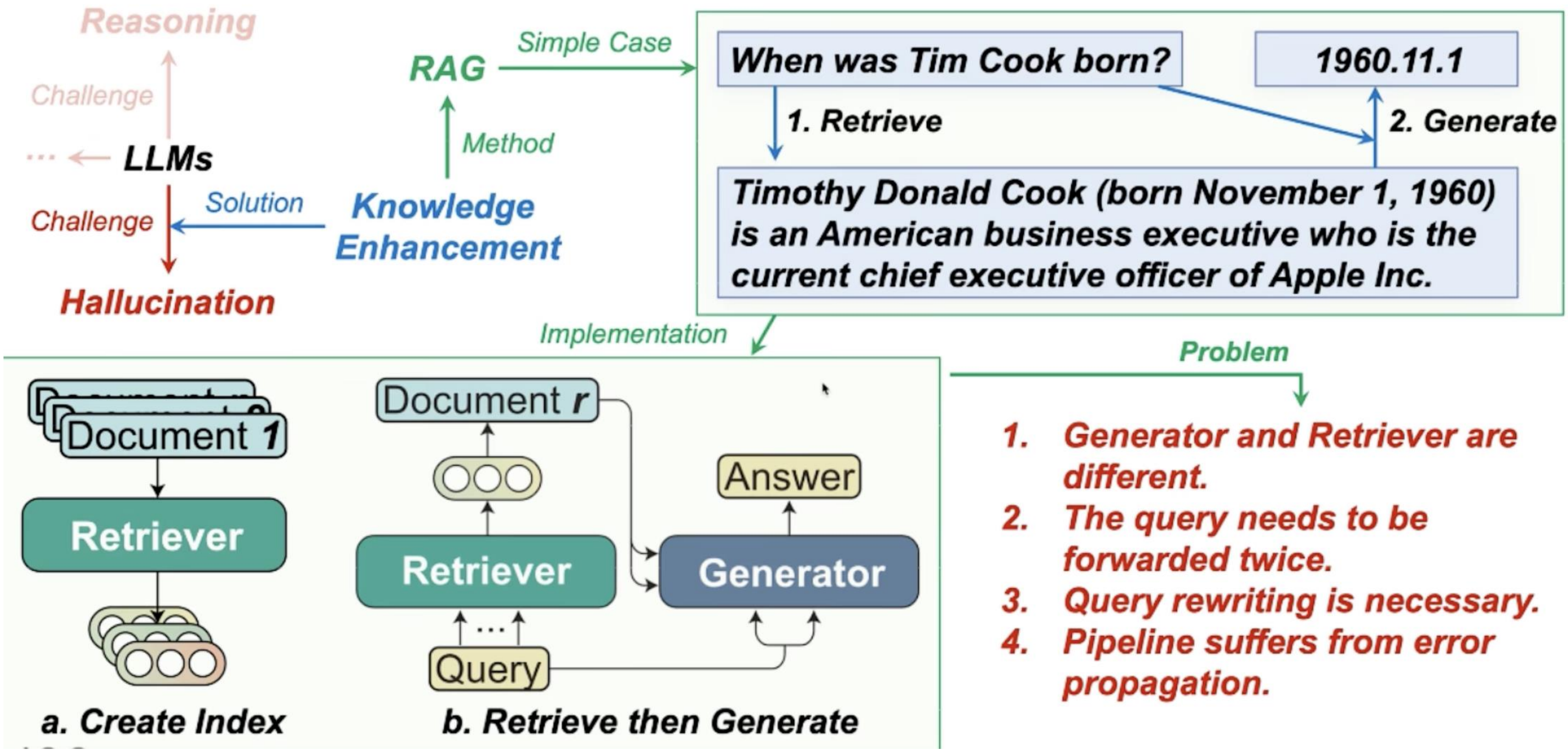
OneGen: Efficient One-Pass Unified Generation and Retrieval for LLMs

