Chapter2. LLM 활용하기

2.1. LLM 활용 방법

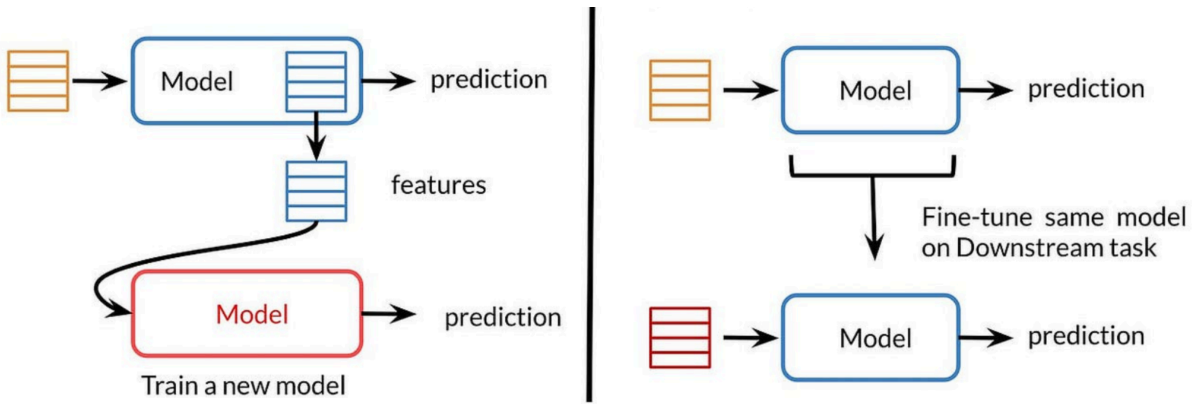
2.1.1. 파인튜닝

Fine-Tuning

- 전이 학습의 한 형태, 기존 LLM을 특정 작업이나 상황에 최적화시키기 위해 추가 학습시키는 과정
- 사전 학습된 모델의 일부 또는 전체 레이어를 재학습, 모델 전체를 업데이트
- 일반적으로 성능이 향상되지만 대규모 학습 데이터, GPU 등 많은 자원 필요

Transfer Learning

- 이미 학습된 모델을 새로운 작업에 적용
- 사전 학습된 모델의 파라미터를 고정한 채, 특정 task에 대하여 학습
  - ex) BERT + softmax ⇒ 감정분석
- 성능은 기존과 비슷하거나 파인 튜닝보다 낮을 수 있지만, 자원(비용, 메모리) 효율적



(왼)전이학습 & (오)파인튜닝
출처 :
https://vitalflux.com/transfer-learning-vs-fine-tuning-differences/#Differences_between_Transfer_Learning_Fine_Tuning

파인튜닝의 한계

1. 과도한 비용 발생

- 추가 학습에 필요한 데이터와 파라미터를 확신할 수 없음
- 기존 모델이 클수록 데규모의 데이터셋과 학습 자원(GPU, 클라우드) 등 필요

2. 학습 데이터셋 확보의 어려움

- 고정된 학습 데이터의 형태(질문-답변) 준비가 어려움

⇒ LLM에서는 파인튜닝보다 RAG를 선호

2.1.2. RAG

RAG(Retrieval-Augmented Generation)

- 정보 검색(Retrieval)과 생성(Generation)을 결합한 자연어 처리(NLP) 기술
- 기존 LLM이 사전학습된 내용으로만 응답하는 것과 달리, 외부 저장소에서 관련 정보를 검색하여 이를 바탕으로 응답
- 보다 정확하고 관련성 높은 답변 생성

1. 정보 검색 단계

- 질문 입력 → 벡터 쿼리 생성
- vector DB에서 가장 관련성 높은 문서/문단 검색

2. 텍스트 생성 단계

- 검색 결과와 질문을 결합하여 LLM에 전달
- 모델이 전달받은 정보를 바탕으로 답변을 생성

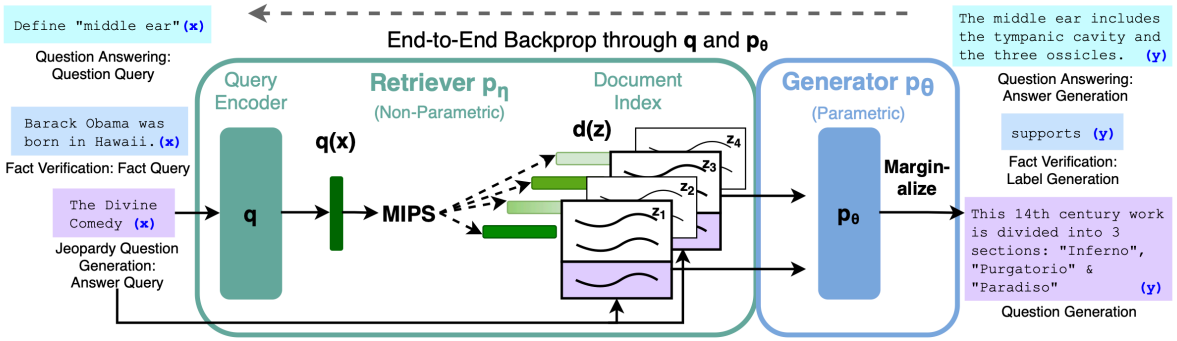


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat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

https://arxiv.org/abs/2005.11401 / rag알고리즘 동작과정

추가) RAG 핵심 알고리즘

1. Dense Passage Retrieval(DPR) (https://arxiv.org/abs/2004.04906)

- Dual Encoder : 질문과 검색 문서를 각각 독립적으로 인코딩하는 두 개의 BERT 모델 사용

> Q1. 싱글 or 트리플 인코더를 사용사용하면?

