Retrieval-Augmented Generation (RAG)

Presenters:
Yuyi Yang, MB, MPH, PhD student
Danni Beaulieu, MS, PhD student
Donghong Cai, PhD student

Presentation Date: 2024.09.24

WashU

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis, Ethan Perez et al.

Facebook AI Research; University College London; New York University

Published: 2020, at NeurIPS
Impact: Over 3400 citations

Why Retrieval-Augmented Generation (RAG)?

- **Challenge with ChatGPT:** Great for general responses, but what if you need a version with new, specialized, or updated information?
- **Pre-trained LLMs:** Store knowledge up to a point, but can't easily incorporate new data.
- **The Problem:** How can we dynamically update models with real-time or specific information like company documents or recent research?

How RAG Solves the Problem

You provide it with your own documents (in the form of urls, pdf files, docx files, txt files).

1. It stores these documents in a vector database.
2. You provide a query (i.e., a question)
3. The **Retriever** is tasked with retrieving contents from the vector database that is relevant to your query (question).
4. The retrieved information is fed into the **Generator** (*think of chatGPT as a generator*), and with this **Augmented** information
5. A response with respect to your provided documents are now generated

Hence the name: **Retrieval-Augmented Generation (RAG)**!

RAG Model Architecture

Two Key Components:

1. **Retriever**: Finds relevant documents (non-parametric memory).
2. **Generator**: Generates responses based on the retrieved documents

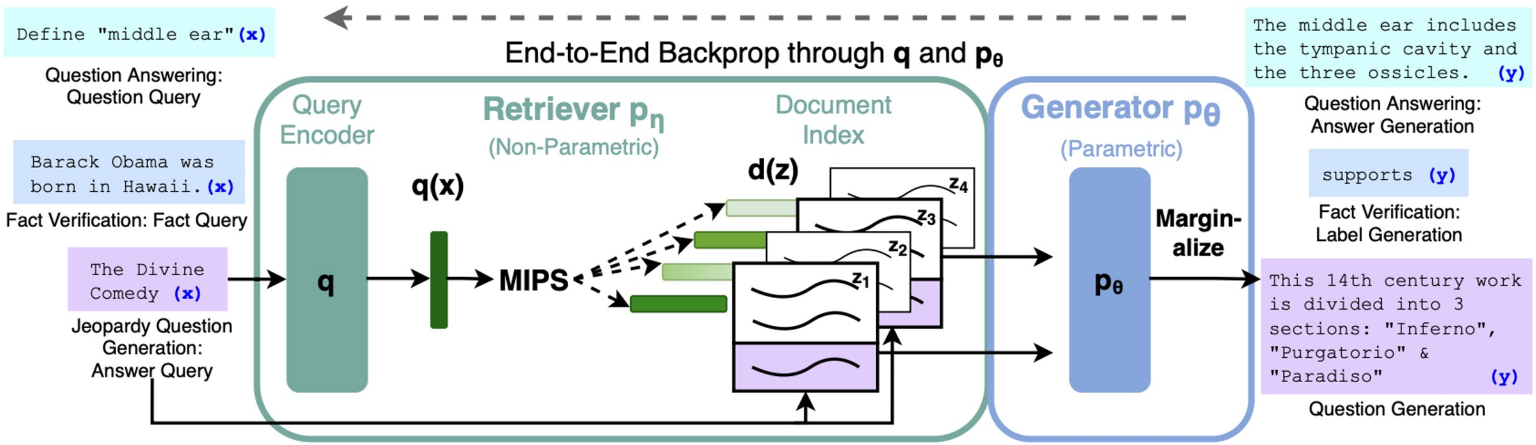


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

RAG Retriever: Dense Passage Retrieval (DPR)

DPR*:

- **Bi-Encoder Architecture**:
  - **Document Encoder** *d(z)*: Encodes documents using BERT.
  - **Query Encoder** *q(x)*: Encodes queries using BERT.
- Uses two encoders to create dense **embeddings**.
- Measures **similarity** using a dot product between the queries and documents embeddings.
- **Closer** embeddings indicate more **relevant** document pairs.

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

*Dense Passage Retrieval for Open-Domain Question Answering (Karpukhin et al., 2020)

RAG Model Architecture

Vector Matching: The retriever finds documents based on Maximum Inner Product Search (**MIPS***) with FAISS for efficient retrieval.

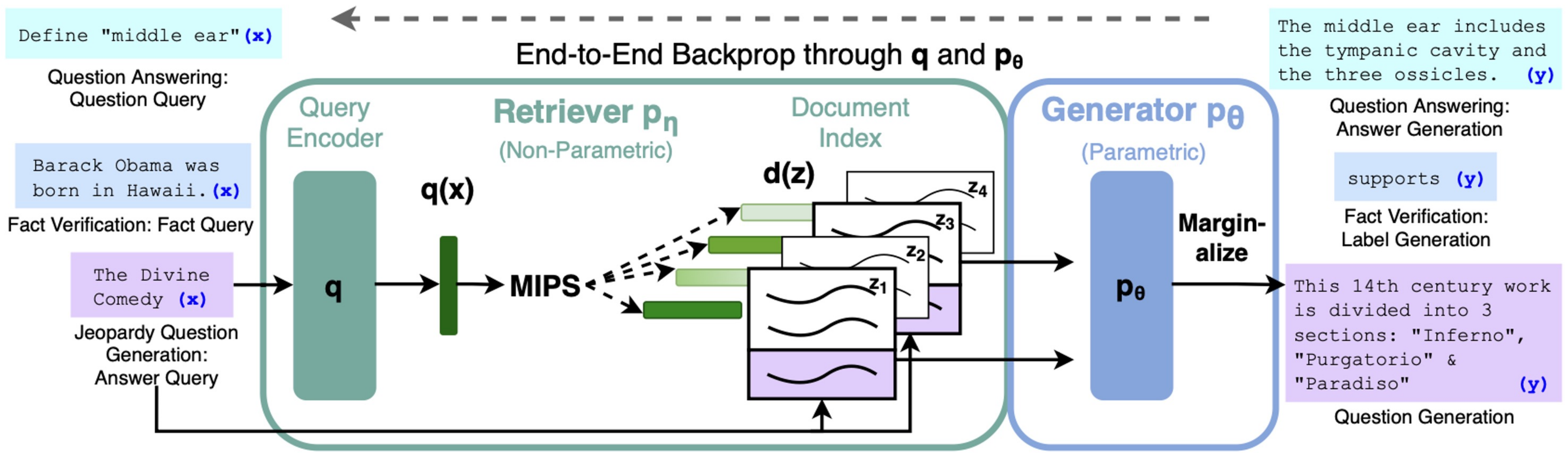


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

*Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (Shrivastava et al., 2014)

RAG Generator: BART

Generator Component: $p_\theta(y_i|x, z, y_{1:i-1})$

- **BART-Large**: A pre-trained sequence-to-sequence transformer with 400M parameters.
- **Pre-training Objective**: Denoising autoencoder with various noising functions.
- **Input and Retrieved Content**: Input *x* and retrieved content z are concatenated for generation.
- **BART as Parametric Memory**: Stores internal knowledge from pre-training, making it the parametric memory.

RAG-Sequence model and RAG-Token model

RAG-Sequence Model:

- Retrieves Top-K documents.
- Uses the same document for the whole output sequence.
- The document is treated as a latent variable for

ma

$$p_{\text{RAG-Sequence}}(y|x) \approx \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y|x, z) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) \prod_i^N p_\theta(y_i|x, z, y_{1:i-1})$$

RAG-Token Model:

- Retrieves Top-K documents.
- Uses different documents for each token.

$$p_{\text{RAG-Token}}(y|x) \approx \prod_i^N \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y_i|x, z, y_{1:i-1})$$ a new document.

Training the RAG Model

Joint Training:

- Retriever and Generator are trained jointly by minimizing **negative marginal log-lik** $\sum_j -\log p(y_j|x_j)$
- Fine-tune **Query Encoder (BERT_q)** and **BART Generator**.
- **Document Encoder (BERT_d)** remains fixed to reduce training cost.

Document Index: Pre-computed index for document retrieval.

Decoding in RAG Models

Decoding Methods:

- **Thorough Decoding:**
  - Extra passes for documents not in beam.
  - Combines probabilities across all documents.
- **Fast Decoding:**
  - Skips extra passes.
  - Faster for longer outputs.

Key Experimental Setup and Results

Wikipedia Dump:

- Used December 2018 Wikipedia dump.
- Split into 100-word chunks (21M documents).
- Documents stored in FAISS for fast retrieval.

Top K Documents:

- During training, top K documents (K = 5 or 10) were retrieved for each query.

Open-Domain Question Answering

RAG vs. REALM & T5+SSM:

- RAG achieves strong results without expensive pre-training.
- RAG sets a new state-of-the-art across QA tasks.

Table 1: Open-Domain QA Test Scores. For TQA, left column uses the standard test set for Open-Domain QA, right column uses the TQA-Wiki test set. See Appendix D for further details.

	Model	NQ	TQA	WQ	CT
Closed Book	T5-11B [52]	34.5	- /50.1	37.4	-
	T5-11B+SSM[52]	36.6	- /60.5	44.7	-
Open Book	REALM [20]	40.4	- / -	40.7	46.8
	DPR [26]	41.5	**57.9**/ -	41.1	50.6
	RAG-Token	44.1	55.2/66.1	**45.5**	50.0
	RAG-Seq.	**44.5**	56.8/**68.0**	45.2	**52.2**

Abstractive QA

- RAG-Sequence beats BART on MS-MARCO NLG by 2.6 Bleu points and 2.6 Rouge-L points.

Table 3: Examples from generation tasks. RAG models generate more specific and factually accurate responses. '?' indicates factually incorrect responses, * indicates partially correct responses.

Task	Input	Model	Generation
MS-MARCO	define middle ear	BART	?The middle ear is the part of the ear between the middle ear and the nose.
		RAG-T	The middle ear is the portion of the ear internal to the eardrum.
		RAG-S	The middle ear includes the tympanic cavity and the three ossicles.
	what currency needed in scotland	BART	The currency needed in Scotland is Pound sterling.
		RAG-T	Pound is the currency needed in Scotland.
		RAG-S	The currency needed in Scotland is the pound sterling.
Jeopardy Question Gener -ation	Washington	BART	?This state has the largest number of counties in the U.S.
		RAG-T	It's the only U.S. state named for a U.S. president
		RAG-S	It's the state where you'll find Mount Rainier National Park
	The Divine Comedy	BART	*This epic poem by Dante is divided into 3 parts: the Inferno, the Purgatorio & the Purgatorio
		RAG-T	Dante's "Inferno" is the first part of this epic poem
		RAG-S	This 14th century work is divided into 3 sections: "Inferno", "Purgatorio" & "Paradiso"

Jeopardy Generation

- RAG-Token outperforms RAG-Sequence and BART on Q-BLEU-1.
- Evaluators rated RAG more factual than BART in 42.7% of cases.

Table 2: Generation and classification Test Scores. MS-MARCO SotA is [4], FEVER-3 is [68] and FEVER-2 is [57] *Uses gold context/evidence. Best model without gold access underlined.

