[2023 후기 졸업과제 중간 보고서]

LLM을 사용한 AI 챗봇 연구

제출일	2024. 03.25	전공	정보컴퓨터공학부
		담당교수	김호원
학번	201824423 201924655	이름	김도훈 이보원

목 차

1. 요구조건 및 제약 사항 분석에 대한 수정사항

1.1. 요구조건

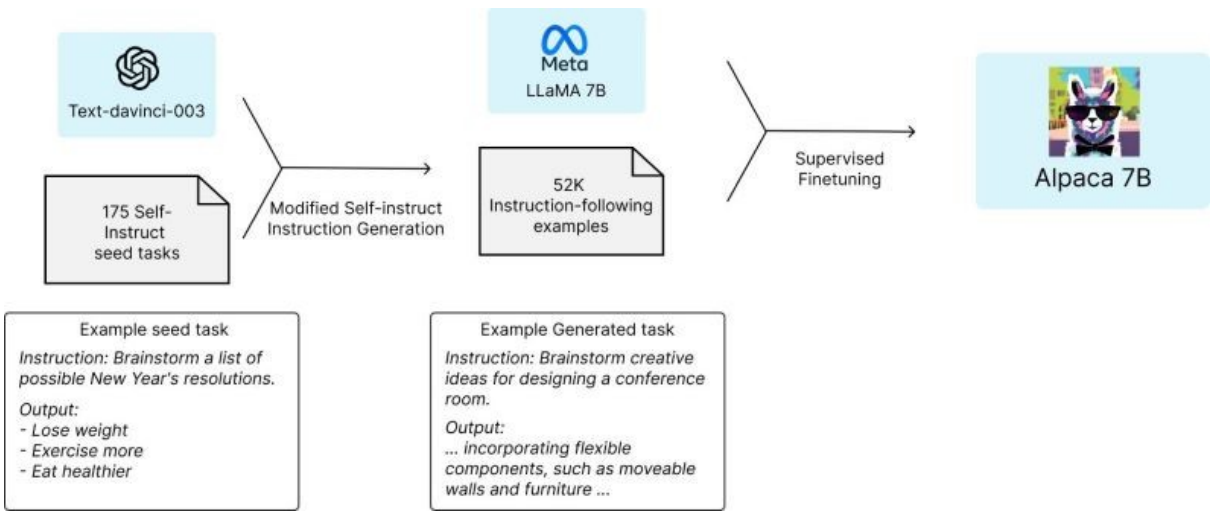
LLM(Large Language Model)을 이용해 바쁜 현대인들에게 영양 균형과 개인의 취향을 고려한 적절한 식사 메뉴를 제안하여 다양하고 새로운 메뉴들을 접할 수 있게 도와주는 메뉴 추천 AI 챗봇을 웹 어플리케이션의 형태로 개발한다.

1.2. 제약 사항 분석에 대한 수정사항

1.2.1 기본 모델 선정

fine tuning을 진행할 기본 모델로써 KoAlpaca를 선정했다. KoAlpaca는 Stanford Alpaca 모델을 학습한 방식과 동일한 방식으로 학습을 진행한 한국어 Alpaca 모델이다. Stanford Alpaca는 Stanford University에서 Meta의 오픈소스 언어 모델인 LLaMA 7B를 fine tuning하여 개발한 모델이다.

KoAlpaca는 2024년 3월 기준 2개의 Polyglot-ko 기반 모델과 4개의 Meta LLaMA 기반 모델이 존재하며 Polyglot-ko 기반 모델과 M eta LLaMA 7B 모델은 Full Finetune을 마친 모델이다. LLaMA 모델은 한국어 데이터셋을 충분히 학습하지 않아 한국어 추론 성능이 낮게 나오는 이슈가 존재하고 너무 큰 파라미터를 가진 모델은 개발 환경인 Colab Free에서 저장용량 부족 이슈가 발생할 가능성이 존재해 Polyglot-ko 5.8B기반 모델을 사용하기로 결정하였다.



