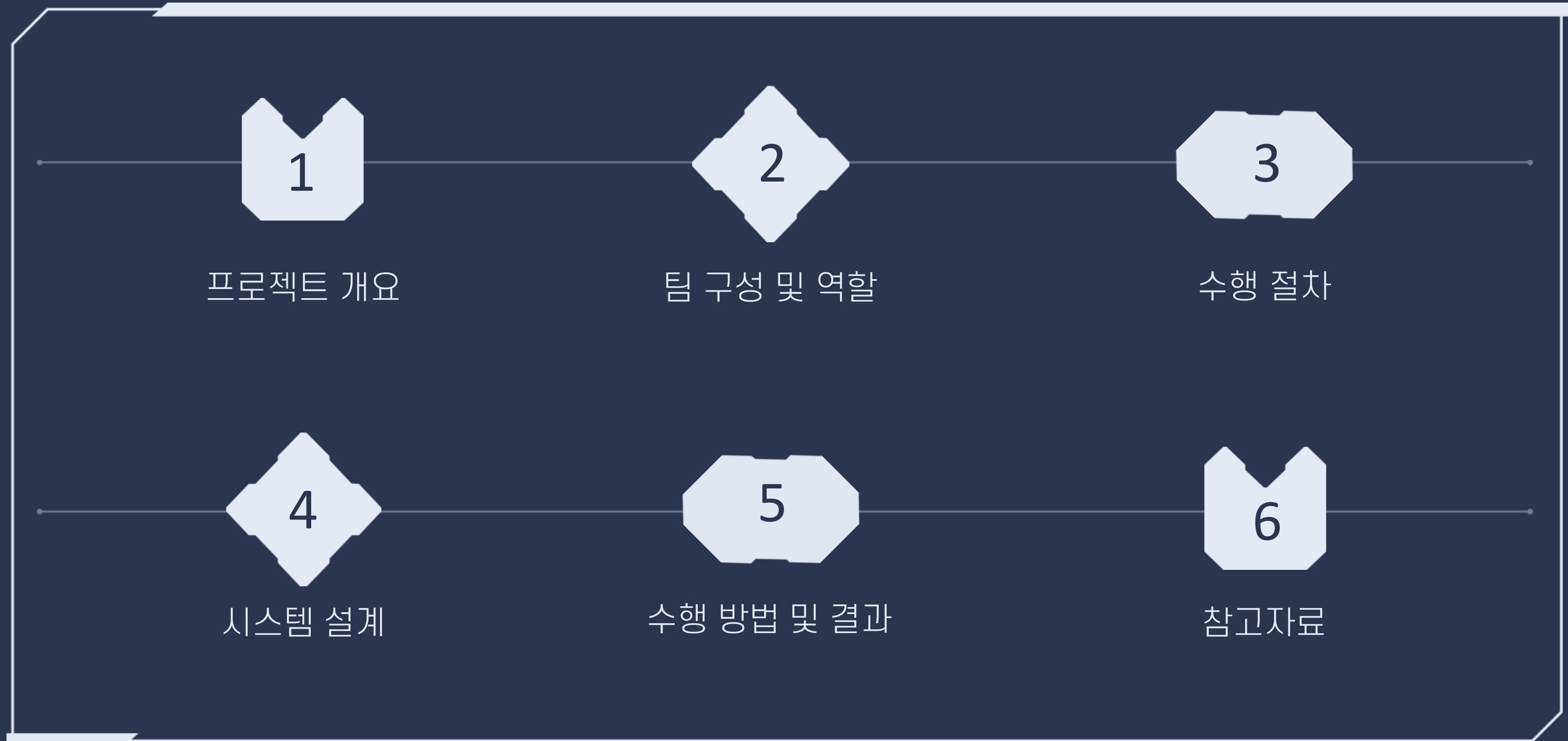


카드핏 (Card-Fit)

사용자 소비 패턴에 딱! 맞는 신용카드 추천 챗봇



✓ 목 차



1. 프로젝트 개요

프로젝트 개요

선정 배경

파이낸셜뉴스 입력 2024.05.21 18:14 수정 2024.05.21 19:04

지갑 속 잠자는 신용카드, 1년새 196만장 늘어

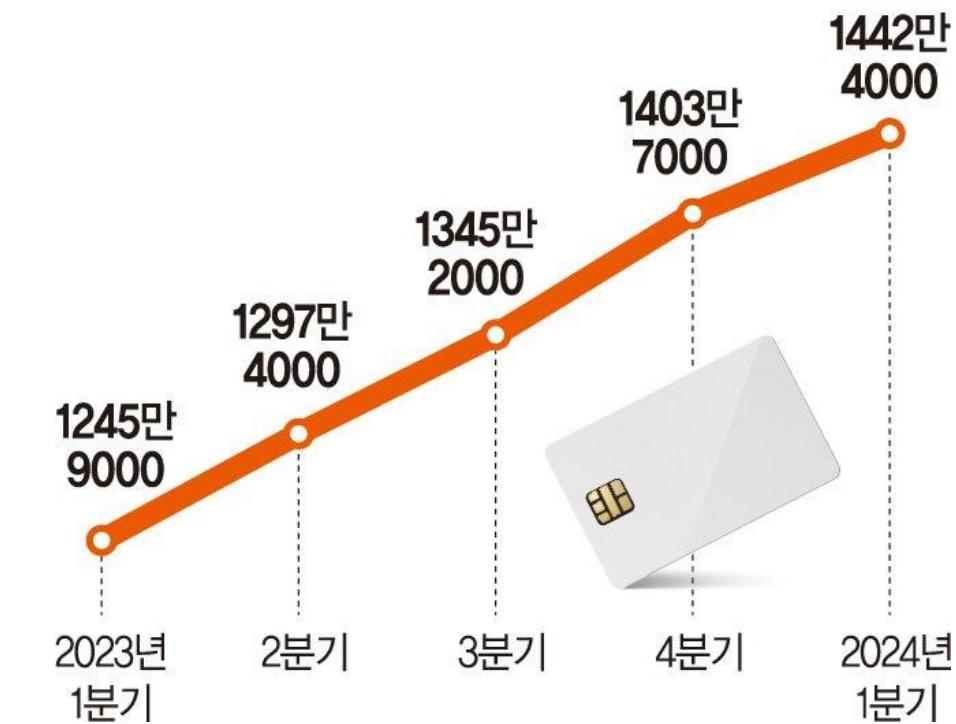
국내 신용카드 산업 이용실적 및 발급수 추이

※ 자료: CEO 스코어



한은 기준금리 동결 기간 중 휴면카드 증가 추이 (단위 : 매) 자료 : 여신금융협회

※ 한은 기준금리 3.5% 동결, 8개 전업카드사 휴면카드 수



휴면 카드 증가로 인한 PainPoint

소비자 부담

- 여러 휴면 카드 보유는 개인 신용점수에 부정적 영향

카드사 비용 부담

- 신규 고객 유치 비용 증가: 고객당 약 15~20만원

보안 위험

- 휴면카드 증가로 분실 및 개인정보 유출 가능성 증가

Insight
→

맞춤형 서비스의 중요성

- 단순한 이벤트성 카드 발급이 아닌,
사용자의 실제 소비 패턴에 맞는 발급 필요

적극적 고객 커뮤니케이션 필요성

- 단순 가입 권유 방식이 아닌,
고객이 혜택을 이해하고 결정하는 능동적 참여 필요

Solution

신용카드 혜택 챗봇



신용카드 혜택 챗봇



영화볼때 혜택있는 카드 알려줘



다음은 영화 할인 혜택이 제공되는 신용카드들입니다. 추천하는 카드는 **일상의 기쁨카드**(IBK 기업은행)과 **부자되세요 홈쇼핑카드 (BC 바로카드)**입니다.

이유: 각각 '영화 할인' 및 '[생활할인 서비스] 영화 3,000원 청구할인(현장/온라인 예매 포함)'의 혜택을 제공하여 영화 관람 비용을 줄일 수 있습니다. 추가로, **삼성 iD CLASSY 카드** (삼성카드), **LG U+ Bora 신한카드 Big Plus** (신한카드), **올리브 Super DC IBK카드** (IBK기업은행), **올바른 FLEX 카드** (NH농협카드) 등도 영화 할인이 제공되는 카드입니다.

