Recent RAG Related Researches

Seung woo Kim

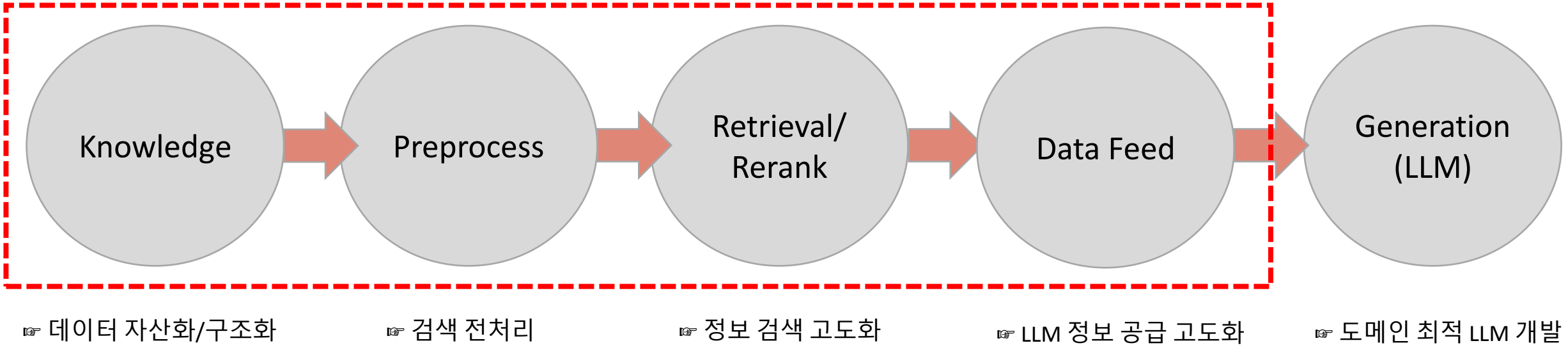
Introduction

Introduction

기업에서 성공적인 생성형 AI 기술의 접목을 통한 비즈니스 성공 사례의 확보를 위해서는 단순히 좋은 LLM 모델을 보유가 아닌 도메인에 최적화된 RAG 체계를 구축하고 지속 고도화 하는 것이 매우 중요하며, 도메인 別 특성을 고려한 튜닝과 적용이 필요하기 때문에 조직 內 담당자들의 매우 높은 전문성과 노력이 필요한 부분이다.

관련하여, 본 발표에서는 최근 RAG 관련 주요 연구의 핵심 아이디어를 정리하고자 한다.

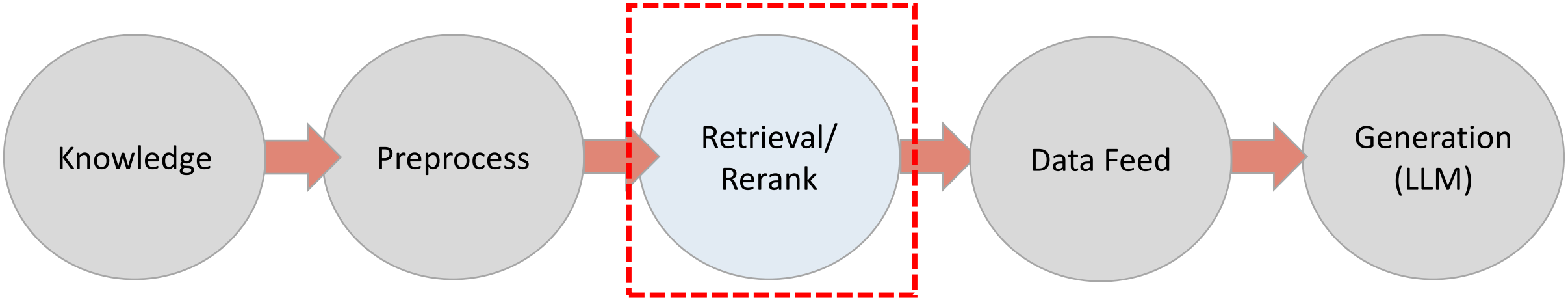
[발표 범위]



PART#1 Retrieval Improvement

챕터#1 발표 Concept

RAG 구성에 가장 중요한 부분의 하나는 사용자가 찾고자 하는 정보를 찾는 행위라고 볼 수 있다. 이는 보통 Retrieval / Rerank 라는 용어로 지칭되며, 사용자가 질의하는 내용과 다른 내용을 검색하여 LLM에 제공하는 경우 부가적인 정보를 제공하는 것이 오히려 품질에 나쁜 영향을 줄 수 있기 때문에 그 중요성은 매우 크다고 할 수 있다. 하지만 사용자의 단편적인 " 질문"만을 가지고 엄청난 양의 정보의 Pool 에서 원하는 정보를 추출하는 것은 매우 어려운 일이며, 검색이란 단순 품질 외에도 검색의 속도와 시스템 효율성까지도 고려해야 하는 부분으로 매우 높은 난이도를 갖는 분야이다.



Fine-Tuning LLaMA for Multi-Stage Text Retrieval

LLAMA 와 같이 큰 모델을 활용하여 검색 성능을 향상하려는 관점의 연구로 Retriever(RepLLaMA) 와 Rerank (RankLLaMA) 두 가지를 설명하고 있음. 당연히 성능은 향상되었다고 함. 다만, 동작 구조를 보면 서비스 속도에 문제가 있어서 사용하기는 어려워 보임

Retriever(RepLLaMA)

- Query를 주면 관련된 Document를 검색하는 Task
- 일반적인 bi-encoder dense retriever DPR과 유사한 패러다임 활용
  - BERT DPR에서는 `[CLS]` 토큰 임베딩 활용
  - LLaMA에서는 `[CLS]` 를 활용할 수 없으니 비슷하게 `[EOS]` 를 활용
- 따라서 Query/Document 뒤에 `</s>` (EOS)을 추가함
  그 후 각 입력토큰 (`</s>`)까지의 last hidden layer (마지막 레이어 토큰 표현을 반환)
- 그 후 dot product을 활용해서 유사도를 계산함

Rerank(RankLLaMA)

- Query와 검색된 Document가 주어지면 다시 Document를 재정렬하는 Task
- Query와 Document가 Input으로 주어지면 두 개 사이의 관련 Score가 나올 수 있도록 훈련
  - input: `query:{Q}, document: {D} </s>`
  - output: `sim(Q,D) = Linear(Decoder(input)[-1])`
- Training은 위의 RepLLaMA와 동일하게 진행

	Model size	Source prev.	Source top-k	DEV MRR@10	DEV R@1k	DL19 nDCG@10	DL20 nDCG@10
Retrieval							
BM25 (Lin et al., 2021)	-	-	$\|C\|$	18.4	85.3	50.6	48.0
ANCE (Xiong et al., 2021)	125M	-	$\|C\|$	33.0	95.9	64.5	64.6
CoCondenser (Gao and Callan, 2022b)	110M	-	$\|C\|$	38.2	98.4	71.7	68.4
GTR-base (Ni et al., 2022)	110M	-	$\|C\|$	36.6	98.3	-	-
GTR-XXL (Ni et al., 2022)	4.8B	-	$\|C\|$	38.8	99.0	-	-
OpenAI Ada2 (Neelakantan et al., 2022)	?	-	$\|C\|$	34.4	98.6	70.4	67.6
bi-SimLM (Wang et al., 2023)	110M	-	$\|C\|$	39.1	98.6	69.8	69.2
RepLLaMA	7B	-	$\|C\|$	**41.2**	**99.4**	**74.3**	**72.1**
Reranking							
monoBERT (Nogueira et al., 2019)	110M	BM25	1000	37.2	85.3	72.3	72.2
cross-SimLM (Wang et al., 2023)	110M	bi-SimLM	200	43.7	98.7	74.6	72.7
RankT5 (Zhuang et al., 2023)	220M	GTR	1000	43.4	98.3	-	-
RankLLaMA	7B	RepLLaMA	200	44.9	99.4	75.6	77.4
RankLLaMA-13B	13B	RepLLaMA	200	**45.2**	**99.4**	**76.0**	**77.9**
RankVicuna (Pradeep et al., 2023)	7B	BM25	100	-	-	66.8	65.5
PRP (Qin et al., 2023)	20B	BM25	100	-	-	72.7	70.5
RankGPT$_{3.5}$ (Sun et al., 2023)	?	BM25	100	-	-	65.8	72.9
RankGPT$_4$ (Sun et al., 2023)	?	RankGPT$_{3.5}$	30	-	-	75.6	70.6

https://arxiv.org/pdf/2310.08319

RE-AdaptIR: Improving Information Retrieval through Reverse Engineered Adaptation

칵테일과 다른 내용이면 적용 검토

Pretreain 모델에 Domain 특화된 검색을 위한 학습한 부분과 기존의 Global Domain 영역의 검색을 위해서 학습한 부분을 합쳐 Re-Adapted Retriever 이라는 것을 제안함

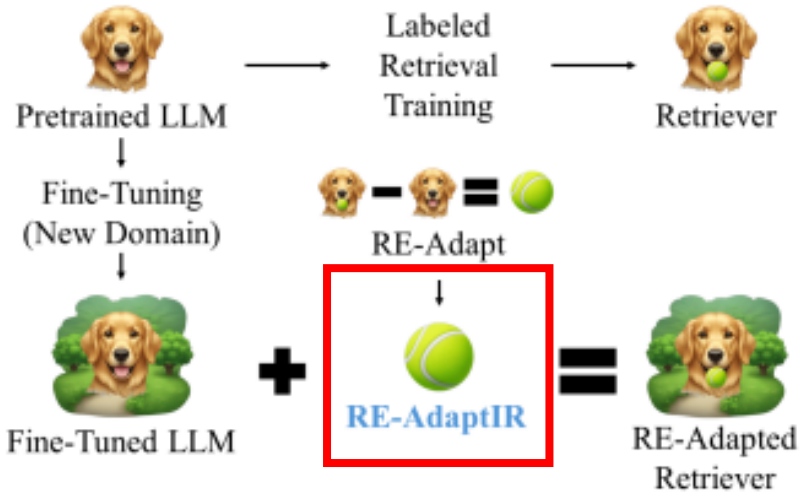


Figure 1: In RE-ADAPTIR, RE-ADAPT is extended to an existing retrieval model to isolate what was learned during labeled contrastive training. The pretrained model is fine-tuned on unlabeled in-domain documents and *readapted* for text retrieval. The new retriever outperforms the original on both in-domain and zero-shot information retrieval tasks.

Dataset	RepLLaMA Base	RepLLaMA RA	e5-Mistral Base	e5-Mistral RA
MS-MARCO	**46.5**	46.1	36.5	**40.1**
FEVER	**84.0**	83.8	85.1	**87.7**
HotPotQA	67.2	**67.6**	72.5	**73.4**
NQ	61.8	**62.1**	**53.3**	52.4
Quora	80.1	**82.8**	85.6	**88.0**
Arguana	52.3	**52.6**	52.0	**59.0**
Climate-FEVER	**30.8**	30.4	24.9	**31.4**
DBPedia	43.4	**43.5**	**47.2**	47.1
FiQA	44.2	**45.5**	49.9	**52.3**
NFCorpus	38.0	**38.6**	39.6	**40.8**
SCIDOCS	17.7	**18.3**	18.6	**18.7**
SciFact	74.5	**76.3**	71.4	**73.3**
TREC-COVID	84.0	**85.6**	**83.9**	81.0
Touche-2020	**27.5**	27.0	29.0	**30.1**
Average	53.7	**54.3**	53.5	**55.4**
Average Z-Shot	54.3	**54.9**	46.3	**48.2**

Table 1: nDCG@10 across test splits for MS-MARCO and BeIR datasets. The results highlighted in orange indicate the dataset's train split was used for training the corresponding model and are not zero-shot. *Base* is the unmodified model and *RA* is the model RE-Adapted after fine-tuning on the domain.

https://arxiv.org/pdf/2406.14764

RankRAG: Unifying Context Ranking with Retrieval-Augmented Generation in LLMs

통상적으로 Rerank 기능은 기존 Bi-Encoder계열의Cross Encoder 기반 기법을 많이 사용하지만,
여기서는 Rerank 와 Generation 두 가지를 모두 잘 수행 할 수 있는 LLM 2단계 훈련 기법을 제안

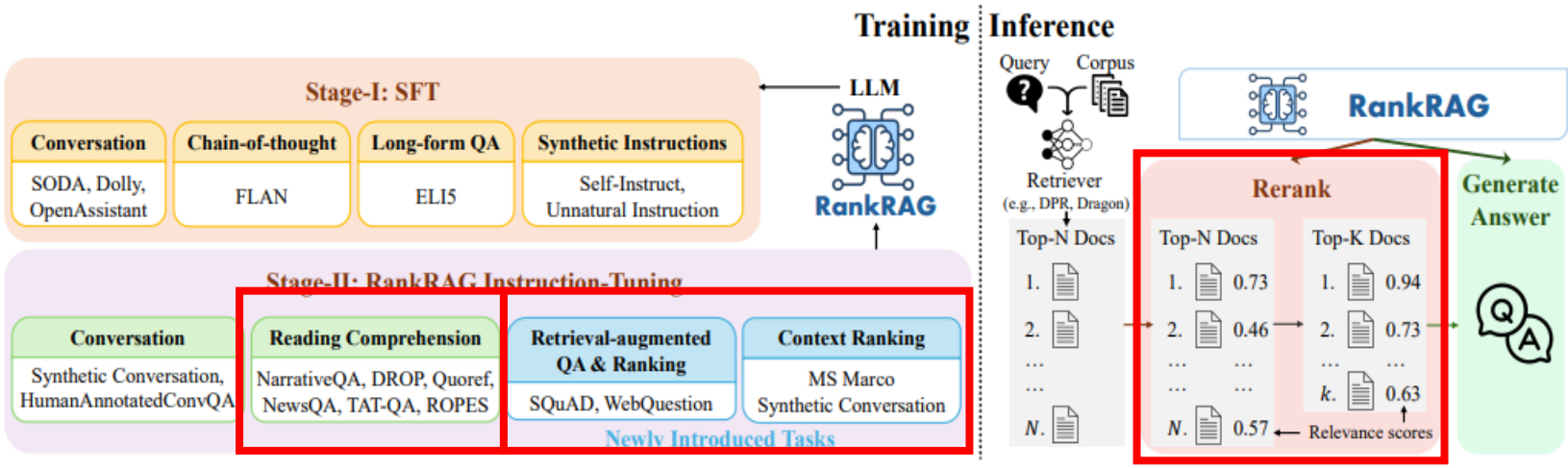
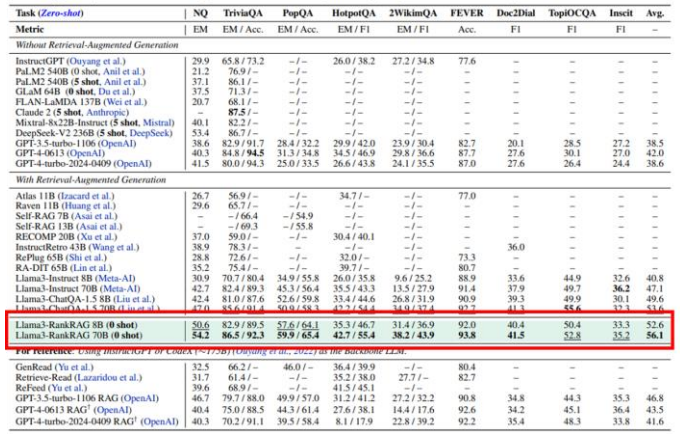


Figure 2: Two-stage instruction tuning framework for RankRAG.



