IERG5050 AI Foundation Models, Systems & Applications
Spring 2025

Retrieval Augmented Generation (RAG) and Tool Use for Foundation Models

Prof. Wing C. Lau
wclau@ie.cuhk.edu.hk
http://www.ie.cuhk.edu.hk/wclau

Acknowledgements

Many of the slides in this lecture are adapted from the sources below. Copyrights belong to the original authors.

- Princeton COS597R: Deep Dive into Large Language Models, Fall 2024, by Prof. Danqi Chen and Sanjeev Arora, https://princeton-cos597r.github.io
- CMU 11-667: Large Language Models: Methods and Applications, Fall 2024, by Profs. Chenyan Xiong and Daphne Ippolito, https://cmu-llms.org
- CMU 11-711: Advanced Natural Language Processing (ANLP), Spring 2024, by Prof. Graham Neubig, https://phontron.com/class/anlp2024/lectures/ ;
- Akari Asai et al, ACL 2023 Tutorial: Retrieval-based Language Models and Applications, https://acl2023-retrieval-lm.github.io
- Akari Asai, Retrieval-augmented LMs: Past, Present and Future, guest lecture for CMU 11-667, https://akariasai.github.io/assets/pdf/akari_cmullm_ralm.pdf
- https://akariasai.github.io/assets/pdf/akari_cmu_anlp_lecture-compressed.pdf
- Wenqi Fan et al, KDD 2024 Tutorial: RAG Meets LLM: Towards Retrieval-Augmented Large Language Models, https://advanced-recommender-systems.github.io/RAG-Meets-LLMs/
- Deng Cai et al, IJCAI 2022 Tutorial: Recent Advances in Retrieval-Augmented Text Generation, https://lemaoliu.github.io/retrieval-generation-tutorial/
- Yunfan Gao et al, Retrieval-Augmented Generation for Large Language Models: A Survey: https://github.com/Tongji-KGLLM/RAG-Survey
- Princeton COS597G: Understanding Large Language Models, Fall 2022, by Prof. Danqi Chen, https://www.cs.princeton.edu/courses/archive/fall22/cos597G/
- Douwe Kiela, Guest Lecture for Stanford CS25: V3 I Retrieval Augmented Language Models, 2023 https://www.youtube.com/watch?v=mE7IDf2SmJg

Challenges and Risks of Deploying LLMs for Real-world Applications

☐ Hallucination

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

☐ Domain-specific knowledge & expertise

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

☐ 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.

☐ Inconsistency

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

LLMs Struggle in Domain-specific Knowledge

OXFORD

Journal of Legal Analysis, 2024, **16**, 64–93

https://doi.org/10.1093/jla/laae003

Advance access publication 26 June 2024

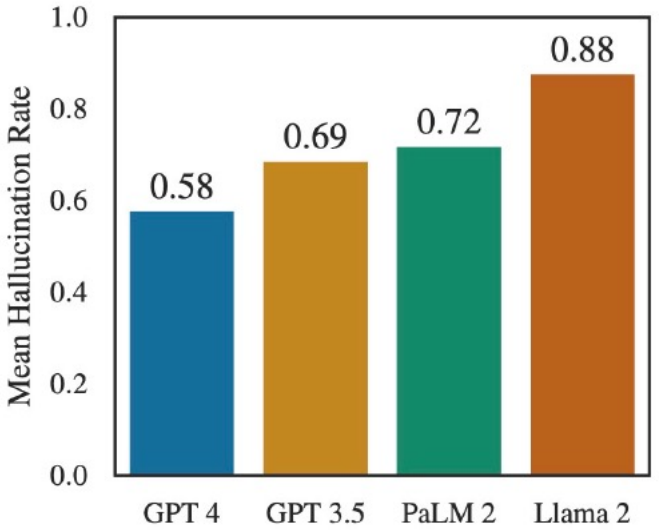
Article

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



Source: Dahl M, et al. 2024, Large legal fictions: Profiling legal hallucinations in large language models.

Serious Consequences of LLM Hallucinations

Air Canada must honor refund policy invented by airline's chatbot

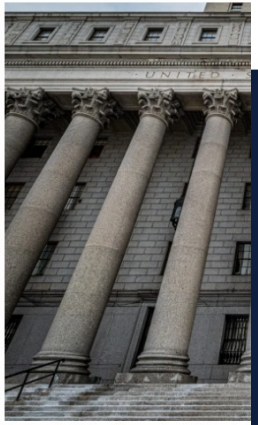
Air Canada appears to have quietly killed its costly chatbot support.

ASHLEY BELANGER - 2/16/2024, 12:12 PM

TECH · LAW

Humiliated lawyers fined $5,000 for submitting ChatGPT hallucinations in court: 'I heard about this new site, which I falsely assumed was, like, a super search engine'

BY RACHEL SHIN
June 23, 2023 at 9:41 AM PDT



Lawyers who filed legal documents with false citati

ERIK MCGREGOR—LIGHTROCKET/GETTY IMAGES

MIT Technology Review

Featured Topics Newsletters Events Podcasts SIGN IN

ARTIFICIAL INTELLIGENCE

Why Meta's latest large language model survived only three days online

Galactica was supposed to help scientists. Instead, it mindlessly spat out biased and incorrect nonsense.

By Will Douglas Heaven November 18, 2022

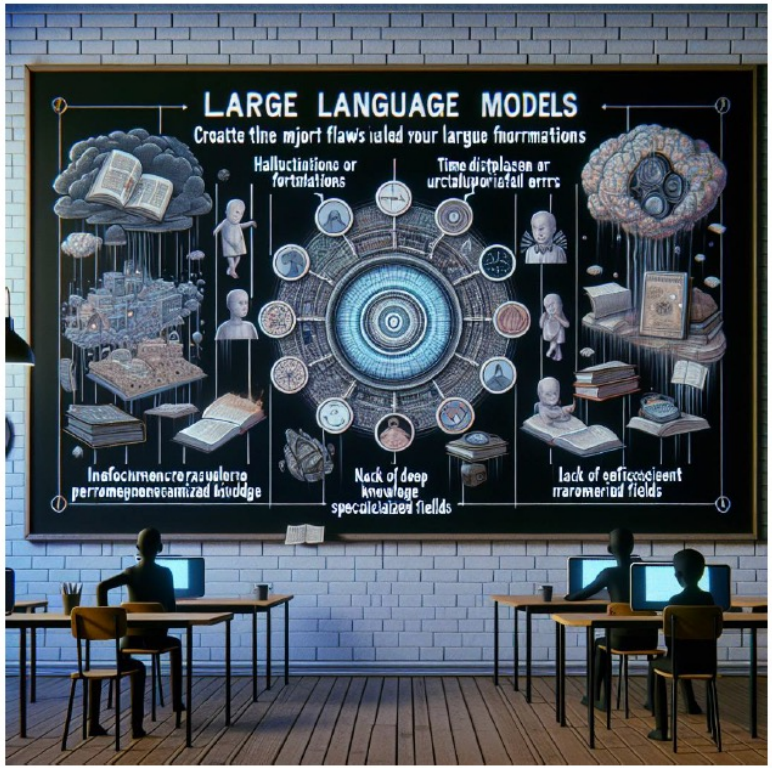
Big Problems when Deploying LLMs for Production-use

Drawbacks of LLMs

- Hallucination
- Outdated information
- Low efficiency in parameterizing knowledge
- Lack of in-depth knowledge in specialized domains
- Weak inferential capabilities

Practical Requirements of Application

- Domain-specific accurate answering
- Frequent updates of data
- Traceability and explainability of generated content
- Controllable Cost
- Privacy protection of data



Draw by DALL·E-3

“Prehistoric” Retrieval Augmented Generation (b4 LLM era)

- **Retrieve** relevant passages efficiently
- **Read** the passages to answer the query

Open-domain QA
SQuAD, TREC, WebQuestions, WikiMovies

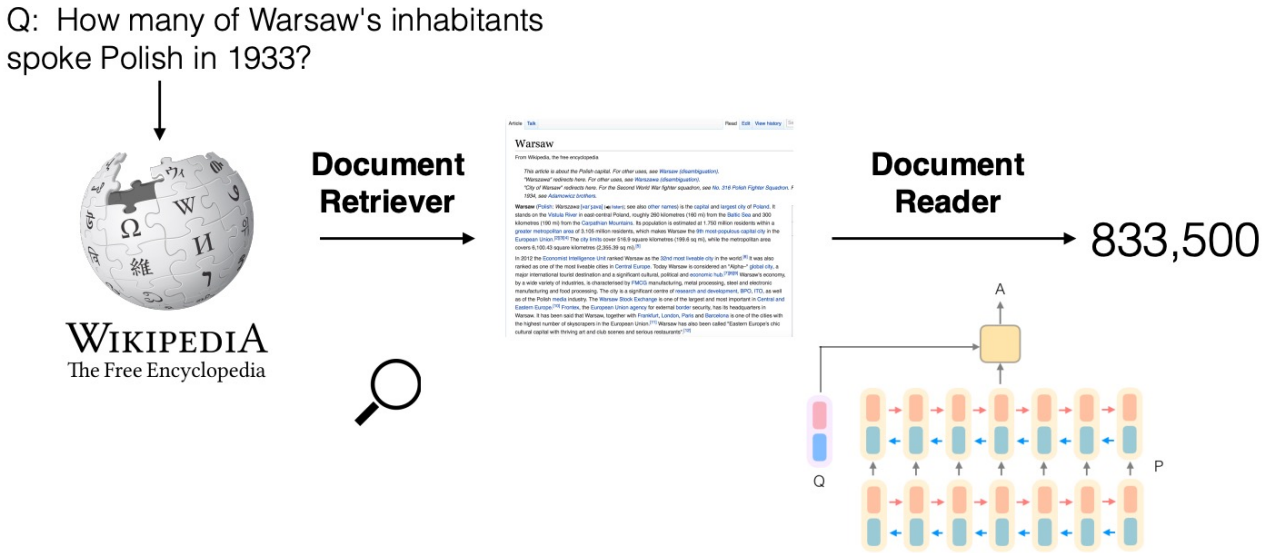
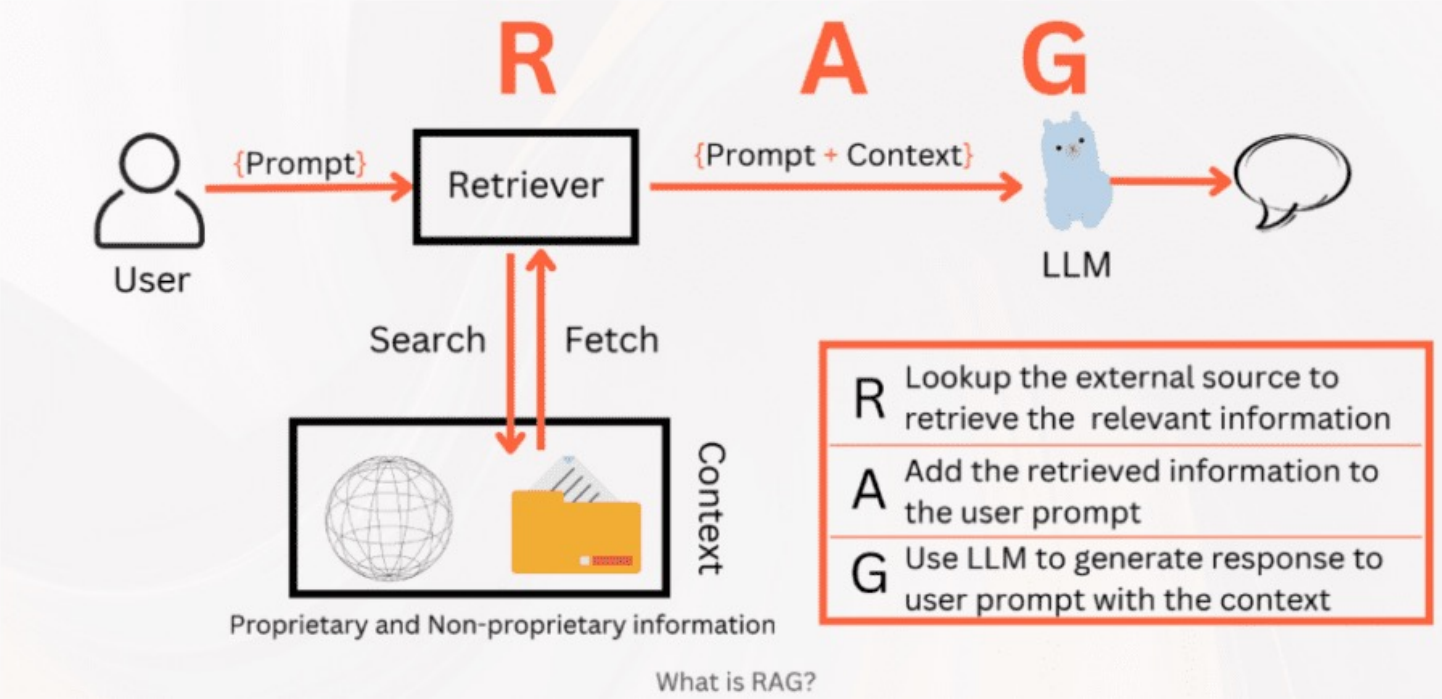


Figure 1: An overview of our question answering system DrQA.

Source: DrQA (Chen et al., 2017)

RAG meets LLMs

Default Retrieval Augmented Generation (RAG) Framework



Source: https://www.datasciencecentral.com/rag-and-its-evolution/

Retrieval Augmented Generation (RAG) can help LLMs !

RAG can help to address many core limitations of parametric LMs:

- Long-tail distribution of data
- Frequent knowledge updates
- Answers requiring verification and traceability
- Specialized domain knowledge
- Data privacy preservation

Q&A
RETRO (Borgeaud et al2021)
REALM (Gu et al, 2020)
ATLAS (Izacard et al, 2023)

Fact Checking
RAG (Lewis et al, 2020)
ATLAS (Izacard et al, 2022)
Evi. Generator (Asai et al, 2022a)

Dialog
BlenderBot3 (Shuster et al.2022)
Internet-augmented generation
(Komeili et a., 2022)

Summary
FLARE (Jiang et al, 2023)

Machine Translation
kNN-MT (Khandelwal et al., 2020)TRIME-MT (Zhong et al., 2022)

Code Generation
DocPrompting (Zhou et al., 2023
Natural ProverWelleck et al., 2022)

Natural Language Inference
kNN-Prompt (Shi et al., 2022)
NPM (Min et al., 2023)

Sentiment analysis
kNN-Prompt (Shi et al., 2022)NPM (Min et al., 2023)

Commonsense reasoning
Raco (Yu et al, 2022)

(Basic) Retrieval Augmented Generations (RAG)

Step 1: Retrieve K documents

Prompt How did US states get their names?



Retriever

❶ Of the fifty states, eleven are named after an individual person.

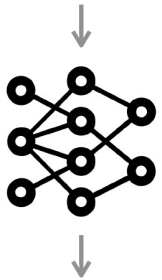
❷ Popular names by states. In Texas, Emma is a popular baby name.

❸ California was named after a fictional island in a Spanish book.

Retrieve

Step 2: Prompt LM with K docs and generate

Prompt How did US states get their names? ❶❷❸



LM

US states got their names from a variety of sources. Eleven states are named after an individual person (e.g, California was named after Christopher Columbus). Some states including Texas and Utah, are named after …

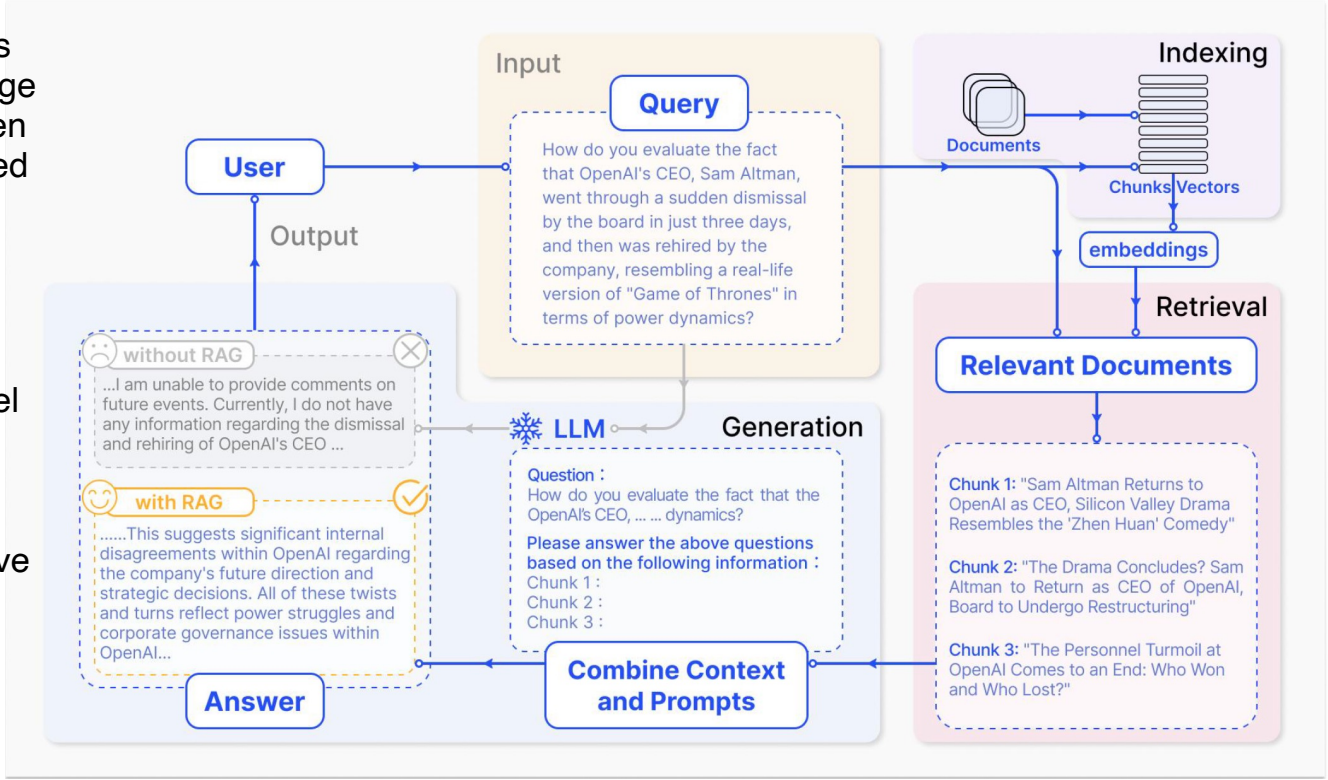
Read

(Basic) Retrieval Augmented Generations (RAG) – more details

When answering questions or generating text, it first retrieves relevant information from a large number of documents, and then LLMs generates answers based on this information.

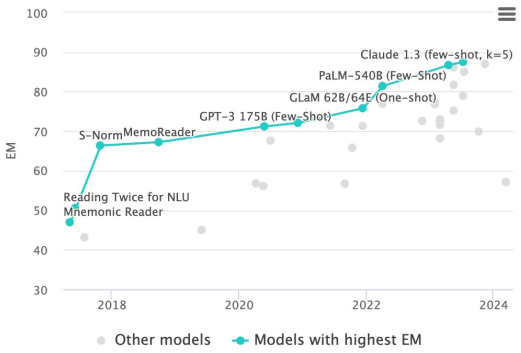
By attaching an external knowledge base, there is no need
to retrain the entire large model for each specific task.

The RAG model is especially suitable for knowledge-intensive tasks.



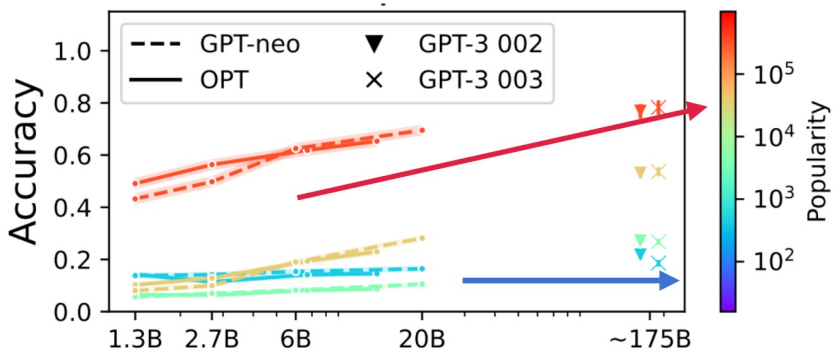
RAG for more Reliable, Adaptable and Attributable LLMs

Fact: Scaling-up of LLMs does NOT solve memorization problem regarding Long-tail facts



On popular facts Performance gets better as scaling.

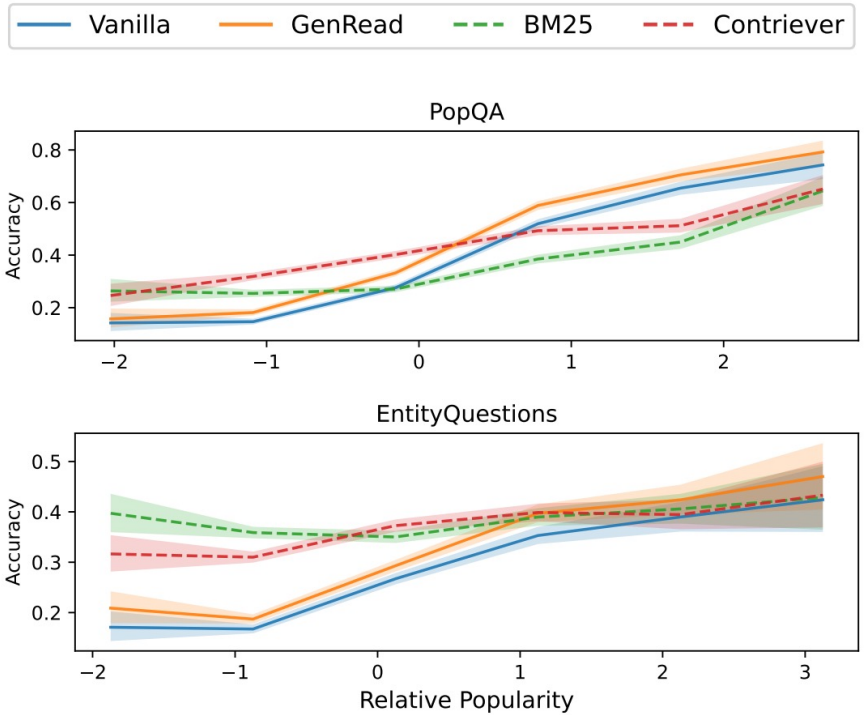
On long-tail facts Almost flat trends.



Scaling may not overcome hallucination in long tail!

Source: Alex Mallen, Akari Asai et al, "When Not to Trust Language Models: ...", ACL 2023

RAG can address Hallucinations regarding Long-tail facts



More later on why and how to fix this ?

On popular facts vanilla ≈ RAG or vanilla > RAG

On long-tail facts RAG > vanilla

RAG gives large gain in long-tail questions!

Source: Alex Mallen, Akari Asai et al, "When Not to Trust Language Models: ...", ACL 2023

RAG for Attributable LLMs

Hallucinations

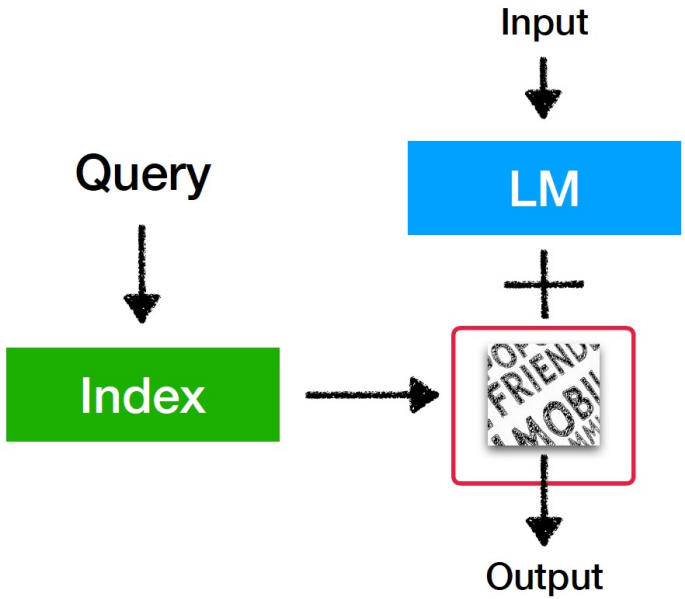
Lack of attributions

Costs of adaptations

Copyright / privacy

Large parameter size

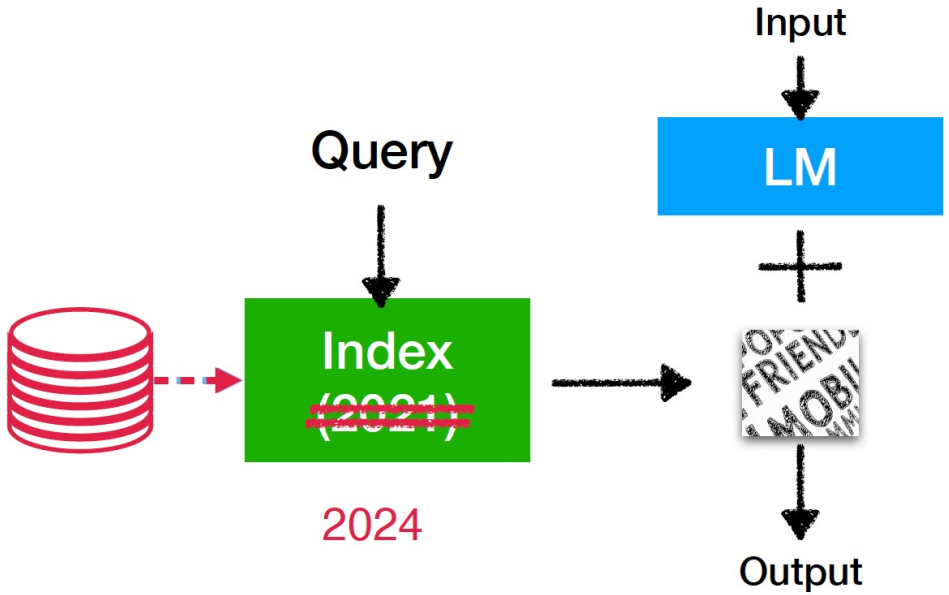
Retrieved text can be used as attributions



RAG for Adaptable LLMs 1/2

- Hallucinations
- Lack of attributions
- **Costs of adaptations**
- Copyright / privacy
- Large parameter size

Replacing datastore (Index) for adaptations without training



Izacard*, Lewis* et al., Atlas: Few-shot Learning with Retrieval Augmented Language Models. JMLR 2023.