  - 싱글 인코더(cross encoder) : 질문과 문서를 하나의 입력 벡터로 사용
    - 장 : 질문-문서 관계까지 고려하여 높은 정확도를 보임
    - 단 : 질문마다 문서를 검색하여 대량 검색에 느림

    ⇒ 상위 K개 문서에서 정확한 문서를 골라낼때 사용 가능
  - 트리플 인코더(triple encoder) :
    - 질문, 정답 문서, 오답 문서를 각각 다른 인코더로 처리 (또는 Anchor–Positive–Negative 삼각 구조), 일반적으로 **훈련 시**에만 사용되며, **inference에는 Dual 구조만 남음**
    - 모델이 **헷갈려할 만큼 정답과 비슷하지만 실제로는 오답인 샘플(hard negative) 훈련에 적합**

> Q2. BERT가 아닌 다른 모델을 사용하면?

  - 다양한 모델 존재하며 사용 가능(자세한 모델의 종류는 생략)

- 유사도 계산 : 질문 벡터와 문서 벡터간 dot product를 계산하여 유사도 측정 (관련 내용 chap3에 나옴)

> 왜 LLM 검색에선 dot product를 많이 쓰나?
>
> - **속도 이점**: 벡터 DB(FAISS 등)에서 **inner product를 빠르게 계산할 수 있도록 최적화**되어 있음
> - **정규화된 임베딩**을 사용한다면: dot product ≈ cosine similarity (즉, 차이 없음)
> - 일부 LLM embedding은 **정규화되지 않은 채 dot product 기반으로 학습**되어서 **dot product가 성능에 더 유리**
>
> 참고) dot product vs 코사인유사도
>
> | 항목 | Dot Product | Cosine Similarity |
> |---|---|---|
> | **고려 요소** | 방향 + 크기 | 방향만 (정규화된 형태) |
> | **값 범위** | -∞ ~ +∞ | -1 ~ +1 |
> | **스케일 영향** | 민감 | 영향 없음 |
> | **사용 상황** | 임베딩 검색 속도 ↑ (벡터 크기 고려 가능) | 패턴 유사성만 고려하고 싶을 때 |

- top-k 문서 선택 : 유사도를 기반으로 관련성이 높은 k개의 문서/문단을 검색

2. end-to-end 학습

검색과 생성 단계가 독립적으로 동작하지 않고 하나의 통합된 모델로서 최적화 됨

2.1.3. 퓨샷 러닝

- 매우 적은 양의 데이터로 학습하는 능력
  - 충분한 양의 학습 데이터를 확보하기 어려울 때 유용
- 모델이 얼마나 적은 예시로 새로운 테스크를 학습/수행하는지에 다라 제로샷, 퓨샷, 원샷 러닝이라고 함

항목	Zero-shot	One-shot	Few-shot
예시 개수	0	1	몇 개
장점	빠름, 사전학습 효과 활용	적은 데이터로 유연성	더 높은 정확도
단점	오답 가능성↑	불안정성 존재	프롬프트 길이 제한 문제

2.2. LLM 활용 시 주의 사항

> 보안 및 규제 측면에서 제약이 많음

정보 필터링

일반 사용자, 특히 대고객 서비스 운영에 사용될 경우, 반드시 입출력 텍스트를 필터링 필요.

특히 개인정보가 입력되지 않도록 주의할 것

- 필터링 방법
  1. 전제 오류 감지
     - 프롬프트에 **논리적 오류**나 **허위 전제**가 있는지 검토
     - NLP 기반 **contradiction detection**, **named entity timeline check** 등 활용
  2. 금지 토픽 차단
     - 프롬프트에 위험하거나 허위 정보 유도 키워드 포함 여부 확인
     - 예: “백신은 위험해, 증거 대봐” → 위험 프롬프트로 간주
  3. 모호성 제거
     - “그 사람은 언제 죽었어?”처럼 **애매한 지시어** 포함된 질문 → 명확하게 수정 유도
     - 또는 "정확한 정보 기반"으로 질문을 다시 구성해서 전달

법적인 규제

특히 공공기관 미 금융산업에서 사용시 규제를 반드시 확인해야 함.

또한, 기업의 보안팀에서 정의한 규정에도 어긋나지 않도록 아키텍처 디자인 단계부터 주의해야 함.

