RAG(Retrieval-Augmented Generation) 시스템 구현을 위한 개방형과 폐쇄형 LLM(Large Language Model) 모델 간의 성능 비교에 관한 연구

김민창*, 이채원, 염세훈

동서울대학교

*minchang1205@naver.com, 021122chaewon@naver.com, shyeom@du.ac.kr

A Comparative Study of Performance Between Large Language Model(LLM) of Open SourceModel and Closed Source Model Implementations for the Retrieval-Augmented Generation (RAG) System

Kim Min Chang*, Lee Chae Won, Sae Hun Yeom

Dong-Seoul Univ.

요 약

본 논문은 개방형 모델(Open Source Model)과 폐쇄형 모델(Closed Source Model)을 비교 분석하기 위해 Llama2, GPT-3.5 Turbo, Gemini-Pro와 같은 LLM(Large Language Model)을 사용하여 RAG(Retrieval Augmented Generation) 시스템을 구현 시 장단점을 비교 평가하였다. RAG 시스템을 구현하기 위해 랭체인(LangChain) 프레임 워크를 활용하였고 성능 평가를 위해 SQuAD(The Stanford Question Answering Dataset) 데이터 셋과 TriviaQA(Trivia Question Answering) 데이터 셋을 사용하여 F1 Score로 성능 평가를 수행하였다. 본 논문은 실험시 Llama2의 메모리의 GPU 메모리 사용량 문제를 해결하기 위해 QLoRA(Quantization Low Rank Adaptation)의 NF4(4bit-Normal Float) 양자화(Quantization) 기법을 적용하였다. 실험 결과 Llama2의 성능이 다른 2개의 LLM보다 성능은 떨어지지만 개방형 모델의 다양한 이점을 고려한다면 성능의 차이는 미세조정(Fine-Tuning) 방식으로 보완될 수 있음을 증명하였다.

Ⅰ. 서 론

LLM의 등장은 자연어 처리분야에서 많은 발전을 이끌어냈다. 이러한 모델들은 텍스트의 이해와 생성 능력의 성능을 향상시켰으나 가장 고성능을 자랑하는 LLM들 대부분이 폐쇄형 모델로 개발되어 각 도메인에서 상업적 활용에 한계가 존재한다. 본 연구에서는 상업적으로 자유롭게 개발 가능한 개방형 모델과 폐쇄형 모델의 성능을 비교 분석을 수행하여 장단점을 비교 평가 하였다. 본 논문은 개방형 모델로 Meta의 Llama2 모델과, 폐쇠형 모델로 Open AI의 GPT-3.5 Turbo와 Google의 Gemini-Pro 모델을 사용하여 RAG를 구현하였다. 그 이유는 LLM의 환각(Hallucination)현상이 실제 사용에 상당한 어려움을 초래하고 LLM의 신뢰성에 대한 우려를 불러일으키는데 이를 완화하기 위해 RAG가 많이 사용되기 때문이다. 본 논문의 연구결과로 더 넓은 범위의 개발자와 기업들이 LLM을 선택하여 RAG를 개발할 때 사용할 LLM 선택 시 도움을 주고자 한다.

Ⅱ. 본론

1. RAG(Retrieval-Augmented Generation)

RAG는 외부 지식 소스에서 검색된 관련 문서를 조건으로 하여 생성 중 LLM을 조정한다. 관련 문맥 문서는 먼저 검색기가 외부에서 검색 외부 소스에서 관련 컨텍스트 문서를 먼저 검색한 다음 원하는 출력을 입력 텍스트와 검색된 문서 모두를 조건부로 생성기에 의해 생성된다[1]. 그림 1은 RAG의 구조도이다.

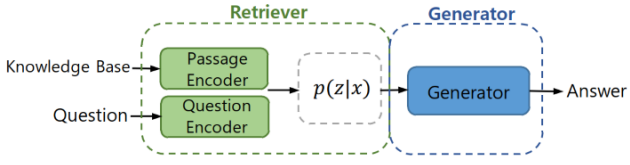
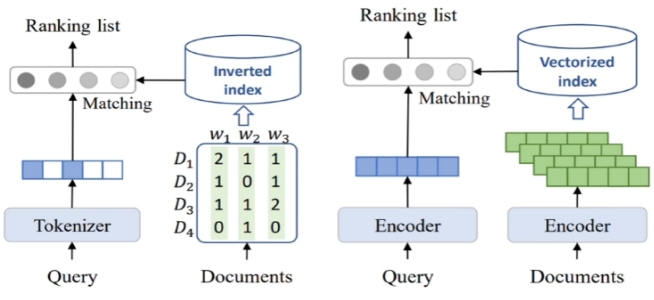


