

24-SPRING KUBIG CONTEST

RAG가 성능이 좋더라구!

-한솔 데코 기준2 AI 경진대회

NLP 분반 4팀

1771 황민아 1871 진서연 최유민

목차

1. Intro.
2. Project Goal
3. Preprocessing
4. Models
5. Inference
6. Conclusion

1. Intro.

대회 소개

한솔데코 시즌2 AI 경진대회

도배 하자 질의 응답 처리 : 한솔데코 시즌2 AI 경진대회

알고리즘 | 언어 | LLM | MLOps | QA | Cosine Similarity

₩ 상금 : 1000만 원

⌚ 2024.01.29 ~ 2024.03.11 09:59 [+ Google Calendar](#)

👤 1,152명 🗓 D-17



“도배 하자 질의 응답 AI 모델 개발”

1. Intro.

데이터 설명

train.csv

Date depend						
	id	질문_1	질문_2	category	답변_1	답변_2
1	TRAIN_000	면진장치가 뭐야?	면진장치에 사용되는 주요 ...	건축구조	면진장치란 지반에서 오는 ...	면진장치란 건물의 지반에...
2	TRAIN_001	내진설계의 종류 좀 알려줘	내진설계에는 어떤 종류가 ...	건축구조	내진 설계의 종류로 내진구...	내진설계에는 내진구조, 제...
3	TRAIN_002	철골구조의 장점이 뭐야?	철골구조의 장점을 알려줘?	건축구조	철골구조는 건물의 외벽에...	철골구조의 장점은 건물의 ...
4	TRAIN_003	철골철근 콘크리트 구조가 ...	철골철근 콘크리트 구조의 ...	건축구조	철근철골콘크리트는 철골...	철골철근콘크리트 구조는 ...
5	TRAIN_004	철골구조는 어떤 방식이 있...	철골구조의 다양한 방식이 ...	건축구조	철골구조는 일반철골구조...	철골구조는 일반철골구조...

Train data

- id : 샘플 고유 번호
- 질문_1, 질문_2 : 샘플 별 동일한 내용으로 구성된 질문 2개
- category : 질문 - 답변 (QA) 샘플의 도메인 세부 분야
- 답변_1, 답변_2, 답변_3, 답변_4, 답변_5 : 샘플 별 질문에 대한 동일한 답변 Reference 5개



Test data

- id & 질문

1. Intro.

Main Idea

“RAG”

Retrieval-Augmented Generation

- > 공통 관심사 : “검색”
- > 학습한 내용에 없는 정보에 대한 응답을 잘 하기 위해서는?
- > 더 정확하고 신뢰도 높은 답변을 위해서는?

1. Intro.

What's RAG?

ChatGPT 3.5 ▾

서연 You
Llama가 뭐야?

ChatGPT
Llama(라마)는 남미의 안데스 산맥 지역에서 주로 발견되는 가축으로, 남미에서 주로 경작용으로 이용됩니다. Llama는 멧돼지와 같이, 남미의 고산 지대에서 자연적으로 발견되는 유일한 가축입니다. 이들은 울음소로 알려져 있으며, 그들의 울음털은 매우 유용한 양모로 사용됩니다. 또한, 라마는 운반과 화물 운송에도 사용되며, 과거에는 신성한 동물로 여겨졌습니다. 요즘에는 관광 목적으로도 많이 이용됩니다. 종종 라마는 알파카와 혼동되기도 하는데, 이들은 밀접한 친척이지만 서로 다른 종입니다.



LLM의 문제점

- > 모르는 질문에 대해서 허위 답변 생성
- > 신뢰할 수 없는 출처로부터 답변 생성
- > 최신 정보보단 오래된 정보 제공

이런 문제를 해결하기 위한 방법,
Retrieval-Augmented Generation!!

1. Intro.

What's RAG?