할루시네이션

할루시네이션(hallucination) : AI모델, 특히 언어 모델이 부정확하거나 관력없는 정보를 생성하는 현상

이러한 현상을 최소화해야 하며 아래와 같은 방법으로 해결할 수 있음

1. 정확한 검색 결과
   - 정확한 외부 정보를 참조하여 답변하면 오류 가능성을 줄일 수 있음
2. 파라미터 조절(temperature = 0)
   - temperature : 언어 모델이 다음 단어를 선택할 때의 확률 분포를 조절하는 파라미터
     - **낮은 temperature (ex. 0 ~ 0.3)**

       → 결정적(deterministic)이고, 가장 확률이 높은 단어를 선택하려 함

→ **정확하고 일관된 응답, 창의성 ↓**

- **높은 temperature (ex. 0.8 ~ 1.5)**

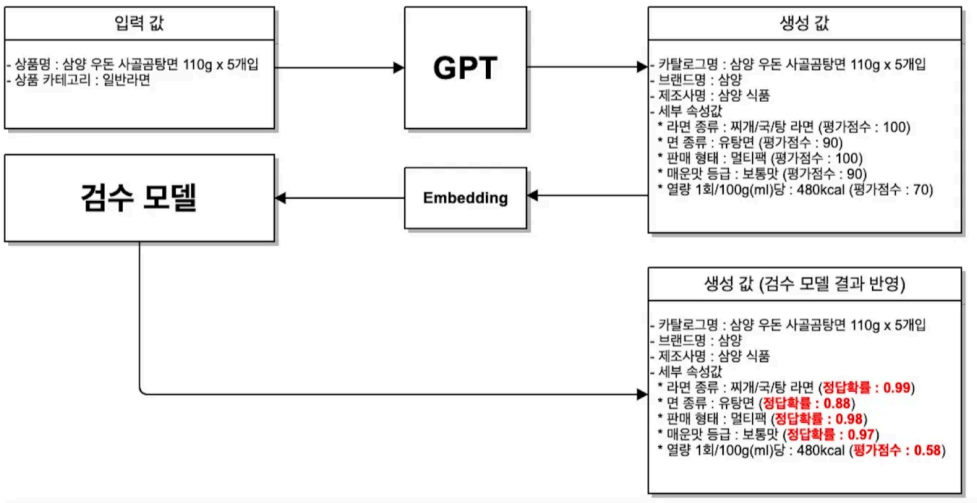
  → 확률이 낮은 단어도 선택될 가능성이 있음

  → **창의적이고 다양성 ↑, 하지만 오류 가능성 ↑ (할루시네이션)**

3. LLM구현 마지막에 할루시네이션 필터링을 추가

- 언어 모델이 응답을 선택한 후, 사후 필터링

  ex) 배달의 민족 - [**GPT를 활용한 카탈로그 아이템 생성]**



출력값을 그대로 사용하지 않고, 검수 모델을 사용하여 응답 결과를 판단하여 성능을 개선한 사례

보안

LLM모델은 모든 사용자가 공유하기 때문에 보안(특히 네트워크) 강화가 필요

- Azure OpenAI 서비스의 LLM 보안 강화 사례

보안 강화 기능	설명	목적
VNet & 프라이빗 엔드포인트	내부망에서만 LLM 호출	데이터 유출 방지
No data logging	호출 로그 저장 안 함	개인정보 보호
Content Filtering	유해 프롬프트/응답 필터링	윤리성·법적 위험 차단
RBAC + AD 연동	부서별 접근 권한 및 행동 기록	내부 통제
RAG + 내부 데이터	외부 정보 없이 응답 생성	정보 신뢰성 + 보안성 강화

2.3. LLM의 한계

편향과 공정성

모델이 학습한 데이터에 따라 편향된 텍스트를 생성할 가능성이 존재

투명성

특정 대답에 대하여 왜 그런 대답을 했는지 사용자에게 설명이 부족할 수 있음

데이터 의존성

너무 좁은 범위의 데이터를 학습한 경우 답변의 일반성이 떨어질 수 있음

정보의 일반화

너무 방대한 데이터를 학습하여 특정 분야에 특화된 답변이 어려울 수 있음

오류 가능성

잘못된 정보로 학습되면 잘못된 답변을 내놓을 수 있음

개인정보 보호

학습에 개인정보가 포함되면 생성시 노출될 수 있음

새로운 정보의 결여

최신 데이터로 학습이 지속되지 않을 경우 과거 데이터에 기반한 답변만 출력함

기업 내 데이터 미활용

특정 기업의 데이터를 학습하지 않아 사용이 어려움 → RAG로 해소 가능

정리

1. LLM을 활용하는 방법
    a. 파인튜닝 : 추가 데이터 학습
    b. RAG : 외부 정보 참조하여 답변 생성
    c. 문맥 내 학습 : 추가 학습 없이 예시만 주고 답변 생성
        i. 퓨샷 러닝/제로샷 러닝/원샷 러닝
2. LLM 사용시 주의사항
    - 할루시네이션을 방지하는 여러 방법 존재
        - 입력 정보 필터링
        - 파라미터 조정
        - 출력시 검증 모델 사용
    - 규제, 규정 확인
    - 보안 강화
3. LLM의 한계
    - AI 모델의 근본적인 한계가 LLM에도 동일하게 적용됨
    - Garbage in, garbage out 유념할 것!