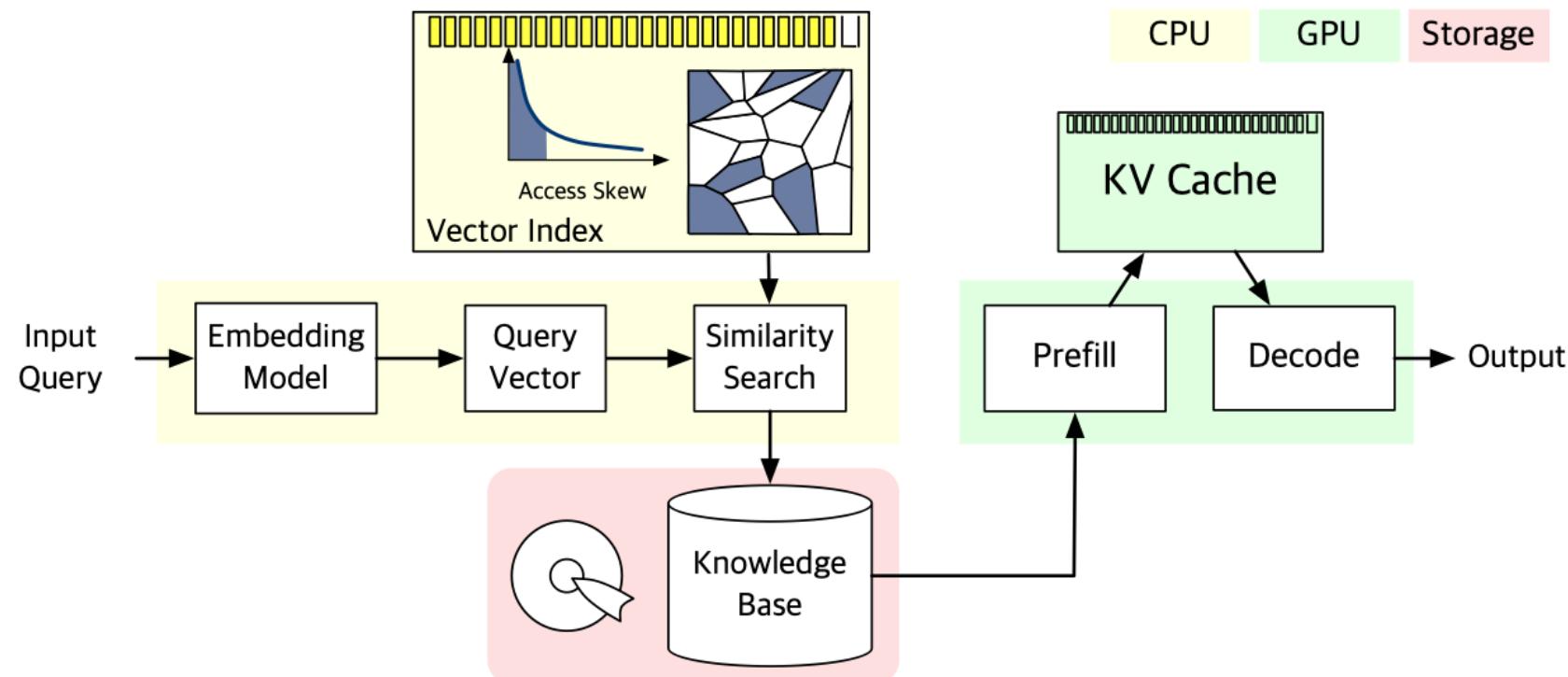
- **Chameleon**

- **FPGA-based near-memory accelerators** for vector search.
- Disaggregated architecture for LLM inference.



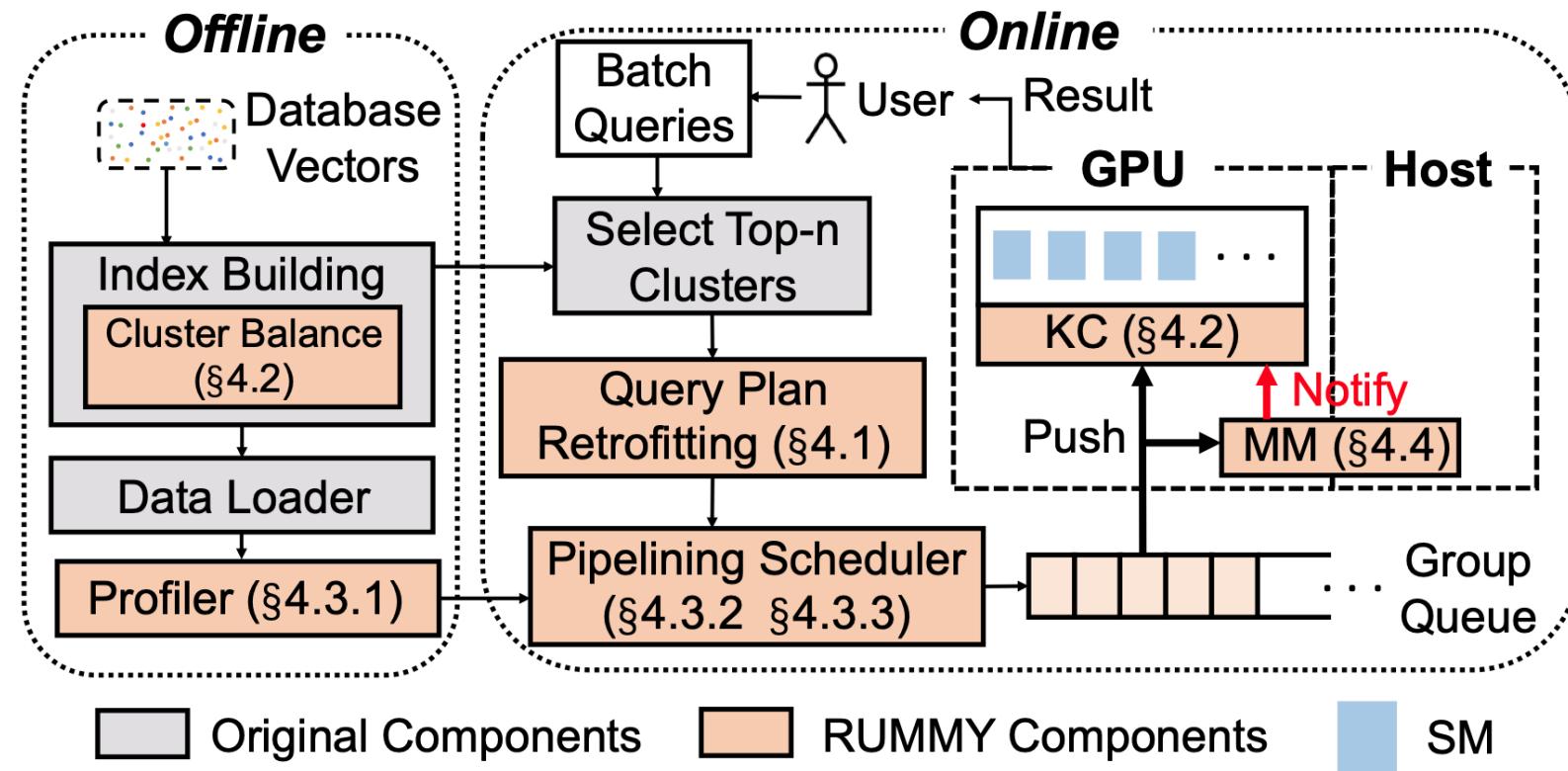
RA-LLM Learning: RAG-Oriented VDB Pipeline

- **VectorLiteRAG**
 - Adaptive vector index partitioning to balance vector search between CPU and GPU



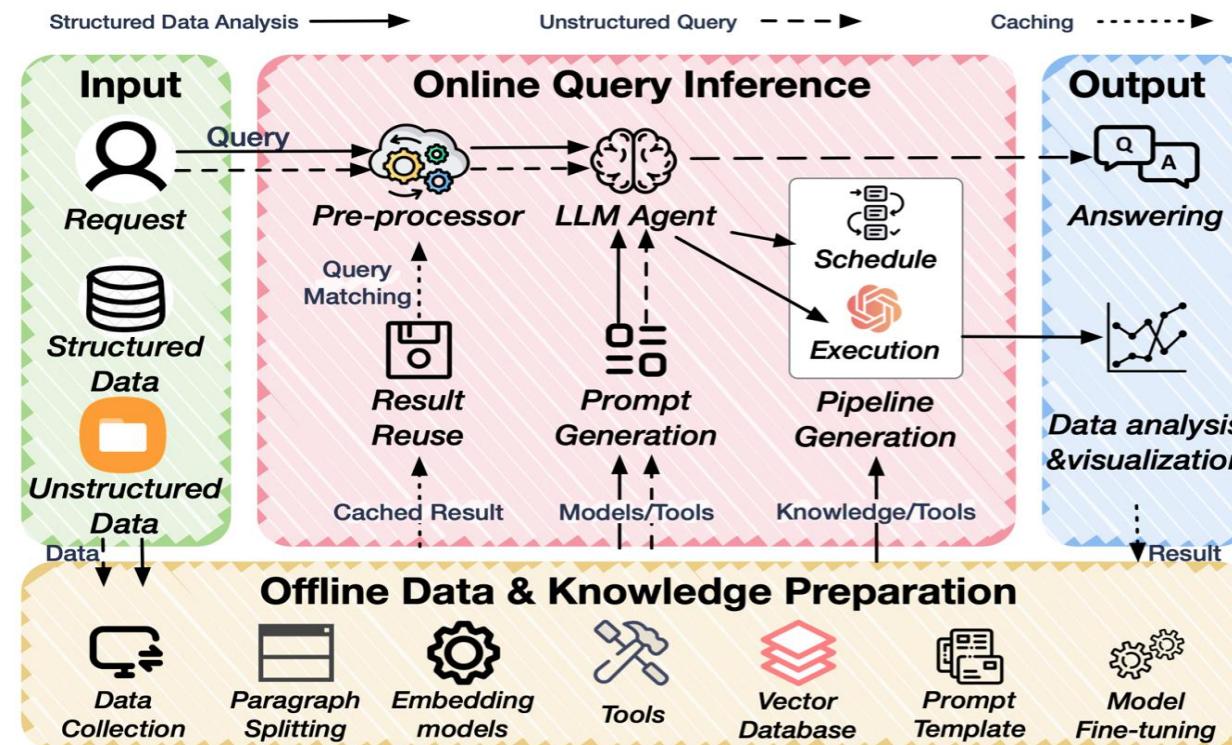
RA-LLM Learning: RAG-Oriented VDB Pipeline

- Rummy
 - Using cluster-balanced indexing, query plan retrofitting, and pipelined scheduling.



RA-LLM Learning: RAG-Oriented VDB Pipeline

- Chat2Data
 - A multi-stage pipeline with adaptive query preprocessing, LLM-driven pipeline generation, and caching.





Tutorial Outline

- **Part 1: Introduction** of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)
- **Part 2: Architecture** of RA-LLMs and **Main Modules** (Dr. Yujuan Ding)
- **Part 3: Data Management** for RA-LLMs (Pangjing Wu)
- **Part 4: Learning Approach of RA-LLMs (Liangbo Ning)**
- **Part 4: Applications** of RA-LLMs (Shijie Wang)
- **Part 5: Challenges and Future Directions** of RA-LLMs (Dr. Wenqi Fan)

Website of this tutorial
Check out the slides and more information!



Part 3: RA-LLM Learning



- **Training-free Methods**
- **Training-based Methods**
 - **Independent Learning**
 - **Sequential Learning**
 - **Joint Learning**

Part 3: RA-LLM Learning

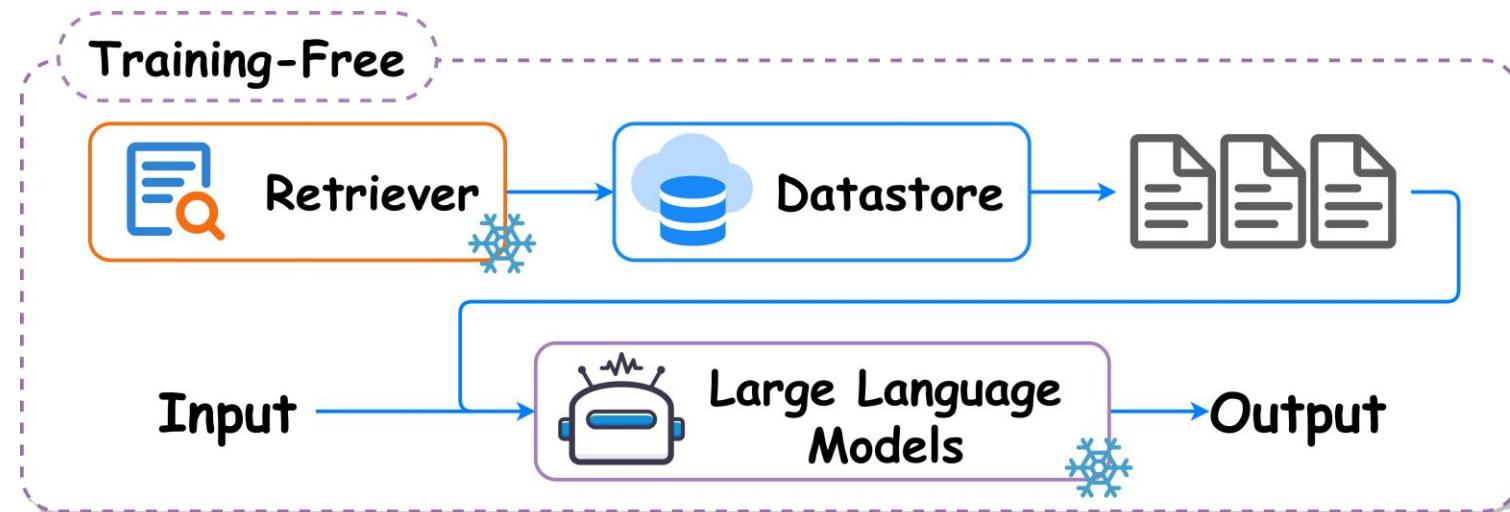


Presenter
Liangbo Ning
HK PolyU

- **Training-free Methods**
- **Training-based Methods**
 - Independent Learning
 - Sequential Learning
 - Joint Learning

RA-LLM Learning: Training-free

- Retrieval models and language models are both frozen.

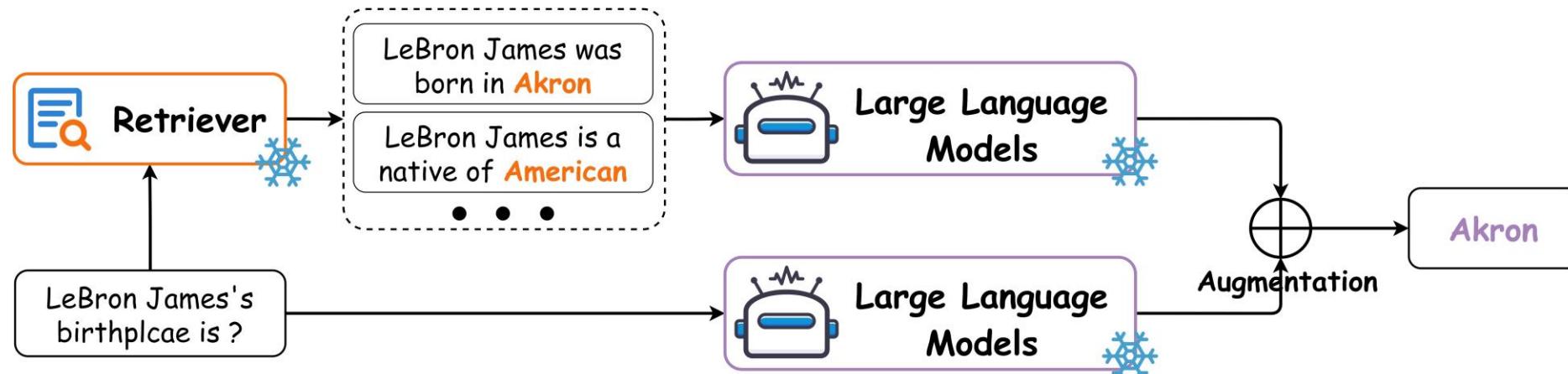


RA-LLM Learning: Training-free

- **Prompt Engineering-based Methods**

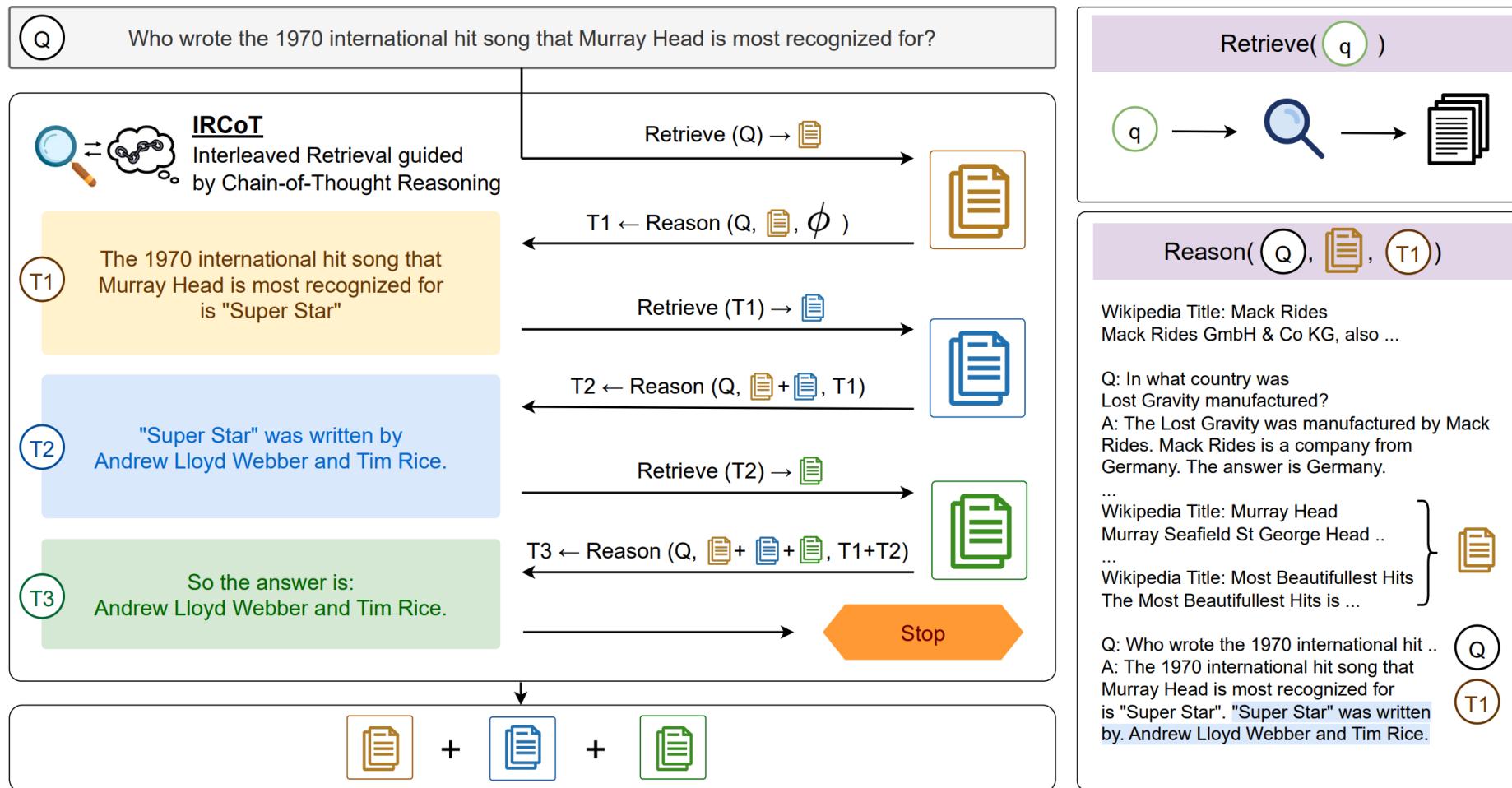


- **Retrieval-Guided Token Generation Methods**



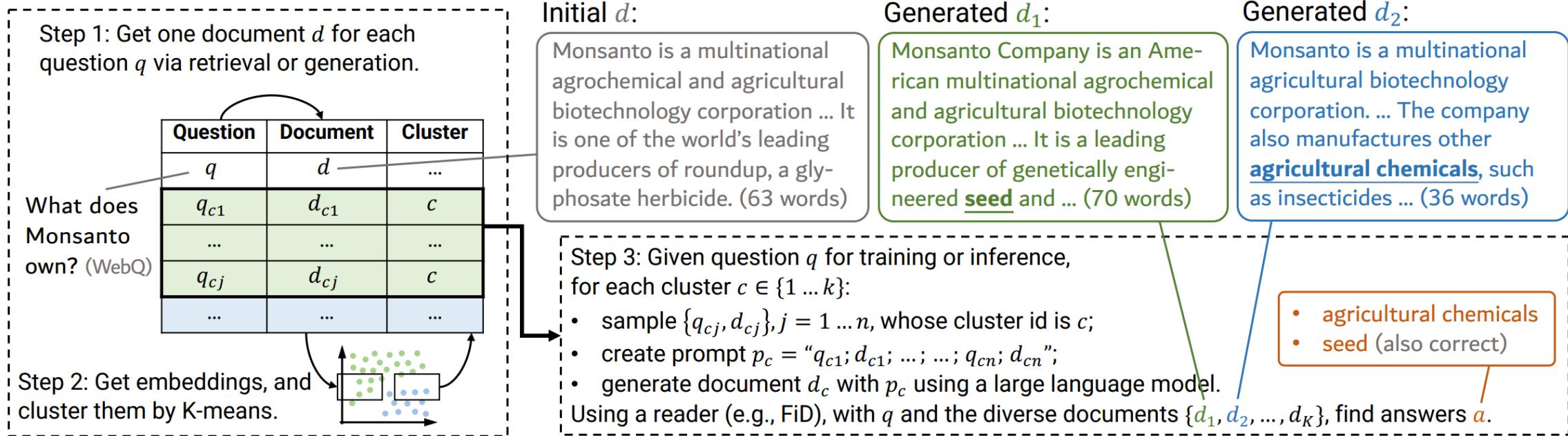
RA-LLM Learning: Training-free

- **IRCoT**



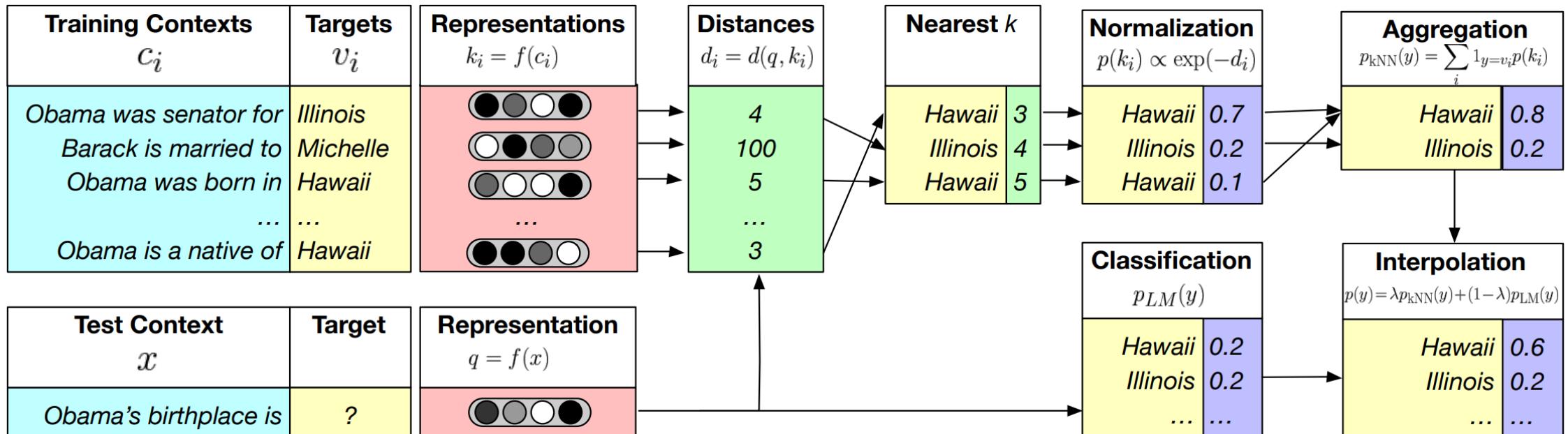
RA-LLM Learning: Training-free

- GENREAD



RA-LLM Learning: Training-free

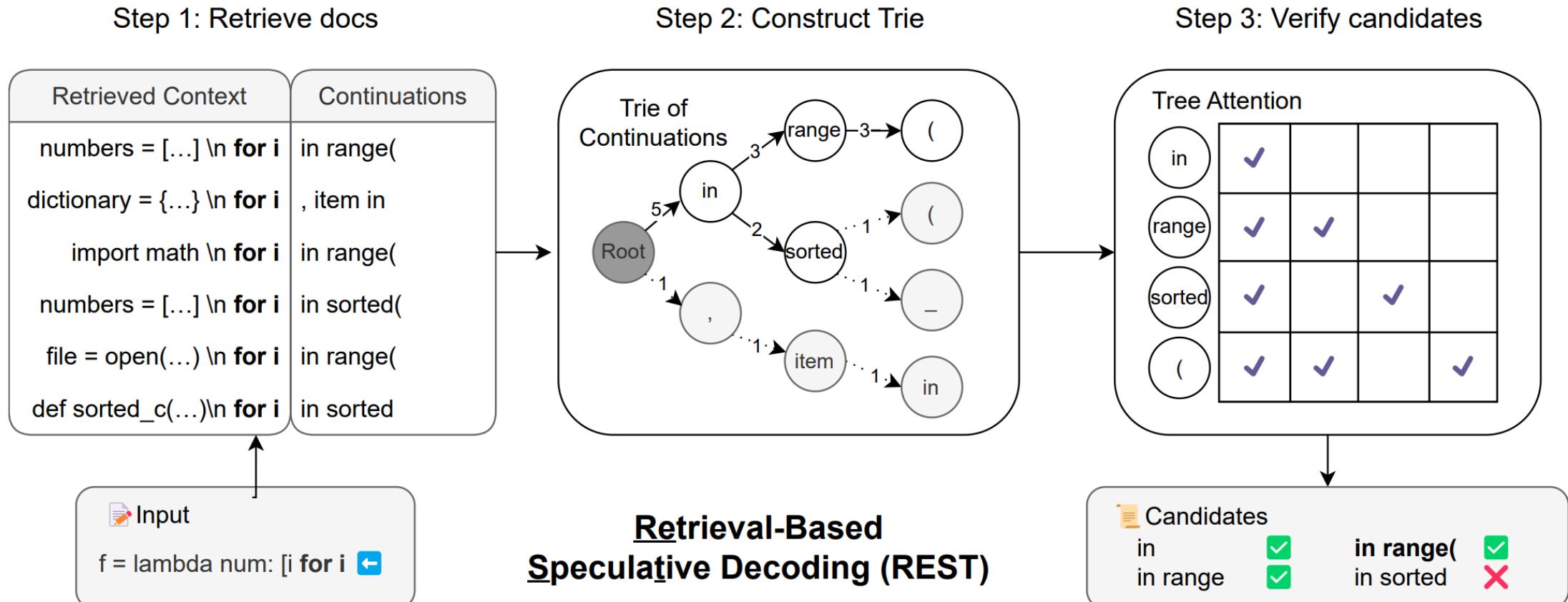
- **k NN-LM**



$$p(y|x) = \lambda p_{kNN}(y|x) + (1 - \lambda) p_{LM}(y|x)$$

RA-LLM Learning: Training-free

- REST



RA-LLM Learning: Training-free

- ✓ Work with off-the-shelf models
- ✗ All components are fixed and not trained
- ✗ Might not achieve optimal learning result of the whole model

Part 3: RA-LLM Learning

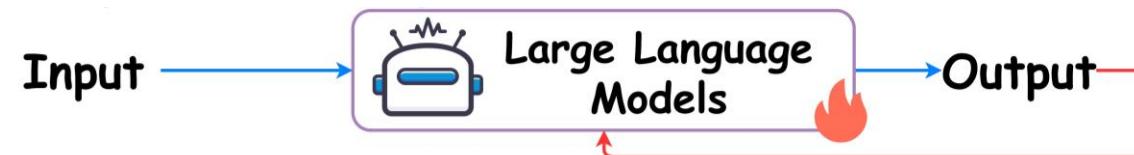


Website of this tutorial

- Training-free Methods
- Training-based Methods
 - Independent Learning
 - Sequential Learning
 - Joint Learning

RA-LLM Learning: Independent Training

- **Retrieval models** and **language models** are trained **independently**.
 - Independent training of large language models.



- Independent training of Retriever.



RA-LLM Learning: Independent Training

- **Retrieval models** and **language models** are trained **independently**.

- Independent training of large language models.

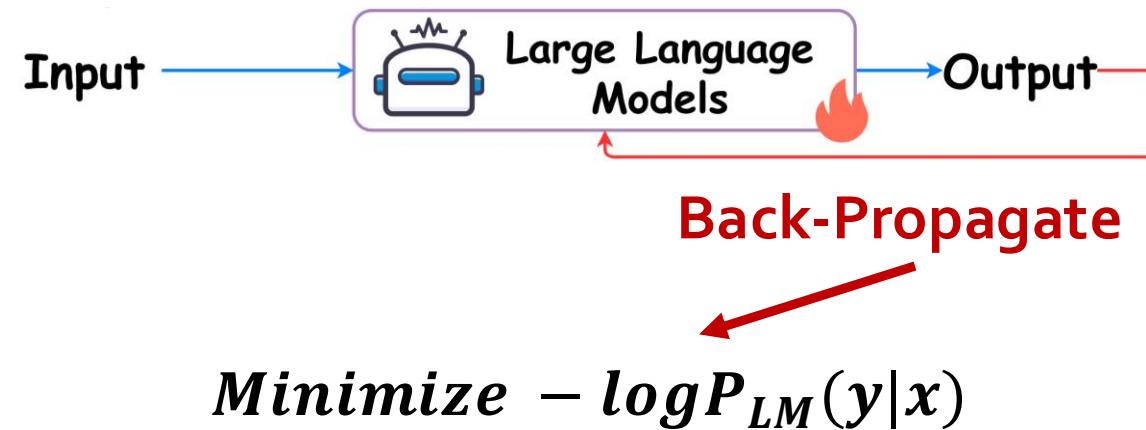


- Independent training of Retriever.