외부의 신뢰할 수 있는 데이터 소스에서 가져온 정보로
AI 모델의 정확성과 신뢰성을 향상하는 기술

Retriever + Generator

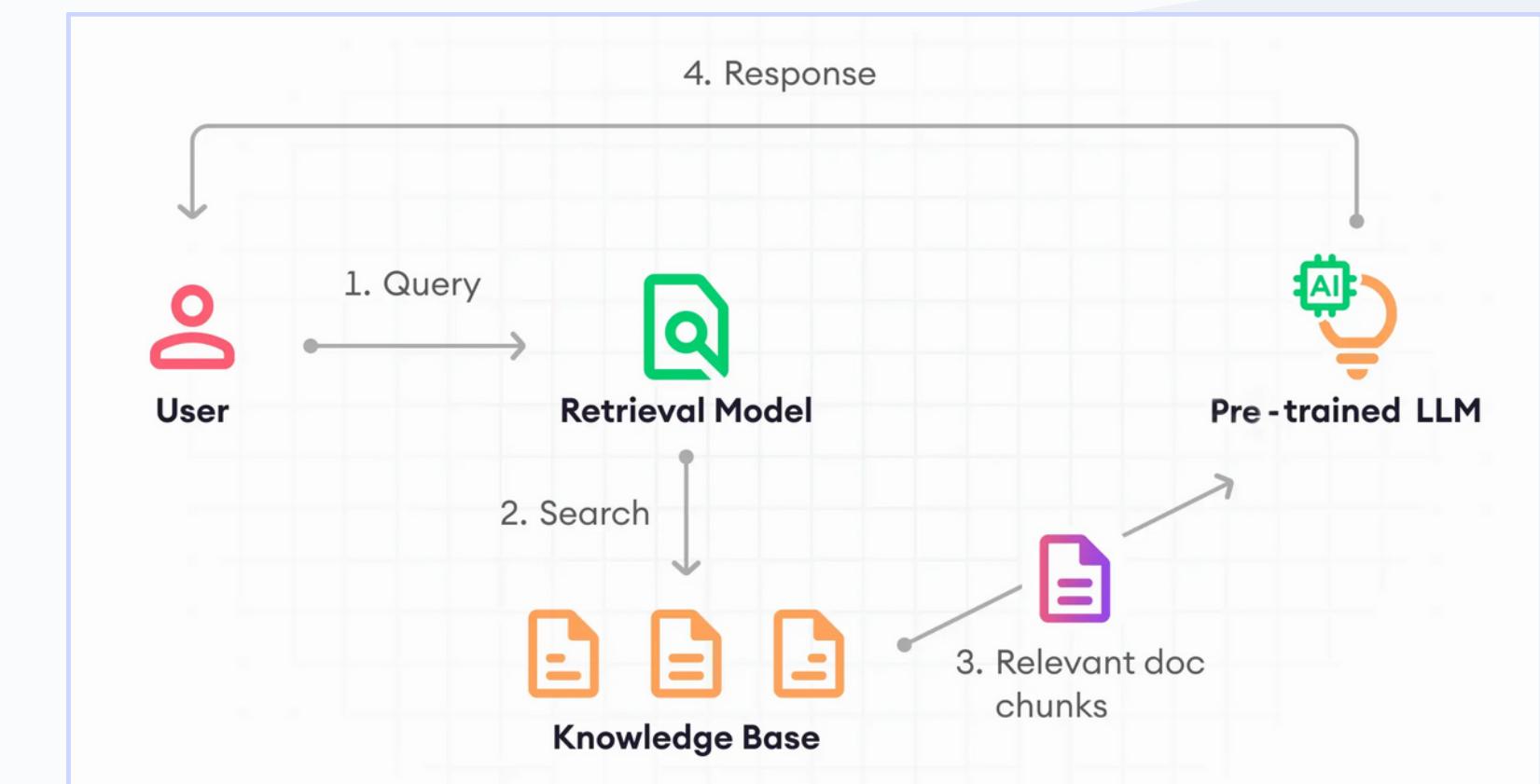
- > **Retriever** : 사용자의 질문에 대한 답변 생성 시 외부 데이터베이스에서 문서나 정보를 검색
- > **Generator** : 앞서 검색한 정보를 통해 유의미한 답변 생성

장점

- > LLM 학습 시 사용된 데이터보다 최근 정보 포함

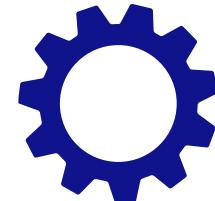
-> 최신 정보를 학습시키는 것보다

외부 데이터베이스 업데이트가 훨씬 효율적



2. Project Goal

Our Insight!

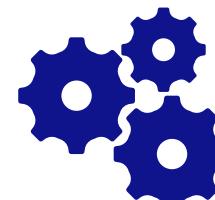


RAG

의 성능은 얼마나 좋을까

LLM finetuning VS RAG

LLM finetuning과 비교했을 때 RAG는 성능이 뛰어날까?



RAG

를 활용한 성능 향상을
위해선?