챗봇 기반 편의성 제공

- 자연어 처리를 활용한 간편한 혜택 조회
- 복잡한 카드 혜택을 대화형 인터페이스로 쉽게 안내

AI 기반 개인화 카드 추천

- 사용자의 라이프스타일을 분석하여 최적의 카드 추천
- 실시간 데이터 기반 맞춤형 카드 추천

✓ 프로젝트 개요

기대효과

소비자

- 다양한 카드 혜택을 비교하고 이해하기 보다 쉬워짐
- 실제 소비 패턴에 맞는 발급 용이
- 휴면카드로 인한 분실 및 개인정보 유출 가능성 감소

카드사

- 신규 고객 유치 비용 감소
- 카드 사용으로 인한 영업이익 증가

2. 팀 구성 및 역할

✓ TEAM PROFILE

김현일



팀장 / 발표자

전체 시스템 설계 및 구축
웹 검색 구현

남상도



팀원

Data Crawling
Retriever/Reranker 구현

이창민



팀원

Data Crawling
언어모델 선정 및 구현

정호영



팀원

Prompt 설계 및 구현
Webpage 설계 및 구현

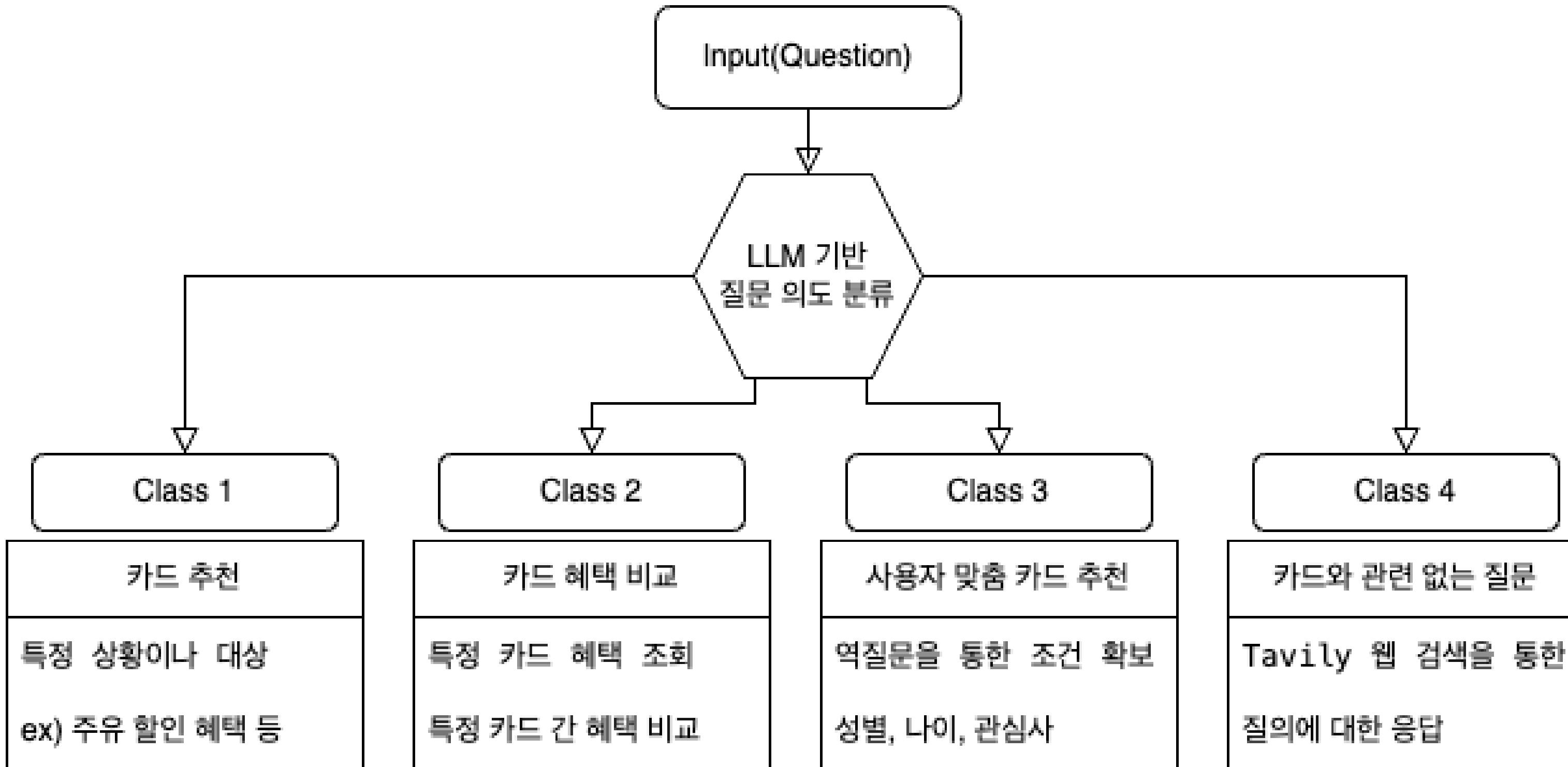
3. 수행 절차

✓ 프로젝트 타임라인

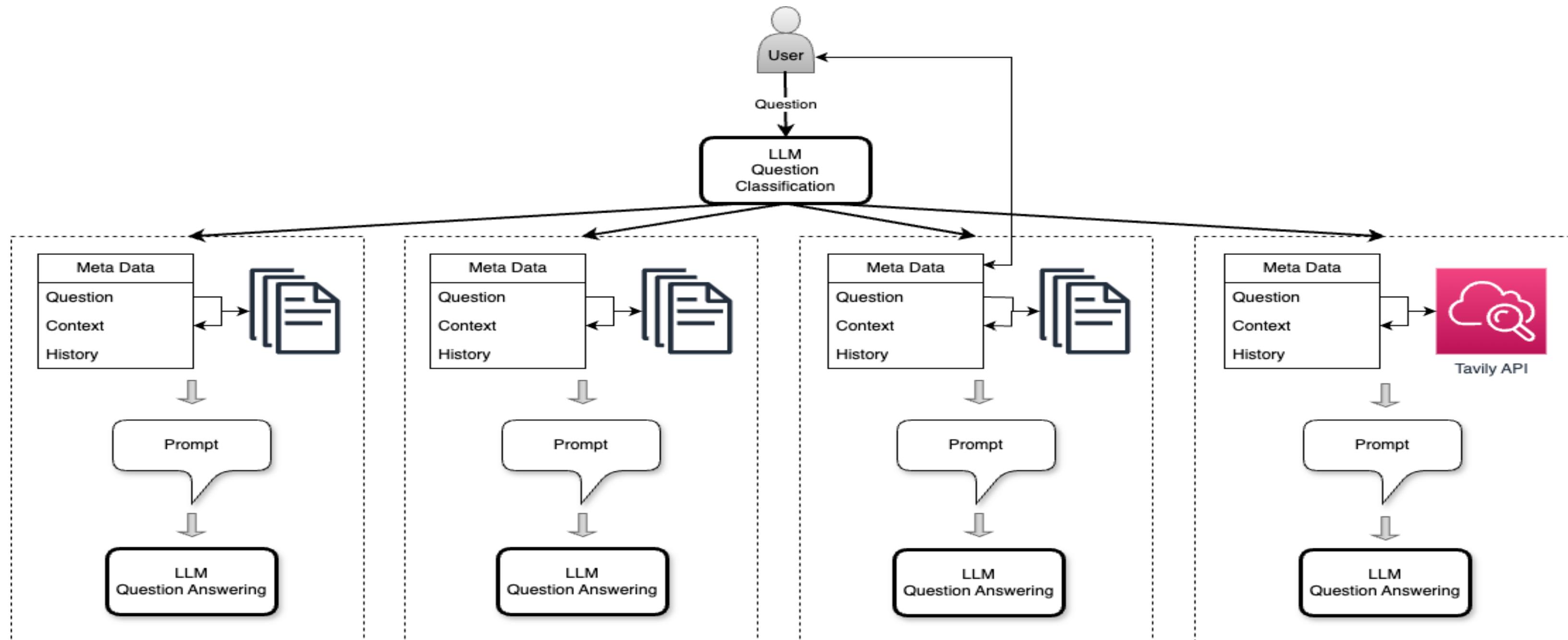
	3/13	3/17	3/19	3/24	3/27	4/1
주제 선정 프로젝트 기획						
데이터 crawling 모델 선별 성능개선 전략 탐색						
성능개선 전략 시도 성능개선 전략 피드백						
최종 성능개선 모델 서빙						
PPT 제작 발표 준비						

4. 시스템 설계

✓ 시스템 설계



✓ 시스템 설계도



 LangChain

5. 수행 방법 및 결과

✓ Data Crawling A

[1. 프로젝트 개요]

- **프로젝트명:** 신용카드 혜택 비교 챗봇을 위한 데이터 크롤링
- **목표:**
 - 카드고릴라 사이트에서 카드 혜택 정보를 자동 수집
 - 구조화된 JSON으로 저장
 - LLM + LangChain + RAG 기반 챗봇에 연동
- **주요 기술스택:** Selenium, Python, FAISS, LangChain, Ollama

[2. 크롤링 기술 도입 과정]

- 1) 초기 접근: requests + BeautifulSoup
JavaScript 렌더링 안됨 → 자세히 보기 버튼이 HTML에 없음
- 2) Playwright (비동기 방식)
일부 카드 미로드 / 팝업 처리 어려움 / 더보기 버튼 비작동
- 3) Selenium으로 최종 전환
 - 동기 방식 / 자바스크립트 렌더링 완벽 지원
 - 팝업/광고 처리 가능 / 안정적인 크롤링

The screenshot shows the CARDGORILLA website interface. At the top, there is a navigation bar with links for 고릴라차트, 카드, 카드사, 캐시백, 콘텐츠, 프리미엄, 마일리지, 호텔, 카드커버, and a search bar. Below the navigation, there are two tabs: 신용카드 and 체크카드. The main content area displays three credit card offers:

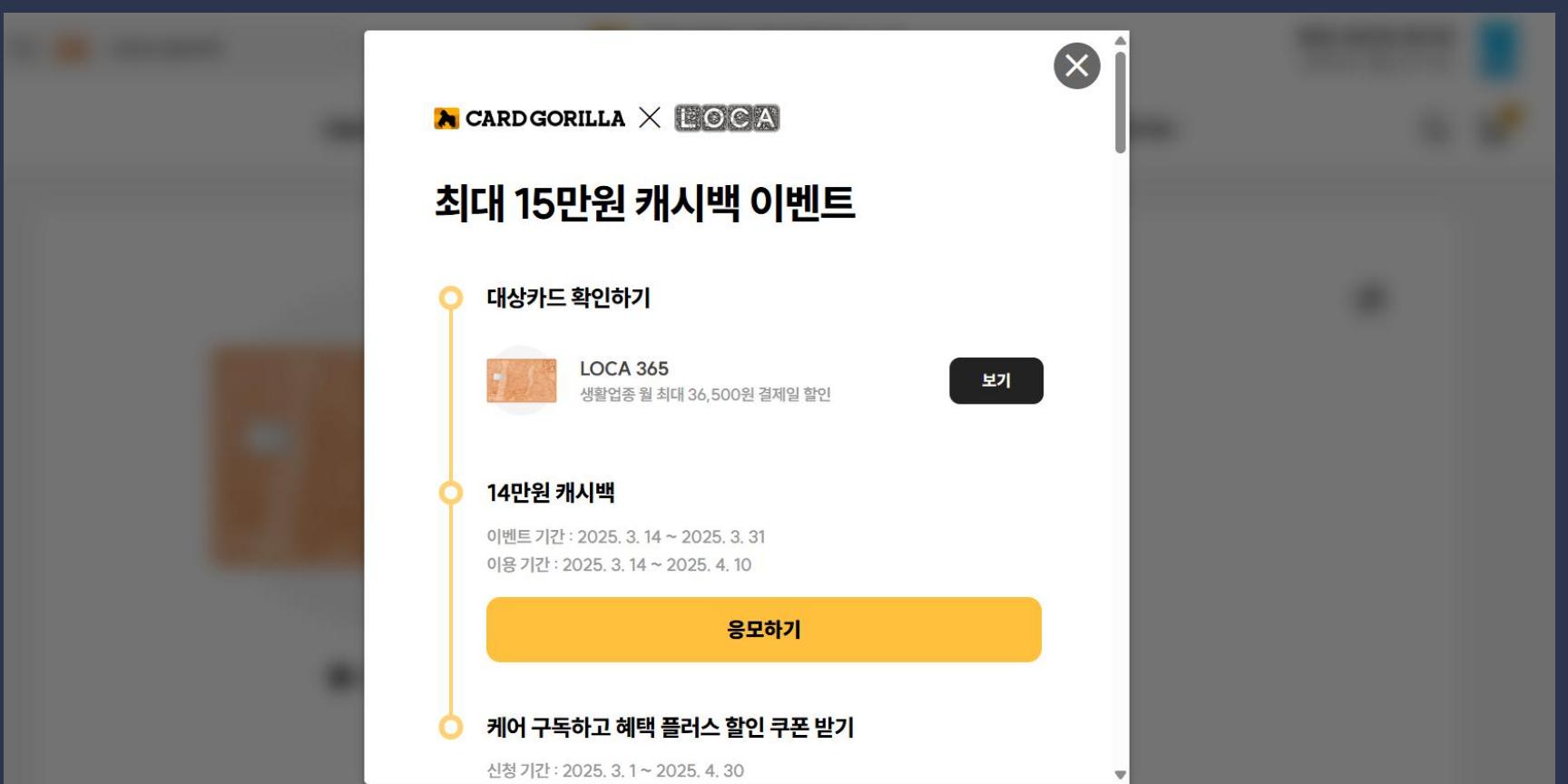
- 신한카드 Mr.Life** (신한카드)
최대 18만원 캐시백
• 공과금 10% 할인
• 마트, 편의점 10% 할인
• 식음료 10% 할인
• 해외겸용 15,000원
• 전월실적 30만원 이상
• 온라인발급 전용 카드
- 삼성카드 & MILEAGE PLATINUM (스카이패스)** (삼성카드)
1,000원당 1마일 기본적립
국내/해외 택1옵션형 1마일 특별적립
인천공항 라운지 무료이용
• 국내전용 47,000원 / 해외겸용 49,000원
• 전월실적 없음
- 삼성카드 taptap** (삼성카드)
최대 20.4만원 혜택
• 스타벅스 50% 할인
• 대중교통 10% 할인
• CGV, 롯데시네마 5,000원 할인

✓ Data Crawling A

```
while True:  
    try:  
        detail_buttons = driver.find_elements(By.CSS_SELECTOR, "a.b_view")  
        start_idx = len(card_db)  
  
        if not detail_buttons:  
            print("[INFO] 더 이상 크롤링할 카드가 없습니다.")  
            break  
  
        for idx in range(start_idx, start_idx + 10):  
            detail_buttons = driver.find_elements(By.CSS_SELECTOR, "a.b_view")  
            if idx >= len(detail_buttons):  
                break
```

a.b_view: 카드 상세 페이지로가는 "자세히 보기" 버튼 리스트

```
# 팝업 닫기  
try:  
    popup_close = WebDriverWait(driver, 2).until(  
        EC.element_to_be_clickable(By.CSS_SELECTOR, "button.swal2-close"))  
    )  
    popup_close.click()  
    time.sleep(0.5)  
except TimeoutException:  
    pass  
  
try:  
    close_button = driver.find_element(By.CSS_SELECTOR, "div.btn_md_close")  
    close_button.click()  
    time.sleep(0.5)  
except NoSuchElementException:  
    pass  
  
# 광고창 감지 닫기  
all_windows = driver.window_handles  
if len(all_windows) > 1:  
    for w in all_windows:  
        if w != driver.current_window_handle:  
            driver.switch_to.window(w)  
            driver.close()  
driver.switch_to.window(driver.window_handles[0])
```



- 이벤트 팝업 (.swal2-close) : swal2는 SweetAlert2라는 자바스크립트 팝업 라이브러리
- 추가 정보 팝업 (div.btn_md_close)
- 자동으로 감지 후 닫고 크롤링 계속 진행

✓ Data Crawling A

```
benefit_blocks = driver.find_elements(By.CSS_SELECTOR, "dl")
for block in benefit_blocks[:-1]:
    try:
        keyword = block.find_element(By.CSS_SELECTOR, "p.txt1").text.strip()
        desc = block.find_element(By.CSS_SELECTOR, "i").text.strip()

        if keyword:
            # 키워드 중복 제거용 (리스트에는 append 전에 검사)
            if keyword not in keyword_list:
                keyword_list.append(keyword)

            # 혜택 설명 병합 처리
            if keyword in benefit_desc:
                if desc not in benefit_desc[keyword]:
                    benefit_desc[keyword] += ", " + desc
                else:
                    benefit_desc[keyword] = desc
            else:
                benefit_desc[keyword] = desc

    except NoSuchElementException:
        continue
except NoSuchElementException:
    pass

card_db[card_name] = {
    "카드 회사": card_company,
    "혜택 키워드": keyword_list,
    "혜택 설명": benefit_desc}
```

카드 상세정보 크롤링

- 카드 이름: strong.card
- 카드 회사: p.brand
- 혜택: dl > p.txt1, i 조합으로 키워드와 설명 추출
- 마지막 dl은 제외 (불필요한 정보 방지)

	공과금	월납요금(공과금) 10% 할인서비스
	편의점	편의점 10% 할인
	병원/약국	병원/약국업종 10% 할인
	생활	세탁소 업종 10% 할인
	온라인쇼핑	온라인 쇼핑 10% 할인
	택시	택시 10% 할인
	푸드	식음료 10% 할인
	대형마트	3대 마트 10% 할인
	주유소	4대 정유사 리터당 60원 할인

```
"신한카드 Mr.Life": {
    "카드 회사": "신한카드",
    "혜택 키워드": [
        "공과금",
        "편의점",
        "병원/약국",
        "생활",
        "온라인쇼핑",
        "택시",
        "푸드",
        "대형마트",
        "주유소"
    ],
    "혜택 설명": [
        "공과금": "월납요금(공과금) 10% 할인서비스",
        "편의점": "편의점 10% 할인",
        "병원/약국": "병원/약국업종 10% 할인",
        "생활": "세탁소 업종 10% 할인",
        "온라인쇼핑": "온라인 쇼핑 쇼핑 10% 할인",
        "택시": "택시 10% 할인",
        "푸드": "식음료 10% 할인",
        "대형마트": "3대 마트 10% 할인",
        "주유소": "4대 정유사 리터당 60원 할인"
    ]
}
```

크롤링 딕셔너리형 구조 (JSON)

✓ Data Crawling B

데이터셋 출처: 카드서퍼 (<https://www.card-surfer.com/>)

카드고릴라, 뱅크샐러드와 비교했을 때, 상세 페이지가 정적으로 구현되어 있음

1. 크롤링 과정요약

- 상세페이지 주소 가져오기 (셀레니움)
- 카드 이름과 혜택 가져오기 (BeautifulSoup)
- 태그 기준으로 혜택 분류하기 (정규표현식)

2. 이슈

- 카드 상세 부분이 기호/표/소제목 등 다양한 태그로 구성되어, 각 태그마다 처리 코드를 작성하는데 많은 시간이 소모됨.
- 결국, 상세 내용은 태그만 삭제한 채, 텍스트만 가져옴.



The screenshot shows a dropdown menu titled "혜택 선택" (Select Benefit) with the following content:

- 4개의 정유사 중 직접 고른 1개 정유사 주유 이용금액 10% 결제일 할인

Below the dropdown is a table with three columns (A, B, C) showing benefit details for various cards. The table includes rows for Samsung iD STATION Card (GS칼텍스), 우리 기후동행카드 (신용), and KUKU X LOCA. The "CSV" button is highlighted in blue.

