Retrieval-Augmented Generation for Large Language Models: A Survey

B989068 홍성준
C089069 홍서윤

HONGIK UNIVERSITY
HIGH-PERFORMANCE DATA PROCESSING & ANALYSIS LAB

Contents

0. Abstract

- RAG(Retrieval-Augmented Generation)는 LLM의 학습된 지식과 동적인 외부 데이터베이스의 저장소를 상호 보완적으로 융합시킴으로써 종합적인 지식을 제공한다.
- RAG의 3가지 주요 패러다임인 Naïve RAG, Advanced RAG, Modular RAG를 소개한다.
- RAG의 핵심 구성 요소인 Retrieval(검색), Generation(생성), Augmentation(증강) 구조를 소개한다.
- RAG를 평가하기 위한 metrics와 benchmark를 소개한다.

1. Introduction

- LLM은 훈련 데이터를 벗어나거나, 최신 정보가 필요한 경우와 같이 쿼리가 확장될 때, 사실적으로 잘못된 내용을 생성하는 문제인 "hallucinations(환각)"현상이 일어난다.
- RAG는 LLM이 질문에 답하거나 텍스트를 생성하기 전에 외부 데이터 소스를 쿼리하여 관련 정보를 얻는 초기 검색 단계를 포함한다.
- 이 과정은 응답이 검색된 증거에 근거함을 보장하여 출력의 정확성과 관련성을 크게 향상시킨다.

2. Definition

RAG 정의

- 정보 검색 메커니즘과 In-Context Learning(ICL)을 결합하여 LLM의 성능을 강화한다.
- RAG의 주요 장점은 작업 별 응용 프로그램을 위해 LLM을 다시 교육할 필요가 없다.
- 대신 외부 지식 저장소를 추가하여 입력을 풍부화하고 이를 통해 모델의 출력 정밀도를 높일 수 있다.

RAG Workflow는 3가지 주요단계로 구성된다.

1. 말뭉치는 이산적인 Chunk로 분할되며, 이를 위해 인코더 모델을 활용하여 벡터 인덱스가 구축됩니다.
2. RAG는 쿼리와 index된 Chunk의 벡터 유사성을 기반으로 Chunk를 식별하고 검색한다.
3. 모델은 검색된 Chunk에서 얻은 컨텍스트 정보에 조건을 걸어 응답을 합성한다.

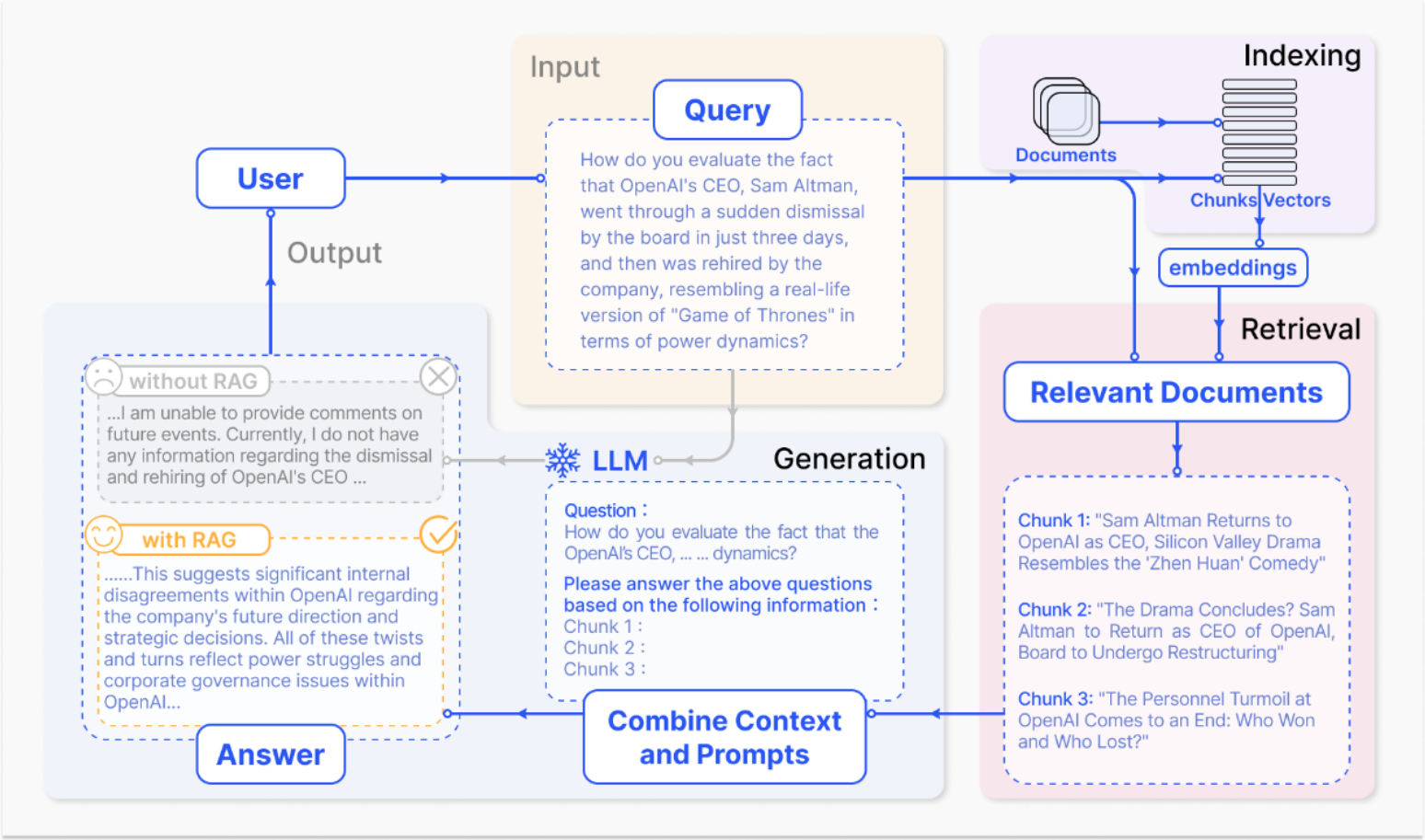
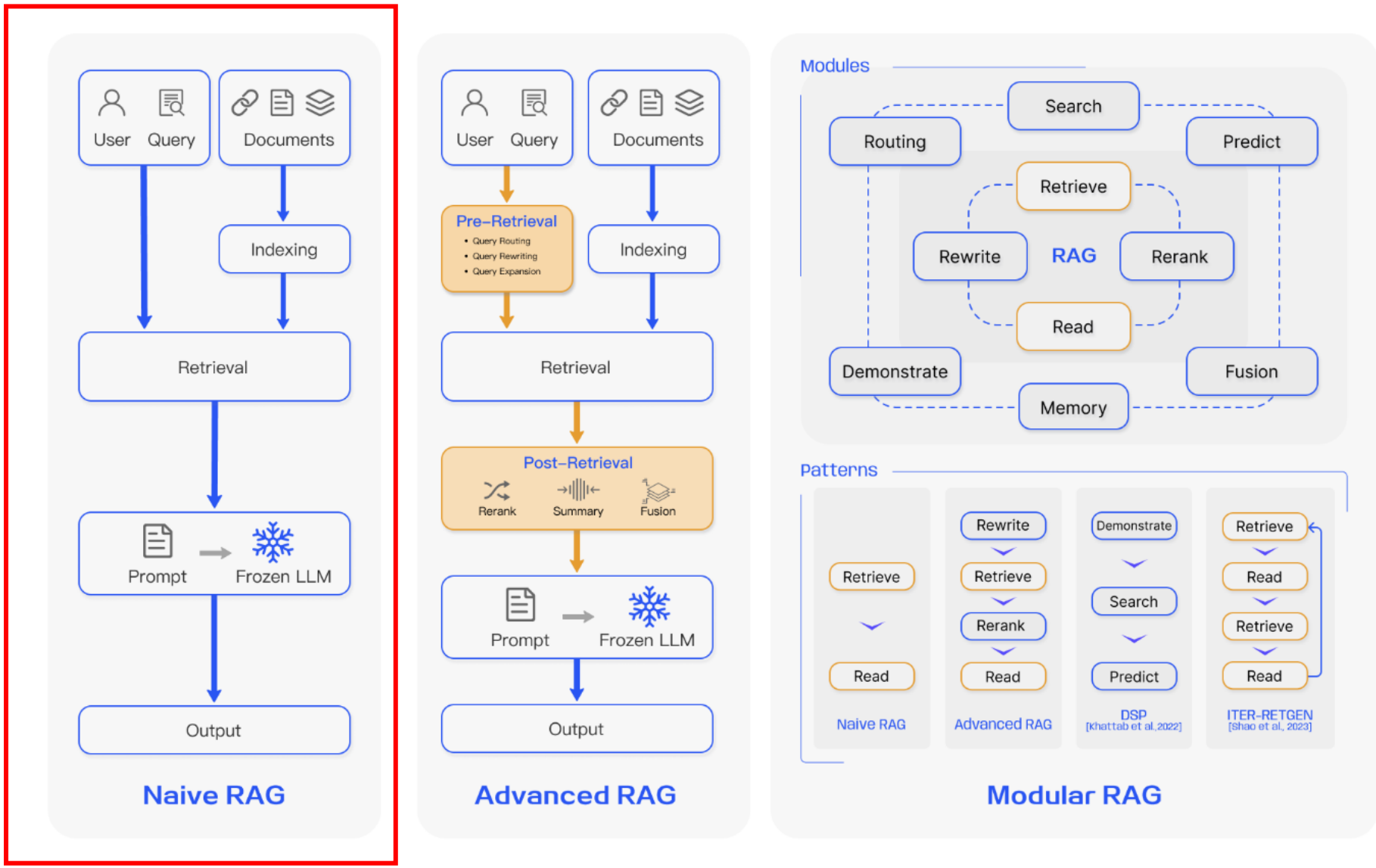