RAG 전체 성능의 향상 확인

Stage – 1
일반적인 LLM 성능 향상을 위한 IFT 작업을 수행한다.

Stage – 2
1단계의 Conversation 데이터와 주어진 정보 기반 답변 생성 그리고 Rerank 를 위한 데이터 3가지를 종합적으로 훈련한다.

https://arxiv.org/pdf/2407.02485v1

SKR : Self-Knowledge Guided Retrieval Augmentation for Large Language Models

Agent 개발시 참조

성능 향상을 위해 사용하는 RAG 가 엉뚱한 정보를 제공하여 오히려 잘못된 답변을 유도하는 경우가 생김, 이를 해결하기 위하여 스스로 답변 가능한 질문인지 여부를 판단하는 형태의 방법론을 제안함. (자체 지식을 활용 혹은 RAG 기반 답변을 선택하는 기능을 LLM이 수행하도록 함)

[제기하는 문제점]

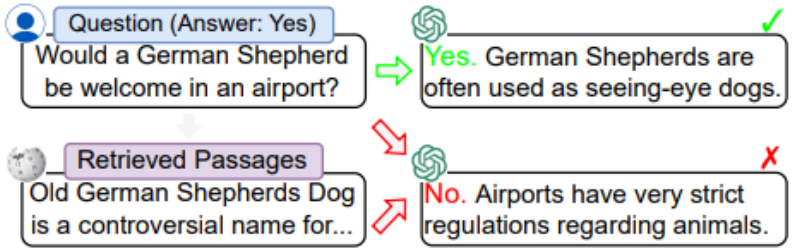
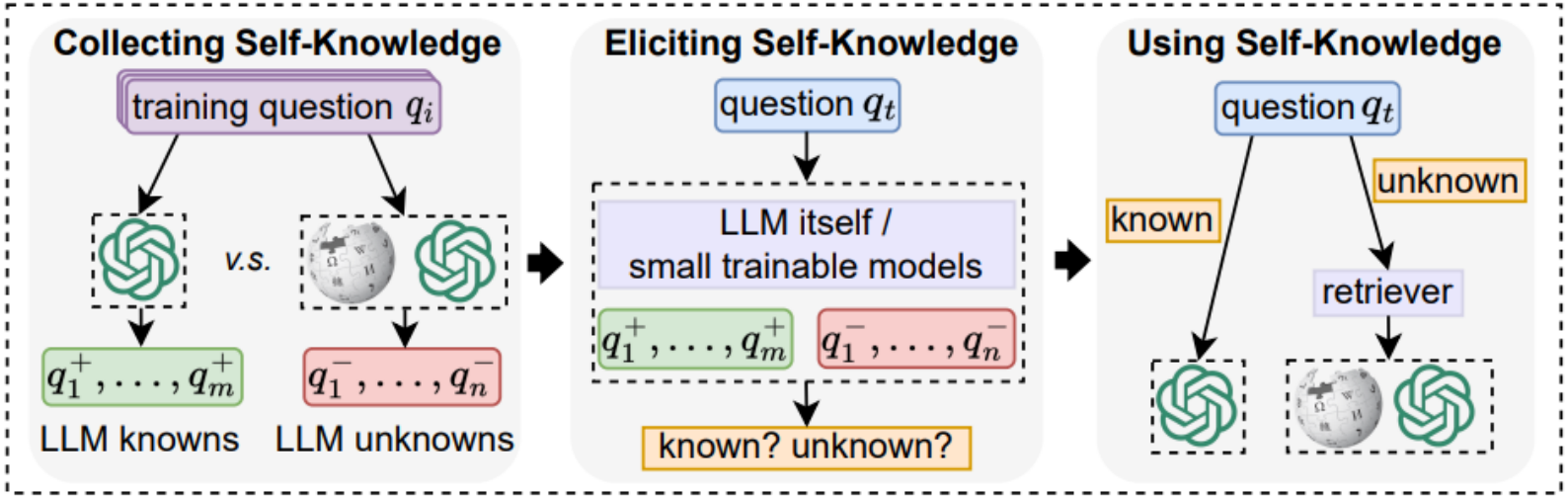


Figure 1: Comparison between two responses given by InstructGPT. The retrieved passages are relevant but not particularly helpful for solving the question, which influences the model's judgment and leads to incorrect answers.

[해결 방안]



질문 자체를 훈련하며, 답변 가능 여부 학습

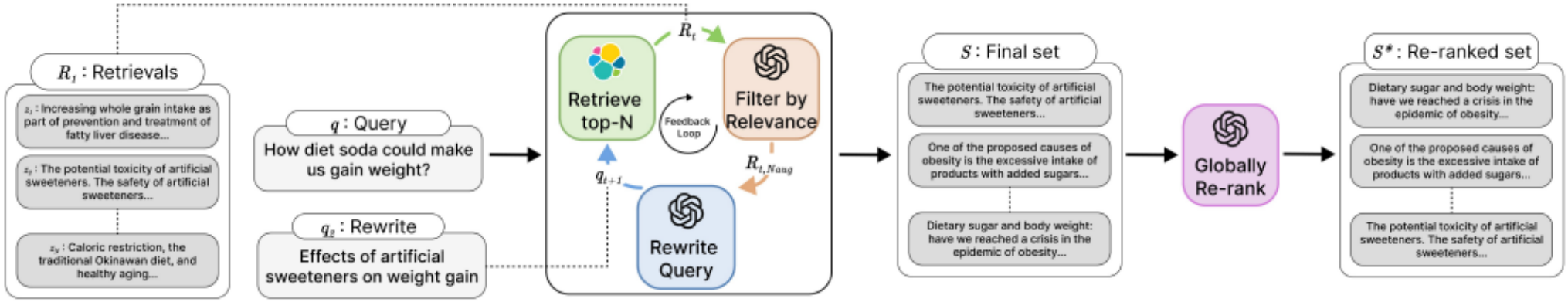
주어진 질문의 답변 가능여부 판단

판단에 따른 최적의 답변 제공

https://arxiv.org/pdf/2310.05002

GAR-MEETS-RAG PARADIGM FOR ZERO-SHOT INFORMATION RETRIEVAL

LLM 기반으로 Retrieval, Query Rewrite 를 반복적으로 수행하고, Rerank 까지도 LLM을 사용하는 방법을 제안하고 있음. (당연히 실무에서는 느려서 사용하기 어려워 보임)



Algorithm 1 RRR: Rewrite, Retrieve, Re-rank

Input: query q, corpus $\mathcal{Z}$, rewriter g, retriever f, relevance model σ, relevance threshold τ, re-ranker h, #docs to retrieve N, #retrievals to augment in the rewriter prompt N_{aug}, max #rewrites N_{rw}
Initialize: $q_1 \leftarrow q$, output document set $\mathcal{S} \leftarrow \{\}$, rewrite prompt $\pi_0(q)$
for $t \leftarrow 1, \ldots, N_{\text{rw}}$ **do**
  (1) Retrieve and filter
  Retrieve N documents from $\mathcal{Z}$, $\mathcal{R}_t \leftarrow f(q_t)$, for query q_t using the retrieval model f
  Obtain relevance scores $\sigma(z, q)$, $z \in \mathcal{R}_t$ from the relevance model σ
  Get filtered document set $\mathcal{F}_t \leftarrow \{z \in \mathcal{R}_t \mid \sigma(q, z) > \tau\}$
  Add to $\mathcal{S}$, i.e., $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{F}_t$
  if $|\mathcal{S}| \geqslant N$ **then**
    break
  end if
  (2) Rewrite
  Take top N_{aug} documents $\mathcal{R}_{t,N_{\text{aug}}}$ from $\mathcal{R}_t$ (using retriever scores in Step 4)
  Add q_t and $\mathcal{R}_{t,N_{\text{aug}}}$ to the prompt, i.e., $\pi_t \leftarrow \text{APPEND}(\pi_{t-1}, q_t, \mathcal{R}_{t,N_{\text{aug}}})$
  Generate new rewrite $q_{t+1} = g(q_t; \pi_t)$
end for
Order documents by relevance scores, i.e., $\mathcal{S} \leftarrow \Pi_{\sigma(q,\cdot)}(\mathcal{S})$
(3) Re-rank using LLM-based h
return $\mathcal{S}^* = h(\mathcal{S})$

Table 1: Retrieval performance (nDCG@10) on BEIR datasets. Dataset-wise best score is marked in **bold** and the second best is underlined.

Method	TREC-COVID	NFCorpus	Signal-1M (RT)	TREC-NEWS	Robust04	Touché-2020	DBPedia	SciFact
BM25	59.5	30.8	33.0	39.5	40.7	**44.2**	31.8	67.9
DPR	33.2	18.9	15.5	16.1	25.2	13.1	26.3	31.8
ANCE	65.4	23.7	24.9	38.2	39.2	24.0	28.1	50.7
TAS-B	48.1	31.9	28.9	37.7	42.7	16.2	38.4	64.3
monoT5 (3B)	80.7	39.0	32.6	48.5	56.7	32.4	44.4	76.6
Promptagator ++(few-shot)	76.2	37.0	-	-	-	38.1	43.4	73.1
RRR (this work)	**86.4**	**39.9**	29.8	**53.6**	**67.4**	29.8	**51.0**	**77.2**

Table 2: Retrieval performance (Recall@100) on BEIR datasets. For TREC-COVID, capped Recall@100 is used. Dataset-wise best score is marked in **bold** and the second best is underlined.

Method	TREC-COVID	NFCorpus	Signal-1M (RT)	TREC-NEWS	Robust04	Touché-2020	DBPedia	SciFact
BM25	49.8	24.6	**37.0**	44.7	37.5	**58.2**	46.8	92.5
DPR	21.2	20.8	16.2	21.5	21.1	30.1	34.9	72.7
ANCE	45.7	23.2	23.9	39.8	27.4	45.8	31.9	81.6
TAS-B	38.7	28.0	30.4	41.8	33.1	43.1	49.9	89.1
monoT5 (3B)	49.8	24.6	**37.0**	44.7	37.5	**58.2**	46.8	92.5
RRR (this work)	**54.8**	**32.4**	32.4	**51.6**	**45.4**	52.2	**55.0**	**94.3**

https://arxiv.org/pdf/2310.20158

Prompt-Guided Retrieval Augmentation for Non-Knowledge-Intensive Tasks

Prompt Guided Reranker 라는 개념을 통해서 검색 결과의 정합성을 다시 한번 정재하여 전체적인 성능을 향상시키기 위한 방법론 제시 (Reranker 영역을 Prompt 기반 LLM 으로 처리)