A	B	C
1 name	benefit	detail
12 삼성 iD STATION 카드 (GS칼텍스)	GS&POINT 적립 서비스	GS&POINT 적립 서비스 정보 GS&
13 삼성 iD STATION 카드 (GS칼텍스)	유의사항	서비스 제공 공통 기준에 대한 설명 전월 이
14 우리 기후동행카드(신용)	국내 가맹점 0.8% 할인	할인 프로그램 안내 항목 상세 내용 할인
15 우리 기후동행카드(신용)	후불 기후동행카드 서비스 이용 안내	후불 기후동행카드 안내 후불 기후동행카드
16 우리 기후동행카드(신용)	유의사항	실적 산정 방법 안내 항목
17 쿠쿠 X LOCA	쿠쿠 렌탈료 결제 시 결제일 할인	쿠쿠 렌탈료 결제 할인 혜택
18 쿠쿠 X LOCA	장기 할부 서비스 및 캐시백	장기 할부 서비스 항목 내용

Below the table, there are sections for "정비소: SpeedMate" (with points about SpeedMate usage and online payment), "주차장" (with points about 10% discount at all parking lots and availability at designated locations), and "차량서비스 유의 사항" (with points about Deep Oil service terms).

✓ Data Crawling A vs B

데이터 선정A (카드고릴라)로 RAG 구축

A: 카드고릴라

- 더 많은 카드 데이터 수 확보.
- JSON 형식으로 FAISS 벡터화 과정이 빠름.
- 카드 단위로 잘 구조화되어 있음 (카드사/혜택/설명 분리)
- 이런 구조는 LLM 기반 검색 및 추천에 벡터화하거나, RAG 기반 질의응답 시스템에 넣기에 매우 적합.

B: 카드서퍼

- CSV 형식으로 FAISS 벡터화 과정이 느림.
- 상세 부분을 태그만 제거한 채 가져왔을 때,
모델이 제대로 이해하지 못함.

상세 부분) 온라인 쇼핑몰 결제일 할인 정보 항목 내용 할인율
5% 할인 대상 쿠팡 네이버플러스 스토어 SSG.COM 롯데ON G
마켓 옵션 11번가 삼성카드 쇼핑 통합 월 할인한도 5,000원
조건 전월 이용금액이 40만원 이상일 경우 제공 발급월 + 1개월까
지는 전월 이용금액이 40만원 미만이어도 제공

✓ 모델 선정 방식

```
from langchain.llms import Ollama
llm = Ollama(model="mistral:latest")
test_questions = [
    "교통비 절감에 좋은 카드는?",  

    "배달앱 혜택이 많은 신용카드는?",  

    "여행할 때 공항라운지 혜택이 있는 카드 추천해줘.",  

    "쿠팡 할인되는 카드 알려줘.",  

    "가장 높은 캐시백 제공하는 카드 알려줘."
]

for question in test_questions:  

    response = llm(question)  

    print(f"🚀 질문: {question}\n→ 답변: {response}\n")
```

모델명	응답 요약	응답 평가
mistral:latest	교통비 관련 카드 유형을 한국어로 논리적으로 분류해 설명	탁월 - 명확한 분류 + 자연어 표현
llama3:latest	영어로 미국 카드 중심 추천 (Chase, Capital One 등)	잘 구성되었지만 한국 사용자에 부적합
nous-Hermes	일반론적 조언만 제공, 구체적 카드명 없음, 영어	너무 포괄적, 실질적 도움 부족
korean-blossom-8b	질문과 무관한 문장(여행, 음식, 운동 등 취향 선택)	완전히 비문맥적, 챗봇 기능 불가

🚀 질문: 교통비 절감에 좋은 카드는?

→ 답변: 교통비를 절감하기 위한 카드라면, 다음과 같은 종류가 있습니다.

1. 저렴한 이자 카드: 이자율이 낮거나 0% 이자가 적용되는 카드입니다. 카드 이용료나 이자로 인한 교통비에 대한 부담을 줄일 수 있습니다.

2. 혜택 지급 카드: 전기차 충고, 택시 요금 등 교통비 관련 서비스 사용 시 포인트나 혜택을 제공하는 카드입니다.

3. 저렴한 로엔 대출 카드: 로엔 대출을 할 때 저렴한 이자율을 가진 카드를 사용하면 교통비의 일부 부담을 줄일 수 있습니다.

4. 캐시백 카드: 일정 금액을 카드로 지불하면 일정 금액을 되돌려주는 카드입니다. 이런 카드를 활용하여 교통비에 필요한 금액의 일부를 되돌려받을 수 있습니다.

5. 포인트 리워드 카드: 일정 금액을 지불하거나 일정 가맹점에서 사용하면 포인트를 축적할 수 있으며, 이 포인트는 교통비 관련 서비스로 교환할 수 있습니다. 각 카드의 장단점, 사용 방법에 대해서는 자세히 연구하시면 됩니다.

✓ 주요 기술 스택 & 이유

사용자 질문

↓
[문맥 유지 + 카드 유사 검색]

↓
[LangChain 체인 구성]

↓
[프롬프트 템플릿에 삽입]

↓
[Ollama LLM (mistral)]

↓
최종 응답 생성

구성 요소

LangChain Core

**FAISS +
HuggingFaceEmbeddings**

Ollama + Mistral

ConversationBufferMemory

Custom Prompt Template

사용 이유

체인 연결, 대화 흐름 제어에
최적화

카드 JSON 데이터 벡터화 및
유사도 검색

로컬에서 빠르고 정확한
자연어 처리

대화 문맥 유지

일관된 출력 형식 유지 및
추천 기준 반영

✓ 데이터 가공

```
"신한카드 Mr.Life": {  
    "카드 회사": "신한카드",  
    "혜택 키워드": [  
        "공과금",  
        "편의점",  
        "병원/약국",  
        "생활",  
        "온라인쇼핑",  
        "택시",  
        "푸드",  
        "대형마트",  
        "주유소"  
    ],  
    "혜택 설명": [  
        "공과금": "월납요금(공과금) 10% 할인서비스",  
        "편의점": "편의점 10% 할인",  
        "병원/약국": "병원/약국업종 10% 할인",  
        "생활": "세탁소 업종 10% 할인",  
        "온라인쇼핑": "온라인 쇼핑 쇼핑 10% 할인",  
        "택시": "택시 10% 할인",  
        "푸드": "식음료 10% 할인",  
        "대형마트": "3대 마트 10% 할인",  
        "주유소": "4대 정유사 리터당 60원 할인"  
    ]  
}
```

항목

문서 단위

검색 정밀도

문장 생성 용이성

중복 제거

프롬프트 구성

1.json 구조 특성

- 카드 하나에 여러 혜택이 묶여 있음
- 즉, 1카드 = 1개 JSON 객체
(혜택이 중첩)

문제점

- 벡터 검색 시 "정확한 혜택" 기준
검색 어려움
- 유사도 검색이 카드 전체 문맥
기준으로 오염됨

1.json

카드 기준 (혜택 묶음)

낮음 (다른 혜택까지 포함됨)

불리함

어려움

복잡

```
{  
    "카드 이름": "신한카드 Mr.Life",  
    "카드 회사": "신한카드",  
    "혜택 키워드": "공과금",  
    "혜택 설명": "월납요금(공과금) 10% 할인서비스"  
},  
{  
    "카드 이름": "신한카드 Mr.Life",  
    "카드 회사": "신한카드",  
    "혜택 키워드": "편의점",  
    "혜택 설명": "편의점 10% 할인"  
},  
{  
    "카드 이름": "신한카드 Mr.Life",  
    "카드 회사": "신한카드",  
    "혜택 키워드": "병원/약국",  
    "혜택 설명": "병원/약국업종 10% 할인"  
}
```

2.json

혜택 기준 (1문서 1혜택)

높음 (질문과 정확히 매칭된
혜택)

유리함 (한 줄 요약에 최적)

카드명 기준으로 필터링 가능

간단한 리스트화 가능

2.Json 구조 특성

기존 하나의 카드 → 혜택
단위로 문서 분리

- 카드당 여러 레코드
→ LLM에서 문장 단위로
검색 가능

✓ 주요 기술 스택 & 이유

```
with open("/home/alpaco/1t/changmin/card_llm_ready.json", "r", encoding="utf-8") as f:  
    raw_data = json.load(f)  
  
# === 카드 문서 변환 ===  
docs = [  
    Document(  
        page_content=f"[{item['카드 이름']}]-[{item['카드 회사']}]-[{item['혜택 키워드']}]-[{item['혜택 설명']}]",  
        metadata={  
            "카드 이름": item["카드 이름"],  
            "카드 회사": item["카드 회사"],  
            "혜택 키워드": item["혜택 키워드"],  
            "혜택 설명": item["혜택 설명"]  
        }  
    ) for item in raw_data  
]
```

- RAG 구조에서 필요한 "텍스트 문서화" 과정
- 혜택 설명은 콘텐츠로, 카드명/ 회사/ 키워드는 메타데이터로 사용

```
embedding_model = HuggingFaceEmbeddings(model_name="jhgan/ko-sroberta-multitask")  
  
if os.path.exists("faiss_index"):  
    vectorstore = FAISS.load_local("faiss_index", embedding_model, allow_dangerous_deserialization=True)  
else:  
    vectorstore = FAISS.from_documents(docs, embedding_model)  
    vectorstore.save_local("faiss_index")  
  
retriever = vectorstore.as_retriever(search_type="similarity", search_kwargs={"k": 15})
```

- 한국어 문장 유사도 기반 검색 최적화
- 검색 속도를 높이기 위해 로컬 인덱스 저장

무작위 추천 대신 유사도 기반 "정확한 카드 선별"

✓ 주요 기술 스택 & 이유

4. 프롬프트 템플릿 구성

```
template = """
너는 한국어로 신용카드 혜택을 친절하게 설명하는 전문가야.
| # 추천 신용카드 목록:
{context}

# 이전 대화:
{chat_history}

# 질문:
{question}

# 답변:
"""

prompt = ChatPromptTemplate.from_template(template)
```

- 한국어 문장 유사도 기반 검색 최적화

- 검색 속도를 높이기 위해 로컬 인덱스 저장

5. 대화 메모리 및 히스토리 함수 6. 체인 구성과 실행 구조

```
def get_chat_history(_):
    return "\n".join(
        [f"{msg.type.capitalize()}: {msg.content}" for msg in memory.load_memory_variables({})["chat_history"]]
    )

# === 체인 구성 ===
chain = (
    {
        "context": retriever | RunnableLambda(format_docs),
        "chat_history": RunnableLambda(get_chat_history),
        "question": RunnablePassthrough(),
    }
    | prompt
    | llm
)
```

LangChain의 ConversationBufferMemory로 대화 히스토리 저장 : 문맥 흐름 유지 → 사용자가 후속 질문 시 맥락 이해

질문 → 유사 카드 추출 → 포맷 → 프롬프트 삽입 → LLM 응답

- LangChain의 Runnable 구조로 유연한 체인 구현
- 문맥, 질문, 카드 context 분리 → 유지보수 용이



Prompt Engineering

Prompt engineering 이란?