Model	Jeopardy B-1	Jeopardy QB-1	MSMARCO R-L	MSMARCO B-1	FVR3 Label Acc.	FVR2 Label Acc.
SotA	-	-	**49.8***	**49.9***	**76.8**	**92.2***
BART	15.1	19.7	38.2	41.6	64.0	81.1
RAG-Tok.	**17.3**	**22.2**	40.1	41.5	72.5	<u>89.5</u>
RAG-Seq.	14.7	21.4	<u>40.8</u>	<u>44.2</u>		

Table 4: Human assessments for the Jeopardy Question Generation Task.

	Factuality	Specificity
BART better	7.1%	16.8%
RAG better	**42.7%**	**37.4%**
Both good	11.7%	11.8%
Both poor	17.7%	6.9%
No majority	20.8%	20.1%

Fact Verification and Classification

- RAG scores within 4.3% of state-of-the-art for 3-way classification.

Table 2: Generation and classification Test Scores. MS-MARCO SotA is [4], FEVER-3 is [68] and FEVER-2 is [57] *Uses gold context/evidence. Best model without gold access underlined.

Model	Jeopardy B-1	Jeopardy QB-1	MSMARCO R-L	MSMARCO B-1	FVR3 Label	FVR2 Acc.
SotA	-	-	**49.8***	**49.9***	**76.8**	**92.2***
BART	15.1	19.7	38.2	41.6	64.0	81.1
RAG-Tok.	**17.3**	**22.2**	40.1	41.5	72.5	89.5
RAG-Seq.	14.7	21.4	40.8	44.2		

Further Study – Generation Diversity

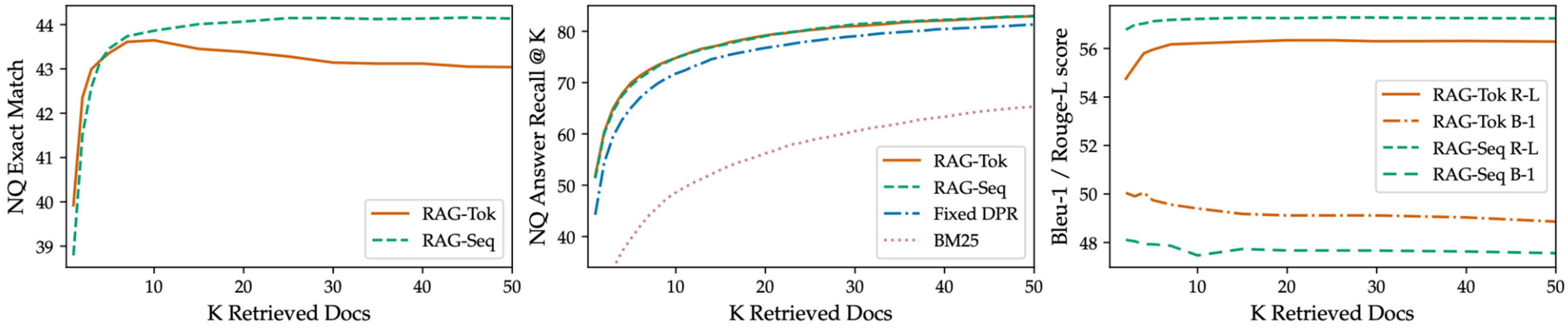
- RAG-Sequence produces more diverse generations than RAG-Token and BART.

Table 5: Ratio of distinct to total tri-grams for generation tasks.

	MSMARCO	Jeopardy QGen
Gold	89.6%	90.0%
BART	70.7%	32.4%
RAG-Token	77.8%	46.8%
RAG-Seq.	83.5%	53.8%

Further Study – Document Retrieval and Performance

- More documents lead to better results in Open-domain QA for RAG-Sequence.
- RAG-Token performance peaks at 10 documents.
- Retrieving more documents improves Rouge-L but reduces Bleu-1 for RAG-Token.



Discussion

Strengths of RAG:

- Built on reliable knowledge sources (like Wikipedia), reducing hallucination and improving accuracy.
- Allows for greater control over the output by using specific documents.
- Applicable to various fields like healthcare, education, and customer service.

Limitations:

- External sources (e.g., Wikipedia) may not always be completely accurate or unbiased.
- There's potential for misuse in creating harmful or misleading

Investigating the Factual Knowledge Boundary of Large Language Models with Retrieval Augmentation

Ren et al.

Gaoling School of Artificial Intelligence, Renmin University of China
Baidu Inc.
Beijing Key Laboratory of Big Data Management and Analysis Methods

Published: July 2023

Investigating the Factual Knowledge Boundary of Large Language Models with Retrieval Augmentation

Can LLMs detect their own knowledge boundaries?

Does Retrieval Augmentation change the boundary or detection?

Does document structure affect Retrieval Augmentation?

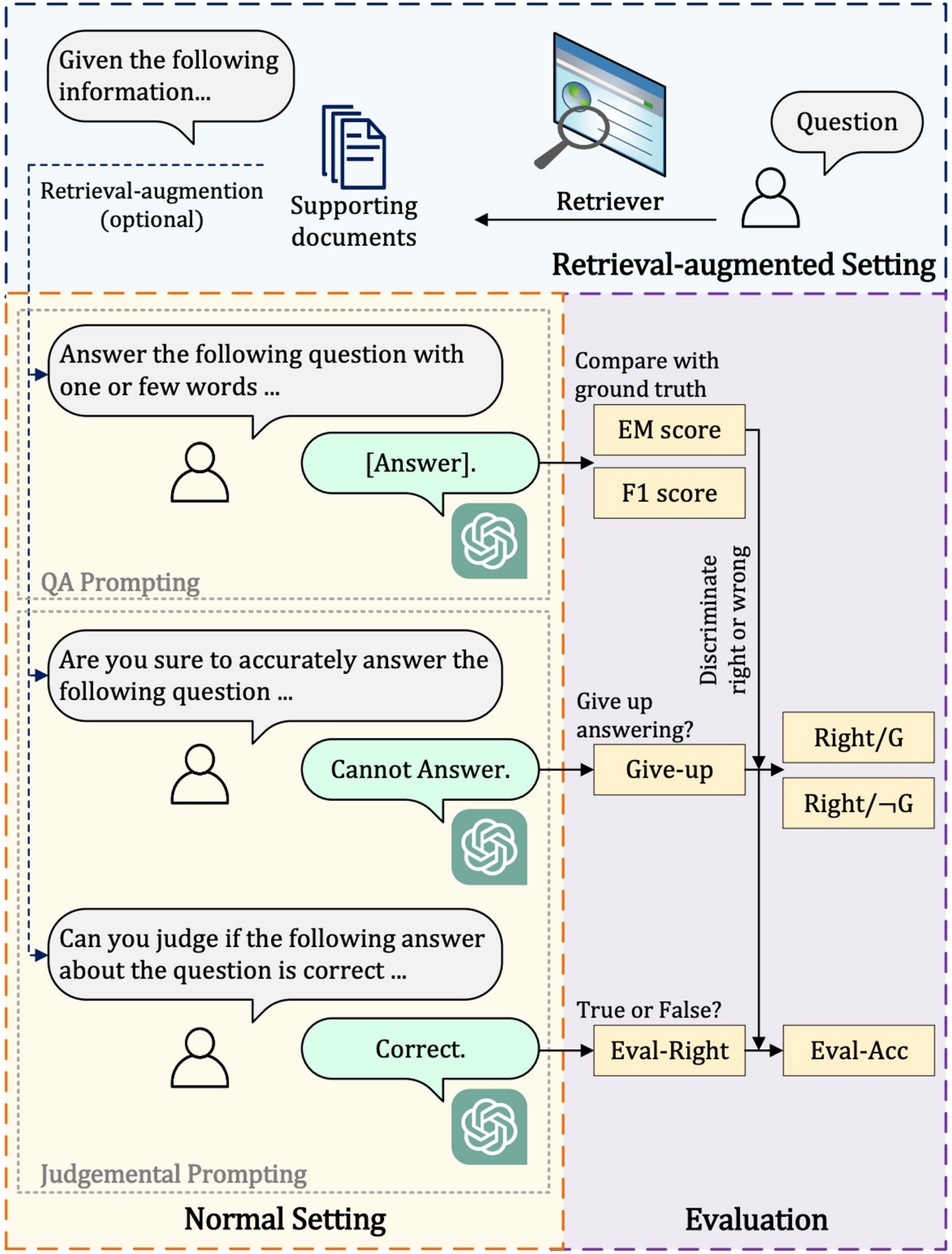
Task Setup

QA Prompting

- *With & without documents*

Judgmental Prompting

- *Priori (before answer):*
  - *Accuracy possible?*
  - *LLM "Give-up"*
  - *Right/G vs Right/¬G*
- *Posterior (after answer)*
  - *Response correct?*
  - *LLM "Eval-Right"*
  - *Eval-Acc*



Source Setup

LLMs

- *text-davinci-003 and gpt-3.5-turbo*

Knowledge QA datasets

- *Natural Questions, TriviaQA, Hotpot QA*

Retrievers

- *RocketQAv2 with Faiss (Dense), BM25 (Sparse), ChatGPT*

10 documents (Wiki or ChatGPT)

- *“Passage-{num}: Title: {title} Content: {content}”*
- *“Passage-{num}: {content}”*

Summary of Results

LLMs bad at knowledge boundaries

- *When to give up and correctness*

RAG helps except for ChatGPT generator with TriviaQA

- *Also organizes internal knowledge for ChatGPT retriever*

Dynamically using RAG based on prior “give up” helps again!

Improvement of more documents levels out at **5-10**

All categories of QA improve except for ChatGPT “which” & “declare”