Author: Jintian Zhang et al. @ ZJU & Ant
Venue: EMNLP'24
Date: 2025.03.13

Introduction



Motivation

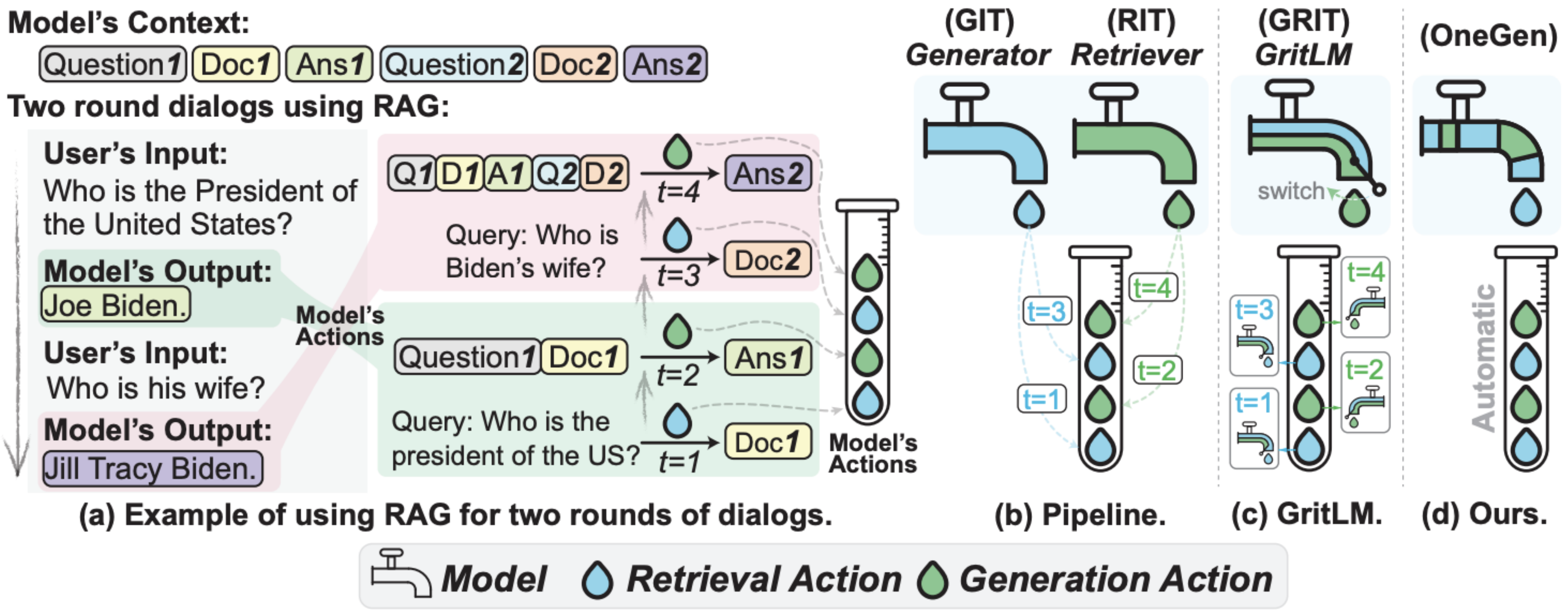
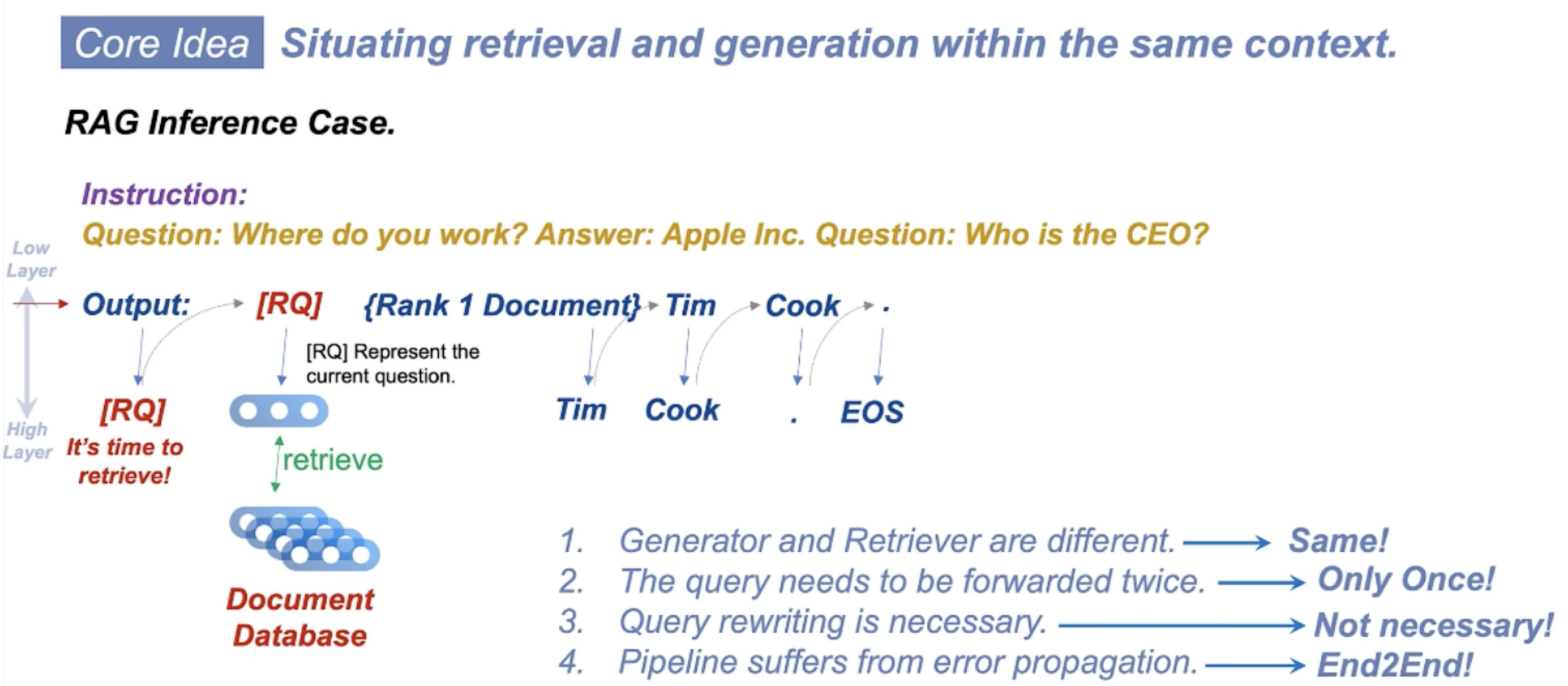