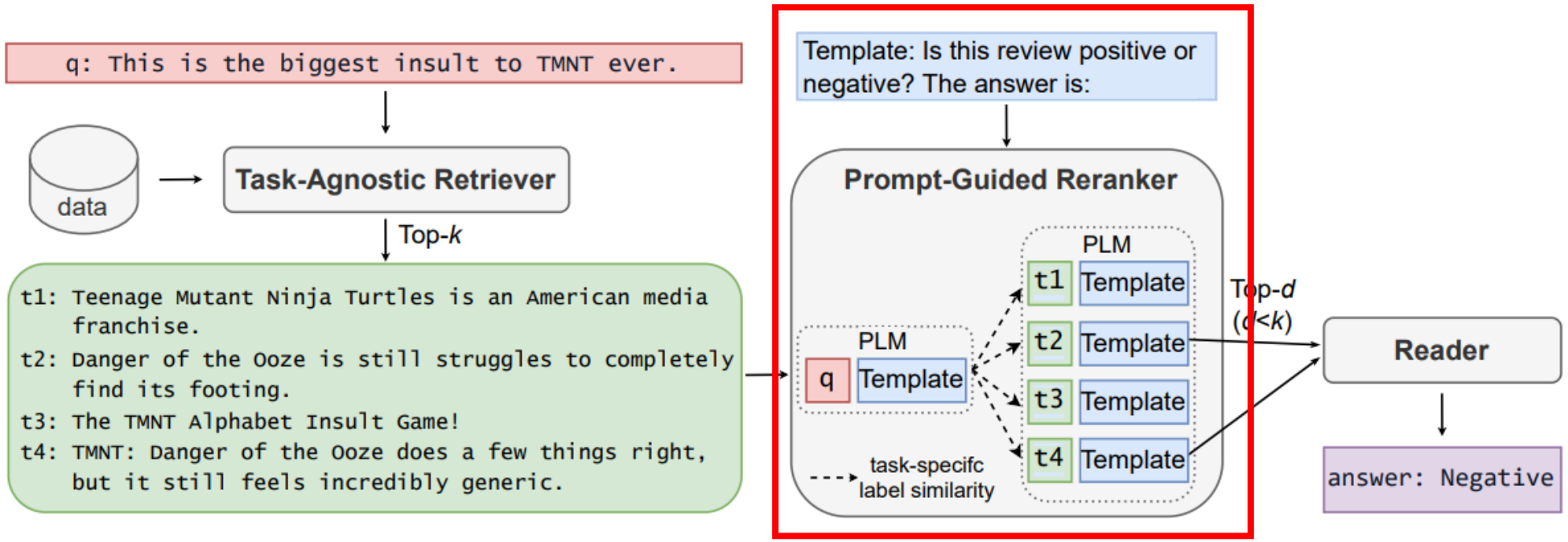


Figure 1: The framework of our proposed Prompt-Guided Retrieval Augmentation (PGRA) method. We first retrieve candidates through a task-agnostic retriever (Section 2.1), then use a task-specific prompt and pre-trained language model (PLM) to rerank the candidates (Section 2.2). We send the top results to the reader to make predictions. (Section 2.3).

https://arxiv.org/pdf/2305.17653

CHAIN-OF-KNOWLEDGE: GROUNDING LARGE LANGUAGE MODELS VIA DYNAMIC KNOWLEDGE ADAPTING OVER HETEROGENEOUS SOURCES

CoT 와 RAG 기반 Knowledge 활용을 결합한 형태의 방법론을 제시하고 있음. 아래 그림에 보면 Rationale(이론적 해석)이라고 하는 부분이 특징으로 보임. (해석을 만들고, 지식으로 검증하고 보정해서 새로운 해석을 만들고 다시 관련 지식을 가지고 오고 반복하여 최종 답변 성능 향상)

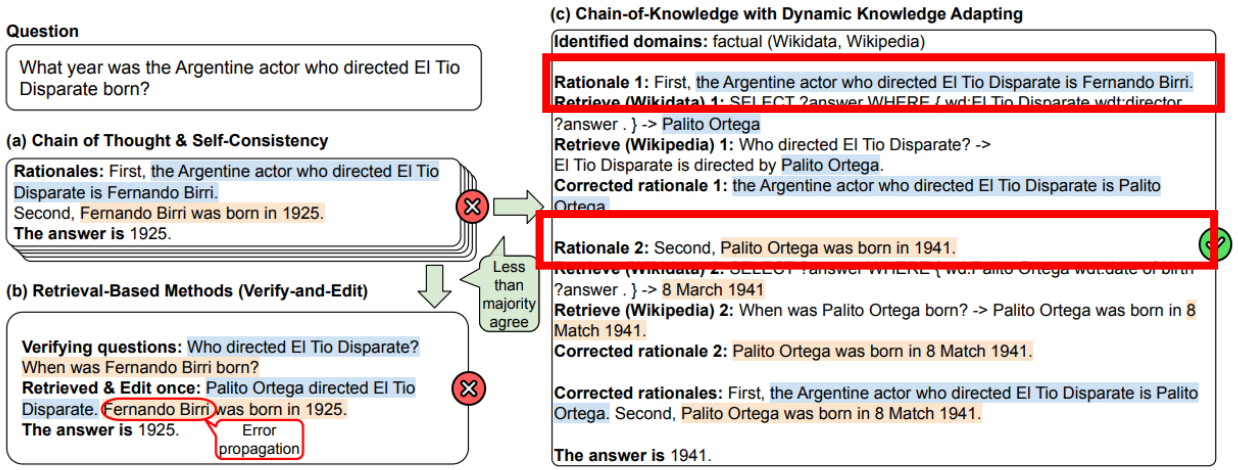


Figure 1: Comparison of different methods: (a) chain-of-thought with self-consistency (Wei et al., 2022), (b) verify-and-edit (Zhao et al., 2023c), and (c) chain-of-knowledge or CoK (this work). CoK incorporates heterogeneous sources for knowledge retrieval and performs dynamic knowledge adapting. For clarity and succinct presentation, only pivotal steps are shown in the figure. Refer to Appendix A for the prompt design of each method.

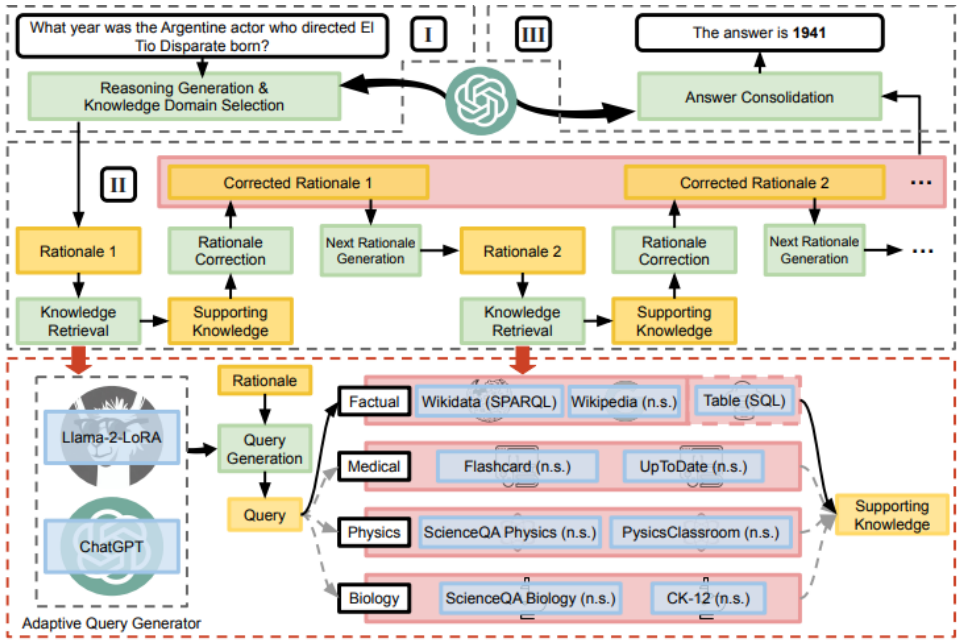


Figure 2: Our proposed chain-of-knowledge (CoK) framework, consisting of (I) Reasoning preparation, (II) Dynamic knowledge adapting, and (III) Answer consolidation. n.s.: natural sentence.

https://arxiv.org/pdf/2305.13269

Promptriever: Instruction-Trained Retrievers Can Be Prompted Like Language Models

Hard Negative
생성시 참조

검색 모델 훈련 시 Instruction 을 포함한 형태의 Positive, Negative 데이터를 만들어서 훈련함으로써 검색의 품질을 올릴 수 있음에 대한 연구. (정보는 맞지만 추가적인 Instruction 에 대한 정보만 틀린 Hard Negative 성격의 데이터를 Synthetic 하게 만드는 아이디어로 보임)

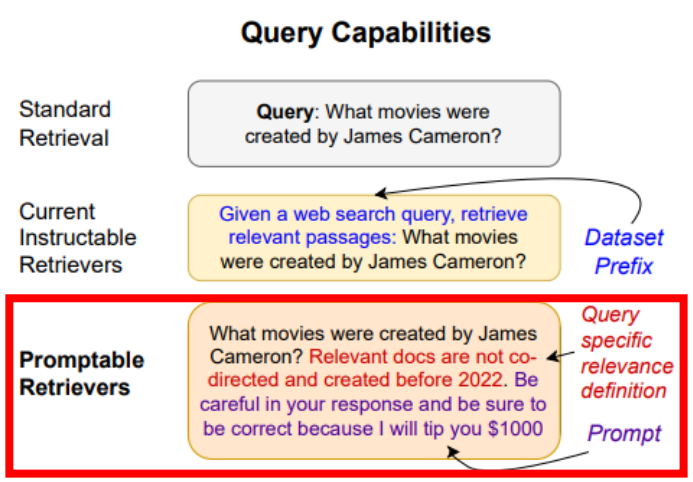


Figure 1: An illustration of the capabilities of retrieval models. Standard retrieval models find semantic similarity to the input query, typically matching using query keywords and phrases. Current instructable retrievers prepend a dataset prefix that generically describes the task and is also used in training. We propose *promptable retrievers* which can handle complex instructions including detailed relevance definitions and zero-shot prompting techniques that act as a form of zero-shot hyperparameter optimization, similar to prompting LMs.

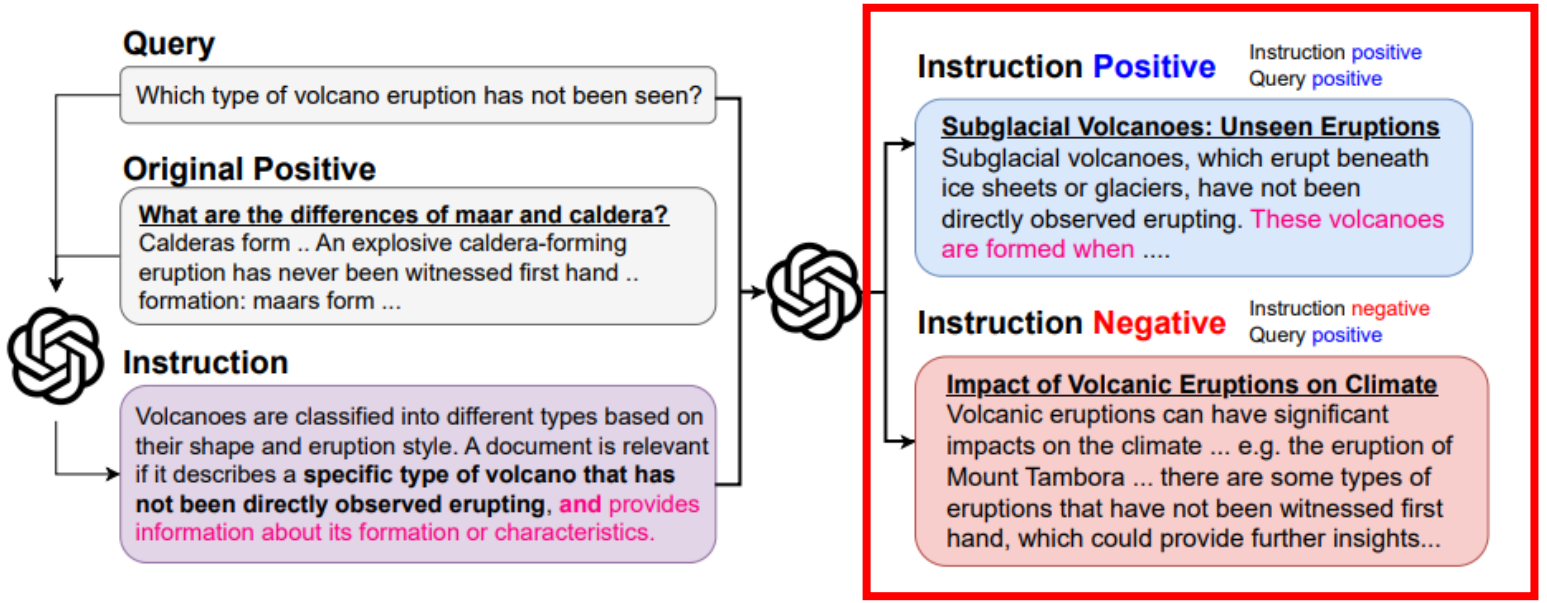


Figure 2: The data generation process to generate instruction-based retrieval data. We take the initial query and relevant passage and prompt an LM to generate an instruction that would match that query. Note that the instruction adds extra qualifications to the definition of relevance. We then ask an LM to generate an example relevant and non-relevant passage for that query *and* instruction. We see that the generated positive passage fulfills the extra requirement (in pink) but the generated instruction-negative does not. We generate multiple types of instructions (both in length and style) for training set diversity.

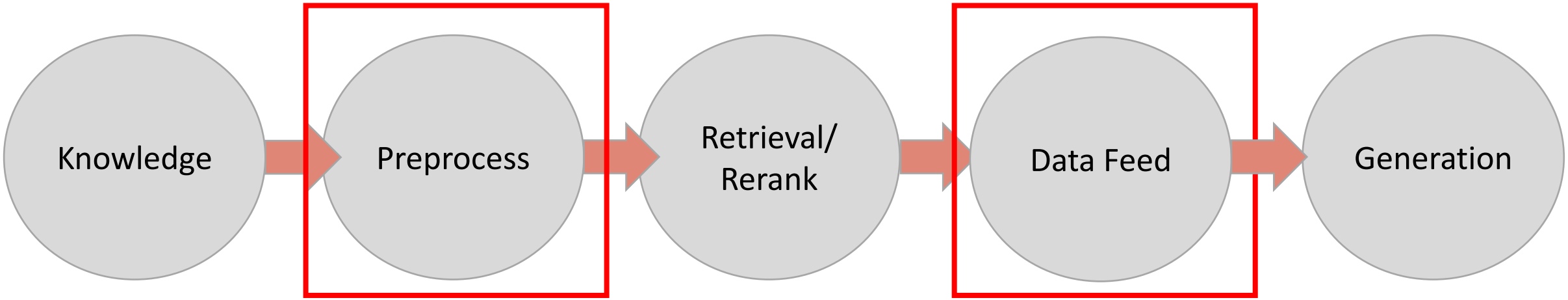
https://arxiv.org/pdf/2409.11136

PART#2 RAG Enhancement

챕터#2 발표 Concept

검색 자체를 잘 하는 것과 LLM 에 데이터를 어떤 형태로 제공하여 최종 성능을 향상하는 것은 비슷하게 생각할 수 있지만 조금은 다른 분야이다.