그림 1. RAG의 구조도

2. 앙상블(Ensemble) Retriever

본 논문에서 성능 비교를 위한 Retriever 모델은 랭체인에서 제공해주는 Ensemble Retriever를 사용하였다. Ensemble Retriever는 Sparse Retriever와 Dense Retriever를 조합한 Retriever이다. Sparse Retriever는 특정 단어에 해당하는 벡터의 차원만이 Non-Zero가 되어 희소 벡터 현상으로 인해 특정 키워드 기반의 검색이 가능하나 유사성을 고려하기 힘들다. Dense Retriever는 밀집 벡터의 형태로 Bi-Encoder의 구조를 가지고 단어간의 FAISS의 유사성 계산으로 검색이 진행된다.[2] 그림 2는 Sparse Retriever와 Dense Retriever의 구조도이다.



a. Sparse Retriever b. Dense Retriever

그림 2. Sparse Retriever와 Dense Retriever의 구조도

3. Llama2

Llama2는 Meta의 개방형 모델로 개발된 Transformer의 Decoder 기반의 모델로 Instruct Tuning을 사용하여 모델을 학습시키고, GQA(Group Query Attention) 방법을 사용해 추론 속도를 높였다. 기본적인 학습 방식은 RLHF(Reinforcement Learning from Human Feedback) 방법론으로 학습이 진행되는데 사전 학습된 모델을 감독 미세 조정을 적용하여 학습

시킨다. 그 후, 모델은 거부 샘플링과 PPO(Proximal Policy Optimization) 강화학습을 반복적으로 학습하여 환각 현상문제를 해결하고, 적절한 답변을 생성하도록 개발되었다. 특히 Llama2 특징은 데이터 셋을 창의적인 답변을 원하는 Helpfulness와 유해한 답변을 하지 않는 Safety를 나눠서 학습시키는 점이다. 이에 따라 Reward Model을 학습 시킬 때 별도의 모델로 나눠서 학습이 진행되고 PPO 학습 시에도 별도로 학습이 진행된다.[3] 그림 3은 Llama2 모델의 동작도 이다.

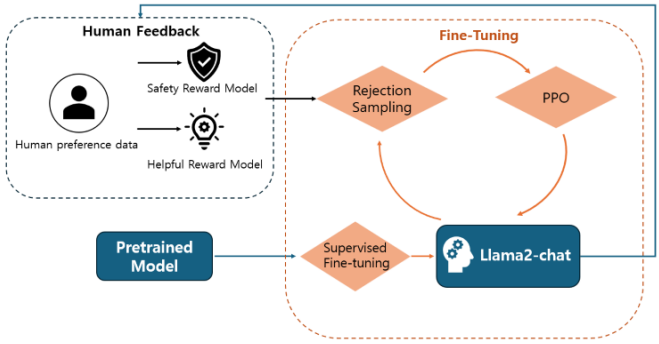


그림 3. Llama2 모델의 동작도

4. NF4 양자화(Quantization)

NF4 양자화는 QLoRA 논문에서 나온 방법론으로 4비트 정수 및 4비트 부동 소수점보다 더 나은 경험적 결과를 제공하는 4비트 NormalFloat로 양자화를 진행하는 방법이다. 매번 새로운 데이터에 대한 분석 정보가 들어올 때마다 사분위수 추정 프로세스를 진행하는 것이 아닌 사전 학습된 신경망 가중치인 표준 편차 σ를 조정하여 [-1,1]로 정규화 하여 모든 가중치를 단일 고정 분포 $N(0,1)$로 사용한다. 또한 0을 정확하게 표현하기 위해 q_i의 음수 부분은 2^{k-1}, 양수 부분은 $2^{k+1}+1$인 비대칭 데이터 유형으로 만든 후 계산하게 된다. 이를 NormalFloat라고 하고 계산된 정규 분포에 맞게 4bit 단위로 양자화가 진행된다.[4] 그림 4는 데이터를 NF4 Quatization 방법으로 양자화하는 과정이다.

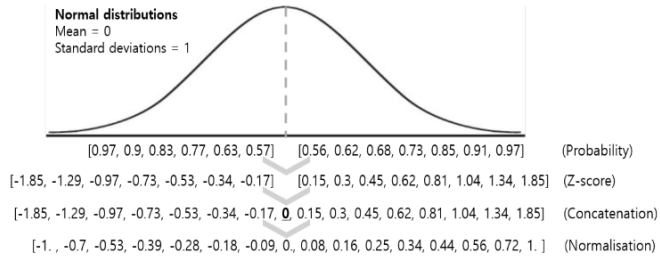


그림 4. NF4 Quatization 수행과정

Ⅲ. 성능 비교

1. 실험 데이터 셋(Set)

본 논문에서 성능 비교를 위하여 사용한 데이터셋은 표1과 같다.