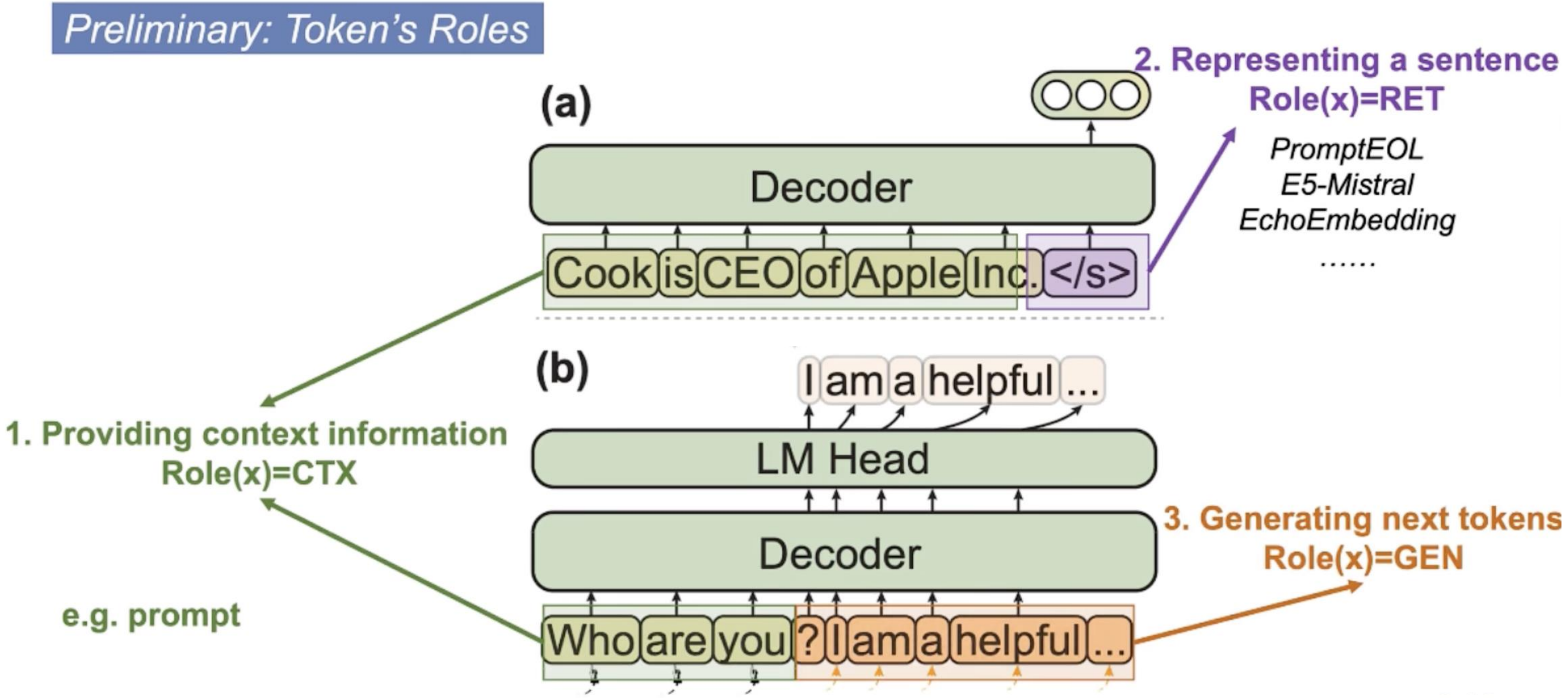
Figure 1: Comparison of Three Methods for RAG Task. (a) Two round dialogs using RAG (Retrieve and Generate twice each). (b) Pipeline approach requiring the deployment of two separate models for retrieval and generation, (c) GritLM (Muennighoff et al., 2024) utilizing a single model with a switching mechanism to integrate retrieval and generation, (d) OneGen (Ours) performing both functions automatically in the same model and the same context.