- *Already LLM strong suit?*

“Good” vs “Bad” documents matter & improve or degrade performance

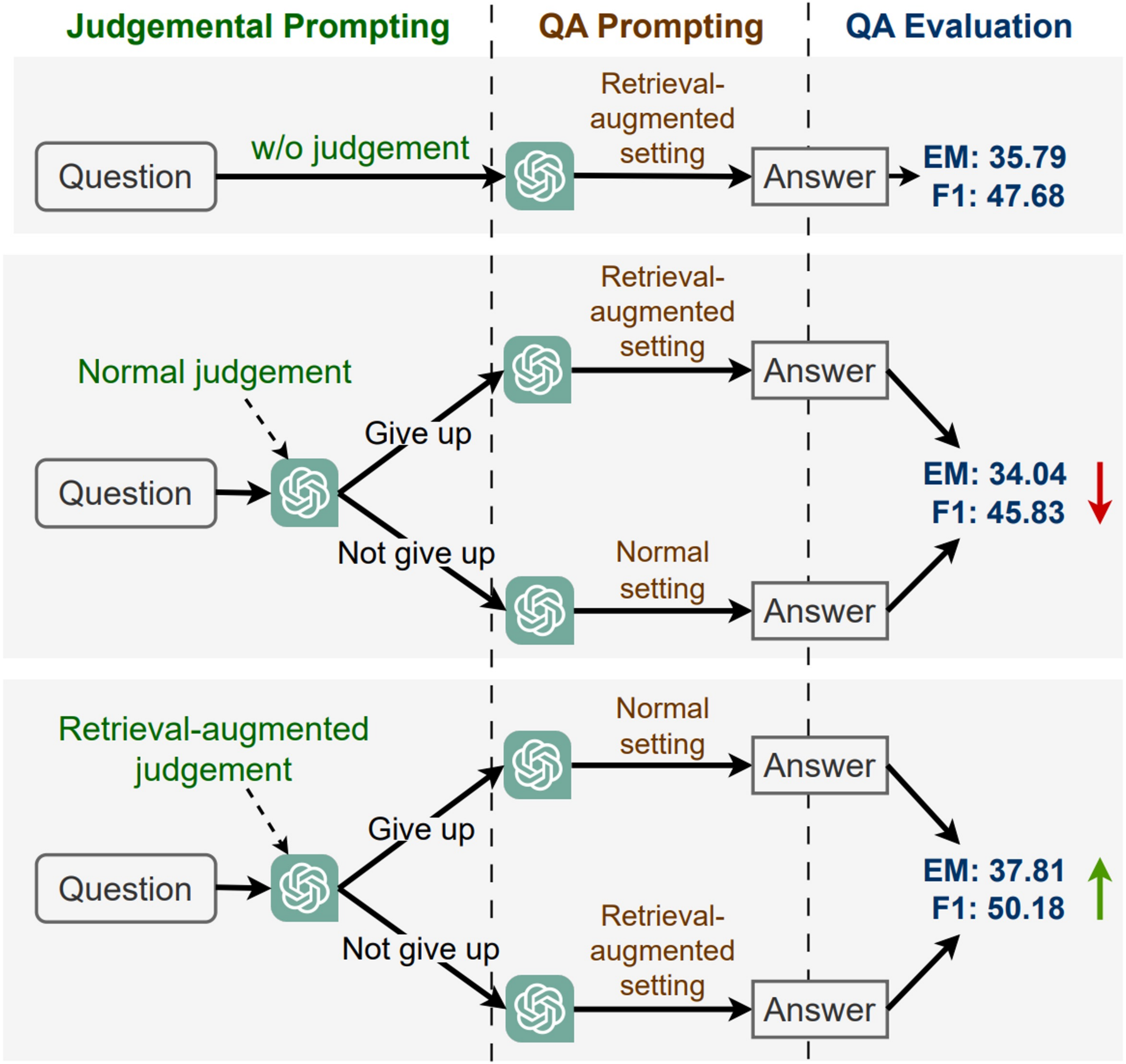
Dataset	LLM	Retrieval Source	QA		Priori Judgement			Posteriori Judgement	
			EM	F1	Give-up	Right/G	Right/¬G	Eval-Right	Eval-Acc
NQ	Davinci003	None	26.37	35.95	27.17%	13.56%	31.15%	71.27%	46.88%
		Sparse	30.44	40.90	20.55%	9.84%	35.77%	41.11%	67.56%
		Dense	**40.58**	52.22	14.52%	14.31%	45.04%	47.78%	69.67%
		Dense+Sparse	40.50	**52.33**	8.92%	12.73%	43.22%	47.37%	69.84%
		ChatGPT	34.18	46.79	6.73%	5.35%	36.26%	44.96%	72.11%
	ChatGPT	None	30.89	42.14	32.05%	14.63%	38.67%	87.09%	36.85%
		Sparse	25.87	35.71	41.41%	8.03%	38.49%	57.76%	52.26%
		Dense	35.79	47.68	27.53%	11.27%	45.11%	63.35%	55.03%
		Dense+Sparse	**36.01**	**47.99**	26.90%	11.33%	45.09%	70.94%	47.54%
		ChatGPT	32.80	45.08	8.34%	5.98%	35.24%	70.94%	47.54%
TriviaQA	Davinci003	None	69.56	74.03	5.65%	36.59%	71.53%	87.90%	72.05%
		Sparse	70.16	75.73	11.37%	28.47%	75.51%	73.45%	78.81%
		Dense	72.59	78.30	8.59%	31.24%	76.48%	77.35%	80.84%
		Dense+Sparse	**72.60**	78.60	6.77%	28.84%	75.78%	76.83%	81.67%
		ChatGPT	71.92	**78.97**	1.88%	19.18%	72.93%	78.24%	83.62%
	ChatGPT	None	**74.77**	**80.11**	12.00%	44.00%	78.97%	92.58%	77.02%
		Sparse	65.31	71.81	19.00%	21.91%	75.48%	84.86%	78.58%
		Dense	69.84	76.58	15.67%	30.25%	77.20%	87.81%	78.90%
		Dense+Sparse	70.10	76.91	13.40%	28.76%	76.49%	88.43%	79.33%
		ChatGPT	69.53	77.67	3.03%	16.53%	71.19%	92.23%	78.84%
HotpotQA	Davinci003	None	16.62	25.53	35.76%	8.34%	21.23%	69.87%	41.93%
		Sparse	28.27	39.65	29.40%	11.18%	35.38%	32.47%	75.46%
		Dense	25.13	35.74	37.60%	10.27%	34.08%	33.94%	74.24%
		Dense+Sparse	**29.40**	**41.02**	25.27%	11.07%	35.60%	33.88%	75.18%
		ChatGPT	25.47	36.93	8.64%	4.31%	27.47%	33.66%	76.15%
	ChatGPT	None	17.81	26.35	66.29%	9.76%	33.63%	55.16%	33.13%
		Sparse	24.52	34.64	54.89%	9.08%	43.31%	47.47%	45.73%
		Dense	21.08	30.12	63.07%	8.33%	42.86%	44.76%	46.69%
		Dense+Sparse	**25.67**	35.76	54.02%	9.72%	44.42%	48.50%	45.37%
		ChatGPT	24.45	**36.60**	12.83%	4.89%	27.33%	63.63%	47.48%

Judgments improve:

- Right & Give-up ↓
- Right & not Give-up ↑
- Eval-Right
  - Closer to Eval-Acc

RAG helps except for with ChatGPT TriviaQA

- Higher EM & F1
- Better Eval-Acc

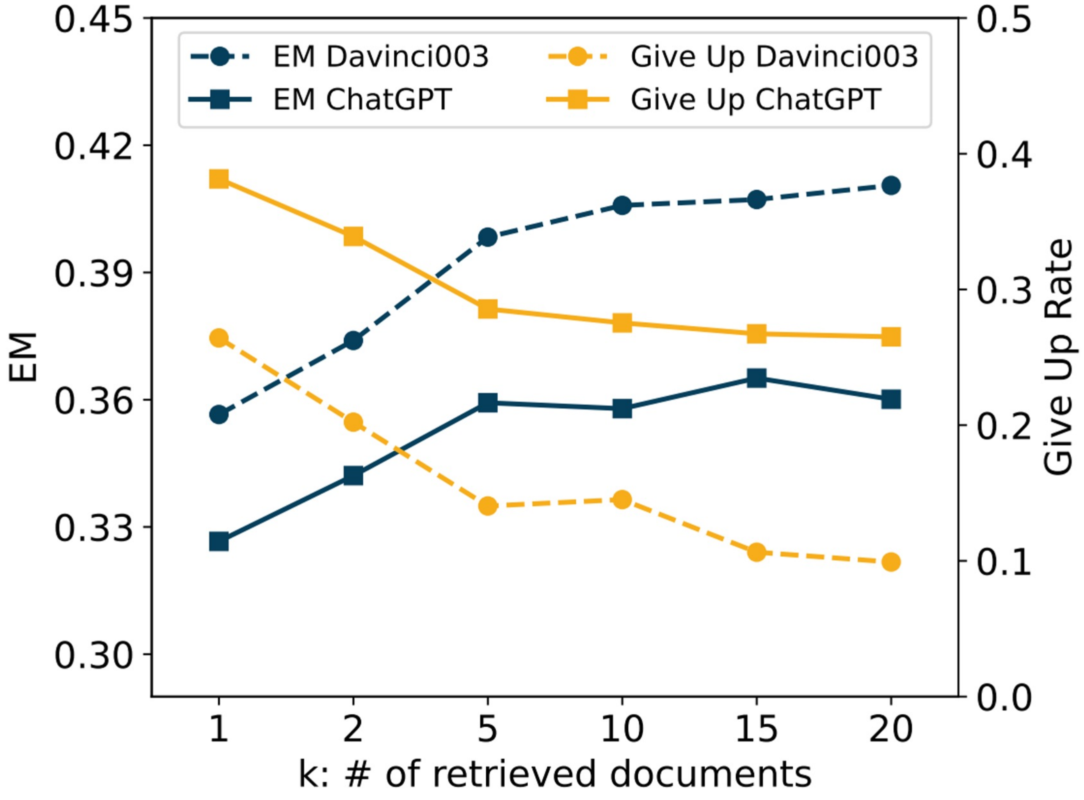


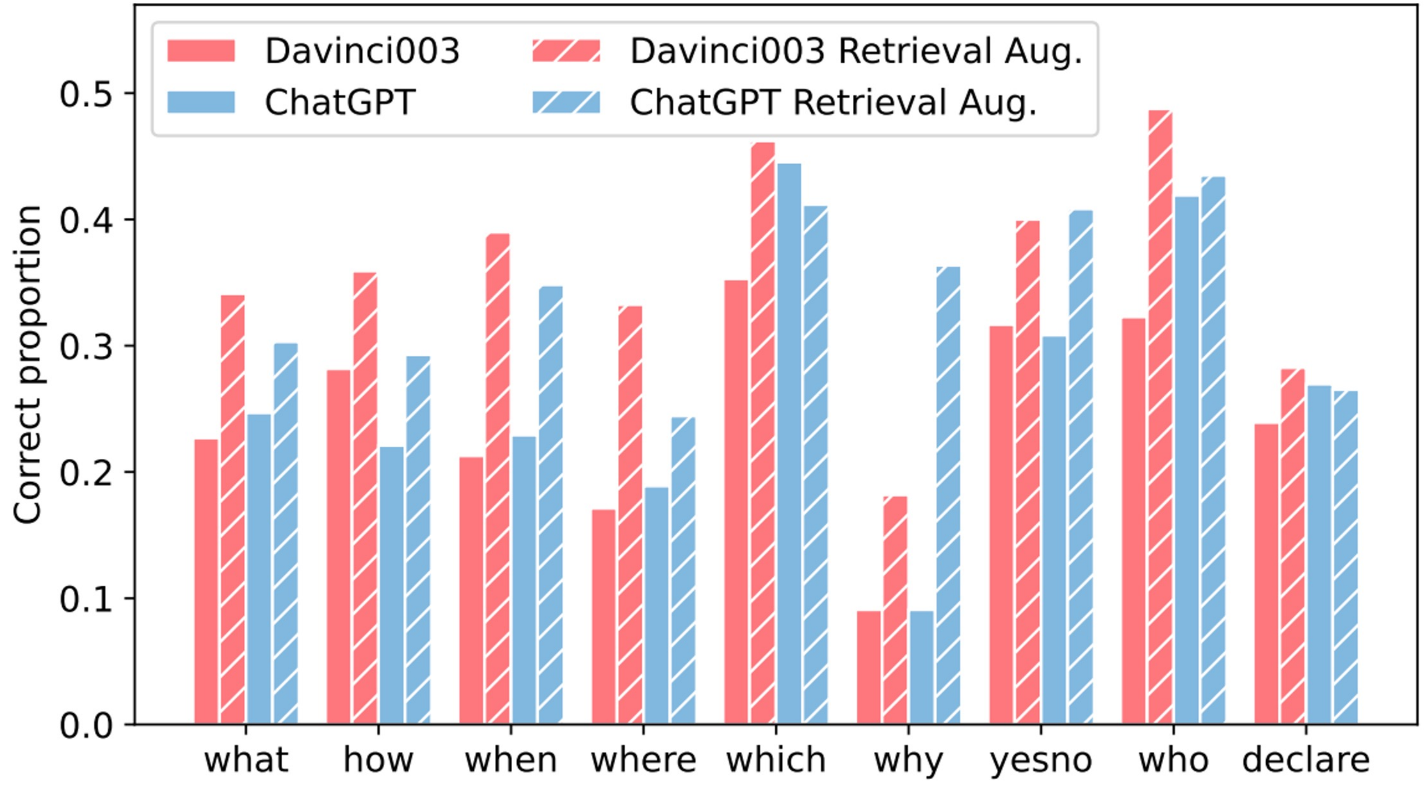
Dynamically using RAG to answer based on “give up” assessment prior