LLM finetuning + RAG

LLM finetuning에 RAG를 얹는 것이 더 나은 성능에 기여할까?
혹은 RAG의 Generator 부분만 contextualize하는 것이 더 효과적일까?

3. Preprocessing

Train data

- 토큰 추가 : 각 모델이 데이터를 잘 이해/처리하고, 질문과 답변을 구분하기 위함

train.csv	train_solar.csv
TRAIN_068,포세린 타일의 단점이 뭐야,포세린 타일을 사용하는 것의 단점은 무엇인가요?,마감재,포세린 타일은 시공 방법에 따라 하자 발생률이 높고 오염에 취약하다는 단점이 있습니다.,"포세린 타일의 단점 중 하나로는 시공 방법에 따라 하자 발생률이 높을 수 있다는 점이 있습니다. ~~~~~~	"< im_start >user 다음 질문에 대해 자세하고 정확하게 답하시오. 질문 : 포세린 타일의 특징은 무엇인가요? 답변 : < im_end > < im_start >assistant 포세린 타일은 밀도와 강도가 뛰어나고 변형이 적고 반영구적인 특징이 있어, 주방, 화장실, 현관 등의 습기가 많은 실내나 실외에서 사용하기에 적합한 재료입니다. 이러한 특성으로 인해 포세린 타일은 시간이 지나도 모양을 유지하고 쉽게 변색되지 않는 장점이 있습니다. 또한, 미세한 공간에 적합한 크기의 타일을 제작하여 디자인의 다양성을 증대시킬 수 있기 때문에 실내 장식에 다양한 활용이 가능합니다.< im_end >"
쉼표 단위로 데이터들이 나열됨	토큰이 질문과 답변을 잘 구분하고 있음

3. Preprocessing

RAG data

질문, 답변 10개 각각 문서화

중복 문제 발생!

문서 1개로 통합

질문	답변
페인트는 무엇인가요?	페인트는 도막을 형성하여 원자재나 완제품의 노화, 산화 등을 방지하고 수명을 연장시...
페인트는 무엇인가요?	페인트는 표면에 도막을 형성하여 원자재나 완제품을 노화, 산화 그리고 수명단축으로부...
페인트는 무엇인가요?	페인트란 소지의 외관에 도막을 형성함으로써, 각종 원자재나 완제품의 노화, 산화 등...
페인트는 무엇인가요?	페인트는 표면에 도막을 형성하여 외부 환경으로부터 보호하는 역할을 합니다. 이를 통...
페인트는 무엇인가요?	페인트는 소재의 외관을 보호하고 조화로운 색상을 부여하여, 원자재나 완제품의 노화,...

질문	답변
페인트는 무엇인가요?	페인트는 도막을 형성하여 원자재나 완제품의 노화, 산화 등을 방지하고 수명을 연장시...
페인트가 뭐야?	페인트는 도막을 형성하여 원자재나 완제품의 노화, 산화 등을 방지하고 수명을 연장시...

'질문: 페인트가 뭐야? 페인트는 무엇인가요? \n답변: 페인트란 소지의 외관에 도막을 형성함으로써, 각종 원자재나 완제품의 노화, 산화 등을 방지하여 수명을 연장시키고, 방수, 방오, 내화, 전자파 차폐, 단열 등의 특수 목적에 부합한 기능성을 부여하며 주위와의 조화로운 색상을 구현하는 핵심 소재입니다. 페인트는~~~'

3. Models

Base Model

1) Kogpt2

2) LLAMA-2-ko-7b

3) SOLAR-10.7B

4) Gemma-7B

SOLAR 등 대형 모델의 memory 문제로 인해
효율적으로 memory를 사용하고자 함

- 모델을 4bit로 로드 : 보다 적은 비트로 표현되는 가중치를 사용해 메모리를 절약
- QLoRA(Quantized Low-Rank Attention) 이용 : 가중치를 더 작은 비트 수로 표현하고, 어텐션 메커니즘을 저렴하게 만드는 기술