Figure 2: A representative instance of the RAG process applied to question answering

3. Primary of paradigm of RAG



3. Primary of paradigm of RAG

Naïve RAG

Naive RAG는 Indexing(인덱싱)-Retrieval(검색)-Generation(생성) 과정을 따른다.

1. Indexing (인덱싱)

- PDF, HTML, Word, Markdown과 같은 다양한 파일 형식을 표준화된 일반 텍스트로 변환한다.
- 언어 모델의 컨텍스트 제한 내에 맞추기 위해 텍스트를 더 작고 관리하기 쉬운 작은 조각(Chunk)으로 분할한다.
- 검색 단계에서 쿼리 벡터와의 유사성을 비교하기 위해서 Chunk를 임베딩 모델을 통해 벡터 표현으로 변환한다.
- 텍스트 Chunk와 벡터 임베딩을 key-value 쌍으로 저장하는 index가 생성된다.

3. Primary of paradigm of RAG

Naïve RAG

2. Retrieval (검색)

- 사용자 쿼리를 수신하면 시스템은 입력을 벡터 표현으로 변환하기 위해 "1. Indexing (인덱싱)" 단계에서 쓰인 인코딩 모델을 활용한다.
- 쿼리 벡터와 벡터화된 Chunk 간의 유사도 점수를 계산한다.
- 시스템은 쿼리와 가장 큰 유사성을 보이는 상위 K Chunk를 우선적으로 검색한다.

3. Generation (생성)

- 제기된 쿼리와 관련 문서가 결합하여 프롬프트로 생성되고 LLM이 응답을 구성한다.
- 대화하는 방식인 경우, 기존 대화 기록들을 프롬프트에 통합하여 모델이 효과적으로 반응할 수 있도록 한다.