[그림 1] Stanford Alpaca 학습 과정

Hyperparameter	Value
$n_{parameters}$	5,885,059,072
n_{layers}	28
d_{model}	4096
d_{ff}	16,384
n_{heads}	16
d_{head}	256
n_{ctx}	2,048
n_{vocab}	30,003 / 30,080
Positional Encoding	Rotary Position Embedding (RoPE)
RoPE Dimensions	64

[그림 2] Polyglot-ko-5.8B 하이퍼파라미터 값

1.2.2 학습 데이터 수집

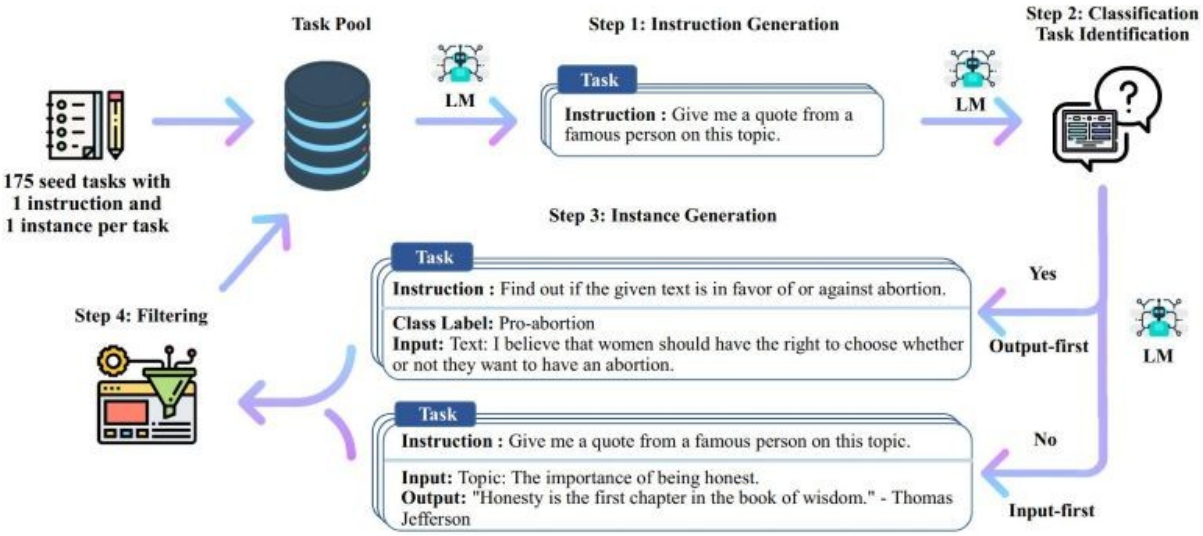
기존에는 학습 데이터 수집을 위해 배달 플랫폼이나 리뷰 사이트를 크롤링하려 했지만 음식 추천에 관련한 대화 데이터가 너무 부족하여 Self-Instruct를 활용하여 데이터셋을 직접 제작하기로 하였다. Alpaca 데이터셋은 그림 3와 같이 지시사항인 Intruction과 Input, Ouput으로 구성되어 있으며 데이터셋 구축시에도 그림과 같은 형식으로 제작한다.

Alpaca 개발자들은 LLM 을 학습시킬때에 고려해야 할 두가지 문제로 강력한 언어모델의 필요성과 고품 질의 instruction-following 데이터가 존재해야 함을 언급했다. Stanford Alpaca는 LLaM A 모델로 첫 번째 문제를 해결했고, self-instruct로 두번째 문제를 해결했다고 밝혔다.

Self-instruct는 그림4과 같이 적은 수의 seed tasks로 새로운 인스턴스를 생성하는 방법이다. seed tasks는 LLM 의 프롬프트로 제공되며 이를 통해 적은 수의 시드 데이터로 fine tuning에 필요한 대규모 데이터셋을 저렴하게 만들 수 있다.

```
{
    "instruction": "Identify the odd one out.",
    "input": "Twitter, Instagram, Telegram",
    "output": "Telegram"
},
```

[그림 3] Alpaca 데이터셋 예시



[그림 4] Overview of Self-Instruct

2. 설계 상세화 및 변경 내역

2.1. 데이터셋 구축

Self-instruct로 데이터셋을 만들기 위해 먼저 seed_tasks.json을 만들어야 한다.

```
{ "id" : "seed_task_0" , "name" : "recommend_menu" , "instruction" : "편의점 간식 추천해줘" , "instances" : [{ "input" : "" , "output" : "젤리나 초콜릿은 어떠세요?" }], "is_classification" : false }
```