- ➢ RAG judgment helps!
- ➢ Normal degrades

Improvement of more documents levels out at 5-10

- ➢ Exact match goes up
- ➢ Give up rates go down
- ➢ Insensitive to order





All categories of QA improve except for ChatGPT “which” & “declare”

- ➢ “why” improves most
- ➢ “who” performs best

Supporting Doc	Davinci003					ChatGPT				
	EM	F1	Give-up	Eval-Right	Eval-Acc	EM	F1	Give-up	Eval-Right	Eval-Acc
None	26.37	35.95	27.17%	71.27%	46.88%	30.89	42.14	32.05%	87.09%	36.85%
Golden	**52.35**	**64.10**	14.96%	50.80%	71.09%	**45.93**	**58.82**	24.35%	67.26%	54.50%
Retrieved	40.58	52.22	14.52%	47.78%	69.67%	35.79	47.68	27.53%	63.35%	55.03%
Highly-related	11.66	21.76	20.06%	31.11%	58.21%	11.27	20.80	47.09%	51.00%	47.27%
Weakly-related	12.99	21.42	40.39%	24.76%	61.68%	9.42	15.83	66.40%	48.75%	46.20%
Random	23.93	32.62	87.89%	21.91%	67.12%	12.74	17.39	90.97%	49.89%	40.01%

"Good" vs "Bad" documents matter

- Golden: contain correct answers
  - *Top 100 sampled top to bottom*
- Highly-related: very relevant but no correct answers
  - *Top 100 sampled top to bottom*
- Weakly-related: somewhat relevant and no correct answers
  - *Top 100 sampled randomly excluding above*
- Random from entire corpus: not relevant and no correct answers

Conclusions

Can LLMs detect their own knowledge boundaries?

→ *Inaccurate & overconfident*

Does Retrieval Augmentation change the boundary or detection?

→ *Helps priori and posteriori judgments*

Does document structure affect Retrieval Augmentation?

→ *Relies on relevance & quality*

REPLUG: Retrieval-Augmented Black-Box Language Models

Shi et al.

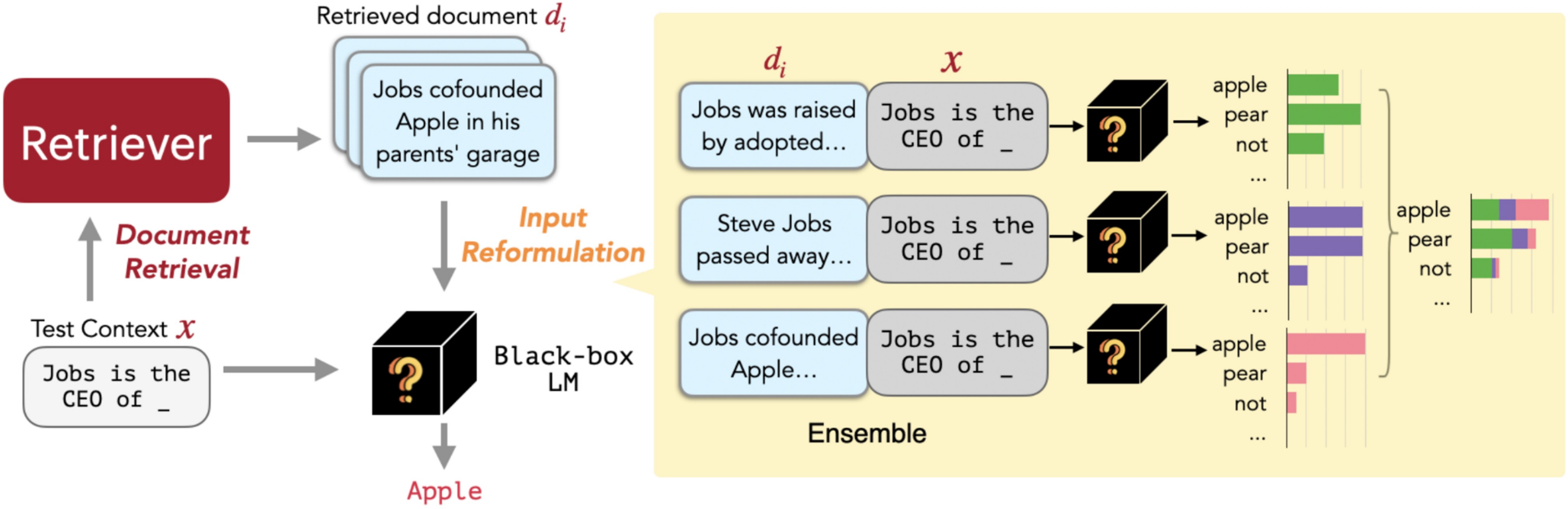
Published: May 2023

RePlug: Retrieval-Augmented Black-Box Language Models

Does RAG work when LLM is a black box?

- Retrieve & Plug (RePlug):
  - *Retrieval component is tunable "plug & play"*
- RePlug with LM-Supervised Retrieval (LSR):
  - *Adapt RePlug based on LLM feedback*
- Test across datasets!
  - *The Pile, MMLU, Open Domain QA*

RePlug Setup



Map each document & query to embedding (use top k by cosine similarity)

Prepend each document to query and choose token by ensemble output

RePlug Details

Dense retriever with dual encoder

- *Cosine similarity **s** for embedding **E***
- *Document **d** from corpus **D** & input **x***

$$s(d, x) = \cos(\mathbf{E}(d), \mathbf{E}(x))$$

$$p(y \mid x, \mathcal{D}') = \sum_{d \in \mathcal{D}'} p(y \mid d \circ x) \cdot \lambda(d, x)$$

Probability **p** of next token **y**

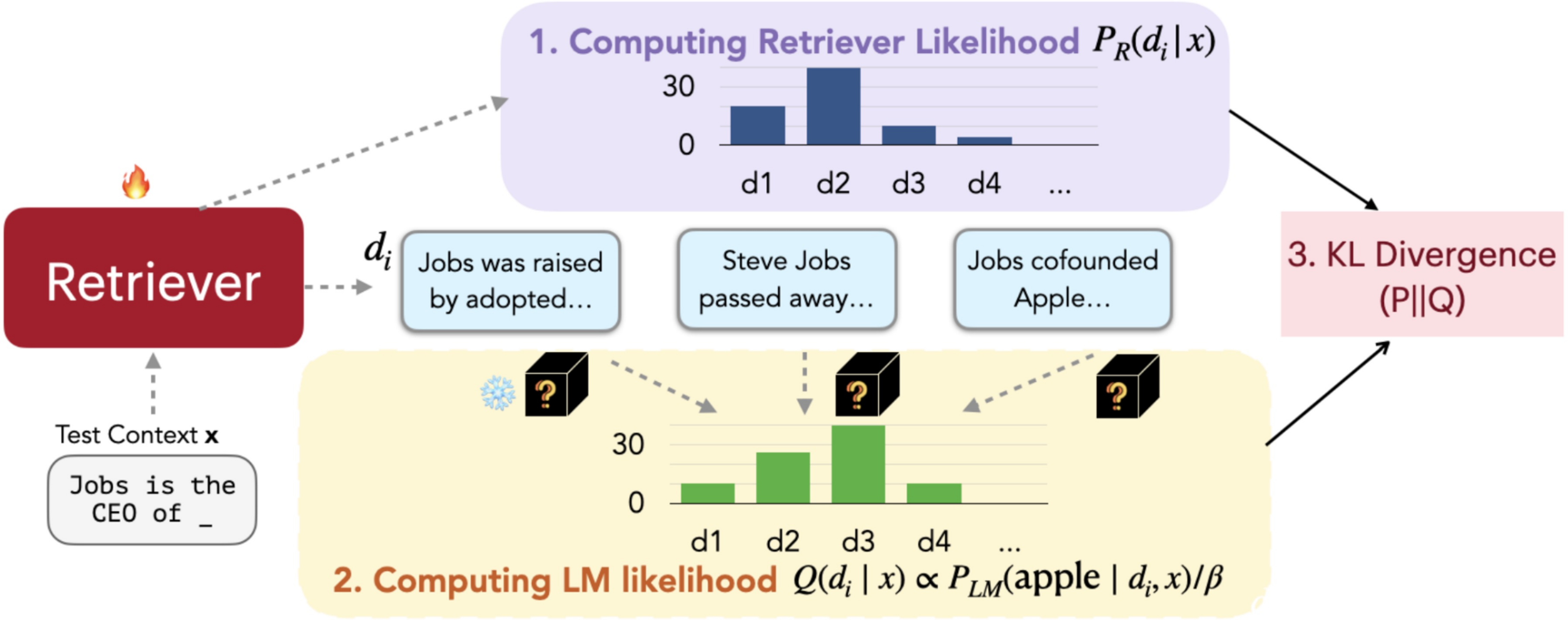
- *Top-k documents **D’** by **s(d,x)***
- *Concatenation of 2 sequences “∘”*
- *Weighted average ensemble*

Weight for ensemble **λ**

- *Reuse similarity score **s(d,x)***

$$\lambda(d, x) = \frac{e^{s(d,x)}}{\sum_{d \in \mathcal{D}'} e^{s(d,x)}}$$

RePlug LSR Setup



Loss of retrieval likelihood (marginalization) & LM likelihood (perplexity)
Recompute embeddings

RePlug LSR Details

Retrieval likelihood ***P***

- *Hyperparameter **γ** softmax temperature*
- *Marginalize over **d** in **D'***

$$P_R(d \mid x) = \frac{e^{s(d,x)/\gamma}}{\sum_{d \in \mathcal{D}'} e^{s(d,x)/\gamma}}$$

$$Q(d \mid x, y) = \frac{e^{P_{LM}(y|d,x)/\beta}}{\sum_{d \in \mathcal{D}'} e^{P_{LM}(y|d,x)/\beta}}$$

LM likelihood ***Q***

- *Perplexity with & without **d***
- *Token **y** more probable*
- *Hyperparameter **β***

Loss function ***L***

- *Minimize KL divergence*
- *Close Retrieval **P** & LM **Q***

$$\mathcal{L} = \frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} KL\Big(P_R(d \mid x) \parallel Q_{\text{LM}}(d \mid x, y)\Big)$$

Summary of Results

Better than random ensemble!