Main Idea



Core Idea
Situating retrieval and generation within the same context.
RAG Inference Case.
Instruction:
Question: Where do you work? Answer: Apple Inc. Question: Who is the CEO?
Low Layer
High Layer
Output:
[RQ]
{Rank 1 Document}
Tim
Cook
.
[RQ] Represent the current question.
[RQ]
It's time to retrieve!
retrieve
Document Database
Tim
Cook
.
EOS
1. Generator and Retriever are different. → Same!
2. The query needs to be forwarded twice. → Only Once!
3. Query rewriting is necessary. → Not necessary!
4. Pipeline suffers from error propagation. → End2End!

Method


Preliminary: Token's Roles
(a)
Decoder
Cook is CEO of Apple Inc. </s>
2. Representing a sentence
Role(x)=RET
PromptEOL
E5-Mistral
EchoEmbedding
……
(b)
I am a helpful ...
LM Head
Decoder
Who are you ? I am a helpful ...
1. Providing context information
Role(x)=CTX
e.g. prompt
3. Generating next tokens
Role(x)=GEN

Method

Training Details

$$\mathcal{L}_g = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|s_i|} \ell_g(f_{\theta\backslash\pi_{\text{Head}}}(x_{(i,\leq j)}), \pi_{\text{Head}})) \cdot \mathbb{1}_g(x_{i,j}).$$

$$\mathcal{L} := \lambda_g \boxed{\mathcal{L}_g} + \lambda_r \boxed{\mathcal{L}_r},$$

$$\mathcal{L}_r = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|s_i|} \ell_r\left(f_{\theta\backslash\pi_{\text{Head}}}(x_{(i,\leq j)}), f_{\theta\backslash\pi_{\text{Head}}}(x_{(i,\leq j)^+}), f_{\theta\backslash\pi_{\text{Head}}}(x_{(i,\leq j)^-})\right) \cdot \mathbb{1}_r(x_{i,j}).$$

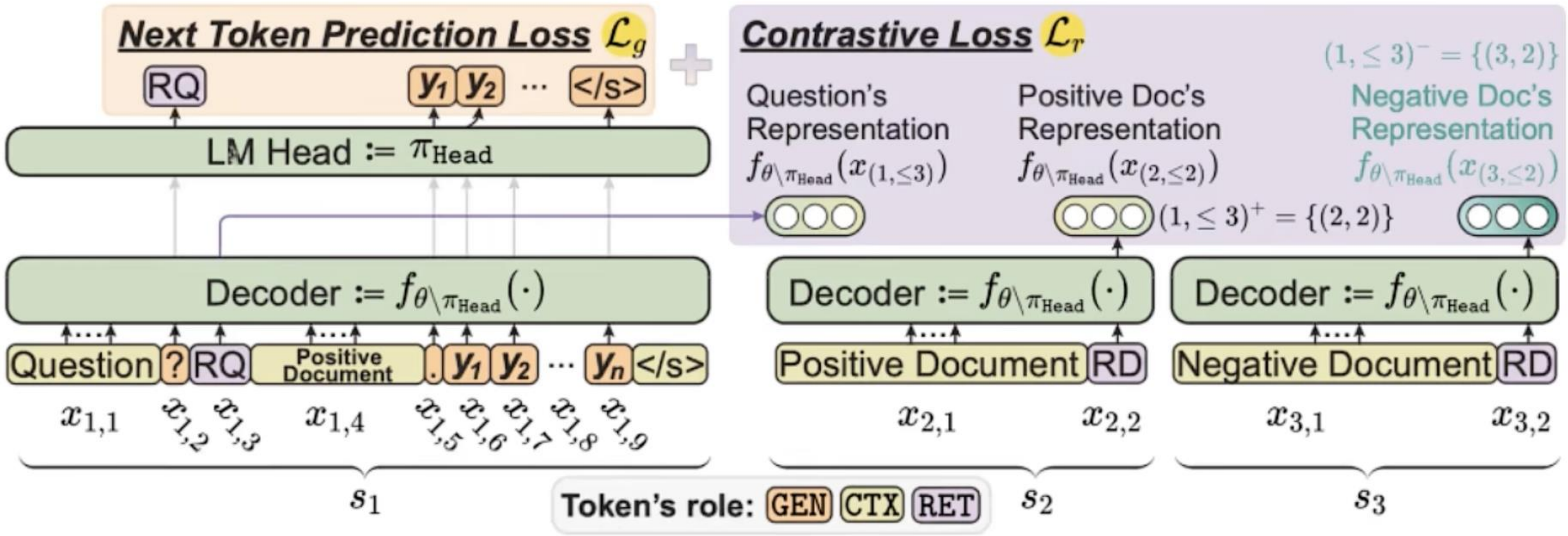


Figure 2: The training framework of unified **One**-pass **Gen**eration and retrieval (**OneGen**), illustrated using RAG. Detailed training process for other tasks can be found in Figure 6 of Appendix.