RAG for Adaptable LLMs 2/2

Hallucinations

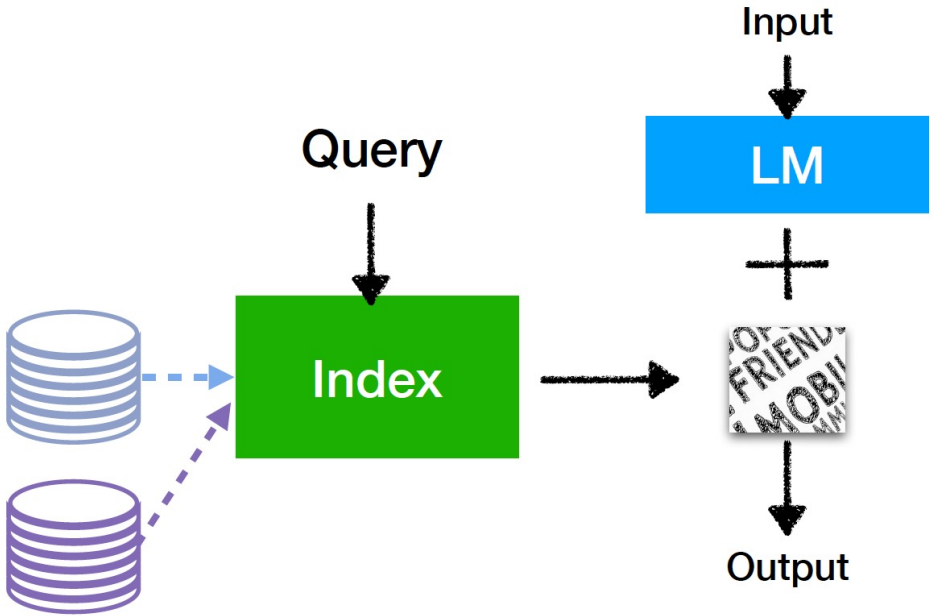
Lack of attributions

Costs of adaptations

Copyright / privacy

Large parameter size

Adding new domain corpora for domain adaptations



RAG for better Copyright/ Privacy compliance

Hallucinations

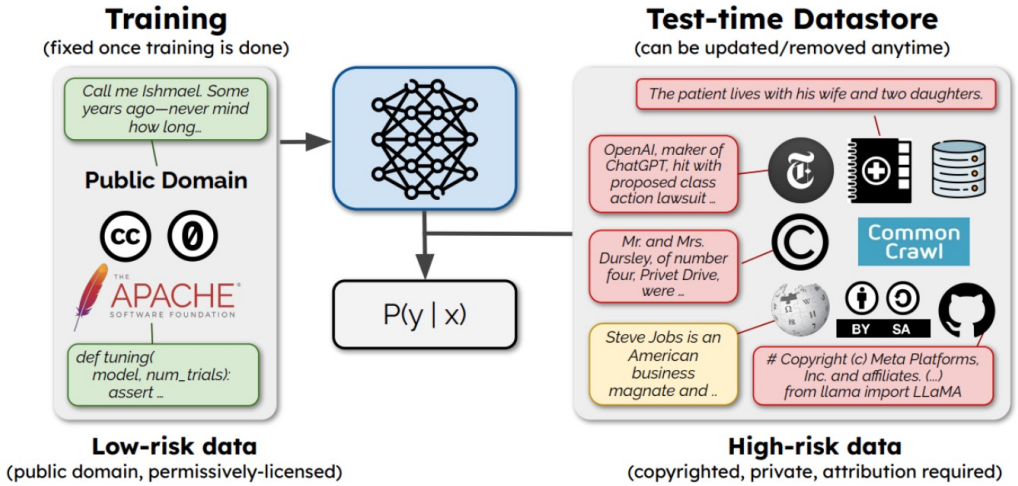
Lack of attributions

Costs of adaptations

Copyright / privacy

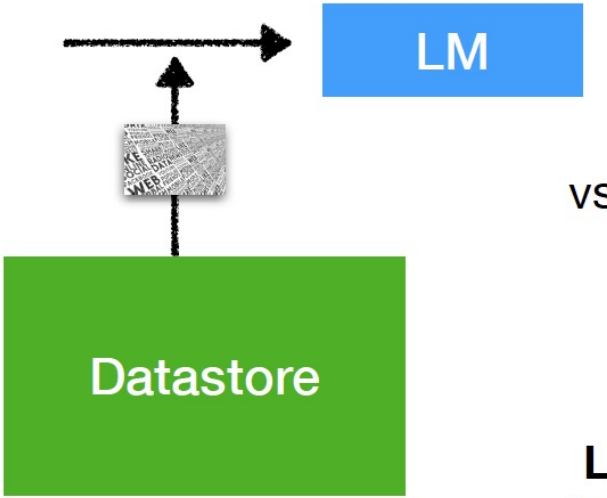
Large parameter size

Segregating copyright-sensitive data from pre-training data

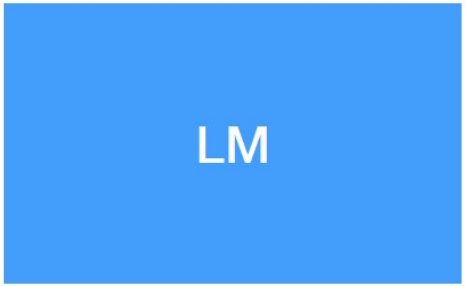


Min* and Gururangan* et al., SILO Language Models: Isolating Legal Risk In a Nonparametric Datastore. ICLR 2024.

RAG helps to scale LMs with (Test-time) Data-store



vs.





Long-term goal: can we possibly reduce the **training** and **inference costs,** and scale down the size of LLMs?

e.g., RETRO (Borgeaud et al., 2021): "obtains comparable performance to GPT-3 on the Pile, despite using **25x fewer parameters**"

RAG helps to keep high-performance LLMs smaller

Hallucinations

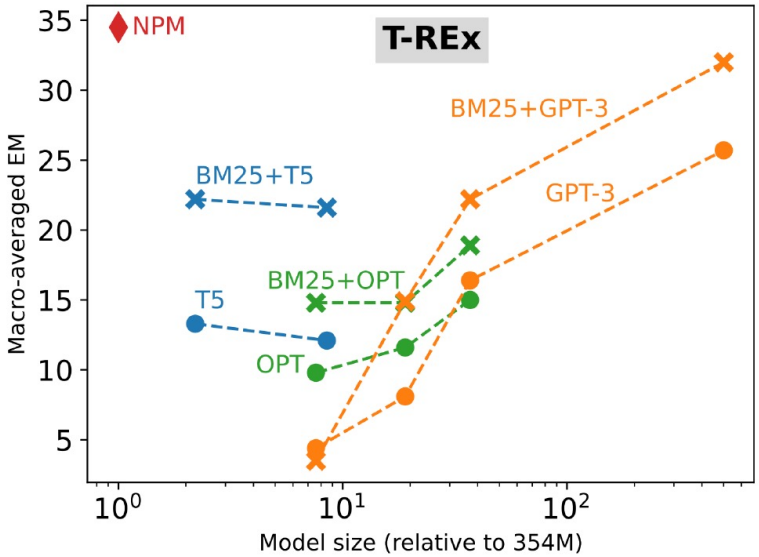
Lack of attributions

Costs of adaptations

Copyright / privacy

Large parameter size

Models with much less parameters can outperforms much larger models!



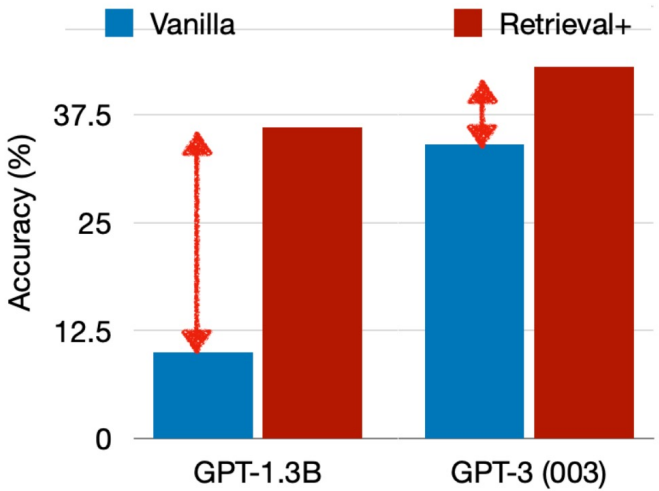
Min et al., Nonparametric Masked Language Modeling. Findings of ACL 2023.

Some early success stories of (basic / naive) RAG

RAG has shown effective in many benchmarks

Application to commercial systems

QA



Mallen*, Asai* et al., When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories (Best Video; Oral) 2023.

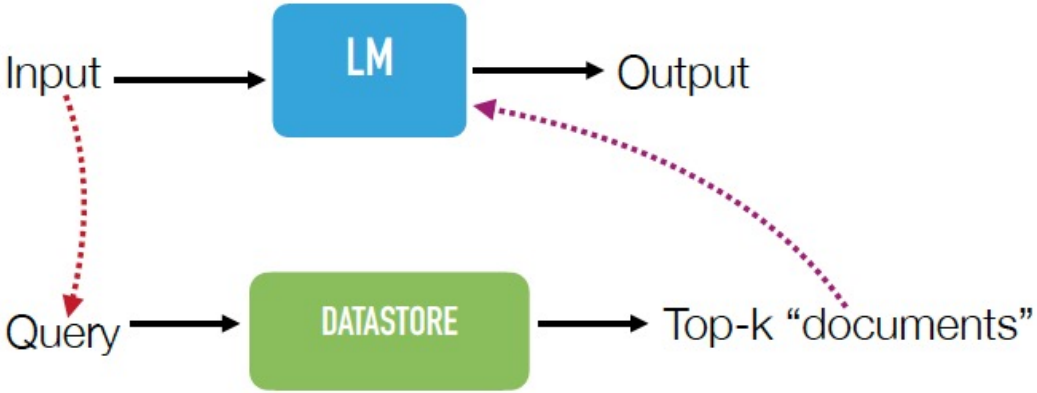
Perplexity
Search & discovery with AI



RAG vs. Fine-Tuning

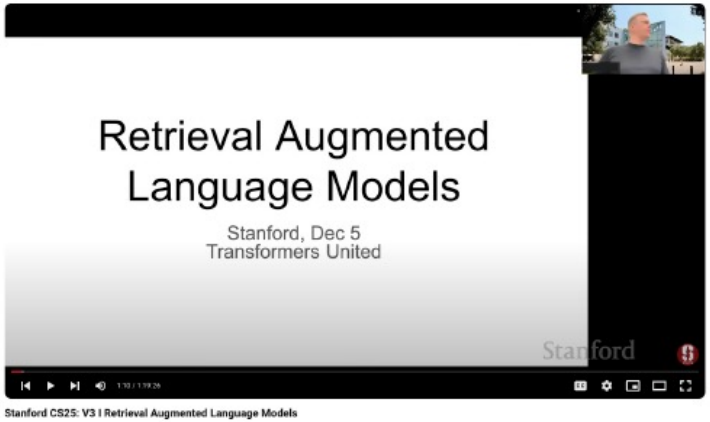
Feature Comparison	RAG	Fine-Tuning
Knowledge Updates	Directly updating the retrieval knowledge base ensures that the information remains current without the need for frequent retraining, making it well-suited for dynamic data environments.	Stores static data, requiring retraining for knowledge and data updates.
External Knowledge	Proficient in leveraging external resources, particularly suitable for accessing documents or other structured/unstructured databases.	Can be utilized to align the externally acquired knowledge from pretraining with large language models, but may be less practical for frequently changing data sources.
Data Processing	Involves minimal data processing and handling.	Depends on the creation of high-quality datasets, and limited datasets may not result in significant performance improvements.
Model Customization	Focuses on information retrieval and integrating external knowledge but may not fully customize model behavior or writing style.	Allows adjustments of LLM behavior, writing style, or specific domain knowledge based on specific tones or terms.
Interpretability	Responses can be traced back to specific data sources, providing higher interpretability and traceability.	Similar to a black box, it is not always clear why the model reacts a certain way, resulting in relatively lower interpretability.
Computational Resources	Depends on computational resources to support retrieval strategies and technologies related to databases. Additionally, it requires the maintenance of external data source integration and updates.	The preparation and curation of high-quality training datasets, defining fine-tuning objectives, and providing corresponding computational resources are necessary.
Latency Requirements	Involves data retrieval, which may lead to higher latency.	LLM after fine-tuning can respond without retrieval, resulting in lower latency.
Reducing Hallucinations	Inherently less prone to hallucinations as each answer is grounded in retrieved evidence.	Can help reduce hallucinations by training the model based on specific domain data but may still exhibit hallucinations when faced with unfamiliar input.
Ethical and Privacy Issues	Ethical and privacy concerns arise from the storage and retrieval of text from external databases.	Ethical and privacy concerns may arise due to sensitive content in the training data.

How to build Retrieval-Augmented LMs (RALMs)



- **Datastore**:
  - What should be **stored** in the datastore?
  - How to **index** the datastore (e.g., granularity)?
  - How to **search** top-k documents efficiently?
- How to **integrated** the retrieved outputs with LMs?
- You can also search the datastore in multiple rounds, and you can search the datastore using "output"!

The work which coined Retrieval Augmented Generation (RAG)



Stanford CS25: V3 I Retrieval Augmented Language Models

Douwe Kiela (Stanford CS25; 2023/12)

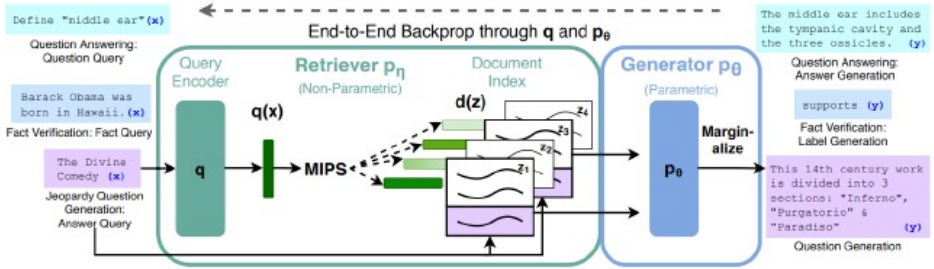
Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis[†‡], Ethan Perez[⋆],

Aleksandra Piktus[†], Fabio Petroni[†], Vladimir Karpukhin[†], Naman Goyal[†], Heinrich Küttler[†],

Mike Lewis[†], Wen-tau Yih[†], Tim Rocktäschel[†‡], Sebastian Riedel[†‡], Douwe Kiela[†]

(Lewis et al., NeurIPS'20)



- An **encoder-decoder** model (BART)
- **Retriever**: DPR (for question answering)
- Fine-tuning on **individual** tasks (question answering, question generation, fact verification)

Key Design Choices for RAG

What to retrieve ?

- Token
- Phrase
- Chunk
- Paragraph
- Entity
- Knowledge graph

When to retrive ?

- Single search
- Each token
- Every N tokens
- Adaptive search

How to use the retrieved information ?

- Input/Data Layer
- Model/Intermediate Layer
- Output/Prediction Layer

Other Issues

Augmentation stage:
- Pre-training
- Fine-tuning
- Inference

Retrieval choice:
- BERT
- Roberta
- BGE
- .....

Model Collaboration



Scale selection

Generation choice:
- GPT
- Llama
- T5
- .....

Key Design Choices of a RAG system: 1. What to Retrieve

coarse

Retrieval granularity

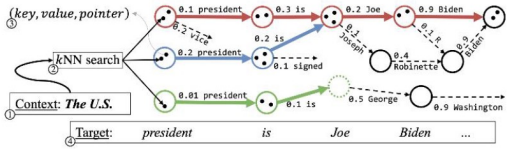
meticulous

low

Level of the structure / organization of the retrieved item(s)

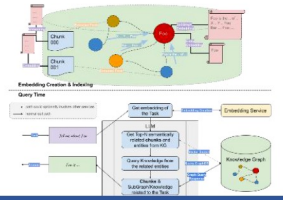
High

Chunk | In-Context RAG 2023



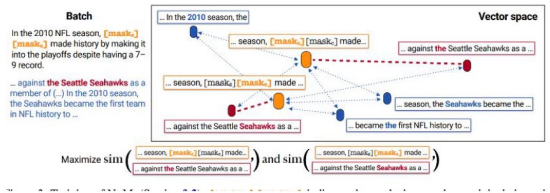
The search is broad, recalling a large amount of information, but with low accuracy, high coverage but includes much redundant information.

Knowledge Graph | 2023

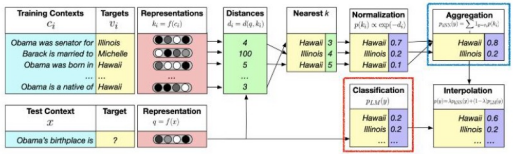


Richer semantic and structured information, but the retrieval efficiency is lower and is limited by the quality of KG.

Phrase | NPM 2023

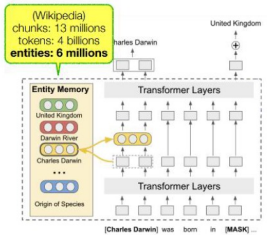


Token | KNN-LMM 2019



It excels in handling long-tail and cross-domain issues with high computational efficiency, but it requires significant storage.

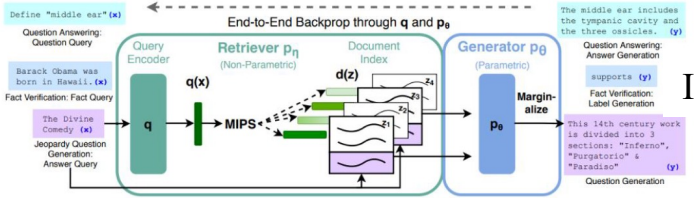
Entity | EasE 2022



2. How to use the retrieved information

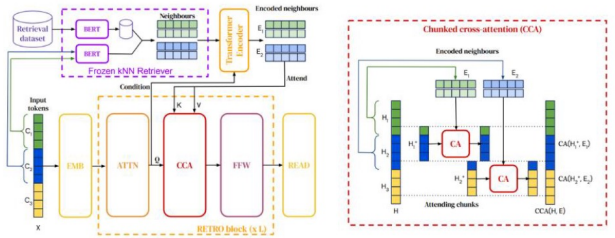
Integrating the retrieved information into different layers of the generation model,during inference process.

Integrate retrieval positions.



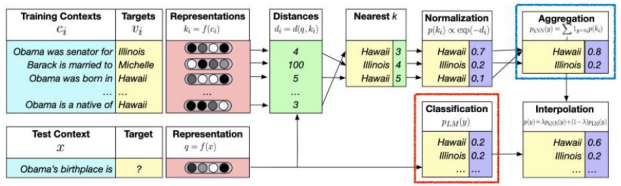
Input / Data layer

Using simple, but unable to support the retrieval of more knowledge blocks, and the optimization space is limited.



Model / Interlayer

Supports the retrieval of more knowledge blocks, but introduces additional complexity and must be trained.



Output /Prediction layer

Ensuring the output results are highly relevant to the retrieval content, but the efficiency is low.

3. When to retrieve

High efficiency, but low relevance of the retrieved documents

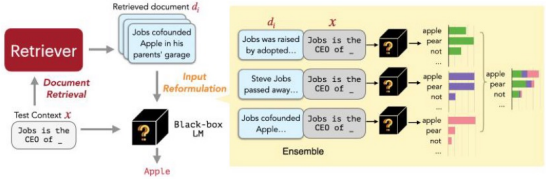
Balancing efficiency and information might not yield the optimal solution

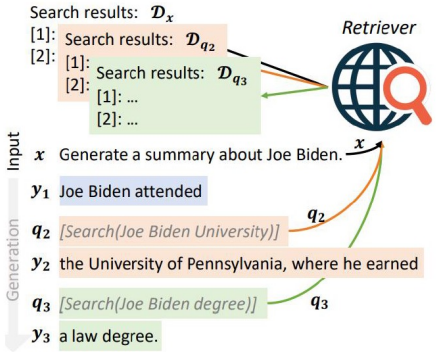
A large amount of information with low efficiency and redundant information.

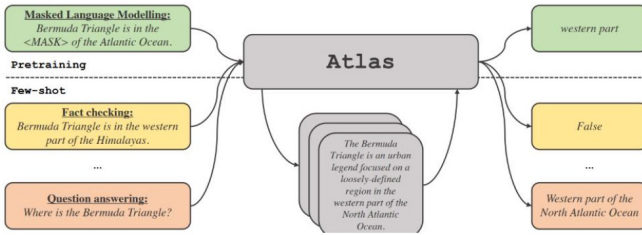
Once | Replug 2023

Adaptive | Flare 2023

Every N Tokens | Atlas 2023







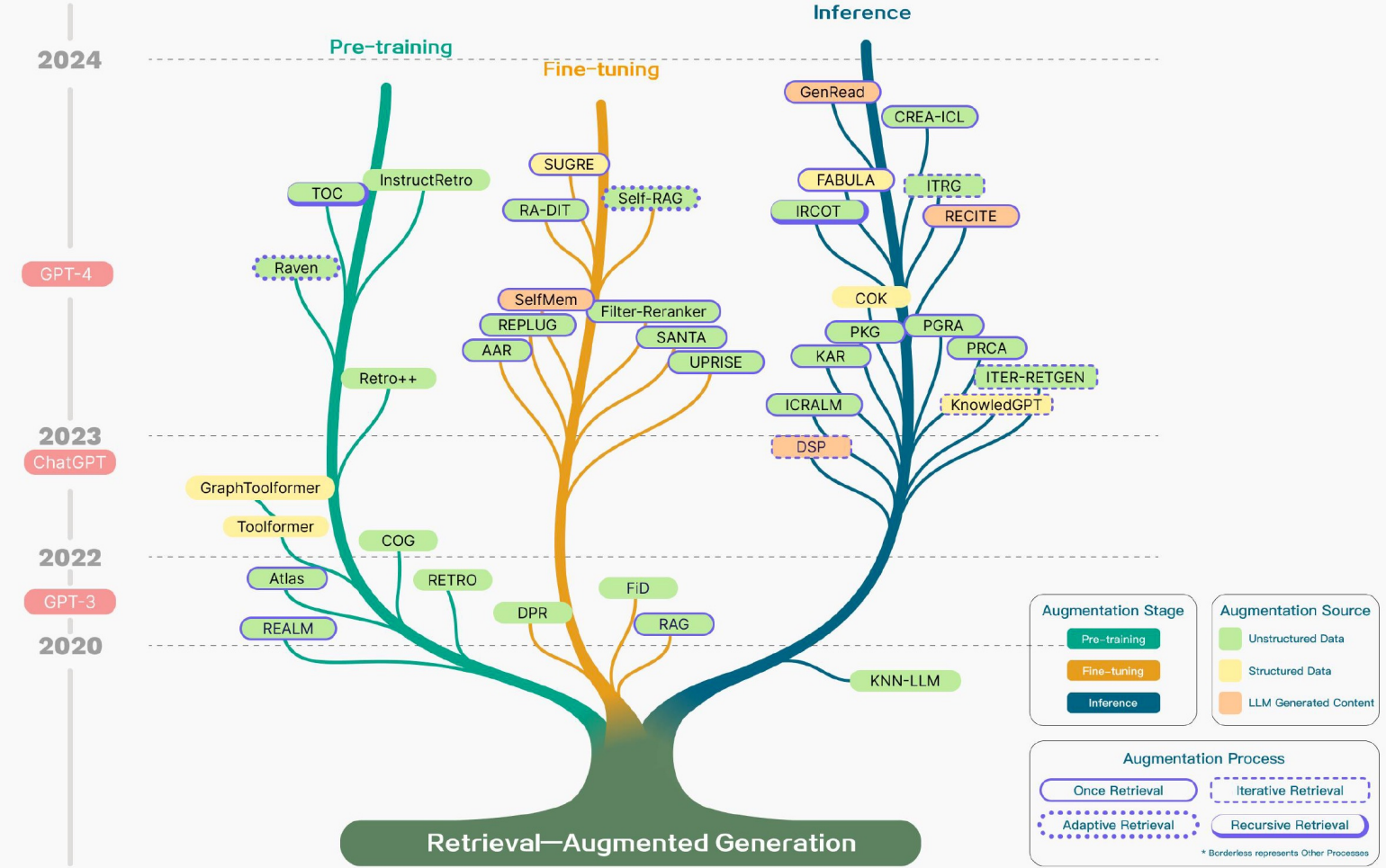
Conducting only one search during the reasoning process.

Adaptively conduct the search.

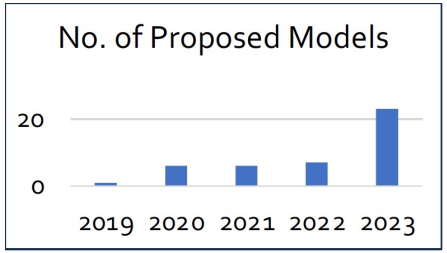
Retrieve once for every N tokens generated.

Low — **Retrieval frequency** — **High**

Evolution of RAG Designs



RAG & RALM Model Development



Citation >1000 | [500, 1000) | [200, 500) | [100, 200) | [50, 100) | [20, 50) | <20

RAG Framework/Pipeline

FiD, IRCoT, REPLUG, AAR, SKR, REFEED, Self-RAG, ToC, kNN-LM, REALM, RAG, SE-FiD, RETRO, OpenBook, DSP, In-Context RALM, ITER-RETGEN, FLARA, RADA, COMBO, SlimPLM

RAG Learning

REALM, RAG, EMDR2, RETRO, Atlas, RAG-end2end, RETRO++, ITER-RETGEN, Self-RAG, PRCA

Retriever Learning

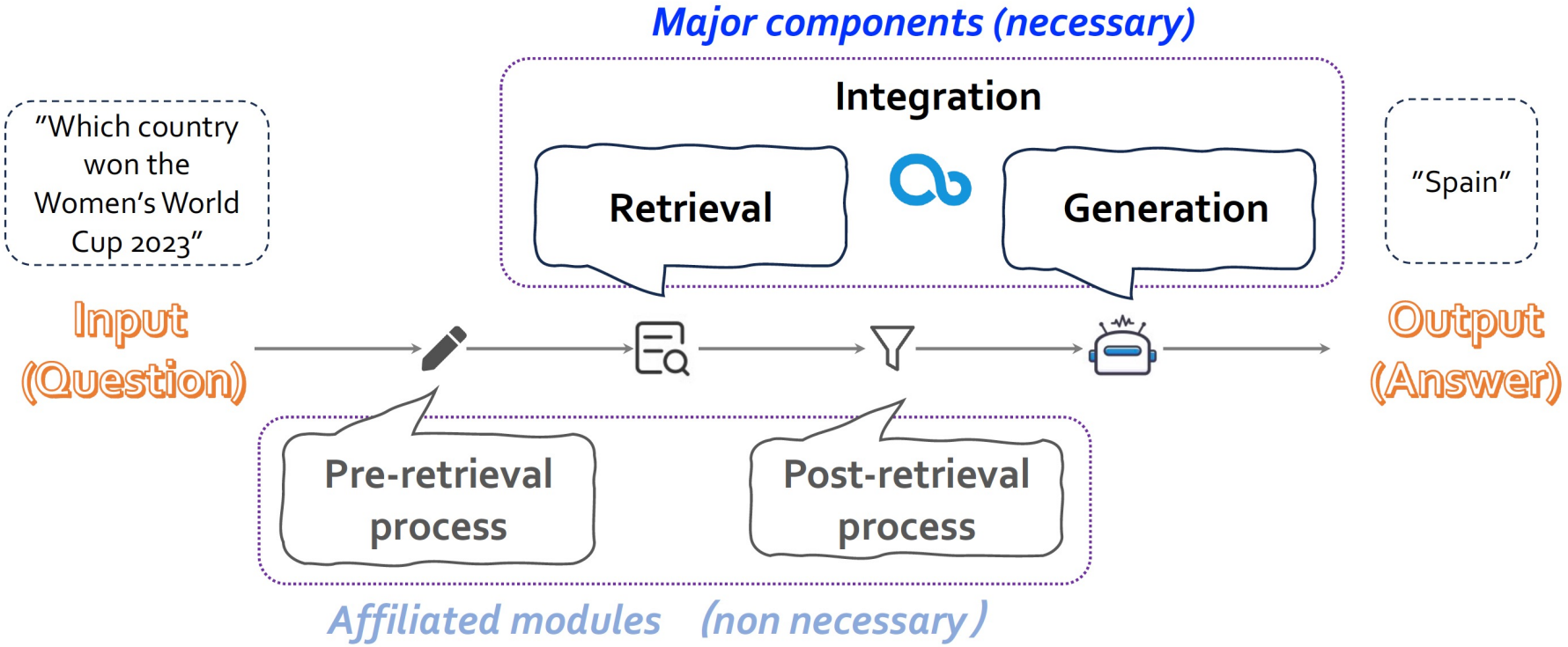
DPR, FiD-KD, Contriever, EPR, FiD-Light, CEIL, UPRISE, UDR, Dr.ICL, SAIL, RADA, LLM-R, REVEN, RA-DIT

Pre-/Post-Retrieval Technique

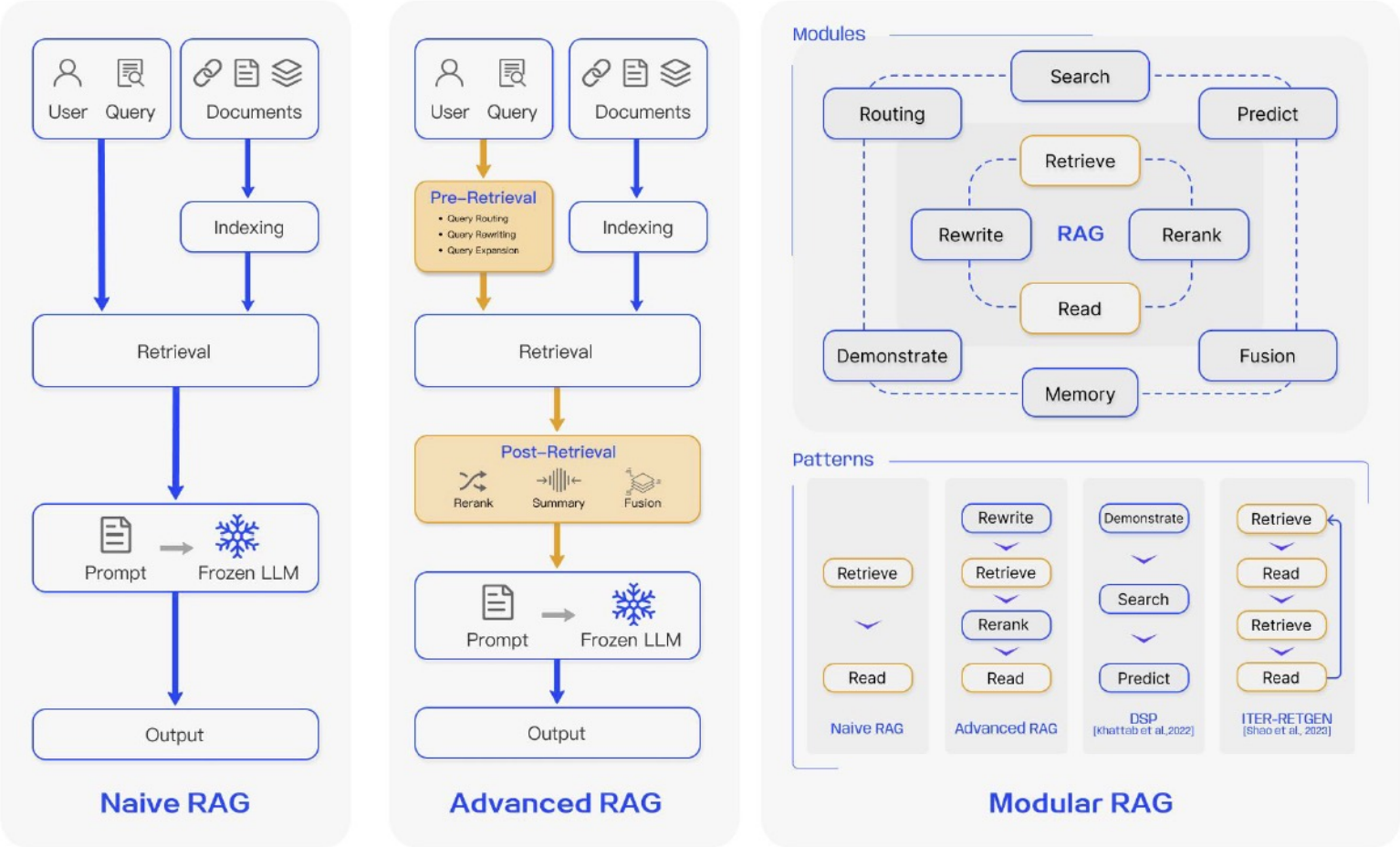
SPALM, Re2G, HyPE, R-BM25, Query2doc, SAIL, QueryRewriter, RECOMP, PRCA, SlimPLM, BlendFilter