3. Models

QLoRA 코드

```
# QLoRA parameters
lora:
    # LoRA attention dimension
    lora_r : 256
    # Alpha parameter for LoRA scaling
    lora_alpha : 128
    # Dropout probability for LoRA layers
    target_modules : ["q_proj",
        "up_proj",
        "o_proj",
        "k_proj",
        "down_proj",
        "gate_proj",
        "v_proj"]
    lora_dropout : 0.05
    task_type : "CAUSAL_LM"
# bitsandbytes parameters
bnb:
    # Activate 4-bit precision base model loading
    use_4bit : True
    # Compute dtype for 4-bit base models
    bnb_4bit_compute_dtype : "float16"
    # Quantization type (fp4 or nf4)
    bnb_4bit_quant_type : "nf4"
    # Activate nested quantization for 4-bit base models (double quantization)
    use_nested_quant : False
```

```
bnb_config = BitsAndBytesConfig(
    load_in_4bit=CFG.bnb.use_4bit,
    bnb_4bit_quant_type=CFG.bnb.bnb_4bit_quant_type,
    bnb_4bit_compute_dtype=compute_dtype,
    bnb_4bit_use_double_quant=CFG.bnb.use_nested_quant,
)

# Check GPU compatibility with bfloat16
if compute_dtype == torch.float16 and CFG.bnb.use_4bit:
    major, _ = torch.cuda.get_device_capability()
    if major >= 8:
        print("=" * 80)
        print("Your GPU supports bfloat16: accelerate training with bf16=True")
        print("=" * 80)

# Load base model
model = AutoModelForCausalLM.from_pretrained(
    CFG.model_name,
    quantization_config=bnb_config,
    device_map=CFG.device_map
)

# Load LoRA configuration
peft_config = LoraConfig(
    r = CFG.lora.lora_r,
    lora_alpha = CFG.lora.lora_alpha,
    lora_dropout = CFG.lora.lora_dropout,
    target_modules = CFG.lora.target_modules,
    task_type = CFG.lora.task_type,
    bias = 'none'
)
```

3. Models

Kogpt2

- SKT-AI에서 2020년 2월 개발
- GPT-2 모델을 Finetuning한
한국어 언어 모델
- 부족한 한국어 성능 극복을 위해 40GB 이상의
텍스트로 학습된 한국어 decoder 언어 모델
- 문장 생성에 최적화되어 있음



3. Models

LLAMA-2-ko-7b

- Meta에서 2023년 2월에 공개한 오픈 모델
- Large Language Model Meta AI의 약자
- 70억~650억 매개변수에 이르는 다양한 모델 크기가 학습됨
- 알파카(Alpaca), 비큐냐(Vicuna) 등의 수많은 파생형 모델들의 탄생에 기여
- 안전성과 유용성에 대한 보상 모델을 제작
- 추론, 코딩, 지식 태스크를 포함하는 벤치마크 테스트에서 ‘MPT’, ‘Falcon’ 등의 타 LLM들보다 성능이 뛰어남



3. Models

LDCC/SOLAR-10.7B

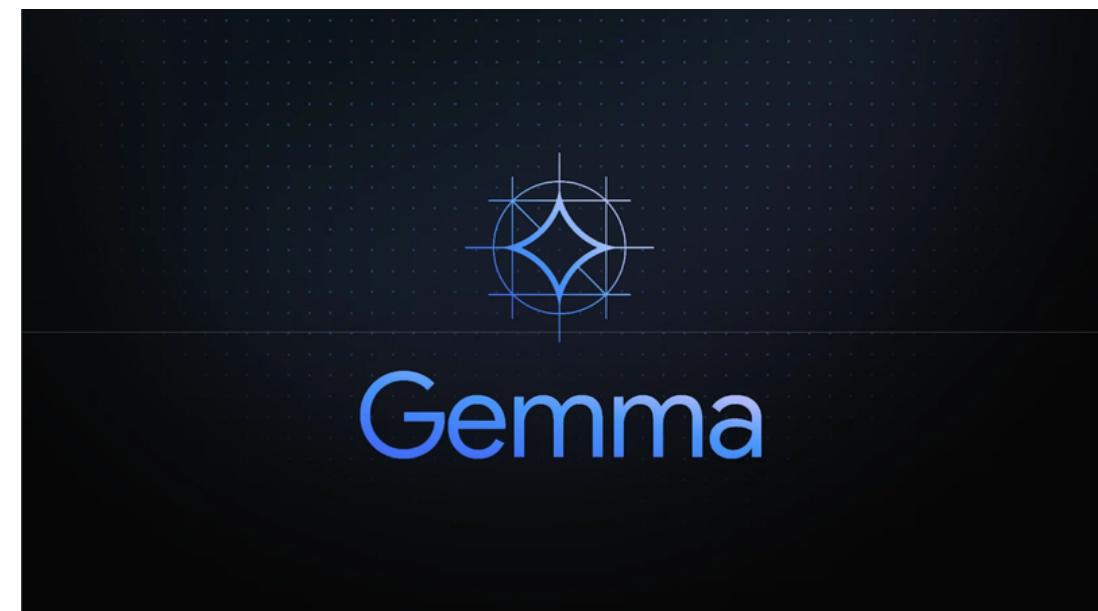
- 국내 기업 “업스테이지”에서 개발
- 세계 최초의 107억 매개 변수 모델
- Depth Up-Scaling(DUS) 방법 사용 : 훈련 및 추론 프레임워크에 복잡한 변경이 필요 없어, 매우 효율적인 사용이 가능하게 함
- 특히 NLP task에서 우수한 성능을 보임
- 다만 모델 확장 과정에서 컴퓨팅 자원이 증가해 성능 저하 문제를 유발
- SOLAR 모델을 한국어 데이터에 Fine Tuning 하여 배포한 모델 -> LDCC/SOLAR10.7B