RA-LLM Learning: Independent Training

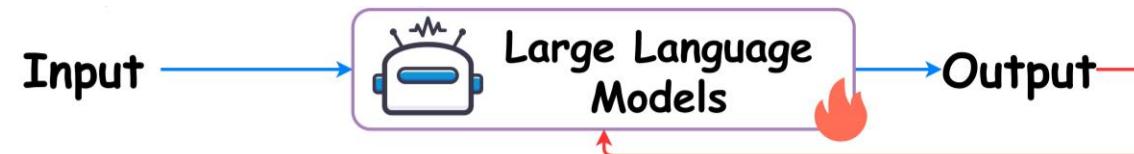
- Independent training of large language models.



.....

RA-LLM Learning: Independent Training

- **Retrieval models** and **language models** are trained **independently**.
 - Independent training of large language models.

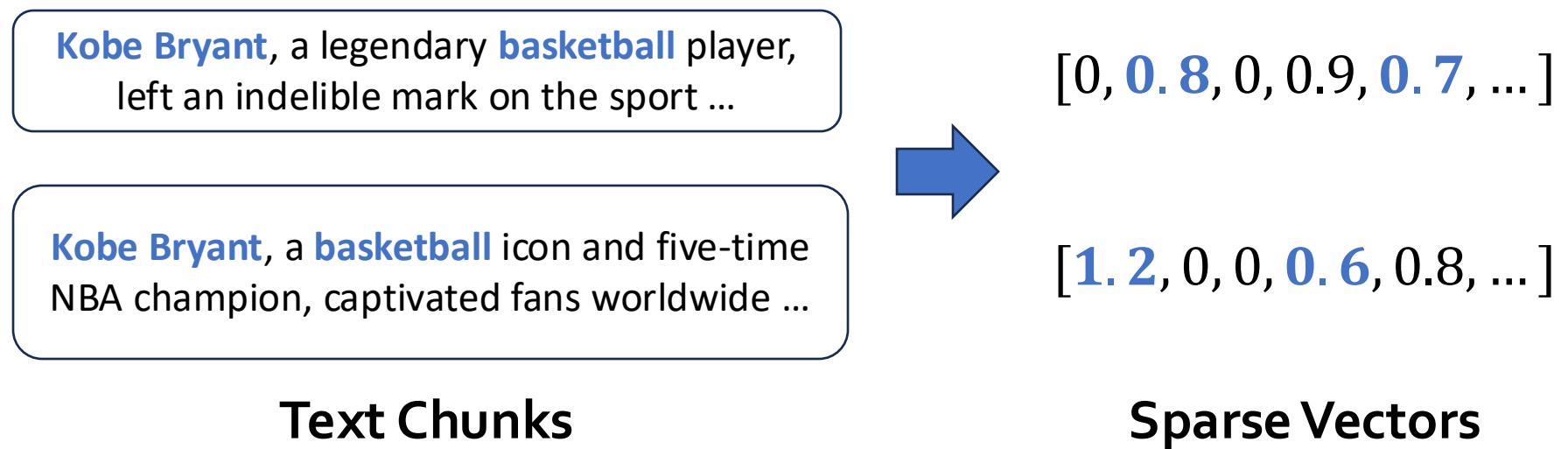


- Independent training of Retriever.



RA-LLM Learning: Independent Training

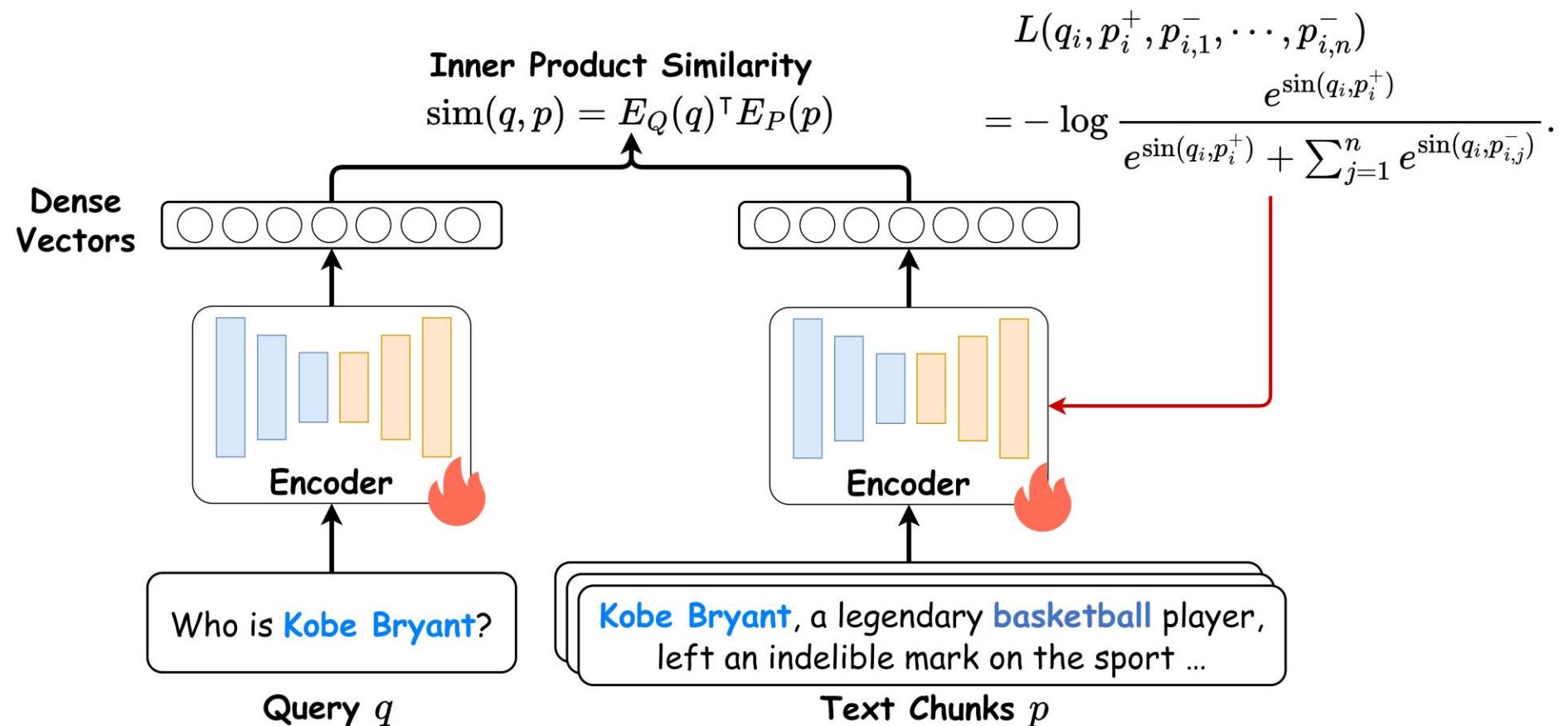
- Sparse retrieval models: TF-IDF / BM25



No training is Needed!

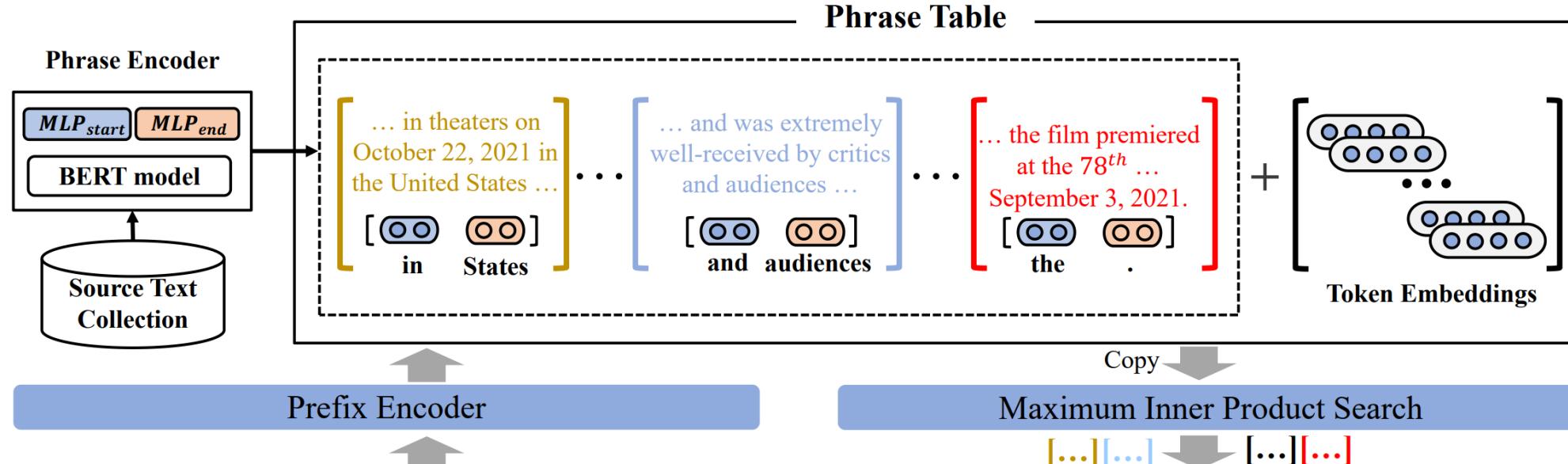
RA-LLM Learning: Independent Training

- Dense retrieval models: DPR



RA-LLM Learning: Independent Training

- Dense retrieval models: CoG



The Dune film was released [in theaters on October 22, 2021 in the United States] [and was extremely well-received by critics
and audiences] [Before] [that] [,] [the film premiered at the 78th International Film Festival on September 3, 2021.]

$$\mathcal{H}_{i+1} = \text{PrefixEncoder}(x_i, \mathcal{H}_i).$$

$$\mathcal{D}_{\text{start}} = \text{MLP}_{\text{start}}(\mathcal{D}), \mathcal{D}_{\text{end}} = \text{MLP}_{\text{end}}(\mathcal{D}).$$

$$\text{PhraseEncoder}(s, e, D) = [\mathcal{D}_{\text{start}}[s]; \mathcal{D}_{\text{end}}[e]] \in \mathbb{R}^d$$

RA-LLM Learning: Independent Training

- **Model Training:**

$$\mathcal{L}_p = -\frac{1}{n} \sum_{k=1}^n \log \frac{\exp(q_k \cdot p_k)}{\sum_{p \in \mathcal{P}_k} \exp(q_k \cdot p_p) + \sum_{w \in V} \exp(q_k \cdot v_w)}$$

$$\mathcal{L}_t = -\frac{1}{m} \sum_{i=1}^m \log \frac{\exp(q_i, v_{D_i})}{\sum_{w \in V} \exp(q_i, v_w)}$$

RA-LLM Learning: Independent Training

- ✓ Work with off-the-shelf models, flexible
- ✓ Each part can be improved independently
- ✗ Lack of integrity between Retrieval and Generation
- ✗ Retrieval models are not optimized specified for the tasks/ domains/ generators

Part 3: RA-LLM Learning

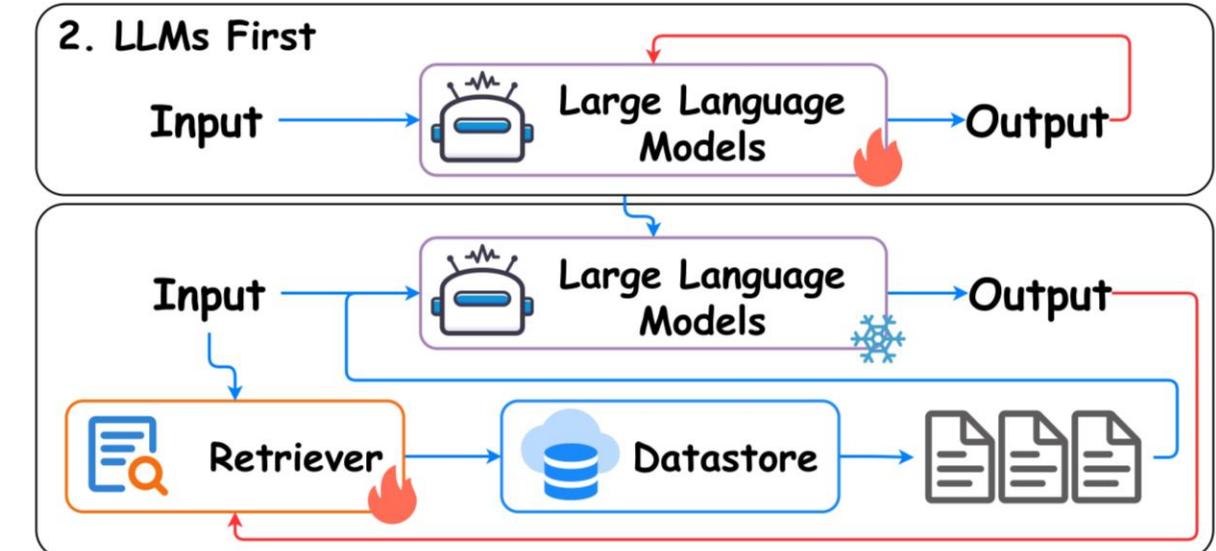
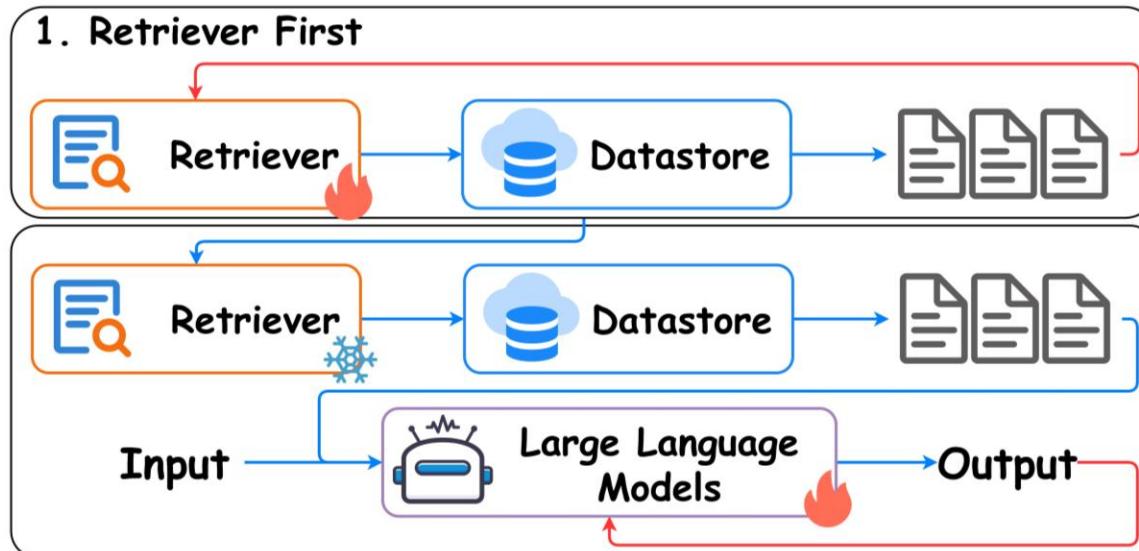


Website of this tutorial

- **Training-free Methods**
- **Training-based Methods**
 - Independent Learning
 - **Sequential Learning**
 - Joint Learning

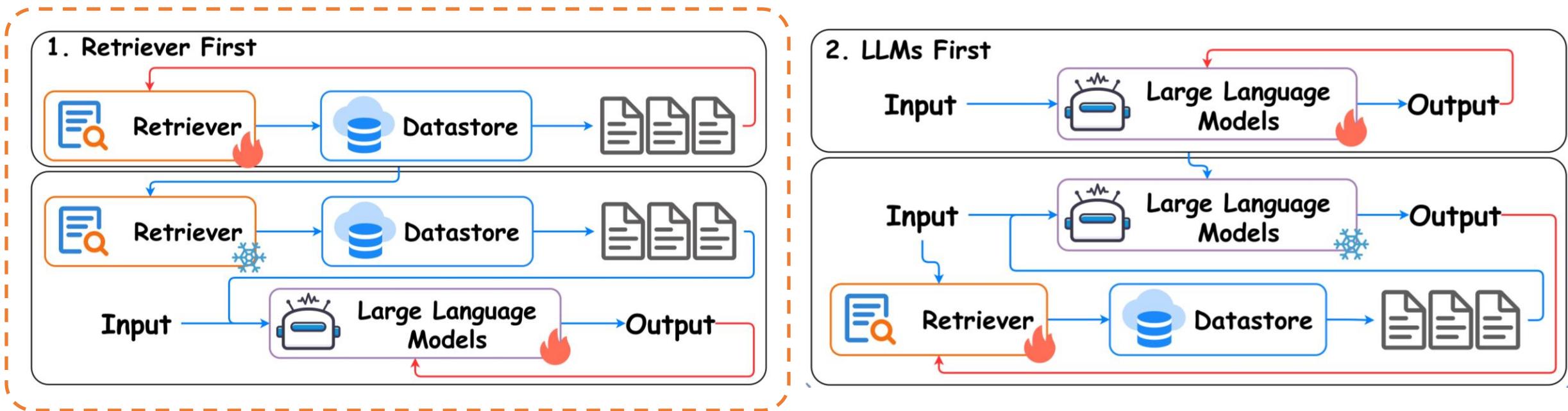
RA-LLM Learning: Sequential Training

- One component is first trained independently and then fixed.
- The other component is trained with an objective that depends on the first one.



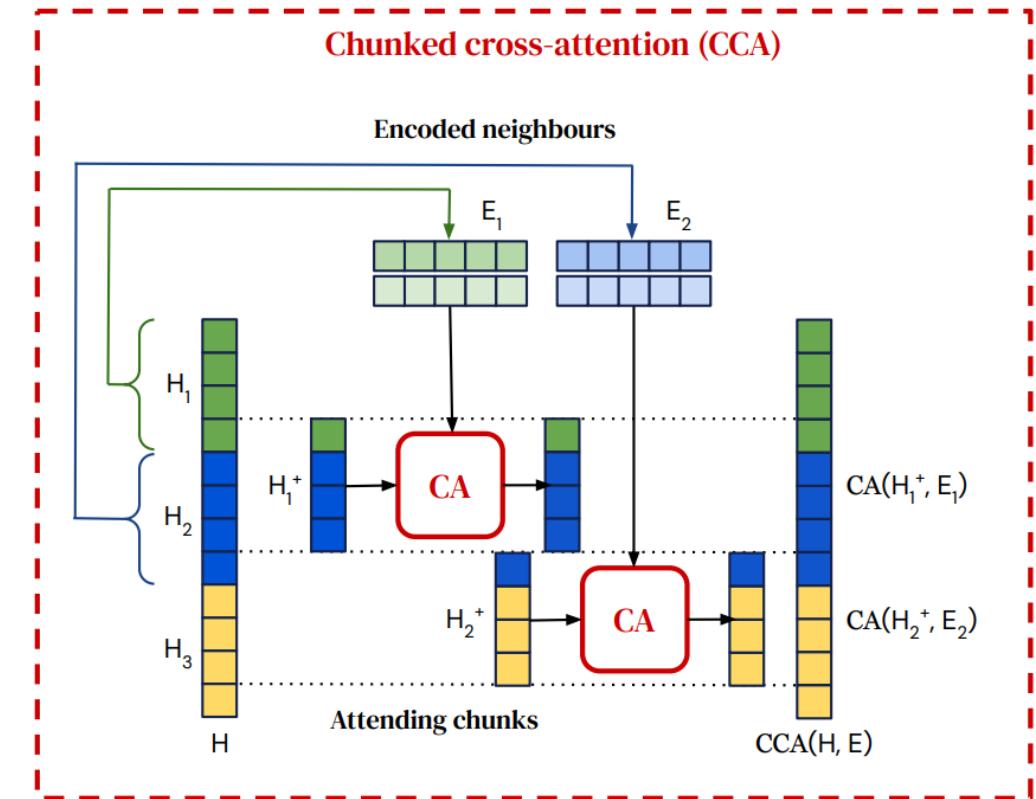
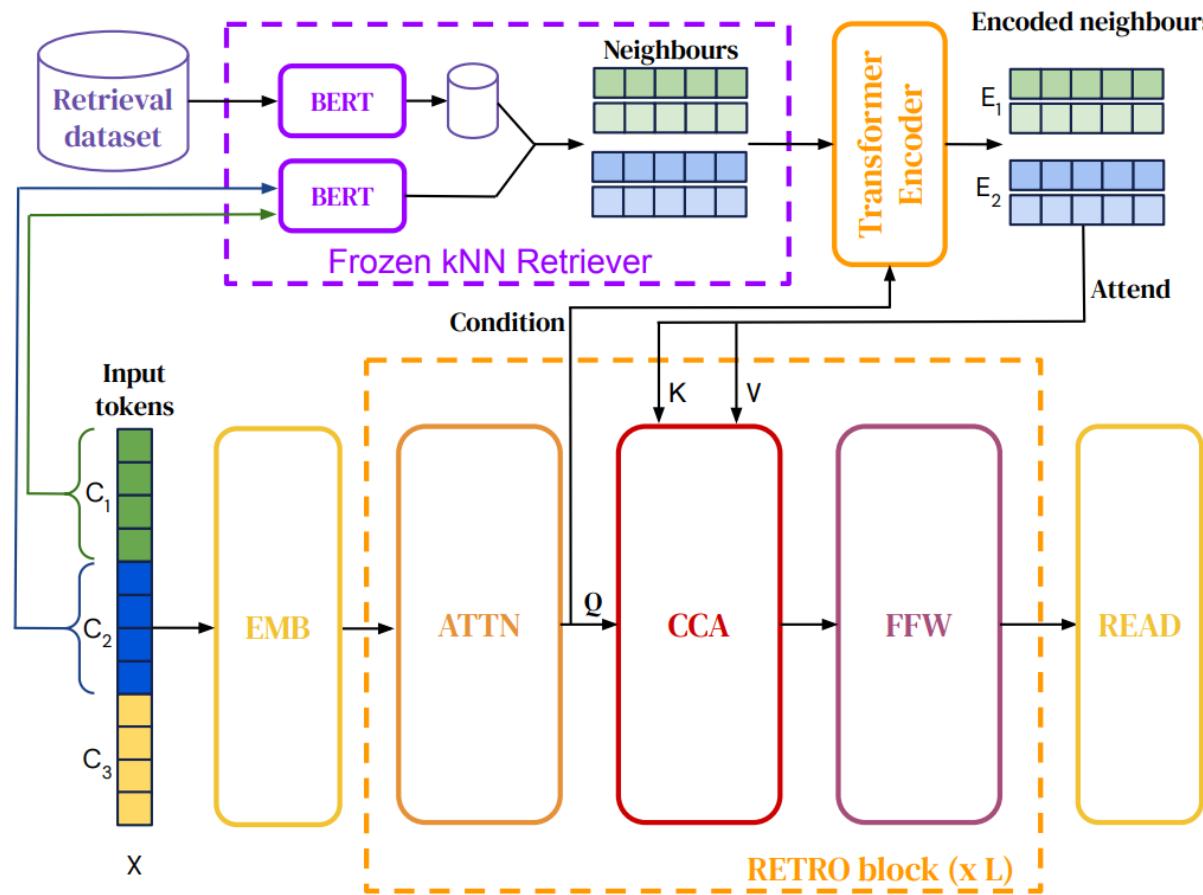
RA-LLM Learning: Sequential Training

- **Retrieval models** is first trained independently and then fixed.
- **Language models** are trained with an objective that depends on **the Retrieval**.



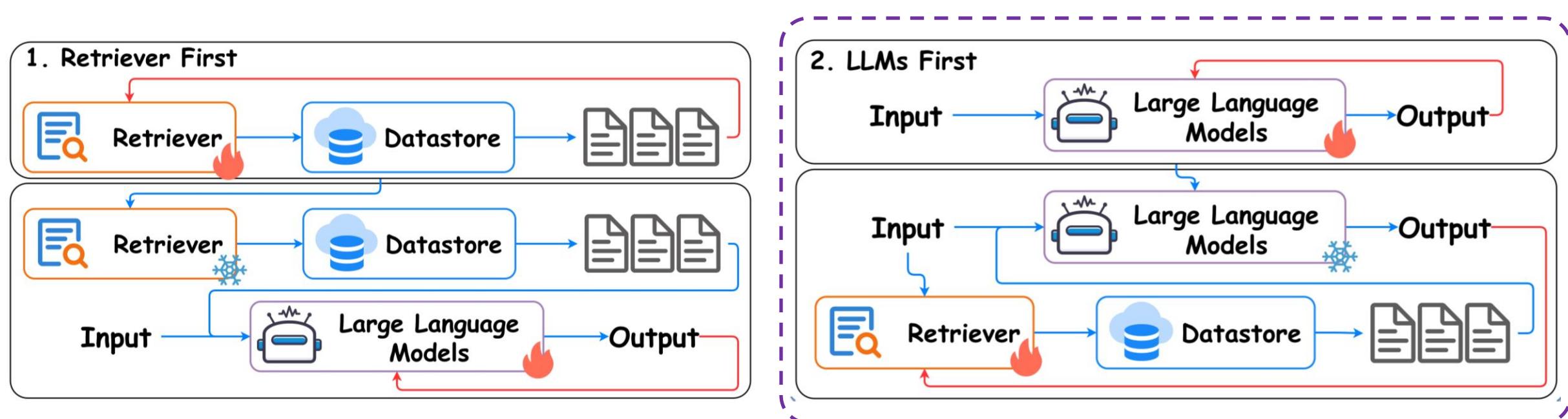
RA-LLM Learning: Sequential Training

- RETRO



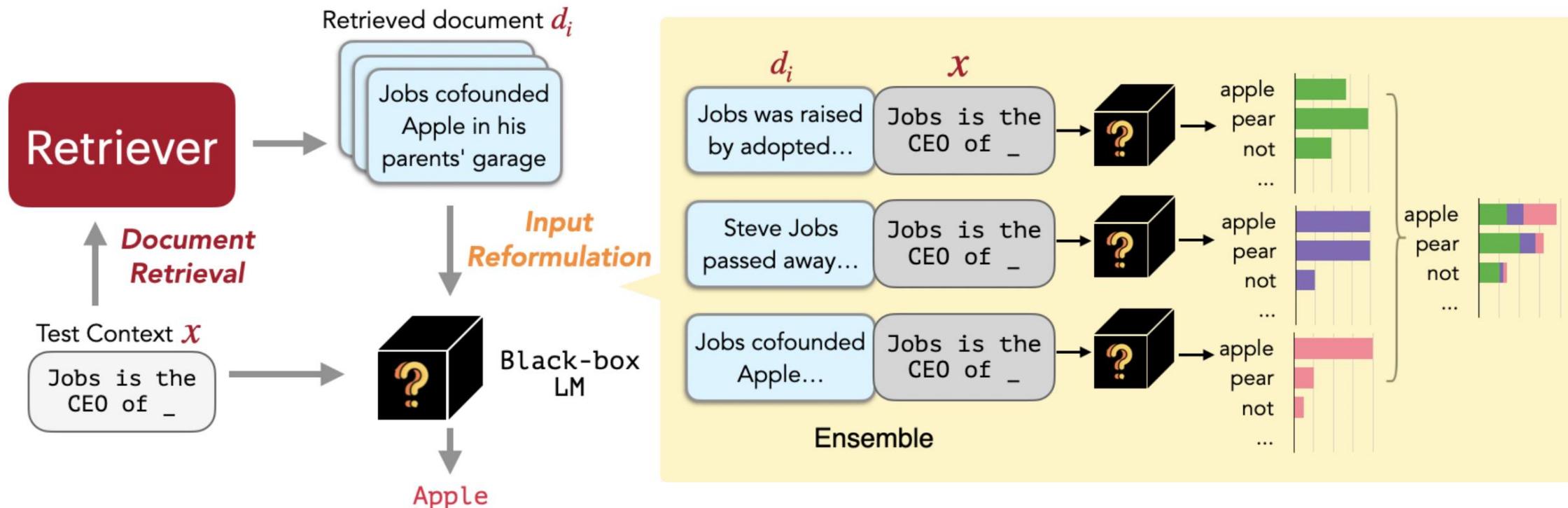
RA-LLM Learning: Sequential Training

- **Language models** are first trained independently and then fixed.
- **Retrieval models** are trained with supervisions from **language models**.