The Pile (GPT-2 & GPT-3): diverse domains (web pages, code, academics)

- ***RePlug LSR*** *(+7.7%) better than RePlug (+4.7%)*

MMLU (Codex): multiple choice QA across disciplines

- ***RePlug LSR*** *(+5.1%) better than RePlug (+4.5%)*

Open Domain NQ & TriviaQA (Codex): collected from Wiki & web

- ***RePlug LSR*** *(+12.0%) better than RePlug (+5.0%)*

Applicable to **diverse language models**

Rare entities benefit from retrieval

The Pile (GPT-2 & GPT-3): Diverse

RePlug LSR (+7.7%) better than RePlug (+4.7%)

Model		# Parameters	Original	+ REPLUG	Gain %	+ REPLUG LSR	Gain %
GPT-2	Small	117M	1.33	1.26	5.3	1.21	9.0
	Medium	345M	1.20	1.14	5.0	1.11	7.5
	Large	774M	1.19	1.15	3.4	1.09	8.4
	XL	1.5B	1.16	1.09	6.0	1.07	7.8
GPT-3 (black-box)	Ada	350M	1.05	0.98	6.7	0.96	8.6
	Babbage	1.3B	0.95	0.90	5.3	0.88	7.4
	Curie	6.7B	0.88	0.85	3.4	0.82	6.8
	Davinci	175B	0.80	0.77	3.8	0.75	6.3

MMLU (Codex): Multiple Choice

RePlug LSR (+5.1%) better than RePlug (+4.5%)

Model	# Parameters	Humanities	Social.	STEM	Other	All
Codex	175B	74.2	76.9	57.8	70.1	68.3
PaLM	540B	77.0	81.0	55.6	69.6	69.3
Flan-PaLM	540B	-	-	-	-	72.2
Atlas	11B	46.1	54.6	38.8	52.8	47.9
Codex + REPLUG	175B	76.0	79.7	58.8	72.1	71.4
Codex + REPLUG LSR	175B	76.5	79.9	58.9	73.2	71.8

Top MMLU LLMs

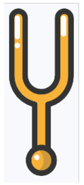
Tuned RAG

Open Domain NQ & TriviaQA (Codex): Wiki/Web

	NQ		TQA	
Model	Few-shot	Full	Few-shot	Full
Chinchilla	35.5	-	64.6	-
PaLM	39.6	-	-	-
Codex	40.6	-	73.6	-
RETRO†	-	45.5	-	-
R2-D2†	-	55.9	-	69.9
Atlas†	42.4	**60.4**	74.5	**79.8**
Codex + Contriever$_{cc}$[2]	44.2	-	76.0	-
Codex + REPLUG	44.7	-	76.8	-
Codex + REPLUG LSR	**45.5**	-	**77.3**	-

Powerful LLMs

Tuned RAG LMs



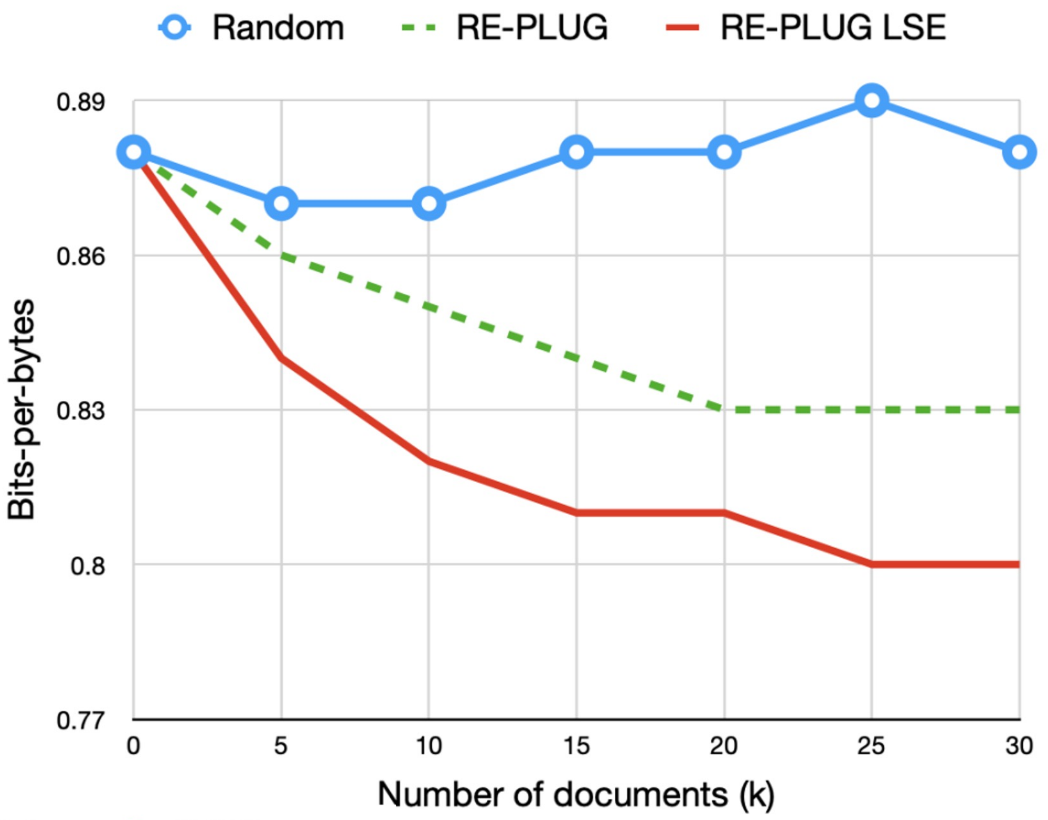
★ Lag behind LMs tuned on full data likely due to near-duplicate questions in training set

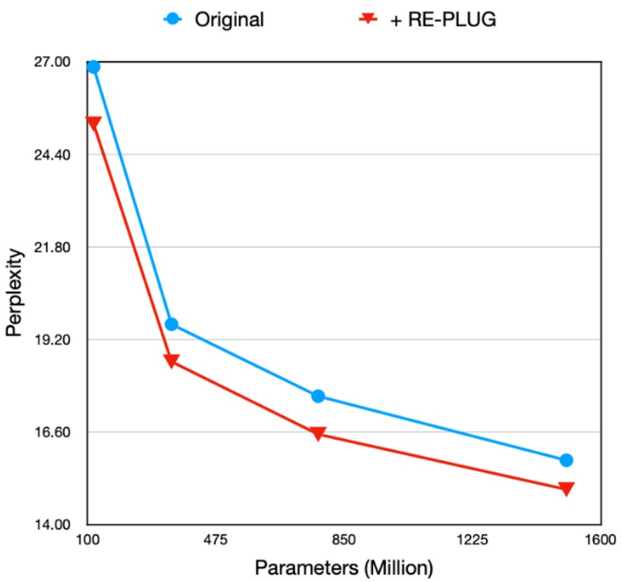
RePlug LSR (+12.0%) better than RePlug (+5.0%)

LLMs with varying sizes (WikiText):

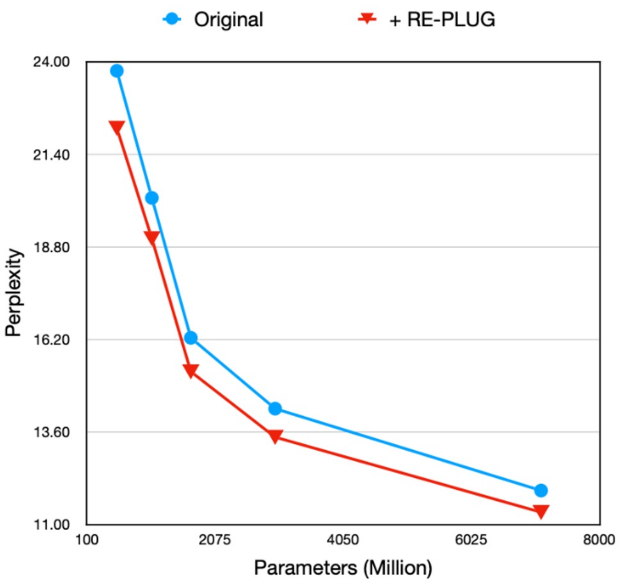
- ➢ GPT2
  - *117M, 345M, 774M, 1.5B*
- ➢ OPT
  - *125M, 350M, 1.3B, 2.7B, 6.7B, 13B, 30B, 66B*
- ➢ BLOOM
  - *560M, 1.1B, 1.7B, 3B and 7B*

Perplexity lower with RePlug

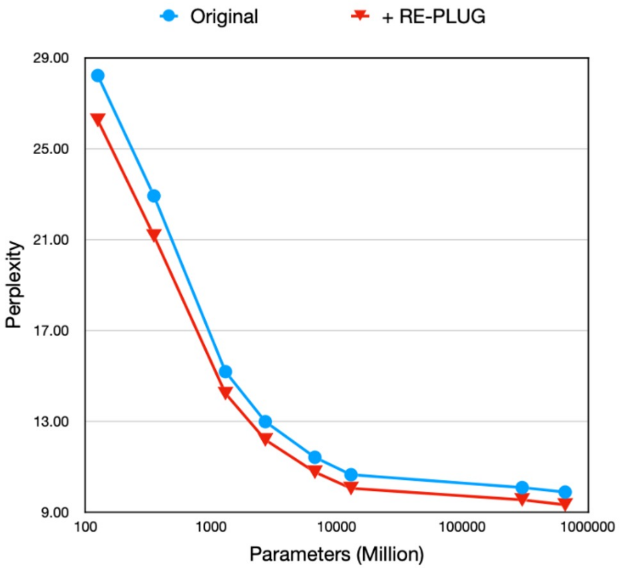




GPT-2



BLOOM



OPT

Better than random ensemble!

Conclusions

Does RAG work when LLM is a black box?

- ☑ Retrieve & Plug (RePlug)
- ☑ RePlug with LM-Supervised Retrieval (LSR)
- ☑ Across datasets (even STEM)!

Even the state-of-the-art large-scale LMs benefit from retrieval!

Limitation: lacks interpretability and knowledge boundary detection

- RAG vs internal knowledge usage

Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection

Akari Asai, Hannaneh Hajishirzi et al.