표 1. 데이터 셋에 대한 설명

데이터 셋	크기	내 용
SQuAD	10.6k	SQUAD는 위키백과 문서 기반의 독해력 데이터 셋으로, SQUAD1.1의 100,000개의 질문과 적대적으로 작성한 50,000개 이상의 답변 불가능한 질문으로 구성된 SQUAD2.0을 사용하였다.
TrivaQA	17.9k	Wikipedia와 웹에서 수집한 66만 2천 개의 문서에서 950만 개의 질문-답변 쌍으로 구성된 데이터 셋으로, 문서의 문맥이 매우 길기 때문에 SQuAD보다 복잡하다.

2. LLM 모델들 간의 RAG 시스템 구현 시 성능 비교

Llama2와는 다르게 GPT-3.5 Turbo 모델과 Gemini-Pro 모델은 폐쇠형 LLM이기 때문에 모델의 자세한 정보가 나와 있지 않아 직접적인 비교가 어려웠으나 각 모델간의 유의미한 성능 차이가 확인되었다. 하지만 개방형 모델의 이점을 살리기 위해 미세조정하여 성능 비교한 결과 SQuAD 데이터셋의 경우 성능이 약 6.9%, 13.7% 떨어지는 것을 확인하였다. 그러나 TriviaQA 데이터셋의 경우 성능이 약 14.83%, 7.5% 더 높은 것을 확인하였다. 실험 환경에서 GPT 3.5-Turbo 모델과 Gemini-Pro 모델은 API로 구현하여 GPU가 필요하지 않으며 Llama2같은 경우 NF4 양자화 기법을 적용하여 GPU 프로세서가 필요하다.

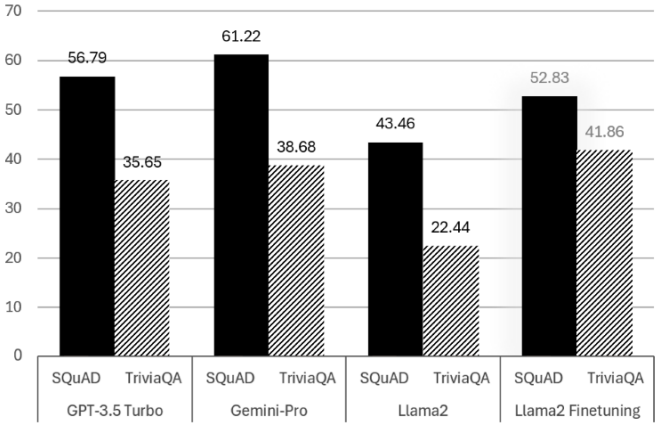


그림 5. SQuAD Dataset에 따른 각 모델 별 F1 Score

Ⅳ. 결론

다양한 인공지능 에이전시(Agency)를 구현할 때 이미 개발되어 있는 LLM을 선택해야 하는 경우 개방형 모델과 패쇄형 모델 각각의 장단점이 고려하여 선택해야 한다. 폐쇄형 모델의 경우 매번 LLM의 사용료를 지불해야 하며 보안성에 대한 문제 혹은 특정 도메인에 맞는 최적의 모델을 구현할 수 없다는 단점 있다. 반면, 개방형 기반의 모델인 Llama2는 컴퓨터 자원을 필수적으로 요구하지만 상업적인 제약 없이 사용 및 적용 도메인에 맞게 미세조정이 가능하다는 장점이 있다. 이러한 특성은 각 도메인에 적합한 인공 지능 에이전시를 개발하는 과정에서 큰 이점을 제공한다. 본 논문에서는 이를 실험 결과로 증명하였고 향후 특정 도메인에서 인공지능 에이전시를 개발시 개방형 모델만으로도 충분한 성능으로 얻을 수 있음을 실험 결과로 증명하였다. 향후 특정 도메인용 인공지능 에이전시 구현시 참고 자료로 활용될 수 있을 것으로 생각된다..

참 고 문 헌

[1] Lei Huang(2023)., A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions., arXiv:2311.05232

[2]https://python.langchain.com/docs/modules/data_connection/retrievers/ensemble/

[3] Hugo Touvron(2023)., Llama 2: Open Foundation and Fine-Tuned Chat Models., arXiv:2307.09288

[4] Tim Dettmers(2023)., QLORA: Efficient Finetuning of Quantized LLMs., arXiv:2305.14314