RA-LLM Learning: Sequential Training

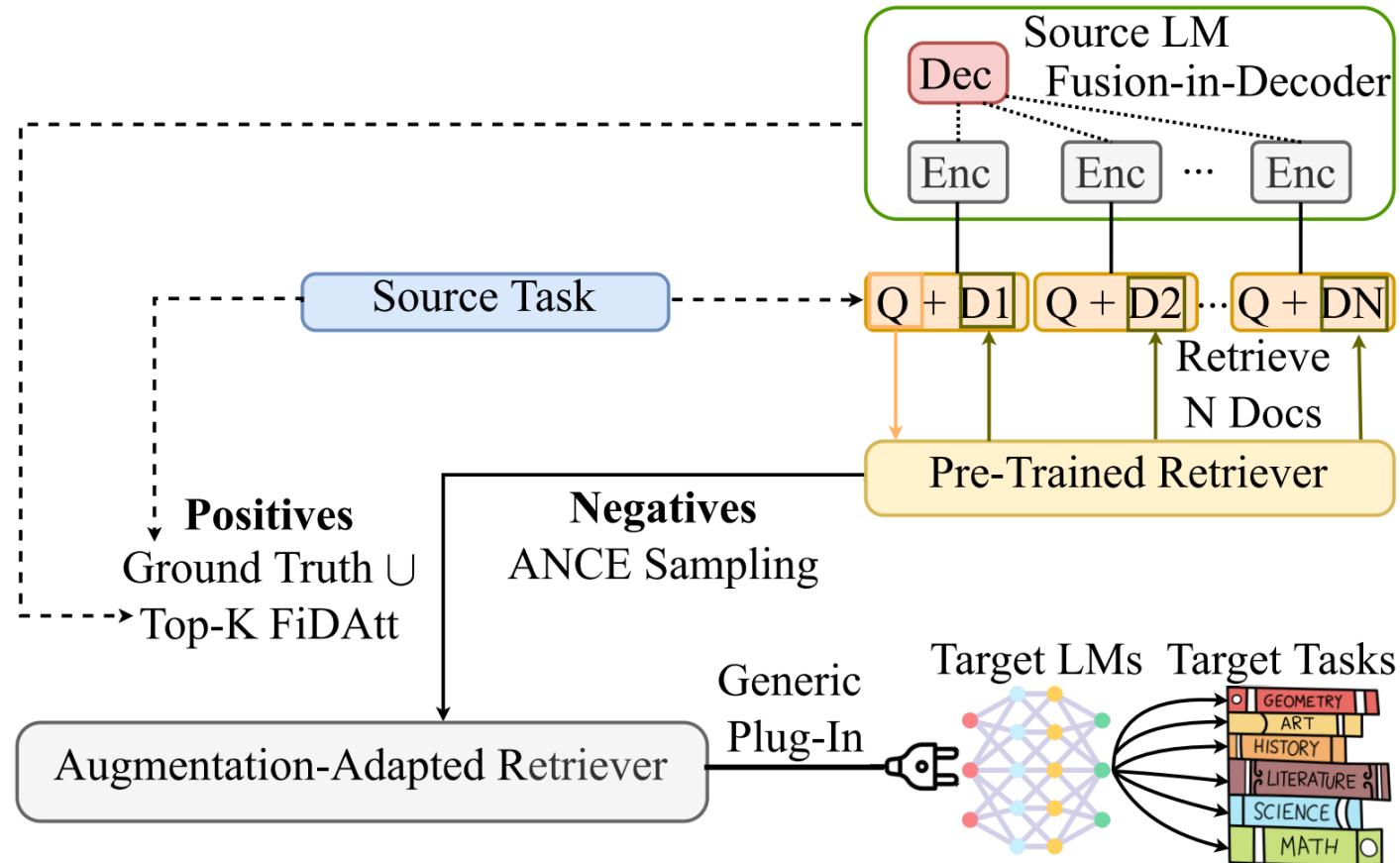
- REPLUG (Retrieve and Plug)



$$\mathcal{L} = \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} KL\left(P_R(d | x) \| Q_{LM}(d | x, y)\right) \quad P_R(d | x) = \frac{e^{s(d, x) / \gamma}}{\sum_{d \in \mathcal{D}'} e^{s(d, x) / \gamma}} \quad Q(d | x, y) = \frac{e^{P_{LM}(y | d, x) / \beta}}{\sum_{d \in \mathcal{D}'} e^{P_{LM}(y | d, x) / \beta}}$$

RA-LLM Learning: Sequential Training

- AAR (Augmentation-Adapted Retriever)



RA-LLM Learning: Sequential Training

- ✓ Work with off-the-shelf models
- ✓ Generators can be trained effectively based on the retrieved results
- ✓ Retrievers can be trained to provide useful information to help the generators
- ✗ One component is still fixed and not trained
- ✗ Might not achieve optimal learning result of the whole modell

Part 3: RA-LLM Learning

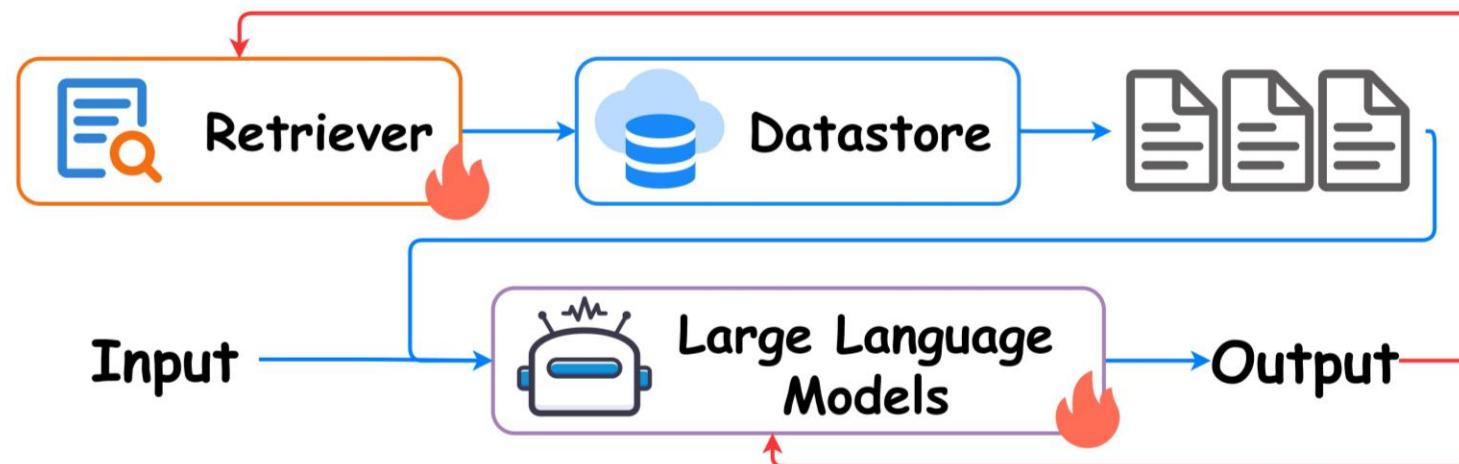


Website of this tutorial

- **Training-free Methods**
- **Training-based Methods**
 - Independent Learning
 - Sequential Learning
 - **Joint Learning**

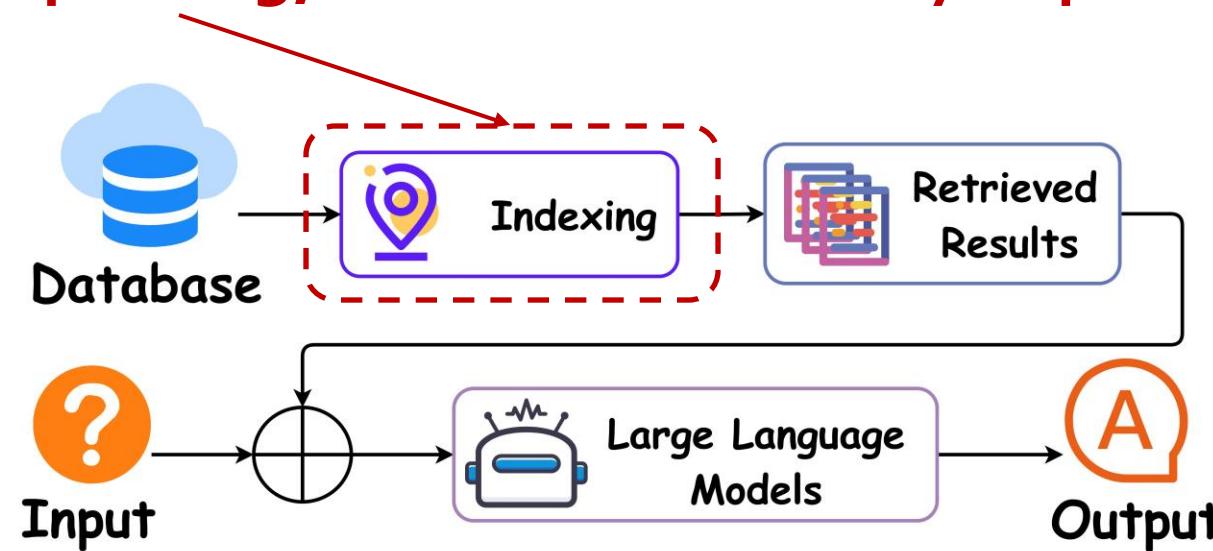
RA-LLM Learning: Joint Training

- **Retrieval models** is and **language models** are trained jointly.



RA-LLM Learning: Joint Training

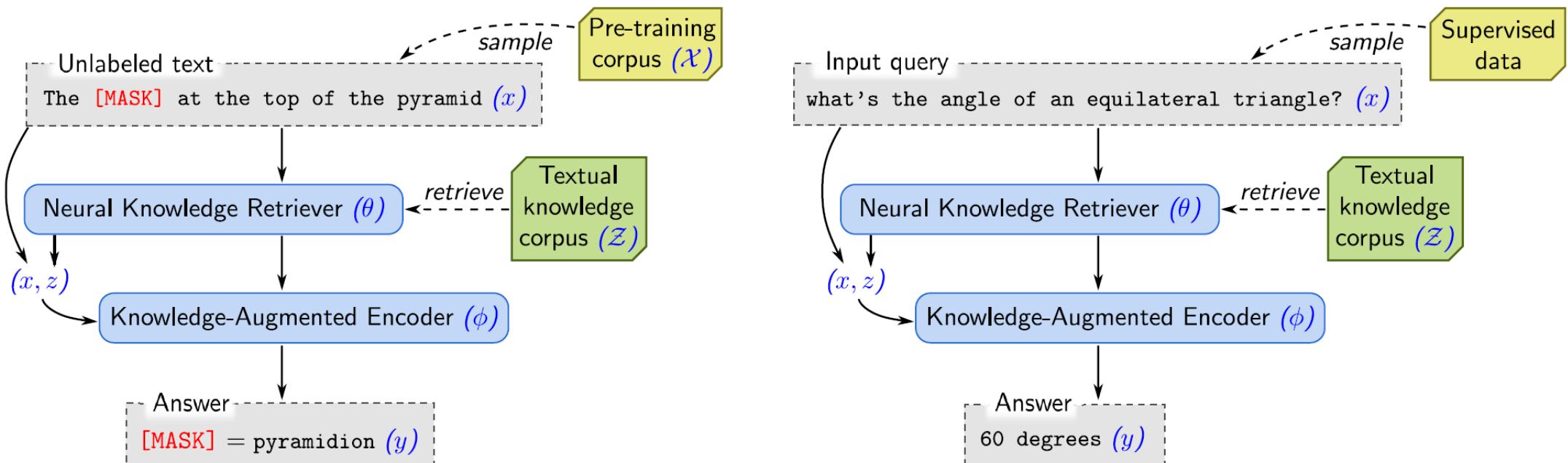
- **Retrieval Index Updating, which could be very expensive!**



- **Solutions:**
 - Asynchronous index updating
 - In-batch approximation

RA-LLM Learning: Joint Training

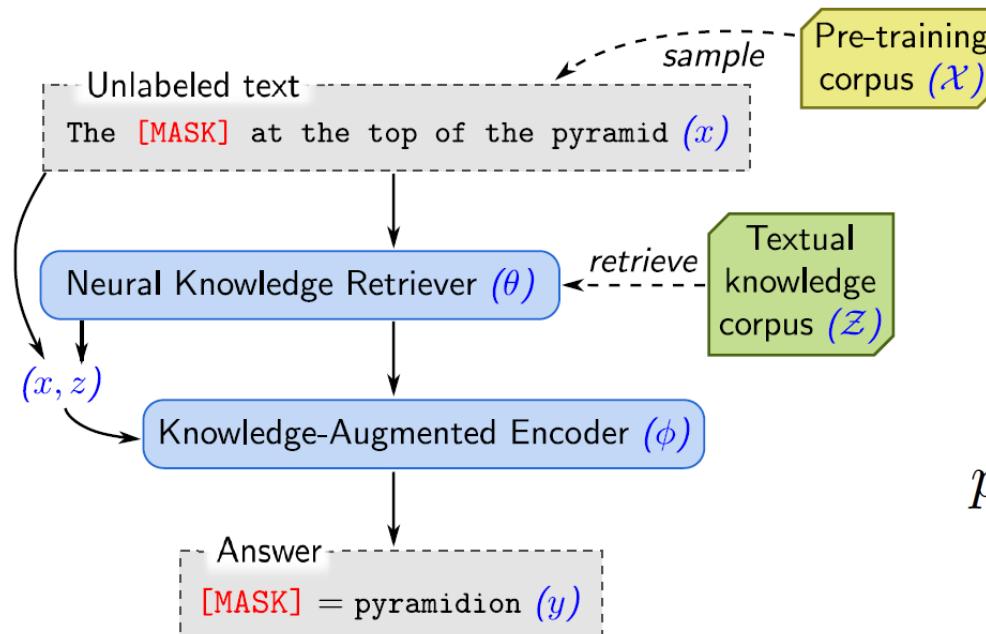
- REALM



Objective function: $p(y | x) = \sum_{z \in \mathcal{Z}} p(y | z, x) p(z | x).$

RA-LLM Learning: Joint Training

- **REALM**

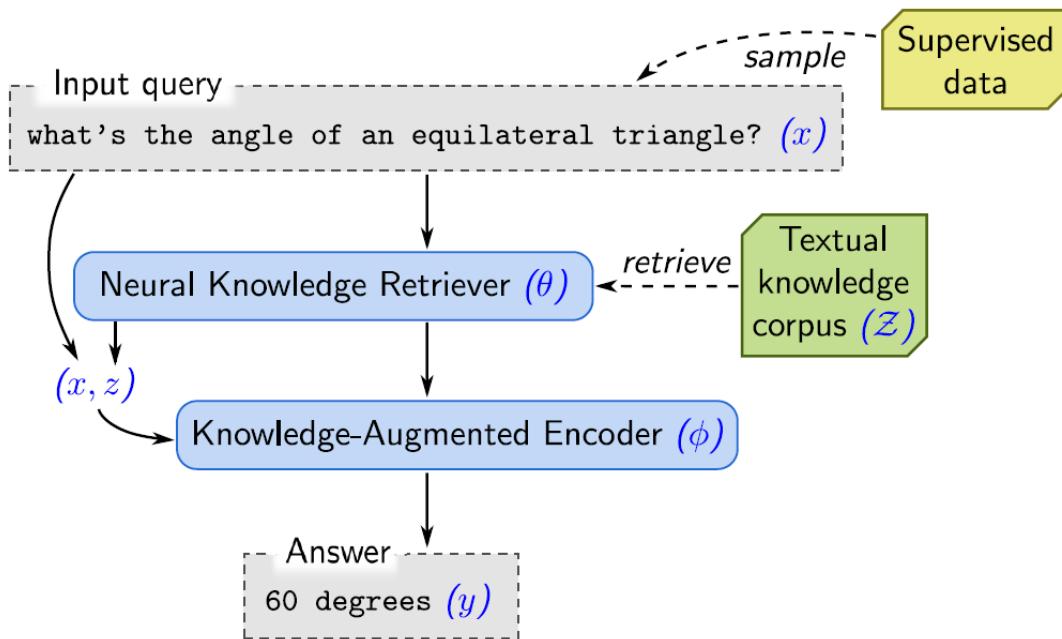


$$p(y | z, x) = \prod_{j=1}^{J_x} p(y_j | z, x)$$

$$p(y_j | z, x) \propto \exp(w_j^\top \text{BERT}_{\text{MASK}(j)}(\text{join}_{\text{BERT}}(x, z_{\text{body}})))$$

RA-LLM Learning: Joint Training

- REALM



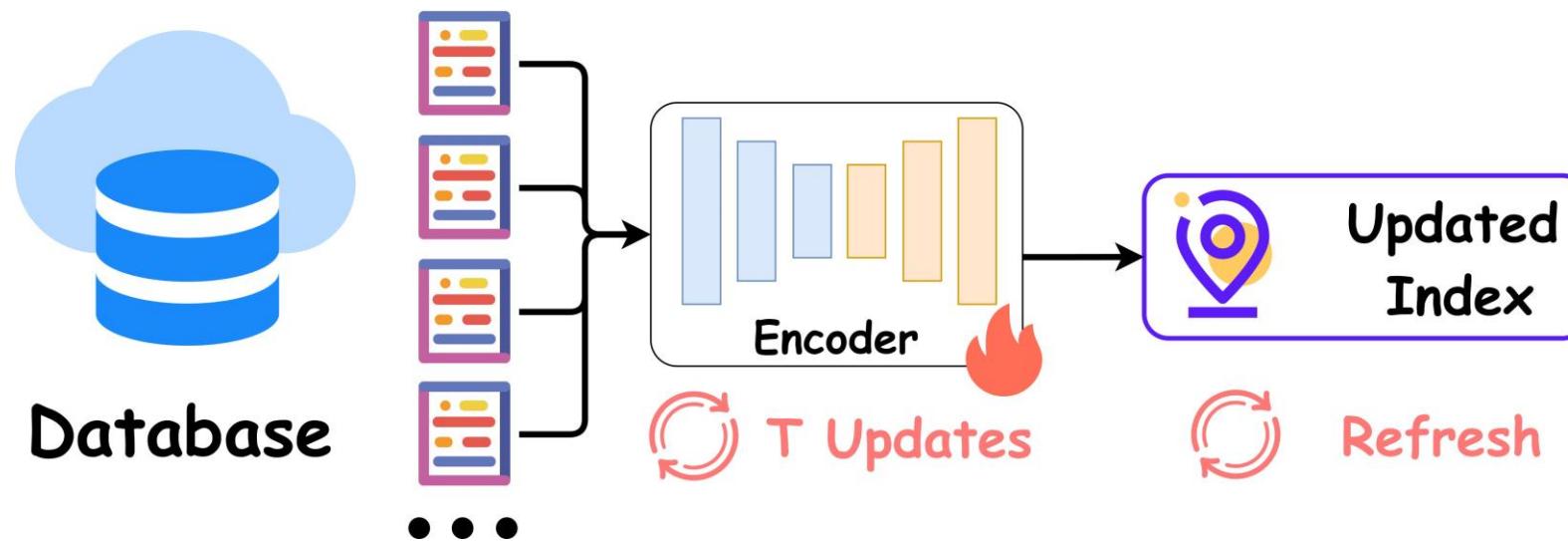
$$p(y | z, x) \propto \sum_{s \in S(z, y)} \exp \left(\text{MLP} \left([h_{\text{START}(s)}; h_{\text{END}(s)}] \right) \right)$$

$$h_{\text{START}(s)} = \text{BERT}_{\text{START}(s)}(\text{join}_{\text{BERT}}(x, z_{\text{body}})),$$

$$h_{\text{END}(s)} = \text{BERT}_{\text{END}(s)}(\text{join}_{\text{BERT}}(x, z_{\text{body}})),$$

RA-LLM Learning: Joint Training

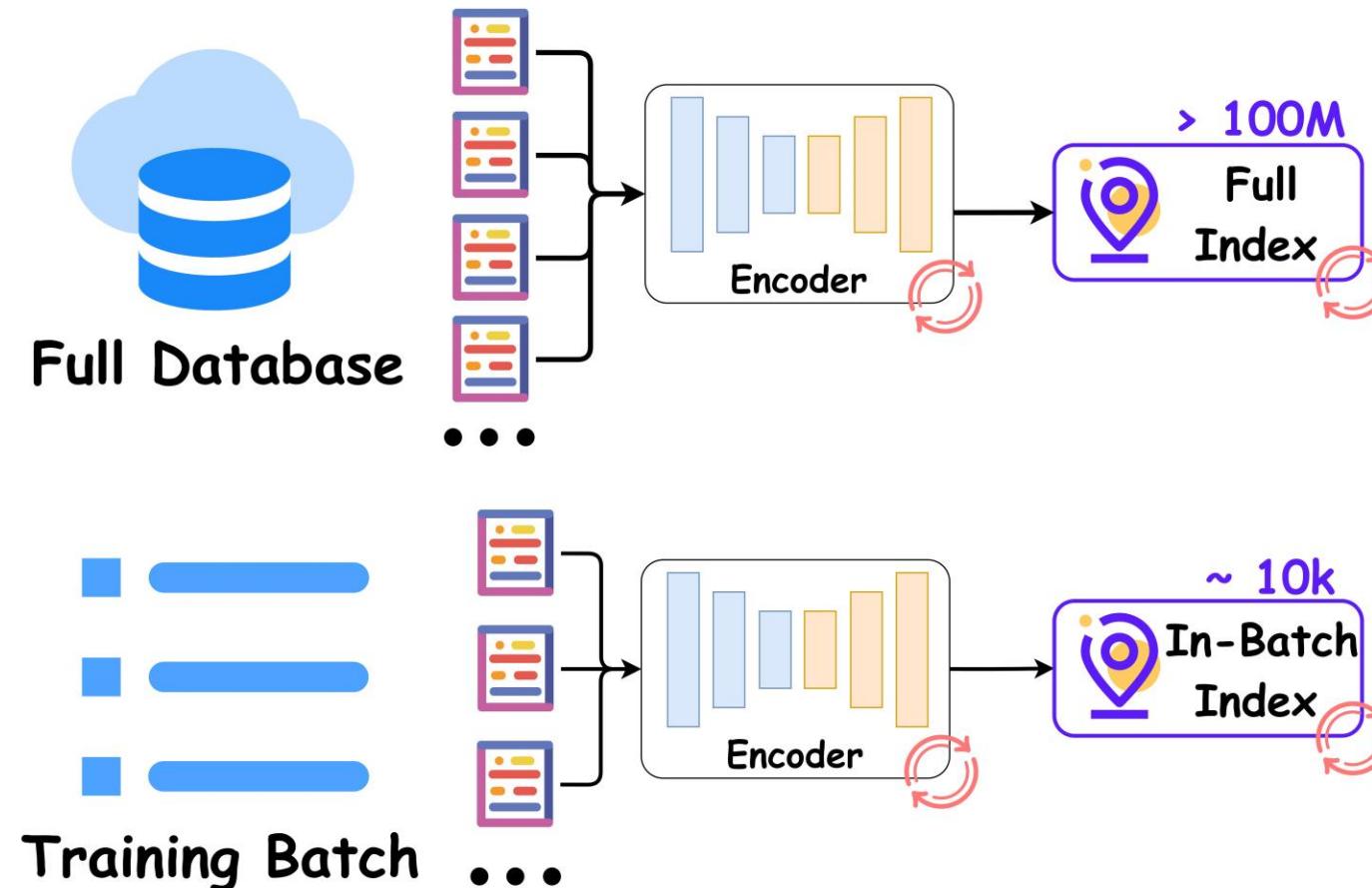
- REALM – Asynchronous Index Update



$$f(x, z) = \text{Embed}_{\text{input}}(x)^{\top} \text{Embed}_{\text{doc}}(z)$$

RA-LLM Learning : Joint Training

- TRIME – In-Batch Approximation

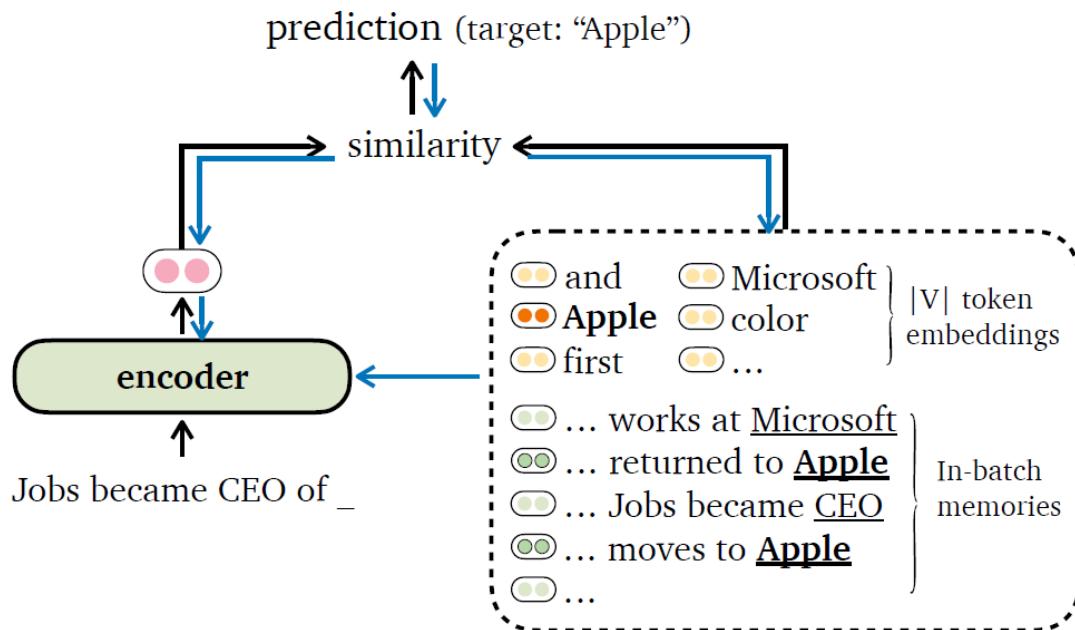


RA-LLM Learning : Joint Training

- TRIME

● Target token's embedding ● Positive in-batch memory
● Other token embeddings ● Negative in-batch memory

↑ Forward pass ↓ Back-propagation



Local Memory: $\mathcal{M}_{\text{local}}(c_t) = \{(c_j, x_j)\}_{1 \leq j \leq t-1}$.

Long-term Memory:

$$\mathcal{M}_{\text{long}}(c_t^{(i)}) = \{(c_j^{(k)}, x_j^{(k)})\}_{1 \leq k < i, 1 \leq j}$$

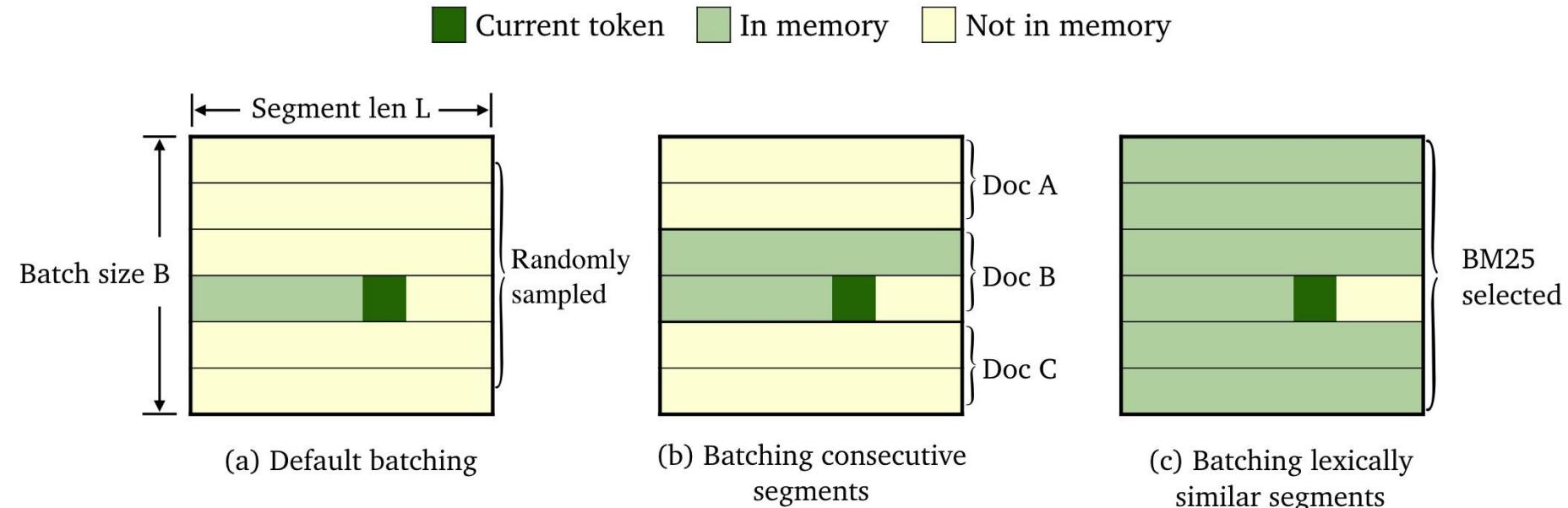
External Memory: $\mathcal{M}_{\text{ext}} = \{(c_j, x_j) \in \mathcal{D}\}$.

Training Objective:

$$P(w | c) \propto \exp(E_w^\top f_\theta(c)) + \sum_{(c_j, x_j) \in \mathcal{M}_{\text{train}}: x_j = w} \exp(\text{sim}(g_\theta(c), g_\theta(c_j))).$$

RA-LLM Learning : Joint Training

- TRIME Data Batching Strategy



Use BM25 scores to find similar text chunks to provide more training signals



Tutorial Outline

- **Part 1: Introduction** of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)
- **Part 2: Architecture** of RA-LLMs and **Main Modules** (Dr. Yujuan Ding)
- **Part 3: Data Management** for RA-LLMs (Pangjing Wu)
- **Part 4: Learning Approach** of RA-LLMs (Liangbo Ning)
- **Part 5: Applications of RA-LLMs (Shijie Wang)**
- **Part 6: Challenges and Future Directions** of RA-LLMs (Dr. Wenqi Fan)

Website of this tutorial
Check out the slides and more information!