Introduction: LLM Personalization

Inference (RAG)

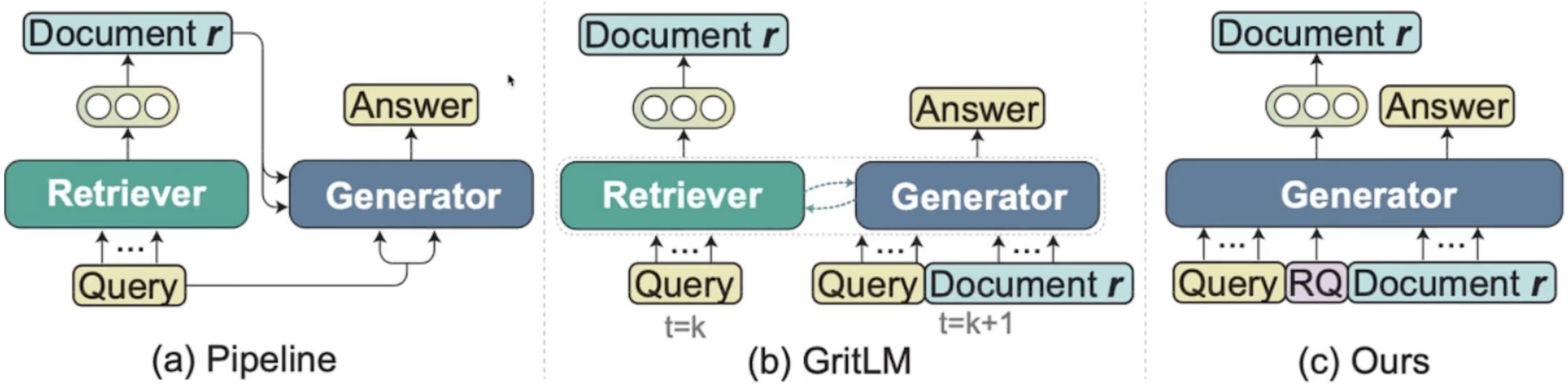


Figure 5: Comparison of three methods for completing RAG task.

Only one model !

Only one forward pass for query !

Introduction: LLM Personalization

Unify View

Method	Loss	Supported Data ($role(x_i) \in \{?\}$)		
		$\{\mathtt{CTX}, \mathtt{GEN}\}$	$\{\mathtt{CTX}, \mathtt{RET}\}$	$\{\mathtt{CTX}, \mathtt{RET}, \mathtt{GEN}\}$
GIT (SFT)	$\mathcal{L} = \mathcal{L}_g$	✓	✗	✗
RIT	$\mathcal{L} = \mathcal{L}_r$	✗	✓	✗
GRIT	$\mathcal{L} = \lambda_g \mathcal{L}_g + \lambda_r \mathcal{L}_r$	✓	✓	✗
OneGen	$\mathcal{L} = \lambda_g \mathcal{L}_g + \lambda_r \mathcal{L}_r$	✓	✓	✓

Table 8: Comparison of four Instruction Tuning

From the perspective of training methods, OneGen is an extension of GIT and RIT, and it can degenerate to GIT and RIT.

Experiment Setup

- Retrieve then Generate (RAG)
  - Single-Hop QA datasets: PopQA, TrivalQA, PubHealth, ARC
  - Multi-Hop QA datasets: HotpotQA, 2WIKI
- Generate then Retrieve
  - Entity Linking datasets: AIDA, OKE15, OKE16, REU, MSN, SPOT, K50