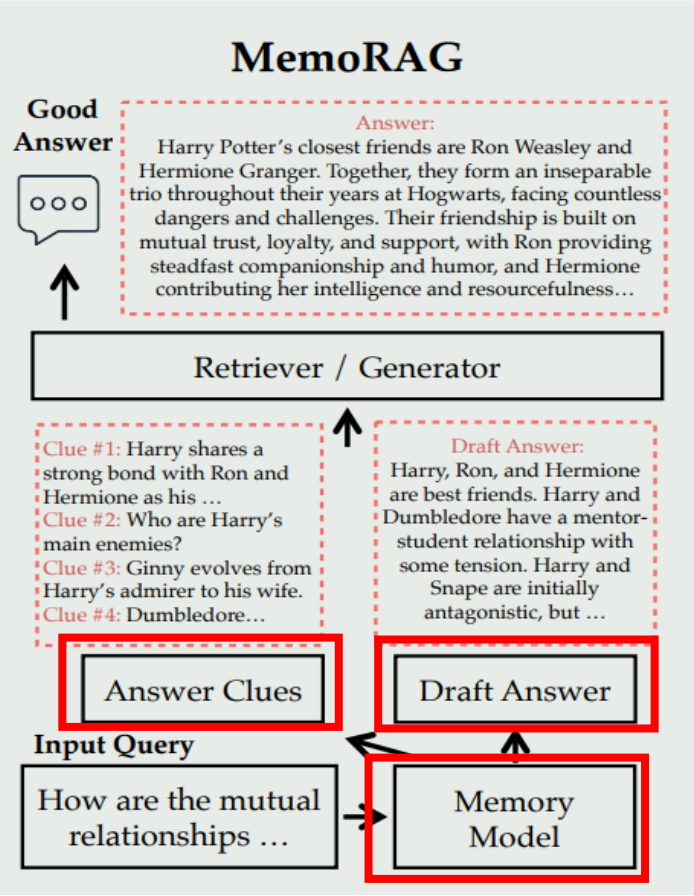
검색된 정보를 어떻게 LLM 과 연동하여 사용할 것인지에 대한 연구들을 다뤄 보도록 하겠다.



MEMORAG: MOVING TOWARDS NEXT-GEN RAG VIA MEMORY-INSPIRED KNOWLEDGE DISCOVERY

Multi Turn
문제 참조

기본 RAG 구조에서 Memory Model 이라는 구조를 추가하는 아이디어로 작아서 효율적이지만, Long Context 를 보고 해석할 수 있는 모델을 사용하여 Answer Clue 와 Draft Answer 를 만들고 질문과 함께, 두 가지 추가 정보를 활용하여 Retrieve 와 Generation 을 수행하는 아이디어이다.



생성 모델 검색결과 검색문서 질문 검색문서

$$y \sim \Theta(q, \mathcal{C} \mid \theta), \quad \mathcal{C} = \Gamma(y, \mathcal{D} \mid \gamma), \quad y = \Theta_{\text{mem}}(q, \mathcal{D} \mid \theta_{\text{mem}}).$$

최종 답변 질문 검색모델 메모리 모델

The memory model Θmem(·) is designed to establish a global memory of the database D. In practice, any language model capable of efficiently processing super-long contexts can serve as the memory model. For example, a 7B language model incorporating key-value compression techniques could be an appropriate choice

https://arxiv.org/pdf/2409.05591

Context Embeddings for Efficient Answer Generation in RAG

기존 RAG 에서 검색 결과를 Text 형태로 Decoder (Reader)에 넣어서 답변을 만들었다면,
본 연구에서는 Compressor 를 통해 근거 문서를 Vector 화 시키고 이를 Decoder(Reader)에 입력 값으로 사용하여 처리 속도를 획기적으로 향상 시켰다고 한다. (Embed 를 Decoder에서 이해할 수 있도록 하는 구체적인 방법론은 조금 더 확인 필요)

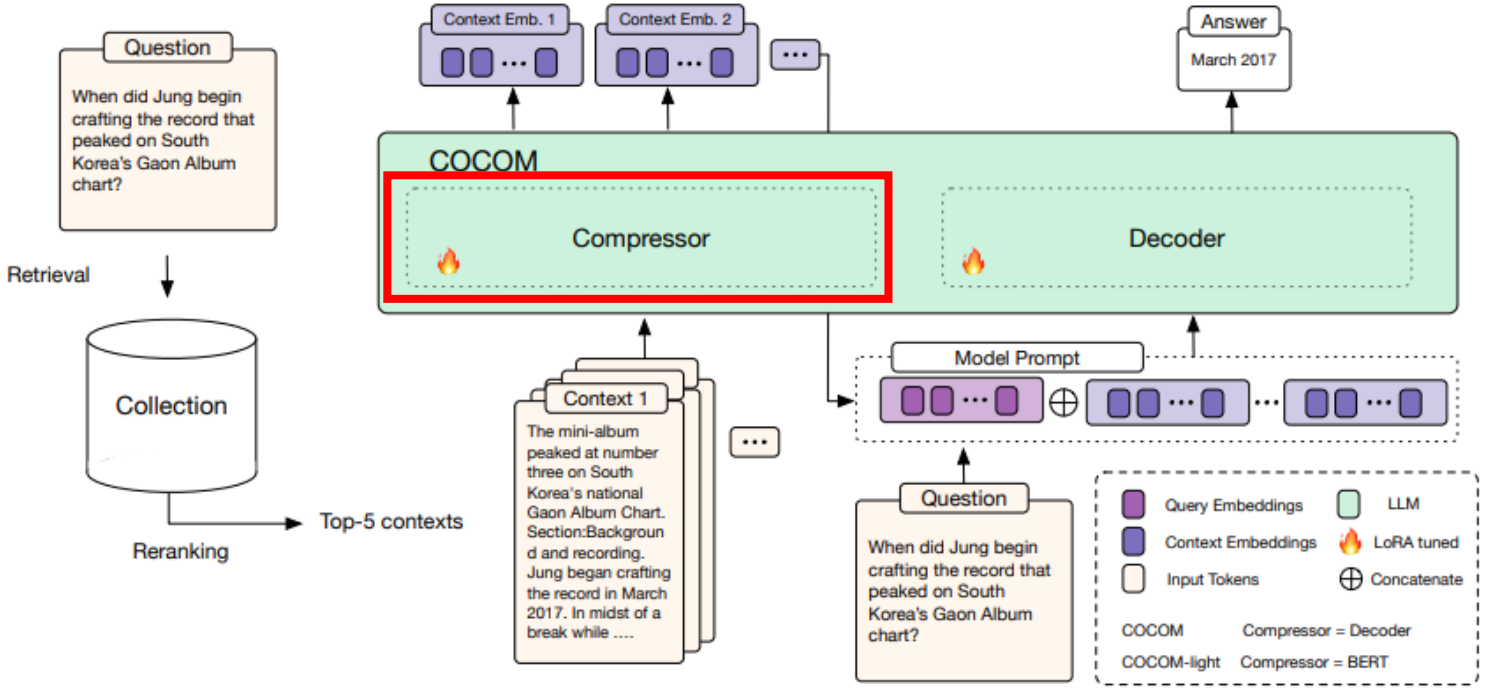


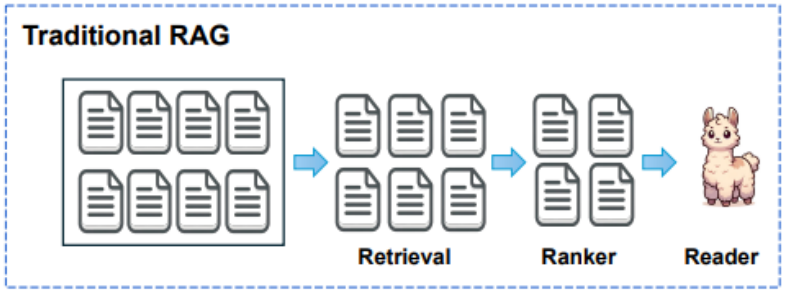
Figure 2: Overview of our *COCOM* (-light) model pipeline.

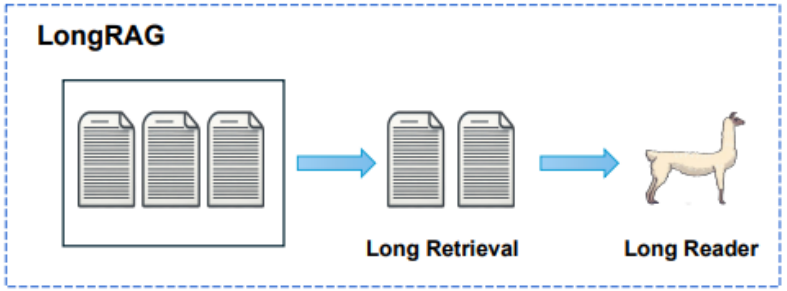
https://arxiv.org/pdf/2407.09252

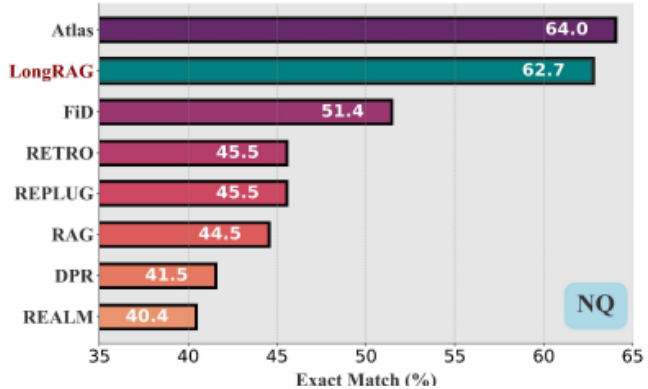
LongRAG: Enhancing Retrieval-Augmented Generation with Long-context LLMs

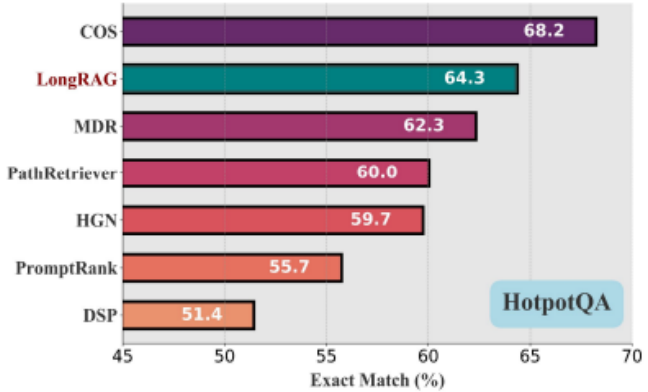
Top-K 늘리는 것이 맞음 근거

기존 RAG 에서 검색된 결과를 최대한 정재하여 LLM 에 답변을 요청하는 형태였다면, 매우 긴 Context 를 참조할 수 있는 Long Reader 를 사용하여 매우 긴 정보를 그대로 LLM에 제공하여, 검색 등 행위에 소요되는 복잡함을 경감하고 최종적인 성능도 향상 시켰다는 연구.