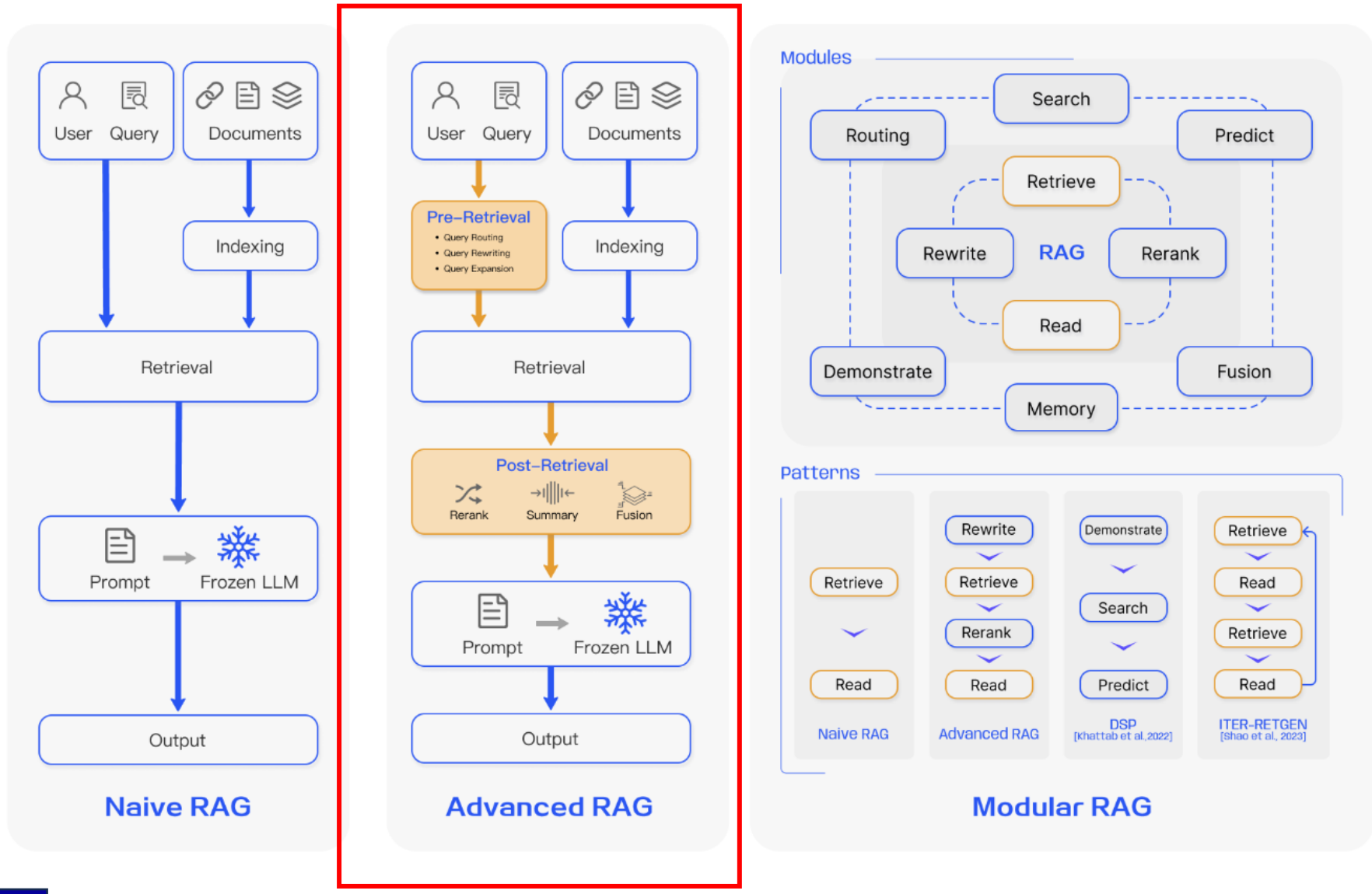
3. Primary of paradigm of RAG

Naïve RAG

Naive RAG는 "검색" , "생성", "증강 " 영역에서 문제점이 발생한다.

- 검색 품질 문제로 인해 정확한 정보를 얻지 못하거나 무관한 답변이 생성될 수 있다.
  - 오래된 정보는 잘못된 결과를 가져올 수 있다.
  - 너무 적은 검색은 연관이 없는 자료를 가져올 수 있다.
- 응답 생성 과정에서는 LLM 모델을 통해 생성된 답변이 제공된 맥락과 관련이 없거나, 독성이나 편향이 있을 수 있다.
- 증강 과정에서는 검색된 내용을 생성 작업과 적절하게 통합하여 일관된 응답을 생성하는 것이 어려울 수 있다. 또한, 중복 및 반복 문제도 발생할 수 있다.
- 증강된 정보에 지나치게 의존하여 단순히 검색된 결과값만 재출력할 가능성이 있다.

3. Primary of paradigm of RAG



3. Primary of paradigm of RAG

Advanced RAG

- Advanced RAG는 Naïve RAG의 부족한 점을 개선하기 위해 개발된 패러다임이다.
- 이는 사전 및 사후 검색 단계를 추가하였다.
- 또한, 슬라이딩 윈도우, 세분화된 분할 및 메타데이터와 같은 기술로 인덱싱 접근 방식을 정교화 하였다.

3. Primary of paradigm of RAG

Advanced RAG - Pre-Retrieval Process

데이터 인덱싱 최적화

- 이는 indexing된 콘텐츠 품질을 향상시키기 위해서 필요하다.
- 이를 위해서 5가지 전략들이 주로 사용된다.

1. 데이터 세분화 강화
    - 텍스트 표준화, 일관성, 사실적 정확성 및 문맥적 풍부함을 향상시키기 위한 절차이다.
    - 관련 없는 내용 제거, 문맥 유지, 오래된 문서 업데이트하는 것을 포함한다.
2. 인덱스 구조 최적화
    - Chunk의 크기를 조정하여 검색에 필요한 관련 정보를 포착하고 여러 가지 다양한 방법으로 데이터에 접근하여 관련 정보를 찾아내며 데이터 간의 관계를 고려하여 관련된 정보를 찾아내는 과정이다.

3. Primary of paradigm of RAG

Advanced RAG - Pre-Retrieval Process

3. 메타데이터 정보 추가

- 각 데이터 Chunk에 날짜, 목적과 같은 참조 메타데이터를 통합하여 검색 결과를 필터링할 때 사용한다.
- 검색 효율성을 향상시키기 위해 장(Chapters) 및 참고 문헌의 하위 항목과 같은 참조 메타데이터를 포함한다.

4. 정렬 최적화

- 문서 간의 내용 불일치와 연결되지 않은 부분을 해결하기 위해 '가상 질문 ' 을 도입한다.

5. 혼합 검색

- 키워드 기반 검색, 의미 검색, 벡터 검색과 같은 다양한 검색 기술을 지능적으로 결합하여 검색 전략을 강화한다.

3. Primary of paradigm of RAG

Advanced RAG - Retrieval Process

- 검색 단계에서는 쿼리와 Chunk간 유사성을 계산하여 적절한 문맥을 찾는 것이 주요 관심사이다.
- 이 과정에서 임베딩 모델은 중요한 역할을 한다.
- Advanced RAG는 Fine-tuning Embedding, Dynamic Embedding model이 사용 될 수 있다.

3. Primary of paradigm of RAG

Advanced RAG - Retrieval

1. Fine-Tuning Embedding

- 특정 전문 도메인 부분에서 검색 관련성을 향상시키기 위해 임베딩 모델을 사용자 정의한다.
- 문서를 Chunk로 분할한 다음, 각 Chunk에 대해 LLM 모델을 사용하여 해당 문맥을 이해하고 질문을 생성한다. 생성된 질문과 해당 문단을 쌍으로 구성하여 학습 데이터를 형성한다.

2. Dynamic Embedding

- Dynamic Embedding이란 단어나 문장을 표현하는 벡터가 문맥에 따라 달라지는 임베딩 기술을 뜻한다.