위와 같은 형태로 130여개의 seed_task를 만들고 Self-Instruct: Aligning LM with Self Generated Instructions 를 참고하여 메뉴 추천에 관한 질문-답변 쌍으로 이루어진 한국어 대화 데이터들을 생성할 수 있다.

```
def encode_prompt(prompt_instructions):
    """Encode multiple prompt instructions into a single string."""
    prompt = open("./prompt.txt").read() + "\n"

    for idx, task_dict in enumerate(prompt_instructions):
        (instruction, input, output) = task_dict["instruction"], task_dict["input"], task_dict["output"]
        instruction = re.sub(r"\s+", " ", instruction).strip().rstrip(":")
        input = "<noinput>" if input.lower() == "" else input
        prompt += f"###\n"
        prompt += f"{idx + 1}. Instruction: {instruction}\n"
        prompt += f"{idx + 1}. Input:\n{input}\n"
        prompt += f"{idx + 1}. Output:\n{output}\n"
    prompt += f"###\n"
    prompt += f"{idx + 2}. Instruction:"
    return prompt
```

▲ 프롬프트 인코딩

```
results = []
for message in messages:
        try:
            response = openai.ChatCompletion.create(
                model=model,
                messages=message,
                max_tokens=target_length,
                temperature=temperature,
                top_p=top_p,
                frequency_penalty=frequency_penalty,
                presence_penalty=presence_penalty,
                stop=stop,
                n=n,
            )

        except openai.error.OpenAIError as e:

            if "Please reduce the length of the messages or completion" in str(e):
                target_length = int(target_length * 0.8)
                print(f"Reducing target length to {target_length}")
            else:
                print(f"Retrying in {backoff_time} seconds...")
                time.sleep(backoff_time)
                backoff_time *= 1.5
            count += 1
        results.append(response["choices"][0])
```

▲ChatGPT API 호출

또한 챗봇이 메뉴 추천뿐만 아니라 간단한 음식 관련 질문 및 메뉴의 영양정보에도 응답 할 수 있도록 네이버 지식인 best에서 음식 관련 질문-답변을 가져오고 AI Hub의 음식 이미지 및 영양정보 텍스트 데이터에서 주요 외식 메뉴의 칼로리 정보를 가져와 데이터셋에 활용하기로 하였다.

2.2. QLoRA 학습

기존에는 LoRA 방식으로 학습을 진행하려 했지만 Colab Free에서 제공되는 GPU의 성능을 고려할 때 학습 중 VRAM이 터지는 상황이 생길 수도 있을 것 같아 LoRA보다 메모리 효율적인 QLoRA를 사용하여 학습시키기로 하였다.

QLoRA(Quantized Low Rank Adapters)는 메모리 사용량을 크게 줄이는 fine tuning 접근법으로, 전체 16비트 fine tuning 성능을 유지하면서 낮은 RAM을 가진 GPU에서 더욱 거대한 매개변수를 가진 모델을 fine tuning 할 수 있다. RTX3090 24GB에서 33B 모델을 finetuning 할 수 있다고 한다. 이를 위해 QLoRA는 몇 가지 기능을 제공한다.

1) 4-bit NormalFloat : QLoRA의 가장 핵심적인 방법론으로 PLM 가중치가 4비트로 양자화된 채 저장되며 그 데이터 타입이 NormalFloat이다.

2) Double Quantization : 양자화 상수를 양자화하여 평균 메모리 설치 공간을 줄인다.

3) Paged Optimization : GPU가 사용하는 VRAM 페이지를 CPU RAM에도 일부 저장할 수 있게 할당해주는 기술이다.