ICLR'24

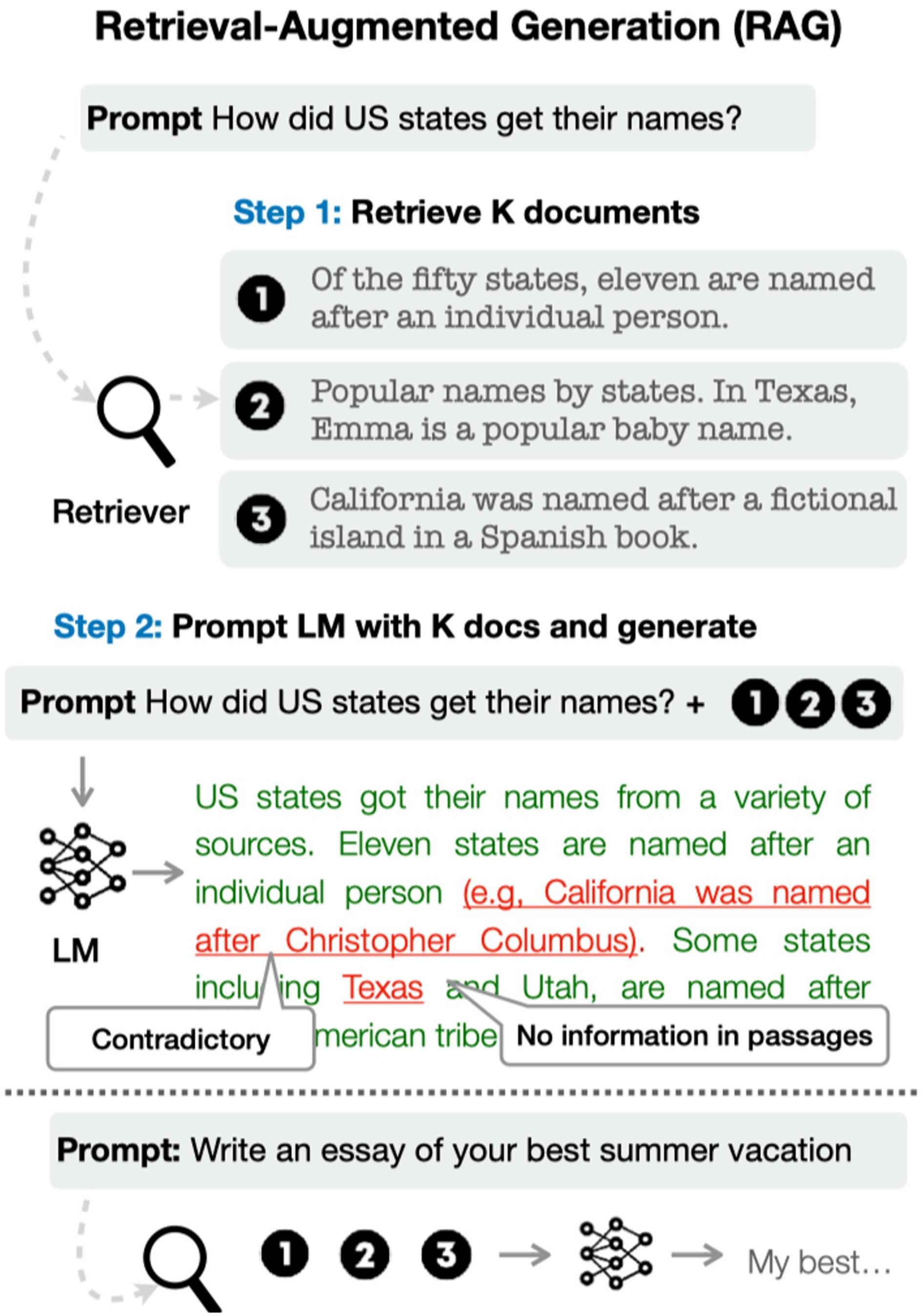
Background

- **Large Language Models**:
  - Successful, but hallucinations (factually incorrect or nonsensical)
- **Retrieval-Augmented Generation (RAG)**:
  - Augment the input of LLMs with relevant retrieved passages
  - Reduce factual errors in knowledge-intensive tasks, like QA

Background

- Problems:
  - Retrieving is not always necessary
  - The retrieved documents may not be relevant
  - No guarantee that generations are entailed by cited evidence

=> Performance can be even poorer



Self-RAG

- Think in 3 Steps:
  1. Do we need retrieval?
  2. Are the retrieved documents relevant?
  3. Given the input query and retrieved documents, is the generations of good quality?

- How?
  - ➢ Through **Reflection Tokens**
    - On-demand retrieval (retrieval token)
    - Self-reflection (critique token)

Reflection tokens

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

- x: input
- y: output
- d: relevant passage

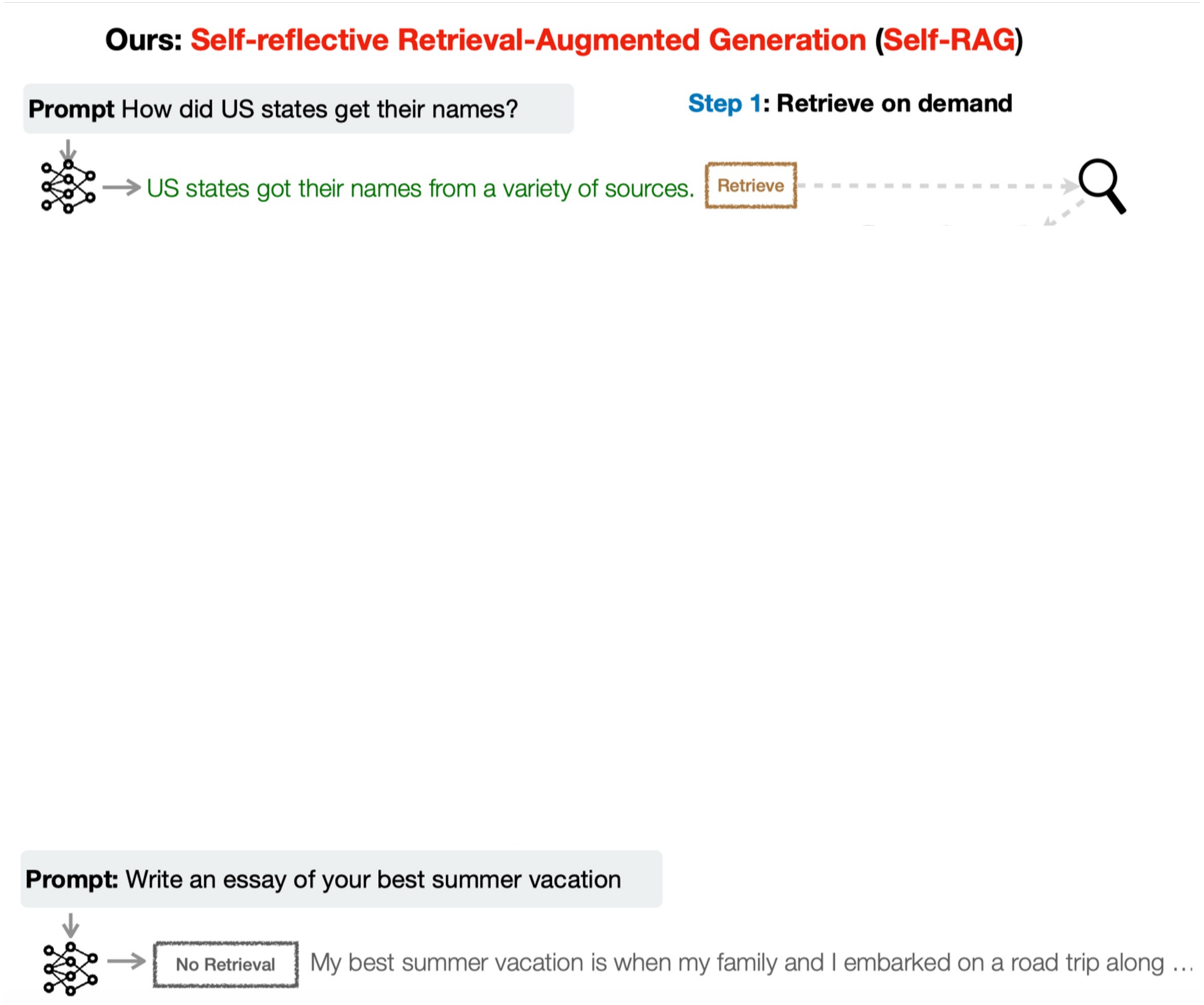
The bold text indicates the most desirable critique tokens

Reflection tokens

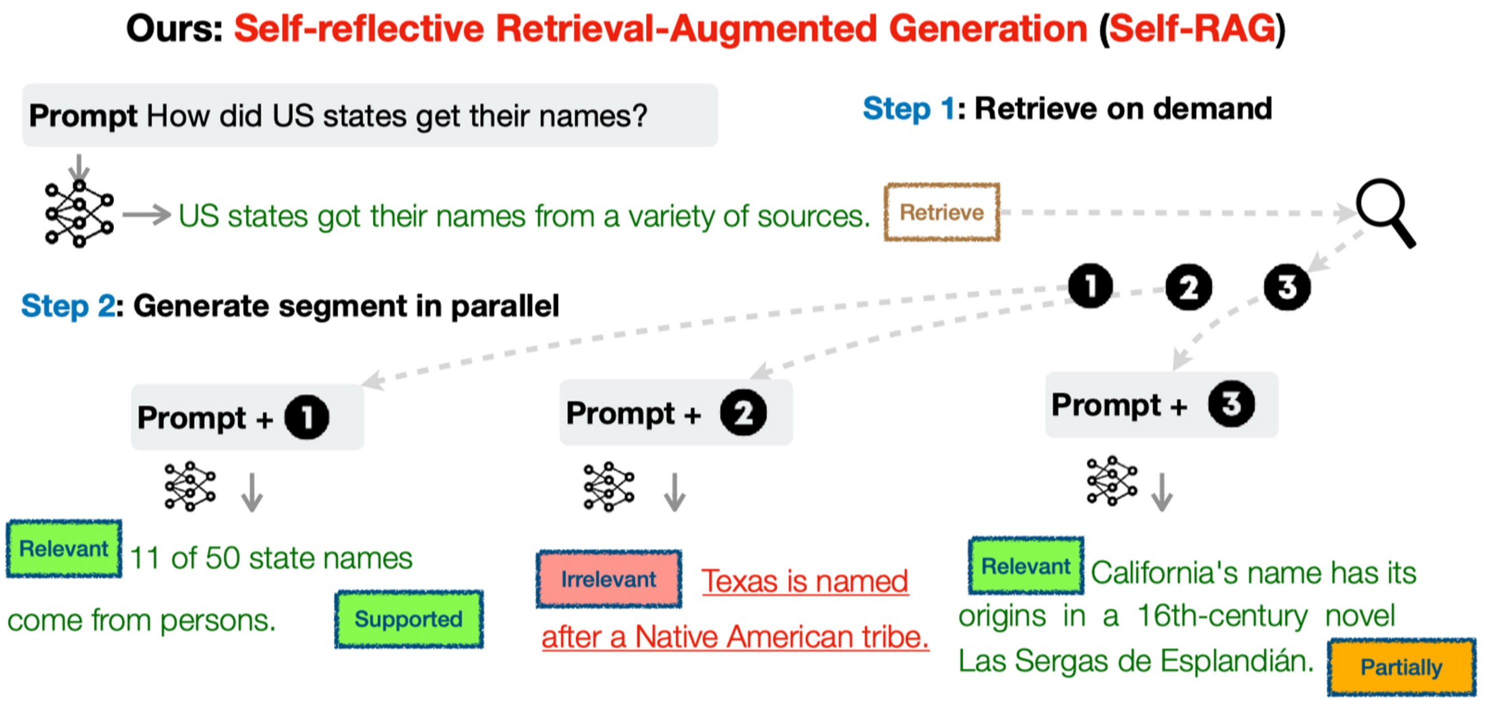
Type	Input	Output	Definitions
Retrieve	x / x, y	{yes, no, continue}	Decides when to retrieve with $\mathcal{R}$
IsRel	x, d	{**relevant**, irrelevant}	d provides useful information to solve x.
IsSup	x, d, y	{**fully supported**, partially supported, no support}	All of the verification-worthy statement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

