몰입하는 개발자
김주성입니다.

CONTACT

Email: 8804who@naver.com
Phone: 010-9085-3194

About Me

INTRODUCTION

몰입하며 성장하고 있는 개발자 김주성입니다.

대학교에서 컴퓨터공학을 전공한 후 부스트캠프 AI Tech에서 NLP 트랙을 수료하였습니다. 수료 후에도 머신러닝, 딥러닝 분야에서 여러 가지 시도를 하며 경험을 쌓고 있습니다.

PROFILE

Name 김주성 JuSeong Kim

Birth 1998.07.18

Blog https://velog.io/@8804who/posts

Github https://github.com/8804who

SKILL

python™ PyTorch

EDUCATION

부스트캠프 AI Tech 5기

NLP Track

2023.03 ~ 2023.08

삼성 SDS 대학생 알고리즘 특강

2022.07 ~ 2022.07

경남대학교 컴퓨터공학과

복수전공 USG 공유대학 스마트제조 ICT

GPA: 3.87/4.5

2017.03 ~ 2023.02

AWARDS & CERTIFICATES

정보처리기사
자격증(2024.06)

SQLD(SQL 개발자)
자격증(2024.04)

데이콘 웹 로그 기반 조회수 예측 해커톤
2위(2024.03)

KB국민은행 제5회 Future Finance A.I. Challenge
특별상(2023.08)

제2회 교원그룹 생성 AI기반 에듀테크 사업 제안대회
우수상(2023.07)

특별상(학과 내 우수 활동 학생)
최우수상(2023.02)

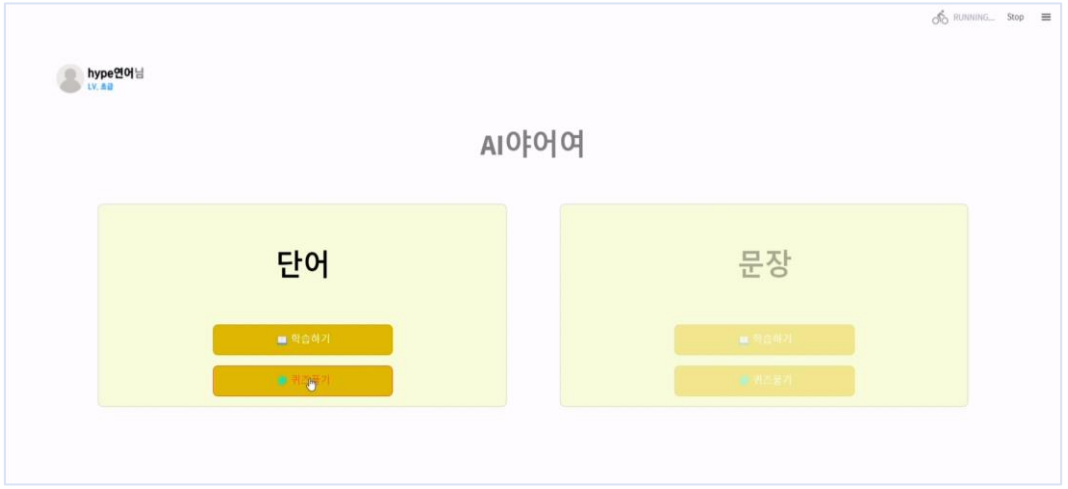
ADsP(데이터분석준전문가)
자격증(2022.07)

컴퓨터활용능력 1급
자격증(2021.07)

Project

AI야어여 제2회 교원그룹 AI 챌린지 생성 AI기반 에듀테크 사업 제안 대회 우수상 수상작

프로젝트 정보



프로젝트 개요	◆ 외국인 또는 아동을 대상으로 한 한국어 교육 서비스 ◆ 국립국어원이 지정한 기초 단어들 혹은 사용자가 원하는 단어를 키워드로 생성형 AI가 문제를 생성 ◆ 생성된 문제를 통해 쉽게 한국어 학습 가능
프로젝트 기간	2023.06~2023.07
프로젝트 진행 인원	5명
프로젝트 역할	생성형 AI API 적용, 프롬프트 엔지니어링
사용 API	BARD, ChatGPT
관련 링크	깃허브 저장소 시연 영상

Project

AI야어여 제2회 교원그룹 AI 챌린지 생성 AI기반 에듀테크 사업 제안 대회 우수상 수상작

프로젝트 아키텍쳐

학습할 키워드 선정
(국립 국어원 선정 or 자유 키워드)