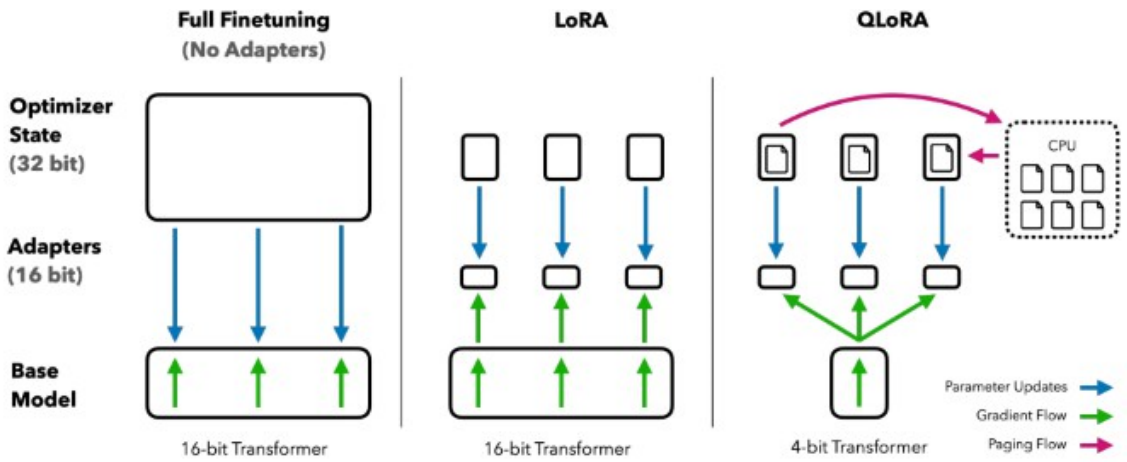


Figure 1: Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.

[그림 5] QLoRA Mechanism

3. 갱신된 과제 추진 계획

중간 보고서 이후 계획은 다음과 같다.

1. Self-instruct를 이용한 지속적인 데이터셋 업데이트
2. 추가 fine-tuning
3. Streamlit, React 등 웹 framework 활용한 프론트/백엔드 구현

<table>
<tr><td>2월</td><td>3월</td><td colspan="2">4월</td><td>5월</td></tr>
<tr><td>데이터 수집 및 전처리</td><td>모델 파인튜닝</td><td colspan="2">데이터 추가 확보 및 fine tuning
웹 어플리케이션 구현</td><td>테스트</td></tr>
<tr><td>착수 보고서</td><td>중간 보고서</td><td colspan="3">최종 보고서</td></tr>
</table>

4. 구성원별 진척도

이름	역할
김도훈	학습 데이터 수집 및 전처리 LLM(KoAlpaca) 파인튜닝 및 학습 결과 시각화
이보원	학습 데이터 수집 및 전처리 UI/UX 디자인

5. 보고 시점까지의 과제 수행 내용 및 중간 결과

5.1. 데이터셋 구축 및 전처리

Self-instruct로 데이터셋을 만들기 위해 seed_task.json 파일을 만들고 Self-Instruct: Aligning LM with Self Generated Instructions 을 참고하여 500 여개 이상의 메뉴 추천 대화를 생성하였고, AI Hub의 음식 이미지 및 영양정보 텍스트의 주요 외식메뉴 칼로리 정보를 추가하였다. 또한 네이버 지식인 best 질문글에서 음식 관련 답변들을 크롤링해서 추가하였다. 현재 기준 600 row 이상의 한국어 대화 데이터셋을 구축했으며 huggingface에서 관리하고, 계속해서 업데이트 할 예정이다.

instruction string · lengths 4 302	input string · classes 25 values	output string · lengths 9 903
삼계탕은 어떤 재료로 만드나요?		삼계탕은 닭고기, 대추, 밤, 마늘, 생강 등을 사용하여 맑은 육…
닭가슴살 요리법 좀 알려줄래?		닭가슴살을 사용한 요리는 다양한데, 그 중에서 간단한 방법으로…
부대찌개 만드는 법 좀 알려줘		부대찌개 만드는 법은 물에 된장과 고추장, 고기를 넣고 삶아 육…
샐러드 어떤 재료로 만들어야 맛있을까		샐러드는 식재료를 잘 고르는 것이 중요한데, 양상추, 토마토, …
떡은 어떻게 만드는지 알려줘		떡 만드는 방법은 밥을 갈아서 반죽을 만든 뒤 원하는 모양으로 …
계란 후라이는 어떻게 만드는 건가요?		계란 후라이는 달걀을 틀에 넣고 기름을 두루 덮어 중약불에서 익…

[그림 6] 데이터셋 preview

5.2. 모델 fine tuning

fine tuning은 Google Colab 에서 진행하였고, GPU는 NVIDIA Tesla T4 1대이다.
필요한 라이브러리는 아래와 같다.