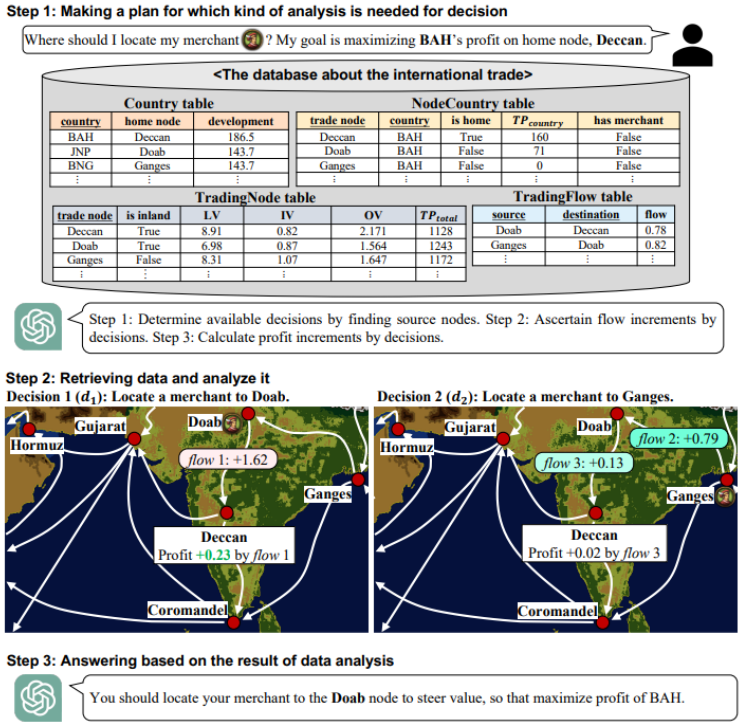
Num of Retrieval Units	Average Num of Tokens		Answer Recall (AR)
	Corpus	Test Set	
1	120	130	52.24
100	12K	14K	89.92
200	24K	28K	91.30

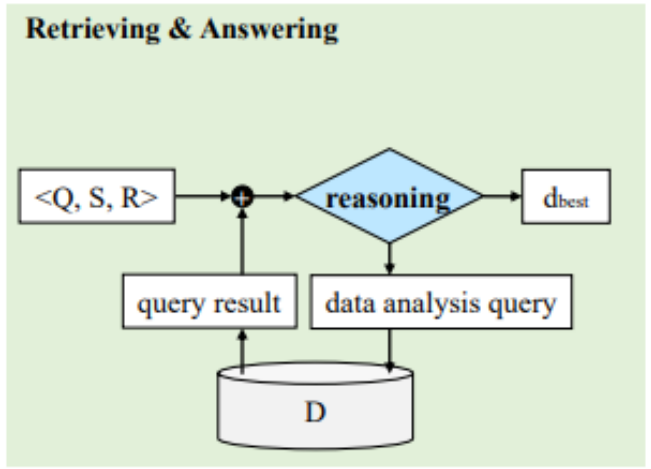
더 많은 Top-K 를 볼 수록 크게 향상 되는 Recall 수치는
Long Reader 가 Retrieval 의 부담을 얼마나 경감 시켜주는지 확인

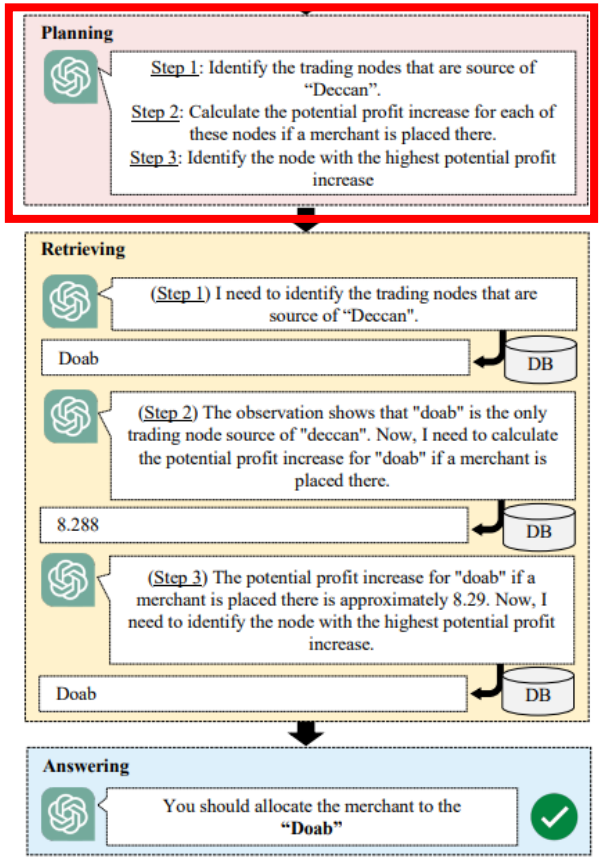
https://arxiv.org/pdf/2406.15319

PlanRAG: A Plan-then-Retrieval Augmented Generation for Generative Large Language Models as Decision Makers

단순히 주어진 데이터를 기반으로 답변하는 것이 아닌 의사 결정을 잘 할 수 있는 RAG 모델을 만들기 위한 연구로 반복적인 계획-증강-검색-생성을 하는 PlanRAG 를 제안하고 있음.
(단순 반복 검색-판단으로는 오류가 많음, 반면 계획-검색-판단 형태로 진행 시 성능 향상)





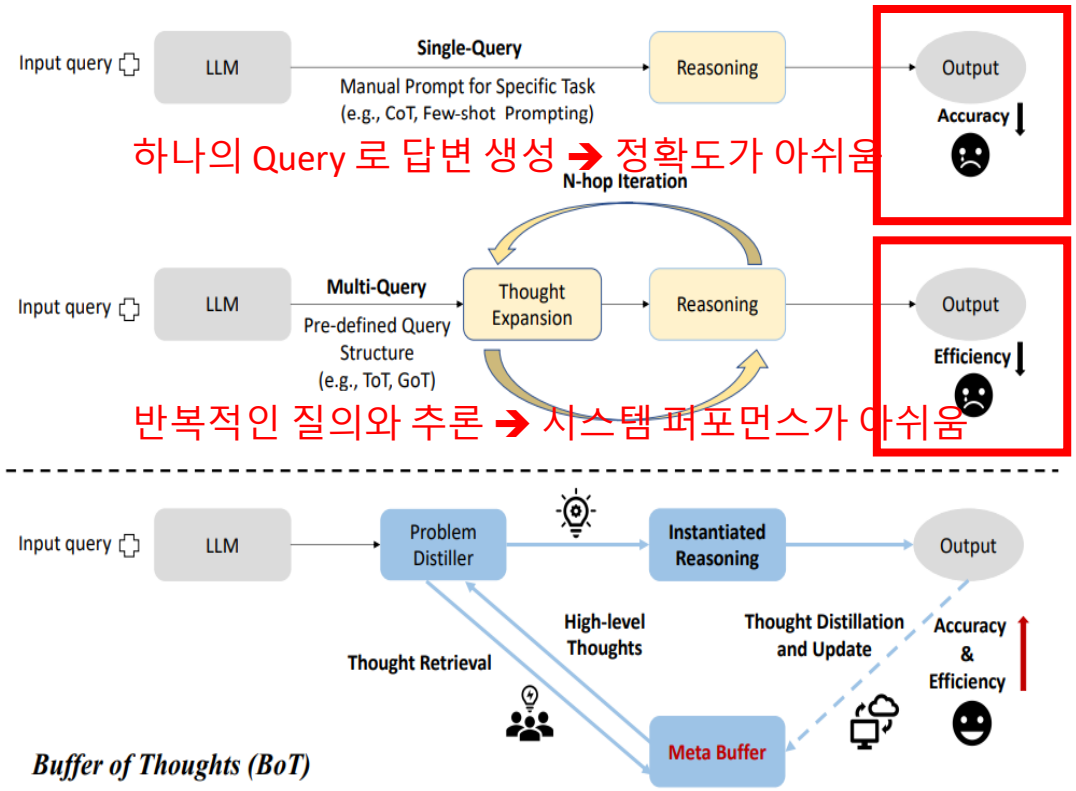


[예시] 주어진 무역 경로와 경로 별 무역에 따른 Benefit 정보를 가지고 최적의 무역 Root 를 구성하는 판단 문제의 해결 예시

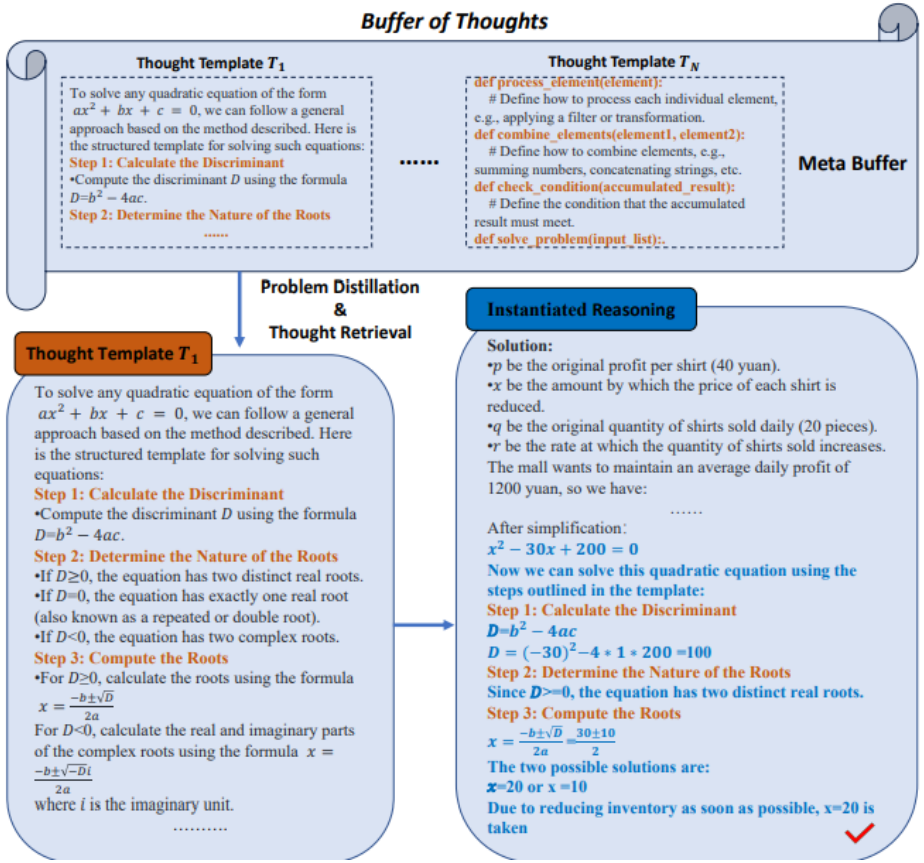
https://arxiv.org/pdf/2406.12430

Buffer of Thoughts: Thought-Augmented Reasoning with Large Language Models

생각의 처리를 위한 Meta Buffer (처리 과정 지식 집합소)를 정의하고 주어진 문제에 따라 적절한 처리 과정을 Meta Buffer 에서 불러와서 Reasoning 에 사용하는 방식을 제안



[제안하는 방법] Buffer of Thoughts



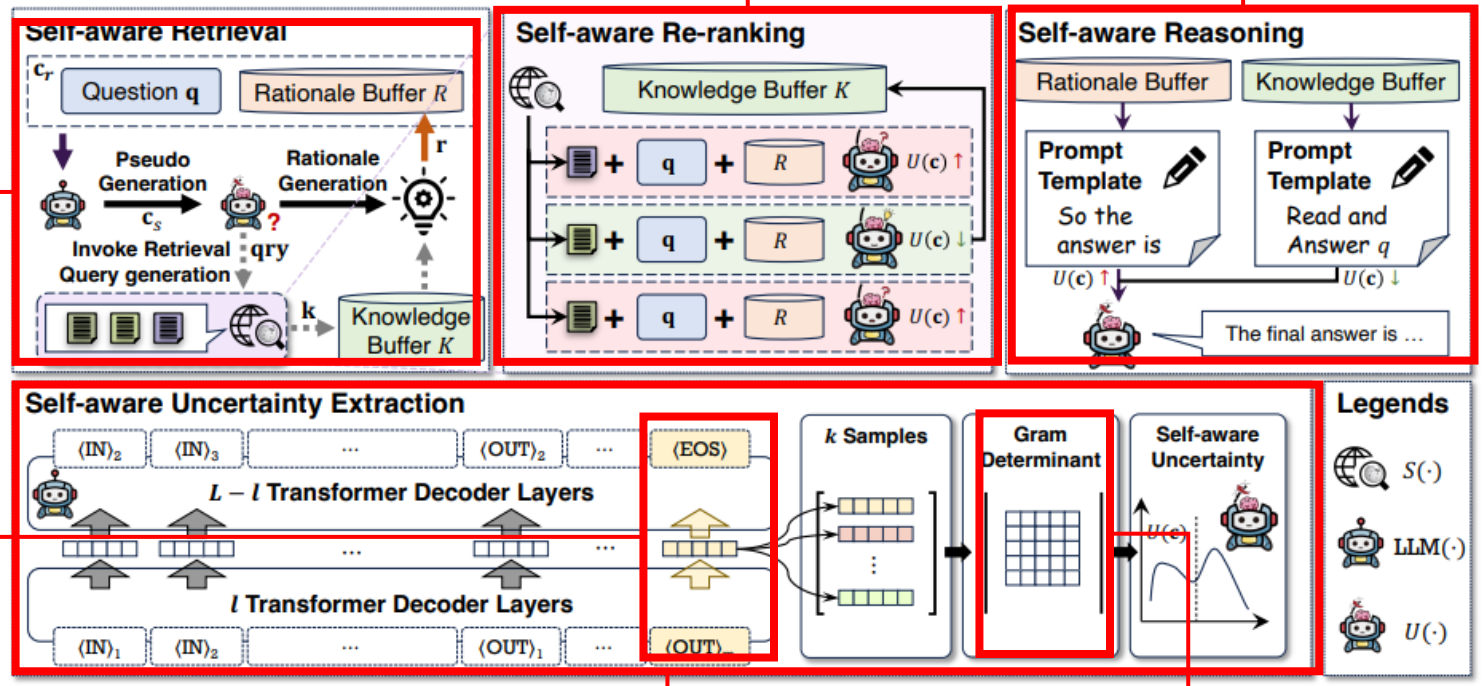
https://arxiv.org/pdf/2406.04271

SEAKR: Self-aware Knowledge Retrieval for Adaptive Retrieval Augmented Generation

RAG 를 구성하다 보면 어떤 케이스에 검색을 해야하고, 어떤 케이스에 어떤 정보를 참조해서 답을 만들어야 할지 케이스가 매우 다양하다, 이러한 분기를 LLM이 모두 스스로 하는 연구 제안

질문의 답변에 지식 검색이 필요한지 여부 판단 및 대략적 답변 생성

질문과 관련 높은 정보의 순위 조정

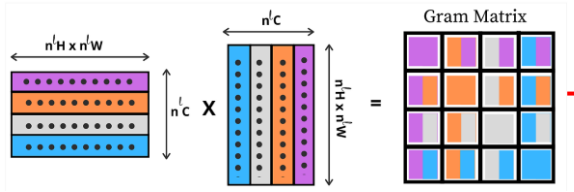


생성 전략의 판단

1) Rationale Buffer로 답변 생성
2) Knowledge Buffer 로 답변 생성
3) 1) + 2)로 답변 생성
4) 1) , 2) 모두 사용하지 않음

모든 Step 에서 판단에 사용
예) Rationale Buffer에 들어갈 답변을 만들지 말지?
예) 만들어진 Rationale 를 사용할지 말지? Knowledge 를 사용할지 말지?