2019 2020 2021 2022 2023 2024

RALM Architecture: The Standard Pipeline



Different RAG approaches



Retriever Design

Various Types of Retrieval Methods

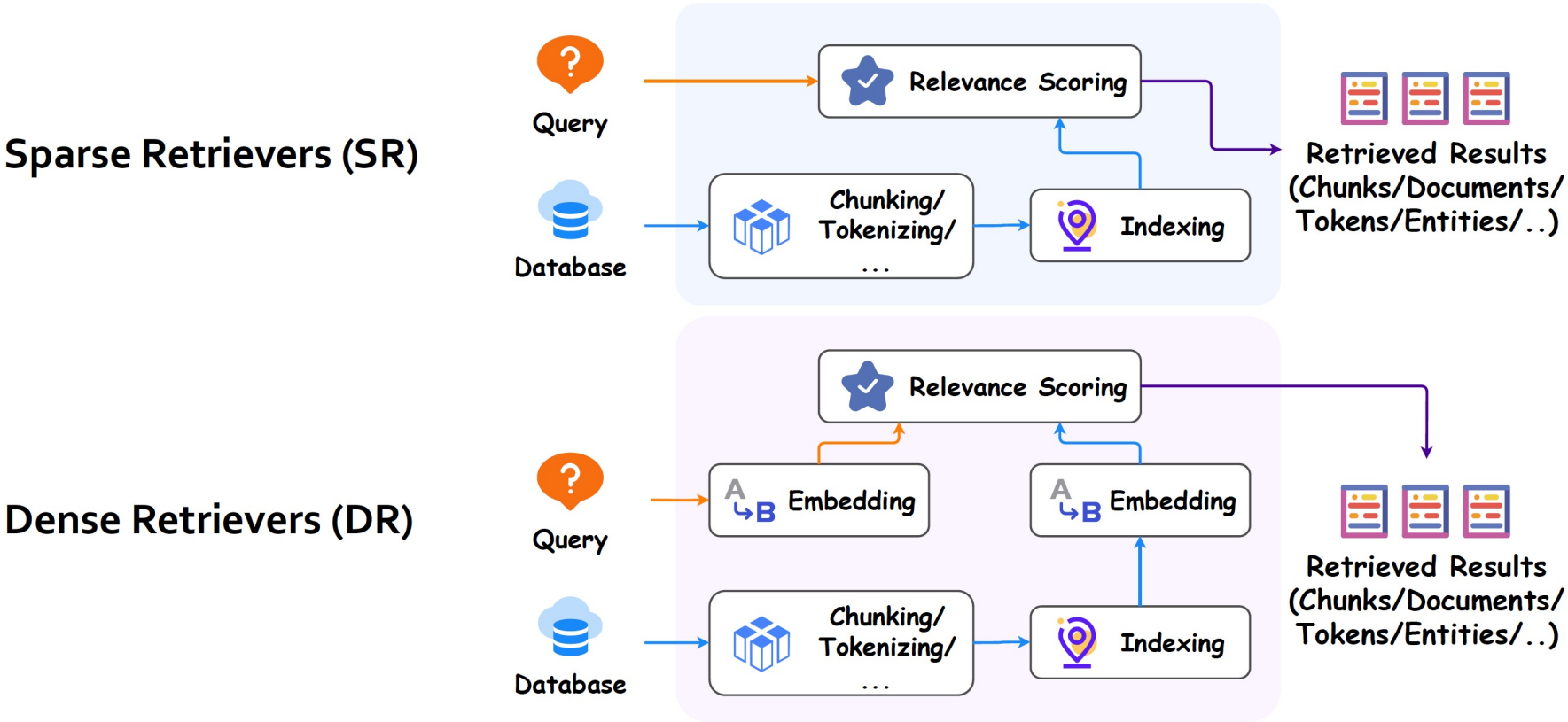
- Sparse Retrieval
- Document-level Dense Retrieval
- Token-level Dense Retrieval
- Cross-encoder Reranking
- Black-box Retrieval (just call Google / Bing Search)
- Hypothetical Document Embeddings

❖ For a detailed coverage, refer to CMU 11-711 ANLP

https://phontron.com/class/anlp2024/lectures/#retrieval-and-rag-feb-15

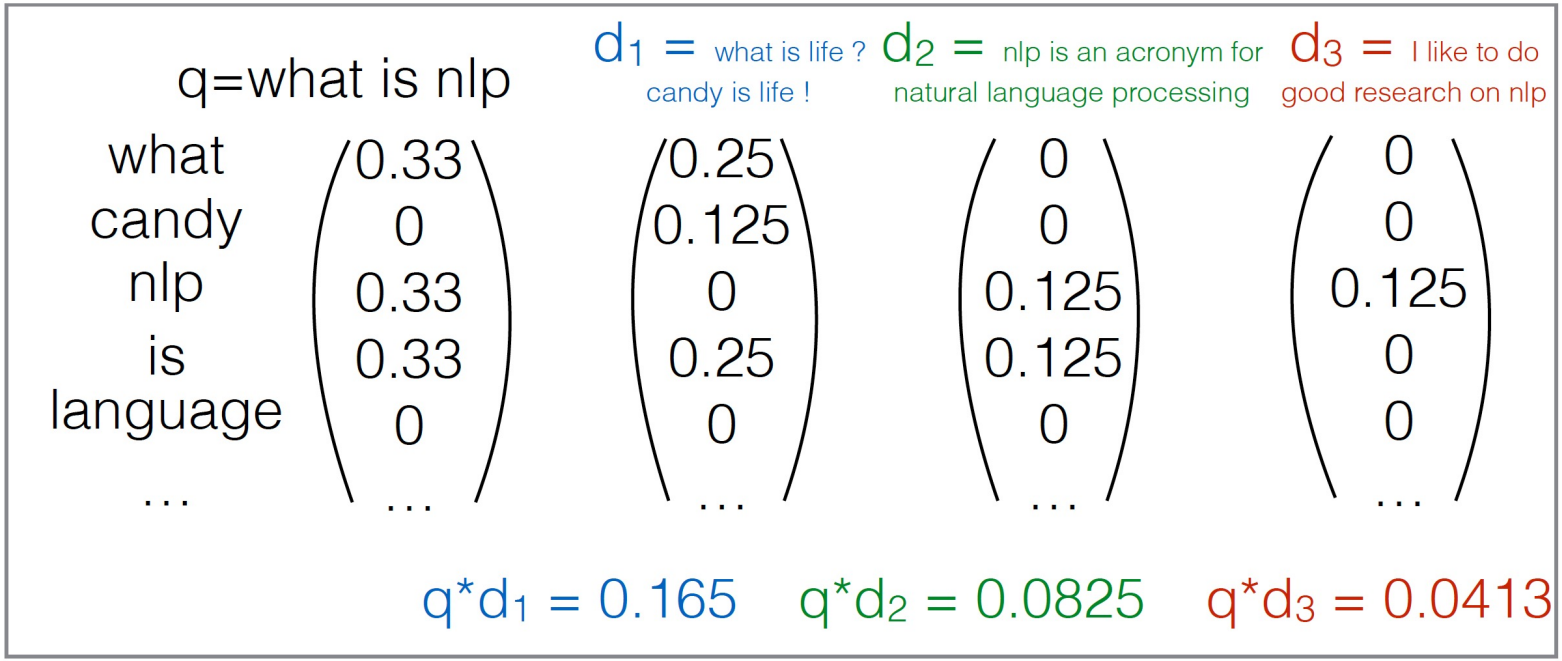
https://www.youtube.com/watch?v=WQYi-1mvGDM&t=2521s

Sparse vs. Dense Retrievers



Sparse Retrieval

- Express the query and document as a sparse word frequency vector (usually normalized by length)



- Find the document with the highest inner-product or cosine similarity in the document collection

Term Weighting for Sparse Retrieval

[see Manning et al. 2009]

- Some terms are more important than others; low-frequency words are often more important
- Term frequency - in-document frequency (TF-IDF)

$$\text{TF}(t,d) = \frac{\text{freq}(t,d)}{\sum_{t'} \text{freq}(t',d)} \qquad \text{IDF}(t) = \log\left(\frac{|D|}{\sum_{d' \in D} \delta(\text{freq}(t,d') > 0)}\right)$$

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \times \text{IDF}(t)$$

- BM25: TF term similar to smoothed count-based LMS

$$\text{BM-25}(t,d) = \text{IDF}(t) \cdot \frac{\text{freq}(t,d) \cdot (k_1 + 1)}{\text{freq}(t,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)}$$

How is retrieval implemented ?

- Sim: a similarity score between two pieces of text

Example $\text{sim}(i,j) = \text{tf}_{i,j} \times \log \frac{N}{\text{df}_i}$

$\text{tf}_{i,j}$: # of occurrences of i in j

N: # of total docs

df_i: # of docs containing i

Sparse retrieval

Example $\text{sim}(i,j) = \text{Encoder}(i) \cdot \text{Encoder}(j)$

Maps the text into an h-dimensional vector

Dense retrieval

- An entire field of study on how to define or learn these similarity functions better
- There are efficient data structures/infrastructure for supporting fast and accurate search from a large datastore

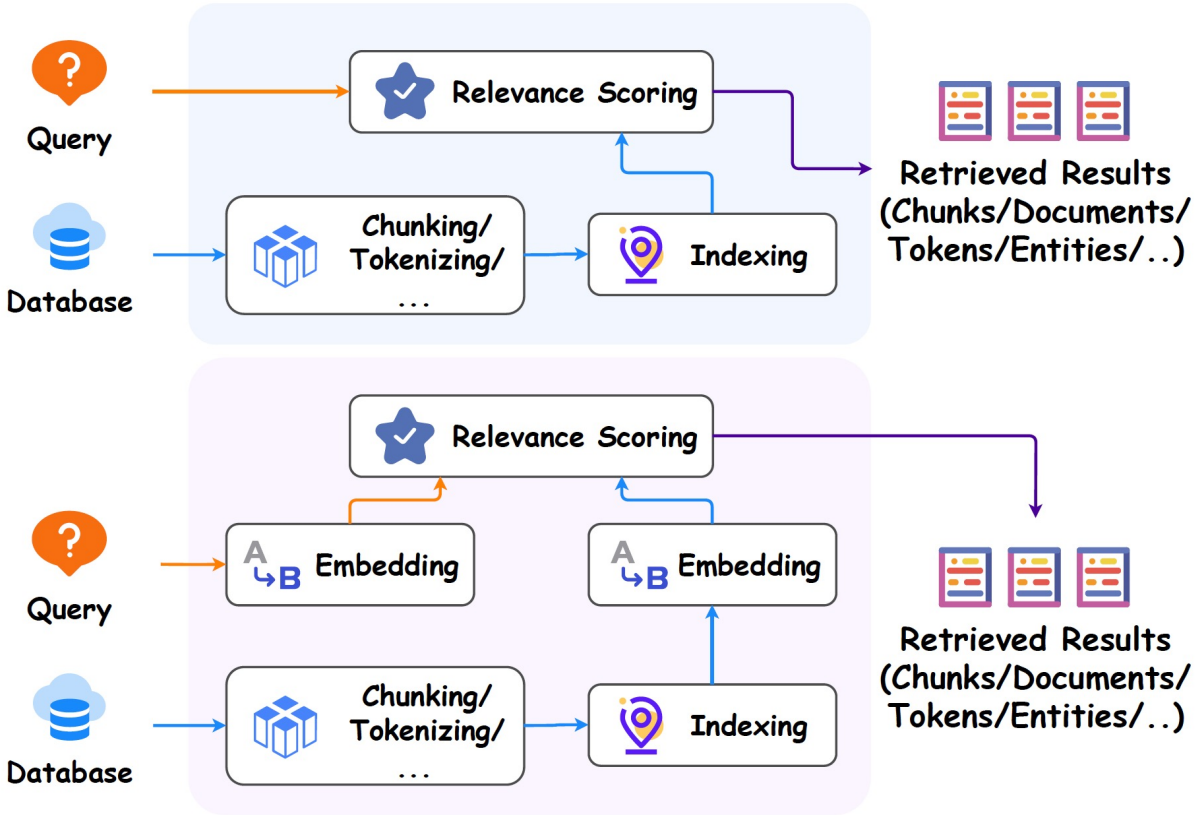
Sparse vs. Dense Retrievers

Sparse Retrievers (SR)

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

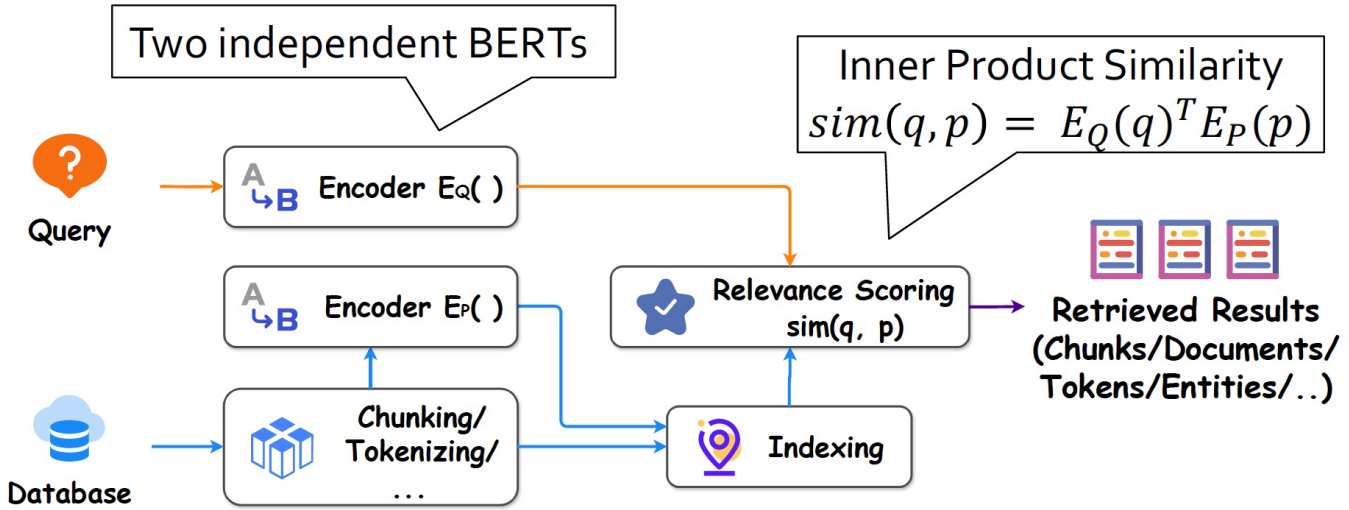
Dense Retrievers (DR)

- 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)



Karpukhin et al. 2020. “Dense Passage Retrieval for Open-Domain Question Answering”

Task-Specific Pre-trained Retriever (Supervised)

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

- Learning Objective

$$L(q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^-)$$

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

- Training data: Question-Passage Sets

$$\mathcal{D} = \{\langle q_i, p_i^+, p_{i,1}^-, \cdots, p_{i,n}^- \rangle\}_{i=1}^{m}$$

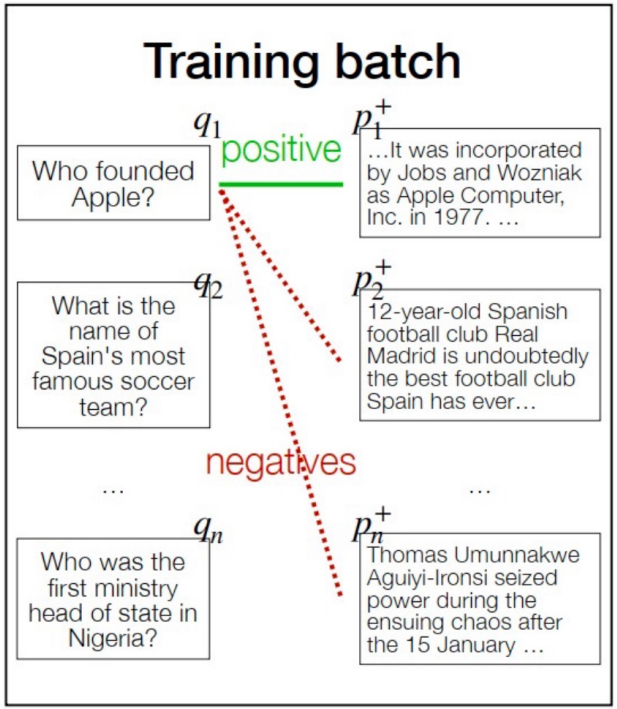
Question

Relevant passage

Irrelevant passages

Negative sample selection?

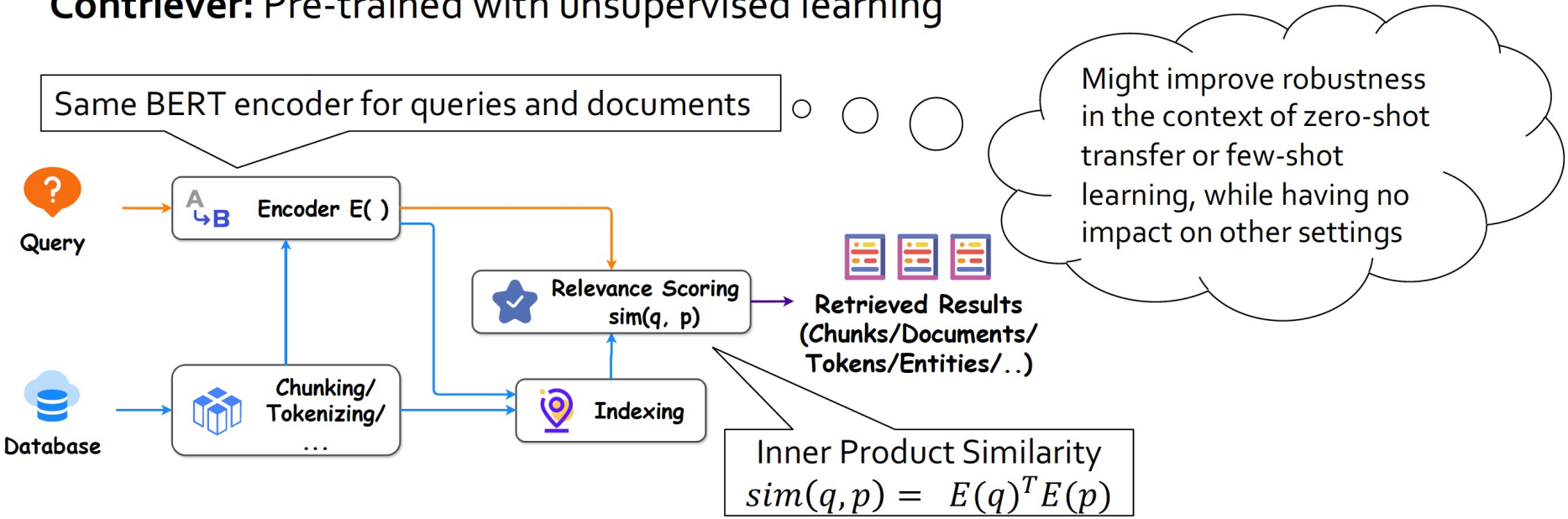
- Training with in-batch negatives



Karpukhin et al. 2020. “Dense Passage Retrieval for Open-Domain Question Answering”

General-Purpose Pre-trained Retriever (Unsupervised)

Contriever: Pre-trained with unsupervised learning



Contrastive learning with unaligned documents

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

Izacard et al. 2022. “Unsupervised Dense Information Retrieval with Contrastive Learning”

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	$\text{REALM}_{\text{Wiki}}$ (Guu et al., 2020)	39.2	-	40.2	46.8	-
Single	$\text{REALM}_{\text{News}}$ (Guu et al., 2020)	40.4	-	40.7	42.9	-
Single	BM25	32.6	52.4	29.9	24.9	38.1
Single	DPR	**41.5**	56.8	34.6	25.9	29.8
Single	BM25+DPR	39.0	57.0	35.2	28.0	36.7
Multi	DPR	**41.5**	56.8	**42.4**	49.4	24.1
Multi	BM25+DPR	38.8	**57.9**	41.1	**50.6**	35.8

Both widely applied in RAG and RA-LLMs

DPR in

RAG, FiD, RETRO, EPR, UDR, …

Contriever in

Self-RAG, Atlas, RAVEN, …

Both better than the sparse retriever!

	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

N.B.: More direct performance comparison of DRP vs. Contriever collected by Frontier LLMs: https://poe.com/s/Zbs9LZ3UpHLFPxLuLolJ

How is retrieval implemented ?

Software: FAISS, Distributed FAISS, SCaNN, etc…

Method	Class name	index_factory	Main parameters	Bytes/vector	Exhaustive	Comments
Exact Search for L2	IndexFlatL2	"Flat"	d	4*d	yes	brute-force
Exact Search for Inner Product	IndexFlatIP	"Flat"	d	4*d	yes	also for cosine (normalize vectors beforehand)
Hierarchical Navigable Small World graph exploration	IndexHNSWFlat	"HNSW,Flat"	d, M	4*d + x * M * 2 * 4	no	
Inverted file with exact post-verification	IndexIVFFlat	"IVFx,Flat"	quantizer, d, nlists, metric	4*d + 8	no	Takes another index to assign vectors to inverted lists. The 8 additional bytes are the vector id that needs to be stored.
Locality-Sensitive Hashing (binary flat index)	IndexLSH	-	d, nbits	ceil(nbits/8)	yes	optimized by using random rotation instead of random projections
Scalar quantizer (SQ) in flat mode	IndexScalarQuantizer	"SQ8"	d	d	yes	4 and 6 bits per component are also implemented.
Product quantizer (PQ) in flat mode	IndexPQ	"PQx", "PQ"M"x"nbits	d, M, nbits	ceil(M * nbits / 8)	yes	
IVF and scalar quantizer	IndexIVFScalarQuantizer	"IVFx,SQ4" "IVFx,SQ8"	quantizer, d, nlists, qtype	SQfp16: 2 * d + 8, SQ8: d + 8 or SQ4: d/2 + 8	no	Same as the IndexScalarQuantizer
IVFADC (coarse quantizer+PQ on residuals)	IndexIVFPQ	"IVFx,PQ"y"x"nbits	quantizer, d, nlists, M, nbits	ceil(M * nbits/8)+8	no	
IVFADC+R (same as IVFADC with re-ranking based on codes)	IndexIVFPQR	"IVFx,PQy+z"	quantizer, d, nlists, M, nbits, M_refine, nbits_refine	M+M_refine+8	no	

Exact Search



CPU vs. GPU

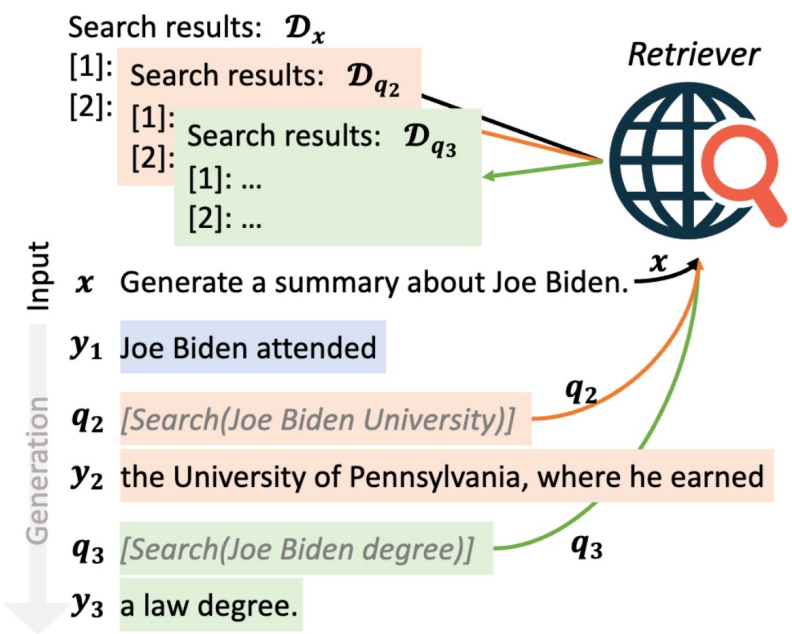
Approximate Search
(Relatively easy to scale to ~1B elements)

Search Engines as Retrievers

Traditional retrieval methods

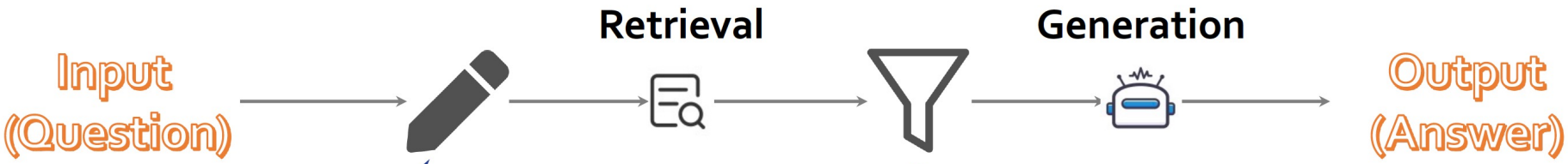
Google Bing

- 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



Komeili et al., 2021, Internet-Augmented Dialogue Generation
Jiang et al. 2023, Active Retrieval Augmented Generation

Pre/Post-Retrieval Techniques



Pre-retrieval process: to improve the adaptation and effectiveness of the query

- Query rewriting
- Query decomposition
- Query expansion

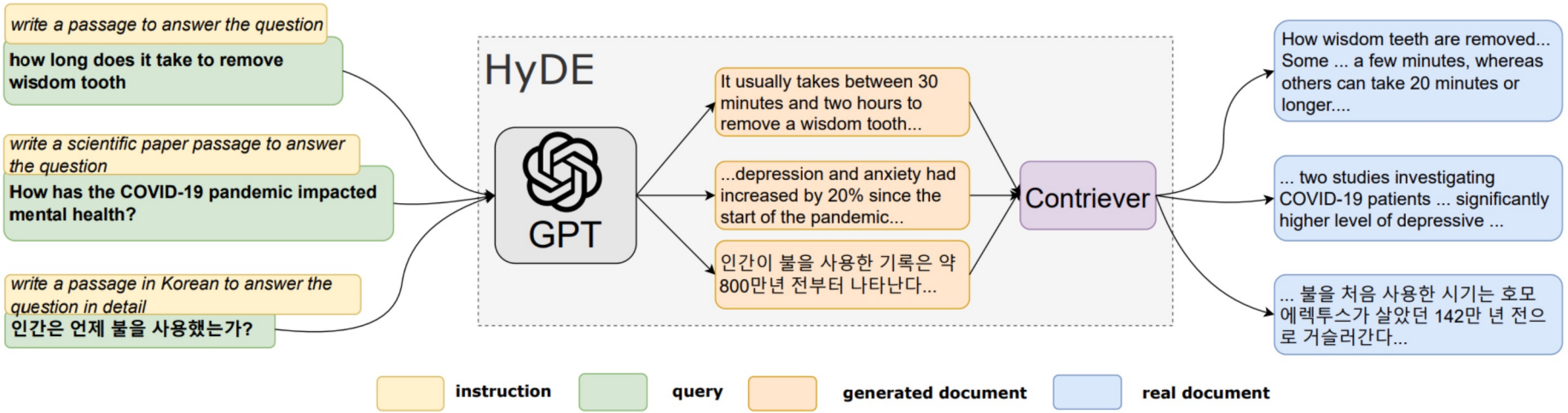
Post-retrieval process: to select better results, merge multiple ones, etc