3. Primary of paradigm of RAG

Advanced RAG - Post-Retrieval Process

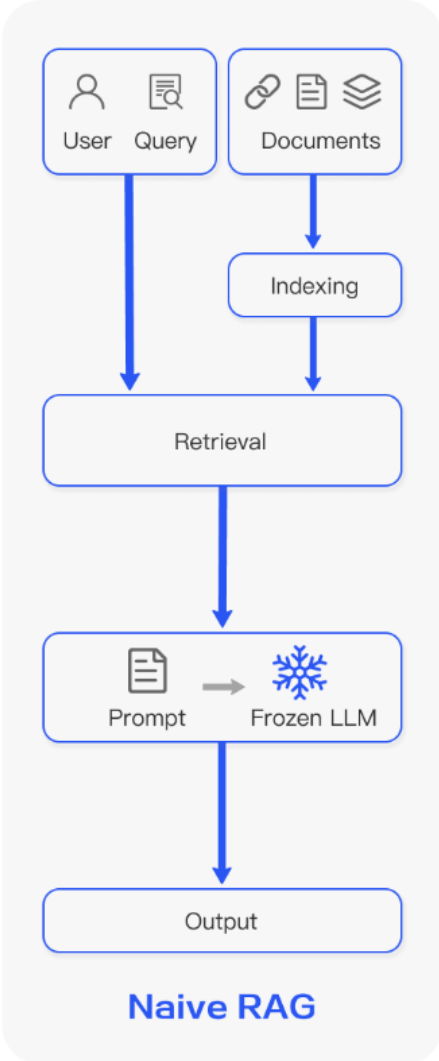
- 데이터베이스에서 가치 있는 문맥을 검색한 후, 쿼리와 병합하여 LLM에 입력하는 과정이다.
- 검색 결과로 나온 모든 문서들을 한꺼번에 LLM에 제공하는 것은 비효율적이며 LLM의 context window 크기를 초과할 수도 있다.
- 이러한 문제를 해결하기 위해서 검색된 문서들의 추가적인 후처리가 필요하다.
- 추가적인 후 처리 방법은 Re-Ranking 방법과, Prompt Compression 방법이 있다.

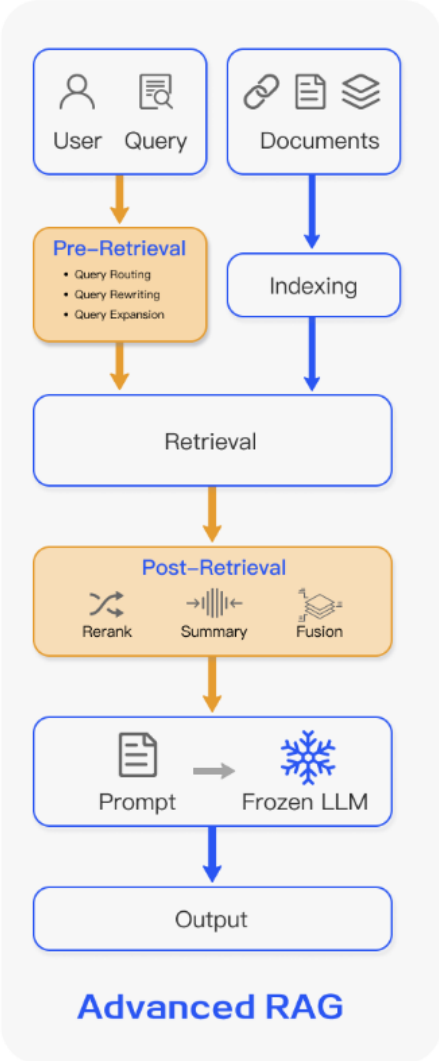
3. Primary of paradigm of RAG

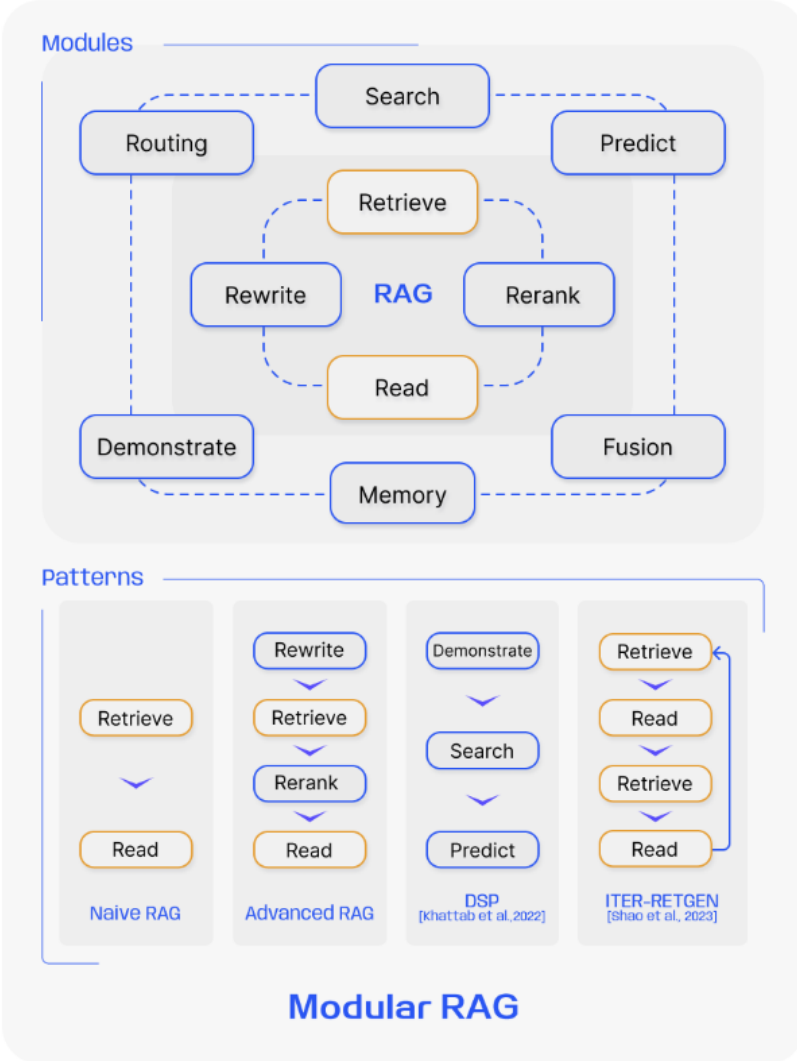
Advanced RAG - Post-Retrieval Process

1. Re-Ranking
   - 검색된 정보를 다시 순위 매기는 것으로, 가장 관련성 높은 내용을 프롬프트의 앞부분으로 이동시키는 전략이다.
   - 이 개념은 LlamaIndex, LangChain, HayStack와 같은 프레임워크에서 구현되었다.
2. Prompt Compression(압축)
   - 검색된 문서에서 발생하는 noise는 RAG 성능에 부정적인 영향을 끼친다.
   - 사후 검색 단계에서 관련 없는 문맥을 압축하고 중요한 단락을 강조하면서 전체 문맥 길이를 줄이는 것이 중요하다.
   - 프롬프트 압축하는 데에는 요약, 중요한 부분 강조, 필터링 등과 같은 여러 가지 방법이 존재한다.

3. Primary of paradigm of RAG







3. Primary of paradigm of RAG

Modular RAG

- Modular RAG는 기존의 Naïve RAG 프레임워크와 달리 다양한 모듈과 기능을 통합하여 더 큰 다양성과 유연성을 제공한다.
  - 유사성 검색을 위한 검색 모듈 통합
  - 검색(Retriever)에서의 fine-tuning 접근 방식 적용과 같은 기능 모듈 통합
- Modular RAG 구조는 여러 모듈간의 직렬화된 파이프라인을 구성하거나, end-to-end 훈련 접근 방식을 허용한다.