Totally 13 new tokens added to the original vocabulary

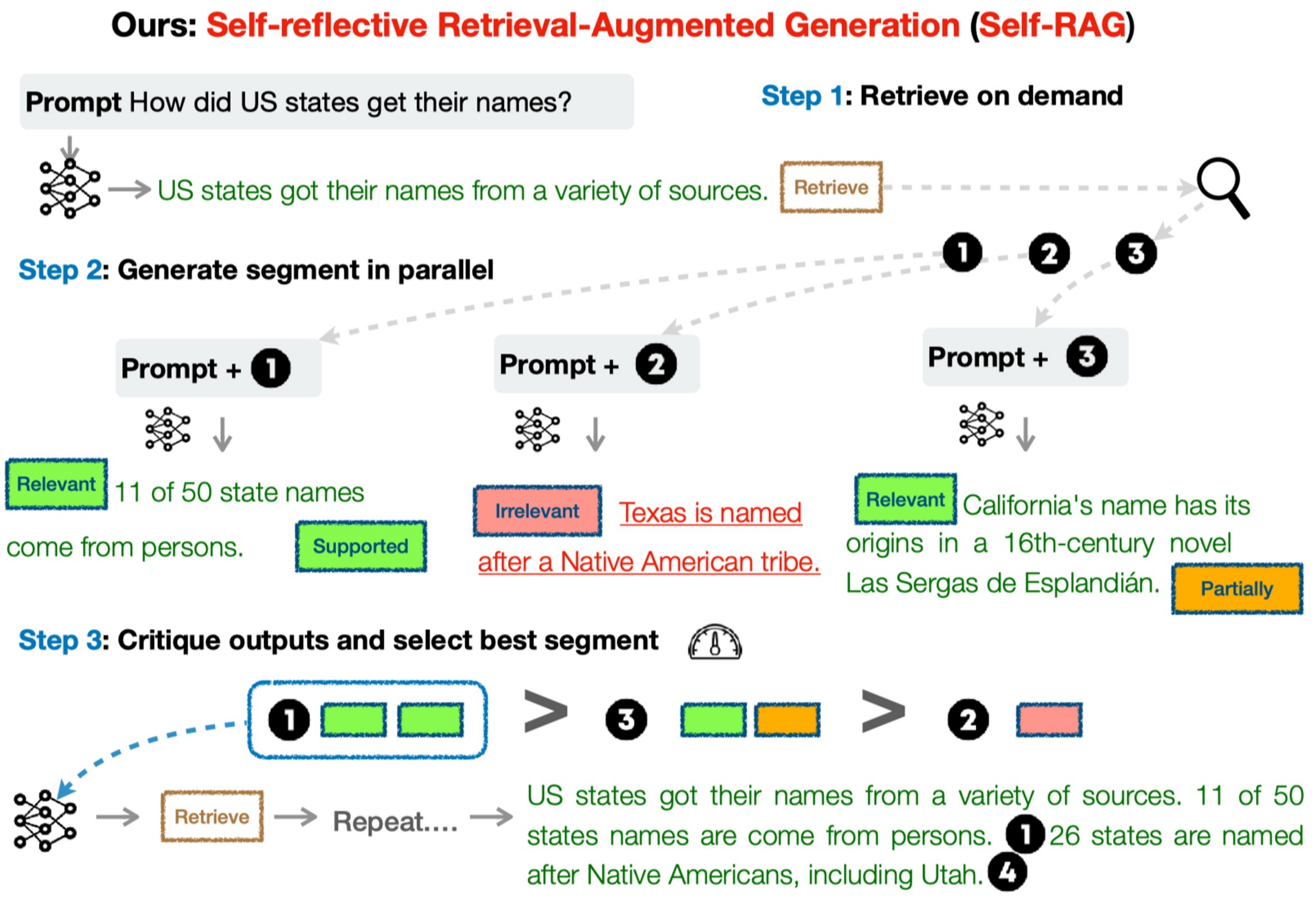
Overview of Self-RAG (Inference)



Overview of Self-RAG (Inference)



Overview of Self-RAG (Inference)



How to train

- Two models: the critic model $\boldsymbol{C}$, the generator model $\boldsymbol{M}$ (both Llama 2-7B)

1. Train $\boldsymbol{C}$ to generate reflection tokens for evaluating retrieved passages and the quality of a given task output.

How to collect the training data for $\boldsymbol{C}$?

GPT-4-based data collections

- Use the instruction and demonstration pairs to prompt GPT-4

Instructions
Given an instruction, please make a judgment on whether finding some external documents from the web (e.g., Wikipedia) helps to generate a better response. Please answer [Yes] or [No] and write an explanation.

Demonstrations
Instruction Give three tips for staying healthy.
Need retrieval? [Yes]
Explanation There might be some online sources listing three tips for staying healthy or some reliable sources to explain the effects of different behaviors on health. So retrieving documents is helpful to improve the response to this query.

Instruction Describe a time when you had to make a difficult decision.
Need retrieval? [No]
Explanation This instruction is asking about some personal experience and thus it does not require one to find some external documents.

Instructions and demonstrations for **Retrieve**

GPT-4-based data collections

- Use the instruction and demonstration pairs to prompt GPT-4

Instructions
Given an instruction and an output, rate whether the response appears to be a helpful and informative answer to the query, from 1 (lowest) - 5 (highest). We call this score perceived utility. The detailed criterion is as follows: 5: The response provides a complete, highly detailed, and informative response to the query, fully satisfying the information needs. 4: The response mostly fulfills the need in the query, while there can be some minor improvements such as discussing more detailed information, having better structure of the response, or improving coherence. 3: The response is acceptable, but some major additions or improvements are needed to satisfy users' needs. 2: The response still addresses the main request, but it is not complete or not relevant to the query. 1: The response is barely on-topic or completely irrelevant.

Instruction Who is the current prime minister of the UK as of 2023?
Output Boris Johnson was the prime minister of the UK from 2019 - 2022.
Perceived utility 2
Explanation While the output provides a factually correct statement about the UK prime minister from 2019 to 2022, this instruction asks who the prime minister is as of 2023, so it doesn't answer the instruction. Therefore, the utility is 2.

Instructions and demonstrations for ISUSE

GPT-4-based data collections

- Use the instruction and demonstration pairs to prompt GPT-4
  - Through manual assessment on few sampled examples -> manual assessments show high agreement with GPT-4 predictions (almost 90% agree)
  - Train ***C*** In such GPT4-generated dataset with next token prediction loss:

$$\max_{\mathcal{C}} \mathbb{E}_{((x,y),r)\sim\mathcal{D}_{critic}} \log p_{\mathcal{C}}(r|x,y), \ \ r \text{ for reflection tokens.}$$

How to train

- Two models: the critic model $\boldsymbol{C}$, the generator model $\boldsymbol{M}$ (both Llama 2 7/13B)

1. Train $\boldsymbol{C}$ to generate reflection tokens for evaluating retrieved passages and the quality of a given task output.

2. Using $\boldsymbol{C}$, update the training corpus by inserting reflection tokens into task outputs **offline**.

How to train

- Two models: the critic model $\boldsymbol{C}$, the generator model $\boldsymbol{M}$ (both Llama 2 7/13B)

1. Train $\boldsymbol{C}$ to generate reflection tokens for evaluating retrieved passages and the quality of a given task output.

2. Using $\boldsymbol{C}$, update the training corpus by inserting reflection tokens into task outputs **offline**.

3. Train $\boldsymbol{M}$ on a curated corpus with interleaving passages retrieved by a retriever and reflection tokens predicted by $\boldsymbol{C}$.

Training

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

The training examples for generator *M*

Input Give an example of a situation where you have to delegate tasks (from `gpt4_alpaca`)

Output [Retrieve=No]The manager realizes that there is an extensive list of tasks that need to be completed in order to make the event a success.[Retrieve=Yes]<p>Delegation–Delegation is the assignment of any responsibility or authority to another person (normally from a manager to a subordinate) to carry out specific activities. It is one of the core concepts of management leadership. However, the person who delegated the work remains accountable for the outcome of the delegated work. Delegation empowers a subordinate to make decisions, i.e. it is a shifting of decision-making authority from one organizational level to a lower one. Delegation, if properly done, is not fabrication. The opposite of effective delegation is micromanagement, where a manager provides too much input,<p>[IsRel =Relevant]This includes tasks such as creating a guest list, designing and sending invitations, booking a venue, coordinating with vendors, planning the events program, and monitoring the budget.[IsSup =Partially Supported][Retrieve=No]Since it is not possible for the manager to handle everything alone, they must delegate tasks to their team members.[Retrieve=No]The manager would then assign each team member specific responsibilities based on their individual skillsets and strengths.[Retrieve=Yes]<p>Joint application design – Subject Matter Experts: These are the business users, the IS professionals, and the outside experts that will be needed for a successful workshop. This group is the backbone of the meeting<p>[IsRel =Irrelevant]For instance, the teams graphic designer might be tasked with designing the invitations, while the event coordinator would be in charge of planning the program and working with vendors...[Retrieve=No]By delegating tasks, the manager can ensure that all necessary tasks are completed in a timely manner, while also empowering their team members to take on responsibilities and contribute to the success of the event.[IsUse =5]

The training examples for generator *M*

Input Synthesize a poem based on this prompt:“The sun is setting” (from `stanford_alpaca`)

Output [Retrieve=No]The stars begin to twinkle bright the trees, scattered throughout the night; [Retrieve=No]The stars begin to twinkle bright the trees, scattered throughout the night [Retrieve=No] They flicker like a silent dance, A peaceful moment in its trance. [Retrieve=No] They flicker like a silent dance, A peaceful moment in its trance. [Retrieve=No] A chorus of crickets sing a lullaby, To every creature that roams the sky [IsUse =5]

Tasks and Datasets

- **Closed-set tasks** (Select the correct answers, using Acc as metric)
  - **PubHealth**: a fact verification dataset about public health
  - **ARC- Challenge**: a multiple-choice reasoning dataset

- **Short-form generations tasks** (whether gold answers are included in the model generations)
  - **PopQA**
  - **TriviaQA-unfiltered**

- **Long-form generation tasks**
  - **Bio:** a biography generation task (use official FactScore metric to evaluate)
  - **ALCE-ASQA:** a long-form QA task (use official correctness (str-em), fluency metric to evaluate)