- Re-ranking
- Compression
- Correction

Hypothetical Document Embeddings – A Pre-Retrieval Technique

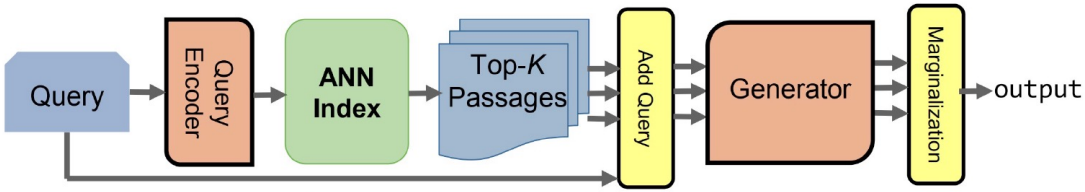
(Gao et al. 2022)

- Generate a “hypothetical document” for the query using an LLM, and try to look it up
- Can be easier than trying to match under-specified query

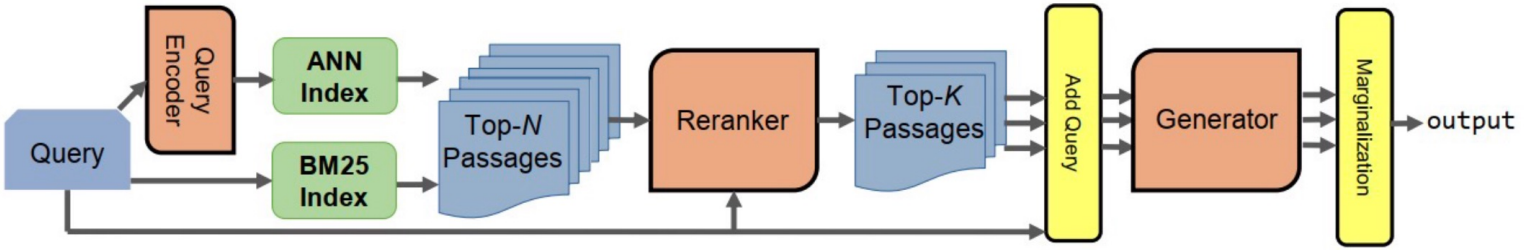


Retrieved Result Rerank (Re2G) – A Post-Retrieval Technique

- 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

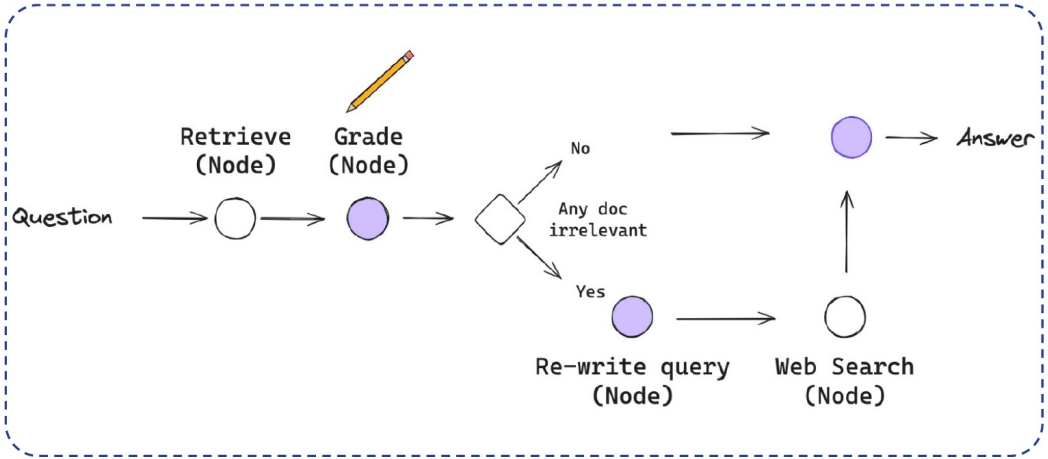


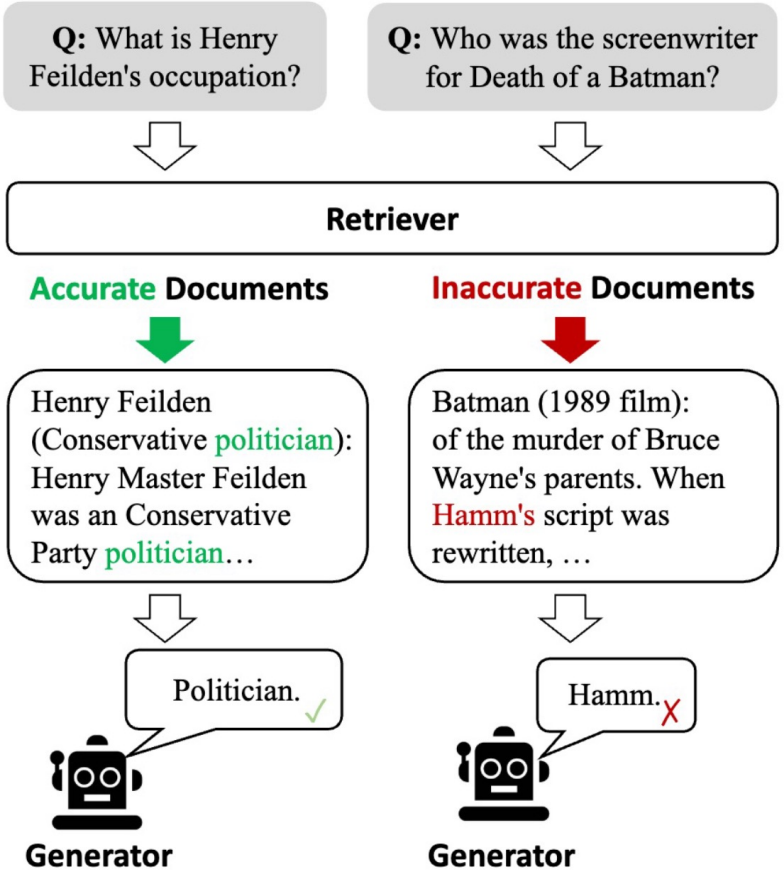
RAG with Reranker

Glass et al. 2022. “Re2G: Retrieve, Rerank, Generate”

Corrective RAG – A Post-Retrieval Technique

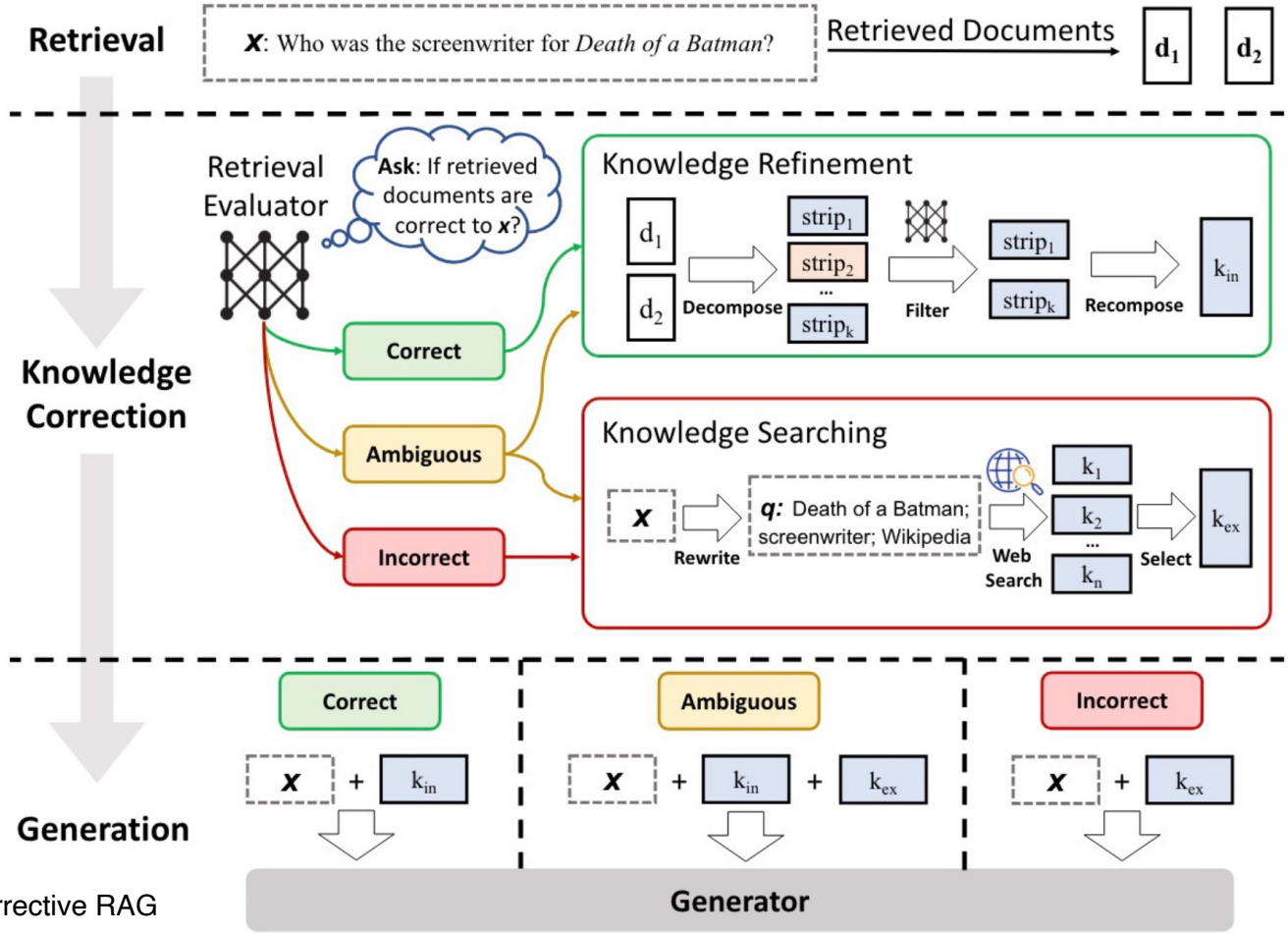
- Grading and Correcting with the help of an LLM





Yan et al., 2024, Corrective Retrieval Augmented Generation

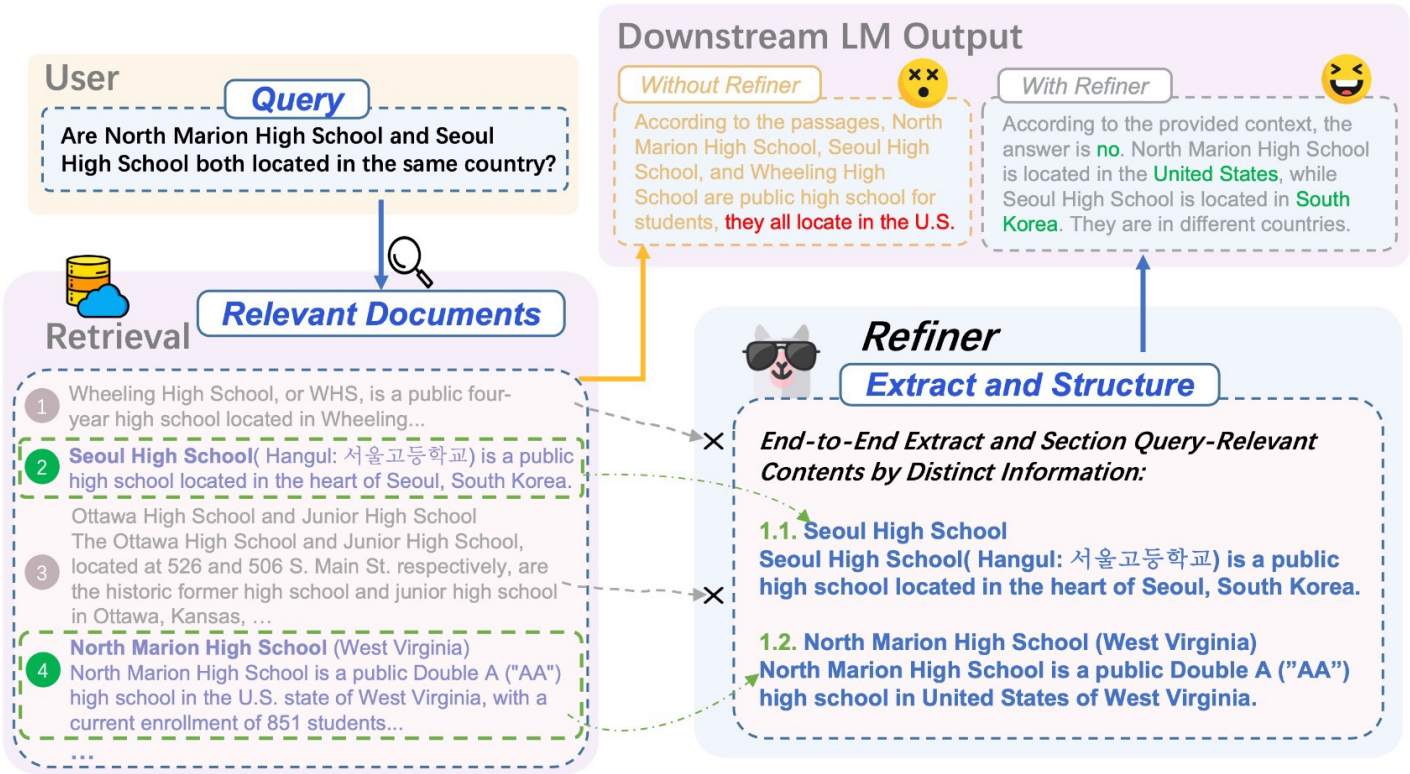
Corrective RAG – A Post-Retrieval Technique



Yan et al., 2024, Corrective RAG

Refiner – A Post-Retrieval Technique

- Leverage a single decoder-only LLM to adaptively extract query relevant contents verbatim along with the necessary context.

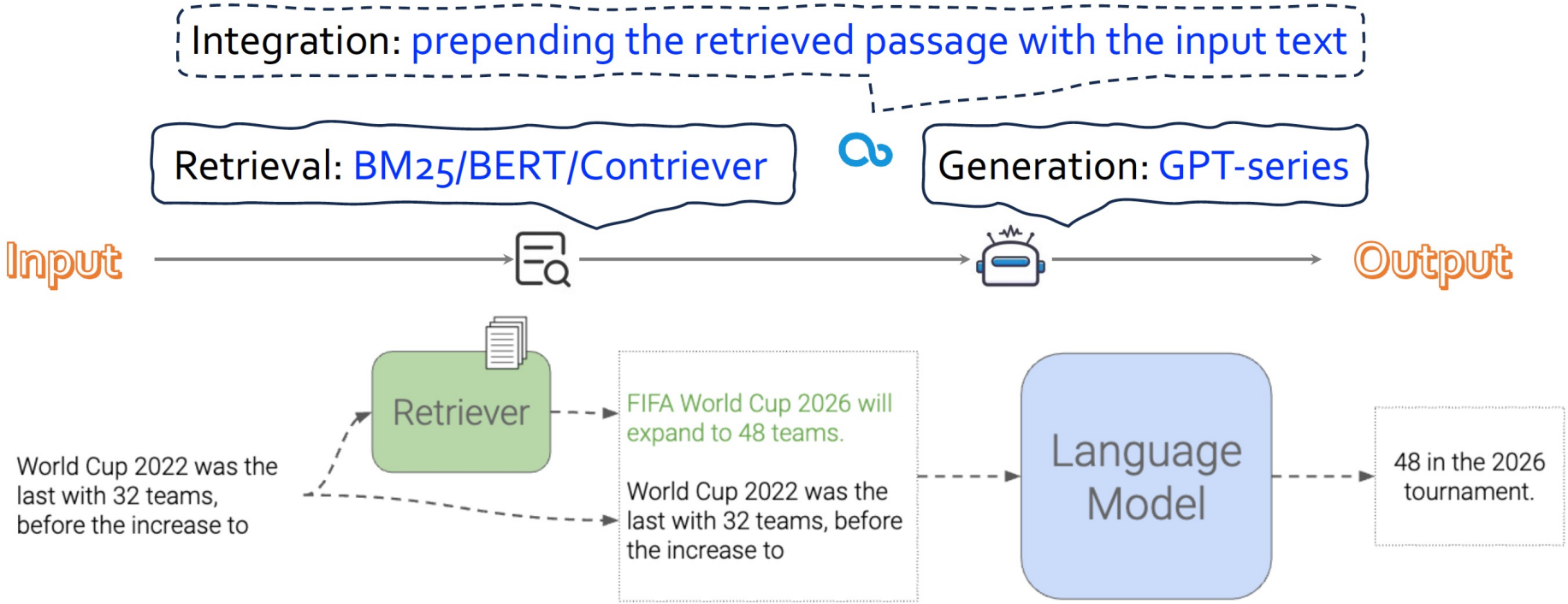


Li et al., 2024, Refiner: Restructure Retrieval Content Efficiently to Advance Question-Answering Capabilities

How & Where to use retrieved info ?

In-Context RALM

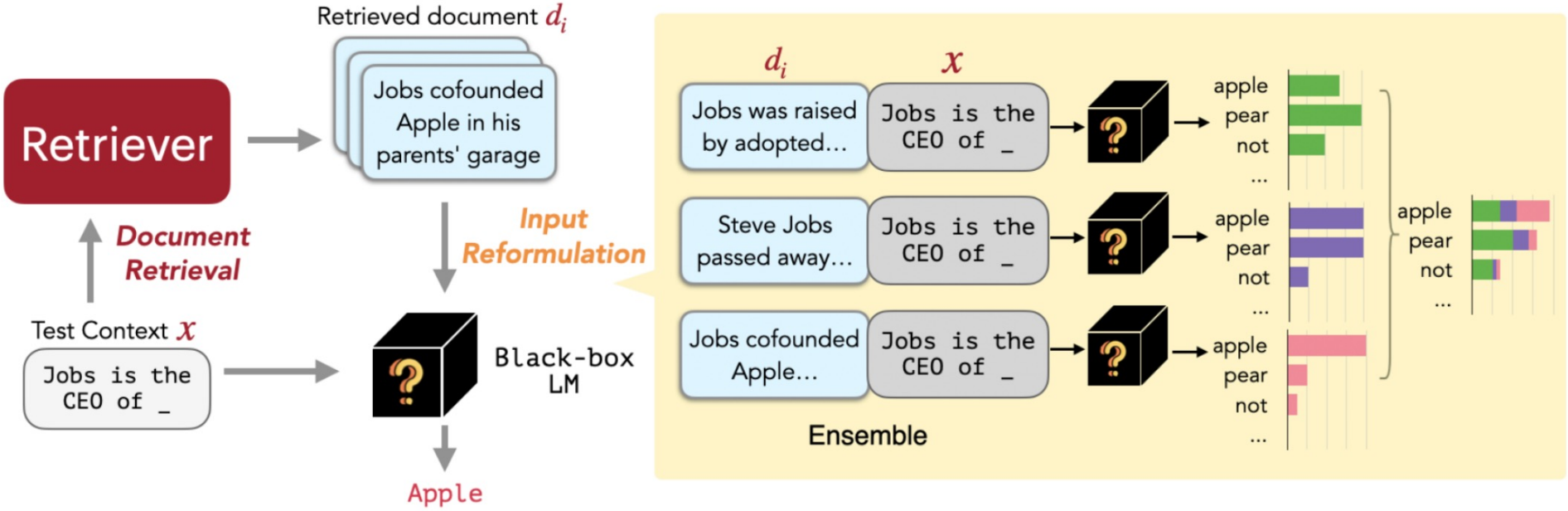
(Retrieved info used at the input layer)



Ram et al. 2023, In-Context Retrieval-Augmented Language Models

REPLUG

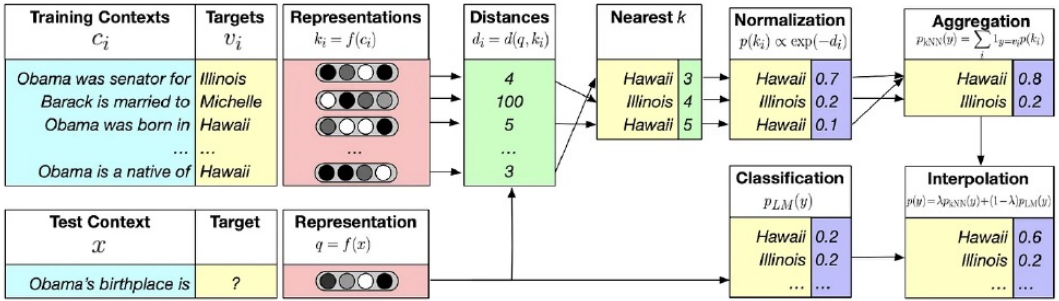
(Retrieved info used at the input layer)



Source: REPLUG (Shi et al., NAACL 2024)

kNN-LM

(Retrieved info used at the output layer)



ARCHITECTURE

- **Token**-level datastore
- Representation = input to last FFN layer
- Integration only at the **output layer**

$$p_{\text{kNN}}(w \mid c_t) \propto \sum_{(c,x)\in\mathcal{D}} \mathbb{I}(x = w) \exp(-\|f_\theta(c) - f_\theta(c_t)\|^2)$$

→ only keep top-K after NN search

$$P(w \mid c_t) = \lambda P_{\text{LM}}(w \mid c_t) + (1 - \lambda) P_{\text{kNN}}(w \mid c_t)$$ (Linear interpolation)

Source: kNN-LM (Khandelwal et al., ICLR 2020)

More Examples of Early RALMs

The paper which coined the term Retrieval Augmented Generation (RAG)

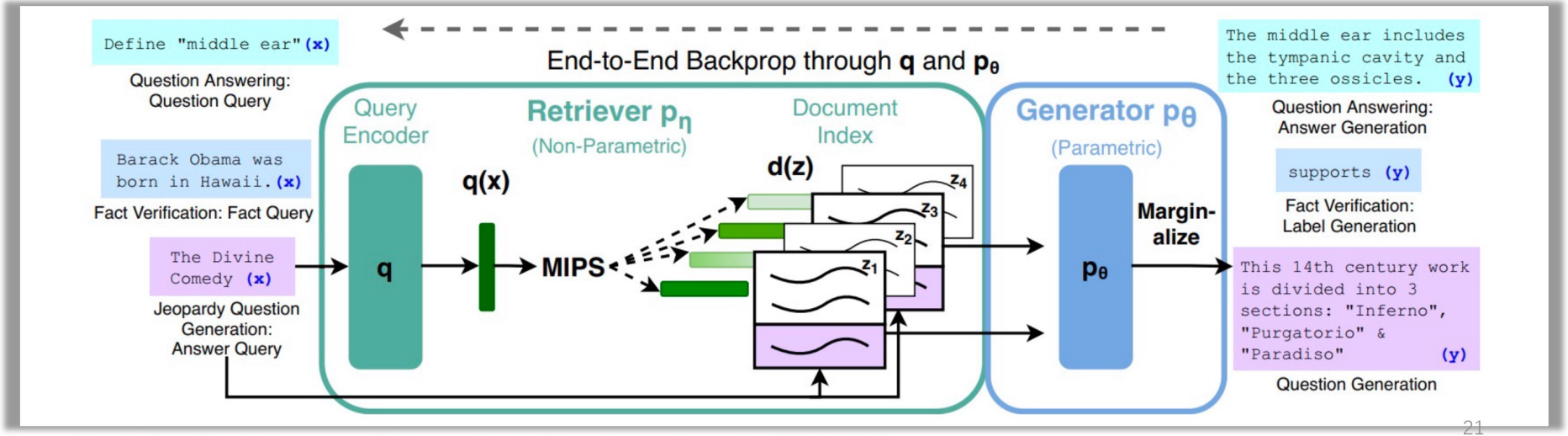
❑ **RAG**

Integration: concatenating each retrieved passage with the question

Retrieval: DPR + MIPS

Generation: BART

Input → Output



Lewis et al. 2020. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks"

Two Diverging Paths for building Retrieval-Augmented LMs in Practice 1/2

Path #1: Build a language model that has a built-in retrieval component

- You need to consider how to build the datastore/index as part of the model
- The optimal model architecture is still an ongoing exploration
- An alternative of scaling today's parametric (Transformer-based) LMs
- Lots of interesting technical challenges, not as successful as we hoped for 🥹

Under the RETRO approach, the LLM (GPT3) is *pretrained* with contextual info retrieved by the retriever, which is a "frozen" (i.e. already-pretrained) BERT.

DeepMind

Improving language models by retrieving from trillions of tokens

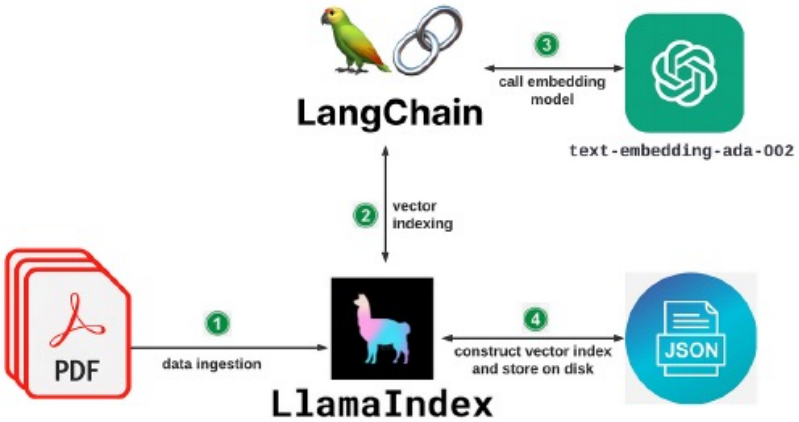
Sebastian Borgeaud[†], Arthur Mensch[†], Jordan Hoffmann[†], Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae[‡], Erich Elsen[‡] and Laurent Sifre[†,‡]

All authors from DeepMind, [†]Equal contributions, [‡]Equal senior authorship

Two Diverging Paths for building Retrieval-Augmented LMs in Practice 2/2

Path #2: view retrieval as one of the "tools" that LMs learn how to use

- Assuming you already have a very powerful LM
- Retrieval can be viewed as an API or "black box" e.g., a search engine
- Research questions: when to call the retriever? how to take retrieved results in context?
- Very popular in developer community ("frozen RAG")



The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

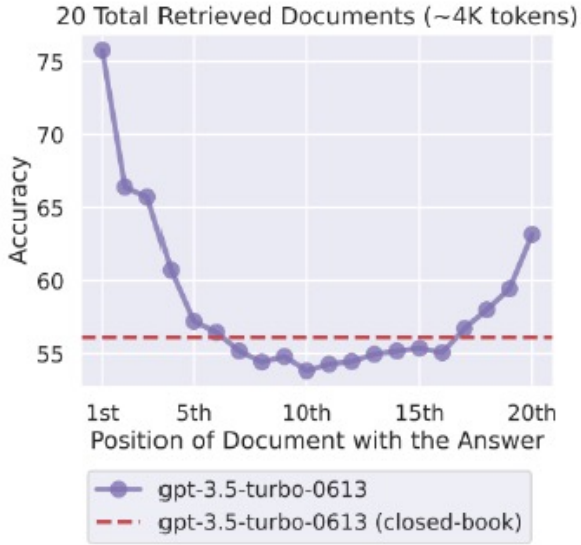
The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

Toolformer (Schick et al., NeurIPS'23)

Long-context LMs vs. Retrieval-Augmented LMs

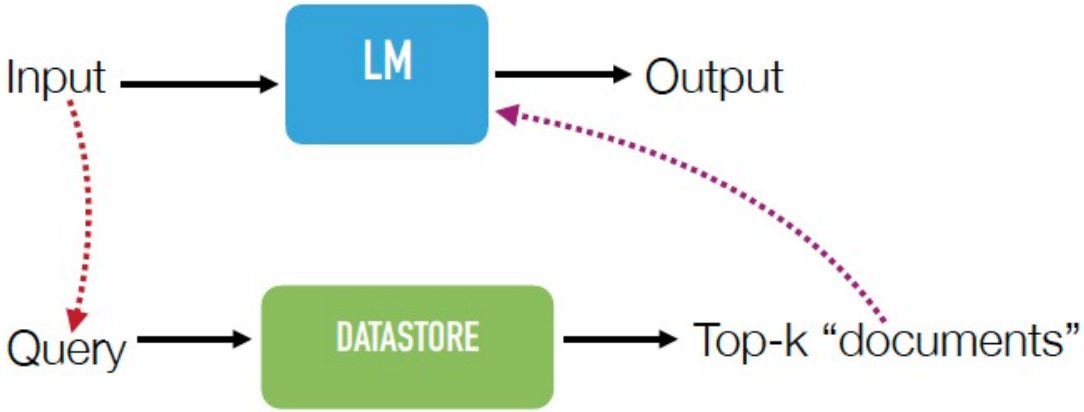
- Today's long-context LLMs support up to millions of tokens in their context window - Do we still need RAG?
- Long-context LMs support better RAG (more documents, more tokens)
- It puts less demand on retriever, but it can't really replace RAG (since the datastore is still much larger)
- Though there are still a lot of questions about whether long-context LMs can really support their contexts



(Liu et al., TACL 2023) "Lost in the Middle"

Example Model Architectures and Training Methods for Retrieval Augmented Language Models (Path #1)

Key Choices for RALMs



ARCHITECTURE

- How is the retriever represented? Granularity of datastore?
- How are retrieved contexts integrated with LMs?
- Sizes of LMs vs datastore?