1) bitsandbytes : 자료형을 조정하는 양자화 기법이 구현되어 있다.

2) transformers : 모델을 만들고 사용할 수 있는 다양한 기능 제공

3) peft : LoRA를 간단하게 할 수 있도록 해준다.

4) accelerate : GPU 활용율을 올려준다.

5) datasets : huggingface에서 데이터셋을 로드하고 전처리 할 수 있게 해준다

```
from datasets import load_dataset

data = load_dataset("DOHUN99/mechuri")
```

학습에 사용할 데이터셋을 다운로드해준다.

```
Generating train split: 616/0 [00:00<00:00, 5411.68 examples/s]
```

616 row의 데이터셋이 다운로드 되며 챗봇에 학습시키기 위한 형식으로 매핑해준 후 학습을 진행한다.

```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig

model_id = "beomi/KoAlpaca-Polyglot-5.8B"
bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.bfloat16
)

tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(model_id, quantization_config=bnb_config, device_map={"":0})
```

기본 모델인 KoAlpaca-Polyglot-5.8B의 huggingface 레포지토리를 확인해보면 모델들이 약 1GB씩 쪼개져 저장되어 있는 것을 볼 수 있다. 이렇게 하면 적은 RAM으로도 모델을 로드 할 수 있게 된다. 또한 앞서 QLoRA에 대해 설명했던 것처럼 quant_type이 4-bit NormalFloat인걸 확인할 수 있다.

```
from peft import prepare_model_for_kbit_training

model.gradient_checkpointing_enable()
model = prepare_model_for_kbit_training(model)
```

peft를 통해 Low bit 학습을 준비한 후 LoRA 파라미터를 정의한다.

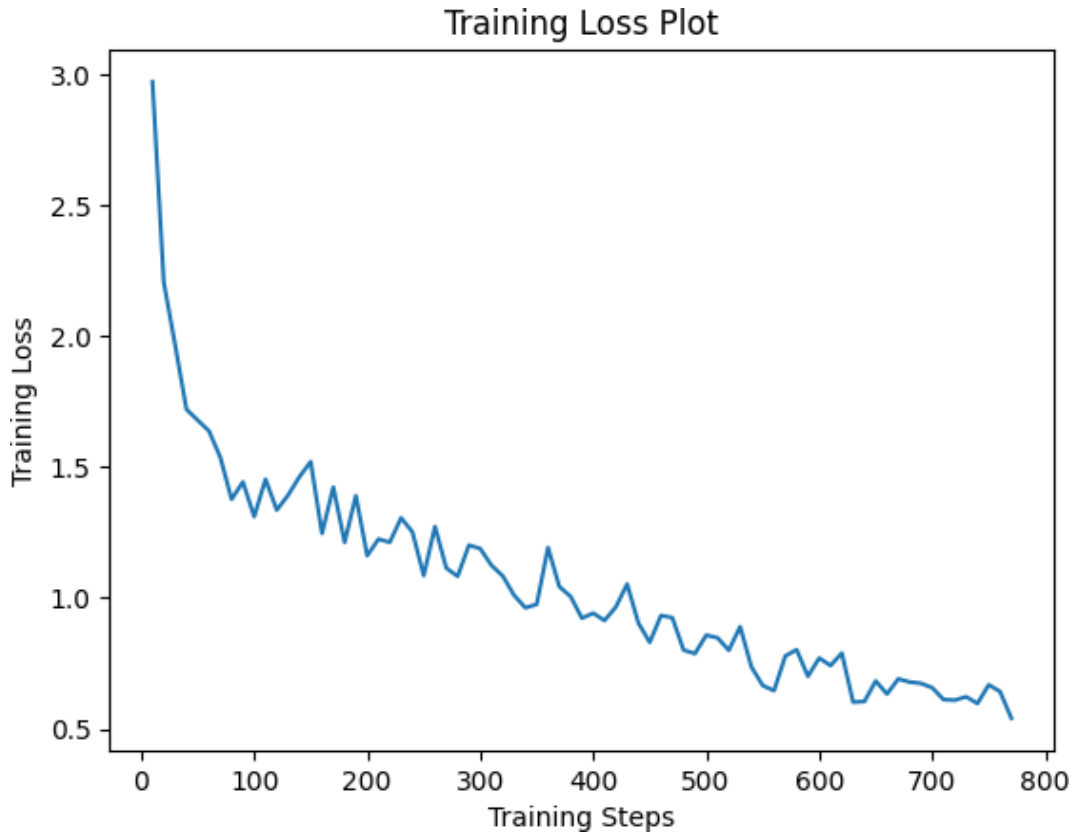
```
        per_device_train_batch_size=8,
        gradient_accumulation_steps=1,
        max_steps=770,
```

batch size는 8, max_steps는 770으로 6160개의 데이터(10 epoch)를 학습시켰다.

```
[770/770 48:07, Epoch 10/10]
```