3. Models

Gemma-7B

- Google에서 2024.02.21에 출시한 오픈 모델
- 라틴어로 “보석”을 뜻하는 gemma에서 따옴
- Google AI인 Gemini의 경량화 버전
- 매개변수가 20억 개인 '젬마 2B'와
70억 개인 '젬마 7B' 두 모델로 출시
- Kaggle, Hugging Face, Nemo, Vertex AI
등을 통해 사용 가능
- 엔비디아와 제휴해 GPU 최적화 지원



3. Models

RAG

- RAG Model
 - VectorDB: ChromaDB
 - Embedding model: sentence-transformer/distiluse-base-multilingual-cased-v1
 - model과 DB 연결 : Langchain 사용
- Query : 각 Test set의 Split 된 질문

```

modelPath = "distiluse-base-multilingual-cased-v1"
model_kwargs = {'device':'cuda'}
encode_kwargs = {'normalize_embeddings': False}
embeddings = HuggingFaceEmbeddings(
    model_name=modelPath,
    model_kwargs=model_kwargs,
    encode_kwargs=encode_kwargs
)

# Load Documents (data loader)
loader = CSVLoader(file_path='./data/train_data.csv', encoding='utf-8')
data = loader.load()

# Load ChromaDb
db = Chroma.from_documents(data, embeddings, persist_directory="./chroma_db")

```

```

db = Chroma(persist_directory="./chroma_db", embedding_function=embeddings)

retriever = db.as_retriever(search_kwargs={"k": 5})

```

```

# generation for rag
if CFG.rag == True:
    docs = retriever.get_relevant_documents(test_question)
    formatted_docs = format_docs(docs)
    prompt = f'''<|im_start|>system\nAct like a wallpapering expert.\nUse the following documents to answer the questions.\nyou must answer the question in Korean.<|im_end|>\n{formatted_docs}<|im_start|>user\n질문 : {test_question} 답변 : <|im_end|>\n<|im_start|>assistant'''

```

4. Inference

Idea - Question Split

Inference 단계에서
성능 향상을
 도해볼 수 있을까?

test.csv	
id	질문
TEST_000	방청 페인트의 종류에는 어떤 것들이 있는지 알고 계신가요? 또한, 원목사이딩을 사용하는 것에 어떤 단점이 있을까요?
TEST_001	도배지에 녹은 자국이 발생하는 주된 원인과 그 해결 방법은 무엇인가요?

한 질문이 “두 문장”으로 이루어져 있을 때,
각각의 문장에 대해 답변을 만든 후 이 두 답변을 결합한 것이,
단일 답변을 생성하는 것보다 더 나은 결과를 얻을 수 있을까?

4. Inference

Result

ex. Kogpt2

질문(TEST_000)	단일 답변 생성	각각의 문장에 대해 답변 생성 후 두 답변 결합
<p>방청 페인트의 종류에는 어떤 것들이 있는지 알고 계신가요? 또한, 원목사이딩을 사용하는 것에 어떤 단점이 있을까요?</p>	<p>방청 페인트는 주로 외장재나 문틀 등에 사용되며, 내구성이 상대적으로 떨어져서 오랫동안 사용할 수 있는 것이 주요한 단점으로 꼽힙니다. 또한 표면이 부드러운 특성을 가지고 있어서 표면의 불규칙한 표면을 처리할 때 추가적인 비용이 발생할 수 있다는 점도 고려해야 합니다.</p>	<p>방청페인트에는 다양한 종류가 있지만 주로 사용되는 것으로는 철골 구조물, 자동차 및 가전제품의 철재 부속품, 선박 및 컨테이너박스의 외관에 사용됩니다. 이러한 페인트는 금속 표면에 우수한 착색력과 내구성을 제공하여 금속 소재를 오랜 기간 동안 보호하는 데 사용됩니다. 또한, 방청 효과와 함께 열 전도율이 높아 에너지 효율이 뛰어나며 광택과 발색이 우수하여 오랫동안 선명하고 아름다운 모습을 유지할 수 있습니다. 원목사이딩의 단점은 주로 가격대가 높고 관리가 어렵다는 점입니다. 또한 습기에 약해 뒤틀림, 부서짐, 수축/팽창이 발생할 수 있다는 점도 단점으로 꼽힙니다.</p>