LEARNING

- How are the LM and retriever trained together?
- Training from scratch? Fine-tuning?
- Pre-training or instruction tuning?

Example: RETRO

DeepMind

Improving language models by retrieving from trillions of tokens

Sebastian Borgeaud[†], Arthur Mensch[†], Jordan Hoffmann[†], Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae[‡], Erich Elsen[‡] and Laurent Sifre[†,‡]

ARCHITECTURE

- How is the retriever represented? Granularity of datastore?
- Sizes of LMs vs datastore?
- How are retrieved contexts integrated with LMs?

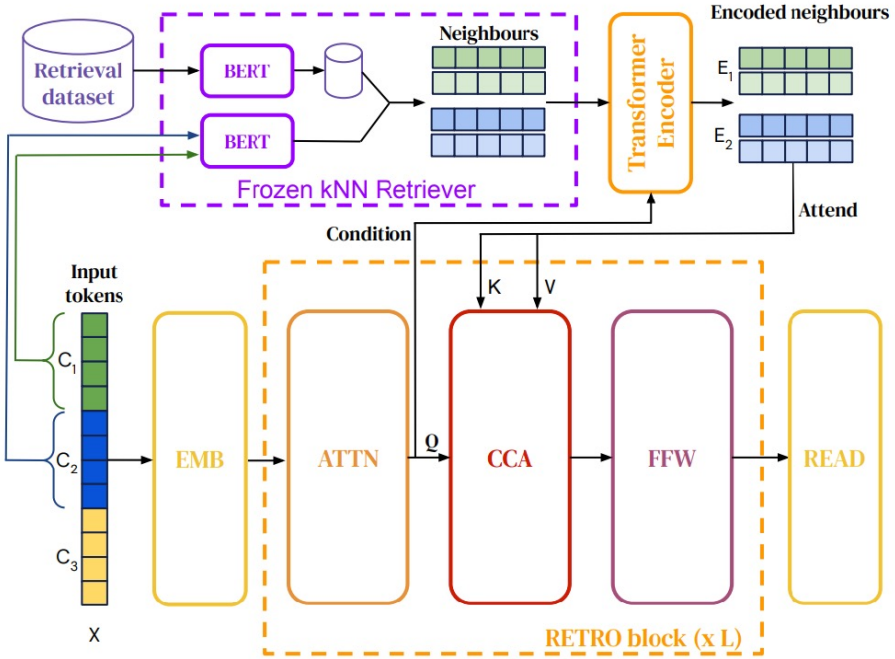
Under the RETRO approach, the LLM (GPT3) is *pretrained* with contextual info retrieved by the retriever, which is a "frozen" (i.e. already-pretrained) BERT.

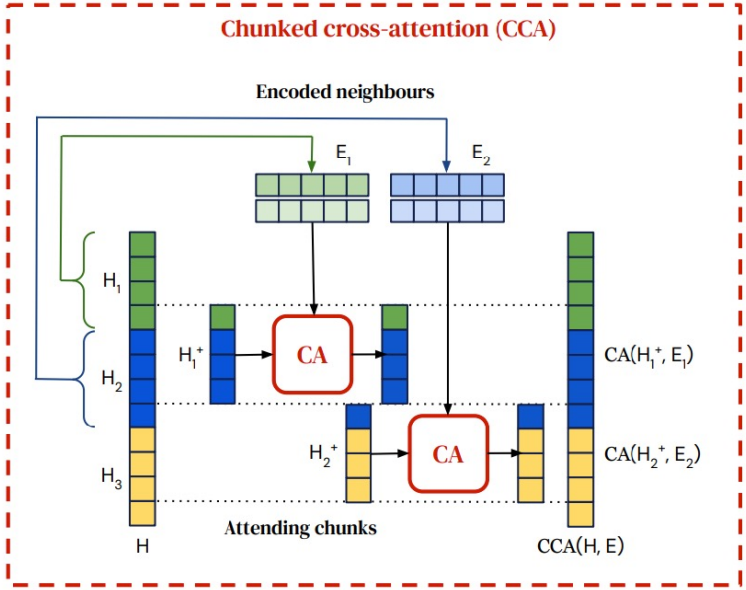
- Granularity: chunks of 64 tokens
- Representation: frozen BERT encoders (pre-trained but for different tasks)
- LMs: 150M-7B parameters; datastore: up to 2T tokens (training tokens 600B tokens)

Example: RETRO

ARCHITECTURE

- How are retrieved contexts integrated with LMs?





The information is integrated into the intermediate layers of Transformers (with an extra encoder)

Comparison b/w RETRO and other RALMs

Model Name	#/ Retrieval Tokens	When to Involve Retrieval	Architecture	Initialization	Re-indexing
RETRO (Borgeaud et al.)	$O(10^{12})$	Pretraining	decoder-only	From Scratch / Pretrained GPT	No
Atlas (Izacard et al.)	$O(10^{9})$	Pretraining	encoder-decoder	Pretrained T5	Yes
REALM (Guu et al.)	$O(10^{9})$	Pretraining	encoder-only	Pretrained BERT	Yes
RAG (Lewis et al.)	$O(10^{9})$	Fine-tuning	encoder-decoder	Pretrained BART	No
DPR (Karpukhin et al.)	$O(10^{9})$	Fine-tuning	encoder-only	Pretrained BERT	No
FiD (Izacard and Grave)	$O(10^{9})$	Fine-tuning	encoder-decoder	Pretrained T5	No
KNN-LM (Khandelwal et al.)	$O(10^{9})$	Inference	decoder-only	Pretrained GPT	No

Table 1: Comparison of different retrieval-augmented models in terms of #/ retrieval tokens, which stage to incorporate retrieval into LMs, the architecture of the backbone LM, whether it requires initialization from the existing LM checkpoint, and whether it requires expensive re-indexing. RETRO is the most scalable retrieval-augmented LM due to its chunk-level retrieval and scalable decoder-only autoregressive LM backbone (Thoppilan et al., 2022; Brown et al., 2020; Smith et al., 2022; Chowdhery et al., 2022) without expensive retrieval index refresh.

Source: RETRO++: Wang et al., “Shall We Pretrain Autoregressive Language Models with Retrieval? A Comprehensive Study,” ENMNLP 2023.

Example: RETRO

LEARNING

- How are the LM and retriever trained together?
- Training from scratch? Fine-tuning?
- Pre-training or instruction tuning?

- The retriever ("encoder") is pre-trained and not updated anymore
- Trained from scratch or fine-tuned ("retrofit")

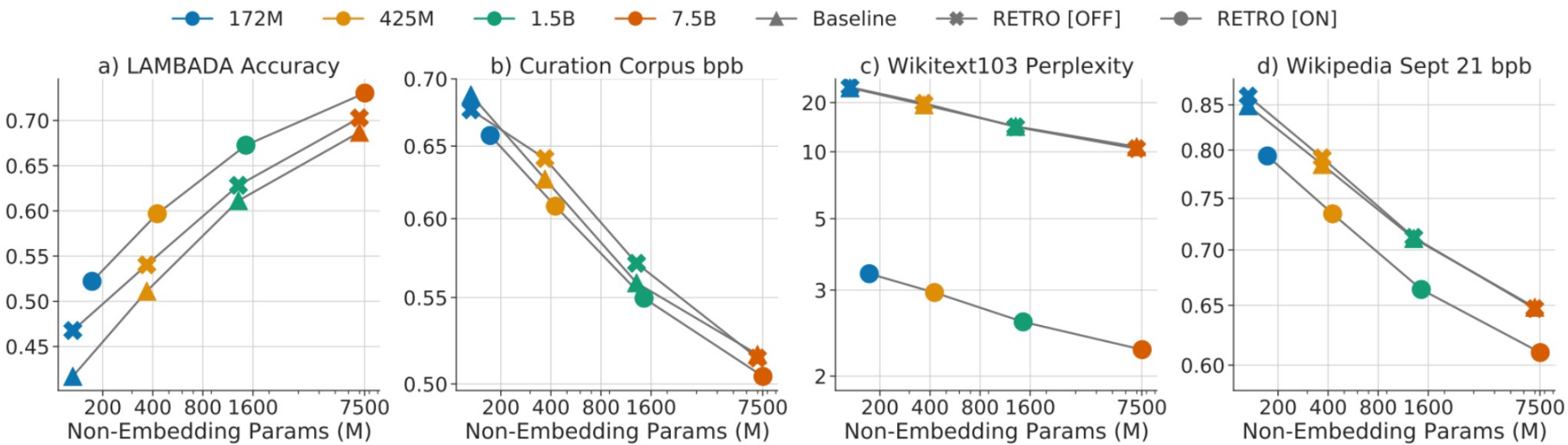
[Submitted on 11 Oct 2023 (v1), last revised 29 May 2024 (this version, v3)]

InstructRetro: Instruction Tuning post Retrieval-Augmented Pretraining

Boxin Wang, Wei Ping, Lawrence McAfee, Peng Xu, Bo Li, Mohammad Shoeybi, Bryan Catanzaro

RETRO: Experiments

Perplexity



In restrospect, too many works focused on improving perplexity instead of downstream tasks at that time..

RETRO: Experiments

Question answering

Model	Test Accuracy
REALM (Guu et al., 2020)	40.4
DPR (Karpukhin et al., 2020)	41.5
RAG (Lewis et al., 2020)	44.5
EMDR^2 (Sachan et al., 2021)	52.5
FID (Izacard and Grave, 2021)	51.4
FID + Distill. (Izacard et al., 2020)	**54.7**
Baseline 7B (closed book)	30.4
RETRO 7.5B (DPR retrieval)	45.5

It is not better than specialized QA models

No evals on any of in-context learnings tasks in GPT-3

"With a 2 trillion token database, our Retrieval-Enhanced Transformer (Retro) obtains ***comparable performance to GPT-3 and Jurassic-1 on the Pile, despite using 25× fewer parameters****."*

RETRO is not open-sourced… :(:(

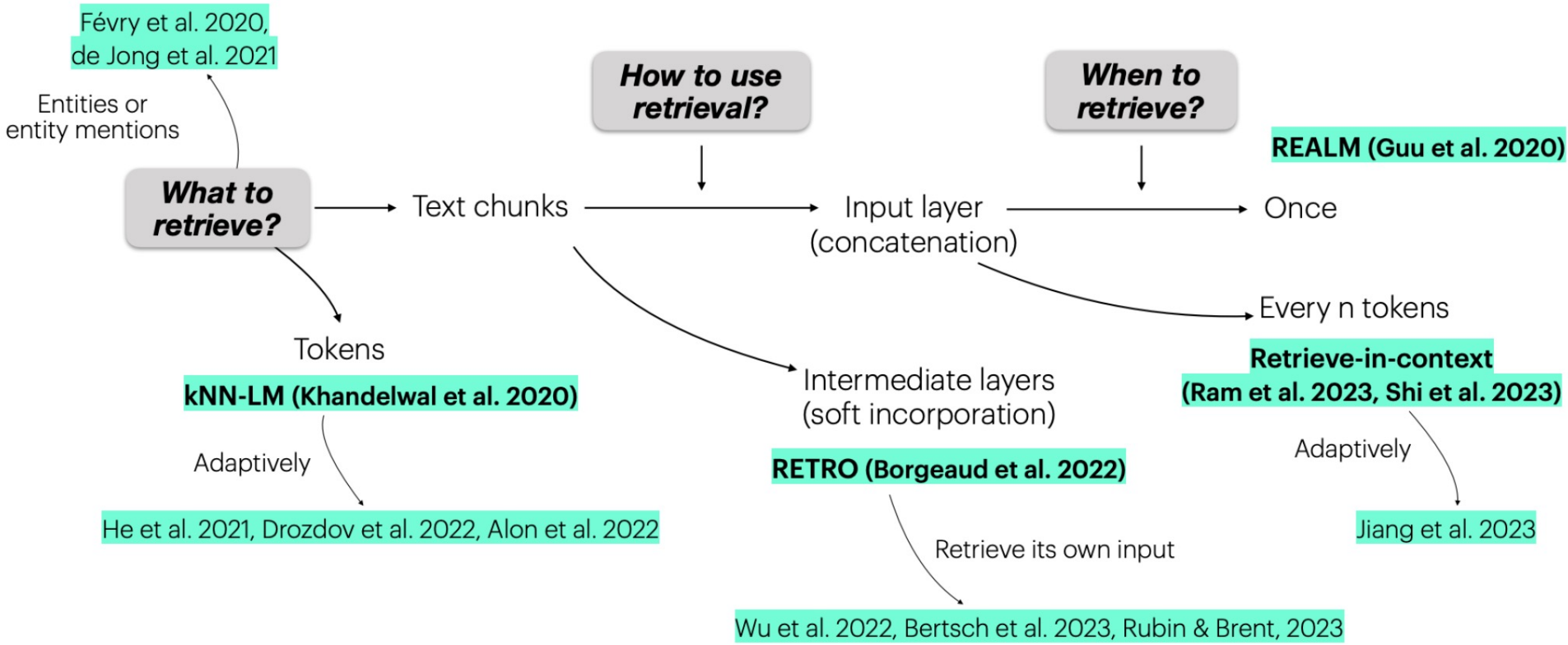
RETRO++ (Nvidia)

Tasks	Small		Medium		XL		XXL	
	GPT	RETRO	GPT	RETRO	GPT	RETRO	GPT	RETRO
Knowledge-intensive Tasks								
HellaSwag	31.3	36.2 ↑4.9	43.2	46.2 ↑3.0	56.7	59.0 ↑2.3	72.3	70.6 ↓1.7
BoolQ	59.3	61.8 ↑2.5	57.4	57.2 ↓0.2	62.2	62.7 ↑0.5	67.3	70.7 ↑3.4
Knowledge-nonintensive Tasks								
Lambada	41.7	41.4 ↓0.3	54.1	55.0 ↑0.9	63.9	64.0 ↑0.1	73.9	72.7 ↓1.2
RACE	34.6	32.5 ↓2.1	37.3	37.3 ↑0.0	40.8	39.9 ↓0.9	44.3	43.2 ↓1.1
PiQA	64.3	64.8 ↑0.5	70.2	68.7 ↓1.5	73.7	74.1 ↑0.4	78.5	77.4 ↓1.1
WinoGrande	52.4	52.0 ↓0.4	53.8	55.2 ↑1.4	59.0	60.1 ↑1.1	68.5	65.8 ↓2.7
ANLI-R2	35.1	36.2 ↑1.1	33.5	33.3 ↓0.2	34.3	35.3 ↑1.0	32.2	35.5 ↑3.3
HANS	51.5	51.4 ↓0.1	50.5	50.5 ↑0.0	50.1	50.0 ↓0.1	50.8	56.5 ↑5.7
WiC	50.0	50.0 ↑0.0	50.2	50.0 ↓0.2	47.8	49.8 ↑2.0	52.4	52.4 ↑0.0
Avg. Acc. (↑)	46.7	47.4 ↑0.7	50.0	50.4 ↑0.4	54.3	55.0 ↑0.7	60.0	60.5 ↑0.5

Table 6: Accuracy (Acc.) on nine downstream tasks evaluated in the zero-shot setting for pretrained LMs with different parameter sizes.

Source: RETRO++: Wang et al., “Shall We Pretrain Autoregressive Language Models with Retrieval? A Comprehensive Study,” ENMNLP 2023.

Other Architectures for RALMs



Recap: The paper which coined the term Retrieval Augmented Generation (RAG)

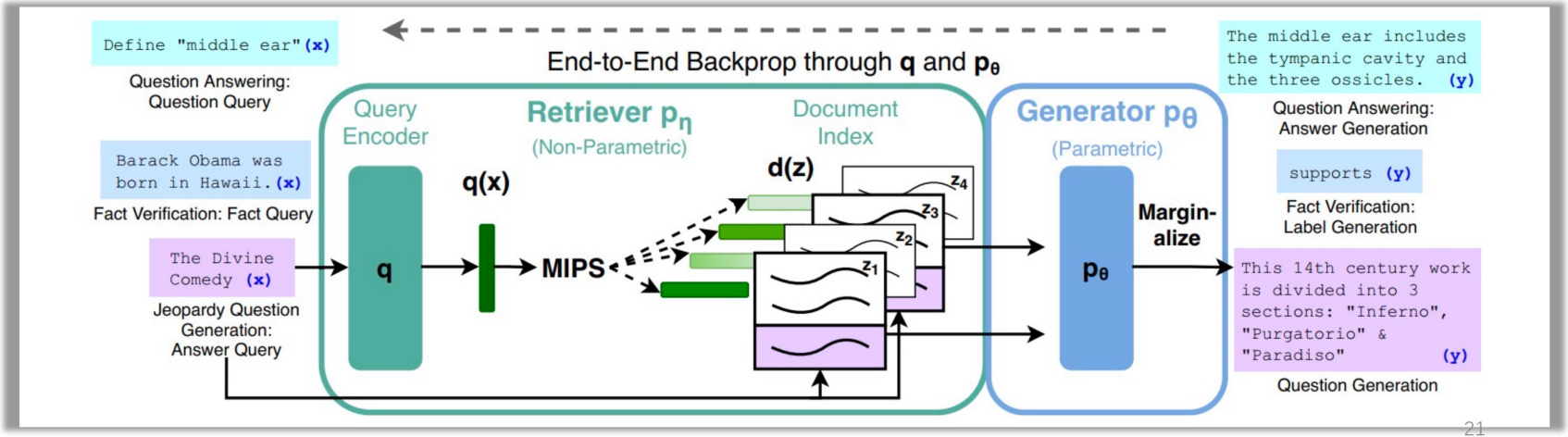
- **RAG**

Integration: concatenating each retrieved passage with the question

Retrieval: DPR + MIPS

Generation: BART





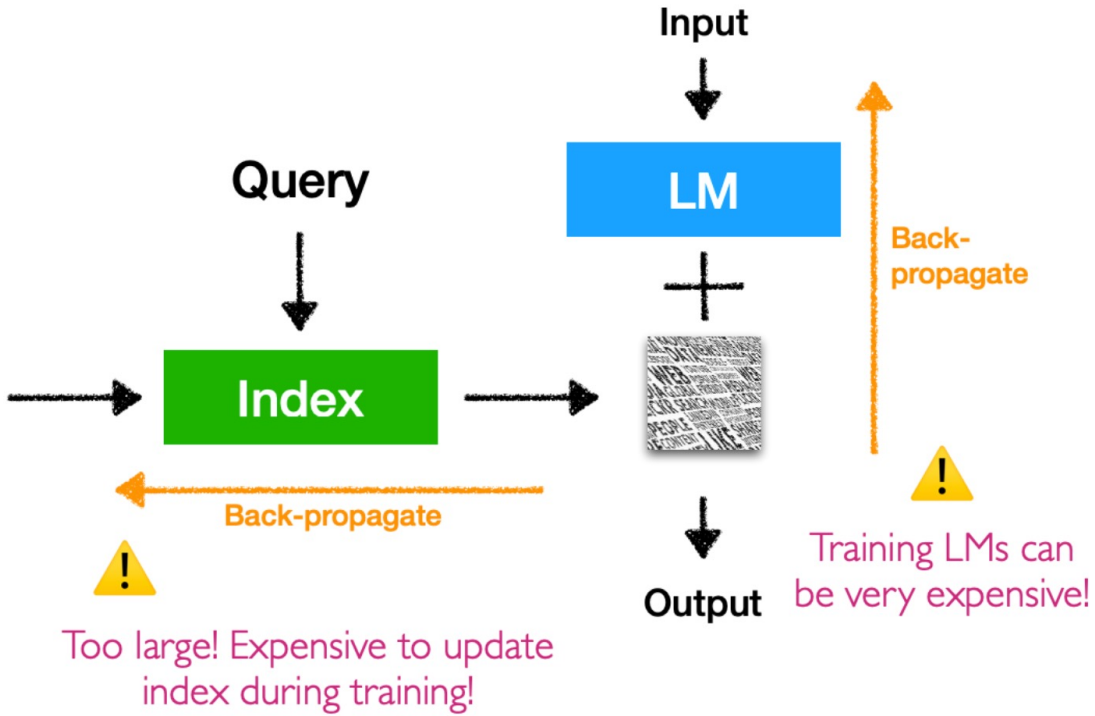
Lewis et al. 2020. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks"

Note: The Retriever and the Generator (i.e. LM) were jointly trained together !!

Why is training so hard for RALMs ?



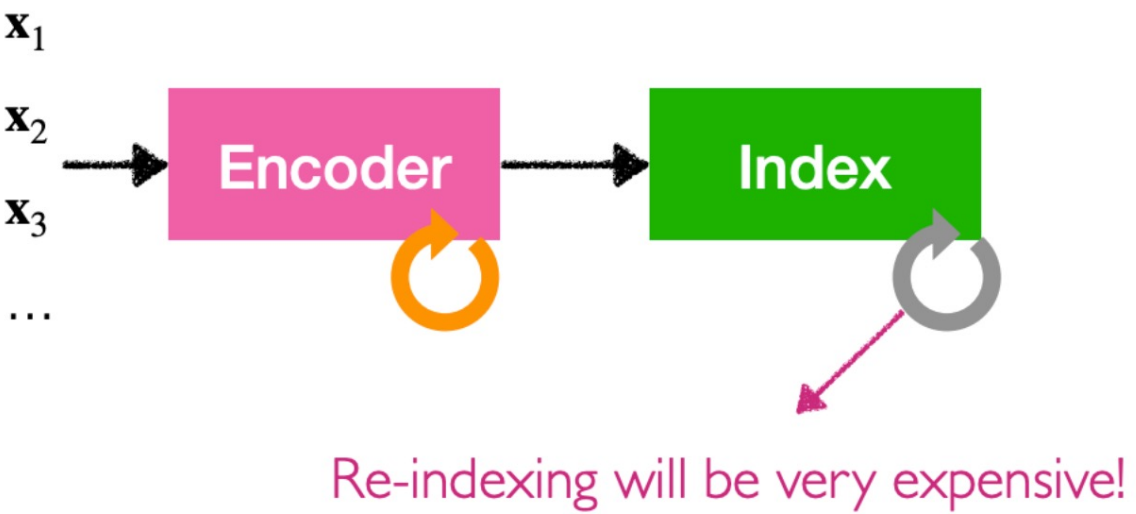
Datastore



Why is training so hard for RALMs ?



Datastore



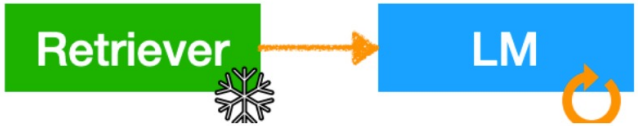
Re-indexing will be very expensive!

Different Training methods

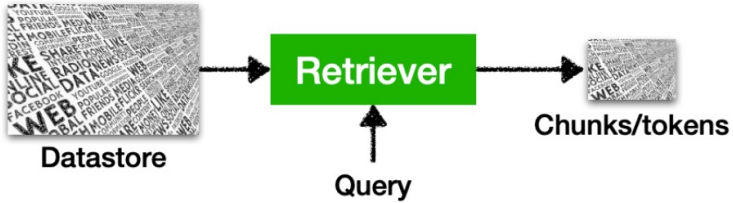
- Independent training
- Sequential training
- Joint training w/ asynchronous index update
- Joint training w/ in-batch approximation

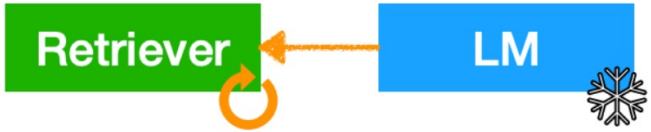
- Training language models





- Training retrieval models

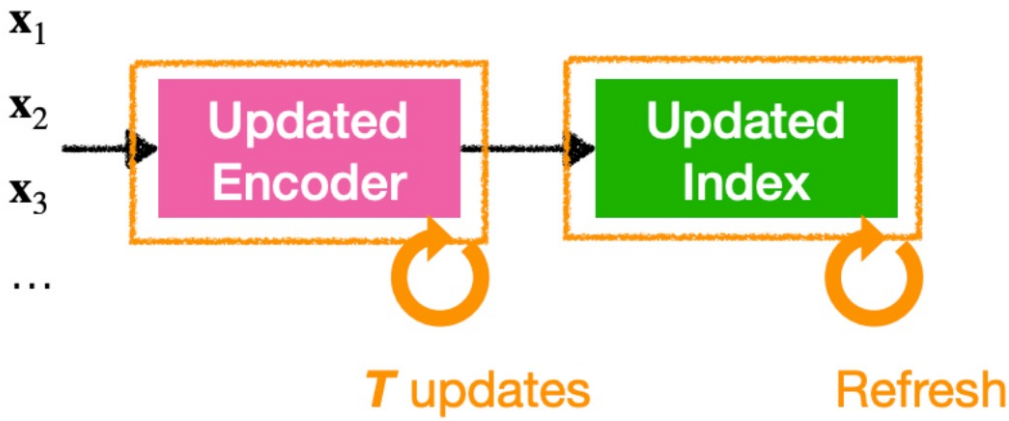




Training with Asynchronous Index Updates



Datastore



Examples: REALM (Guu et al., ICML'20), ATLAS (Izacard et al., JMLR'23)

Training with in-batch approximations

In-batch approximation



Full corpus

Encoder

Full index

>100M

Re-indexing will be very expensive!



Training batch

Encoder

In-batch index

~10K

Computed on the fly for each batch!