Website

PART 4: Application of RA-LLMs

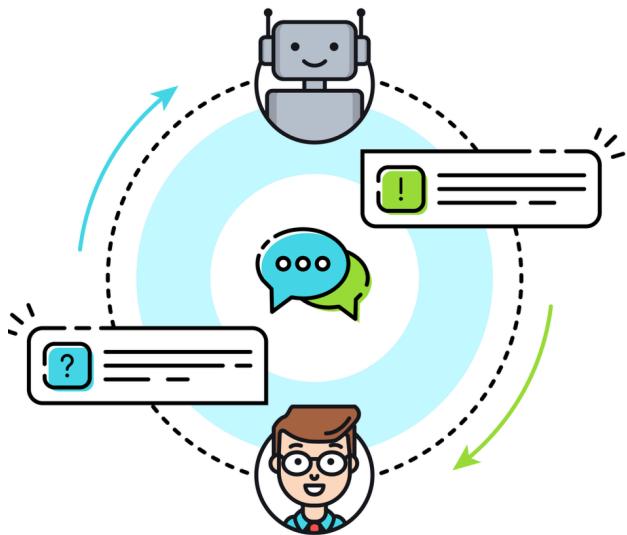


Presenter
Shijie Wang
HK PolyU

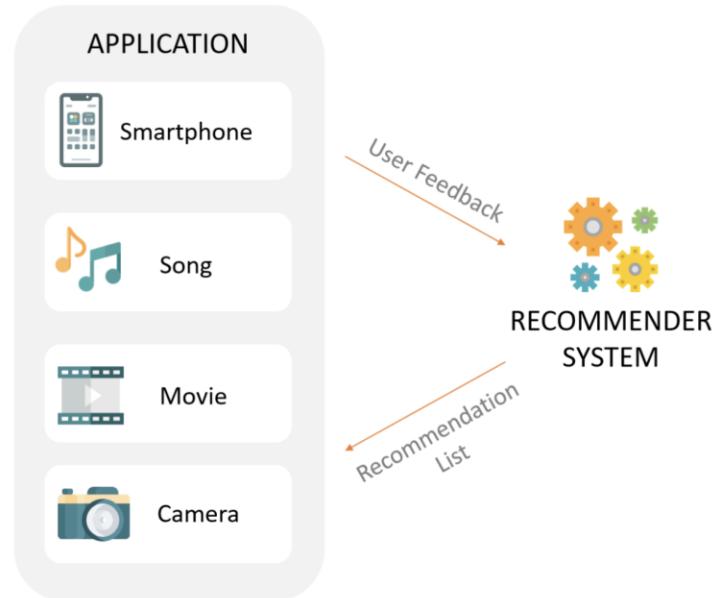
- **NLP applications**
- **Downstream tasks**
- **Domain-specific applications**

RA-LLM Applications

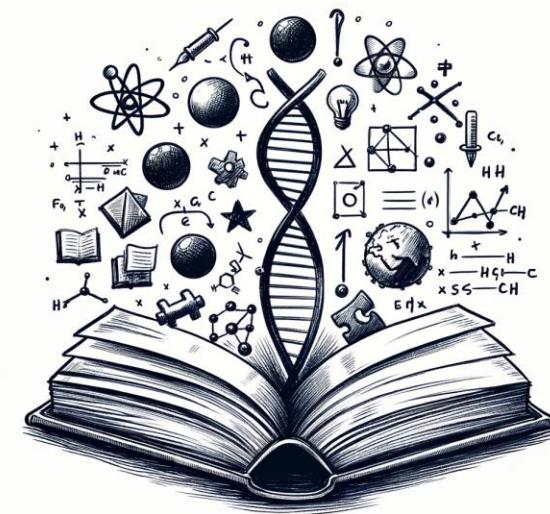
- **Various applications**



Chatbots



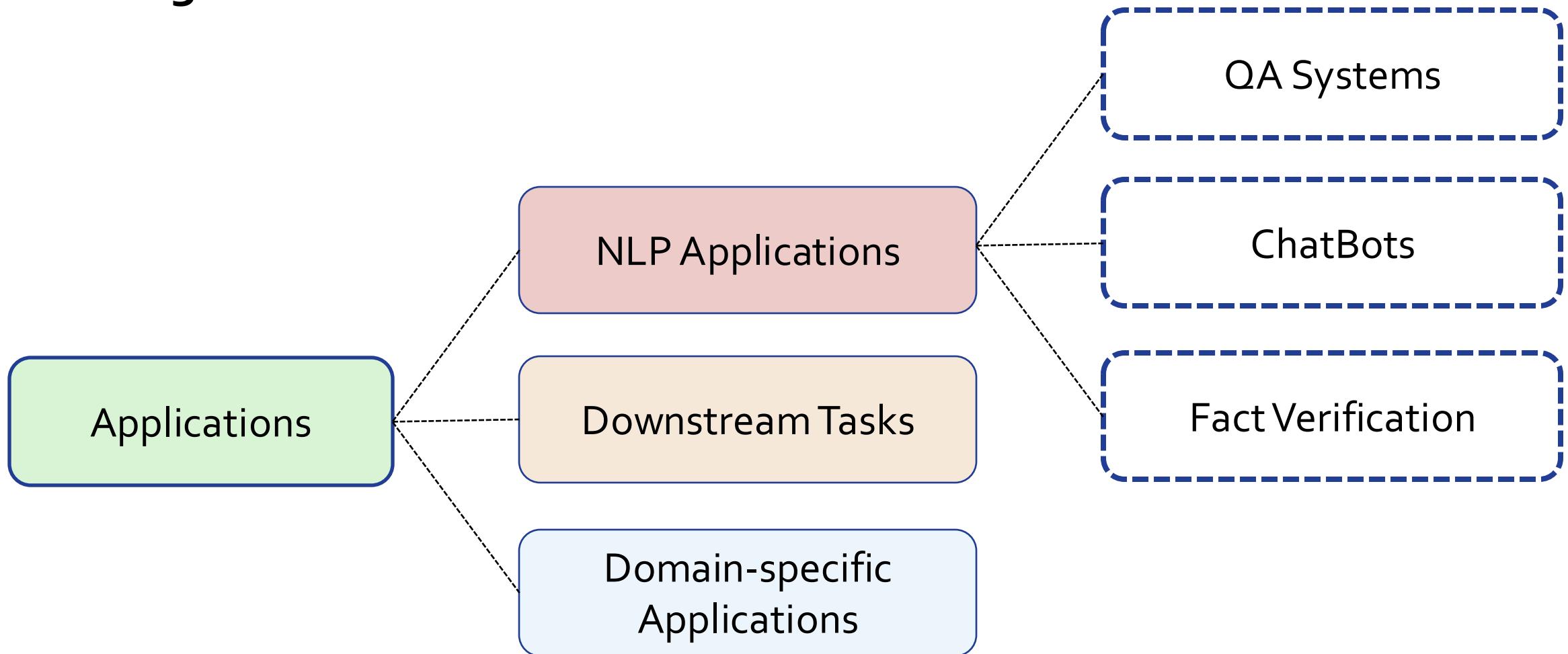
Recommendation



AI for Science

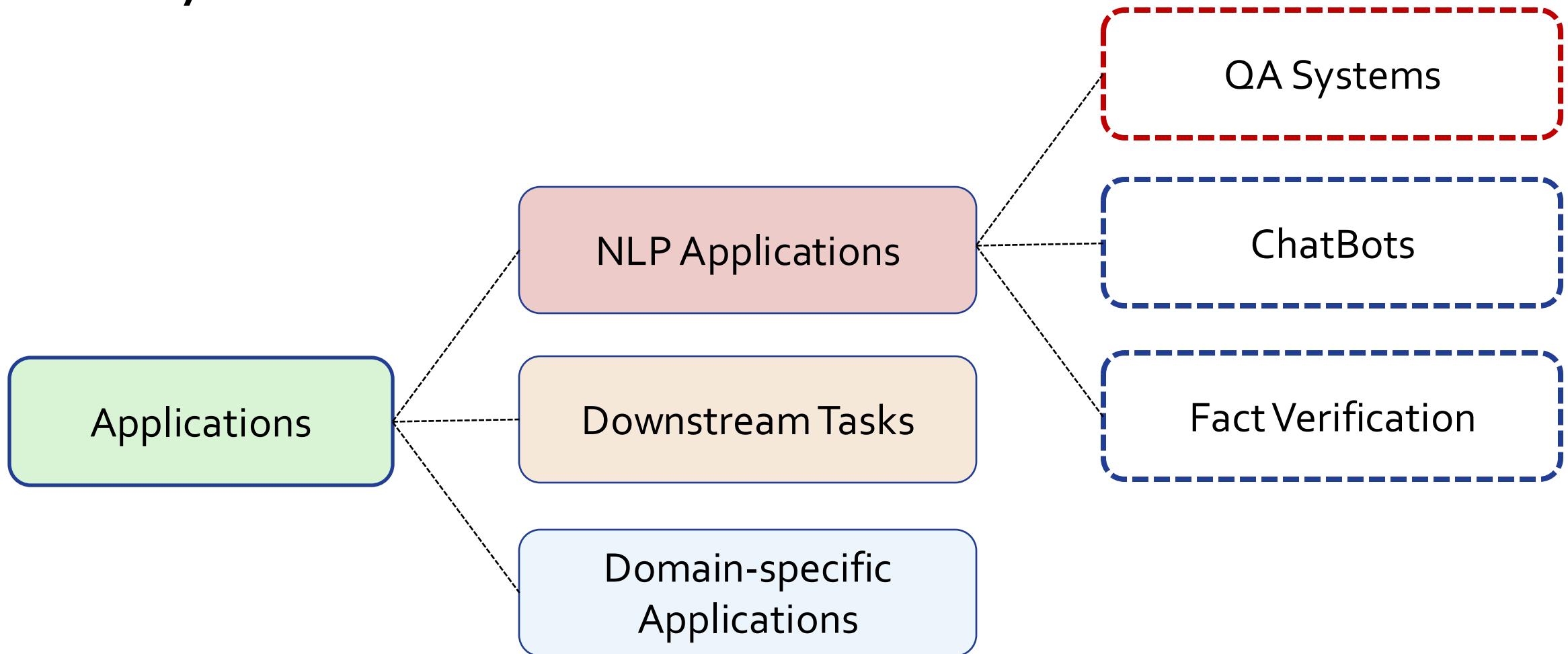
RA-LLM Applications: NLP Applications

- **Categories**



RA-LLM Applications: NLP Applications

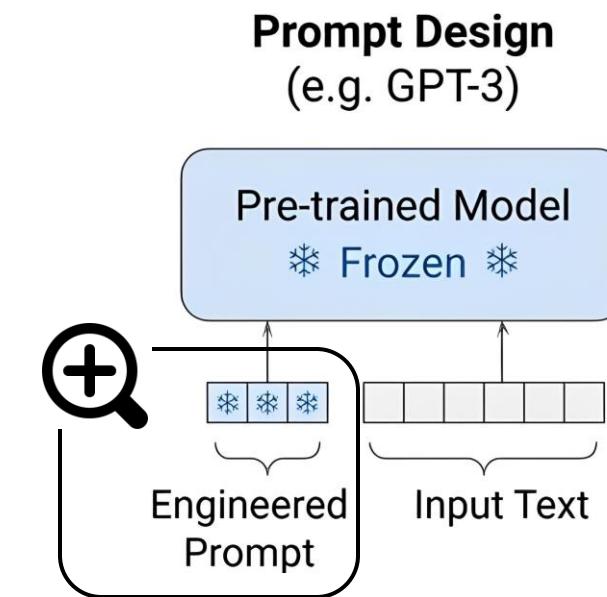
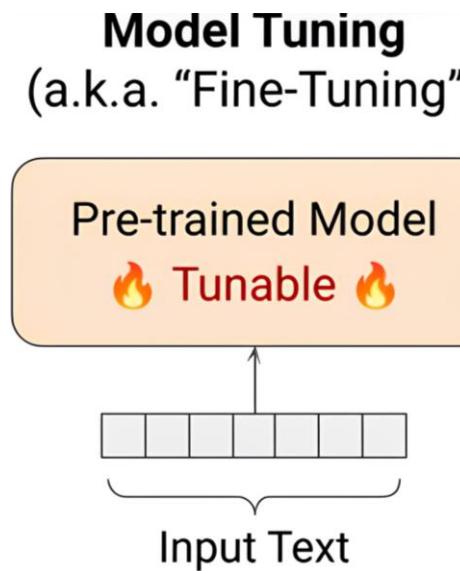
- **QA Systems**



RA-LLM Applications: QA Systems

- **QA systems**

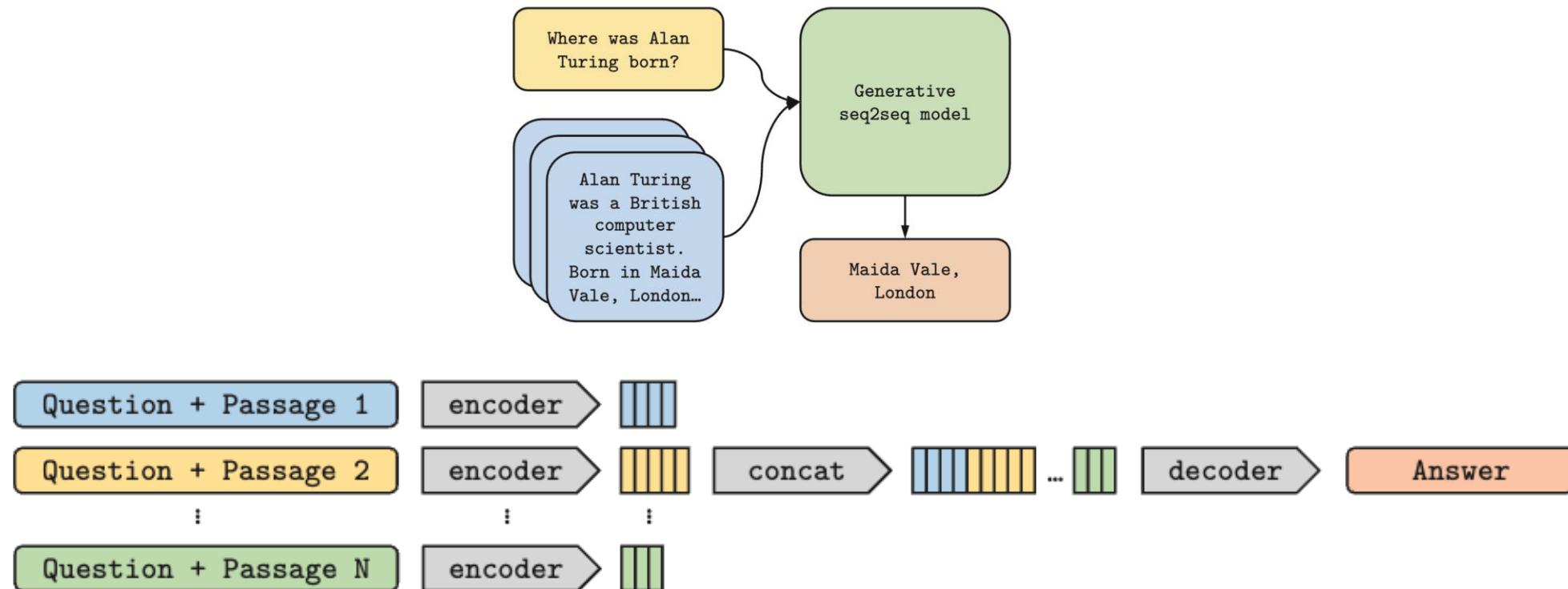
- Challenges:
 - Open-domain QA
 - Domain-specific QA
- How to solve?
 - Fine-tuning
 - Prompting



RA-LLM Applications: QA Systems

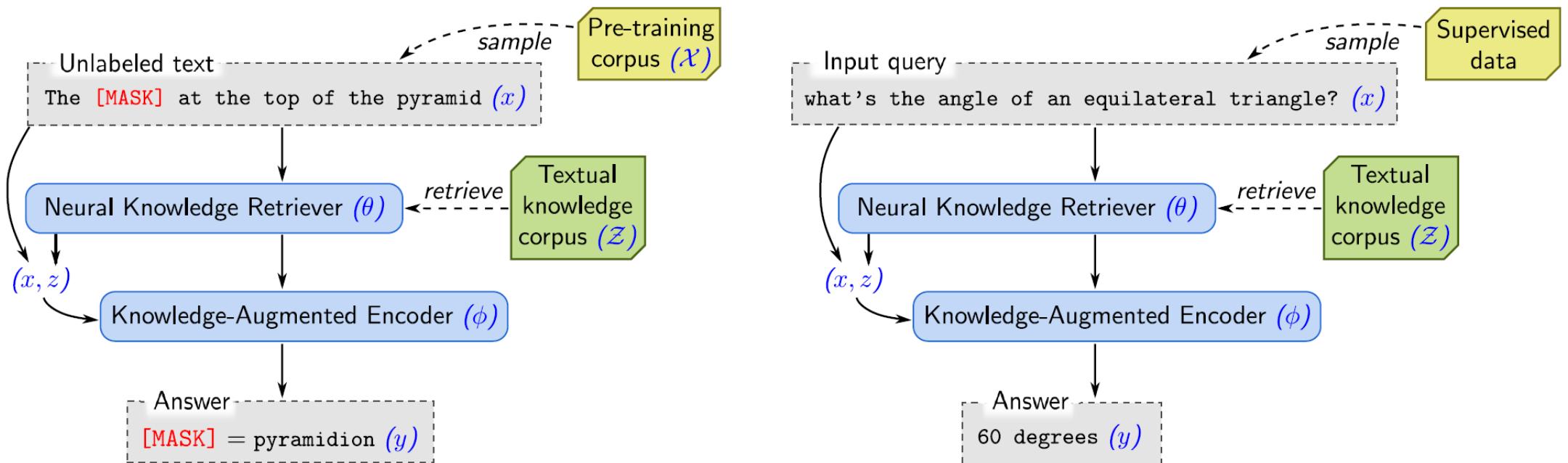
- **Retrieves for open-domain QA**

- Retrieves support text passages from an external source of knowledge



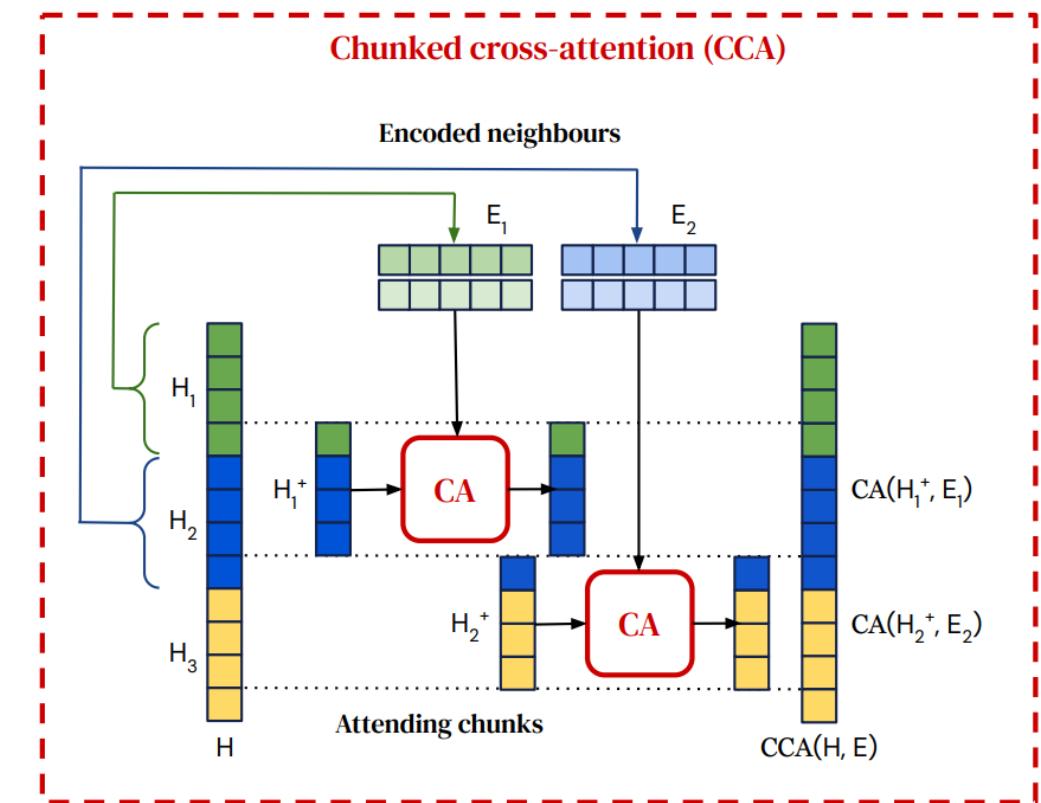
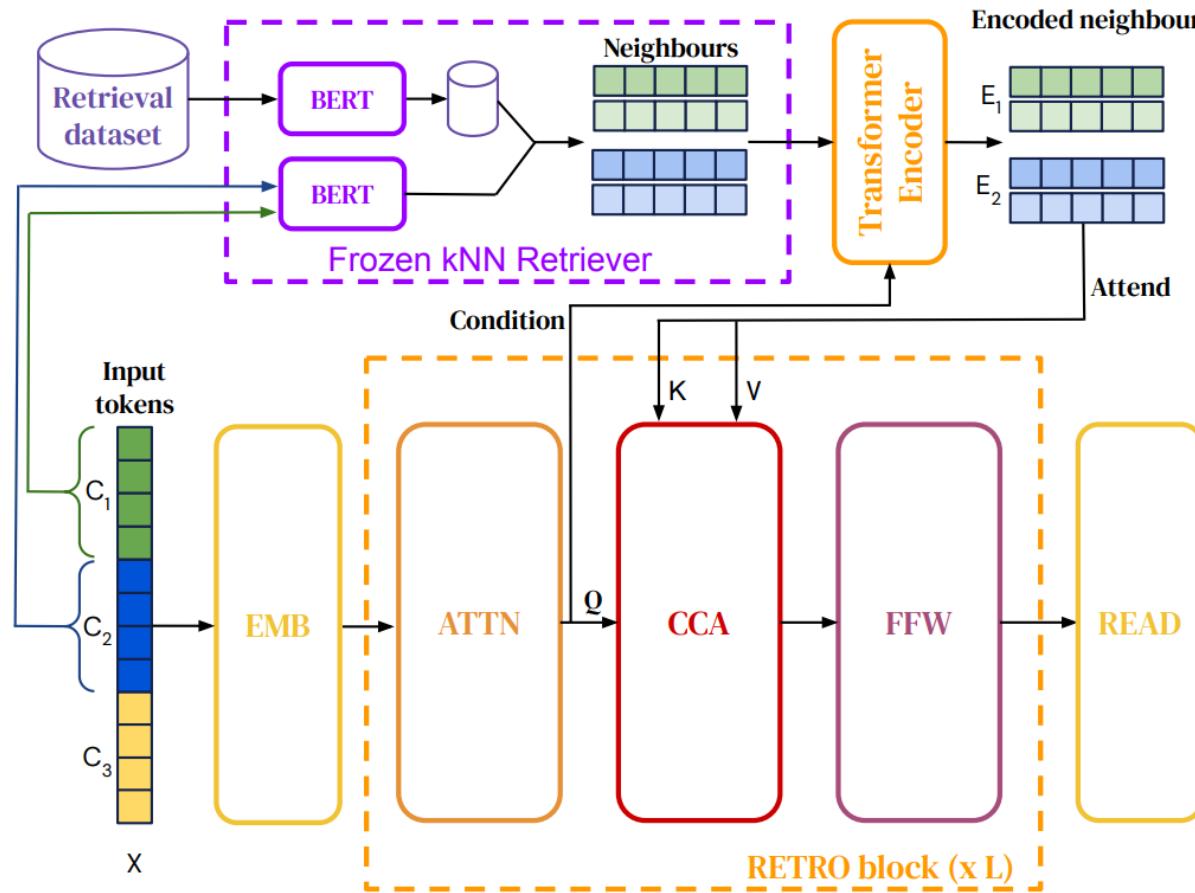
RA-LLM Applications: QA Systems

- **REALM**



RA-LLM Applications: QA Systems

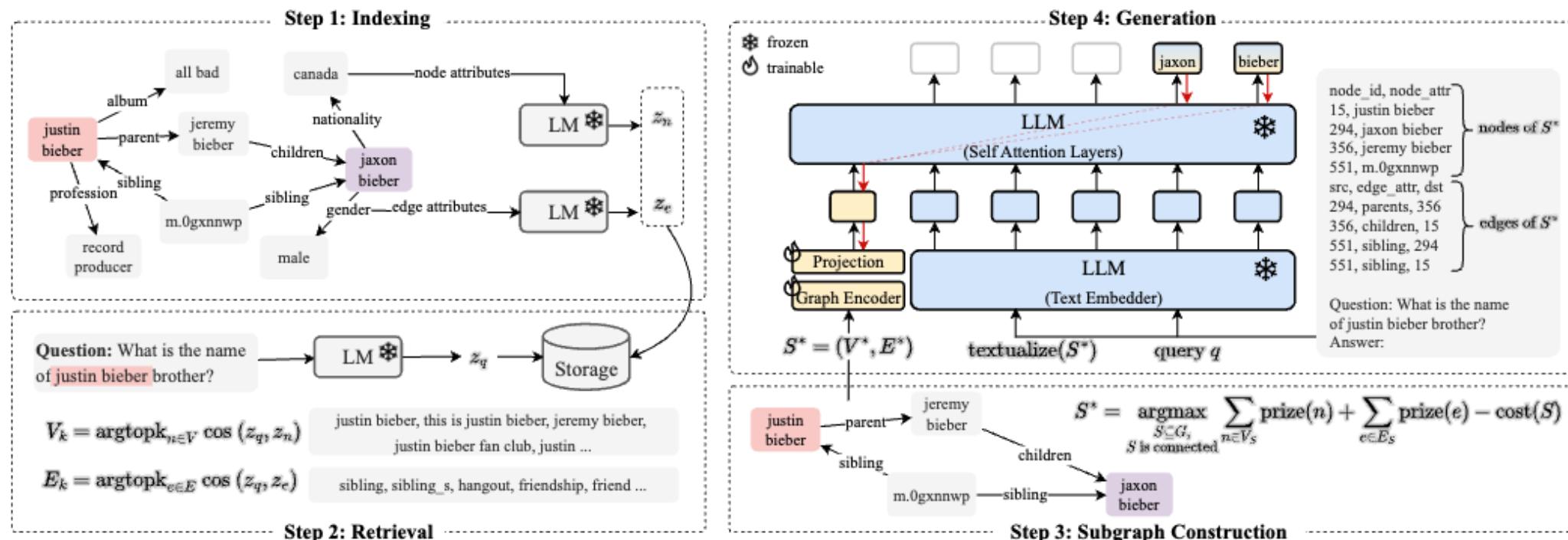
- RETRO (Retrieval-enhanced transformer)