To generate, it outputs o with m tokens ending with an ⟨EOS⟩ token: LLM(c) = ⟨OUT⟩ 1 · · · ⟨OUT⟩m⟨EOS⟩. We aim to extract how certain LLMs are that o is a correct continuation for c. To this end, we follow INSIDE (Chen et al., 2023a) and measure the uncertainty in the hidden space of the ⟨EOS⟩ token



Reranker
보완 모델 적용

MetRAG : Similarity is Not All You Need: Endowing Retrieval-Augmented Generation with Multi–layered Thoughts

유사도 기반의 검색이 좋지 않은 결과를 주는 케이스가 많음. 이러한 단점을 보완하기 위하여 Utility Model 을 제안하며, Retreival 과 Utility Model 의 결과를 종합적으로 사용하여 최적의 결과를 도출 할 수 있다는 연구.

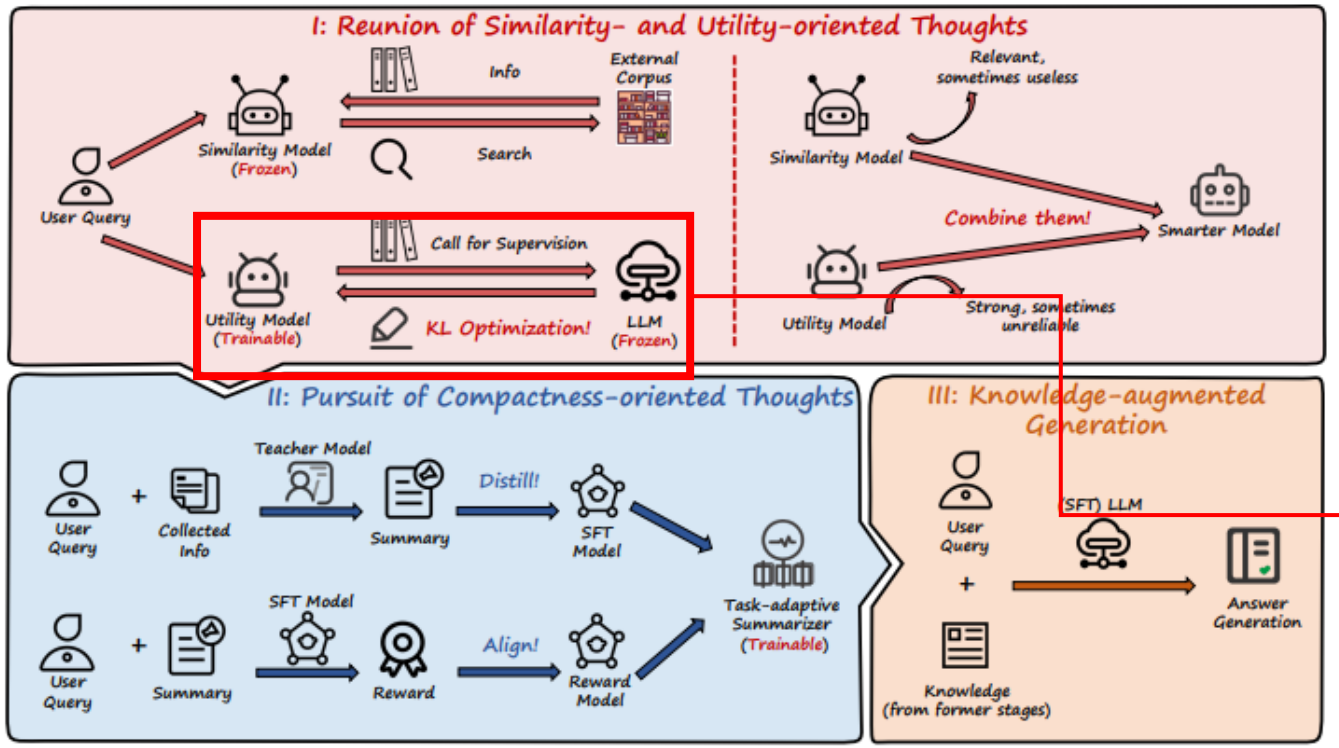


Figure 2: The proposed METRAG framework, where we endow retrieval-augmented generation with multi-layered thoughts from Stage I and II, and utilize the derived knowledge in Stage III for answer generation.

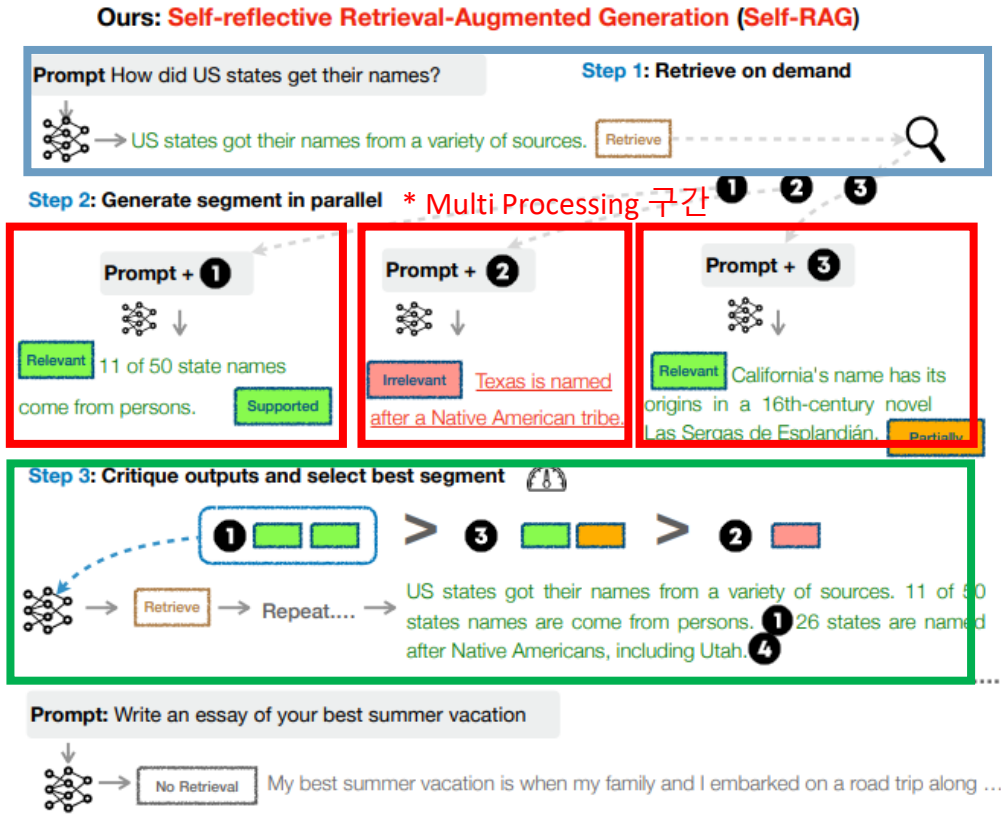
핵심은 Utility 모델인데 정체는 결국 LLM의 우수한 성능을 활용하자는 것
(LLM 을 통해 질문-문서 Pair 가지고 유용성을 판단하도록 하고, 이러한 결과를 훈련 데이터로 별도의 Utility 모델을 만드는 것으로 보임

Inspired by the great success of LLMs in a variety of tasks, we aim to incorporate an LLM for supervision on document utility (In this paper, we define the utility of one document w.r.t. a question by its usefulness in assisting an LLM to answer this question, which is modelled by the normalization of the probability of generating correct answers with a specific LLM.)

https://arxiv.org/pdf/2405.19893

SELF-RAG: LEARNING TO RETRIEVE, GENERATE, AND CRITIQUE THROUGH SELF-REFLECTION

단순 검색➔생성이 아닌 검색 後 관련 여부, 설명력 수준을 판단하고 이를 기반으로 어떤 근거를 기준으로 답변을 생성시 더 좋은 답변이 나오는지 Rank 를 판단하고 최종 답변을 선택하는 형태
(이 연구 또한 LLM 이 모든 Task 에서 성능이 우수하다는 점을 이용하여 Rerank 영역 활용 형태)



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.

Algorithm 1 SELF-RAG Inference

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

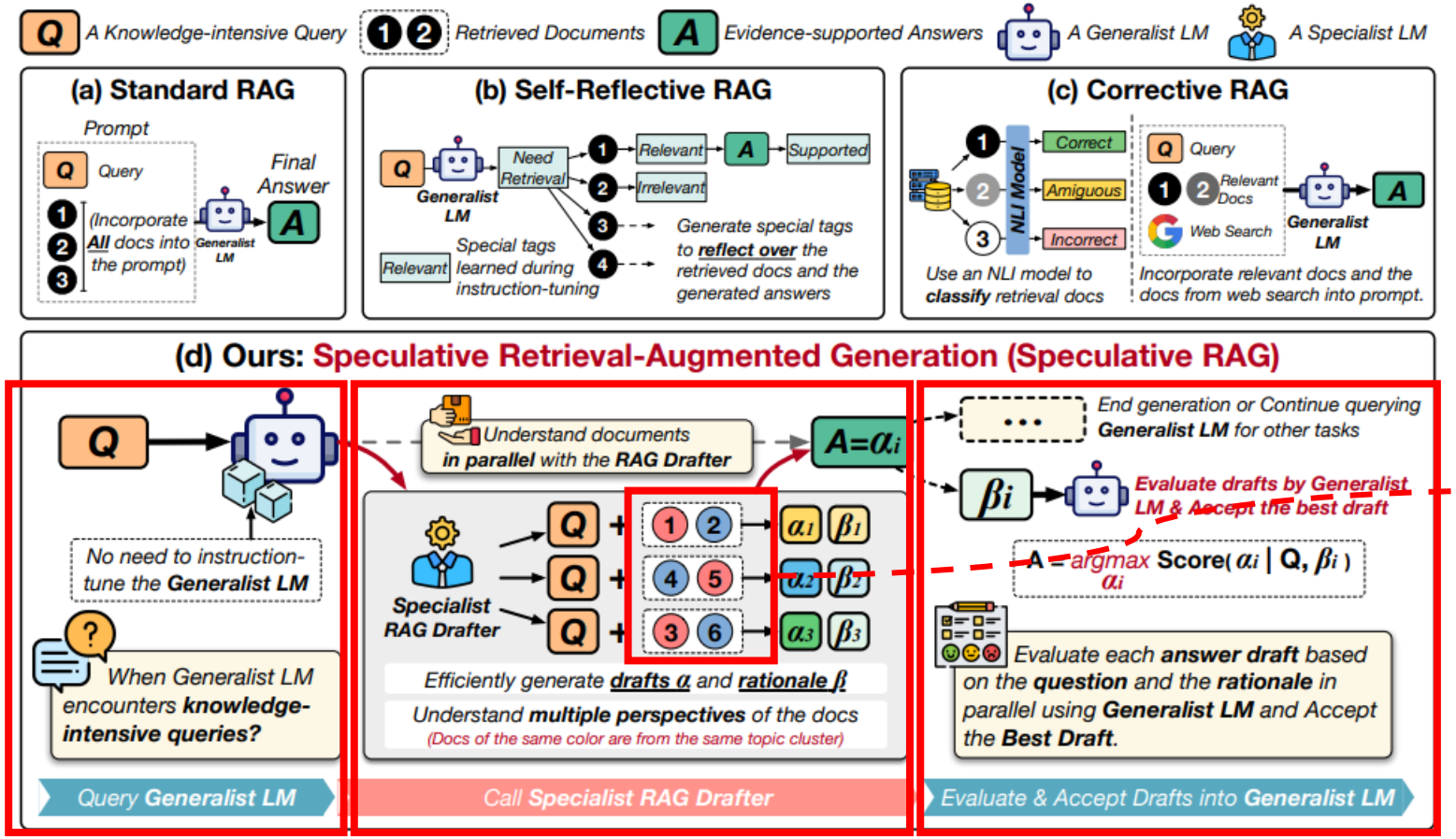
1: **Input:** input prompt x and preceding generation $y_{<t}$, **Output:** next output segment y_t
2: $\mathcal{M}$ predicts Retrieve given $(x, y_{<t})$
3: **if** Retrieve == Yes **then**
4: Retrieve relevant text passages **D** using $\mathcal{R}$ given (x, y_{t-1}) ▷ Retrieve
5: $\mathcal{M}$ predicts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

https://arxiv.org/pdf/2310.11511

Speculative RAG: Enhancing Retrieval Augmented Generation through Drafting

RAG
적용 검토

General LM 보다 소형의 Specialist RAG Drafter 를 적용하여 다양한 형태의 정보를 조합하여 다양한 후보 답변/이유(설명)을 만들고 이를 다시 General LM이 평가하여 Best Draft 를 선택하는 방식



색상이 도메인 클러스터를 의미함 (google 도 연구에서 도메인 지식 그룹을 중요하게 생각하고 조합하여 사용)