-LLM이 원하는 답을 보다 정확하게 생성하도록 프롬프트를 최적화하는 기법.

Prompt engineering이 성능 개선에 주효한 이유

1. OpenAI의 프롬프트 엔지니어링 가이드

“Effective prompt engineering can significantly improve the output quality of large language models. By structuring prompts strategically, users can achieve more relevant and coherent responses.”

(출처: OpenAI. *Prompt Engineering Guide*. OpenAI, 2024, platform.openai.com/docs/guides/prompt-engineering.)

2. 구글 클라우드의 프롬프트 엔지니어링 안내서

“Prompt engineering is a powerful technique for optimizing large language models (LLMs) by providing clear, structured instructions to guide the model’s responses.”

(출처: Google Cloud. *What is Prompt Engineering?* Google, 2024, cloud.google.com/discover/what-is-prompt-engineering.)

✓ Prompt Engineering

여러 prompt 기법을 적절하게 혼용하면 보다 효과적이다 –

“The combination of role prompting, few-shot examples, and explicit task instructions significantly improves the performance of large language models, especially in domain-specific question answering tasks.” – *Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review* (arXiv:2310.14735v5)

출처: Zhu, Liting, et al. *Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review*. arXiv preprint arXiv:2310.14735v5, 2024. arXiv.org

Prompt 기법 선정 기준

- 간단하지만 효과적인 방법 (학습 부담↓, 성능 향상↑)
- LLM의 추론력을 최대한 활용할 수 있는 방법
- 특정 도메인(RAG 기반 신용카드 추천)에 적합한 기법

주요 사용 Prompt 기법

Ontology 기반 조건 프롬프트: 카드 추천 기준(예: 연회비, 혜택 차별성 등)을 명시하여, 답변 품질 구체화

역할 프롬프트 (Role Prompting): LLM에게 '신용카드 전문가'라는 역할을 부여하여, 더 일관되고 신뢰성 있는 응답을 유도

Structured 프롬프트: 카드 추천 형식을 제공함으로써, 출력 형태의 일관성과 이해도를 높임

```
57 |     template = """
58 |     너는 신용카드 혜택을 잘 설명해주는 어시스턴트야.
59 |     반드시 한국어로만 답변하고, 영어는 절대 사용하지 마.
60 |     아래 문맥(context)과 이전 대화 내용을 참고해서 사용자 질문에 친절하고 구조화된 형식으로 답변해줘.
61 |
62 |     답변은 다음 형식을 따라줘:
63 |
64 |     1. 카드 이름
65 |        - 주요 혜택 요약
66 |     2. 카드 이름
67 |        - 주요 혜택 요약
68 |
69 | #이전 대화:
70 | {chat_history}
71 |
72 | #질문:
73 | {question}
74 |
75 | #문맥:
76 | {context}
77 |
78 | #답변:
79 | """
```

✓ 분류 Prompt

Prompt 설정 도중 대면한 문제점

- Structured prompt를 사용 시 챗봇의 답변 품질이 크게 향상되지만 사용자의 질문 유형 변화에 유연하게 대처하지 못하는 현상 발생
(i.e. '20대 대학생에게 좋은 카드 추천해줘' 와 'A카드와 B카드 혜택 비교해줘')

따라서 사용자의 질문을 다른 카테고리로 분류하고 각 카테고리별로 별도의 structured prompt를 설정함으로써 해당 기법의 장점을 취하고 문제점을 해소

위 과정에서 사용자에게 역으로 질문을 하여 사용자에게 맞는 맞춤형 카드를 추천하는 카테고리 및 카드 또는 혜택과 관련이 없는 질문을 필터링하는 카테고리 또한 신설

```
분류 prompt_initial draft

# 질문 분류 프롬프트
classification_prompt = ChatPromptTemplate.from_template(
    """
    다음 사용자 질문을 읽고 아래 4개의 카테고리 중 가장 알맞는 번호만 반환해줘 :
    1. 특정 상황이나 대상에 맞는 카드 추천
    2. 특정 카드 혜택 조회 또는 카드 간 비교
    3. 사용자 맞춤 카드 추천 (역질문 필요)
    4. 카드와 무관한 질문

    질문: {question}
    """
)
```

✓ 분류 Prompt

초기 분류 prompt 문제점

3번과 4번 카테고리를 잘 구분하지 못하여 명백히 3번 카테고리로
분류되어야 할 질문을 4번 카테고리로 분류하는 문제 발생

개선방안

- 분류 prompt에 명확한 예시 추가
- prompt 수정: “웬만하면 4번은 피하라”는 유도 문구 추가 및
4번은 카드와 완전 무관한 잡담일 때만 쓰도록 유도
- prompt를 자연어 질의 응답 형식으로 변경

final draft

```
# 질문 분류 프롬프트
classification_prompt = ChatPromptTemplate.from_template(
    """
    다음 사용자 질문이 어떤 목적에 해당하는지 판단해줘.
    총 4가지 유형이 있어. 설명과 예시를 잘 참고해서 가장 적절한 번호 하나만 골라줘.
    답변은 번호 하나만 줘 (예: 1)

    가능한 경우 1~3번 중에서 선택하려고 노력해줘.
    4번은 카드랑 정말 관련 없는 잡담일 때만 골라줘.
    """)
```

[카테고리 유형]

1. 특정 상황이나 대상에 맞는 카드 추천
예시:
- 주유소 할인되는 카드 알려줘
- 20대 여성 직장인에게 좋은 카드 추천해줘
- 카페 자주 가는 사람에게 혜택 좋은 카드 뭐 있어?
- 온라인 쇼핑 자주 하는 사람에게 맞는 카드 알려줘
- 병원 자주 다니는 사람에게 유리한 카드 있어?
- 택시 할인되는 카드 알려줘
- 여행 자주 가는 사람에게 좋은 카드 추천해줘

2. 특정 카드 혜택 조회 또는 카드 간 비교
예시:
- 삼성카드 the O 혜택 뭐야?
- 현대 the Green이랑 신한 Deep Oil 비교해줘
- 신한 Air Platinum 카드 혜택 알려줘
- Deep Oil이랑 Deep On 카드 차이점은 뭐야?
- 이디야 할인되는 카드가 뭐야?

3. 사용자 맞춤 카드 추천 (역질문 필요)
예시:
- 나한테 맞는 카드 추천해줘
- 어떤 카드가 나랑 잘 맞을까?
- 카드 하나 만들까하는데 뭐가 좋을까?
- 나 요즘 카드 알아보고 있어, 뭐가 괜찮아?
- 카드에 대해 잘 모르는데 추천해줄 수 있어?
- 신용카드 하나 만들건데 나한테 맞는 걸로 추천해줘!
- 아무 카드나 쓰기 싫은데 뭐가 좋을까?
- 혜택 많은 카드보단 나한테 잘 맞는 카드가 필요해
- 요즘 카드 뭐가 잘 나가? 나한테 어울리는 거 추천해줘
- 처음 카드 만들건데 어디서부터 시작해야 할까?
- 복잡한 건 싫은데 간단하고 내 생활에 맞는 카드 없어?

4. 카드와 무관한 질문 (잡담 등)
예시:
- 오늘 날씨 어때?
- 너는 누구야?
- 심심한데 뭐 하지?
- 밥 뭐 먹을까?
- 넌 몇 살이야?