생성형 AI를 통해
생성한 문제를 통해 학습

Streamlit

사용자 지정
단어 전달

Bard

주어진 단어를
이용한 문장 생성

사용자 지정 단어를 포함한
문장으로 생성한 문제를 전달

문장을 이용하여
학습용 문제 생성

DALL·E

문장을 표현한
그림 생성

사용자 지정 단어를 포함한
문장으로 생성한 그림을 전달

생성형 AI API 적용 & 프롬프트 엔지니어링

```python
def make_sentence_free(keyword):
    '''
    :param keyword: 문장 생성에 사용할 키워드
    :return: bard를 사용해서 생성한 keyword 관련 문장
    '''

    # 프롬프트 문장+사용자 키워드
    input_text = f'이미지 생성 모델에 사용 가능하고 \'{keyword}\'에 관련된 초등학생 수준의 간단한 문장을 1줄 생성해줘'

    # 문장 생성시 기대하지 않은 결과가 나오는 경우가 존재하므로 예외처리를 통해 잘못된 문장이 생성되는 것 방지
    while True:
        try:
            response = bardapi.core.Bard().get_answer(input_text)
            sentence = response['choices'][0]['content'][0].split('\n\n')[1]
            sentence = re.sub(r"[^ㄱ-ㅣ가-힣\s]", "", sentence).lstrip().rstrip()
        except:
            print('bard 문장 생성 오류')
            time.sleep(20)
        else:
            # 문장이 한글 대신 다른 언어로 작성된 경우 다시 생성
            if isKoreanIncluded(sentence):
                break
            else:
                continue
    return sentence
```

BARD API 적용(키워드를 통한 문장 생성)

- ◆ 사용자가 지정한 키워드를 이용한 문장을 생성하도록 API에 요청
- ◆ 대상이 어린이 또는 한국어에 익숙하지 않은 외국인
  ⇒ 초등학생 수준의 간단한 문장을 요청하니 짧고 쉬운 문장 생성

CHAT GPT API 1 (빈 칸이 있는 문장 생성)

```python
def make_blank_free(sentence):
    '''
    :param sentence: bard에서 생성한 문장
    :return: sentence에 빈칸을 뚫은 문장(str)과 후보 단어 4개(리스트)
    '''
    input_text = f'\'{sentence}\' 문장에서 단어 1개를 ___으로 대체해서 출력하고 ___ 자리에 있던 단어를 출력해 줘'

    while True:
        # gpt에서 제대로된 답변을 했는지 확인
        try:
            response = openai.ChatCompletion.create(
                model="gpt-3.5-turbo",
                messages=[
                    {"role": "system", "content": "문장에서 단어 1개를 ___으로 대체해서 출력하고 ___ 자리에 있던 단어를 출력해달라는 요청을 받으면 반드시 '단어 1개를 ___으로 대체해서 출력하고 ___ 자리에 있던 단어를 출력해 줘' 형식으로 출력해줘"},
                    {"role": "user", "content": '\'우주에는 수없이 많은 별이 빛나고 있습니다.\' 문장에서 단어 1개를 ___으로 대체해서 출력하고 ___ 자리에 있던 단어를 출력해 줘'},
                    {"role": "assistant", "content": '우주에는 수없이 많은 ___이 빛나고 있습니다.\':\'별\''},
                    {"role": "user", "content": '\'천문학자는 별을 보지 않는다\' 문장에서 단어 1개를 ___으로 대체해서 출력하고 ___ 자리에 있던 단어를 출력해 줘'},
                    {"role": "assistant", "content": '___는 별을 보지 않는다\':\'천문학자\''},
                    {"role": "user", "content": '\'그래도 좋은 날이 앞으로 많기름\' 문장에서 단어 1개를 ___으로 대체해서 출력하고 ___ 자리에 있던 단어를 출력해 줘'},
                    {"role": "assistant", "content": '그래도 ___ 날이 앞으로 많기름\':\'좋은\''},
                    {"role": "user", "content": input_text}
                ]
            )
            generated_sentence, word = response.choices[0].message.content.split(':')[:2]
            generated_sentence, word = generated_sentence.strip(), word.strip()
            generated_sentence, word = re.sub('\'', '', generated_sentence), re.sub('\'', '', word)
        except openai.error.RateLimitError:
            print('open ai 사용량 제한')
            time.sleep(20)
        else:
            if re.sub('___', word, generated_sentence) == sentence:
                break
```