RA-LLM Applications: QA Systems

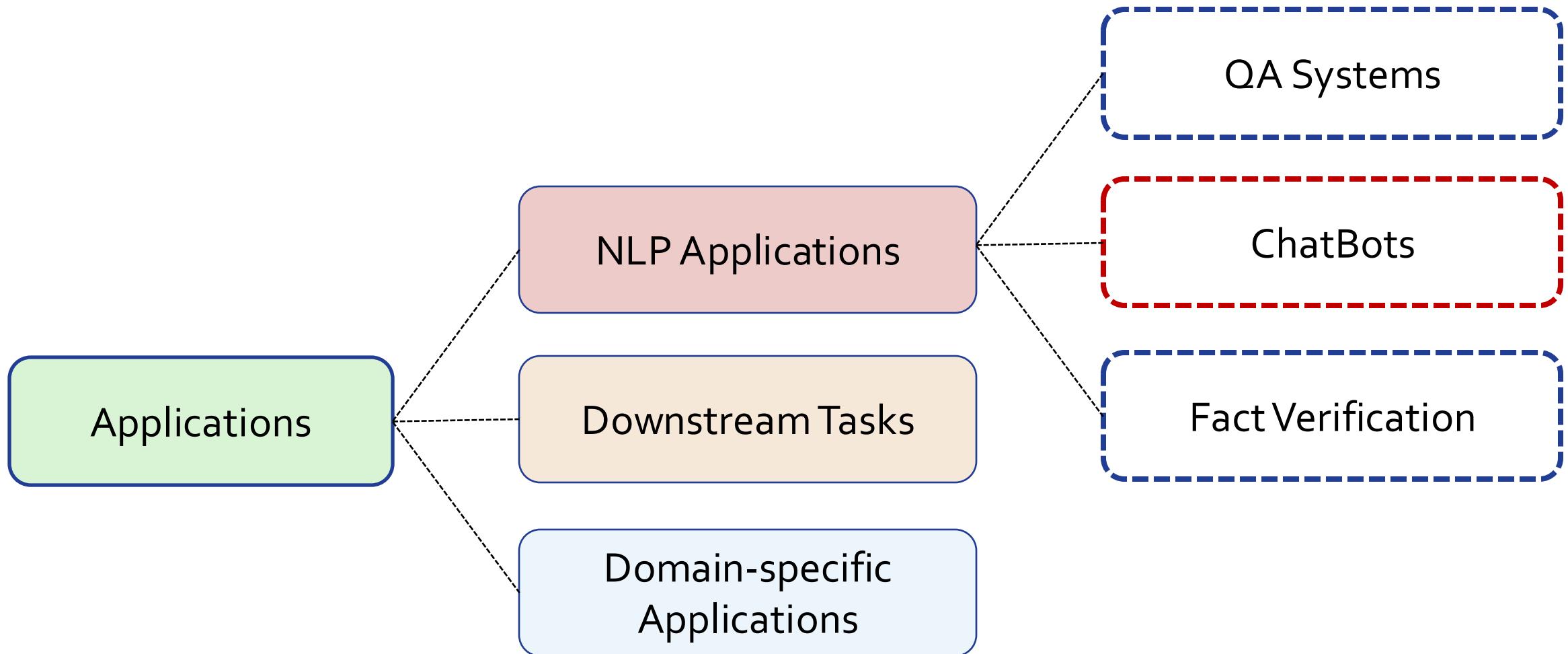
• G-Retriever

Retrieves from knowledge graph for question-answering



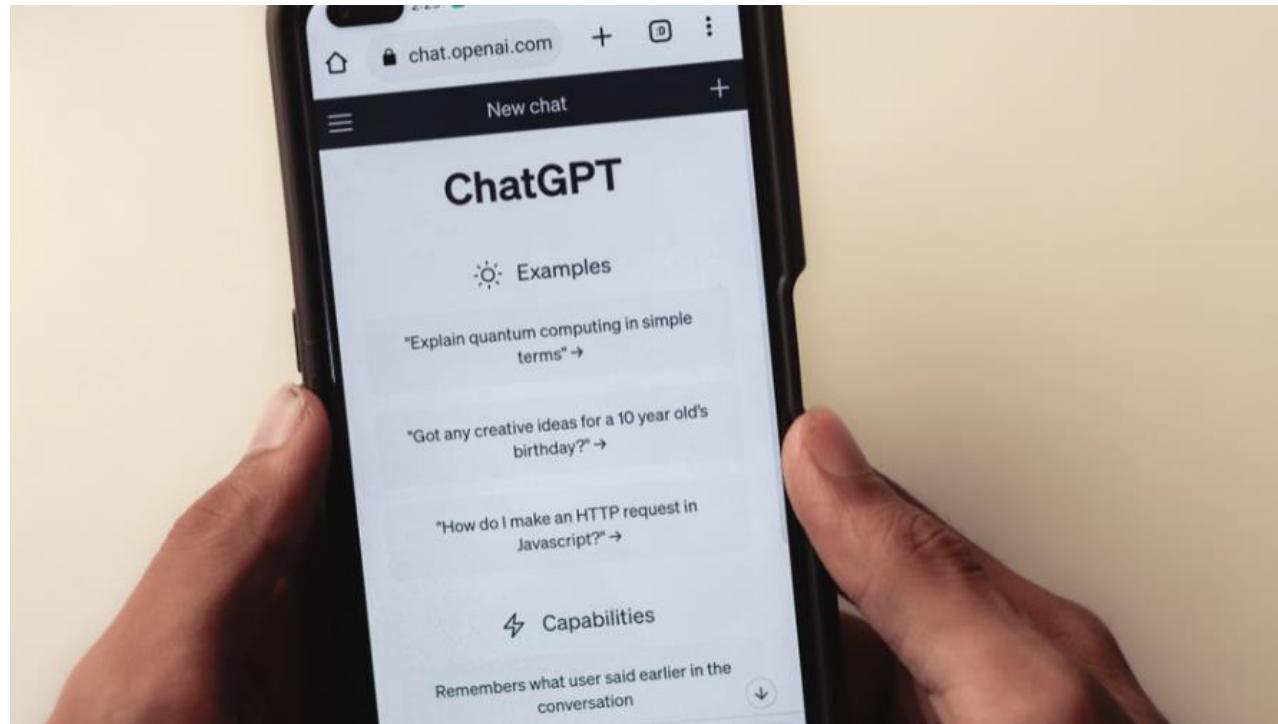
RA-LLM Applications: NLP Applications

- **ChatBots**



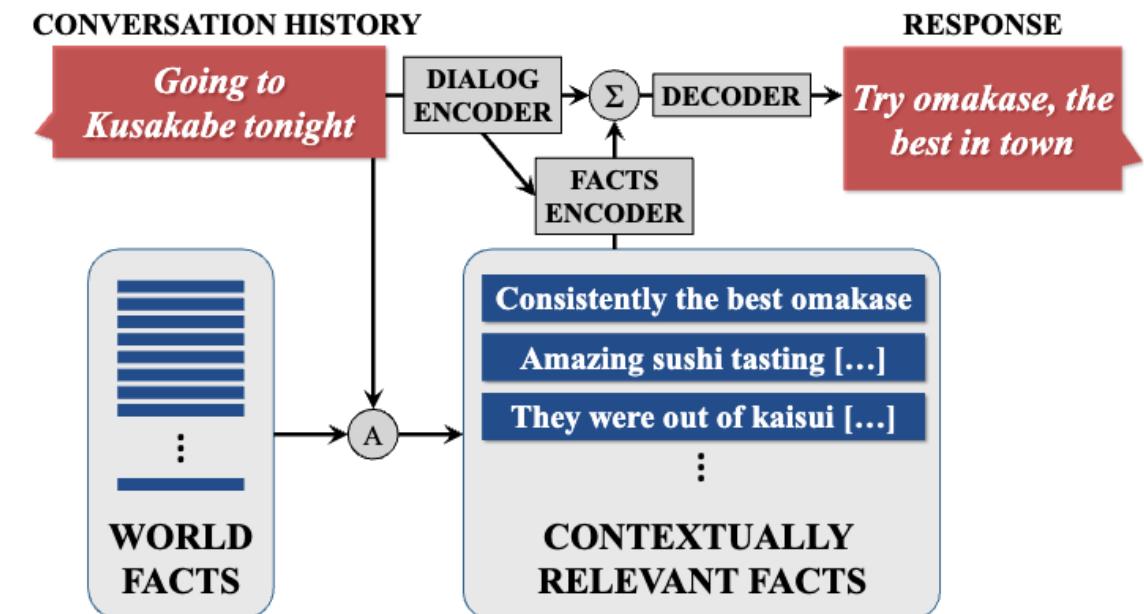
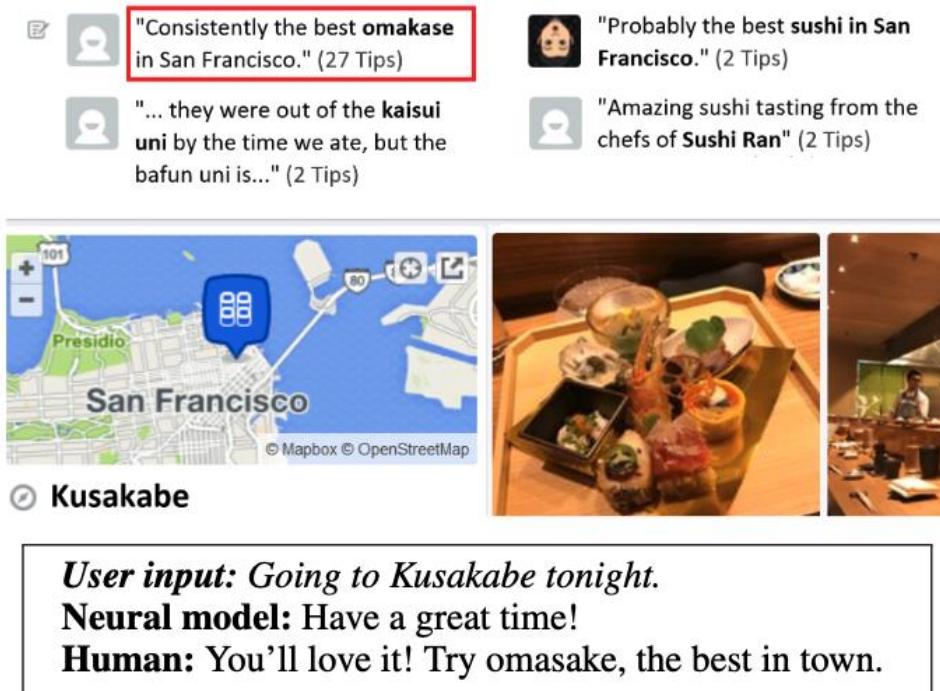
RA-LLM Applications: Chatbots

- **ChatBots**



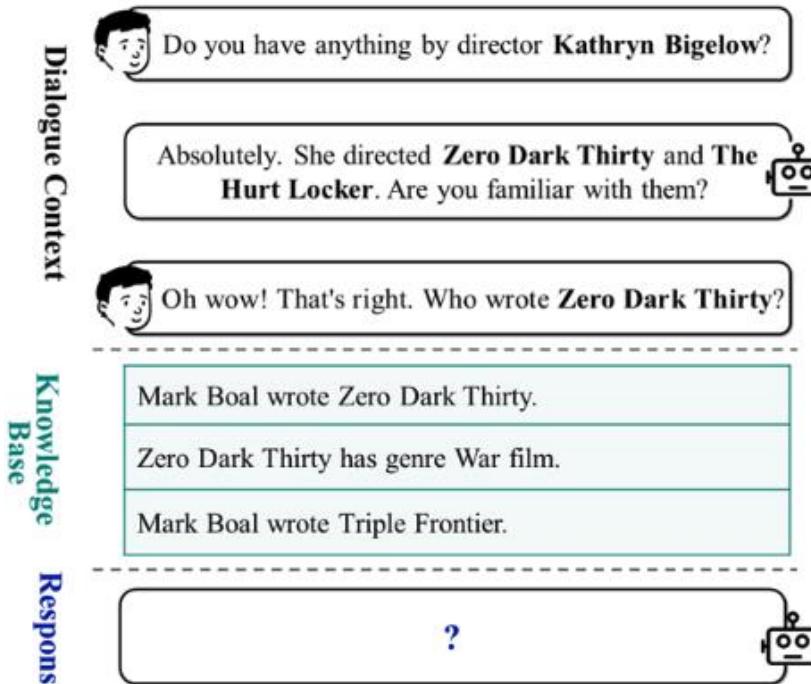
RA-LLM Applications: Chatbots

- Knowledge-grounded model

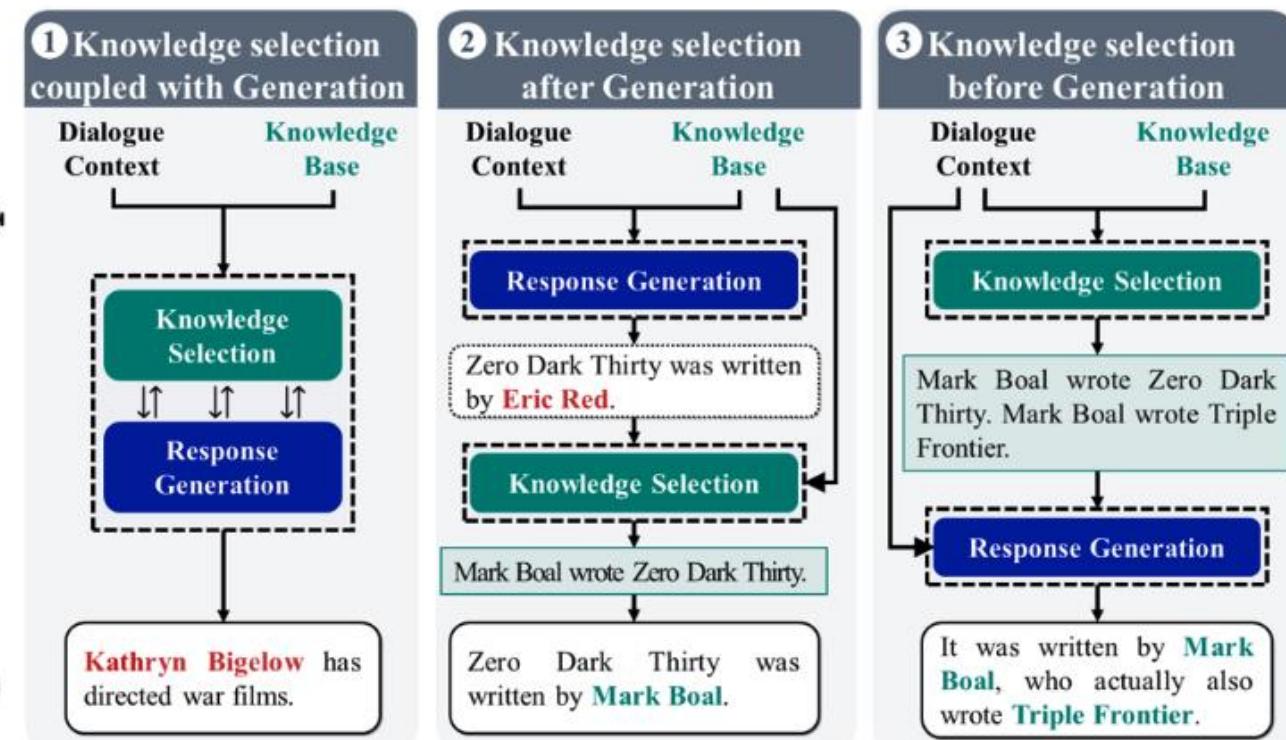


RA-LLM Applications: Chatbots

- **GATE**



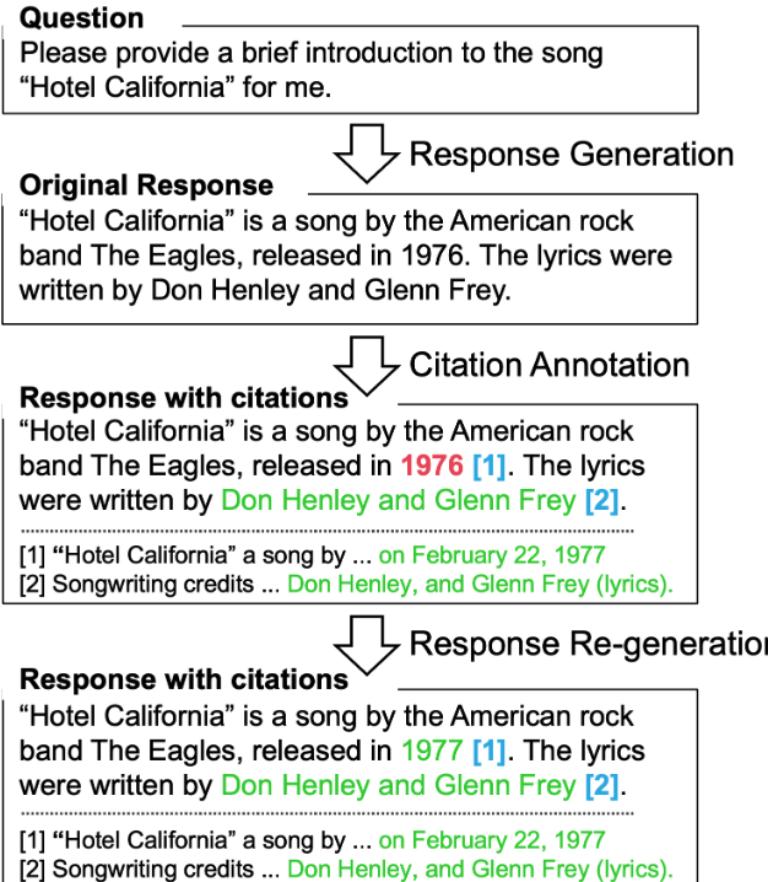
(a) Example of knowledge-grounded dialogue



(b) Three categories of knowledge selection

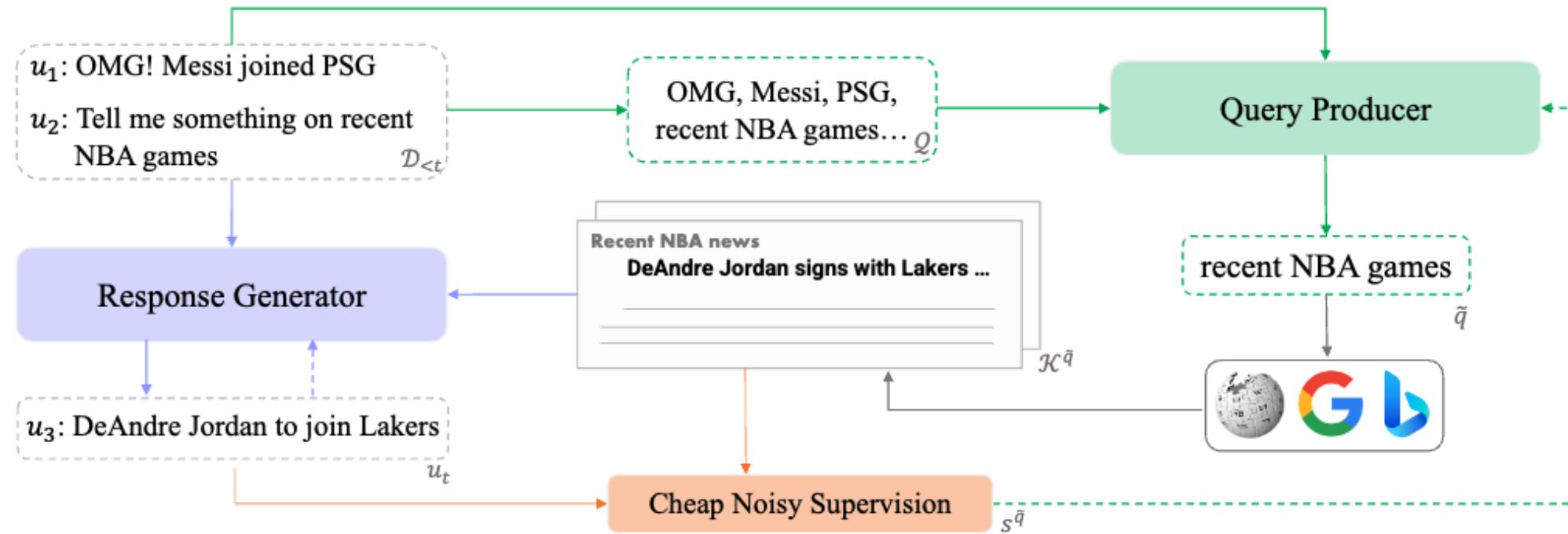
RA-LLM Applications: Chatbots

- CEG



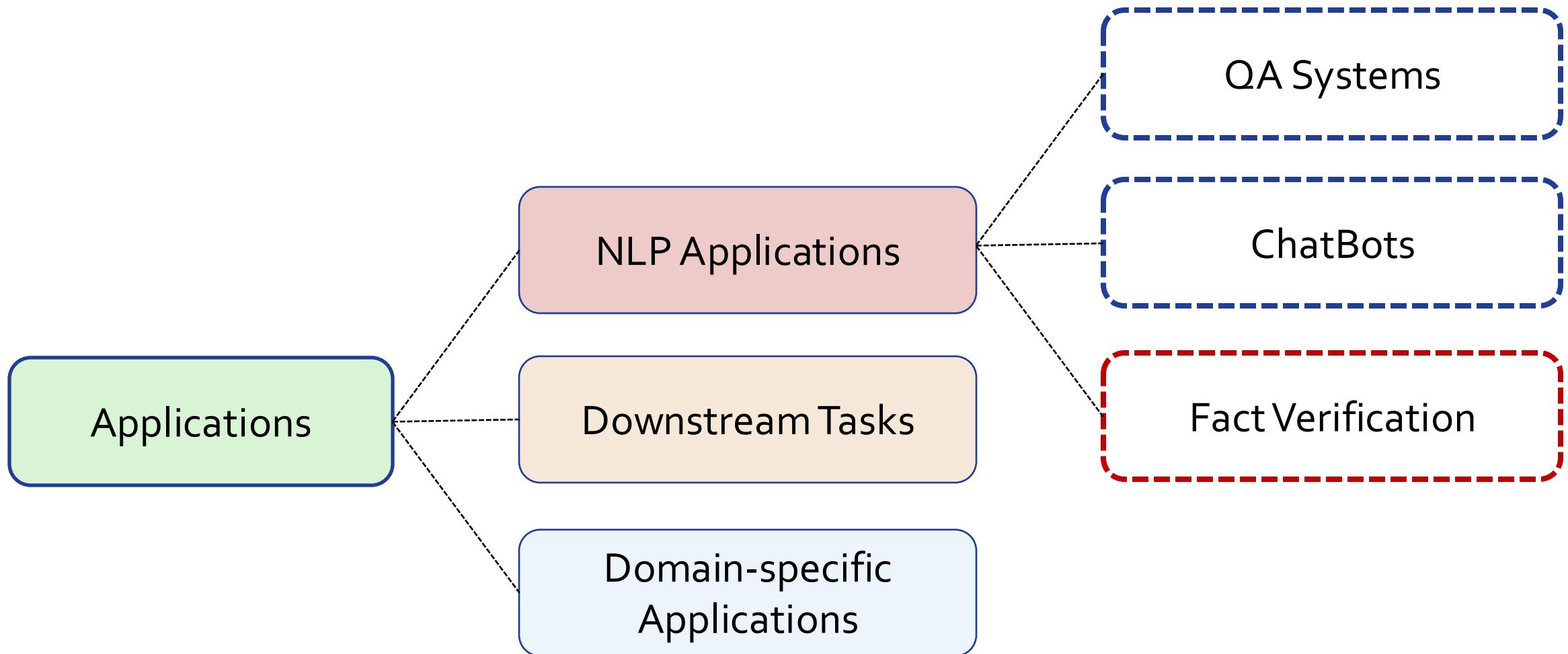
RA-LLM Applications: Chatbots

- **Search-engine-augmented chatbots**



RA-LLM Applications: NLP Applications

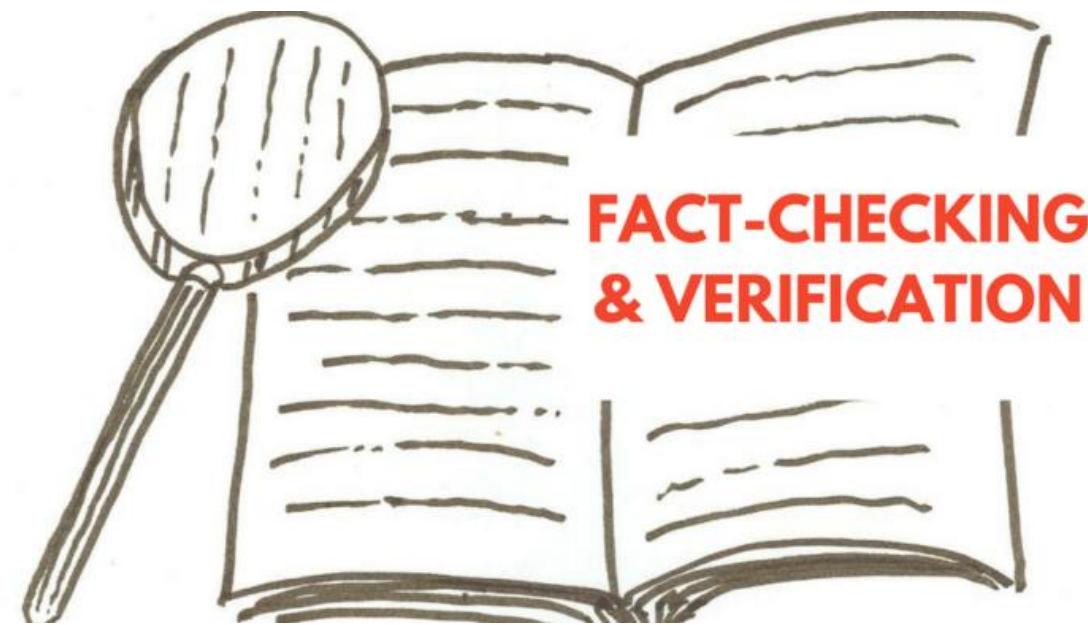
- Fact verification



RA-LLM Applications: Fact Verification

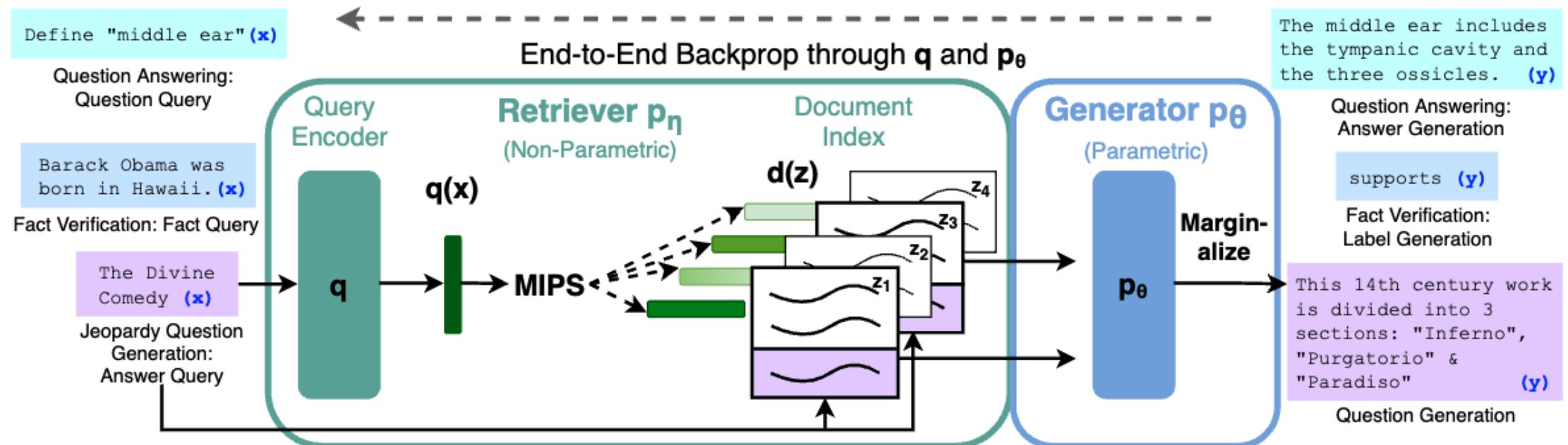
- **Fact verification**

Fact Verification is a critical task in verifying the accuracy and reliability of information



RA-LLM Applications: Fact Verification

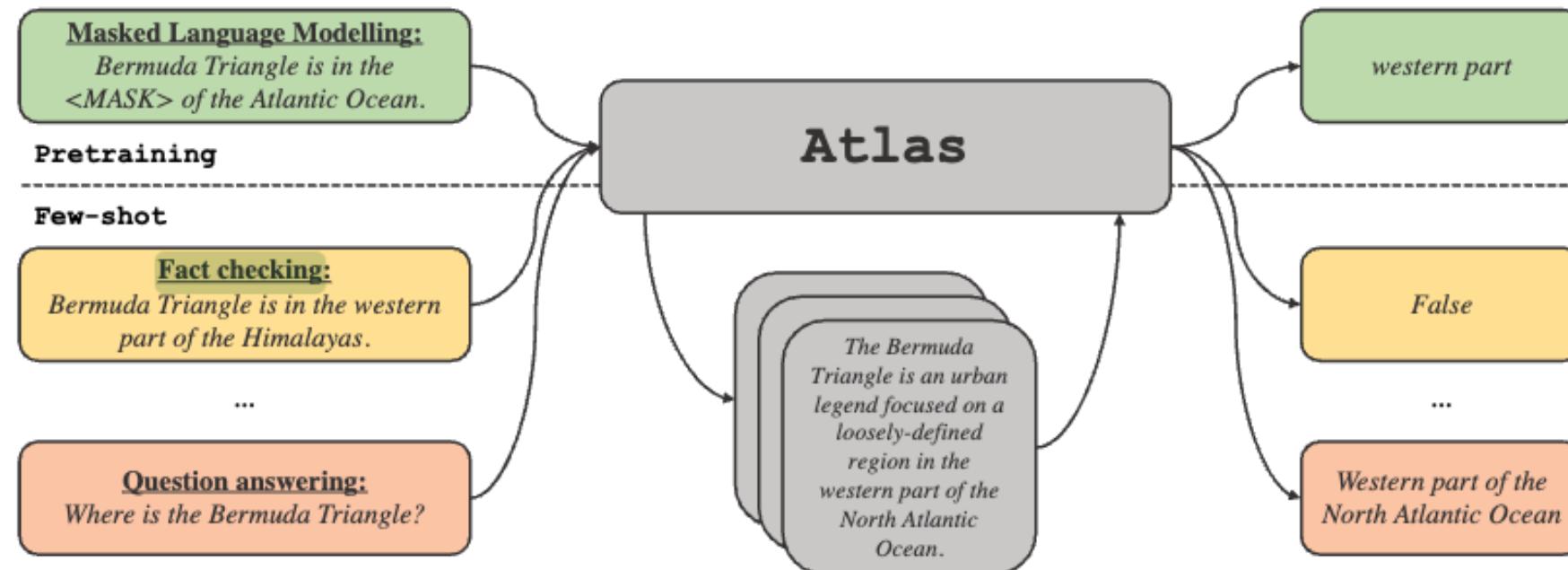
- Fact verification



RA-LLM Applications: Fact Verification

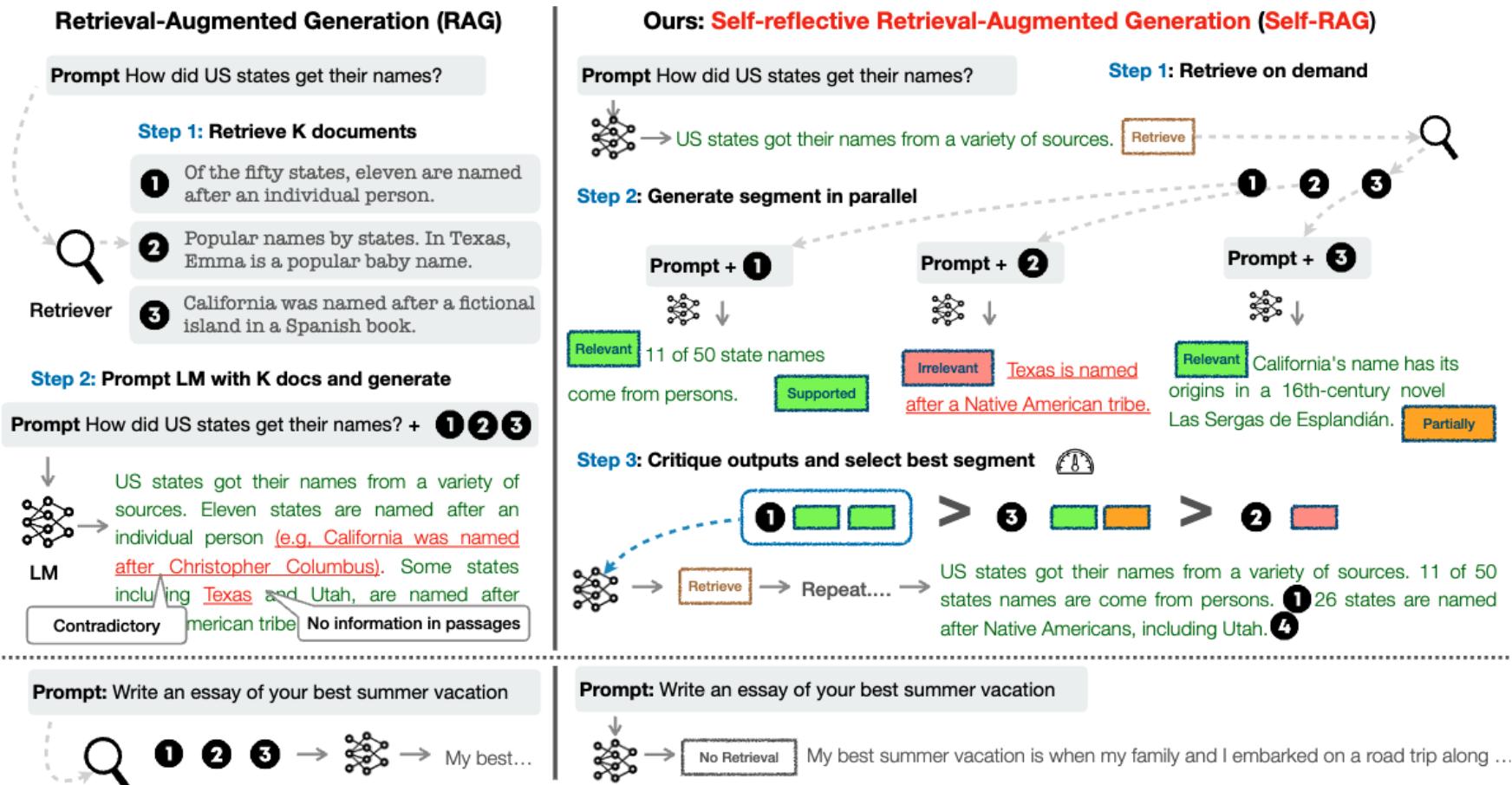
- **Fact verification**

- Fact verification is usually together with other NLP tasks (such as Q & A)
- ATLAS:



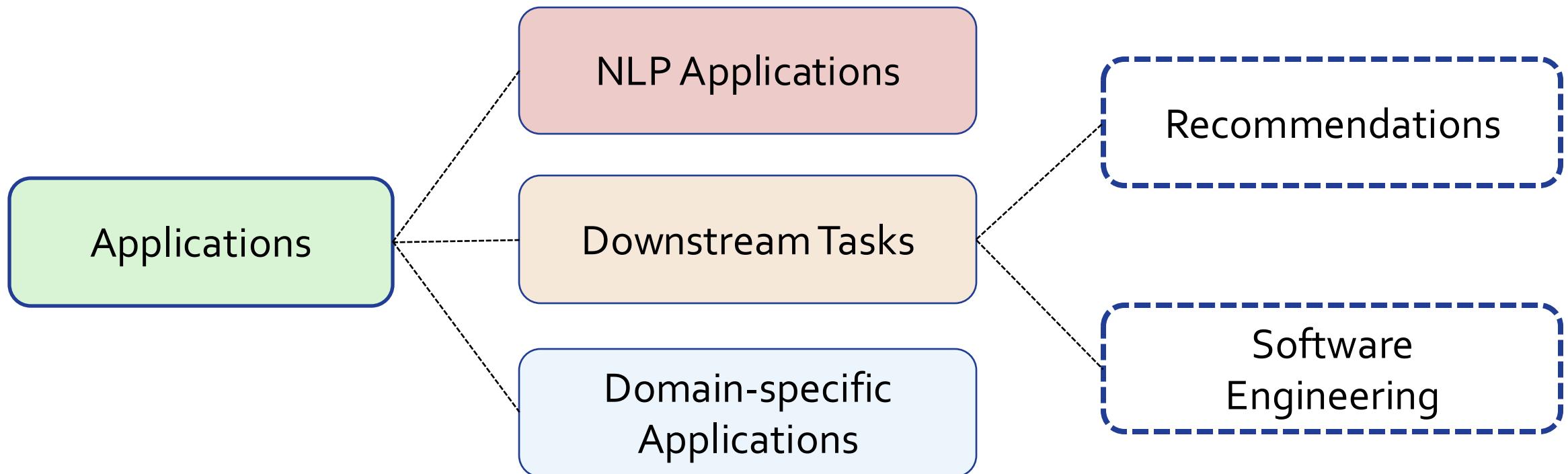
RA-LLM Applications: Fact Verification

• Self-RAG



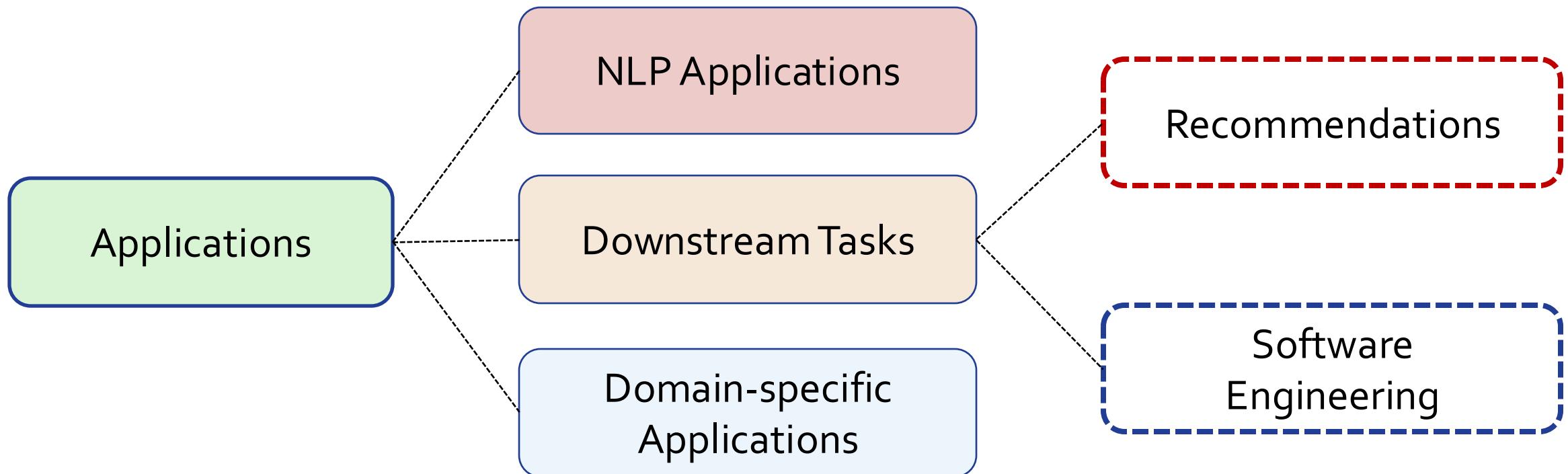
RA-LLM Applications: Downstream Tasks

- **Downstream tasks**



RA-LLM Applications: Recommendations

- **Recommendations**



RA-LLM Applications: Recommendations

- **Recommendations**

Recommendation has been widely applied in online services

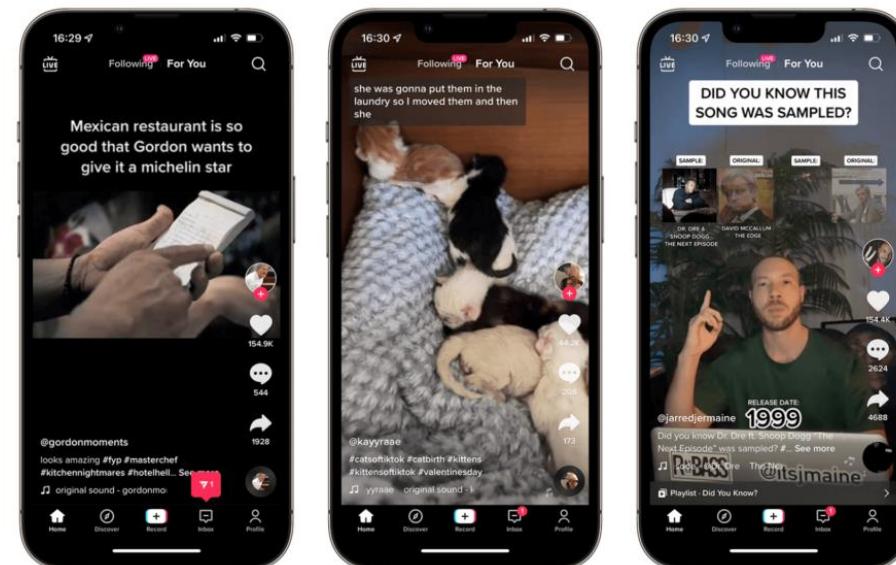


News/Video/Image Recommendation

TikTok's recommendation algorithm

Top 10 Global Breakthrough
Technologies in 2021

MIT
Technology
Review



RA-LLM Applications: Recommendations

• LLMs in recommendations



RA-LLM Applications: Recommendations

- Conventional item-based LLM reasoning process



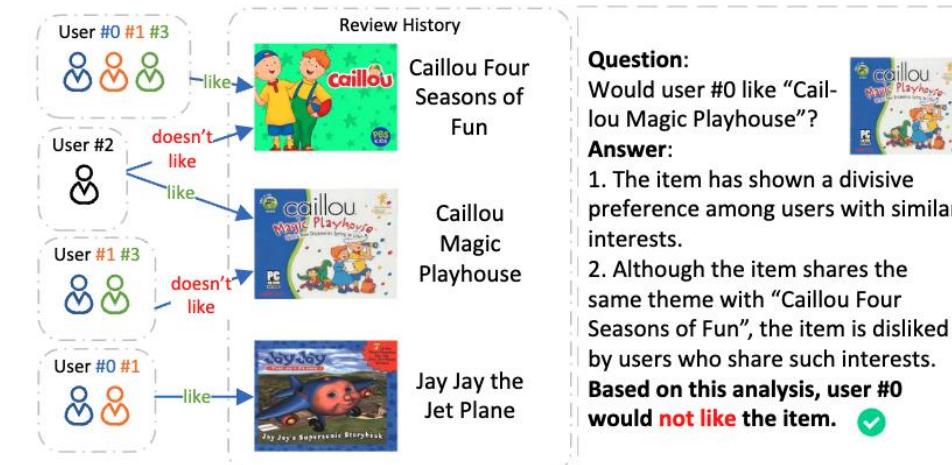
(a) Conventional item-based [16, 42] LLM reasoning process.

RA-LLM Applications: Recommendations

- Collaborative retrieval augmented LLM reasoning process



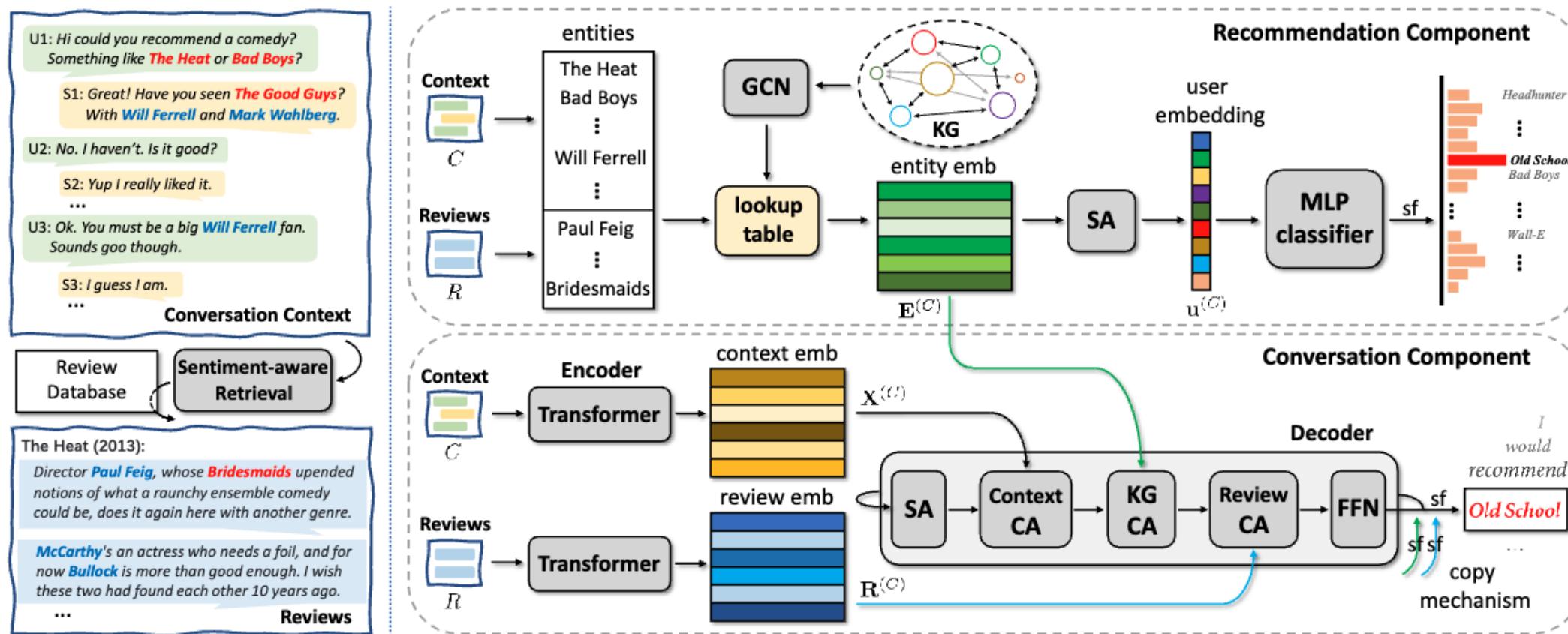
(a) Conventional item-based [16, 42] LLM reasoning process.



(b) Collaborative Retrieval Augmented LLM reasoning process.

RA-LLM Applications: Recommendations

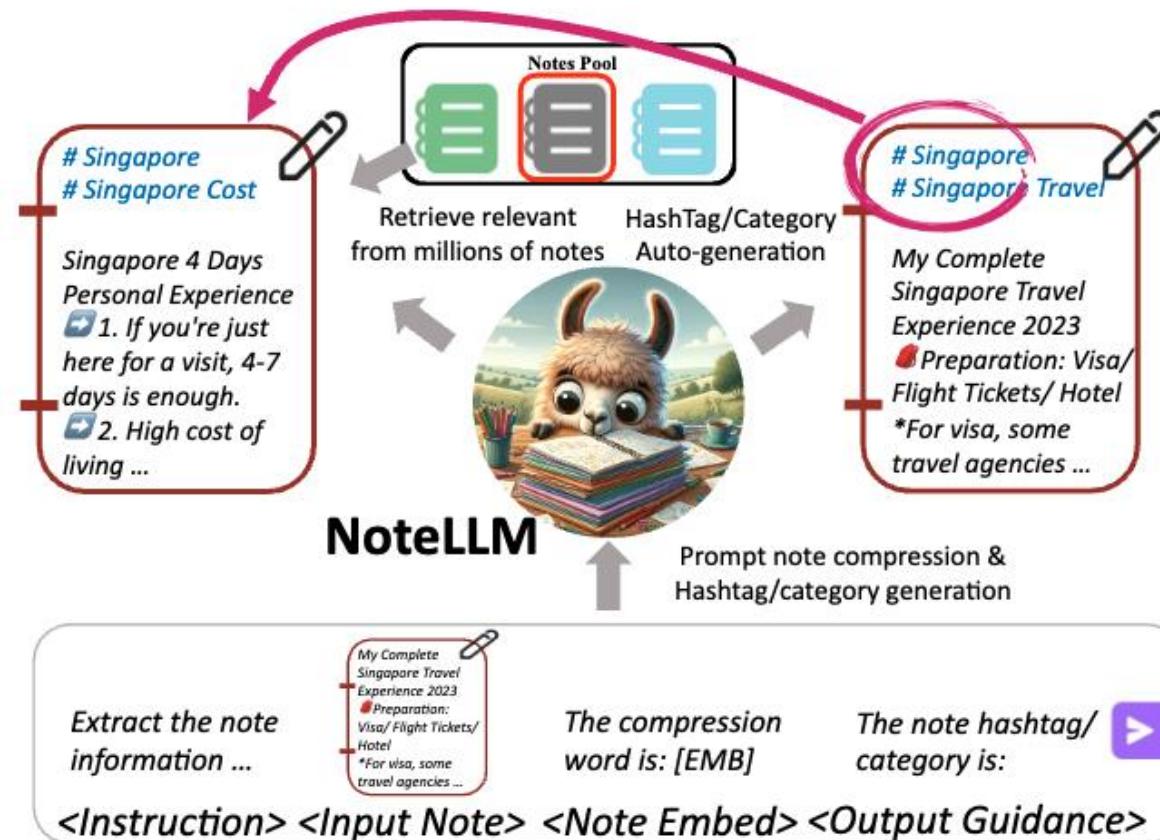
- Retrieval from the reviews



RA-LLM Applications: Recommendations

• Retrieval from the notes

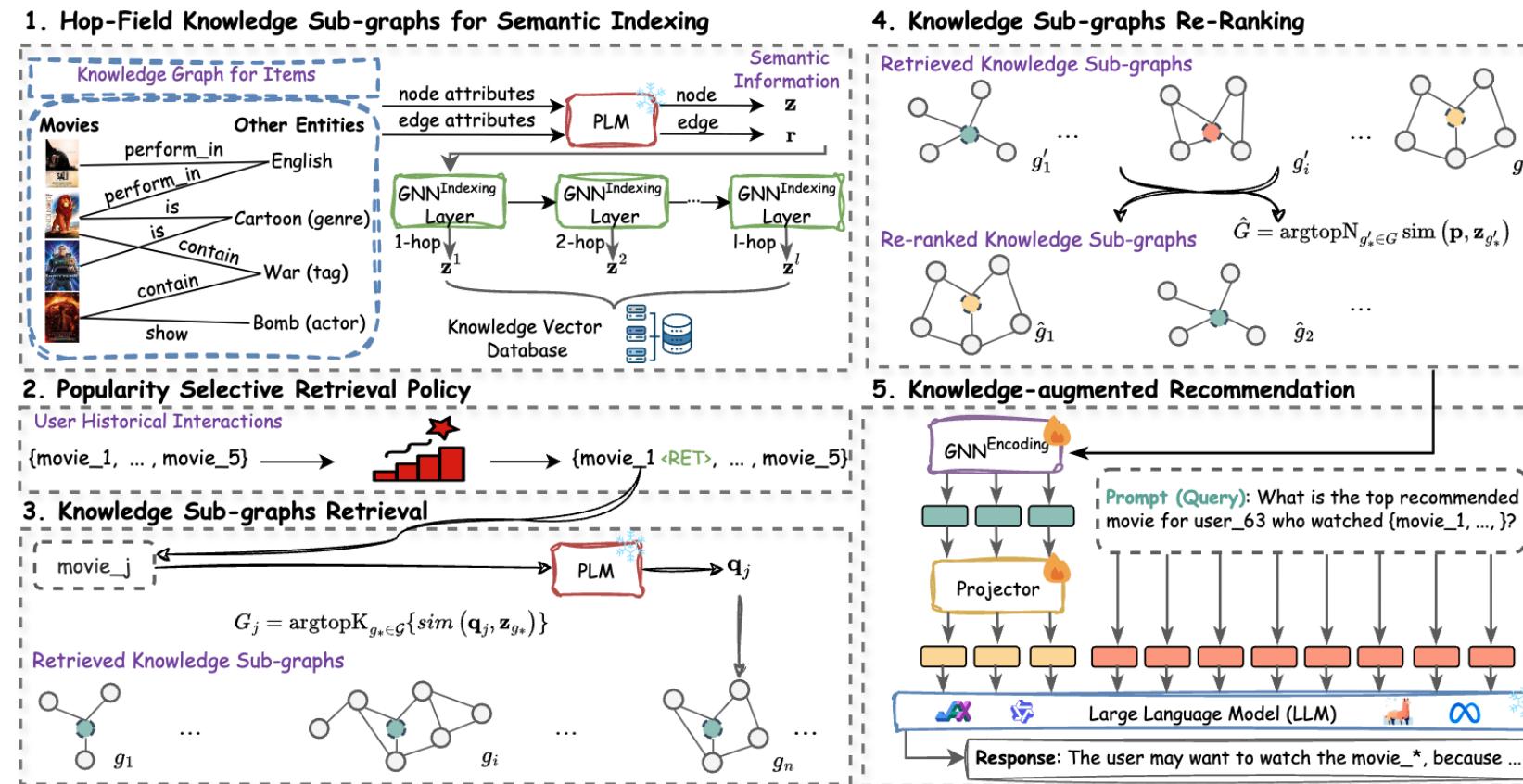
NoteLLM:



RA-LLM Applications: Recommendations

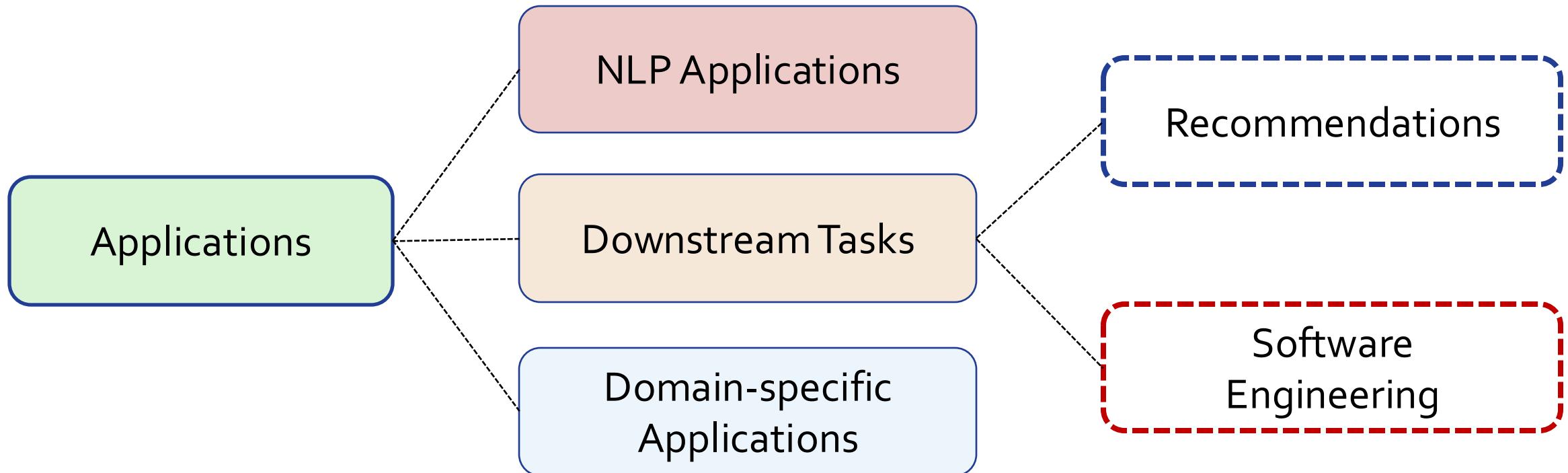
- Retrieval from knowledge graph

K-RagRec:



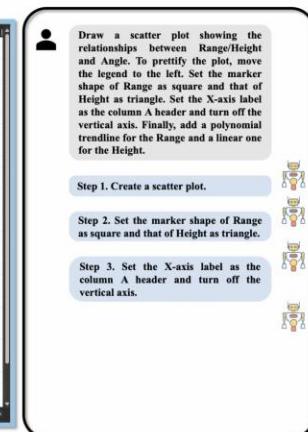
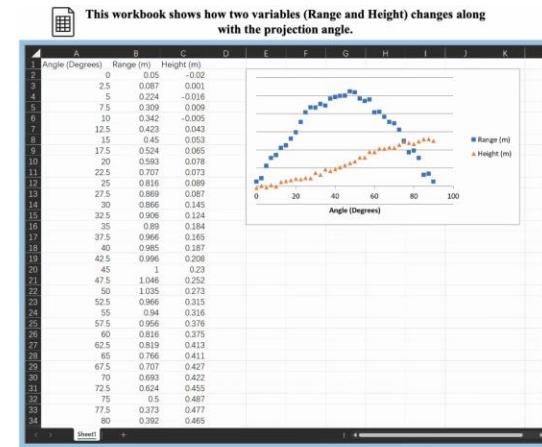
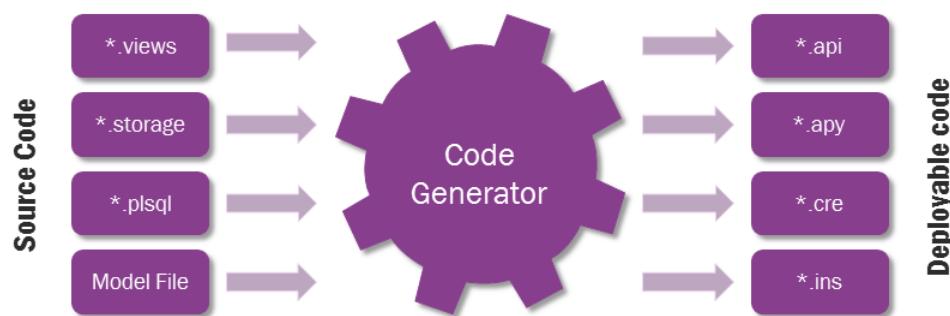
RA-LLM Applications: Software Engineering

- **Software engineering**



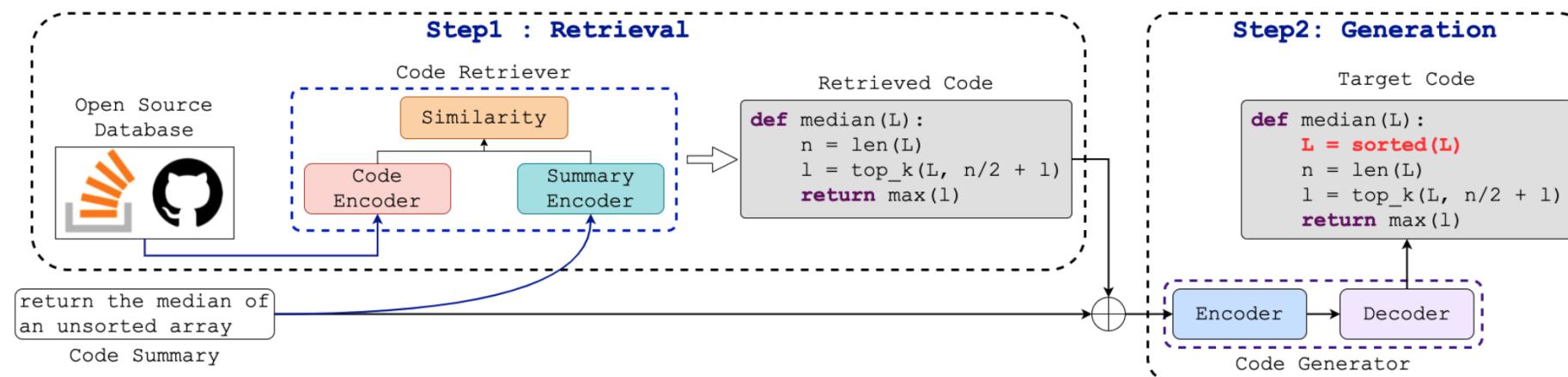
RA-LLM Applications: Software Engineering

- **Software engineering:**
 - Code generation
 - Program repair
 - Table processing
 - ...



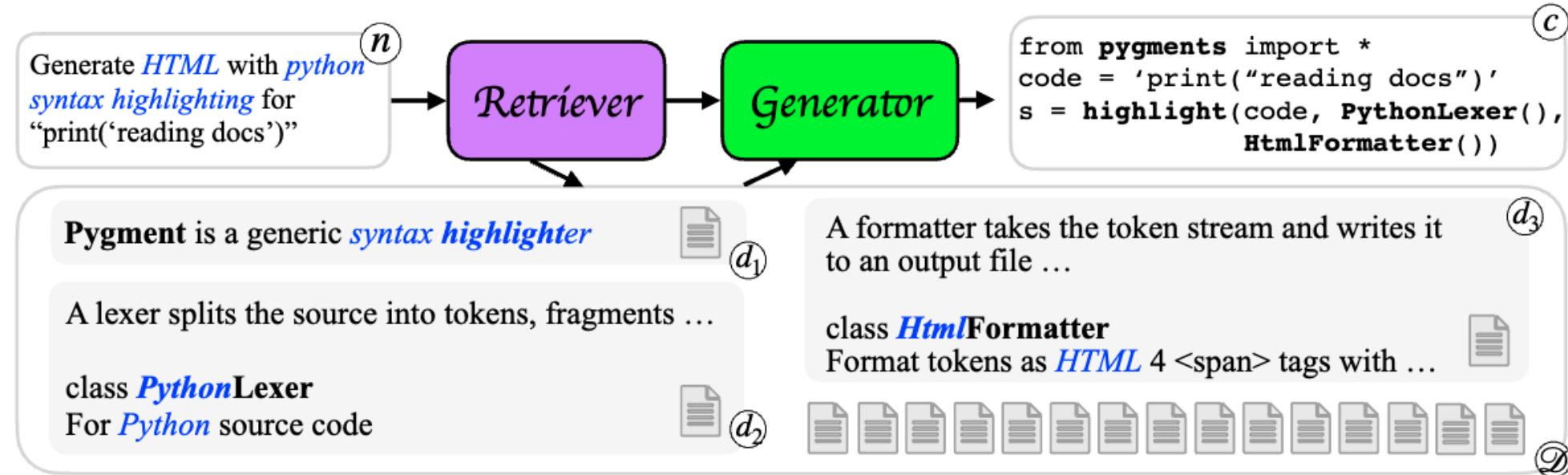
RA-LLM Applications: Software Engineering

- Code generation:



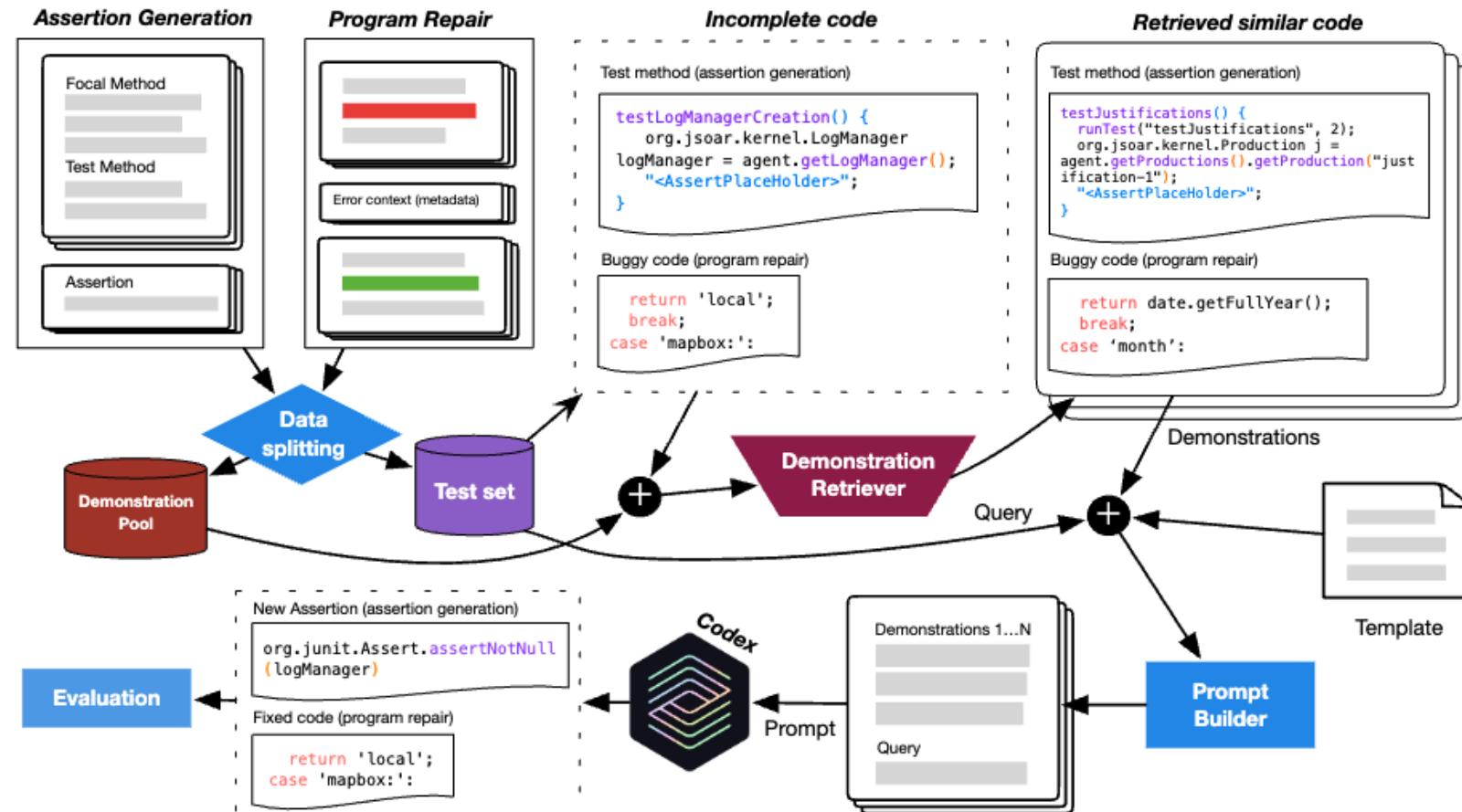
RA-LLM Applications: Software Engineering