3. Primary of paradigm of RAG

Modular RAG – New Modules

- 기존의 RAG 프레임워크에서 추가되거나 확장된 새로운 기능을 제공하는 모듈을 일컫는다.
- 1. Search
  - 이전 RAG (Naïve RAG, Advanced RAG)의 유사성 검색과 달리, 특정한 상황에 맞게 조정되며 추가적인 데이터에 직접적으로 검색을 수행할 수 있도록 설계되었다.
- 2. Memory
  - LLM의 기억 능력을 이용하여 정보를 검색한다.
  - 이 모듈은 현재 입력에 맞는 유사한 메모리를 찾아내고 이를 활용하여 검색 작업을 수행한다.

3. Primary of paradigm of RAG

Modular RAG – New Modules

- 3. Fusion
  - Fusion 모듈은 다양한 관점으로 사용자 쿼리를 확장하는 multi-query 접근법을 통해 검색 시스템을 강화하는 모듈이다.
  - 이는 원본 및 확장된 쿼리 모두에 대해 병렬 벡터 검색을 수행하고 결과를 최적화하기 위해 지능적으로 순위를 다시 지정하고, 최상의 결과를 새로운 쿼리와 연결하는 작업이 포함한다.
- 4. Routing
  - 사용자 쿼리에 대한 적합한 데이터 저장소를 선택하기 위한 모듈이다.
- 5. Predict
  - 검색된 정보에서의 중복 및 noise 문제를 해결하기 위해 데이터 소스에서 직접 검색하는 대신 LLM을 활용하여 필요한 문맥을 생성한다.

3. Primary of paradigm of RAG

Modular RAG – New Modules

6. Task Adapter

- 다양한 downstream task에 적응할 수 있도록 조정하는 모듈이다.
- 이는 UPRISE 방법을 사용하거나, PROMPTAGATOR 방법을 사용하기도 한다.
- UPRISE : zero-shot 작업에 대해서 적절한 프롬프트를 찾기 위해 사전에 구축된 데이터에서 자동으로 관련 정보를 추출하고 제공한다.
- PROMPTAGATOR : LLM을 few-shot 쿼리 생성기로 활용하고 생성된 데이터를 기반으로 task별 검색기를 생성한다.

3. Primary of paradigm of RAG

Modular RAG – New Patterns

- Modular RAG는 RAG 프로세스 내에서 특정 문제 맥락에 맞게 조정할 수 있다.
- 기존의 모듈 추가 또는 교체 (Adding or Replacing Modules)
  - 검색-읽기(RR; Retrieval-Read) 구조를 유지하면서 특정 기능을 향상시키기 위해 추가 모듈을 도입한다.
  - 예를 들어, 재작성-검색-읽기(RRR; Rewrite-Retrieve-Read) 프로세스를 통해 검색 쿼리를 조작하고, 읽기 모듈의 하위 작업 성능을 향상시킬 수 있다.
- 모듈 간 조직적 흐름 조정(Adjusting the Flow between Modules)
  - 언어 모델과 검색 모델 간의 상호작용을 강화하기 위해 모듈 간 조직적 흐름을 조정한다. 이는 특정 문제 맥락에 기반하여 모듈 내에서의 대체 또는 재구성을 가능하게 한다.

3. Primary of paradigm of RAG

Modular RAG - Optimizing the RAG Pipeline

- RAG 파이프라인 최적화는 RAG 시스템의 효율성과 정보 품질을 향상시키는 것을 목표로 한다.
- 주요한 RAG 파이프라인 최적화 방법론들은 다음과 같다.
- Hybrid Search, Recursive Retrieval and Query Engine, StepBack-prompt, Subqueries, HyDE(Hypothetical Document Embeddings)

3. Primary of paradigm of RAG

Modular RAG - Optimizing the RAG Pipeline

- Hybrid Search : 키워드 기반 검색, 의미론적 검색, 벡터 검색 등과 같은 다양한 기술을 지능적으로 혼합하여 검색하는 방법으로, RAG 시스템이 다양한 질의 유형과 정보 요구에 적응할 수 있다.
- Recursive Retrieval and Query Engine : 초기 검색 단계에서 더 작은 Chunk를 획득하여 주요 의미를 포착한 후, 후반 단계에서 더 많은 맥락적 정보를 가진 더 큰 Chunk를 언어 모델에 제공하는 방법이다.
- StepBack-prompt : LLM이 특정 사례에서 벗어나 일반적인 개념이나 원리에 대해 추론하기 위한 방법이다.
- Subqueries :서브쿼리는 다양한 쿼리 전략을 다양한 시나리오에 적용할 수 있으며, LlamaIndex와 같은 프레임워크에서 제공하는 쿼리 엔진을 사용할 수 있다.
- HyDE(Hypothetical Document Embeddings) : LLM을 사용하여 가상의 문서(= LLM의 답변)를 생성하고, 이를 임베딩한 후, 이 임베딩을 사용하여 실제 문서를 검색하는 방법이다. 쿼리에서 임베딩 유사성을 찾는 대신 한 답변에서 다른 답변으로의 임베딩 유사성에 중점을 둔다.

4. Core Components Of RAG

Retrieval

Q1. 어떻게 정확한 의미 표현을 달성할 수 있나?

A. 정확한 의미론적 표현 추출 (Enhancing Semantic Representations)

- Chunk optimization
- Fine-tuning Embedding Models

Q2. 쿼리와 문서의 의미 공간을 어떻게 정렬할 수 있나?

A. 사용자의 질의(Query)와 검색 대상이 되는 문서들의 의미론적 공간 일치

- Query Rewriting
- Embedding Transformation

Q3. 검색 모듈의 출력을 어떻게 LLM의 선호도와 일치 시킬 수 있나?

A. 검색기 출력과 LLM의 선호도 일치

- Fine-tuning Retrievers
- Adapters

4. Core Components Of RAG

Retrieval – 어떻게 정확한 의미 표현을 달성할 수 있나?

- 1. Chunk Optimization
  - 외부 문서 처리의 첫 단계는 입력된 문서를 적당한 크기의 조각(chunk)으로 분할하는 것이다. 이렇게 분할된 조각은 모두 같은 크기의 벡터로 임베딩되므로 Chunk의 크기를 적절히 조절해야 한다. 이를 통해 적절한 크기의 Chunk로 외부 문서를 분할하는 경우 검색 결과의 정확성과 관련도가 크게 향상될 수 있다.
- 2. Fine-Tuning Embedding Models
  - 적절한 청크 크기를 설정한 후 의미론적 공간에 청크와 질의를 임베딩하여 Fine-Tuning을 통해 정확한 의미 표현을 얻을 수 있습니다. 이러한 미세조정을 통해 사용자 Query와 콘텐츠 관련성과의 관계를 더 잘 이해할 수 있도록 돕는다.
  - 또한 특정 도메인에 적합한 데이터셋을 활용하여 모델을 미세 조정할 수 있다.

4. Core Components Of RAG

Retrieval – 쿼리와 문서의 의미 공간을 어떻게 정렬할 수 있나?