질문: {question}
답변 (번호만):
""")

✓ Retriever & Reranker

Retriever: 사용자 질문(Query)를 받아 RAG(FAISS) 검색

검색방식	Similarity (채택)	Similarity_score_threshold	MMR
	신용카드 혜택 특성상, <u>다양성보다는 정확성이</u> 중요하기 때문에 채택	'최소 유사도' 설정 가능 (score_threshold : 0.01) 유사도를 최소로 할때 만족	'다양성' 설정 가능 (lambda_mult : 0.01) 다양성을 최소로 할때 만족

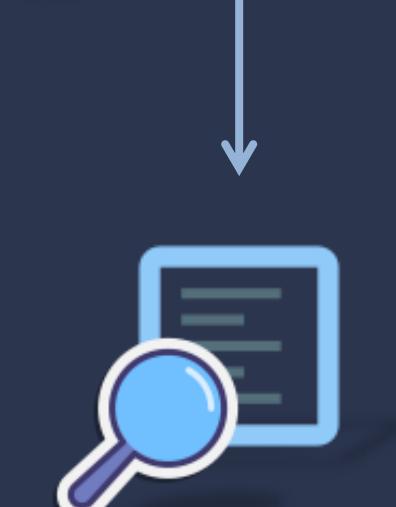
ReRanker: Retriever 검색 결과를 더 정교하게 비교 및 재정렬

모델	Dongjin-kr/ko-reranker (채택)	ms-kr-distilroberta-v1
학습데이터	한국어 데이터로 추가 Fine-tuning	KLUE, 한국어 텍스트
모델크기	약 110M 파라미터	약 66M 파라미터
주요목적	질문-문서 관련성 평가	문장 간 유사도 평가
선정기준	질문한 혜택이 거의 포함	질문한 혜택이 간혹 포함되지 않음

RAG (FAISS)
- 약 3,000 개



Retriever
- fetch_k: 750
- k: 75



ReRanker
- Docs: 15

✓ Retriever & Reranker

Reranker 적용이 더 나은 정보 제공

Retriever	Retriever + ReRanker (채택)
<ul style="list-style-type: none">- 750개 선별 -> 15개 추출- 무조건 10개를 추출해서, LLM에 전달.- 사용자 질문(Query)의 단어가 포함된 혜택을 선택.- 카드사별 '표현 어휘'에 의존하여 <u>같은 회사의 카드를 계속 추천하게 됨.</u>	<ul style="list-style-type: none">- 750개 선별 -> 75개 비교 -> 15개 추출- 10개까지 추출해서, LLM에 전달. (모델이 판단하여 10개 미만으로 전달할 때도 있음)- 보다 자세한 설명이 있는 혜택을 선택.- <u>비교적 다양한 회사의 카드를 추천함.</u>

✓ Retriever & Reranker

결과 비교

질문: 무이자 할부가 긴 카드 추천해줘	
Retriever	Retriever + ReRanker (채택)
<ol style="list-style-type: none">1. LOCA 365 카드 (롯데카드) - 국내 가맹점 2~3개월 무이자 할부2. LOCA 100 Life (롯데카드) - 국내 가맹점에서 5만원 이상 2~3개월 할부 결제 시 …3. [LOCA] Edu (롯데카드) - 국내 가맹점 2~3개월 무이자 할부4. [LOCA] Pet (롯데카드) - 국내 가맹점 2~3개월 무이자 할부5. [LOCA] Auto (롯데카드) - 국내 가맹점 2~3개월 무이자 할부6. [LOCA] Wellness (롯데카드) - 국내 가맹점 2~3개월 무이자 할부7. [부자되세요 덤카드] (BC 바로카드) - [덤3] 국내 가맹점 3개월 이상 유이자 할부 …	<ol style="list-style-type: none">1. S-OIL LPG KB국민카드 (KB국민카드) - 백화점, 대형마트, 정비, 손해보험 업종…2. LOCA 365 카드 (롯데카드) - 국내 가맹점 2~3개월 무이자 할부3. 원더카드 (원더 FREE+) (하나카드) - 국내 5만원 이상 결제건 2~3개월 무이자…4. UPTURN카드 (BC 바로카드) - 전국 대형할인점, 백화점에서 최대 3개월 무이자…5. SmileCard the Club (현대카드) - 모든 가맹점에서 5만원 이상 결제 시 2~3개…6. 원더카드 (원더 T) (하나카드) - 무이자 할부…7. LOCA 100 Life (롯데카드) - 국내 가맹점에서 5만원 이상 2~3개월 할부…

💡 Insight:

카드사별 표현 방식이 달라서, 혜택이 더 좋음에도 불구하고 사용된 단어나 문장의 품질에 따라 RAG 문서가 선택되지 않을 가능성 존재.

- RAG 데이터를 구성할 때, 데이터의 문장력이 ‘균등’ 해야함.
- 카드 혜택과 같이 숫자가 사용되는 질문은 문맥 파악과 함께 수학적 사고도 함께 필요. Ex) 10% 이상인 혜택에 대해, 10%에 가까운 혜택만 검색.

Fine-Tuning

양자화(Quantization) - NF4

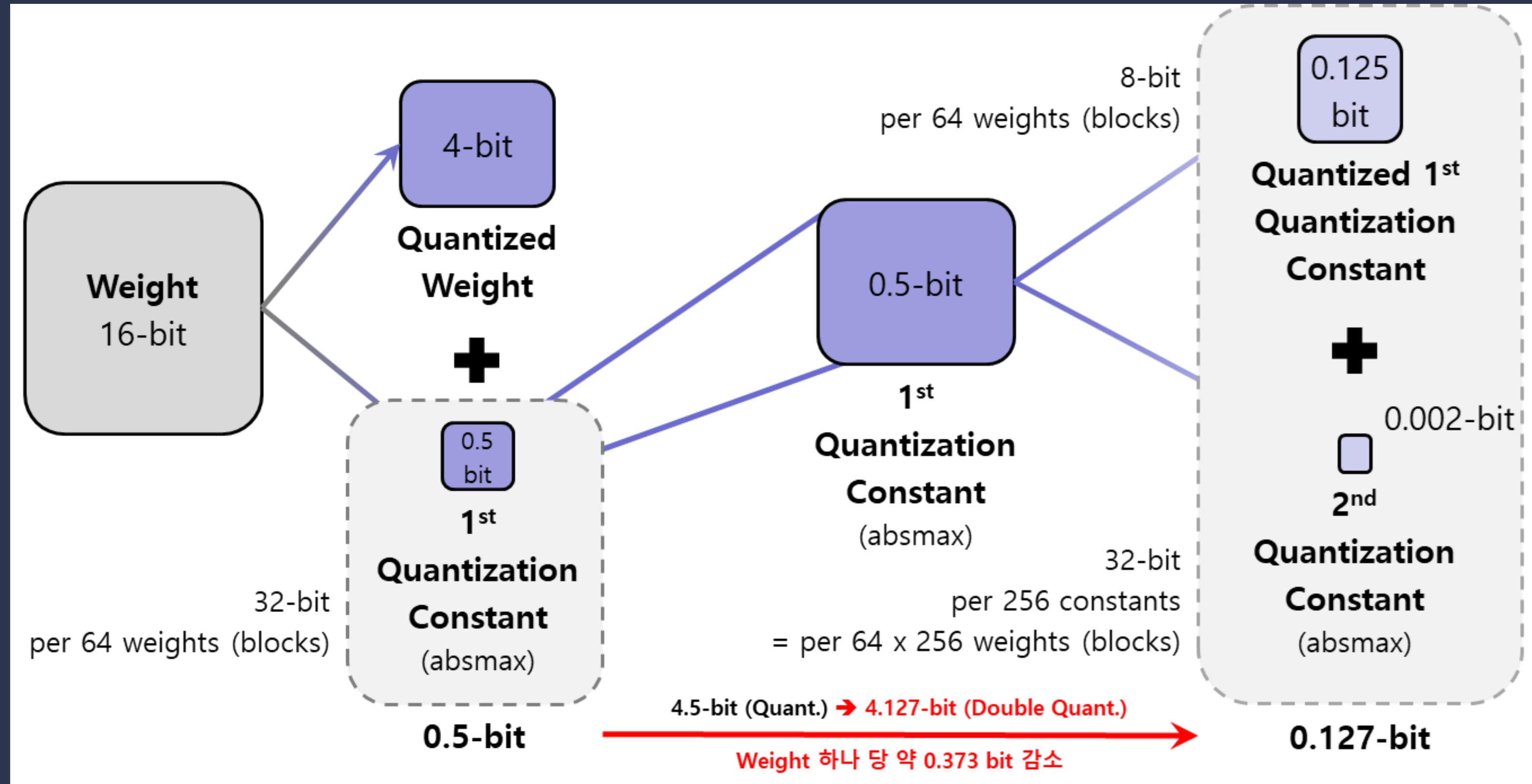
4-bit NormalFloat Quantization (NF4)

Input weight tensor 양자화 과정 예시

Input weight tensor	-3.1	0.1	2	-0.5	0.2	1.2
Block (paper에서는 64)	-3.1	0.1	2	-0.5	0.2	1.2
quantization constants	← Find absmax	3.1		1.2		
Normalize with absmax	-1.0	0.0323	0.6452	-0.4167	0.1667	1.0
Find closest 4-bit value	-1.0	0.0	0.7229	-0.3949	0.1609	1.0
Find corresponding index	0	8	16	3	8	1