[General LM]
지식 집약적인 질문인지 판단

[Special LM (작고 효율적인 모델)]
다양한 정보를 조합하여 Draft 생성
* Multi Processing 구간

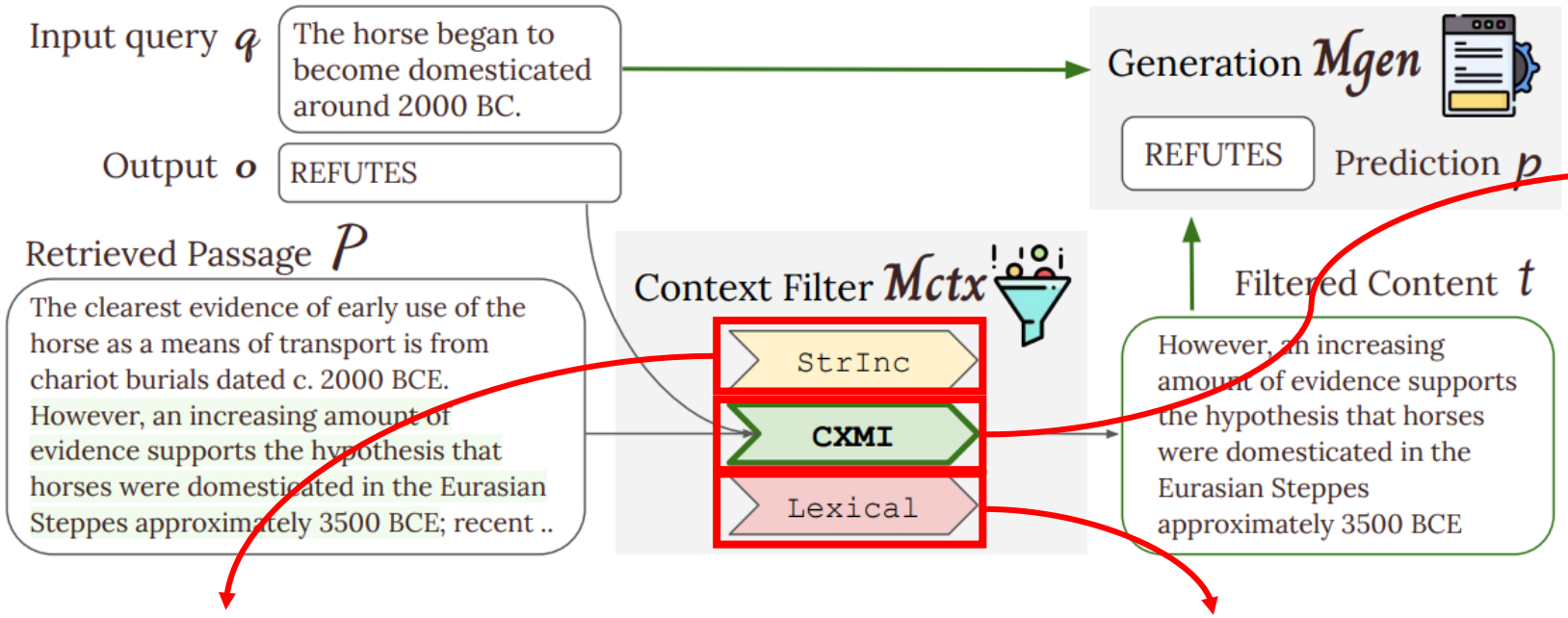
[General LM]
Draft/Rationale 을 평가하여 최적 답변 선택

https://arxiv.org/pdf/2407.08223

MCTX : Learning to Filter Context for Retrieval-Augmented Generation

근거 문서 필터링 등 적용 검토

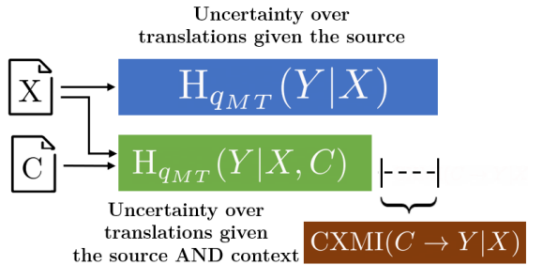
검색 결과의 신뢰도를 향상시키기 위하여 STRIC, CXMI, Lexical 과 같은 3가지 방법을 사용하여 검색 결과를 필터링하고 이렇게 필터링을 통과한 검색 결과만 LLM 생성에 사용



STRINC: whether passages contain the generation output

LEXICAL overlap: how much unigram overlap the content and output has

Conditional cross-mutual information (CXMI): how much more likely the generator is to generate the output when the content is provided



CXMI(C → Y |X) = HqMTA (Y |X) − HqMTC (Y |X, C)

$H_{q_{MT_A}}$ is the entropy

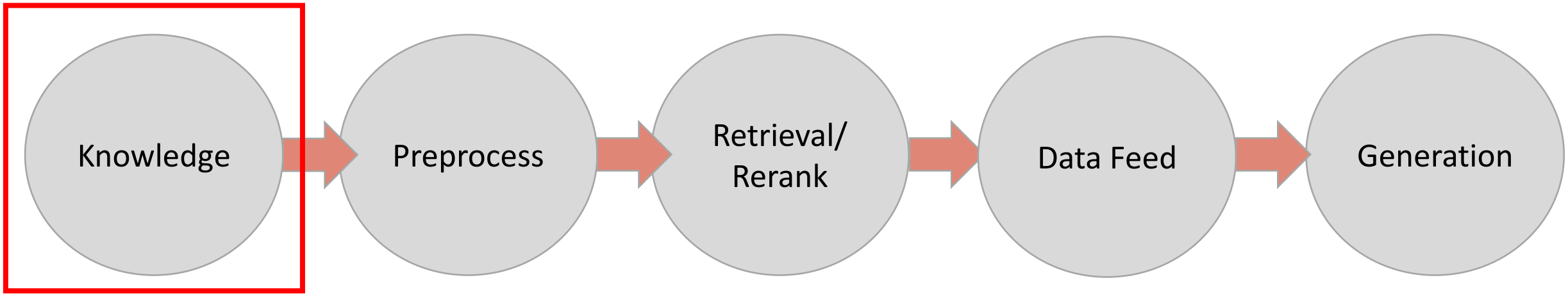
[CXMI 개념 출처 논문]
Measuring and Increasing Context Usage in Context-Aware Machine Translation
https://aclanthology.org/2021.acl-long.505.pdf

https://arxiv.org/pdf/2311.08377

PART#3 Knowledge Management

챕터#3 발표 Concept

정보를 어떻게 관리하고 어떤 단위로 활용하는 것이 전체 RAG 성능의 향상에 도움이 되는지 관련된 연구들을 조사하고 요약하여 설명하고자 한다.



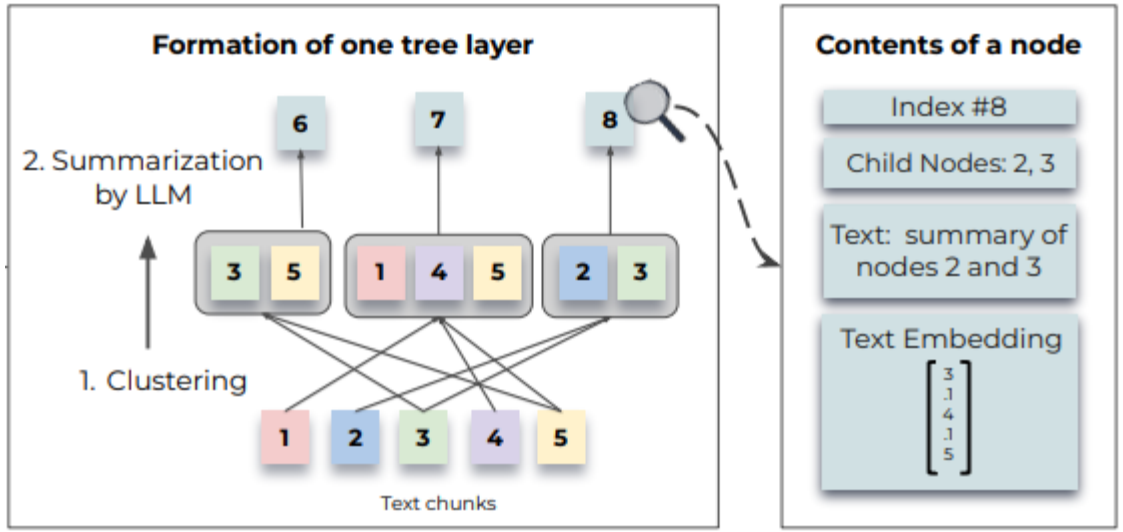
RAPTOR: RECURSIVE ABSTRACTIVE PROCESSING FOR TREE-ORGANIZED RETRIEVAL

Index 구성시 참조

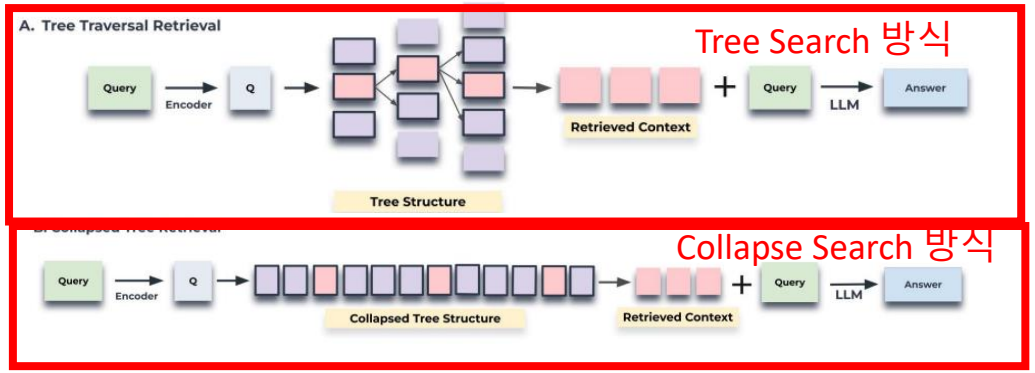
기존의 방법들은 짧은 Chunk 단위로 정보를 검색하여 RAG 에 활용하기 때문에 롱테일 지식을 통합하기가 어렵습니다. RAPTOR 에서는 Tree 형태로 정보를 구조화하여 긴 문서 전반의 정보를 통합하여 추론할 수 있도록 도와줍니다.

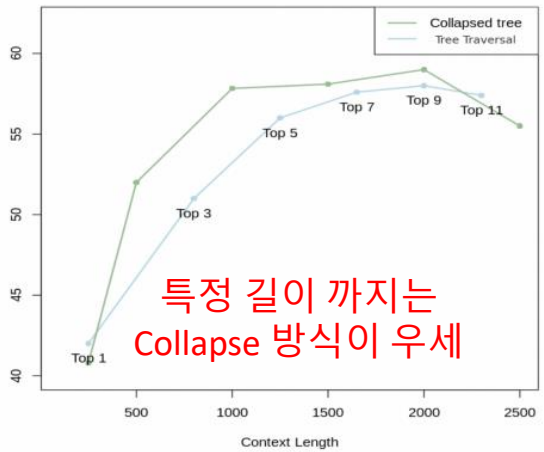
[Tree 정보의 생성]

Tree construction process: RAPTOR recursively clusters chunks of text based on their vector embeddings and generates text summaries of those clusters, constructing a tree from the bottom up.



[검색 시 활용]





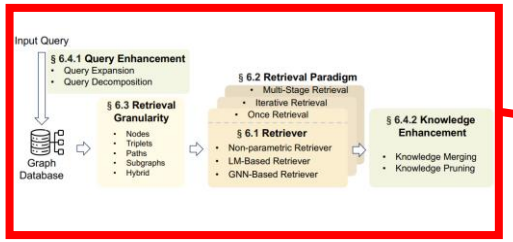
Model	ROUGE
SBERT with RAPTOR	**30.87%**
SBERT without RAPTOR	29.26%
BM25 with RAPTOR	**27.93%**
BM25 without RAPTOR	23.52%
DPR with RAPTOR	**30.94%**
DPR without RAPTOR	29.56%