- RAG 애플리케이션에서 어떠한 검색기는 사용자 질의와 문서들을 인코딩할 때 동일한 임베딩 모델을 사용하기도 하지만, 사용자 질의와 문서들을 인코딩할 때 서로 다른 임베딩 모델을 사용하는 Retrieval도 있다. 또한, 사용자의 질의가 충분한 정보를 포함하지 않는 경우에는 사용자 질의의 의미론적 표현이 검색에 부적절할 수 있다.
- 검색 품질을 높이기 위해 사용자의 질의와 문서의 의미론적 공간을 일치시키는 다음 2가지 방법들을 소개한다 :
- 1. Query Rewriting : Query와 문서 간의 의미론적 일치를 향상시키기 위해 질의를 변형하는 방법이다.
- 2. Embedding Transformation : 먼저 임베딩 모델을 사용하여 Query와 문서를 벡터로 변환하고, 이들 간의 의미론적 유사성을 높이는 방식으로 검색 결과의 관련성(releavance)과 Query에 대한 응답의 정확성을 향상시킨다.

4. Core Components Of RAG

Retrieval – 검색 모듈의 출력을 어떻게 LLM의 선호도와 일치 시킬 수 있나?

- 위 방법들을 사용하여 검색 적중률(retrieval hit rate)을 높이더라도 검색된 결과 문서들이 LLM이 필요로 하는 문서가 아닐 수 있다. 이러한 경우 RAG의 최종적인 성능 효과가 개선되지 않기 때문에, 검색기의 출력(output)과 LLM의 선호도(preferences)를 일치시키는 방법이 있다.
- 1. Fine-tuning Retrievers : LLM에서의 피드백 신호를 활용하여 검색 모델을 세밀하게 조정한다. 이러한 접근 방식은 검색기와 LLM 간의 상호 작용을 개선하여 검색 성능을 향상시키고 사용자 조회에 더 정확한 응답을 제공하도록 목표로 한다.
- 2. Adaptor : Fine-tuning 모델에서 나타나는 API를 통한 기능 통합이나 제한된 로컬 계산 리소스로 인한 제약 등과 같은 문제를 외부 어댑터를 통합하여 align을 돕는다.

4. Core Components Of RAG

Generation

- Generation은 검색된 결과를 활용하여 사용자에게 제공할 답변을 생성한다.
- 검색된 정보를 효과적으로 활용하여 정확하고 관련성 높은 답변을 생성하기 위해, 정보를 압축하고 순위를 다시 매기는 후처리 과정과 입력 데이터에 적응하는 최적화 과정을 수행하고 있다.
- 검색 결과의 후처리는 검색기가 검색한 정보들을 대규모 문서 데이터베이스를 사용하여 추가적인 처리나 필터링, 최적화하는 과정으로, 검색 품질을 향상시키는 것이 목적이다.

  생성기의 최적화는 생성된 텍스트가 자연스럽고 검색된 문서를 효과적으로 활용하여 사용자의 쿼리 요구를 더 잘 충족시키도록 보장하는 데 목표가 있다.

4. Core Components Of RAG

Generation - Post-retrieval with Frozen LLM

- 포괄적인 내부 지식을 활용하는 LLM에는 context length limitation과 중복 정보에 대한 취약점과 같은 문제가 여전히 존재한다.
- 검색 후처리는 검색기에 의해 검색된 관련 정보를 처리, 필터링 또는 최적화하는 것을 포함한다.
- 검색 결과의 품질을 향상시켜 사용자의 요구 사항이나 후속 작업과 보다 밀접하게 일치하도록 한다.
- 1. Information Compression : 정보 압축을 하여 Noise를 줄이고, 맥락 길이 제한을 해결하고, 생성 효과를 향상시킨다.
- 2. Re-ranking : 결과 문서들을 재배열하여 가장 관련성 높은 항목들을 상위에 배치함으로써 검색 결과의 수를 줄이는 것이다.

4. Core Components Of RAG

Generation – Fine Tuning LLM for RAG

- General Optimization Process 은 생성된 텍스트가 자연스럽고 검색된 문서를 효과적으로 활용하여 사용자의 쿼리 요구를 더 잘 충족시키도록 보장하는 것을 목표로 한다.
- 일반적으로, RAG의 생성기에 대한 Fine-Tuning 방법은 검색된 정보를 활용하여 생성된 텍스트의 품질과 유용성을 향상시키는 것에 중점을 둔다.
- 대조 학습은 긍정적 예시와 부정적 예시를 모두 제시하여 학습 데이터의 패턴만 따르지 않게 하고, 다양한 유형의 데이터를 사용하여 일반화 능력이 향상된다.

4. Core Components Of RAG

Augmentation

- 증강 단계(the stage of augmentation)
  - Pre-training Stage, Fine-Tuning Stage, Inference Stage
- 데이터 소스의 증강(augmentation data sources)
  - 비정형 데이터를 사용한 증강, 정형 데이터를 사용한 증강, LLM이 생성한 콘텐츠를 사용한 증강
- 증강 절차(the process of augmentation)
  - 반복적 검색(Iterative Retrieval), 적응형 검색(Adaptive Retrieval)

4. Core Components Of RAG

Augmentation – 증강 적용 단계

- 검색 증강은 Pre-Training, Fine-Tuning 및 Inference과 같은 단계에서 적용할 수 있습니다.
- 1. Pre-Training 단계에서의 Augmentation 적용
  - 증강된 사전훈련 모델은 텍스트 생성 품질 및 도메인별 모델의 개발을 용이하게 할 수 있다.
- 2. Fine-Tuning 단계에서의 Augmentation 적용
  - Retriever의 Fine-Tuning을 통해 의미 표현의 품질을 향상시킨다.
  - Generator의 Fine-Tuning을 통해 출력을 맞춤화 할 수 있다.
- 3. Inference 단계에서의 Augmentation 적용
  - 추가적인 훈련 없이 사전 훈련된 모델의 기능을 활용하는 경량이고 비용 효율적인 대안을 제공할 수 있다.

4. Core Components Of RAG

Augmentation – Data Source

- 다양한 데이터 소스는 지식의 서로 다른 세분화 정도(granularity)와 측면(dimension)을 제공하며, 증강 시에는 서로 다른 처리 방식이 필요합니다. 이러한 데이터 소스는 크게 비정형 데이터, 정형 데이터, 그리고 LLM이 생성한 콘텐츠의 3가지 범주로 나눈다.
- 비정형 데이터를 사용한 증강은 더 작은 단위를 사용하는 경우 드물게 나타나는 패턴이나 도메인의 시나리오를 더 처리 할 수 있다.
- 정형 데이터를 사용한 증강은 검증된 고품질 Data Context를 제공하여 모델이 잘못된 정보를 생성할 가능성을 줄일 수 있다.
- LLM이 생성한 콘텐츠를 사용한 증강은 LLM 자체가 생성한 콘텐츠를 검색 목적으로 사용하는 새로운 접근방식이다.