Training with in-batch approximations

Training Language Models with Memory Augmentation

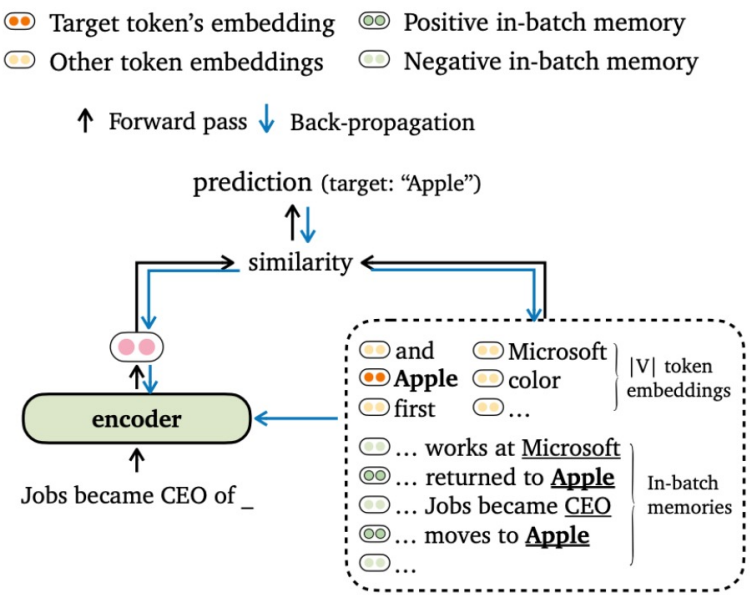
Zexuan Zhong[†] **Tao Lei**[*] **Danqi Chen**[†]
[†]Princeton University
{zzhong, danqic}@cs.princeton.edu, taole@google.com

TRIME (Zhong et al., EMNLP'22)

Nonparametric Masked Language Modeling

Sewon Min[1,2] **Weijia Shi**[1,2] **Mike Lewis**[2] **Xilun Chen**[2]
Wen-tau Yih[2] **Hannaneh Hajishirzi**[1,3] **Luke Zettlemoyer**[1,2]
[1]University of Washington [2]Meta AI [3]Allen Institute for AI
{sewon,swj0419,hannaneh,lsz}@cs.washington.edu
{mikelewis,xilun,scottyih}@meta.com

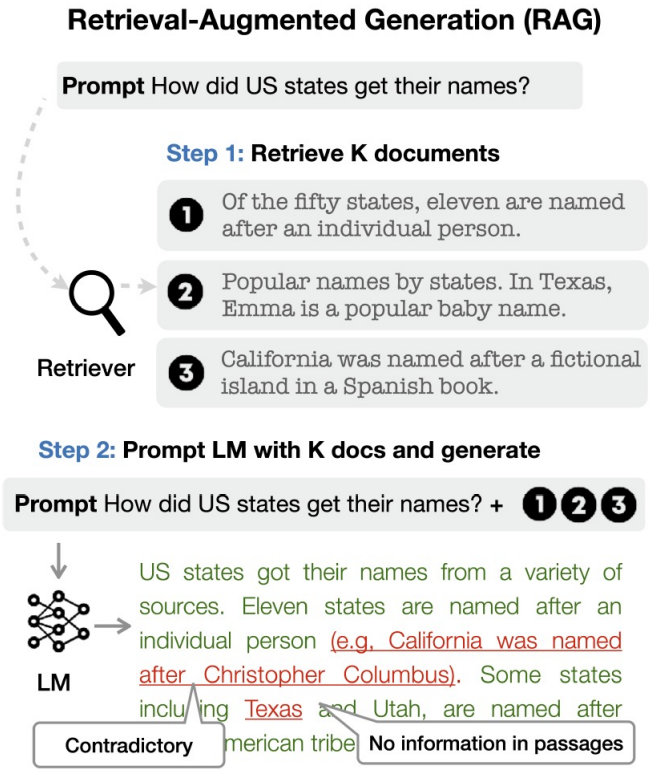
NPM (Min et al., ACL'23 Findings)

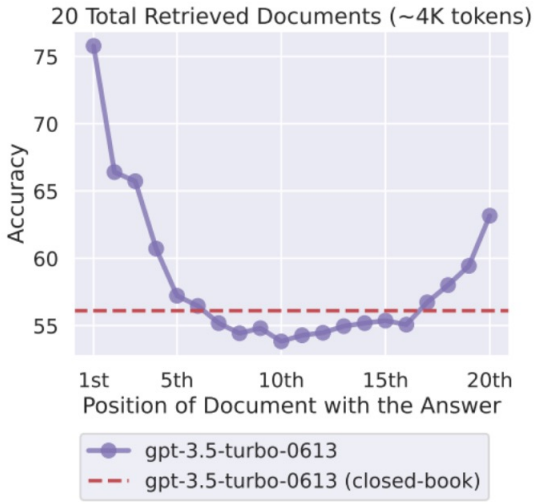


Advanced “Frozen” RAG frameworks (Path #2)

Currently, the dominate approach adopted by Developers & Industry

Key Challenges



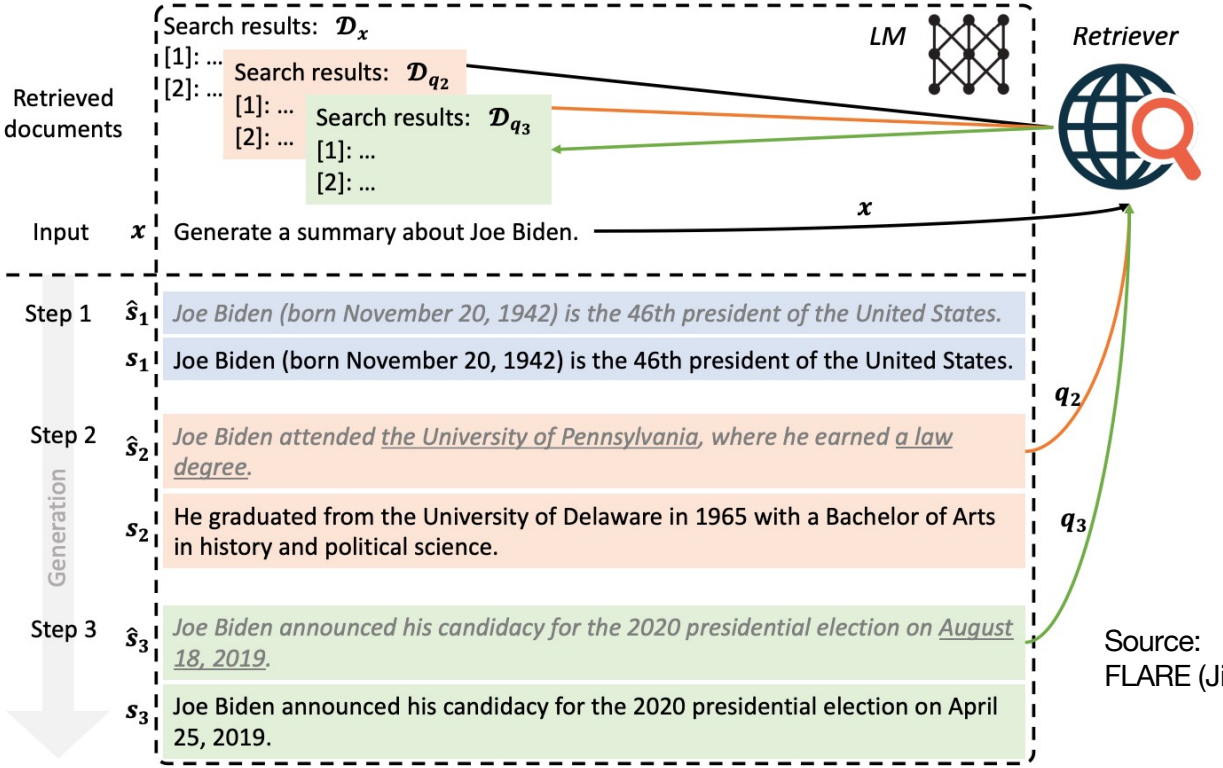


#1. Can LMs **discern** relevant and irrelevant passages?

#2. Can LMs leverage all the contexts **effectively**?

#3. Can LMs **synthesize** information from different passages just in context?

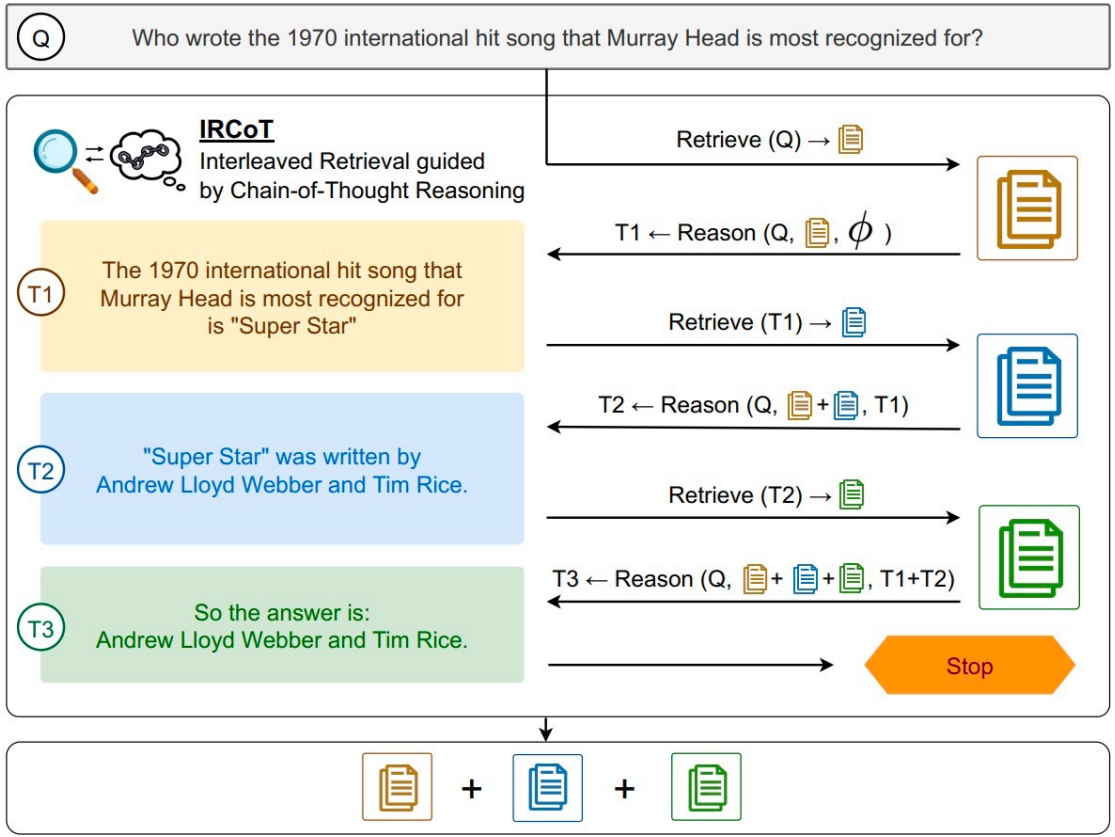
Active Retrieval Augmented Generation (FLARE)

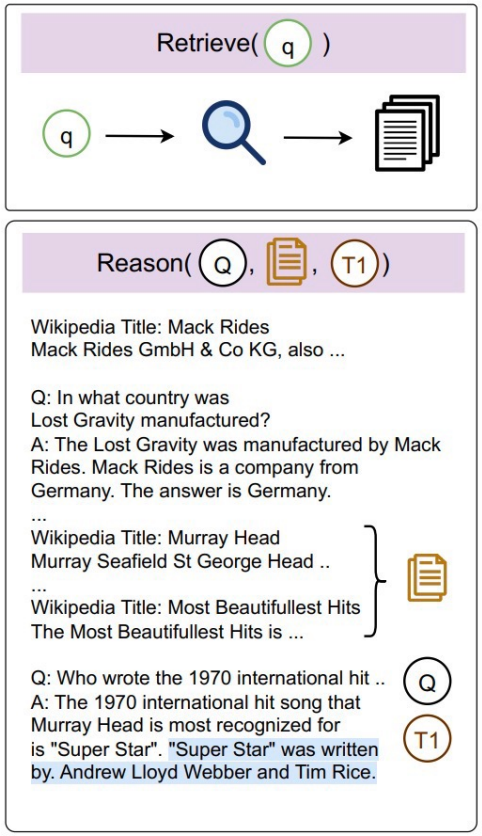


Source:
FLARE (Jiang et al., EMNLP'23)

- The LLM dynamically determines the Timing and Scope of Retrieval.

Yet Another recent RALM: Interleaved Retrieval guided by Chain-of-Thought (IRCoT)





Trivedi, et al. 2023. “Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions”

Promises and Limitations of Retrieval-Augmented LLMs !

- Parametric LMs have numerous challenges to build reliable systems.
- Retrieval-augmented LMs such as RAG can effectively address them.

When Not to Trust Language Models: Investigating Effectiveness of Parametric and Non-Parametric Memories (ACL 2023)
https://arxiv.org/abs/2403.03187

Reliable, adaptable and attributable LMs with retrieval (Arxiv 2024)
https://arxiv.org/abs/2403.03187

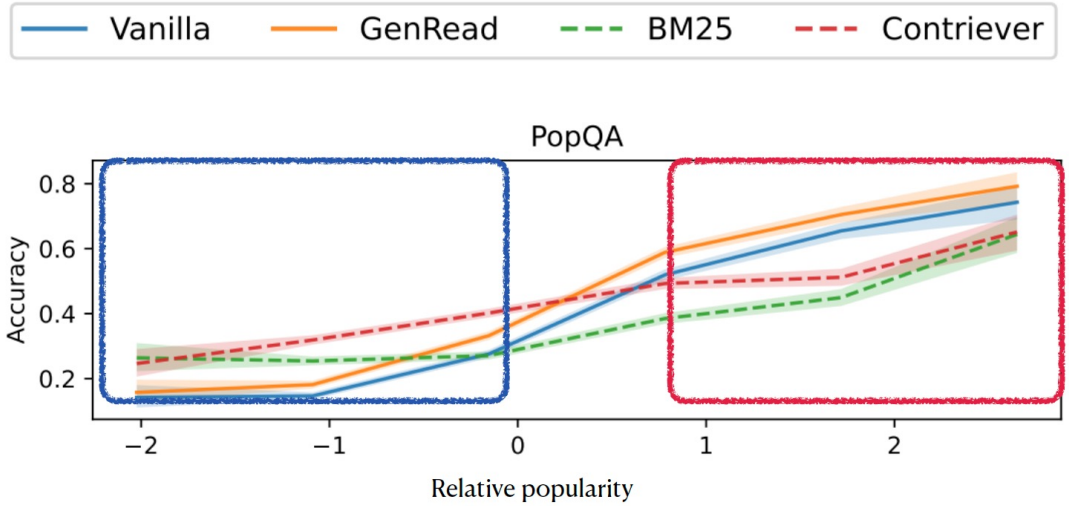
Let's talk about how we can improve traditional RAG!

RAG hurts in Popular Knowledge memorized by Vanilla LLMs

On popular facts vanilla ≈ RAG or vanilla > RAG

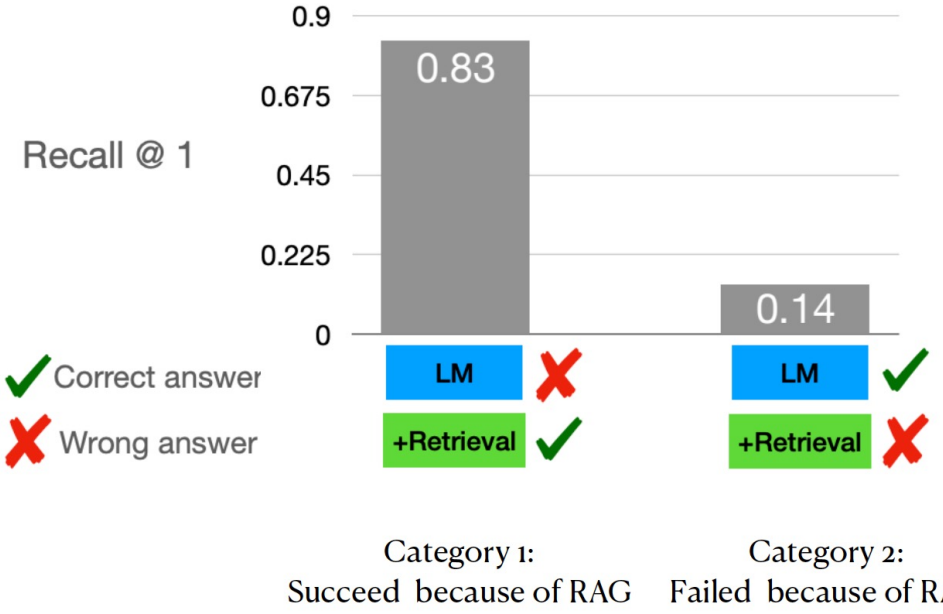
Why?

On long-tail facts RAG > vanilla



Incorrect Retrieval can easily confuse LLMs

Failed RAG can counterfactually make LMs answer incorrectly





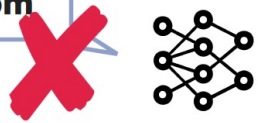
In what city was Zijah Sokolovic born?





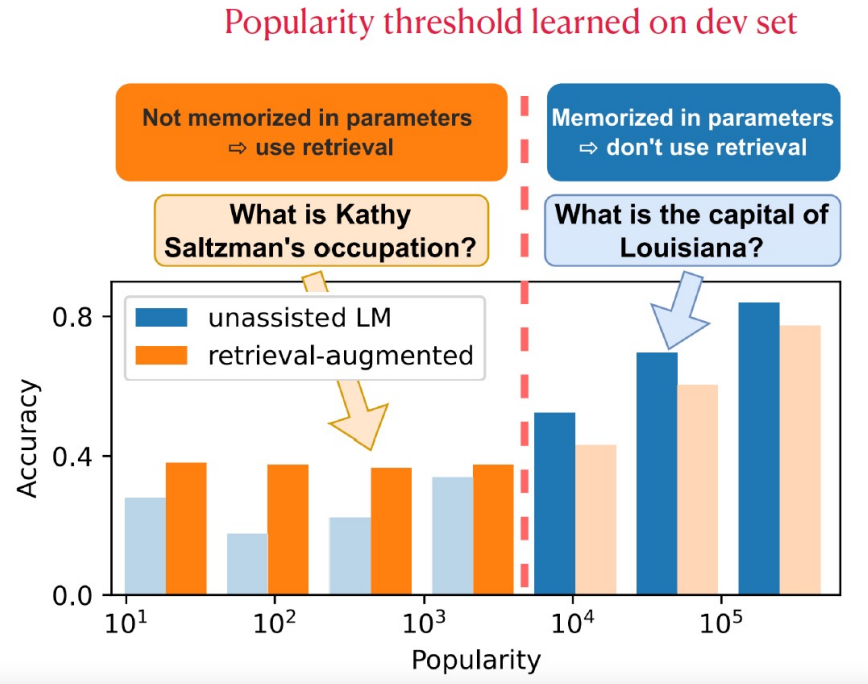
Ali Sokol ... (born 8 May 1921 in **Orahovac, Kingdom of Serbs**, Croats and Slovenes, died 23 September 1974) was a Yugoslav pulmonologist

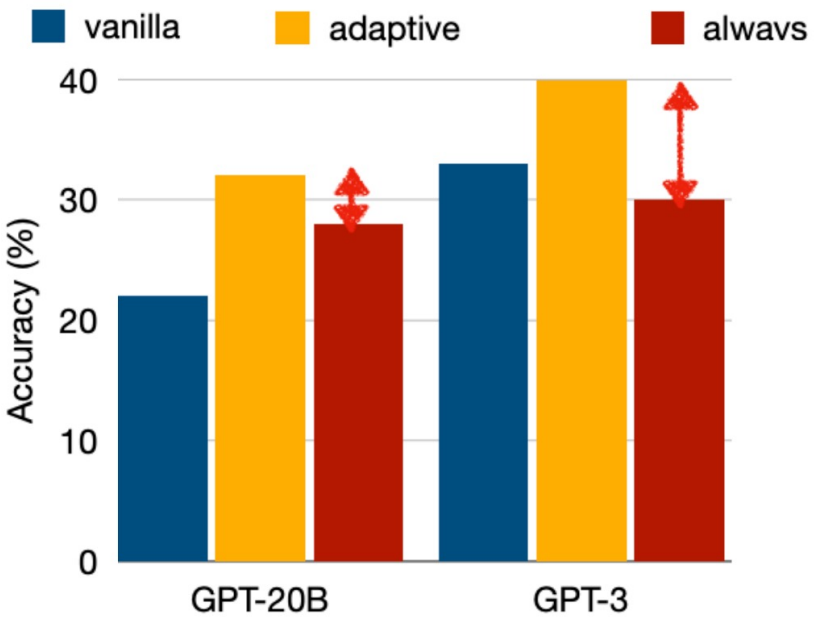
Zijah Sokolovic´ was born in **Orahovac, Kingdom**



Simple Hack to fix it: Threshold-based Adaptive Retrieval

Simple adaptive RAG significantly improve RAG performance & efficiency





Challenges remain for current Naïve RAG systems: Versatility

What are the latest discoveries from the James Webb Space Telescope?

The James Webb Space Telescope is designed to peer into the dusty clouds of gas where stars and planetary systems are born. Webb has captured the first direct image of an exoplanet, and the Pillars of Creation in the Eagle Nebula[1][2]. Additionally, the telescope will be used to study the next interstellar interloper[3].

(*Some generated statements may not be fully supported by citations, while others are fully supported.)

Cited Webpages

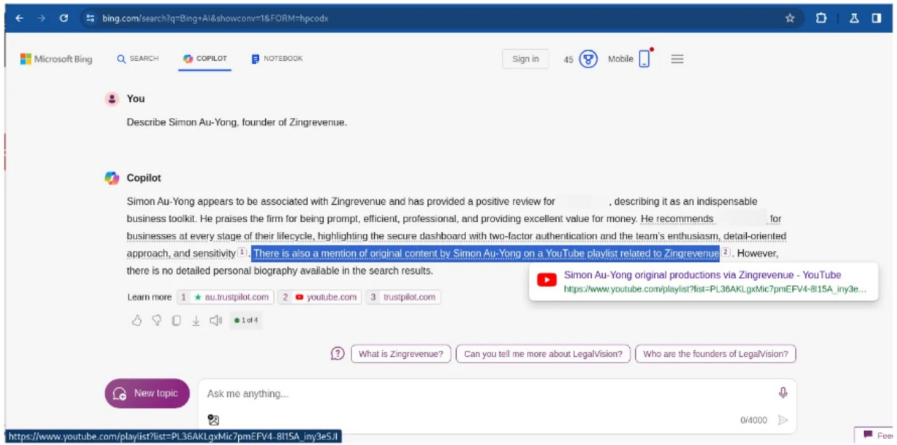
[1]: nasa.gov (✗citation does not support its associated statement)
NASA's Webb Confirms Its First Exoplanet
... Researchers confirmed an exoplanet, a planet that orbits another star, using NASA's James Webb Space Telescope for the first time. ...

[2]: cnn.com (⚠citation partially supports its associated statement)
Pillars of Creation: James Webb Space Telescope ...
... The Pillars of Creation, in the Eagle Nebula, is a star-forming region captured in a new image (right) by the James Webb Space Telescope that reveals more detail than a 2014 image (left) by Hubble ...

[3]: nasa.gov (✅citation fully supports its associated statement)
Studying the Next Interstellar Interloper with Webb
...Scientists have had only limited ability to study these objects once discovered, but all of that is about to change with NASA's James Webb Space Telescope...The team will use Webb's spectroscopic capabilities in both the near-infrared and mid-infrared bands to study two different aspects of the interstellar object.

Liu et al. Evaluating Verifiability in Generative Search Engines. Findings of EMNLP 2023.

> I asked Bing Copilot to describe me. It did and said that there is a mention of original content by Simon Au-Yong on a YouTube playlist related to Zingrevenue (my company). The link is at the bottom of the screenshot and there is a button that should send me to that playlist. But the playlist is made up.



Marcus. No, RAG is probably not going to rescue the current situation. 2024.

When Users Shift from traditional QA to Open-ended Instructions



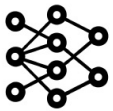
How many of US states got their names from an individual person?

Simple factoid question



Of the fifty states, eleven are named after an individual person. Six of those are named in honor of European monarchs: the two Carolinas, the two Virginias, Georgia, and Louisiana.

Eleven



Short answer based on single document

When Users Shift from traditional QA to Open-ended Instructions



Make a table for me summarizing how different US states got their names, grouping them together.

Group	States
Named after indigenous communities	Alabama, Alaska, Arizona, Arkansas, Colorado, Connecticut, Delaware, Florida, Hawaii, Idaho, Illinois, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Mexico, North and South Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming
Named after royalty and other important figures	Georgia, Louisiana, Maryland, New York, North and South Carolina, Pennsylvania, Virginia, Washington, West Virginia

Hint: Only 10 states here…?

10 references

WIKIPEDIA
LIST OF STATE AND TERRITORY NAME ETY…

BIA
ORIGIN OF NAMES OF US STATES | INDIA…

MENTALFLOSS
HOW ALL 50 STATES GOT THEIR NAMES | …

cohere

Is this fully correct?

When Users Shift from traditional QA to Open-ended Instructions



Make a table for me summarizing how different US states got their names, grouping them together.

Group	States
Named after indigenous communities	Alabama, Alaska, Arizona, Arkansas, Colorado, Connecticut, Delaware, Florida, Hawaii, Idaho, Illinois, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Mexico, North and South Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming
Named after royalty and other important figures	Georgia, Louisiana, Maryland, New York, North and South Carolina, Pennsylvania, Virginia, Washington, West Virginia

DELAWARE: **Named for Lord De La Warr**, first governor and captain-general of Virginia, who in 1630 explored the bay and river area where his name was first applied.

Indian Affairs (.gov)
https://www.bia.gov › as-ia › opa › online-press-release

Origin of Names of US States | Indian Affairs

10 references

WIKIPEDIA
LIST OF STATE AND TERRITORY NAME ETY...

BIA
ORIGIN OF NAMES OF US STATES | INDIA...

MENTALFLOSS
HOW ALL 50 STATES GOT THEIR NAMES | ...

cohere

When Users Shift from traditional QA to Open-ended Instructions



Make a table for me summarizing how different US states got their names, grouping them together.

Group	States
Named after indigenous communities	Alabama, Alaska, Arizona, Arkansas, Colorado, Connecticut, Delaware, Florida, Hawaii, Idaho, Illinois, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Mexico, North and South Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming
Named after royalty and other important figures	Georgia, Louisiana, Maryland, New York, North and South Carolina, Pennsylvania, Virginia, Washington, West Virginia



10 references

WIKIPEDIA
LIST OF STATE AND TERRITORY NAME ETY...

BIA
ORIGIN OF NAMES OF US STATES | INDIA...

MENTALFLOSS
HOW ALL 50 STATES GOT THEIR NAMES | ...

cohere

When Users Shift from traditional QA to Open-ended Instructions



Make a table for me summarizing how different US states got their names, grouping them together.

Complex instruction

Group	States
Named after indigenous communities	Alabama, Alaska, Arizona, Arkansas, Colorado, Connecticut, Delaware, Florida, Hawaii, Idaho, Illinois, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Mexico, North and South Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming
Named after royalty and other important figures	Georgia, Louisiana, Maryland, New York, North and South Carolina, Pennsylvania, Virginia, Washington, West Virginia

Long-form answers

Requiring aggregating multiple evidence

10 references

WIKIPEDIA
LIST OF STATE AND TERRITORY NAME ETY...

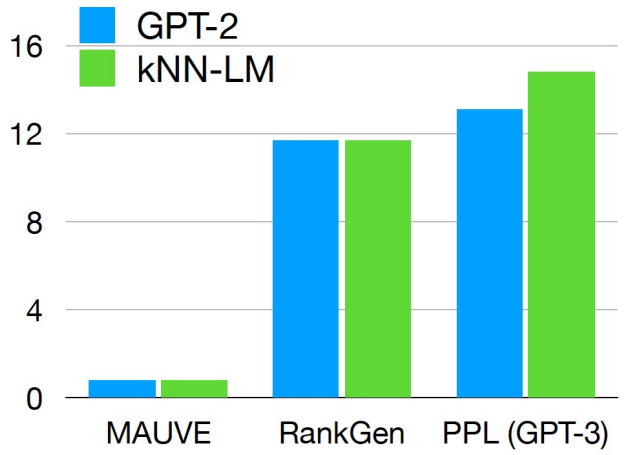
BIA
ORIGIN OF NAMES OF US STATES | INDIA...

MENTALFLOSS
HOW ALL 50 STATES GOT THEIR NAMES | ...

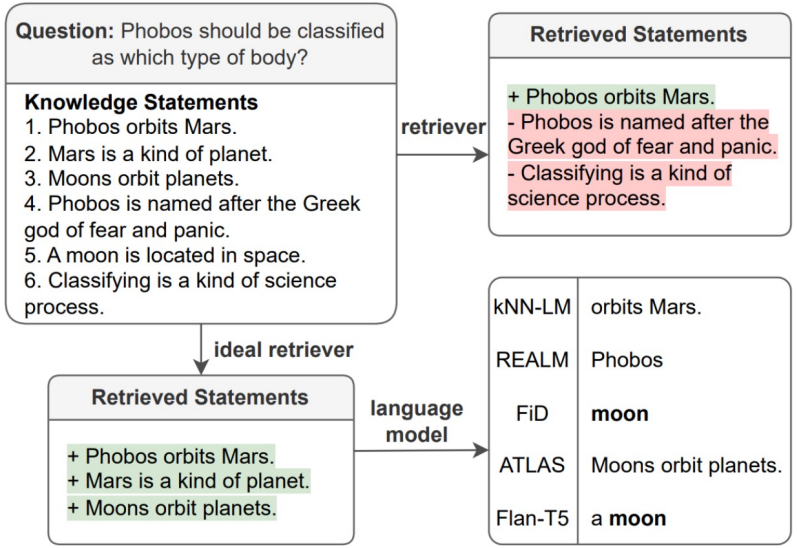
cohere

Challenges remain for current Naïve RAG systems: Reliability

Limited effectiveness beyond information-seeking QA-like tasks



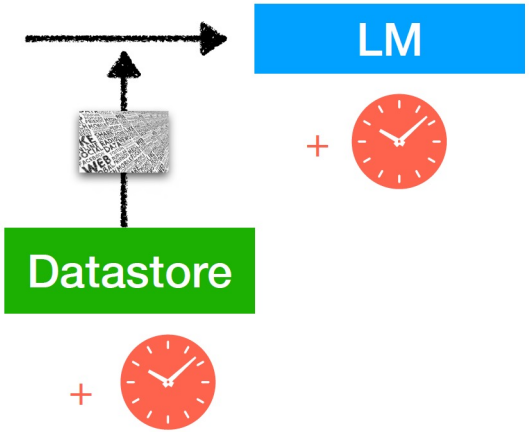
Wang et al. kNN-LM Does Not Improve Open-ended Text Generation. ACL 2023.