검색에 RAPTOR 적용 시 성능 향상

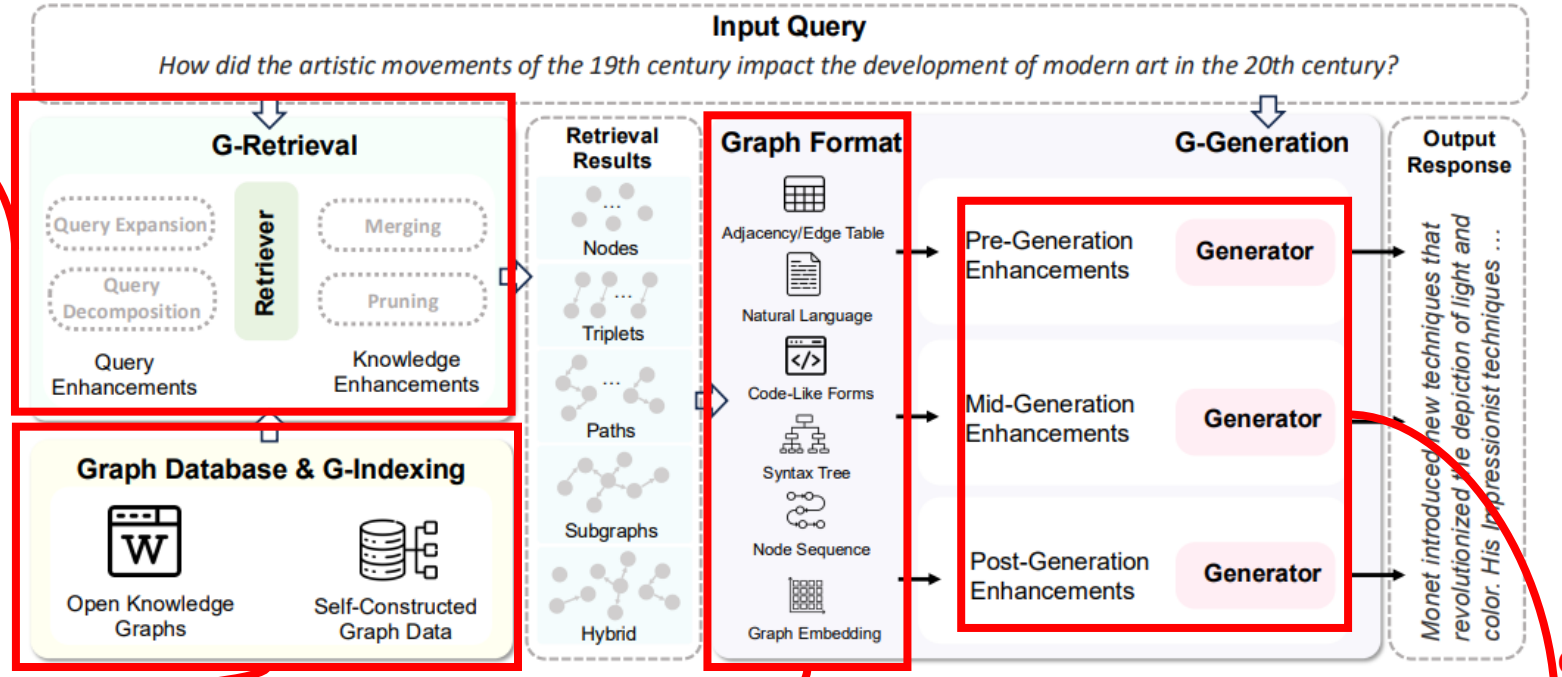
https://arxiv.org/pdf/2401.18059

Graph Retrieval-Augmented Generation: A Survey

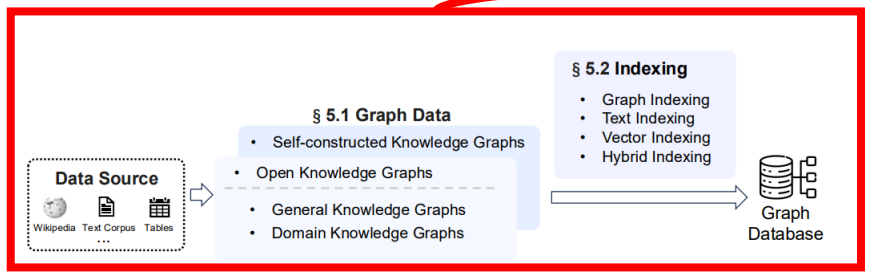
본 논문은 Graph Augmented RAG 관련 Survey 논문으로 전체 RAG 활용 관점에서 Garph정보 생성, Graph 기반 검색, Graph 정보 표현 방법 및 Graph 정보 활용으로 나눠 설명하고 있다.



Graph 기반 검색

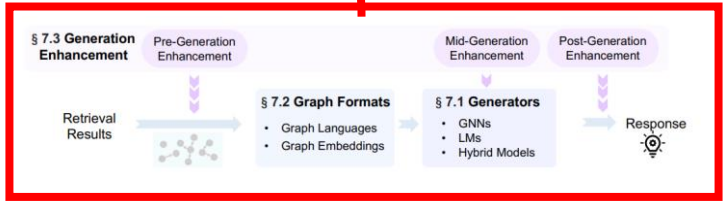


Graph 정보 생성



Graph 정보의 표현

Graph 정보 활용



https://www.arxiv.org/pdf/2408.08921

PART#4 기타 카테고리 연구

RAFT: Adapting Language Model to Domain Specific RAG

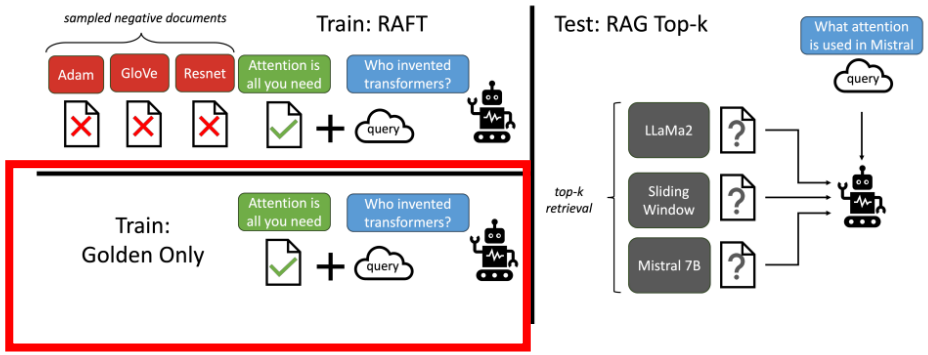
LLM 훈련시 검토

도메인에 특화된 RAG 처리를 위한 LLM 방법론에 대한 연구로 LLM 모델 학습 시 모두 오답인 정보와 질문, 정답 데이터 구성을 일부 포함하는 것이 성능 향상을 보여줬다는 부분과 CoT 가 효과적이었다는 두 가지 내용이 핵심으로 보임

[Golden Only Train]

We demonstrate that our RAG approach trains the model to perform better RAG on the set of documents it is trained on i.e., in-domain. By removing the golden documents in some instances, we are compelling the model to memorize answers instead of deriving them from the context.

P % of data: Q + D∗ + D1 + D2 + . . . + Dk → A∗
(1 − P) % of data: Q + D1 + D2 + . . . + Dk → A∗



[CoT Train]

We also conduct an analysis to evaluate the effectiveness of the Chain-of-Thought approach in enhancing the model's performance. As indicated in Table 2, simply providing the answer to a question may not always be adequate. This approach can lead to a rapid decrease in loss, resulting in the model beginning to overfit. Incorporating a reasoning chain that not only guides the model to the answer but also enriches the model's understanding can improve the overall accuracy and prevent overfitting to concise answers. In our experiments, integrating the Chain-of-Thought significantly enhances training robustness.

	PubMed	HotpotQA	HuggingFace	Torch Hub	TensorFlow
RAFT w.o CoT	68.30	25.62	59.07	**86.56**	83.21
RAFT	**73.30**	**35.28**	**74.00**	84.95	**86.86**

https://arxiv.org/pdf/2403.10131

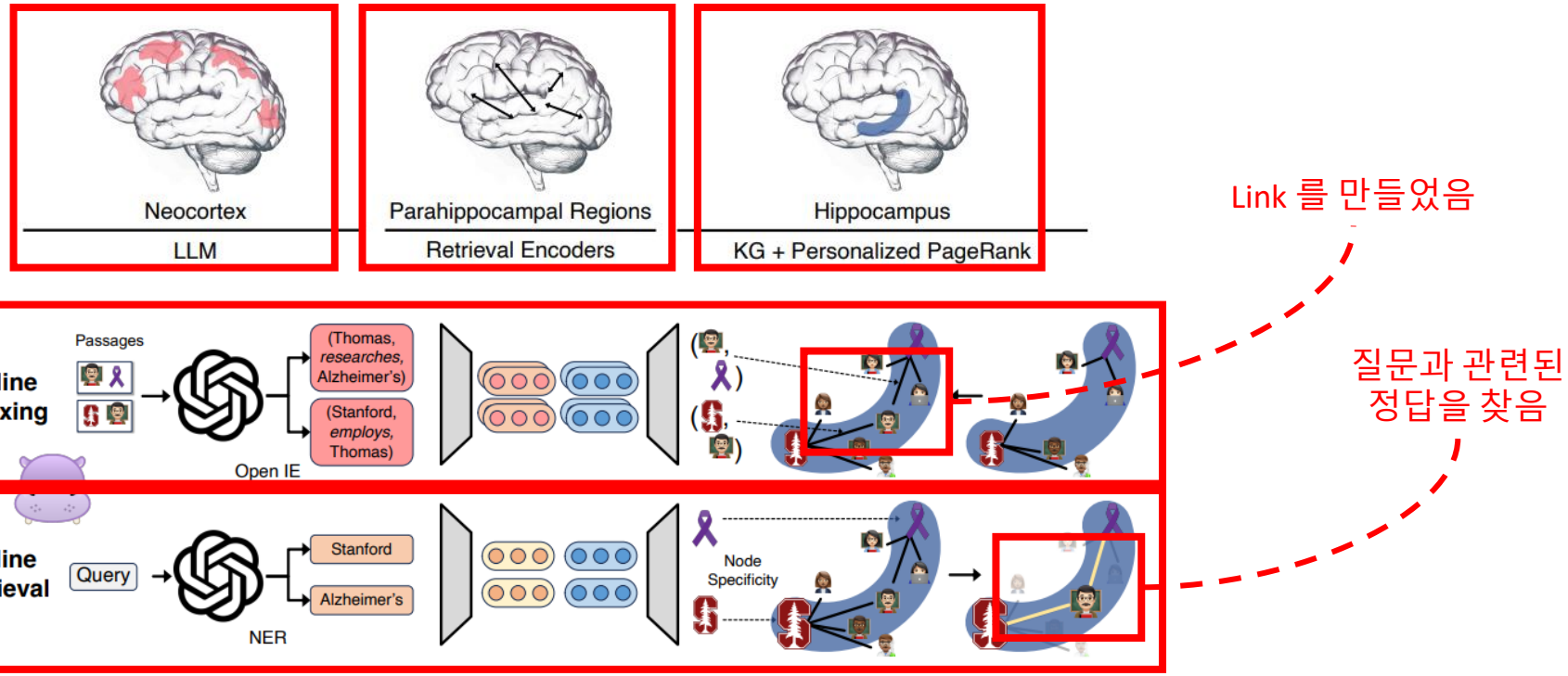
HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models

인간의 장기 기억 시스템에서 영감을 받아 개발된 새로운 증강 생성(RAG) Framework 으로 인간 뇌의 해마와 시피질의 상호작용을 모방하여 설계됨. LLM, Graph DB, Personalized PageRank를 결합하여 인간의 기억 기능을 모방함. (그냥 Graph 에 데이터 넣고, Graph 에서 찾는 것 같은데??)

[신피질]
정보에서 NER 를 추출하여 Triple 형태로 추출

[부해마피질]
Embeding 을 통해서 추출한 단어를 배치하고 거리가 일정 기준 이상 가까우면 Edge 연결

[해마]
Knowledge Graph 에서 Personalized Page Rank 탐색



Link 를 만들었음

정보를 연결하고 만드는 과정

질문과 관련된 정답을 찾음

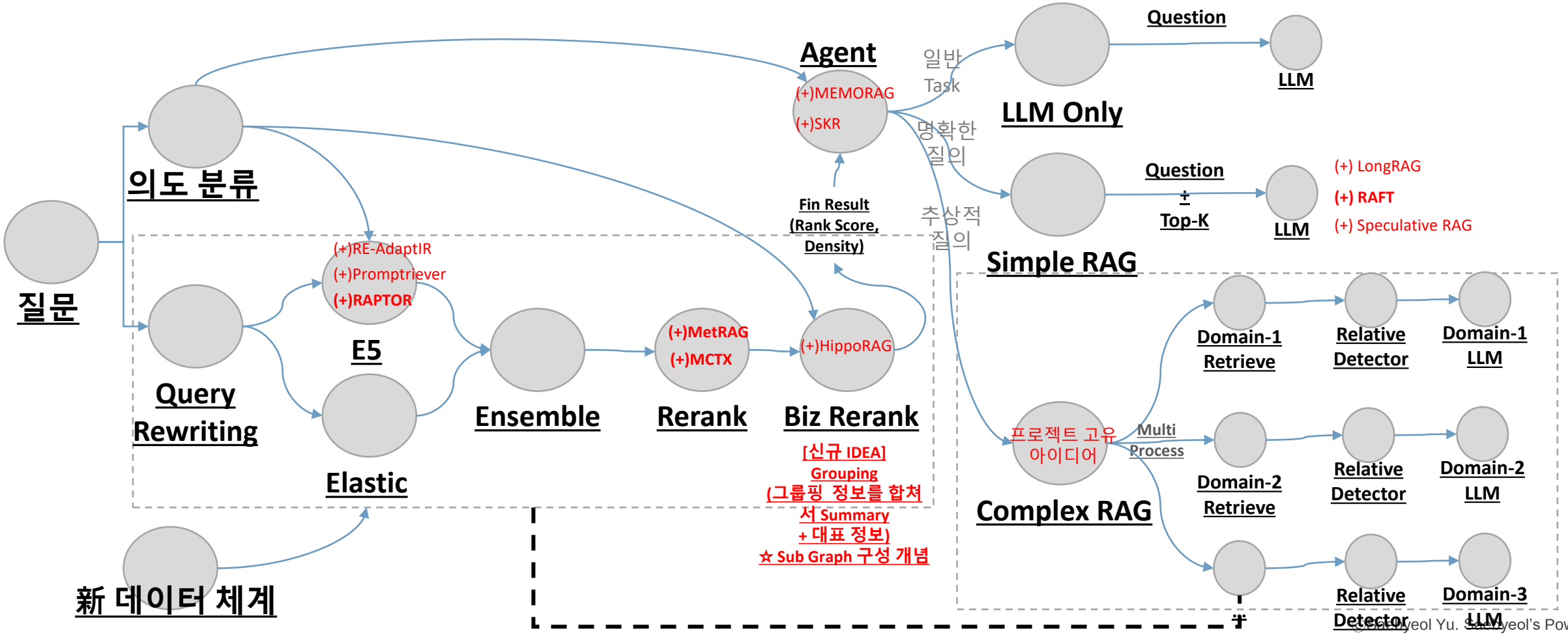
질문을 가지고 답을 찾는 과정

https://arxiv.org/pdf/2405.14831

PART#5 My Design

My Design (~ing)

크게 3가지 Case 를 Agent 가 분류하여 처리하는 형태 ⓐ일반적인 LLM 기반 답변, ⓑ Knowledge Intensive Simple (명확한 근거 Chunk 탐색 시) , ⓒ 추상적인 질의 답변 (다양한 원천으로부터 복합적 답변 필요시) + Global 기업들의 연구 내용 中 반영 가능한 내용 검토 필요



THANK YOU