4. Core Components Of RAG

Augmentation – Process

- RAG 시스템은 한 번의 검색과 생성 절차를 거치지만, 중복 문서가 검색되는 경우에는 핵심 정보가 모호하게 표현되거나 정답과는 반대되는 정보를 포함하는 등의 생성 절차에 부정적인 영향을 미칠 수 있다.
- 또한, 이렇게 한 번의 검색과 생성 절차만 수행하는 경우에는 다단계 추론(multi-step reasoning)이 필요한 복잡한 질의에 대해서는 제대로 대답하지 못할 수 있다.
- 이러한 문제를 해결하기 위해 반복적 검색(iterative retrieval), 재귀적 검색(recursive retrieval), 적응형 검색(adaptive retrieval)과 같은 방식으로 검색 절차를 최적화한다.

4. Core Components Of RAG

Augmentation – 반복적 검색(iterative retrieval)

- 반복적 검색(Iterative Retrieval)은 사용자의 질의(query)와 모델이 생성한 텍스트를 기반으로 추가적인 문서를 수집하는 과정이다. 이 방법은 검색기과 생성기 사이의 상호작용을 강화하고, 보다 정확하고 관련성 높은 답변 생성을 가능하게 한다.
- 재귀적 검색(recursive retrieval)은 하나의 검색 작업 결과를 다음 검색의 입력으로 사용하는 과정이다. 이 과정은 이전 검색 결과를 기반으로 검색 쿼리를 반복적으로 개선하는 것을 의미한다.
- 적응형 검색(adaptive retrieval)은 RAG 시스템에서 검색 타이밍과 내용을 능동적으로 판단하고 최적화하는 접근 방식이다. 이는 기존의 수동적 검색 방법과 달리, 언어 모델이 검색 과정을 더 효율적으로 관리하고 관련성 높은 정보를 획득하도록 한다.

4. Core Components Of RAG

RAG vs Fine-Tuning

- RAG은 모델에 맞춤 정보 검색용 교과서를 제공하는 것과 같으며, 특정 쿼리에 완벽하게 적합하다.
- Fine-Tuning은 시간이 지남에 따라 지식을 내재화 하는 학생과 유사하며, 특정 구조, 스타일 또는 형식을 복제하는 데 더 적합하다.
- Fine-Tuning은 기본 모델 지식을 강화하고 출력을 조정하며 복잡한 지침을 가르치는 등 모델의 성능과 효율성을 향상시킬 수 있다. 그러나 새로운 지식을 통합하거나 새로운 사용 사례를 신속하게 반복하는 데는 그다지 좋지 않다.
- RAG와 Fine-Tuning 두 가지 방법은 상호 배타적이지 않으며 상호 보완적일 수 있으며, 모델의 능력을 서로 다른 수준에서 향상시킬 수 있습니다.

4. Core Components Of RAG

RAG vs Fine-Tuning

Feature Comparison	RAG	Fine-Tuning
Knowledge Updates (지식 업데이트)	직접 검색 지식베이스를 업데이트하여 정보가 현재 유지되도록 하고 자주 재교육이 필요하지 않아 동적 데이터 환경에 적합하다.	정적 데이터를 저장하여 지식 및 데이터 업데이트를 위해 재교육이 필요하다.
External Knowledge (외부 지식)	외부 리소스를 효과적으로 활용하며 특히 문서나 다른 구조화/비구조화된 데이터베이스에 액세스하기에 적합하다.	사전 교육된 대형 언어 모델과 외부에서 획득한 지식을 조정하는 데 사용될 수 있지만 자주 변경되는 데이터 원본에는 적합하지 않을 수 있다.
Data Processing (데이터 처리)	데이터 처리 및 처리에 필요한 최소한의 데이터를 포함한다.	고품질의 데이터셋을 생성하는데 의존하며 제한된 데이터셋은 성능 향상을 가져오지 않을 수 있다.

4. Core Components Of RAG

RAG vs Fine-Tuning

Model Customization (모델 맞춤)	정보 검색 및 외부 지식 통합에 중점을 두지만 모델 동작이나 작문 스타일을 완전히 맞춤화하지 않을 수 있다.	특정 톤이나 용어에 기반하여 LLM 동작, 작문 스타일 또는 특정 도메인 지식을 조정할 수 있다.
Interpretability (해석 가능성)	응답을 특정 데이터 원본으로 역추적하여 높은 해석 가능성과 추적 가능성을 제공한다.	블랙 박스와 유사하여 모델이 특정한 방식으로 반응하는 이유가 항상 명확하지 않아 상대적으로 낮은 해석 가능성을 제공한다.
Computational Resources (계산 자원)	검색 전략 및 데이터베이스 관련 기술을 지원하기 위해 컴퓨팅 자원에 의존합니다. 또한 외부 데이터 원본 통합 및 업데이트를 유지 해야 한다.	고품질의 훈련 데이터셋 준비 및 정의, 미세 조정 목표 정의 및 해당 계산 자원 제공이 필요하다.

4. Core Components Of RAG

RAG vs Fine-Tuning

Latency Requirements (대기 시간 요구 사항)	데이터 검색을 포함하여 더 높은 대기 시간이 발생할 수 있다.	미세 조정 후의 LLM은 검색없이 응답할 수 있어 대기 시간이 감소한다.
Reducing Hallucinations (환각 줄이기)	각 답변이 검색된 증거에 근거하기 때문에 본질적으로 환각에 취약하지 않다.	특정 도메인 데이터를 기반으로 모델을 훈련하여 환각을 줄일 수 있지만, 익숙하지 않은 입력에 직면할 때 여전히 환각을 나타낼 수 있다.
Ethical and Privacy Issues (윤리적 및 개인 정보 보호 문제)	외부 데이터베이스에서 텍스트를 저장하고 검색하는 데서 윤리적 및 개인 정보 보호 문제가 발생합니다.	훈련 데이터에서 민감한 콘텐츠로 인한 윤리적 및 개인 정보 보호 문제가 발생할 수 있습니다.

5. RAG Evaluation

Evaluation Targets

- 1. Retrieval Quality
  - 검색 품질을 평가하는 것은 Retrieval의 결과물이 제공한 컨텍스트의 효과를 파악하는 데 중요하다.
  - Hit Rate, MRR 및 NDCG와 같은 metric이 일반적으로 사용된다.
- 2. Generation Quality
  - 생성 품질을 평가하는 것은 Generator가 검색된 컨텍스트를 기반으로 일관된 및 관련성 있는 답변을 생성하는 능력을 중점적으로 살펴보는 것이다.
  - 라벨이 지정되지 않은 콘텐츠의 경우, 평가에는 생성된 답변의 충실성, 관련성 및 비해를 포함한다.
  - 반면에 라벨이 지정된 콘텐츠의 경우, 모델에 의해 생성된 정보의 정확성에 초점이 맞추어진다

5. RAG Evaluation

검색의 품질 평가

생성의 품질 평가

Evaluation Aspects

- **세 가지 주요 품질 점수**
  1. Context Relevance : 검색된 컨텍스트의 정확성과 구체성을 평가하여 관련성을 보장하고 부수적인 콘텐츠와 관련된 처리 비용을 최소화한다.
  2. Answer Faithfulness : 생성된 답변이 검색된 컨텍스트에 충실하도록 보장하여 일관성을 유지하고 모순을 피한다.
  3. Answer Relevance : 생성된 답변이 제기된 질문과 직접 관련이 있는지를 요구하여 핵심 질문에 효과적으로 대답한다.