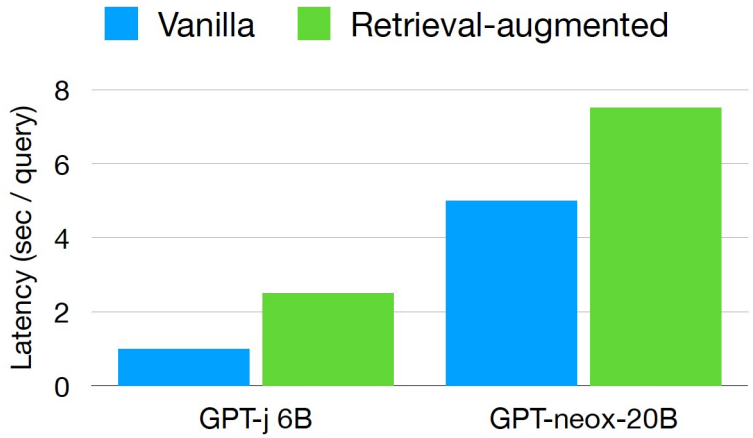


BehnamGhader et al. Can Retriever-Augmented Language Models Reason? The Blame Game Between the Retriever and the Language Model. EMNLP Findings 2023.

RAG increases Latency !

Efficiency | Effective datastore | New application



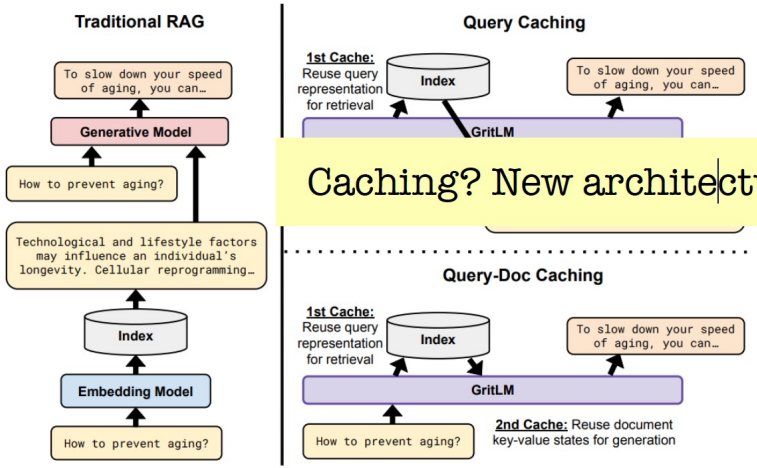


Mallen*, Asai* et al., When Not to Trust Language Models:
Investigating Effectiveness of Parametric and Non-Parametric Memories 2023.

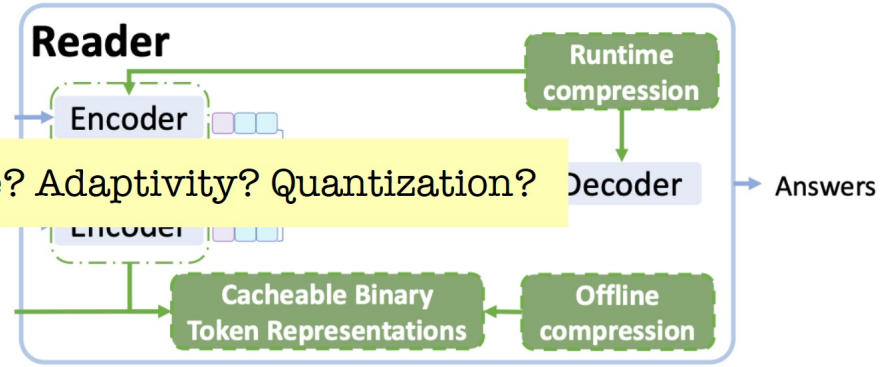
Need Efficient Algorithms/ Architecture for RAG

Efficiency | Effective datastore | New application

GRIT
(Muennighoff et al., 2024)



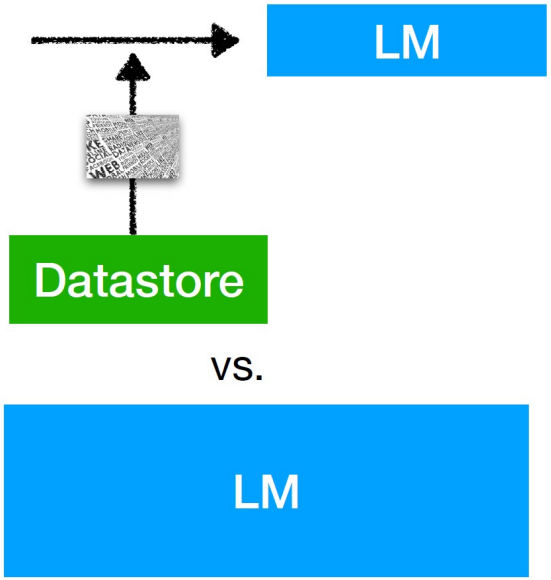
BTR
(Cao et al., 2024)

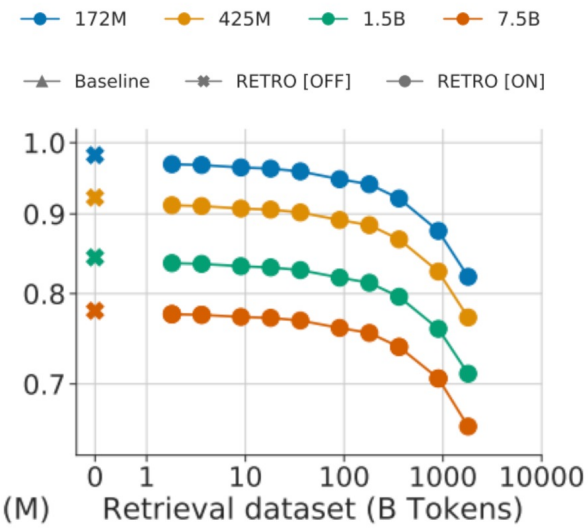


Caching? New architecture? Adaptivity? Quantization?

Need more Scalable Data-store for RAG

Efficiency | **Effective datastore** | New application

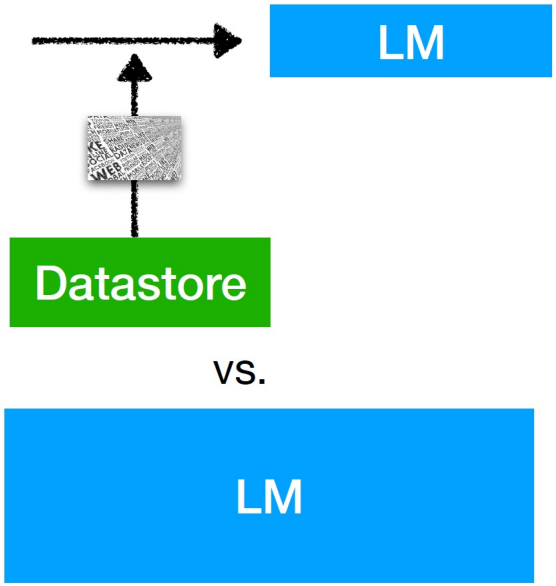




RETRO
(Borgeaud et al., 2021)

Need more Scalable Data-store for RAG

Efficiency | **Effective datastore** | New application



How should we scale RAG?

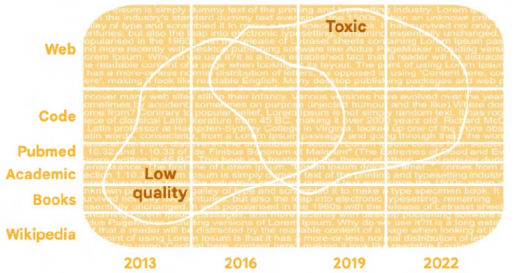
	LM: # of parameters	Datastore: # of tokens
kNN-LM (Khandelwal et al., 2020)	250M	≤ 3B
NPM (Min et al., 2023)	350M	1B
Atlas (Izacard et al., 2022)	11B	~30B
RETRO (Borgeaud et al., 2021)	7B	2T
REPLUG (Shi et al., 2023)	≤ 175B	~5B

Need more Scalable Data-store for RAG

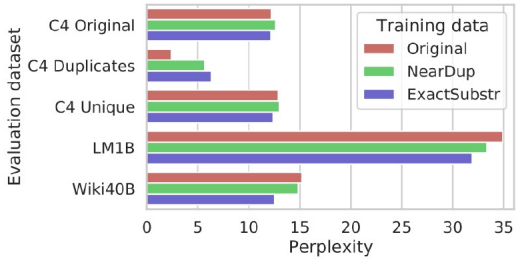
Efficiency | **Effective datastore** | New application

Data-centric approaches to build effective large-scale datastore?

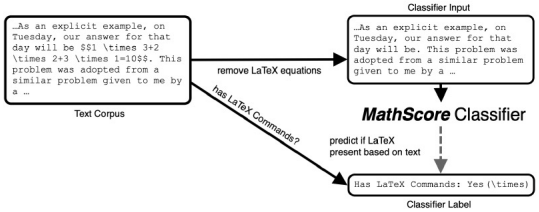
Quality & Composition
(Longpre et al., 2023)



Deduplication
(Lee et al., 2023)



Data Filtering
(Paster et al., 2023)



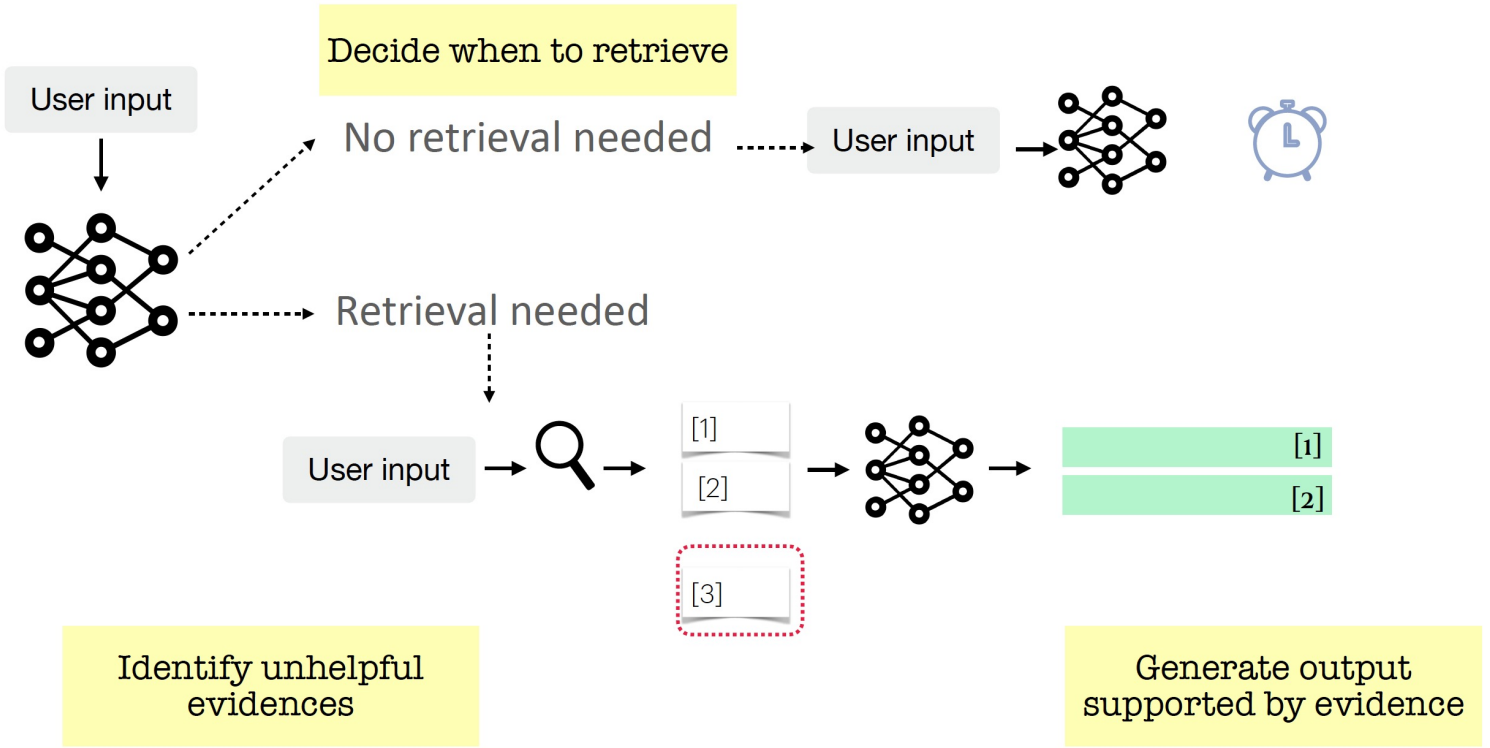
An Example of SOTA RAG approach:

Self-Reflective Retrieval-Augmented Generation (Self-RAG)

Source: Self-RAG (Asai et al., ICLR 2024)

Self-Reflective Retrieval-Augmented Generation (Self-RAG)

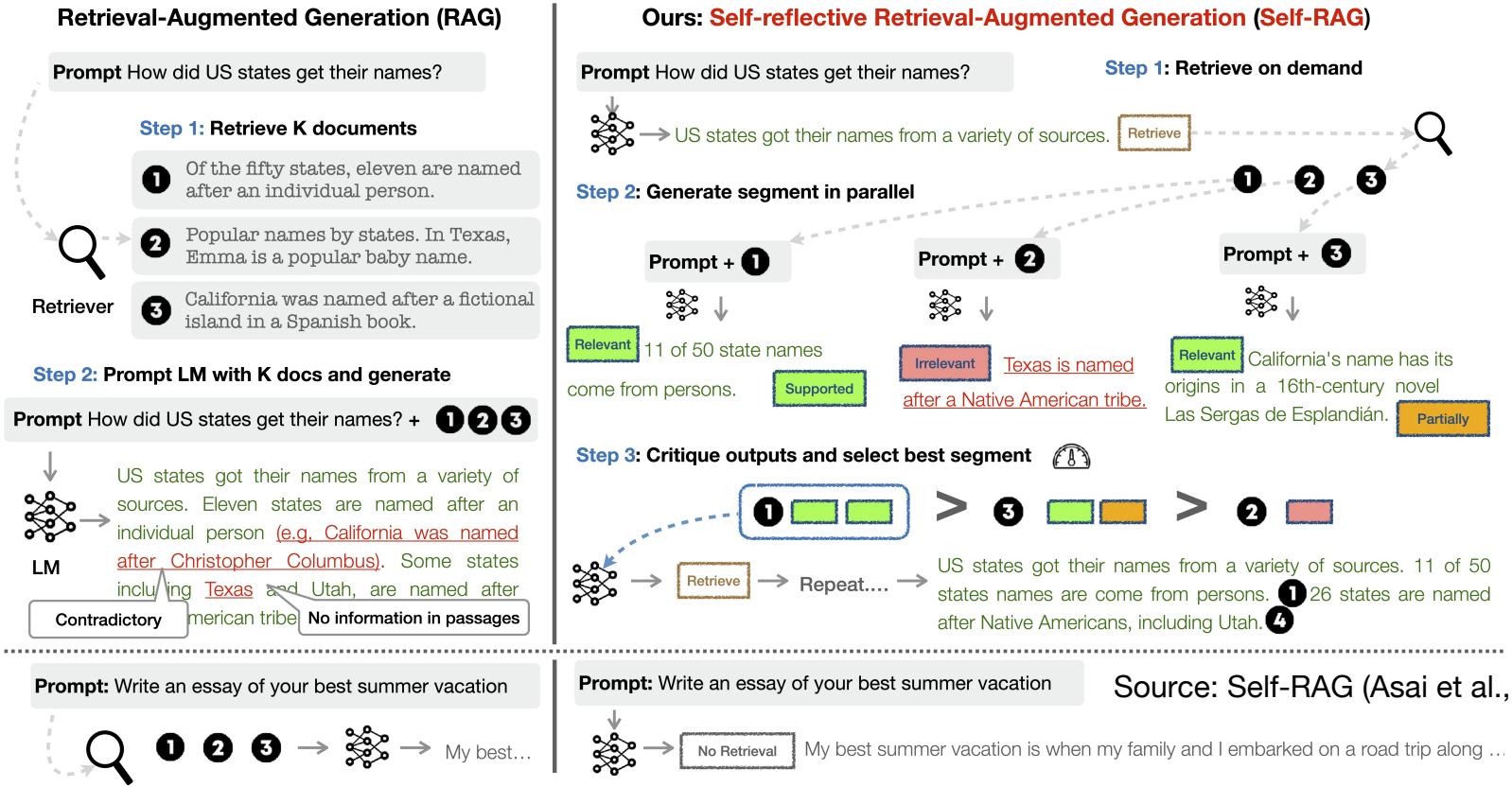
Goal of Self-RAG — Learn to Retrieve, Generate and Critique



Source: Self-RAG (Asai et al., ICLR 2024)

Self-Reflective Retrieval-Augmented Generation (Self-RAG)

- Self-RAG is an LLM which has learnt (been trained) to *judiciously* retrieve, generate and critique its (initial) outputs to enhance final responses through self-reflections !



Source: Self-RAG (Asai et al., ICLR 2024)

How to train Self-RAG to be so powerful and intelligent ?

STEP 1: Use a Frontier LLM (GPT-4) to create a supervised dataset D_{critic} by designing prompts to drive GPT-4 to generate "*reflection tokens*", which serve as commands / markers to instruct an arbitrary LLM to *retrieve* and *critique* on whether the "preliminary" output generated by an LLM is *relevant*, *grounded (supported by evidence)* and *useful*, and also to ask the LLM to *generate* revised output accordingly.

STEP 2: The supervised dataset D_{critic} from STEP 1 serves as "demonstrating examples" to train (i.e. *instruction-tune*) a new in-house *Critic LLM, C,* to perform the same task as GPT-4 in STEP 1: this can be seen as "distilling" the behavior of GPT-4 to the new *Critic LLM, C*.

STEP 3: The Critic LLM, *C*, is used to generate a new training dataset D_{gen} which include prompts, responses, as well as the *reflection tokens* defined in STEP 1, *in*serted by *C.*

STEP 4: The training dataset D_{gen} from STEP 3 is used to train a new "Generator LLM", *M*, which can automatically carry out retrieval, critique and output revision actions as part of its regular "next token prediction" behavior as standard LLM, where its set of "next tokens" can now include action-triggering tokens such as *retrieve, critique, generate,* as defined in STEP 1.

How to train Self-RAG to be so powerful and intelligent ?

Algorithm 2 SELF-RAG Training

Input input-output data $\mathcal{D} = \{X, Y\}$, generator $\mathcal{M}$, $\mathcal{C}$ θ
Initialize $\mathcal{C}$ with a pre-trained LM
Sample data $\{X^{sample}, Y^{sample}\} \sim \{X, Y\}$ ▷ **Training Critic LM (Section 3.2.1)**
for $(x, y) \in (X^{sample}, Y^{sample})$ **do** ▷ Data collections for $\mathcal{C}$
    Prompt GPT-4 to collect a reflection token r for (x, y)
    Add $\{(x, y, r)\}$ to $\mathcal{D}_{critic}$
Update $\mathcal{C}$ with next token prediction loss ▷ Critic learning; Eq. 1
Initialize $\mathcal{M}$ with a pre-trained LM ▷ **Training Generator LM (Section 3.2.2)**
for $(x, y) \in (X, Y)$ **do** ▷ Data collection for $\mathcal{M}$ with $\mathcal{D}_{critic}$
    Run $\mathcal{C}$ to predict r given (x, y)
    Add (x, y, r) to $\mathcal{D}_{gen}$
Update $\mathcal{M}$ on $\mathcal{D}_{gen}$ with next token prediction loss ▷ Generator LM learning; Eq. 2

Source: Self-RAG (Asai et al., ICLR 2024)

How to create the Training Dataset (D_{gen}) for the Generator module (M) of Self-RAG

Algorithm 3 $\mathcal{M}_{gen}$ Data creation

Input Input-output data $\mathcal{D} = X, Y$
for $(x, y) \in \{X, Y\}$ **do**
    Given (x, y) $\mathcal{C}$ predicts Retrieve
    if Retrieve is predicted **then**
        Retrieve relevant passages $\mathbf{D}$ using $\mathcal{R}$ given (x, y) ▷ Retrieve passages
        for $d \in \mathbf{D}$ **do**
            $\mathcal{C}$ predicts IsRel for each d ▷ Predict relevance of passages
            $\mathcal{C}$ predicts IsSup for each (y, d) ▷ Predict supports of outputs
        $\mathcal{C}$ predicts IsUse for each d ▷ Predict overall utility ($t = \mathcal{T}$ only)
        Sample d
    else if Retrieve is not predicted **then**
        $\mathcal{C}$ predicts IsUse given x, y
    Add augmented (x, y, d, r) to $\mathcal{D}_{gen}$

Source: Self-RAG (Asai et al., ICLR 2024)

Input Datasets used by Critic-LM C to generate D_{gen}

Dataset name	category	Data source	the number of instances
GPT-4 Alpaca	Instruction-following	Open-Instruct	26,168
Stanford Alpaca	Instruction-following	Open-Instruct	25,153
FLAN-V2	Instruction-following	Open-Instruct	17,817
ShareGPT	Instruction-following	Open-Instruct	13,406
Open Assistant 1	Instruction-following	Open-Instruct	9,464
Wizard of Wikipedia	Knowledge-intensive	KILT	17,367
Natural Questions	Knowledge-intensive	KILT	15,535
FEVER	Knowledge-intensive	KILT	9,966
OpenBoookQA	Knowledge-intensive	HF Dataset	4,699
Arc-Easy	Knowledge-intensive	HF Dataset	2,147
ASQA	Knowledge-intensive	ASQA	3,897

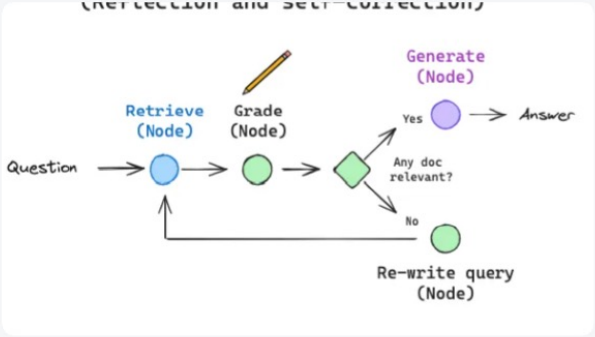
Table 3: The generator LM $\mathcal{M}$ training data statistics.

base LM	Retrieve	ISSUP	ISREL	ISUSE
Llama2-7B	**93.8**	**93.5**	80.2	**73.5**
FLAN-3B	85.6	73.1	**82.0**	72.1

Figure 5: Reward prediction accuracy using GPT-4 predictions as ground-truth predictions.

Real-World Impact of Self-RAG

LangChain By LangChain Release Notes Case Studies LangChain GitHub Docs Sign in Subscribe



Self-Reflective RAG with LangGraph

6 MIN READ FEB 7, 2024

Integrated into major RAG libraries (LlamaIndex, LangChain ... etc)

LlamaIndex • Feb 13, 2024

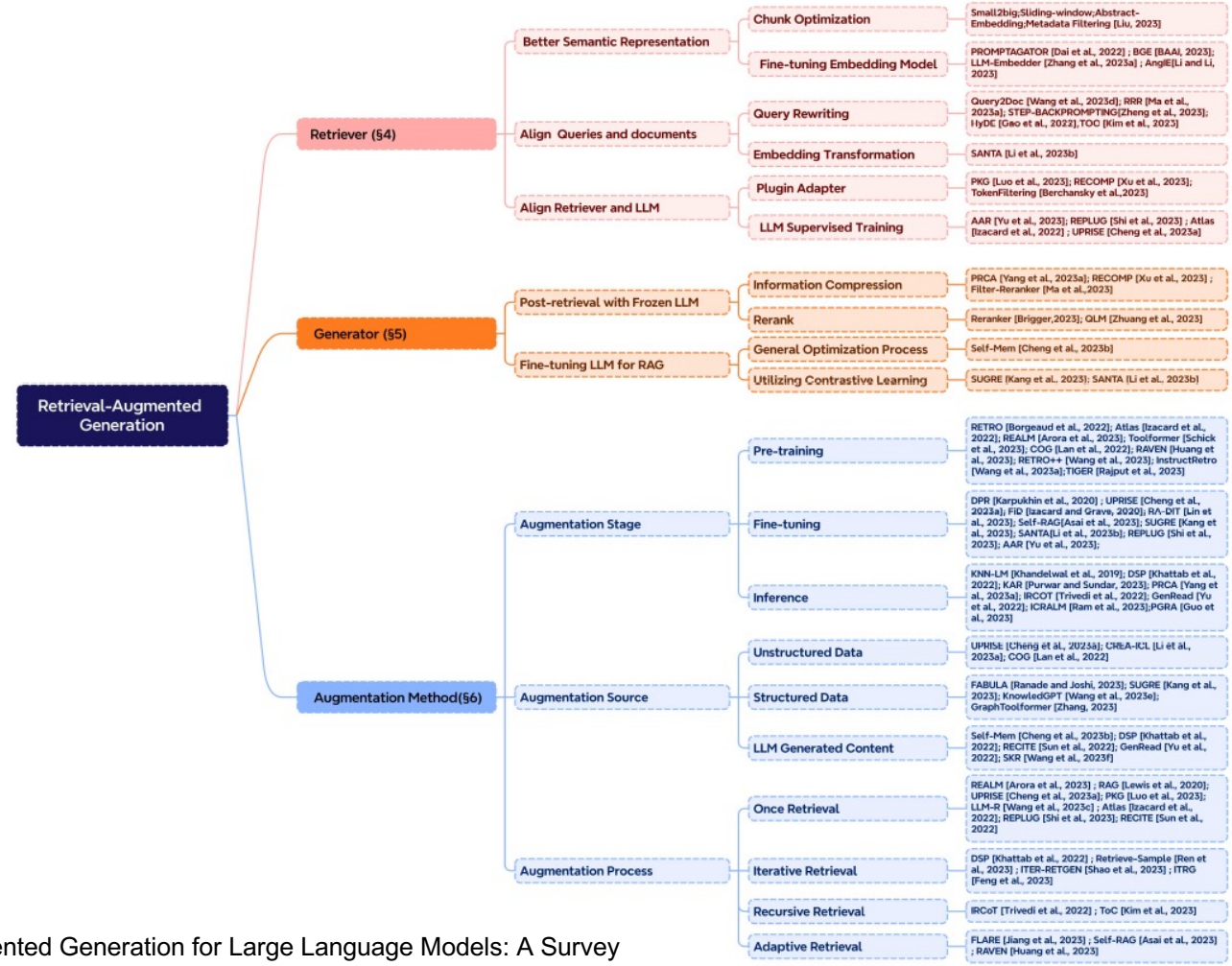
LlamaIndex Newsletter 2023–02–13

Newsletter Llamaindex AI Rag LLM

🤩 **The highlights:**

1. **Self-RAG**: Introducing Self-RAG, now part of LlamaIndex as a LlamaPack. Boosts LLM training and RAG workflows with dynamic capabilities. Notebook, Tweet.

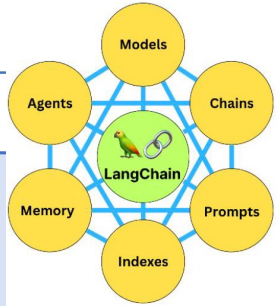
Summary of RAG related Research



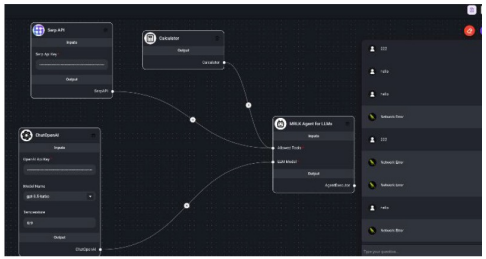
Source: Retrieval-Augmented Generation for Large Language Models: A Survey

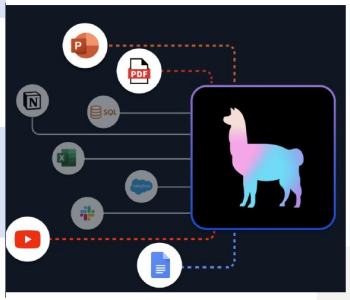
Existing RAG Stacks in the Industry

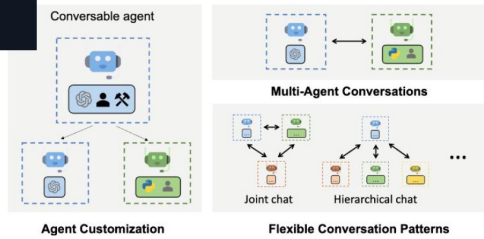
Name	Pros	Cons
LangChain	Modular, full-featured	Inconsistent behavior ,API conceals details,complexity and low flexibility.
LlamaIndex	Focus on RAG	Requires combination use, low customization.
FlowiseAI	Easy to get started, visualized workflows.	Does not support complex scenarios.
AutoGen	Adapts to multi-agent scenarios.	Low efficiency, requires multiple rounds of dialogue.