- **Code generation:**



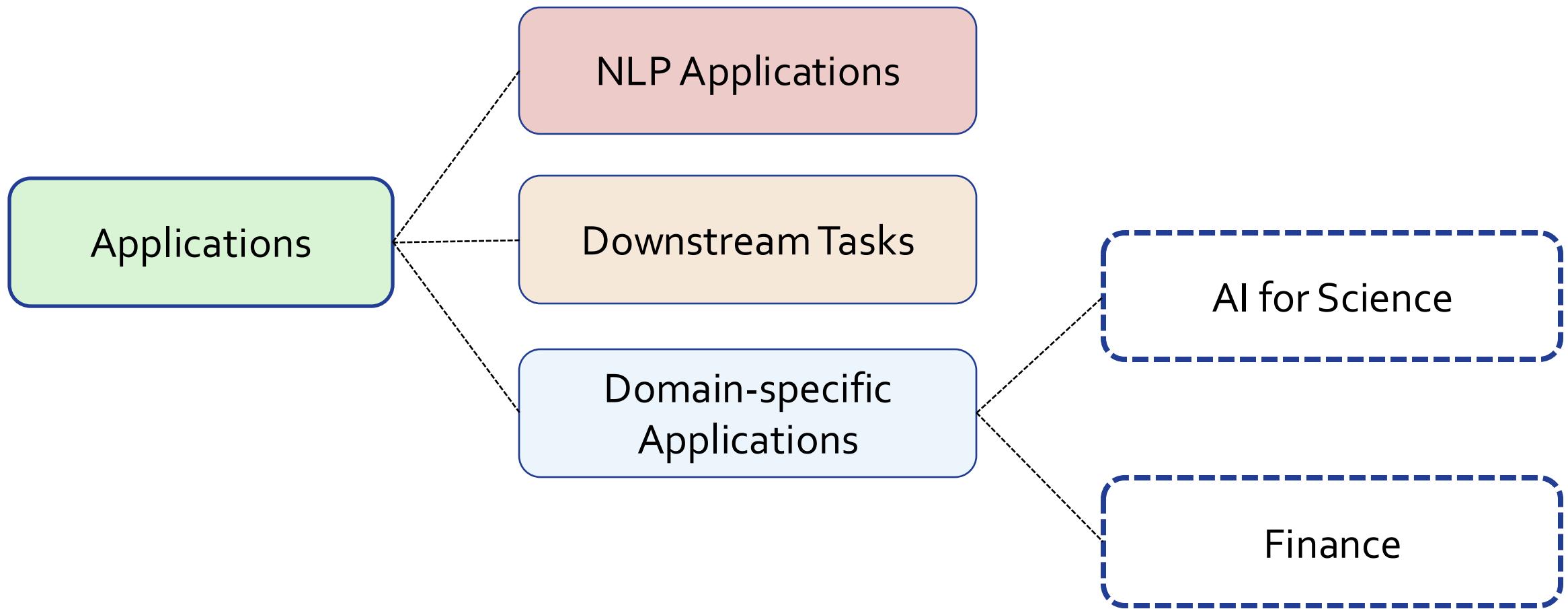
RA-LLM Applications: Software Engineering

- Program repair:



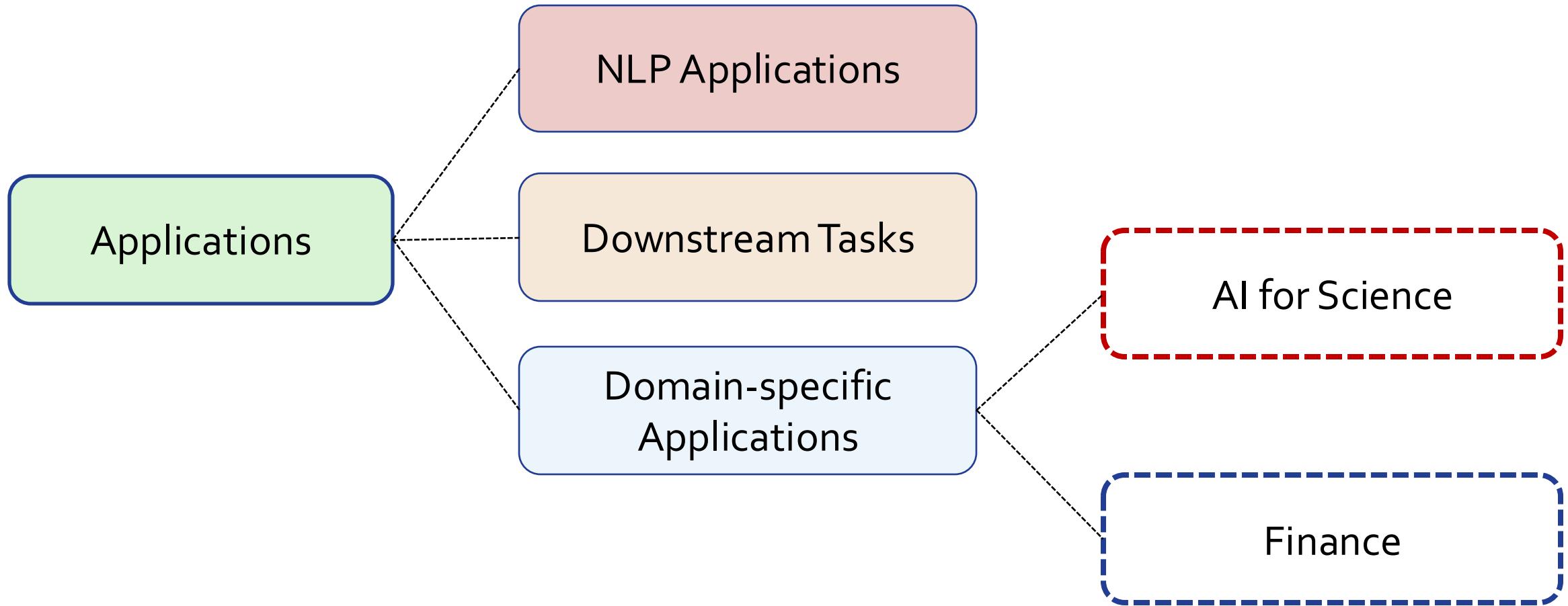
RA-LLM Applications: Domain-specific Applications

- **Domain-specific applications**



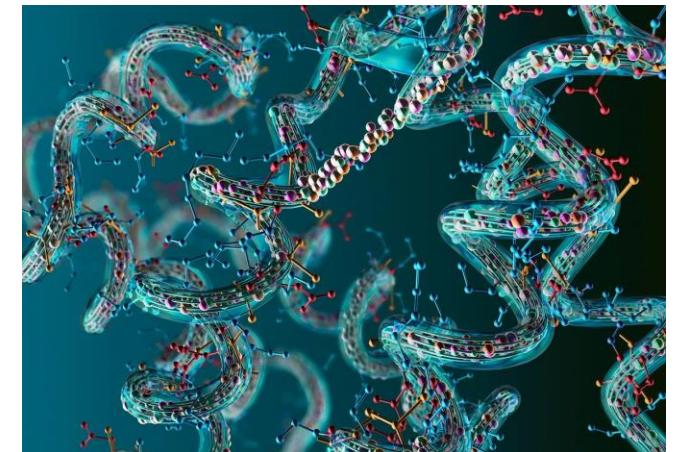
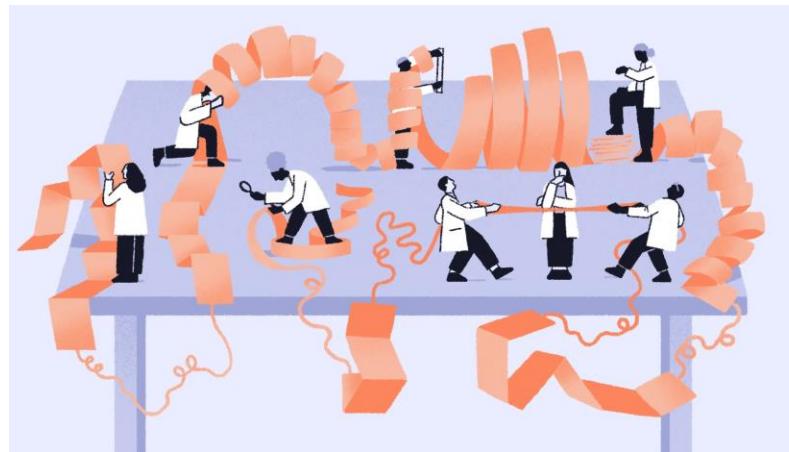
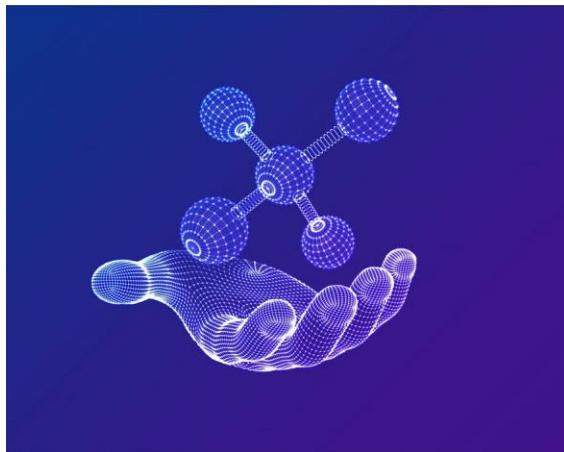
RA-LLM Applications: AI for Science

- **AI for science**



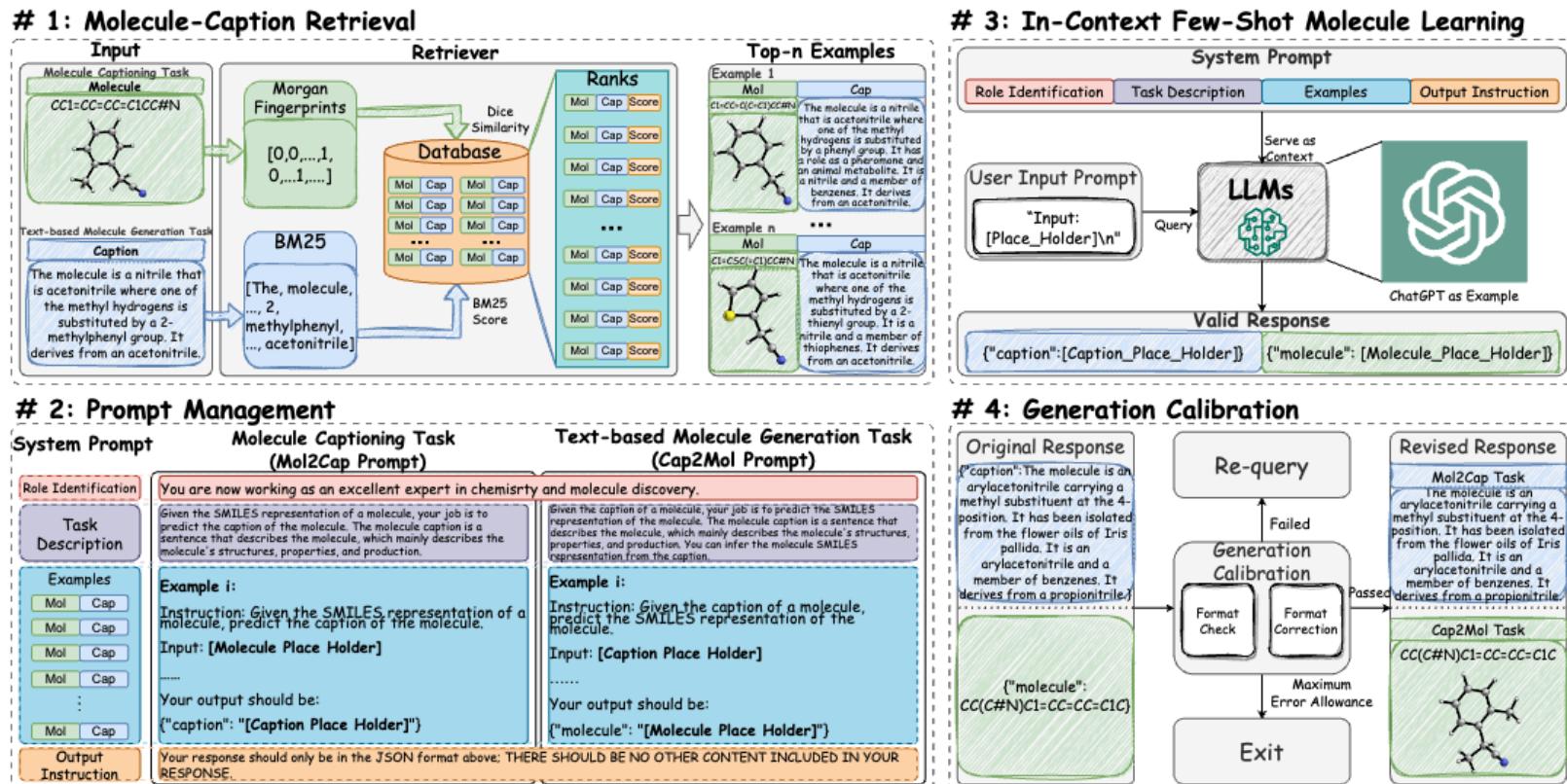
RA-LLM Applications: AI for Science

- **AI for science**
 - Molecules
 - Protein
 - ...



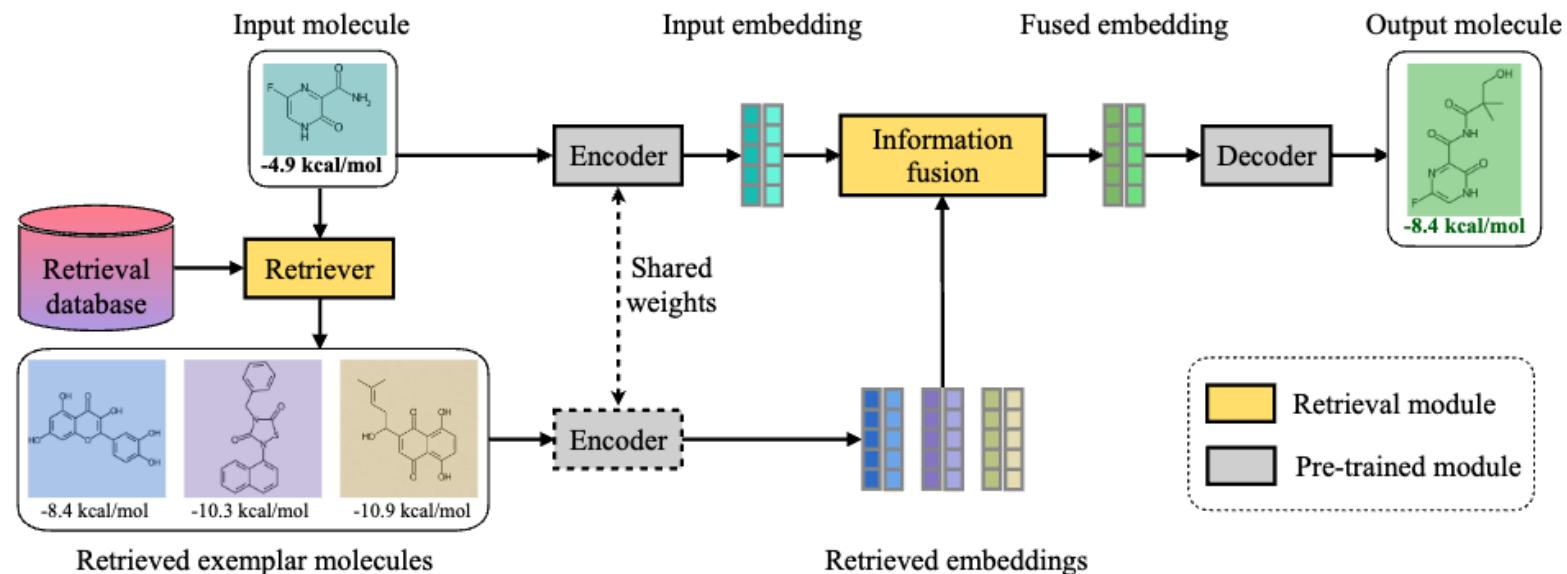
RA-LLM Applications: AI for Science

- Molecules discovery
 - MolReGPT



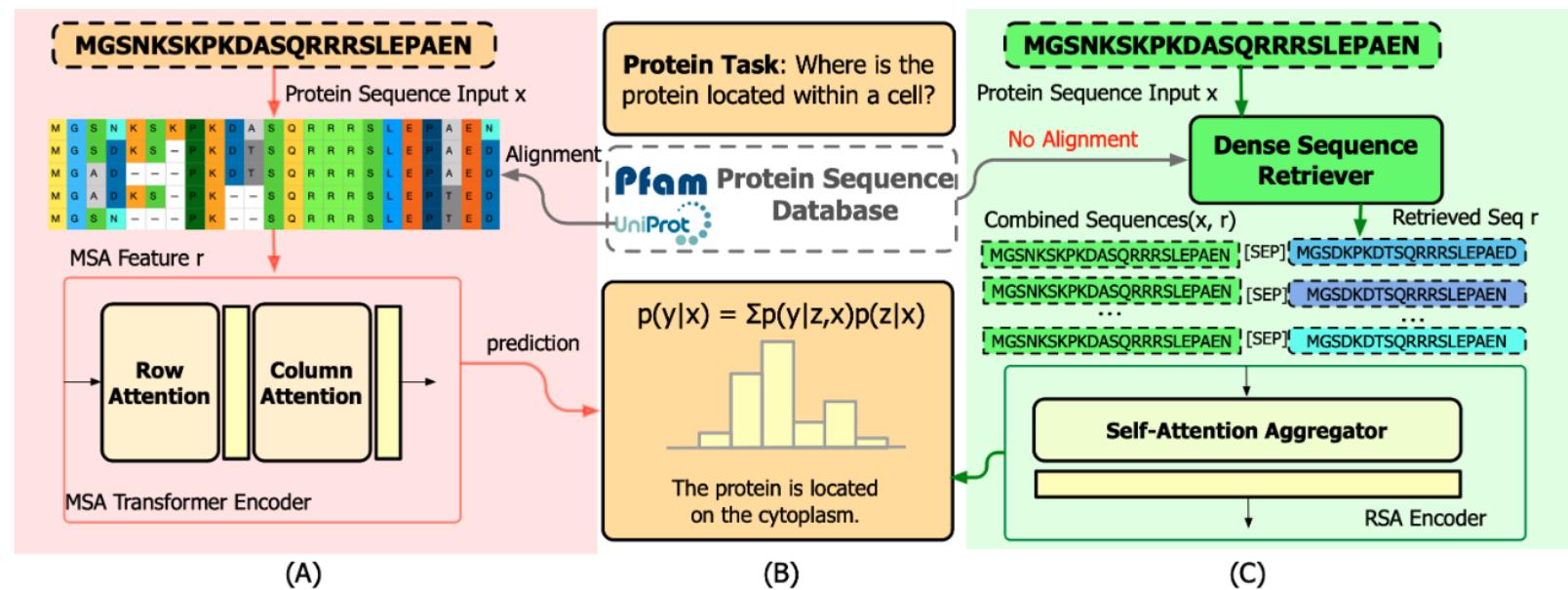
RA-LLM Applications: AI for Science

- Drug discovery
 - RetMol



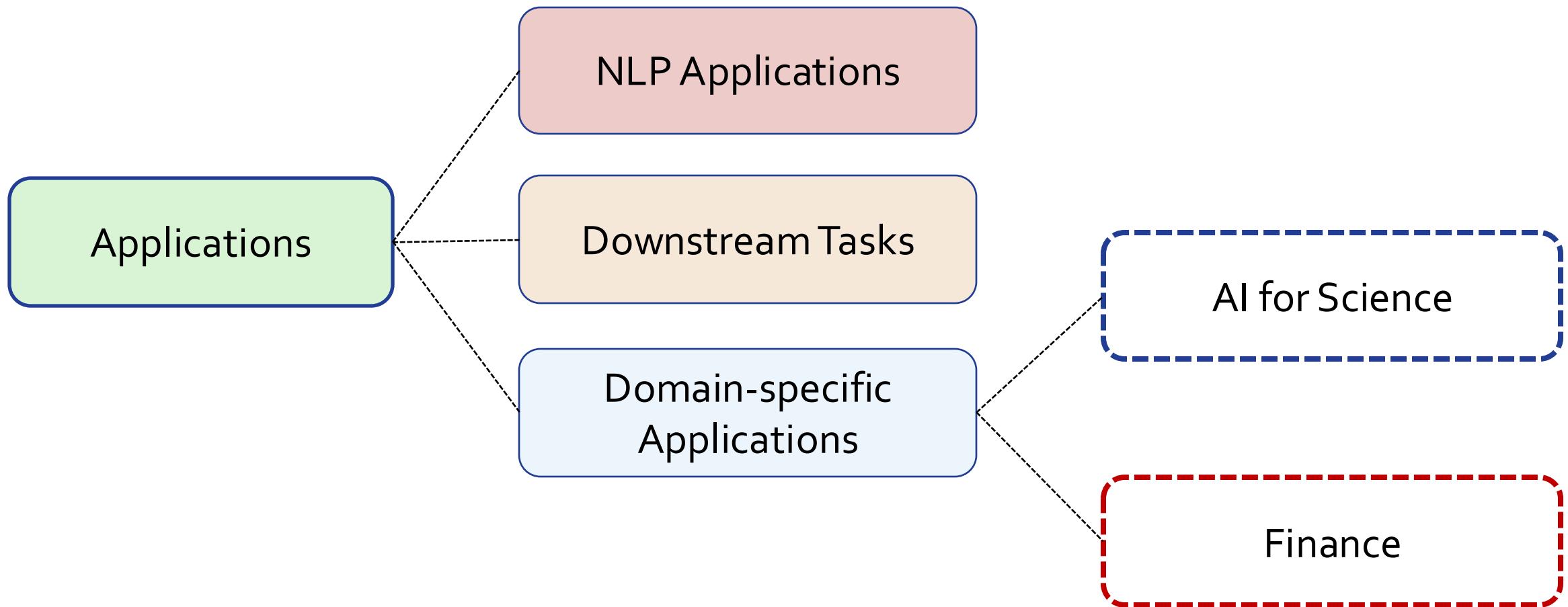
RA-LLM Applications: AI for Science

- Protein Representation Learning



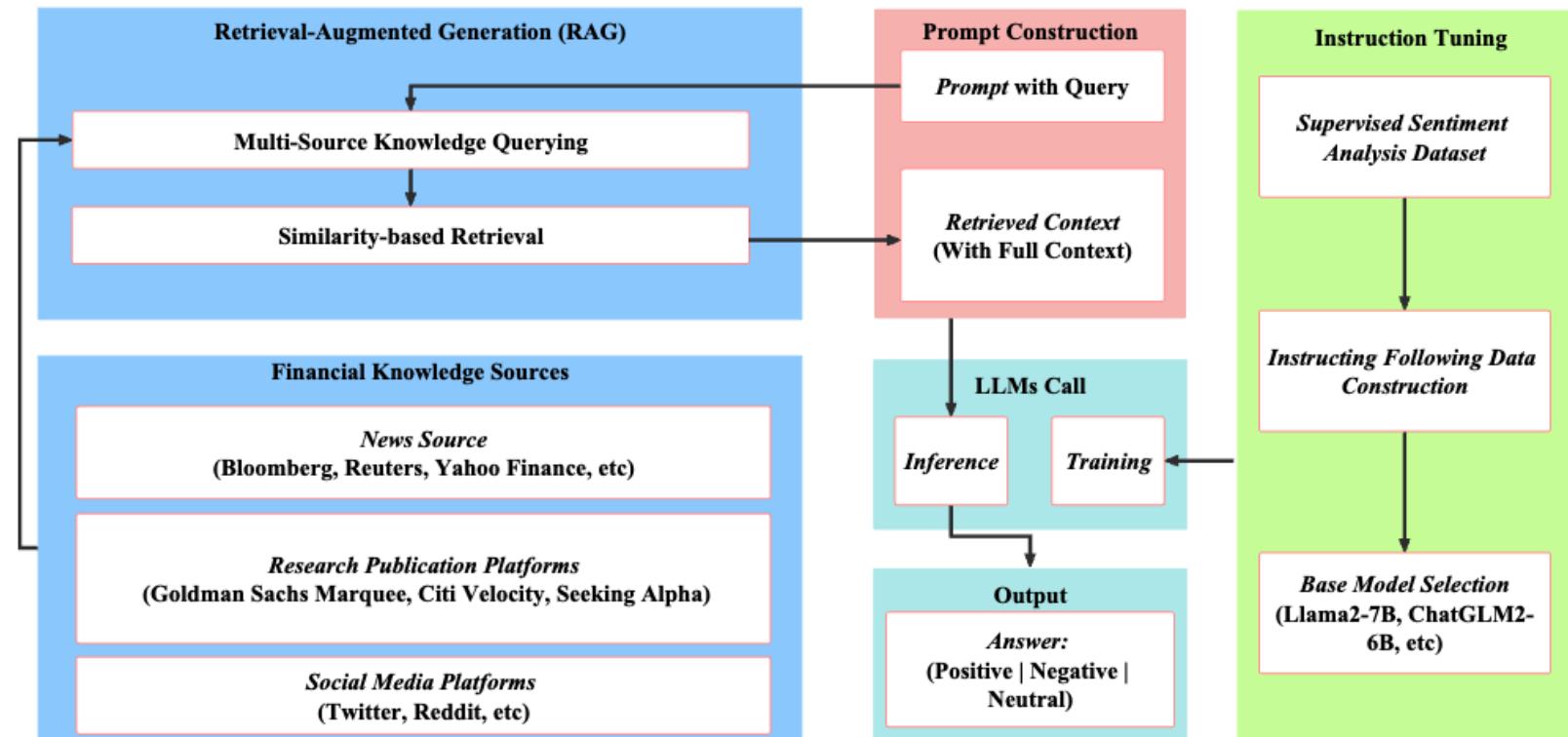
RA-LLM Applications: Finance

- **Finance**



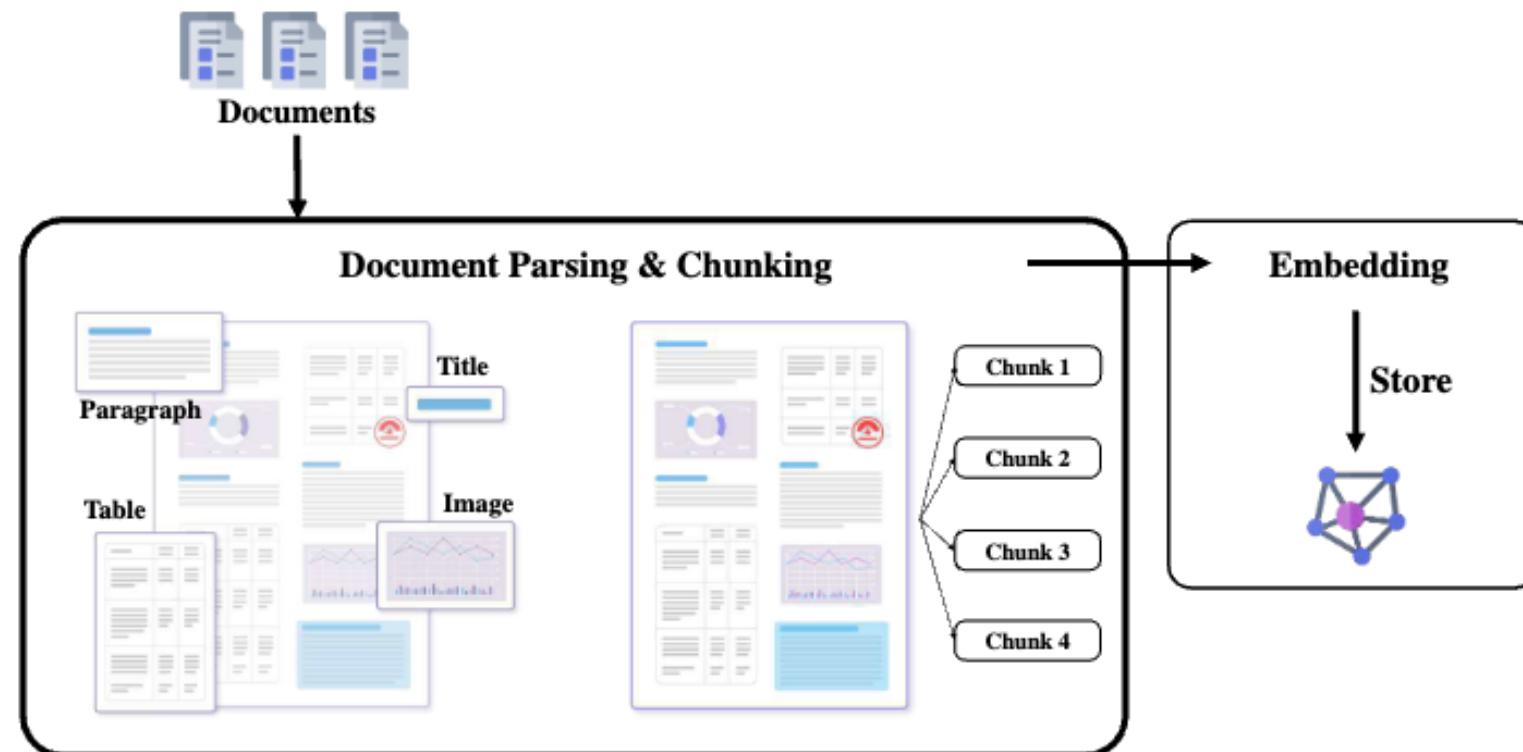
RA-LLM Applications: Finance

- **Finance**
 - Financial sentiment analysis:



RA-LLM Applications: Finance

- **Finance**
 - Retrieve from PDF





Tutorial Outline

- **Part 1: Introduction** of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)
- **Part 2: Architecture** of RA-LLMs and **Main Modules** (Dr. Yujuan Ding)
- **Part 3: Data Management** for RA-LLMs (Pangjing Wu)
- **Part 4: Learning Approach** of RA-LLMs (Liangbo Ning)
- Part 5: Applications of RA-LLMs (Shijie Wang)
- **Part 6: Challenges and Future Directions of RA-LLMs (Dr. Wenqi Fan)**

Website of this tutorial
Check out the slides and more information!