5. RAG Evaluation

검색의 품질 평가

생성의 품질 평가

Evaluation Aspects

- **적응성과 효율성을 나타내는 네 가지 능력**

1. Noise Robustness : 모델이 질문과 관련이 있지만 구체적인 정보가 없는 노이즈 문서를 처리하는 능력을 평가한다.
2. Negative Rejection : 검색된 문서가 질문에 대답할 필요가 있는 지식을 포함하지 않을 때 모델이 답변을 하지 않는 능력을 평가한다.
3. Information Integration : 복잡한 질문에 대응하기 위해 모델이 여러 문서에서 정보를 종합하는 능력을 평가한다.
4. Counterfactual Robustness : 알려진 부정확성을 인식하고 무시하는 모델의 능력을 테스트한다. 심지어 잠재적인 오류에 대한 지시가 주어졌을 때도 그렇다.

5. RAG Evaluation

Evaluation Benchmarks and Tools

- 각 RAG의 평가 측면의 구체적인 Metric을 요약한 표이다.
- 주로 Accuracy가 사용되며, 각 평가 기준에 맞게 다양한 Metric이 사용될 수 있다.

Table 2: Summary of metrics applicable for evaluation aspects of RAG

	Context Relevance	Faithfulness	Answer Relevance	Noise Robustness	Negative Rejection	Information Integration	Counterfactual Robustness
Accuracy	✓	✓	✓	✓	✓	✓	✓
EM					✓		
Recall	✓						
Precision	✓			✓			
R-Rate							✓
Cosine Similarity			✓				
Hit Rate	✓						
MRR	✓						
NDCG	✓						

5. RAG Evaluation

Evaluation Benchmarks and Tools

- RAG 모델의 평가 프레임워크를 설명하며, 벤치마크 테스트(RGB, RECALL)와 자동화된 평가 도구(RAGAS, ARES, TruLens)로 구성된다.
- 이러한 도구들은 RAG 모델의 성능을 측정하는 데만 아니라 다양한 평가 측면에서 모델의 능력을 이해하는 데도 유용한 Quantitative Metrics을 제공한다.

Evaluation Framework	Evaluation Targets	Evaluation Aspects	Quantitative Metrics
RGB†	Retrieval Quality Generation Quality	Noise Robustness Negative Rejection Information Integration Counterfactual Robustness	Accuracy EM Accuracy Accuracy
RECALL†	Generation Quality	Counterfactual Robustness	R-Rate (Reappearance Rate)
RAGAS‡	Retrieval Quality Generation Quality	Context Relevance Faithfulness Answer Relevance	* * Cosine Similarity
ARES‡	Retrieval Quality Generation Quality	Context Relevance Faithfulness Answer Relevance	Accuracy Accuracy Accuracy
TruLens‡	Retrieval Quality Generation Quality	Context Relevance Faithfulness Answer Relevance	* * *

6. Future Prospect

Future Challenge of RAG

1. Context Length : RAG의 효능은 LLM의 Context Window 크기에 제한을 받는다. 대규모 언어 모델의 컨텍스트 창 크기를 거의 무한한 크기로 확장하기 위한 지속적인 노력이 진행 중이며, 이러한 변화에 대한 RAG의 적응은 중요한 연구 문제이다.

2. Robustness : 검색 중에 noise나 모순된 정보가 존재할 경우 RAG의 출력 품질에 부정적인 영향을 미칠 수 있다.

3. Hybrid Approaches (RAG+FT) : RAG와 fine-tuning을 결합하는 것이 주요 전략으로 떠오르고 있지만, 추가적인 연구가 필요한 상황이다.

6. Future Prospect

Future Challenge of RAG

4. Expanding LLM Roles : LLM은 RAG 프레임워크 내에서 검색 및 평가에 활용된다. LLM의 잠재력을 RAG 시스템에서 더욱 활용할 방법을 찾는 것은 계속해서 연구되는 방향이다.

5. Scaling 법칙 : LLM에 대한 스케일링 법칙은 확립되어 있지만, RAG에 대한 적용 가능성은 확실하지 않다.

6. Production-Ready RAG : 검색 효율성을 향상시키고 대규모 지식 베이스에서 문서 회수를 개선하며, 문서 소스나 메타데이터의 무단 누출을 방지하는 등의 데이터 보안을 보장하는 것은 아직 해결되지 않은 중요한 엔지니어링 도전 과제이다.

6. Future Prospect

Ecosystem of RAG

- RAG의 확장성과 다양성은 특히 의학, 법률 및 교육과 같은 전문 분야에서 추가 연구를 요구한다.
- 이러한 분야에서 RAG는 전문 도메인 지식 질문 응답에서 전통적인 fine-tuning 접근법과 비교하여 교육 비용을 줄이고 성능을 향상시킬 수 있는 잠재력을 갖추고 있다.
- 동시에 RAG의 평가 프레임워크를 개선하는 것이 다양한 작업에서 그 효능과 유용성을 극대화하기 위해 중요하다.
- 또한, RAG 기반 모델의 해석 가능성을 향상시키는 것은 여전히 주요 목표이다.
- 이를 통해 모델이 생성한 응답의 추론 과정을 이해할 수 있게 되어 RAG 응용 프로그램의 사용에 대한 신뢰성과 투명성을 증진시킬 수 있다.
- LangChain과 LLamaIndex와 같은 주요 도구들은 ChatGPT의 출현과 함께 인기를 얻어, 광범위한 RAG 관련 API를 제공하고 LLMs의 영역에서 필수적인 역할을 한다.

7. Conclusion

- RAG가 언어 모델의 능력을 향상시키기 위해 Parameterized 언어 모델의 지식을 외부 지식 베이스의 광범위한 Non-Parameterized된 데이터와 통합함으로써 상당한 발전을 이루었다는 점을 강조한다.
- RAG 프레임워크 내에서 Naive, Advanced, Modular RAG로 세 가지 발전적인 패러다임을 구체화하였다.
- RAG의 기술적 통합은 파인 튜닝 및 강화 학습과 같은 다른 인공 지능 방법론과의 결합을 통해 그 능력을 더욱 확장되었다.
- 그러나 RAG의 응용 landscape가 확대되는 가운데, 그 진화에 맞추기 위해 평가 방법론을 개선하는 필요성이 절박하다.

Thank You!