LangChain


FlowiseAI


LlamaIndex



AutoGen

Retrieval-Augmented LMs (RALMs) have been widely deployed

Retrieval-augmented LMs have been widely used both in academia and industry



cohere

Gemini



perplexity

How RAG-Powered AI Applications Have A Positive Impact On Businesses

Forbes Technology Council

Jyotishko Biswas Forbes Councils Member

Forbes Technology Council COUNCIL POST | Membership (Fee-Based)



BAIR

BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

Subscribe About Archive BAIR

The Shift from Models to Compound AI Systems

"60% of LLM applications use some form of retrieval-augmented generation (RAG)"

RAG to enable NEW applications

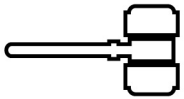
Efficiency | Scaling datastores | **New application**



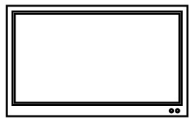
Medical



Science



Legal



Code



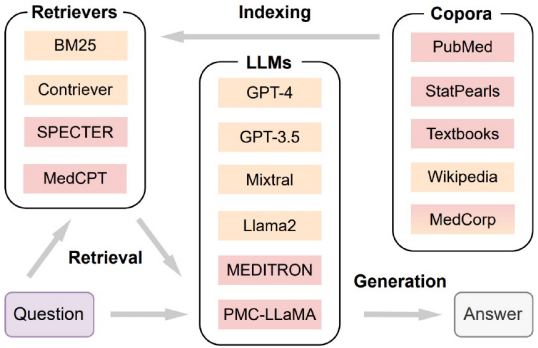
Finance

RAG to enable NEW applications

Efficiency | Scaling datastores | **New application**



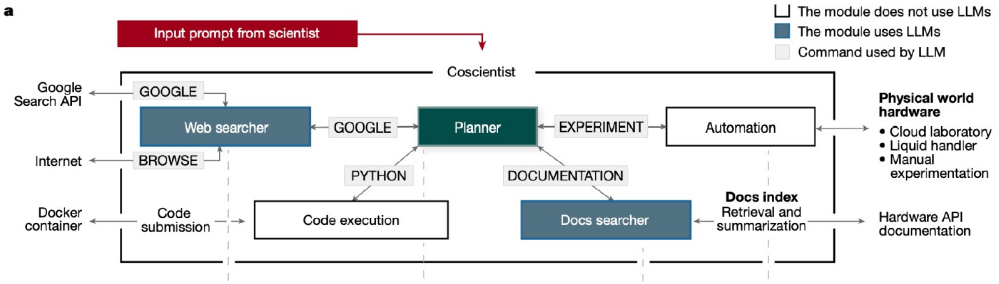
MedRAG
(Xiong et al., 2024)



Realistic evaluation?



Automous chemistry experiments
(Boiko et al., 2024)

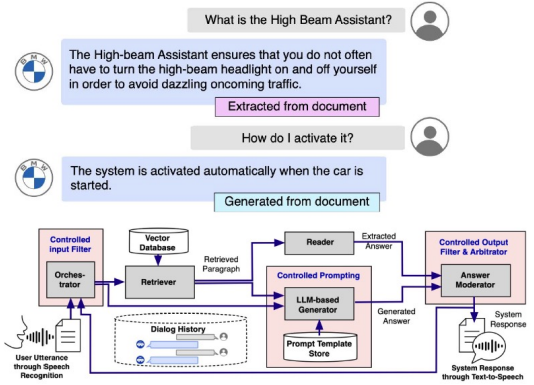


RAG pipelines optimized for expert tasks?

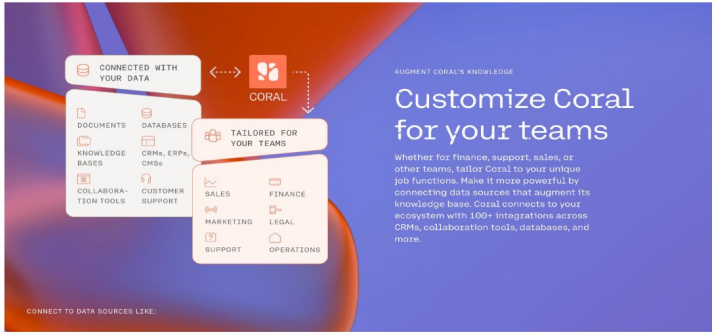
RAG Applications in the Industry



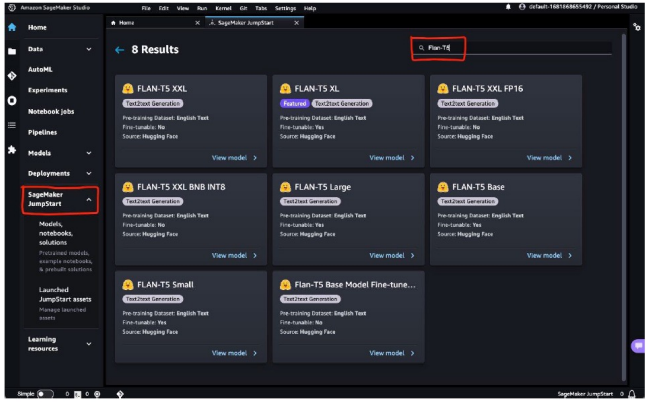
NetEase - ChatBI



BMW - CarExpert



Cohere - Coral

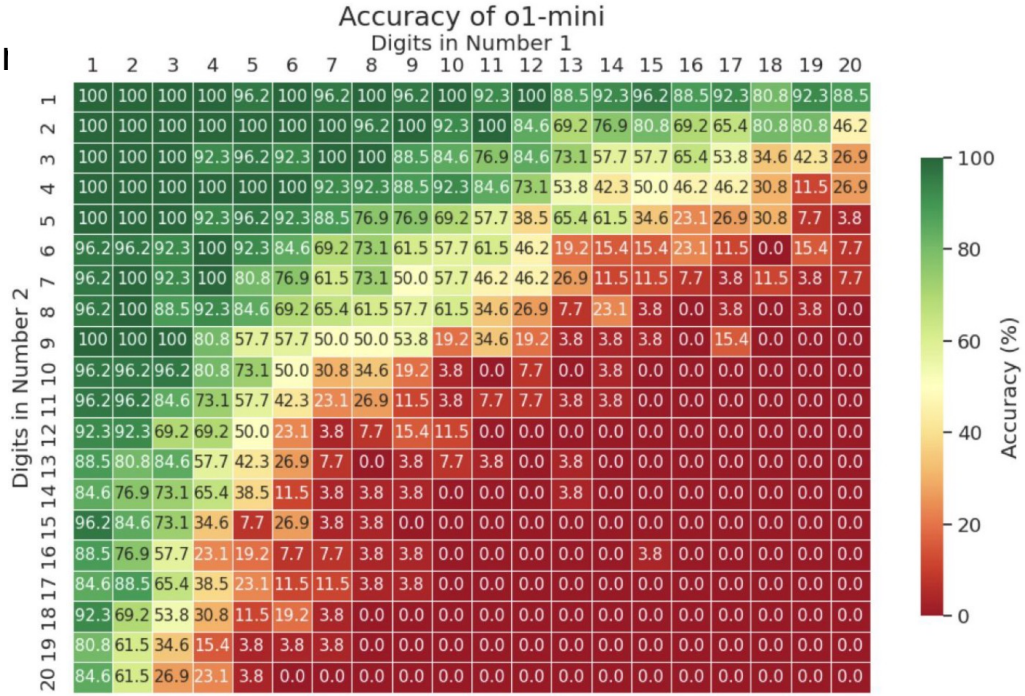


Amazon - Kendra

Tool Uses for LLM

Why LLMs need to be good Tool Users ?

- LLMs are not the solution for everything (yet ☺) !
  - GPTo1 can't even solve multiplicatio of 10+ digits !
  - But does it really need to ?
  - Humans use a calculator to do so …
  - Why shouldn't LLMs ?



Multiplication Accuracy of OpenAI O1 (Yuantian Deng, X)

Why Tools: Things LLMs are Bad at

- Numerical/symbolic operations
    1. Calculation
    2. Logic deduction
    3. Exact operations
- Knowledge not in their pretraining corpus
    1. Tail factual knowledge
    2. New information
    3. Private information
- Interaction with the external world
    1. Non natural language interfaces
    2. Physical world
    3. Environmental information (time, e.g.)

❖ Some of these items look familiar ? Recall the motivation for RAG

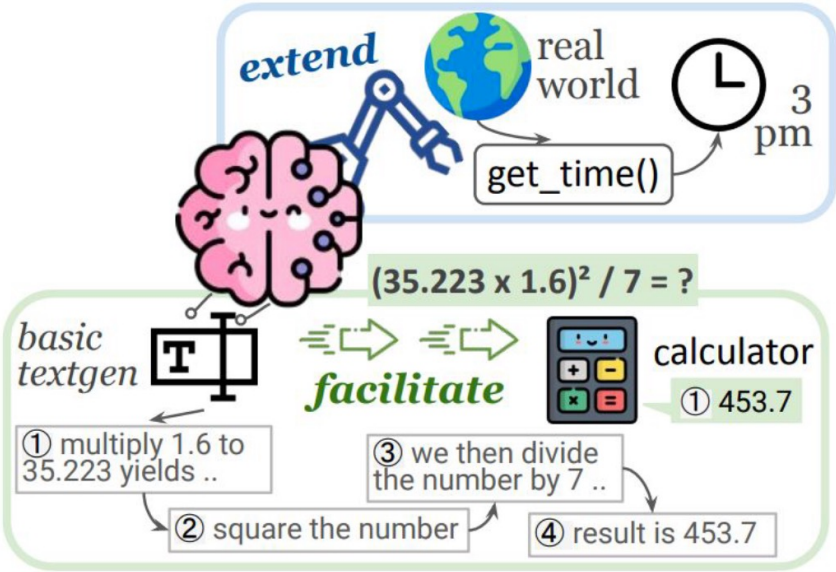
What is a Tool for LLM ?

Definition:

An LM-used tool is a function interface to a computer program that runs externally to the LM, where the LM generates the function calls and input arguments in order to use the tool [1]

A tool is:

- A Computer Program
- External to the LLM
- Used through generated function calls



LLM with Tools [1]

[1] Wang et al. 2024. What Are Tools Anyway? A Survey from the Language Model Perspective.

Tools benefit LLMs a lot !

Toolformer: Language Models Can Teach Themselves to Use Tools

Timo Schick Jane Dwivedi-Yu Roberto Dessì† Roberta Raileanu
Maria Lomeli Luke Zettlemoyer Nicola Cancedda Thomas Scialom

Meta AI Research †Universitat Pompeu Fabra

ART: Automatic multi-step reasoning and tool-use for large language models

Bhargavi Paranjape[1] Scott Lundberg[2] Sameer Singh[3] Hannaneh Hajishirzi[1,4]
Luke Zettlemoyer[1,5] Marco Tulio Ribeiro[2]
[1]University of Washington, [2]Microsoft Research, [3]University of California, Irvine,
[4]Allen Institute of Artificial Intelligence, [5]Meta AI

VOYAGER: An Open-Ended Embodied Agent with Large Language Models

Guanzhi Wang[1,2*], Yuqi Xie[3], Yunfan Jiang[4*], Ajay Mandlekar[1*],
Chaowei Xiao[1,5], Yuke Zhu[1,3], Linxi "Jim" Fan[1*], Anima Anandkumar[1,2†]
[1]NVIDIA, [2]Caltech, [3]UT Austin, [4]Stanford, [5]UW Madison
*Equal contribution †Equal advising Corresponding authors
https://voyager.minedojo.org

Gorilla: Large Language Model Connected with Massive APIs

Shishir G. Patil[1*] Tianjun Zhang[1,*] Xin Wang[2] Joseph E. Gonzalez[1]
[1]UC Berkeley [2]Microsoft Research
sgp@berkeley.edu

TOOLLLM: FACILITATING LARGE LANGUAGE MODELS TO MASTER 16000+ REAL-WORLD APIS

Yujia Qin[1*], Shihao Liang[1*], Yining Ye[1], Kunlun Zhu[1], Lan Yan[1], Yaxi Lu[1], Yankai Lin[3†],
Xin Cong[1], Xiangru Tang[4], Bill Qian[4], Sihan Zhao[1], Lauren Hong[1], Runchu Tian[1],
Ruobing Xie[5], Jie Zhou[5], Mark Gerstein[4], Dahai Li[2,6], Zhiyuan Liu[1†], Maosong Sun[1†]
[1]Tsinghua University [2]ModelBest Inc. [3]Renmin University of China
[4]Yale University [5]WeChat AI, Tencent Inc. [6]Zhihu Inc.
yujiaqin16@gmail.com

On the Tool Manipulation Capability of Open-source Large Language Models

Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, Jian Zhang
SambaNova Systems, Inc.
Palo Alto, CA, USA
{qiantong.xu,jian.zhang}@sambanovasystems.com

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face

Yongliang Shen[1,2,*], Kaitao Song[2,*,†], Xu Tan[2],
Dongsheng Li[2], Weiming Lu[1,†], Yueting Zhuang[1,†]
Zhejiang University[1], Microsoft Research Asia[2]
{syl, luwm, yzhuang}@zju.edu.cn, {kaitaosong, xuta, dongli}@microsoft.com

https://github.com/microsoft/JARVIS

TROVE: Inducing Verifiable and Efficient Toolboxes for Solving Programmatic Tasks

Zhiruo Wang[1] Graham Neubig[1] Daniel Fried[1]

Common Tool Categories and Examples for LLM

Category	Example Tools

Knowledge access	`sql_executor(query: str) -> answer: any`
	`search_engine(query: str) -> document: str`
	`retriever(query: str) -> document: str`

Computation activities	`calculator(formula: str) -> value: int | float`
	`python_interpreter(program: str) -> result: any`
	`worksheet.insert_row(row: list, index: int) -> None`

Interaction w/ the world	`get_weather(city_name: str) -> weather: str`
	`get_location(ip: str) -> location: str`
	`calendar.fetch_events(date: str) -> events: list`
	`email.verify(address: str) -> result: bool`

Non-textual modalities	`cat_image.delete(image_id: str) -> None`
	`spotify.play_music(name: str) -> None`
	`visual_qa(query: str, image: Image) -> answer: str`

Special-skilled LMs	`QA(question: str) -> answer: str` `translation(text: str, language: str) -> text: str`

Source: Wang et al. 2024. What Are Tools Anyway? A Survey from the Language Model Perspective.

How to Enable LLMs to Use Tools ?

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

Steps for Sample Tool Usage by LLMs [2]:

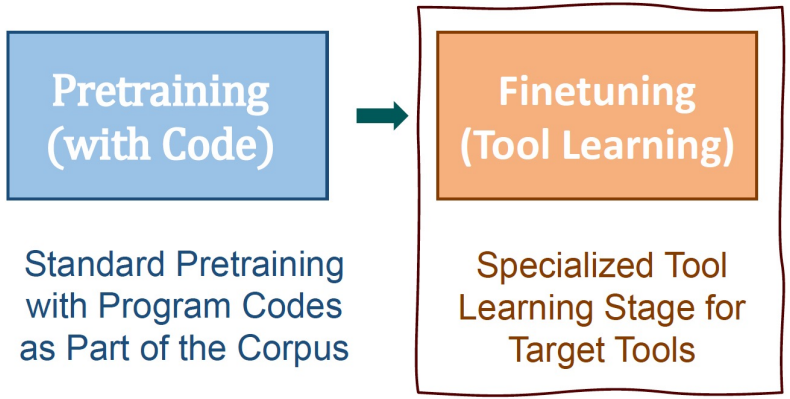
1. Function Calls predicted by LLMs
2. Tool execute that Function Call
3. Returned results as part of the LLM Context

=> LLMs need to LEARN:

1. When to use Tools
2. Which Tool to use
3. How to Incorporate Tools' results

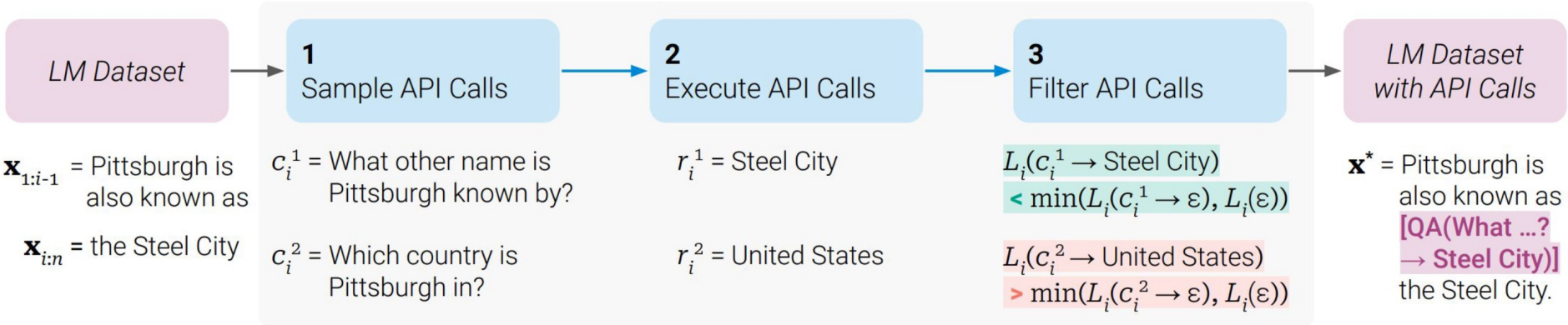
How to teach LLM to use tools ?

- Interference-time Prompting or
- Pretraining



[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Make LLMs Tool Users



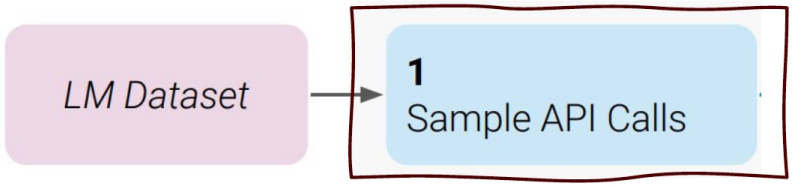
Key Idea (from Toolformer [2]):

Curate a corpus with tool calls and finetune LLMs with it

- Prompts an LLM to produce tool calls and execute them
- Filter the tool calls based on their effectiveness

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Make LLMs Tool Users: Sample API Calls



Write customized prompts to ask the LLM itself to generate potential API calls for each tool:

- Prompts it to generate APIs calls for positions likely to need API calls
  - (e.g. digits for calculator)
- Keep Top-K positions in a sequence with highest probability API calls above a given threshold
- Keep m candidate API calls for each of the chosen position

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of API calls:

Input: Joe Biden was born in Scranton, Pennsylvania.

Output: Joe Biden was born in [QA("Where was Joe Biden born?")] Scranton, [QA("In which state is Scranton?")] Pennsylvania.

Input: Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

Output: Coca-Cola, or [QA("What other name is Coca-Cola known by?")] Coke, is a carbonated soft drink manufactured by [QA("Who manufactures Coca-Cola?")] the Coca-Cola Company.

Input: x

Output:

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Make LLMs Tool Users: Execute API Calls

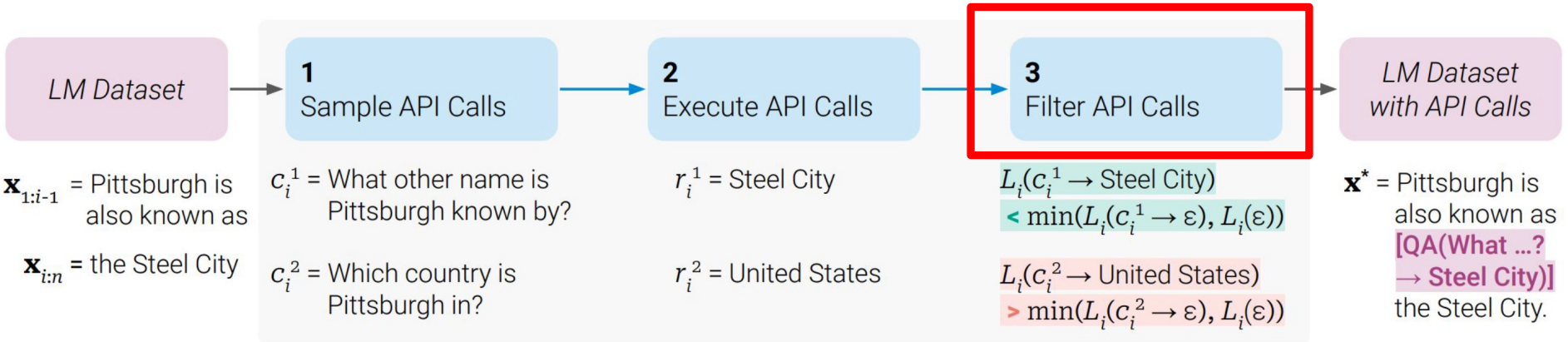


- Perform the tools for the sampled API calls
- Add tool output to the text sequence

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Make LLMs Tool Users: Filter Un-useful Tool Calls



What is a useful API call ?

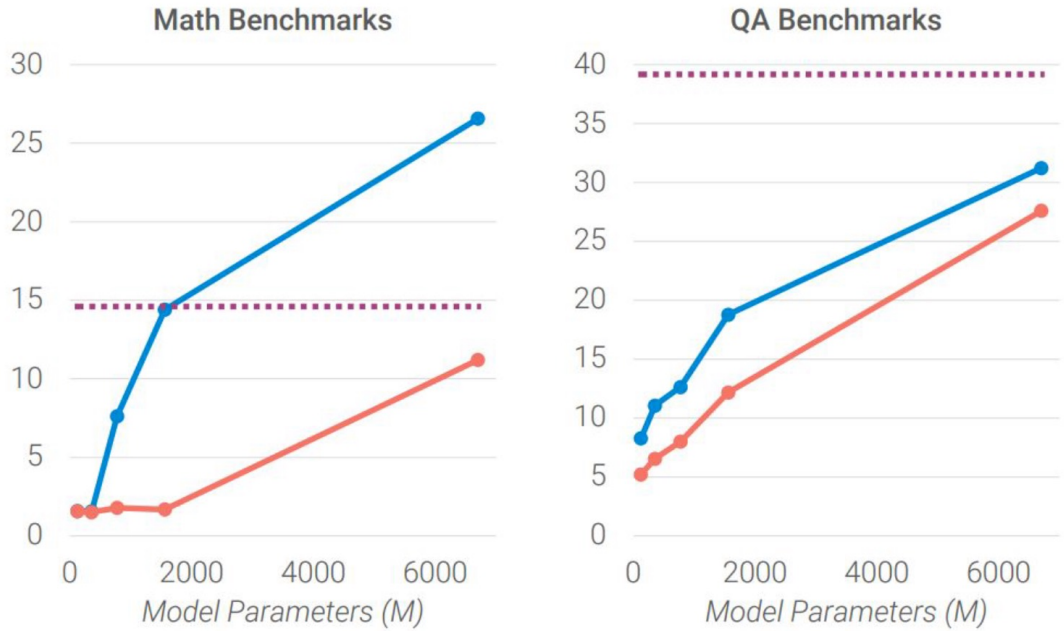
- LM performance in the rest tokens improved with the API call vs. without
- Filter out not useful ones to remove noise/ failed tool uses

$$L_i(c_i^2 \rightarrow \text{United States}) > \min(L_i(c_i^2 \rightarrow \varepsilon), L_i(\varepsilon))$$

Performance with API call + output — Performance with API call — Performance without API call

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Performance gain due to Tool Use



Significantly Improving GPT's Performances [2]

- Using at Max 25k examples per API

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

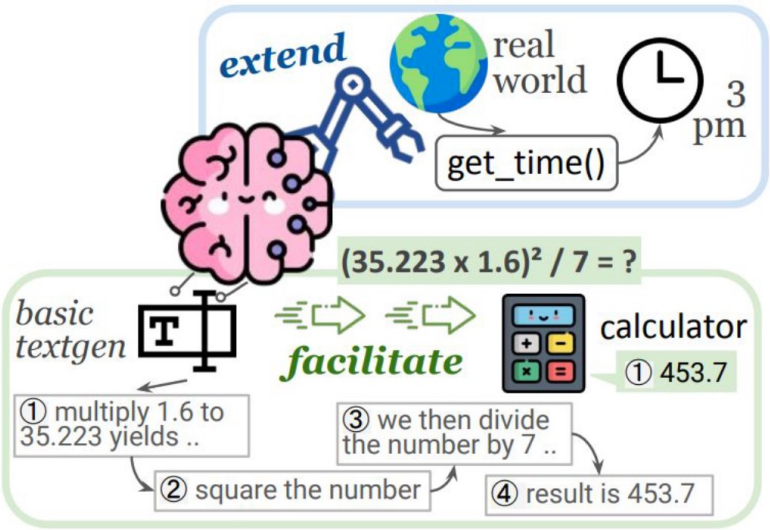
Example Tool Calls

Example	$L_i^- - L_i^+$	Useful
Os Melhores Escolas em Jersey 2020 <API> **MT(Os Melhores Escolas em Jersey) → The Best Schools in Jersey** </API> On this page you can search for Universities, Colleges and Business schools in Jersey	0.70	✓
Enjoy these pictures from the <API> **Calendar() → Today is Friday, April 19, 2013.** </API> Easter Egg Hunt.	0.33	✓
85 patients (23%) were hospitalised alive and admitted to a hospital ward. Of them, <API> **Calculator(85 / 23) → 3.70** </API> 65% had a cardiac aetiology [. . .]	−0.02	✗
But hey, after the <API> **Calendar() → Today is Saturday, June 25, 2011.** </API> Disneyland fiasco with the fire drill, I think it's safe to say Chewey won't let anyone die in a fire.	−0.41	✗
The last time I was with <API> **QA(Who was last time I was with?) → The Last Time** </API> him I asked what he likes about me and he said he would tell me one day.	−1.23	✗

Examples of kept and filtered API Calls [2]

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Summary of Tool Use



Why LLM needs tools and what are they

- To aid LLMs on tasks beyond their ability
- Mainly tools for knowledge, symbolic, and external environment operations

How to make LLMs effective tool users

- Existing LLMs with coding ability can be prompted to generate noisy tool calls
- Leverage the noisy tool call ability to curate tool use data
- Finetune LLMs on the tool use data to enhance its abilities

[2] Snihck et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools.

Retrieval-Augmented LMs (RALMs) have been widely deployed

Retrieval-augmented LMs have been widely used both in academia and industry



cohere

Gemini



perplexity

How RAG-Powered AI Applications Have A Positive Impact On Businesses

Forbes Technology Council

Jyotishko Biswas Forbes Councils Member

Forbes Technology Council COUNCIL POST | Membership (Fee-Based)



BAIR

BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

Subscribe About Archive BAIR

The Shift from Models to Compound AI Systems

"60% of LLM applications use some form of retrieval-augmented generation (RAG)"

Example: Deep Research w/ LLMs via Tool-use

Question

As of Q4 2024, A mid-sized retailer, X operates 50 stores across the southwestern U.S. and aims to reduce overstocking costs by 10% while still maintaining at least 98% product availability for its top-selling items during the upcoming holiday quarter. how can X Retail forecast next quarter's optimal inventory levels per store to balance cost minimization against high service levels?

LLM Models

Tools

Search

Code

Reasoning...

I need to identify Relevant External Economic Indicators ***[Web-Search]***

Query: US Q4 2024 Inflation and CCI

Searching...

...As of Q4 2024, the United States inflation rate stands at 2.7%, the Consumer Confidence Index (CCI) is at 109 ....

Result: 2.7% inflation and 109 CCI

...

To model demand volatility amid inflation and shifting consumer sentiment, I need to search historical sales under relevant economic indicators. ***[Web-Search]****...*

Searching...

Then, we can use ARIMAX with external regressors to predict per-store sales ***[Code]*** *...*

Query: Use ARIMAX to predict Q4 per-store sales given...

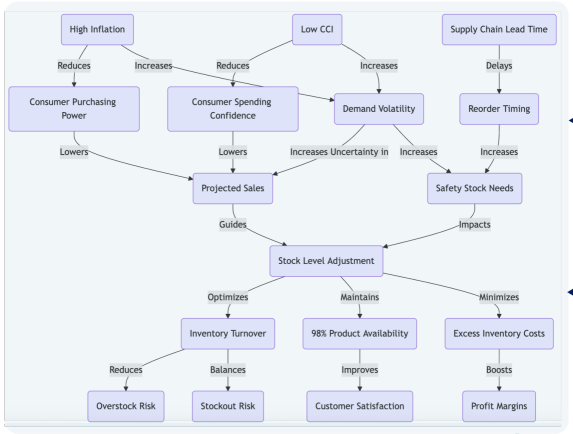
Result: The projected sales will be...

Coding...

```
...SARIMAX( sales,
exog=[cci, inflation],
order=(1,1,1), seasonal_orde
r=(0,1,1,4))
results = model.fit()
exog_future =
pd.DataFrame({"cci":
[115], "inflation": [3.5]},
index=[pd.Timestamp("2025-
03-31")])
Forecast = ...
```

...

So putting this all together, the strategies would be ***[Mind Map]***



Mind Map

Source: Junde Wu et al, Agentic Reasoning: Reasoning LLMs with Tools for the Deep Research, https://arxiv.org/pdf/2502.04644