Experiment

Models trained and reinforced using private data

RAG+LMs trained with private data

Baselines without retrieval

Baselines with retrieval

LM	Short-form PopQA (acc)	Short-form TQA (acc)	Closed-set Pub (acc)	Closed-set ARC (acc)	Long-form generations (with citations) Bio (FS)	ASQA (em)	ASQA (rg)	ASQA (mau)	ASQA (pre)	ASQA (rec)
LMs with proprietary data										
Llama2-c$_{13B}$	20.0	59.3	49.4	38.4	55.9	22.4	29.6	28.6	–	–
Ret-Llama2-c$_{13B}$	51.8	59.8	52.1	37.9	79.9	32.8	34.8	43.8	19.8	36.1
ChatGPT	29.3	74.3	70.1	75.3	71.8	35.3	36.2	68.8	–	–
Ret-ChatGPT	50.8	65.7	54.7	75.3	–	40.7	39.9	79.7	65.1	76.6
Perplexity.ai	–	–	–	–	71.2	–	–	–	–	–
Baselines without retrieval										
Llama2$_{7B}$	14.7	30.5	34.2	21.8	44.5	7.9	15.3	19.0	–	–
Alpaca$_{7B}$	23.6	54.5	49.8	45.0	45.8	18.8	29.4	61.7	–	–
Llama2$_{13B}$	14.7	38.5	29.4	29.4	53.4	7.2	12.4	16.0	–	–
Alpaca$_{13B}$	24.4	61.3	55.5	54.9	50.2	22.9	32.0	70.6	–	–
CoVE$_{65B}$ *	–	–	–	–	71.2	–	–	–	–	–
Baselines with retrieval										
Toolformer*$_{6B}$	–	48.8	–	–	–	–	–	–	–	–
Llama2$_{7B}$	38.2	42.5	30.0	48.0	78.0	15.2	22.1	32.0	2.9	4.0
Alpaca$_{7B}$	46.7	64.1	40.2	48.0	76.6	30.9	33.3	57.9	5.5	7.2
Llama2-FT$_{7B}$	48.7	57.3	64.3	65.8	78.2	31.0	35.8	51.2	5.0	7.5
SAIL*$_{7B}$	–	–	69.2	48.4	–	–	–	–	–	–
Llama2$_{13B}$	45.7	47.0	30.2	26.0	77.5	16.3	20.5	24.7	2.3	3.6
Alpaca$_{13B}$	46.1	66.9	51.1	57.6	77.7	**34.8**	36.7	56.6	2.0	3.8
Our SELF-RAG $_{7B}$	54.9	66.4	72.4	67.3	**81.2**	30.0	35.7	**74.3**	66.9	67.8
Our SELF-RAG $_{13B}$	**55.8**	**69.3**	**74.5**	**73.1**	80.2	31.7	**37.0**	71.6	**70.3**	**71.3**

Experiment

	Short-form		Closed-set		Long-form generations (with citations)					
	PopQA	TQA	Pub	ARC	Bio			ASQA		
LM	(acc)	(acc)	(acc)	(acc)	(FS)	(em)	(rg)	(mau)	(pre)	(rec)
LMs with proprietary data										
Llama2-c$_{13B}$	20.0	59.3	49.4	38.4	55.9	22.4	29.6	28.6	–	–
Ret-Llama2-c$_{13B}$	51.8	59.8	52.1	37.9	79.9	32.8	34.8	43.8	19.8	36.1
ChatGPT	29.3	74.3	70.1	75.3	71.8	35.3	36.2	68.8	–	–
Ret-ChatGPT	50.8	65.7	54.7	75.3	–	40.7	39.9	79.7	65.1	76.6
Perplexity.ai	–	–	–	–	71.2	–	–	–	–	–
Our SELF-RAG $_{7B}$	54.9	66.4	72.4	67.3	**81.2**	30.0	35.7	**74.3**	66.9	67.8
Our SELF-RAG $_{13B}$	**55.8**	**69.3**	**74.5**	**73.1**	80.2	31.7	**37.0**	71.6	**70.3**	**71.3**

- SELF-RAG outperforms ChatGPT in PubHealth, PopQA, biography generations, and ASQA (Rouge and MAUVE).

- SELF-RAG outperforms other RAG+LMs that trained with private data baselines

Experiment

LM	Short-form PopQA (acc)	TQA (acc)	Closed-set Pub (acc)	ARC (acc)	Long-form generations (with citations) Bio (FS)	(em)	(rg)	ASQA (mau)	(pre)	(rec)
LMs with proprietary data										
Llama2-c$_{13B}$	20.0	59.3	49.4	38.4	55.9	22.4	29.6	28.6	–	–
Ret-Llama2-c$_{13B}$	51.8	59.8	52.1	37.9	79.9	32.8	34.8	43.8	19.8	36.1
ChatGPT	29.3	**74.3**	70.1	**75.3**	71.8	35.3	36.2	68.8	–	–
Ret-ChatGPT	50.8	65.7	54.7	**75.3**	–	**40.7**	**39.9**	**79.7**	65.1	**76.6**
Perplexity.ai	–	–	–	–	71.2	–	–	–	–	–
Baselines without retrieval										
Llama2$_{7B}$	14.7	30.5	34.2	21.8	44.5	7.9	15.3	19.0	–	–
Alpaca$_{7B}$	23.6	54.5	49.8	45.0	45.8	18.8	29.4	61.7	–	–
Llama2$_{13B}$	14.7	38.5	29.4	29.4	53.4	7.2	12.4	16.0	–	–
Alpaca$_{13B}$	24.4	61.3	55.5	54.9	50.2	22.9	32.0	70.6	–	–
CoVE$_{65B}$ *	–	–	–	–	71.2	–	–	–	–	–
Baselines with retrieval										
Toolformer*$_{6B}$	–	48.8	–	–	–	–	–	–	–	–
Llama2$_{7B}$	38.2	42.5	30.0	48.0	78.0	15.2	22.1	32.0	2.9	4.0
Alpaca$_{7B}$	46.7	64.1	40.2	48.0	76.6	30.9	33.3	57.9	5.5	7.2
Llama2-FT$_{7B}$	48.7	57.3	64.3	65.8	78.2	31.0	35.8	51.2	5.0	7.5
SAIL*$_{7B}$	–	–	69.2	48.4	–	–	–	–	–	–
Llama2$_{13B}$	45.7	47.0	30.2	26.0	77.5	16.3	20.5	24.7	2.3	3.6
Alpaca$_{13B}$	46.1	66.9	51.1	57.6	77.7	**34.8**	36.7	56.6	2.0	3.8
Our SELF-RAG $_{7B}$	54.9	66.4	72.4	67.3	**81.2**	30.0	35.7	**74.3**	66.9	67.8
Our SELF-RAG $_{13B}$	**55.8**	**69.3**	**74.5**	**73.1**	80.2	31.7	**37.0**	71.6	**70.3**	**71.3**

SELF-RAG outperforms 65B LLMs with sophisticated prompt engineering

Ablation Study

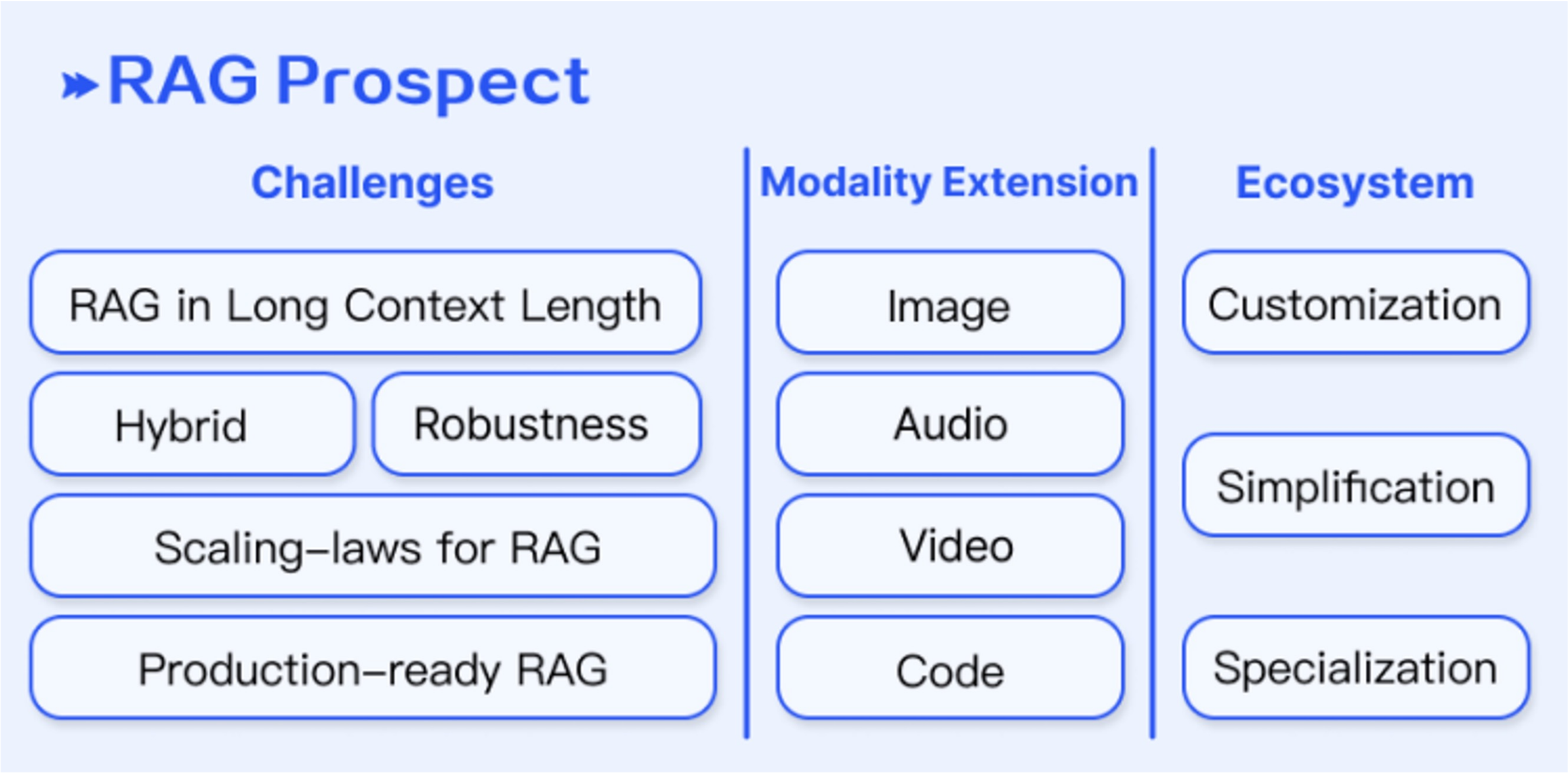
	PQA (acc)	Med (acc)	AS (em)
SELF-RAG (50k)	45.5	73.5	32.1
Training			
No Retriever $\mathcal{R}$	43.6	67.8	31.0
No Critic $\mathcal{C}$	42.6	72.0	18.1
Test			
No retrieval	24.7	73.0	–
Hard constraints	28.3	72.6	–
Retrieve top1	41.8	73.1	28.6
Remove ISSUP	44.1	73.2	30.6

Conclusion

- SELF-RAG, a new & SOTA framework to enhance the quality and factuality of LLMs through **retrieval on demand** and **self-reflection**.

- SELF-RAG trains an LM to learn to **retrieve, generate,** and **critique** text passages and its own generation by predicting the next tokens from its original vocabulary as well as designed **reflection tokens**.

Future Directions



Thanks!