- ◆ 문제의 예제로 사용하기 위해 단어 하나를 '___'로 대체한 문장을 API에 요청
- ◆ 빈칸이 없는 문장을 주거나 잘못된 문장을 제공하는 경우가 발생
  ⇒ 퓨샷러닝을 활용한 결과 잘못된 문제를 제공하는 빈도가 크게 감소
  ⇒ 하지만 여전히 잘못된 문제를 제공하는 경우가 존재
  ⇒ 정상적인 문장이 생성될 때까지 새로 생성(오류의 빈도가 감소하여 사용 가능)

CHAT GPT API 2(빈 칸에 들어갈 단어 목록 생성)

```python
def make_blank_subject(sentence, word):
    '''
    :param sentence: bard에서 생성한 문장, word: 문장 생성에 사용한 단어
    :return: sentence에 빈칸을 뚫은 문장(str)과 후보 단어 4개(리스트)
    '''

    sentence = re.sub(word, '___', sentence)

    input_text = f'\'{sentence}\'에서 \'___\' 자리에 어울리는 단어를 4개 추천해 줘'
    while True:
        try:
            response = openai.ChatCompletion.create(
                model="gpt-3.5-turbo",
                messages=[
                    {"role": "system",
                     "content": "문장에서 \'___\' 자리에 어울리는 단어를 4개 추천해 달라는 요청을 받으면 반드시 \'단어1:단어2:단어3:단어4\' 형식으로 출력해줘"},
                    {"role": "user",
                     "content": '\'우주에는 수없이 많은 ___이 빛나고 있습니다.\'에서 \'___\' 자리에 어울리는 단어를 4개 추천해 줘'},
                    {"role": "assistant", "content": '\'행성:태양:블랙홀:은하\''},
                    {"role": "user", "content": '\'___는 별을 보지 않는다\'에서 \'___\' 자리에 어울리는 단어를 4개 추천해 줘'},
                    {"role": "assistant", "content": '\'물리학자:과학자:수학자:의사\''},
                    {"role": "user", "content": '\'그래도 ___ 날이 앞으로 많기름\'에서 \'___\' 자리에 어울리는 단어를 4개 추천해 줘'},
                    {"role": "assistant", "content": '\'나쁜:슬픈:우울한:기쁜\''},
                    {"role": "user", "content": input_text}
                ]
            )
            generated_words = response.choices[0].message.content
            generated_words = re.sub('\'', '', generated_words)
            generated_words = generated_words.split(':')
        except openai.error.RateLimitError:
            print('open ai 사용량 제한')
            time.sleep(20)
        else:
            if len(generated_words) == 4:
                break

    if word not in generated_words:
        generated_words[3] = word
    words = generated_words
    random.shuffle(words)
    answer = words.index(word)
    return words, answer
```

- ◆ 문제의 보기로 사용하기 위해 문장의 빈칸에 들어갈 만한 단어 목록을 API에 요청
- ◆ 포맷이 통일되지 않고 간혹 단어 목록을 제대로 주지 않는 문제 발생
  ⇒ 문장 생성과 마찬가지로 퓨샷러닝을 활용한 결과 빈도 크게 감소
  ⇒ 잘못된 응답의 빈도가 줄었지만, 여전히 존재
  ⇒ 정상적인 보기 목록이 생성될 때까지 새로 생성(오류의 빈도가 감소하여 사용 가능)

프로젝트 후기

프롬프트 엔지니어링의 중요성

최근 LLM이 대세가 되면서 프롬프트 엔지니어링도 많은 관심을 받고 있었기 때문에 생성형 AI를 이용한 프로젝트를 진행해 보았습니다. 이번 프로젝트에서는 간단하게 프롬프트 엔지니어링에 대해서 알아보고 실제로 프로젝트에 적용을 해보았습니다.
이번 프로젝트를 진행하며 프롬프트 엔지니어링의 중요성에 대해 어느 정도 깨닫게 되었는데 프롬프트 엔지니어링을 통해 후처리를 상당히 간소화시킬 수 있다는 것이 매우 인상적이었습니다.
하지만 프롬프트 엔지니어링도 결국 토큰을 사용하는 방식이기 때문에 적은 토큰으로 자신이 원하는 기능을 제대로 구현할 수 있는 효율적인 프롬프트를 개발하는 것이 중요할 것 같다는 생각이 들었습니다.

생성형 AI를 믿어도 될까?