10 epoch 학습에 약 48분이 소요되었다. 추후 데이터셋 업데이트와 추가 학습을 진행할 예정이다.

pandas와 matplotlib을 이용해 Training loss를 시각화 하였다.



[그림 7] Training Loss Plot

최적의 손실 값은 문제와 데이터의 성격에 따라 다르지만 1 epoch 학습 이후 loss율이 크게 떨어졌고 step값이 늘어날수록 꾸준히 하향하는 그래프를 그렸다. 데이터를 추가적으로 확보하고 step을 늘려 학습한다면 loss 값이 0에 수렴하게 될 것으로 기대된다.

5.3. 텍스트 생성

```
def gen(x):
    q = f"### 질문: {x}₩n₩n### 답변:"
    # print(q)
    gened = model.generate(
        **tokenizer(
            q,
            return_tensors='pt',
            return_token_type_ids=False
        ).to('cuda'),
        max_new_tokens=200,
        early_stopping=True,
        do_sample=True,
        eos_token_id=2,
    )
    print(tokenizer.decode(gened[0]))
```

답변을 생성하는 함수는 위와 같다.

max_new_tokens 값이 커질수록 긴 문장을 끊김없이 생성해 내지만 생성 속도가 느려진다.

```
gen('다이어트에 좋은 아침 메뉴 추천해줘')
```

```
Setting `pad_token_id` to `eos_token_id`:2 for open-end generation.
### 질문: 다이어트에 좋은 아침 메뉴 추천해줘

### 답변: 다이어트에 좋은 아침에는 요거트, 과일, 견과류가 좋을 것 같아요.
```

```
gen('후라이드 치킨 칼로리 알려줘')
```

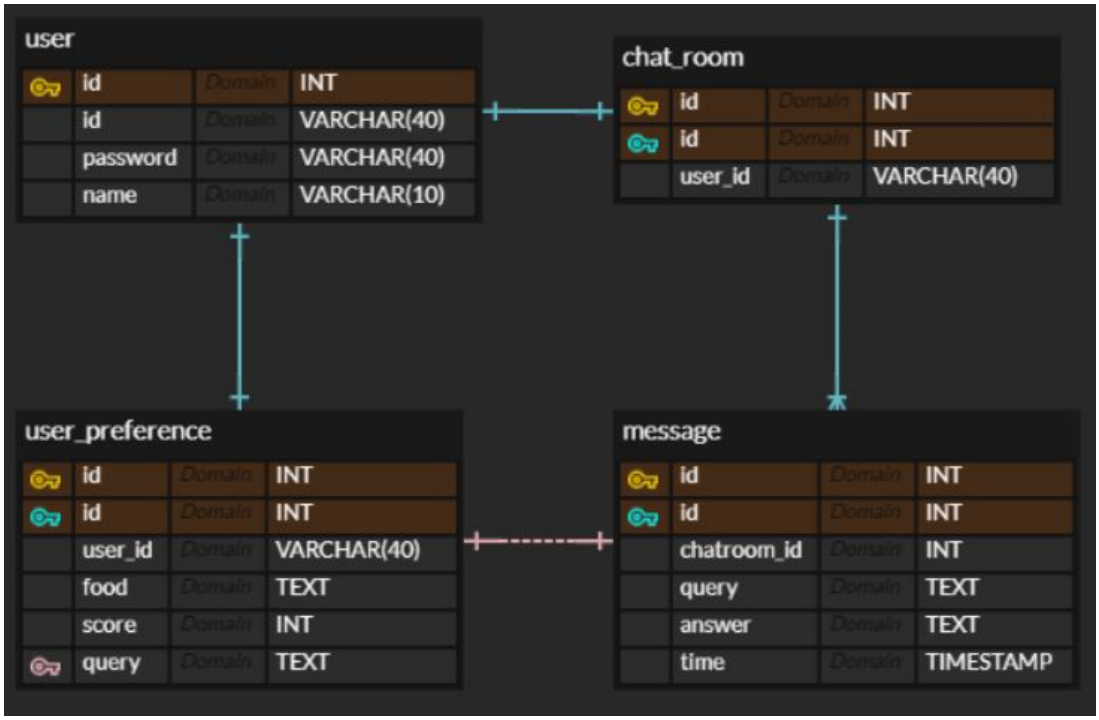
```
/usr/local/lib/python3.10/dist-packages/transformers/generat
  warnings.warn(
Setting `pad_token_id` to `eos_token_id`:2 for open-end gene
### 질문: 후라이드 치킨 칼로리 알려줘

### 답변: 후라이드치킨은 일반적으로 900g당 약 900kcal입니다.
```