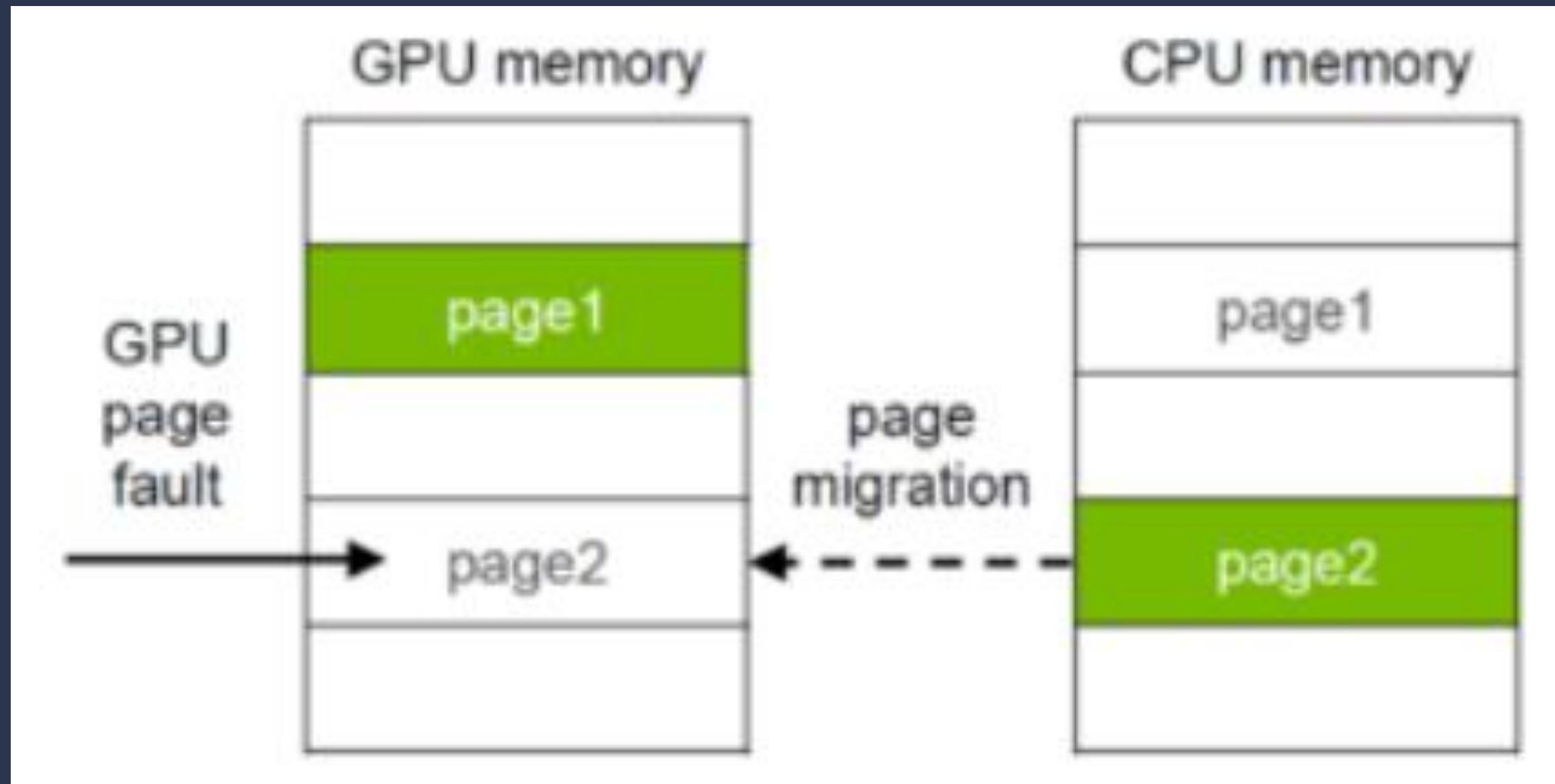
Fine-Tuning

양자화(Quantization) - Double Quantization



Fine-Tuning

양자화(Quantization) - Paged Optimization



✓ Fine-Tuning

양자화 Implementation

```
BASE_MODEL = "unsloth/mistral-7b-instruct-v0.2-bnb-4bit"
✓ 0.0s
```

`card_test2.json`은 다운로드와 메모리 부족 문제를 방지하기 위해 지원하는 4bit 사전 양자화 모델입니다.

```
tourbit_models = [
    "unsloth/mistral-7b-bnb-4bit",
    "unsloth/mistral-7b-instruct-v0.2-bnb-4bit",
    "unsloth/llama-2-7b-bnb-4bit",
    "unsloth/gemma-7b-bnb-4bit",
    "unsloth/gemma-7b-it-bnb-4bit", # Gemma 7b의 Instruct 버전
    "unsloth/gemma-2b-bnb-4bit",
    "unsloth/gemma-2b-it-bnb-4bit", # Gemma 2b의 Instruct 버전
    "unsloth/llama-3-8b-bnb-4bit", # Llama-3 8B
] # 더 많은 모델은 https://huggingface.co/unsloth에서 확인할 수 있습니다.
```

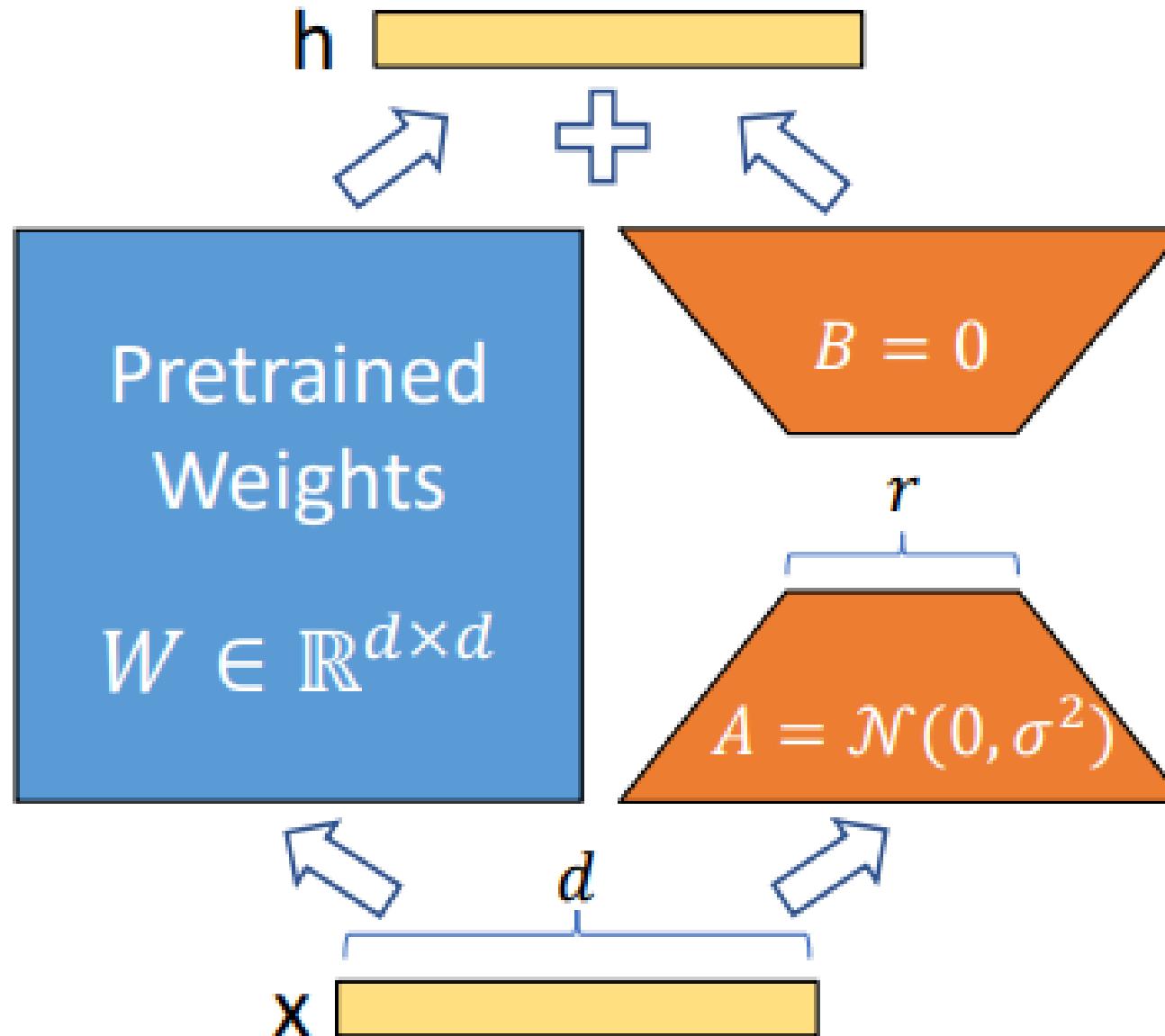
```
model_id = "EleutherAI/polyglot-ko-12.8b" #huggingFace

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.bfloat16
)

model = AutoModelForCausalLM.from_pretrained(model_id, quantization_config=bnb_config,
```

Fine-Tuning

LoRA(Quantization)



A neural network contains many dense layers which perform matrix multiplication. The weight matrices in these layers typically have full-rank. When adapting to a specific task, Aghajanyan et al. (2020) shows that the pre-trained language models have a low “intrinsic dimension” and can still learn efficiently despite a random projection to a smaller subspace. Inspired by this, we hypothesize the updates to the weights also have a low “intrinsic rank” during adaptation. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, we constrain its update by representing the latter with a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$, and the rank $r \ll \min(d, k)$. During training, W_0 is frozen and does not receive gradient updates, while A and B contain trainable parameters. Note both W_0 and $\Delta W = BA$ are multiplied with the same input, and their respective output vectors are summed coordinate-wise. For $h = W_0x$, our modified forward pass yields:

$$h = W_0x + \Delta Wx = W_0x + BAx \quad (3)$$

Figure 1: Our reparametrization. We only train A and B .