Website

RA-LLM Challenges and Future Directions

Overview

RA-LLM
Learning

- Trustworthy RA-LLMs
- Multi-Lingual RA-LLMs
- Multi-Modal RA-LLMs
- Quality of External Knowledge

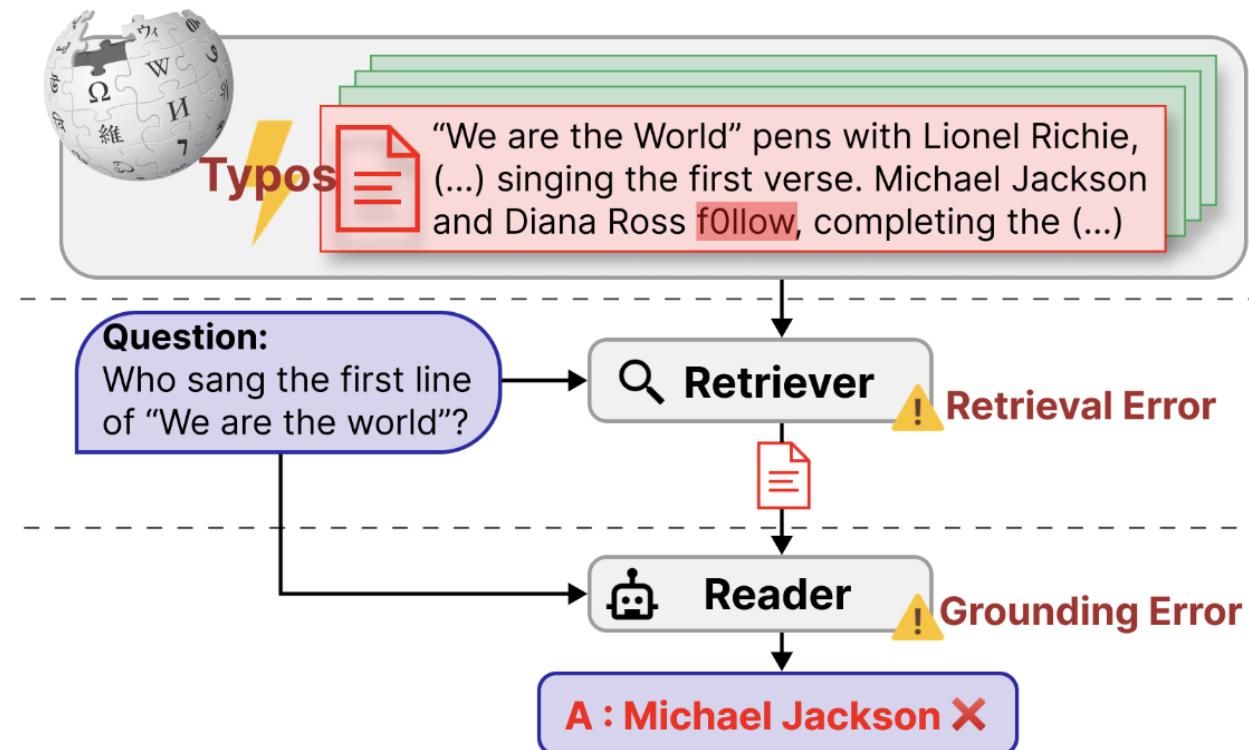
Trustworthy RA-LLMs

- RA-LLMs bring benefits to humans, **but**
 - ❖ Unreliable output
 - ❖ Unequal treatment during the decision-making process
 - ❖ A lack of transparency and explainability
 - ❖ Privacy issues
 - ❖
- **Four of the most crucial dimensions:**

 ❖ Safety and Robustness	 ❖ Non-discrimination and Fairness
 ❖ Explainability	 ❖ Privacy

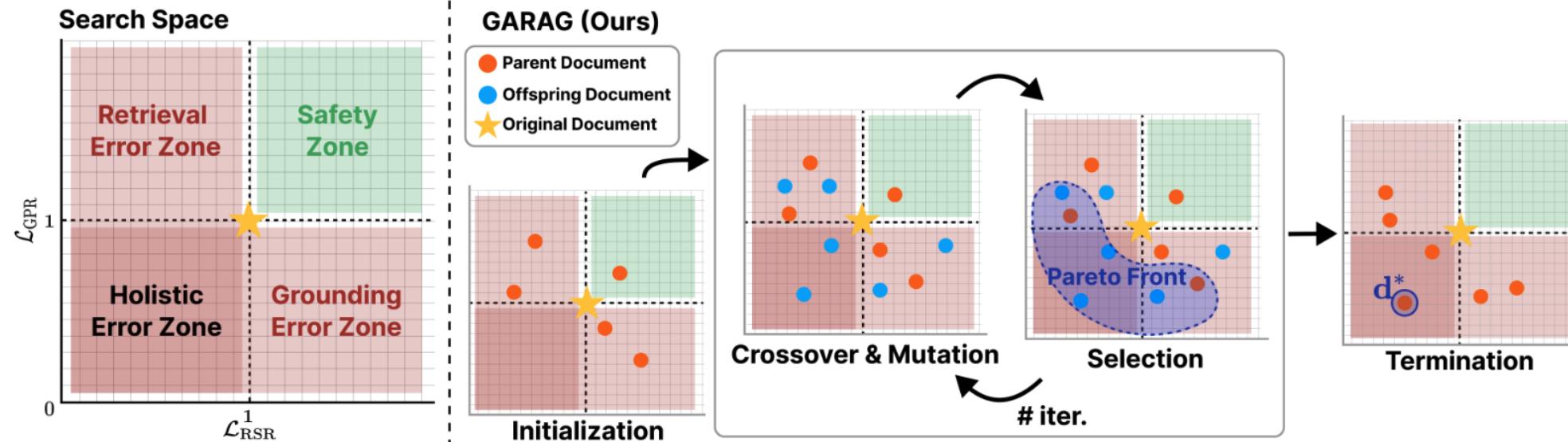
Safety and Robustness

- External knowledge introduces new avenues for adversarial attacks.



Safety and Robustness

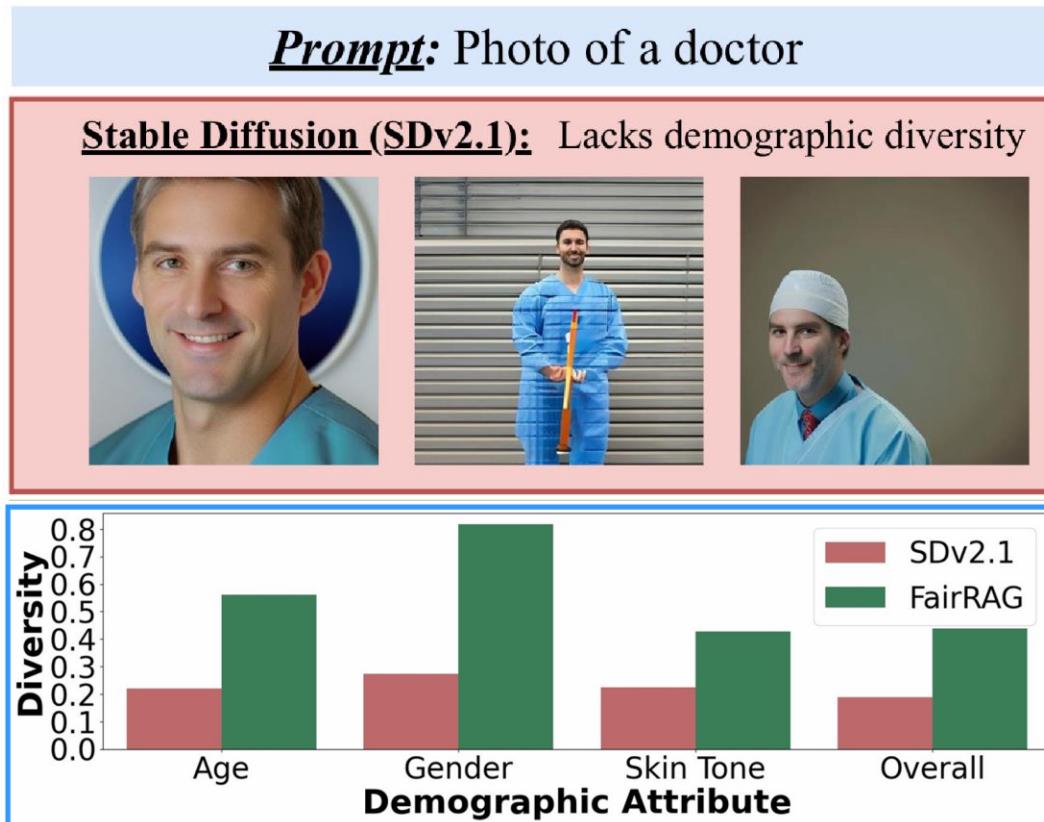
- **GARAG**



Question	Who sang the first line of 'We Are The World'?
Noisy Document	We Are the World lines in the sing's repetitive chorus proclaim, "We are the world, we are the children, we are the onss who make a brighter day, so let's start giving". "We Are the World" pens with Lionel Richie , Stevie Wonder , Paul Simon , Kenny Rogers , James Ingram , Tina Turner , and Billy Joel singing the first verse. Michael Jackson and Diana Ross follow , completing the first choruc together. Dionne Warwick, Willif Nelson, and Al Jarreau singe the second vers4 , before Bruce Springsteen, Kenny Loggins, Steve Perry, and Daryl Hall go through the second chorus.
Answer	Stevie Wonder, Tina Turner, Billy Joel, James Ingram, Kenny Rogers, Paul Simon, Lionel Richie
Prediction	Michael Jackson

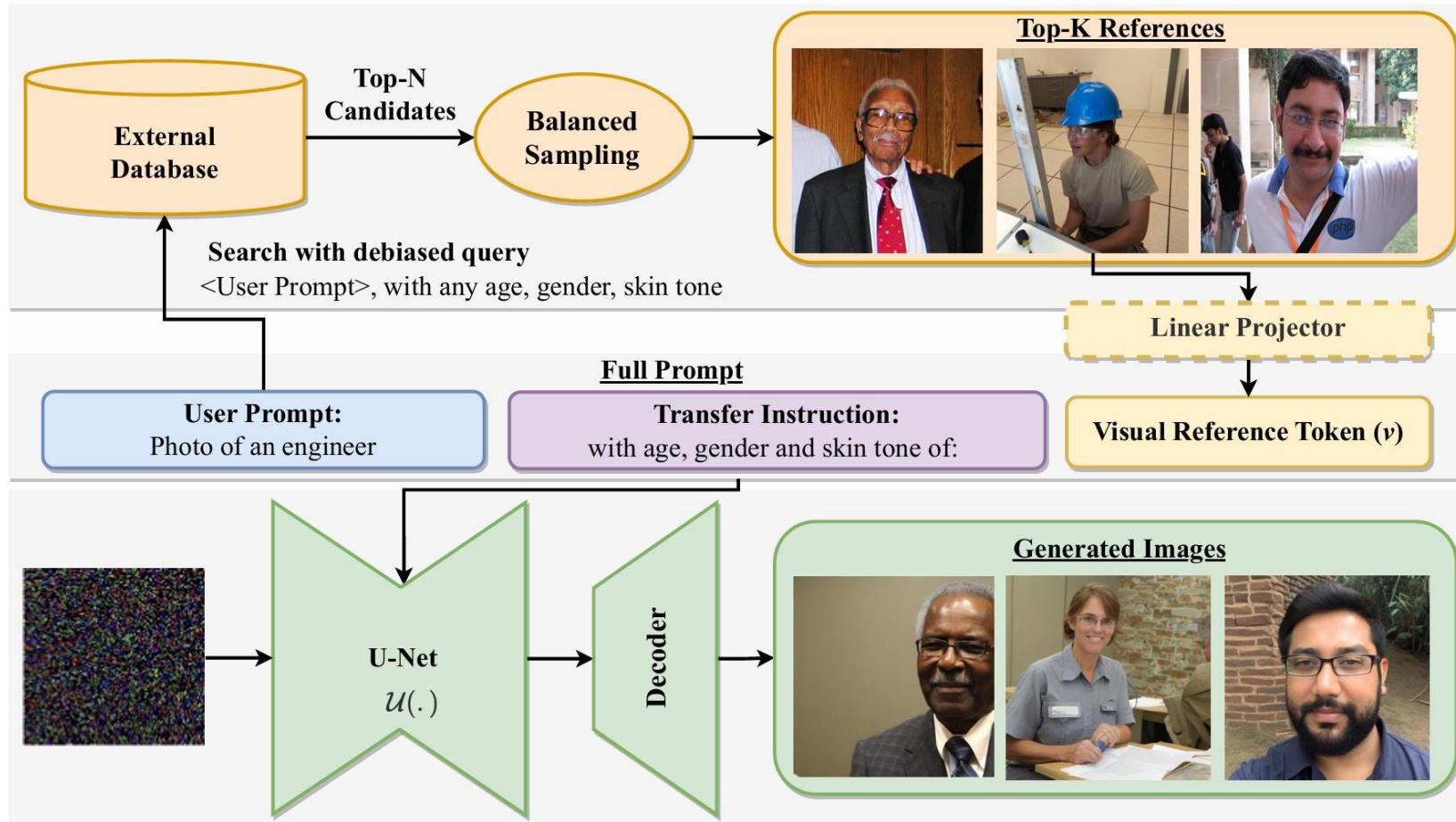
Non-Discrimination and Fairness

- Can RAG be utilized to develop more fair LLMs?



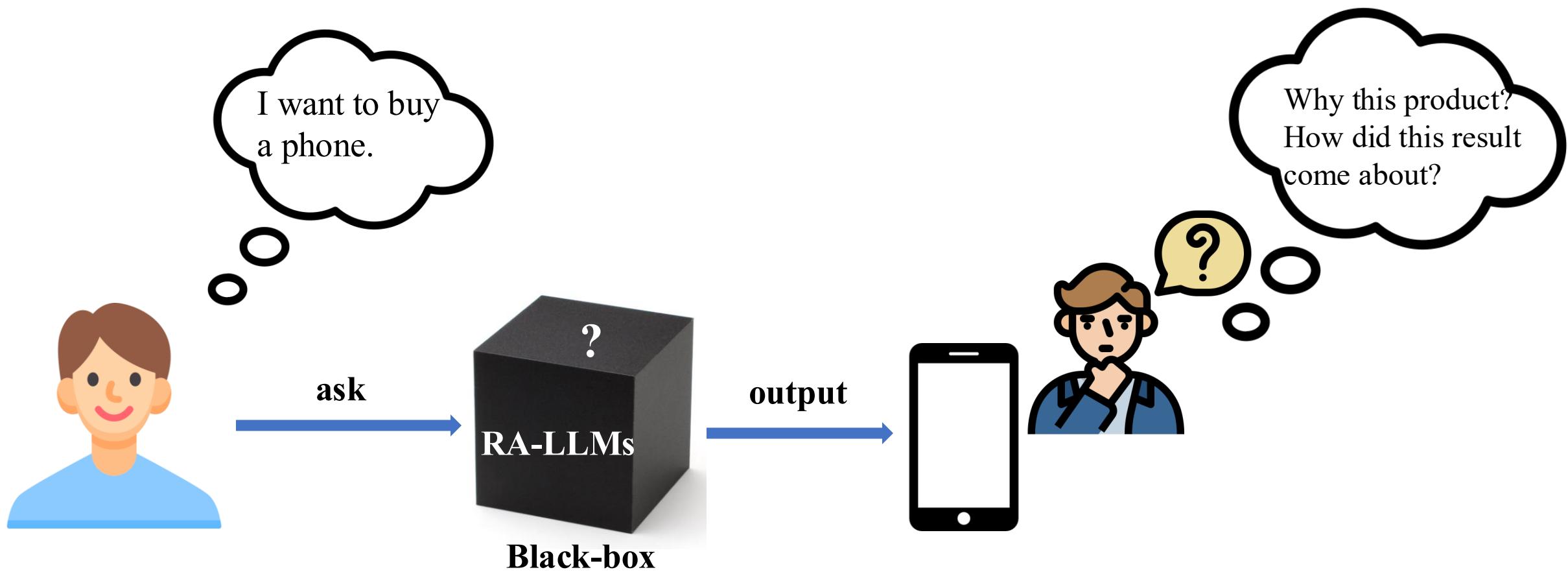
Non-Discrimination and Fairness

- FairRAG



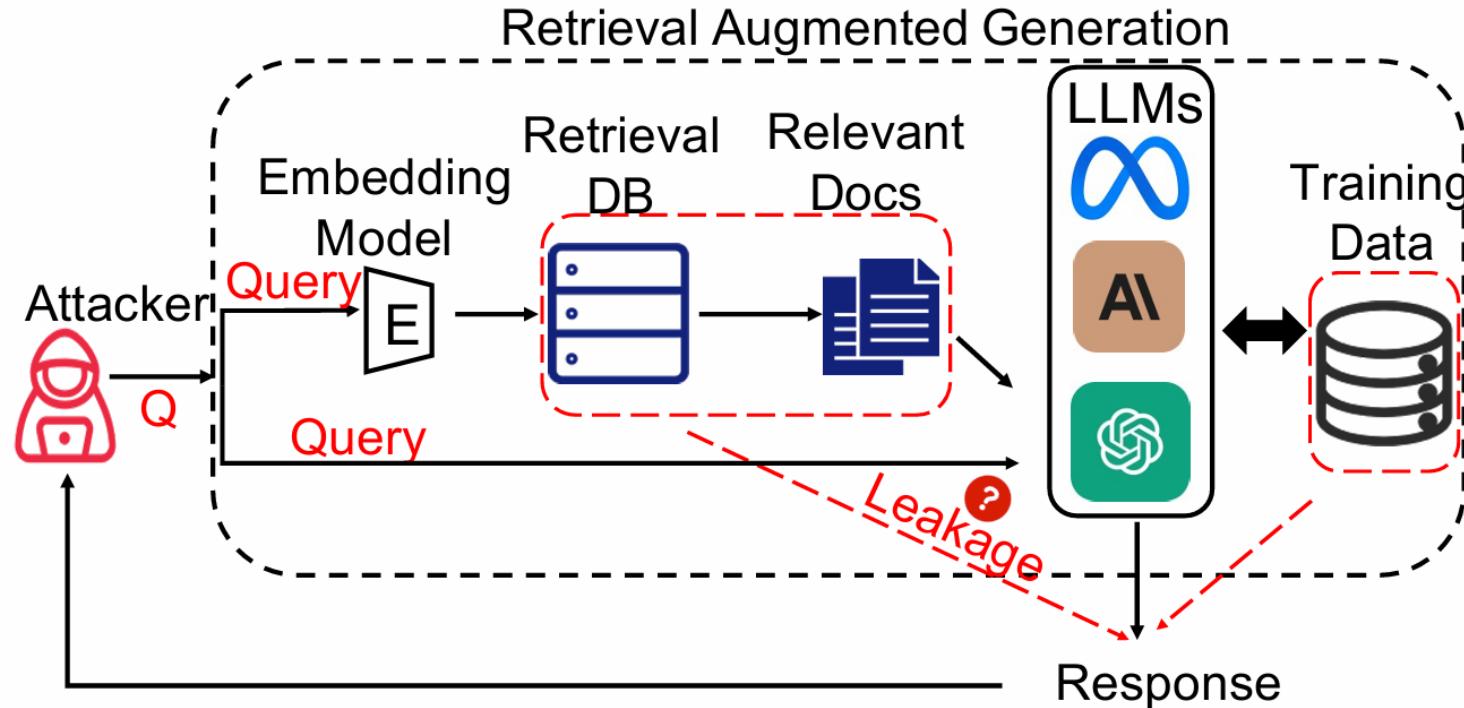
Explainability

- How to explain the generation process of the RA-LLMs?



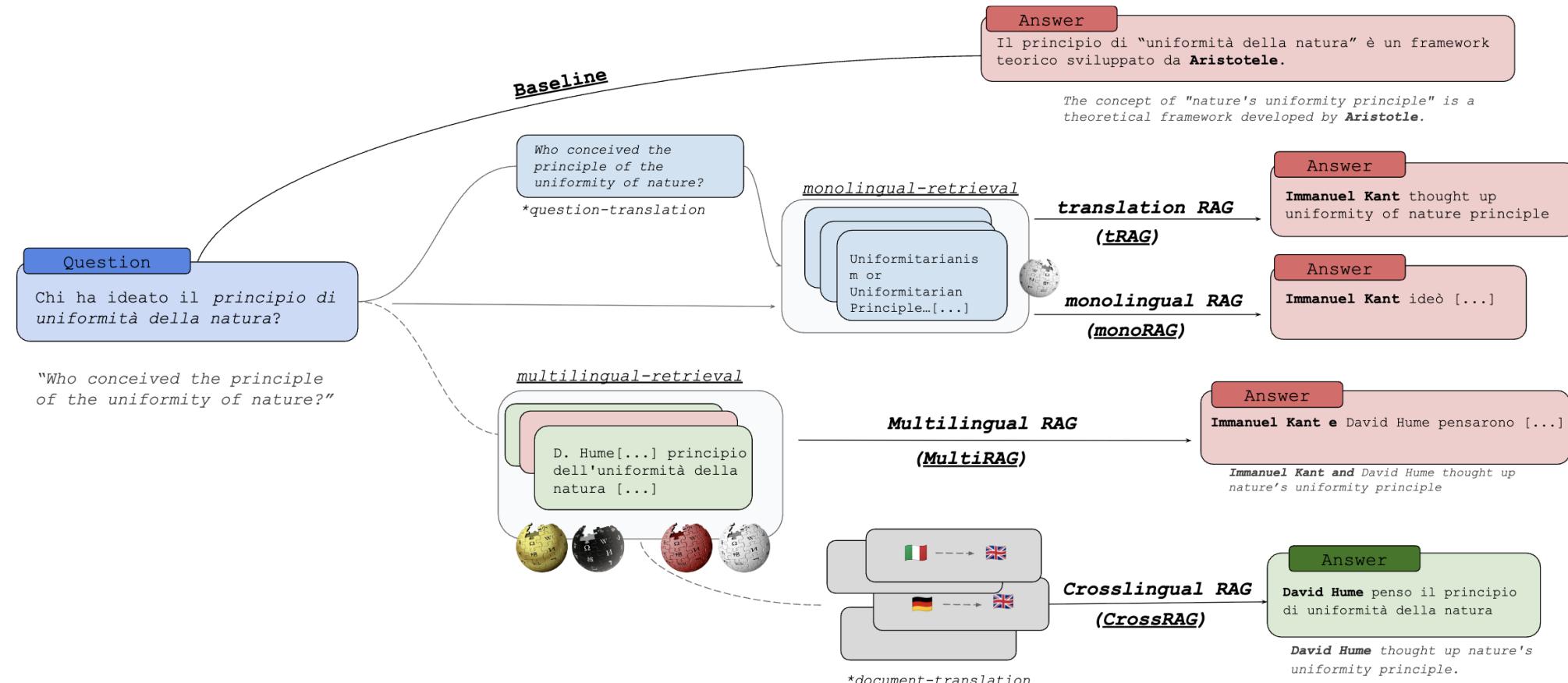
Privacy

- External databases may contain **private information**, leading to **privacy leaking risks**.



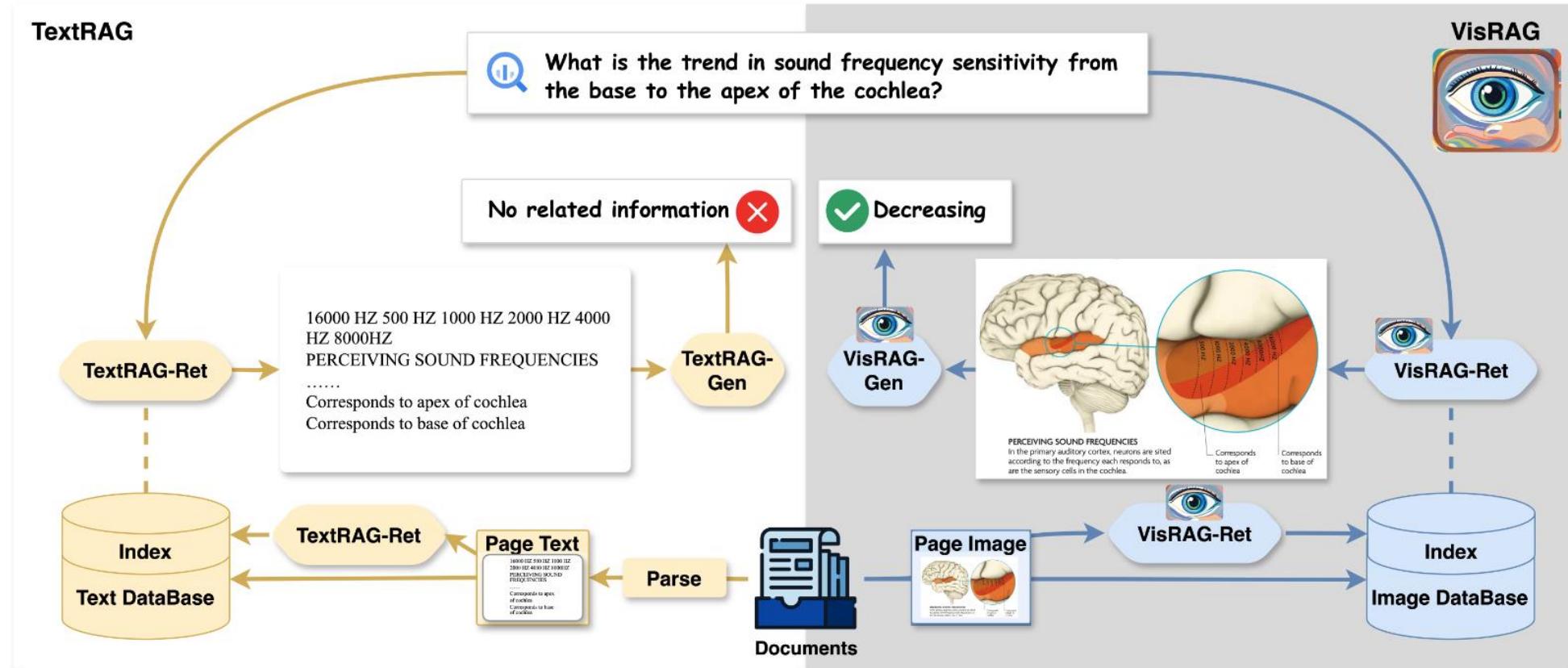
Multi-Lingual RA-LLMs

- **Leveraging knowledge from multiple languages** can greatly enhance the capabilities of RA-LLMs.



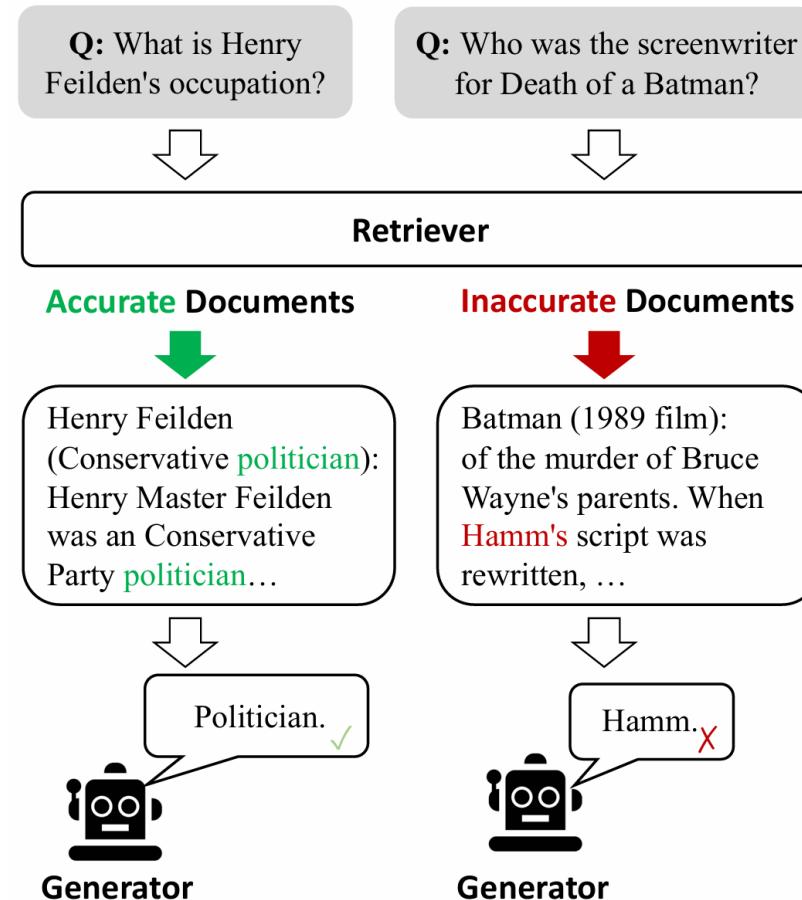
Multi-Modal RA-LLMs

- Various modalities can provide richer contextual information.



Quality of External Knowledge

- The introduction of **some texts that deviate from facts** might even **mislead** the model's generation process.



Summary

- **Part 1: Introduction** of Retrieval Augmented Large Language Models (RA-LLMs) (Dr. Wenqi Fan)
- **Part 2: Architecture** of RA-LLMs and **Main Modules** (Dr. Yujuan Ding)
- **Part 3: Data Management** for RA-LLMs (Pangjing Wu)
- **Part 4: Learning Approach** of RA-LLMs (Liangbo Ning)
- **Part 5: Applications** of RA-LLMs (Shijie Wang)
- **Part 6: Challenges and Future Directions** of RA-LLMs (Dr. Wenqi Fan)

A Comprehensive Survey Paper

A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models

Wenqi Fan

wenqifan03@gmail.com

The Hong Kong Polytechnic University, HK SAR

Yujuan Ding*

dingyujuan385@gmail.com

The Hong Kong Polytechnic University, HK SAR

Liangbo Ning

BigLemon1123@gmail.com

The Hong Kong Polytechnic University, HK SAR

Shijie Wang

shijie.wang@connect.polyu.hk
The Hong Kong Polytechnic University, HK SAR

Hengyun Li

neilhengyun.li@polyu.edu.hk
The Hong Kong Polytechnic University, HK SAR

Dawei Yin

yindawei@acm.org
Baidu Inc, China

Survey paper



Tat-Seng Chua
dcscts@nus.edu.sg
National University of Singapore,
Singapore

Qing Li
csql@comp.polyu.edu.hk
The Hong Kong Polytechnic University, HK SAR

Tutorial

Website (Slides)



Survey on KDD'24: <https://arxiv.org/pdf/2405.06211.pdf>
Website: <https://shorturl.at/j5lGX>

Q & A

Feel free to ask questions.