5.4. 데이터베이스 설계

- 사용자가 회원가입을 하면 user 테이블에 사용자의 아이디, 비밀번호, 이름이 저장된다.

- 사용자는 각자 하나의 채팅방을 사용할 수 있으며, 사용자가 메시지를 보내면 메시지의 내용과 챗봇이 보낸 답변이 message 테이블에 저장된다.
- 사용자가 보낸 메시지를 바탕으로 음식의 선호도가 user_preference 테이블에 저장되어 관리된다.



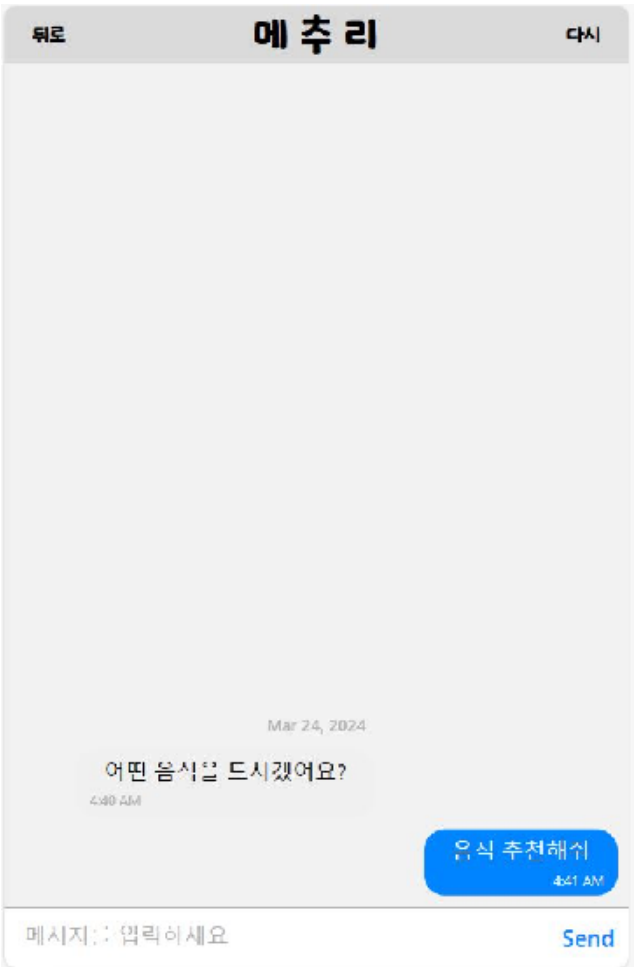
[그림 8] 데이터베이스 설계도(임시)

5.5. UI 디자인

- 메인 화면

[그림 9] 메인 화면

- 채팅방 화면



[그림 10] 채팅방 화면

6. 참고 자료

1. https://crfm.stanford.edu/2023/03/13/alpaca.html

2. https://github.com/Beomi/KoAlpaca?tab=readme-ov-file

3. https://arxiv.org/abs/2212.10560 // Self-instruct Paper

4. https://github.com/yizhongw/self-instruct?tab=readme-ov-file

5.https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=74

6. https://arxiv.org/abs/2305.14314 // QLORA: Efficient Finetuning of Quantized LLMs

7. https://cloud.google.com/vertex-ai/docs/model-garden/lora-qlora?hl=ko

8. https://gist.github.com/Beomi