- Expectation: Performance does decrease while improving efficiency

Main Result

LLMs	Retriever			Dataset				AVG.
	Name	Dataset Name	Dataset Size	PopQA	TQA	Pub	ARC	
Toolformer (Schick et al., 2023)	Contriever	MS MARCO	1×10^6	-	48.8	-	-	-
Llama2$_{7B}$ (Touvron et al., 2023)	Contriever	MS MARCO	1×10^6	38.2	42.5	30.0	48.0	39.7
Alpaca$_{7B}$ (Dubois et al., 2023)	Contriever	MS MARCO	1×10^6	46.7	64.1	40.2	48.0	49.8
SAIL$_{7B}$ (Luo et al., 2023b)	Contriever	MS MARCO	1×10^6	-	-	69.2	48.4	-
Llama2-FT$_{7B}$ (Touvron et al., 2023)	Contriever	MS MARCO	1×10^6	48.7	57.3	64.3	65.8	59.0
Mistral$_{7B}$ (Jiang et al., 2023a)	Contriever	MS MARCO	1×10^6	23.2	49.3	52.0	39.0	40.9
GritLM$_{7B}$ (Muennighoff et al., 2024)	GritLM$_{7B}$	E5S(w/ TQA)	2×10^6	**58.0**	**66.5**	49.7	24.5	49.7
Self-RAG$_{7B}$ (Asai et al., 2024)	Contriever	MS MARCO	1×10^6	<u>52.5</u>	65.0	<u>72.2</u>	<u>67.3</u>	<u>64.3</u>
Self-RAG$_{7B}$ (+OneGen)	*Self*	*Sampled*	$\mathbf{6 \times 10^4}$	<u>52.5</u>	<u>65.7</u>	**75.1**	**70.1**	**65.8**

Table 2: Performance comparison across different datasets. "TQA" means TriviaQA, "Pub" means PublicHealth. The best and second-best results are indicated in bold and underlined. The complete table is shown in Table 9 of appendix. The details about Self-RAG are shown in appendix F.1.

BackBone	Retriever	Generation Performance				Retrieval Performance	
		HotpotQA		2WIKI		HotpotQA	2WIKI
		EM	F1	EM	F1	Recall@1	Recall@1
Llama2-7B	Contriever	52.83	65.64	70.02	74.35	73.76	68.75
	self	**54.82**	**67.93**	**75.02**	**78.86**	**75.90**	**69.79**
Llama3.1-7B	Contriever	53.72	66.46	70.92	75.29	69.79	66.80
	self	**55.38**	**68.35**	**75.88**	**79.60**	**72.55**	**68.98**
Qwen2-1.5B	Contriever	48.55	**61.02**	68.32	72.66	72.41	67.70
	self	**48.75**	60.98	**73.84**	**77.44**	**72.70**	**69.27**
Qwen2-7B	Contriever	53.32	66.22	70.80	74.86	74.15	69.01
	self	**55.12**	**67.60**	**76.17**	**79.82**	**75.68**	**69.96**