Dacon public score

0.6416

0.69859

5. Conclusion

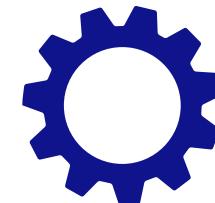
Result

	FineTuning	RAG	Dacon public score
Kogpt2	0.642(U.Q.) 0.699	-	0.710
SOLAR-10.7B	-	0.660	0.680
LLAMA-2 -ko -7b	-	0.621(U.Q.) 0.701	-
Gemma-7B	-	-	0.632

U.Q. : Unsplitted Question

5. Conclusion

Insight

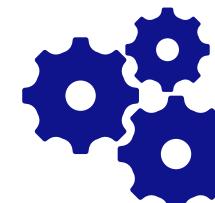


RAG

의 성능은 얼마나 좋을까

LLM finetuning VS RAG

LLM finetuning과 비교했을 때 RAG는 성능이 뛰어날까?



RAG

를 활용한 성능 향상을
위해선?

LLM finetuning + RAG

LLM finetuning에 RAG를 얹는 것이 더 나은 성능에 기여할까?
혹은 RAG의 Generator 부분만 contextualize하는 것이 더 효과적일까?

5. Conclusion

Insight

Fine Tuning, RAG - 언제 사용하나?

Fine Tuning	RAG	FT + RAG
<ul style="list-style-type: none">모델 사이즈에 알맞은 학습 데이터가 충분할 때GPU Resource가 충분할 때	<ul style="list-style-type: none">도메인이 Specific할 때모델의 Pre-trained Data에 존재하지 않는 Data일 때Data가 적을 때	<ul style="list-style-type: none">데이터도 충분하고, 도메인이 Specific할 때새로운 데이터가 계속 업데이트 될 때

한솔데코 AI 경진대회

- 작은 train data 크기 : 데이터 증강을 시도했다면, 더 좋은 성능을 보였을 수도 있었을 것
- 도배라는 분야가 LLM이 학습하기 어려운 특수한 분야라는 점

-> **RAG가 더 적합**

5. Conclusion

소감 및 한계점

소감

- 다양한 LLM의 finetuning을 통해 LLM의 장점과 한계에 대해 더 깊이 있는 이해를 얻어갈 수 있었음
- Domain-specific한 데이터를 처리함에 있어 전처리와 증강 등이 중요한 역할을 한다는 것을 느낌
- 모델을 구성하는 다양한 컴포넌트를 합치는 과정에서 협업 툴과 의사소통의 중요성을 느낌

후속 연구

- LLM 모델 크기 증가 - 70B LLAMA 사용
- Train Data Augmentation
- QLoRA 대신 Full Fine tuning 사용
- Instruction Tuning 고도화

감사합니다

Thank you

Appendix

평가 산식(Cosine Similarity)의 모호성

코사인 유사도는 벡터 간의 각도를 기반으로 유사성을 측정함.
이에 단어나 문장의 의미의 다양성을 완전히 반영하지 못할 수 있음.

```
# 질문_1, 질문_2 사이의 평균 cosine similarity
print(cosine_similarity_by_column(train, '질문_1', '질문_2'))
# 답변 사이의 평균 cosine similarity
print(cosine_similarity_by_column(train, '답변_1', '답변_2'))
print(cosine_similarity_by_column(train, '답변_2', '답변_3'))
print(cosine_similarity_by_column(train, '답변_3', '답변_4'))
print(cosine_similarity_by_column(train, '답변_4', '답변_5'))
✓ 30.2s
0.7938607565202513
0.7341158253991086
0.8505087580925189
0.8439465081839828
0.8411438416527666
```

train data에 있는 답변_1 ~ 답변_5의
cosine_similarity를 계산해본 결과,
0.73~0.85 사이의 값이 도출됨