✓ Fine-Tuning

LoRA Implementation

```
model = FastLanguageModel.get_peft_model(  
    model,  
    r=16,           # 0보다 큰 어떤 숫자도 선택 가능! 8, 16, 32, 64, 128이 권장됩니다.  
    lora_alpha=32,   # LoRA 알파 값을 설정합니다.  
    lora_dropout=0.05, # 드롭아웃을 지원합니다.  
    target_modules=[  
        "q_proj",  
        "k_proj",  
        "v_proj",  
        "o_proj",  
        "gate_proj",  
        "up_proj",  
        "down_proj",  
    ], # 타겟 모듈을 지정합니다.  
    bias="none", # 바이어스를 지원합니다.  
    # True 또는 "unslosh"를 사용하여 매우 긴 컨텍스트에 대해 VRAM을 30% 덜 사용하고, 2배 더 큰 배치 크기를 지원합니다.  
    use_gradient_checkpointing="unslosh",  
    random_state=123, # 난수 상태를 설정합니다.  
    use_rslora=False, # 순위 안정화 LoRA를 지원합니다.  
    loftq_config=None, # LoftQ를 지원합니다.  
)
```

Fine-Tuning

QLoRA

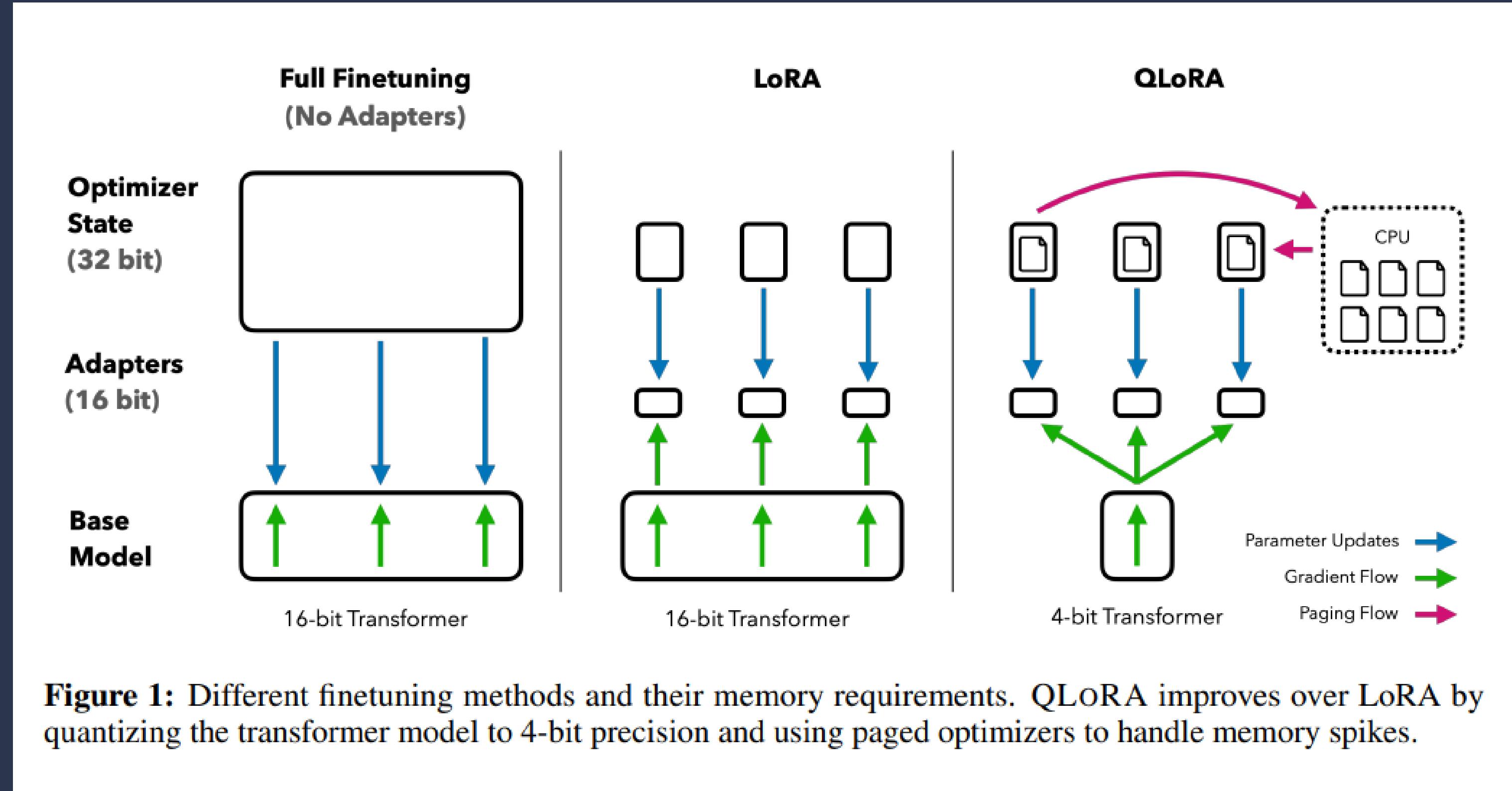


Figure 1: Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.

✓ Web Search - Tavily API



AI 최적화: LLM과의 원활한 통합을 위해 설계됨

최신 정보: 실시간으로 업데이트되는 정보를 제공

다양한 검색 모드: 일반 검색, 뉴스 검색, 이미지 검색 등
다양한 모드를 제공

콘텐츠 필터링: 신뢰할 수 있는 소스의 정보만을 제공

무료 사용량: 월 1,000회의 무료 API 호출을 제공

✓ Web Search - Tavily API

Tavily Implementation

```
web_search = TavilySearchResults(max_results=2)
search_results = web_search.invoke(search_query)
```

```
# template for web search result.
template_for_web = """
너는 사용자의 질문에 대한 답변을 잘 설명해주는 어시스턴트야.
반드시 한국어로만 답변하고, 영어는 절대 사용하지 마.
아래 검색 결과(context)에서 사용자 질문에 대한 답변을 찾아서 친절하고 구조화된 형식으로 답변해줘.
검색 결과(context)에서 질문에 대한 답변을 찾지 못했으면, 답변을 찾지 못했다고 말해줘.
```

```
web_chain = (
{
    "context": itemgetter("context"),
    "question": itemgetter("question"),
}
| prompt_for_web
| llm
| StrOutputParser()
)
```

답변은 다음 형식을 따라줘:
웹에서 검색한 결과는 다음과 같습니다.
...

✓ 웹페이지 구현

작동 예시 - 1번 카테고리

신용카드 혜택 챗봇

 영화볼때 혜택있는 카드 알려줘

 다음은 영화 할인 혜택이 제공되는 신용카드들입니다. 추천하는 카드는 **일상의 기쁨카드**(IBK 기업은행)과 **부자되세요 홈쇼핑카드 (BC 바로카드)**입니다.

이유: 각각 '영화 할인' 및 '[생활할인 서비스] 영화 3,000원 청구할인(현장/온라인 예매 포함)'의 혜택을 제공하여 영화 관람 비용을 줄일 수 있습니다. 추가로, **삼성 iD CLASSY 카드**(삼성카드), **LG U+ Bora 신한카드 Big Plus**(신한카드), **Olleh Super DC IBK카드**(IBK기업은행), **올바른 FLEX 카드**(NH농협카드) 등도 영화 할인이 제공되는 카드입니다.

작동 예시 - 2번 카테고리

 일상의 기쁨카드랑 부자되세요 홈쇼핑카드 혜택 비교해줘

 두 신용카드, 일상의 기쁨카드(IBK기업은행)와 부자되세요 홈쇼핑카드(BC 바로카드),는 각자의 특성에 따라 다른 할인 혜택을 제공합니다.

- 일상의 기쁨카드**: 온라인 영화 관람시 3,000원 청구할인 제공
- 부자되세요 홈쇼핑카드**: [생활 할인 서비스]에서 영화 할인이 제공될 수 있습니다. 현재 [생활 할인 서비스]는 신규 회원 등록 시 50,000원 할인금액을 제공하고 있으므로, 영화에 대한 할인이 적용될 수 있습니다.

추천하는 이유:

- 일상의 기쁨카드**: 온라인 영화 관람을 자주 하는 경우에 적합합니다. 3,000원 할인금액으로 영화 관람비용을 줄일 수 있습니다.
- 부자되세요 홈쇼핑카드**: [생활 할인 서비스]를 이용하면 다양한 업체와 상점에서 할인을 받을 수 있으므로, 영화 관람비용뿐만 아니라 일상적으로 다양한 상품을 구매할 경우에도 적합합니다.

또한, 부자되세요 홈쇼핑카드는 [생활 할인 서비스]를 이용하여 일상적으로 할인을 받아볼 수 있지만, 일상의 기쁨카드는 온라인 영화 관람 시에만 3,000원 할인금액을 제공합니다.

카드에 대해 궁금한 걸 입력해보세요! >

✓ 웹페이지 구현

작동 예시 – 3번 카테고리

나한테 딱 맞는 카드 추천해줘 >

사용자 맞춤 카드 추천

사용자 맞춤 추천을 위해 아래 내용을 입력해주세요.

나이는 어떻게 되시나요?
35

성별을 선택해주세요
남성

직업을 알려주세요
개발자

취미나 관심사 (예: 영화, 커피, 쇼핑 등)
카페가기

추천 카드 보기



1. KB KARD PREMIUM: 매주 거래금액에 따라 포인트 적립, 카페 할인 이용 가능
2. Shinhan Card T Money PLUS: T몰 점포에서의 할인/캐시백, 월납부 일정금액 달성 시 추가 포인트 적립
3. Hyundai Card S-PASS: 카페 점포 할인/캐시백, 매주 거래금액에 따라 보너스 포인트 적립

작동 예시 – 4번 카테고리

신용카드 혜택 챗봇

미국 대통령 이름이 뭐야?

미국의 대통령은 현재 도널드 트럼프입니다. 2025년 1월 20일에 공식 취임하였습니다.

6. 참고자료



참고자료

- OpenAI. *Prompt Engineering Guide*. OpenAI, 2024, platform.openai.com/docs/guides/prompt-engineering.
- Google Cloud. *What is Prompt Engineering?* Google, 2024, cloud.google.com/discover/what-is-prompt-engineering.
- Zhu, Liting, et al. *Unleashing the potential of prompt engineering in Large Language Models: a comprehensive review*. arXiv preprint arXiv:2310.14735v5, 2024. [arXiv.org](https://arxiv.org/abs/2310.14735)
- 신문기사, CEO Score Daily, 2024, <https://ceoscoredaily.com/page/view/2024070317450749375>
- 신문기사, 파이낸셜 뉴스, 2024, <https://www.fnnews.com/news/202405211814482348>

Q & A

Thank You