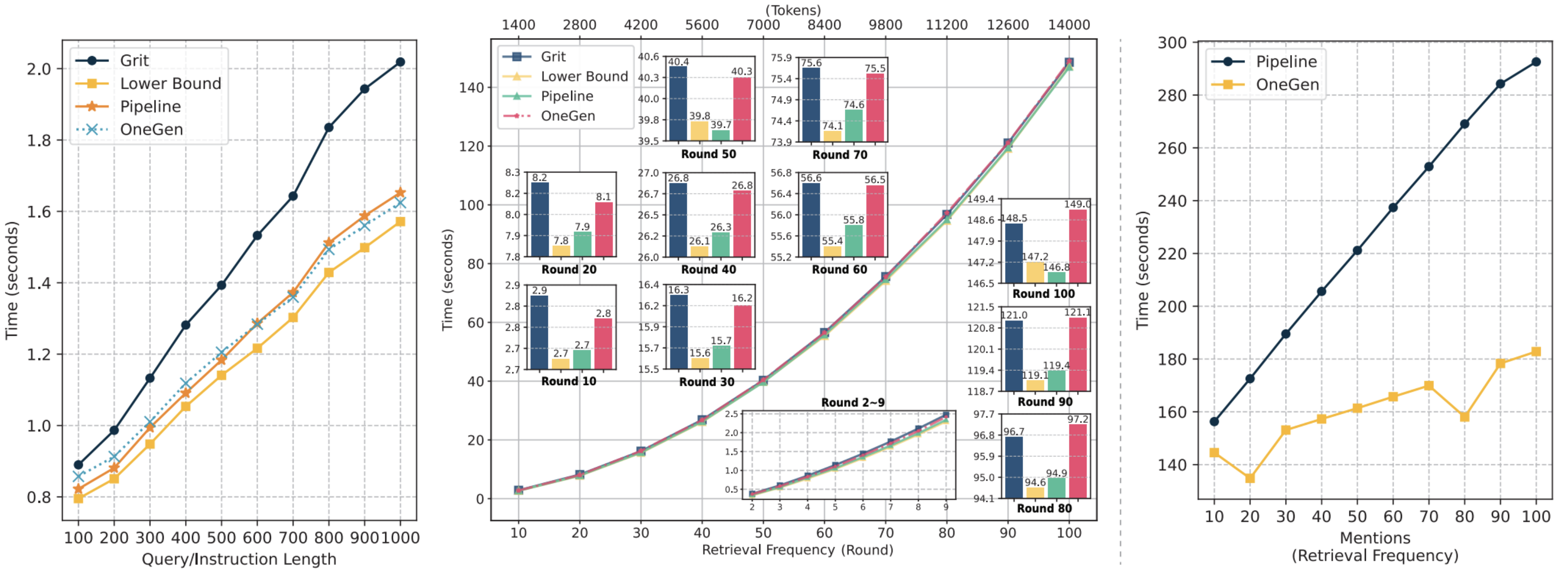
Table 3: In RAG for Multi-Hop QA settings, performance comparison across different datasets using different LLMs.

Method	Cand. Size	Training Data♦	In-domain	Out-of-domain						AVG.
			AIDA	OKE15	OKE16	REU	MSN	SPOT	K50	
Neural EL♦	< 30	AIDA	76.3	60.6	53.8	44.0	56.5	19.5	38.2	49.8
REL 2019◇	< 30	-	85.4	<u>66.5</u>	57.7	<u>53.0</u>	**77.8**	<u>24.9</u>	54.0	59.9
GENRE♦	< 30	WIKI 6M+AIDA	<u>85.3</u>	54.9	44.4	46.3	69.3	24.6	56.9	54.5
ReFinED♦	< 30	WIKI 6M+AIDA	**88.6**	**66.6**	<u>61.2</u>	49.8	<u>74.7</u>	22.2	<u>62.8</u>	<u>60.8</u>
Llama2$_{7B}$ (+OneGen)♦	1.25M	WIKI 60K+AIDA	83.1	63.5	**64.3**	**61.1**	74.2	**28.8**	**72.7**	**64.0**

Table 5: EL task performance on in-domain and out-of-domain test sets. The best value is in bold and the second best is underlined. The '♦' denotes end2end method, while the '◇' denotes pipelines.

Experiment

- Efficiency:



(a) Runing Time Variation in RAG Task with Query Length.

(b) Runing Time Variation in RAG Task with Dialogue Turns for an extreme scenario (Fixed Query Length per Retrieval).

(c) Runing Time Variation in Entity Linking Task with Query Length.

Figure 3: Efficiency analysis of OneGen on RAG and Entity Linking tasks. All baselines maintain the same settings. For RAG, the output is 10 tokens, with a document length of 30 tokens. Figure (a) illustrates the impact of query length on RAG efficiency across five dialogue rounds. Figure (b) examines the influence of retrieval frequency and token length on RAG efficiency. Figure (c) depicts how retrieval frequency affects efficiency in Entity Linking tasks.

Conclusion

- OneGen is the first to enable LLMs to conduct vector retrieval during the generation.
- OneGen harmonizes and expands both generative and representative instruction tuning.
- The results confirm that integrating generation and retrieval within the same context does not negatively impact the generative capabilities of LLMs, while also providing significant enhancements in retrieval capabilities.
- OneGen is pluggable, effective, training-efficient, and inference-efficient.