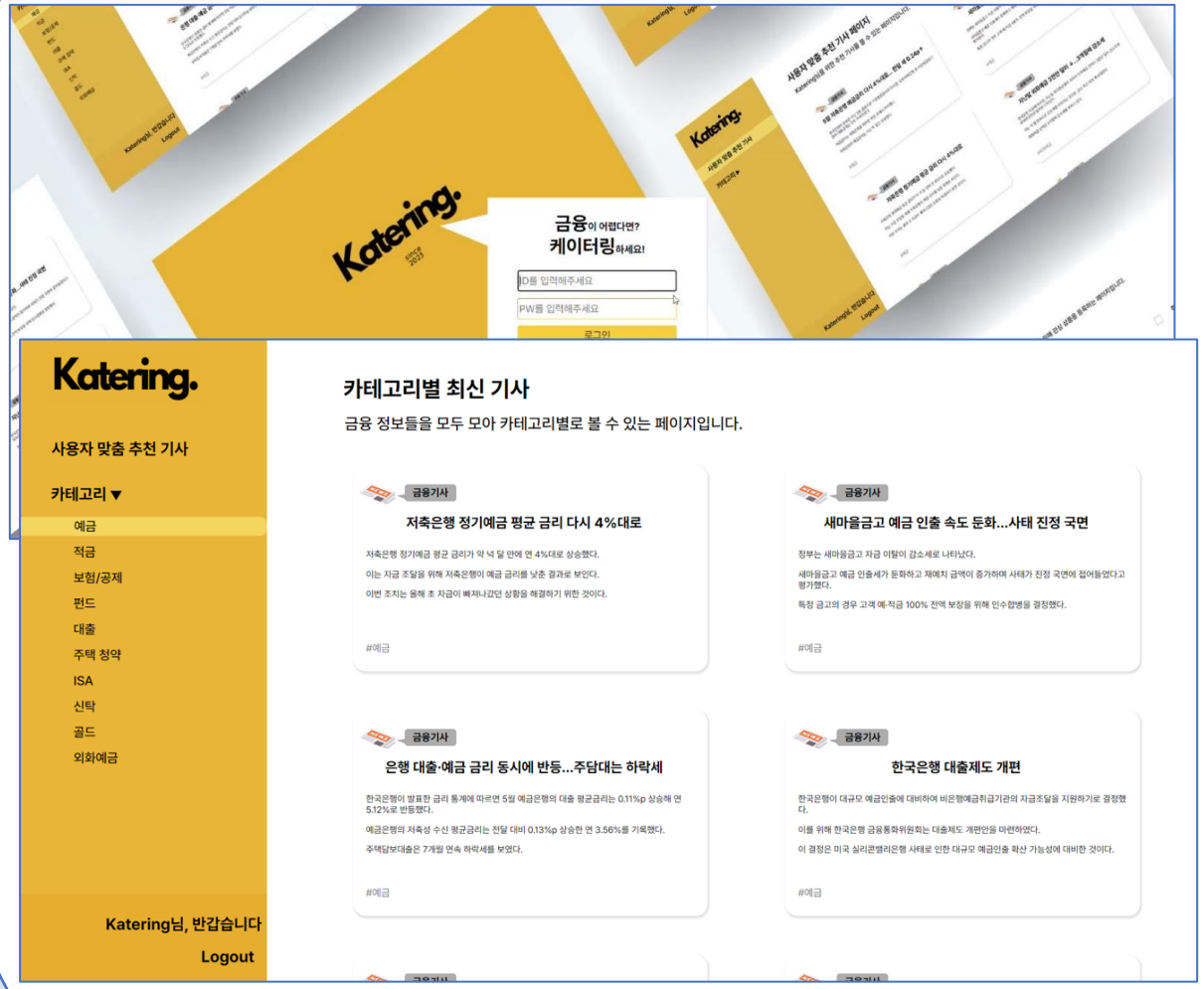
이번 프로젝트의 중요한 주제가 생성형 AI였기 때문에 생성형 AI가 기능 구현에 핵심적으로 사용이 되었습니다. 하지만 생성형 AI가 잘못된 응답을 하는 경우가 종종 발생하여 그런 문제를 처리하는데 생각보다 많은 노력이 필요했습니다.
이번 프로젝트에서는 예상 트래픽이 아주 적었다는 점과 프롬프트 엔지니어링이 잘못된 응답의 빈도를 크게 줄여줬다는 점 때문에 제대로 된 응답을 할 때까지 새롭게 요청하는 방식이 가능했지만, 많은 트래픽이 예상되는 대형 서비스에서는 이런 방식을 적용하기 힘들 것으로 생각합니다.
실제로 대형 서비스에서도 종종 잘못된 응답이 나타나는 경우가 있습니다. 이후에 생성형 AI를 활용하는 프로젝트를 다시 진행하게 된다면 이 문제에 대해 좀 더 깊게 고민해 보아야 할 것 같습니다.

Project

Katering

KB국민은행 제5회 Future Finance A.I. Challenge 특별상 수상작

프로젝트 정보

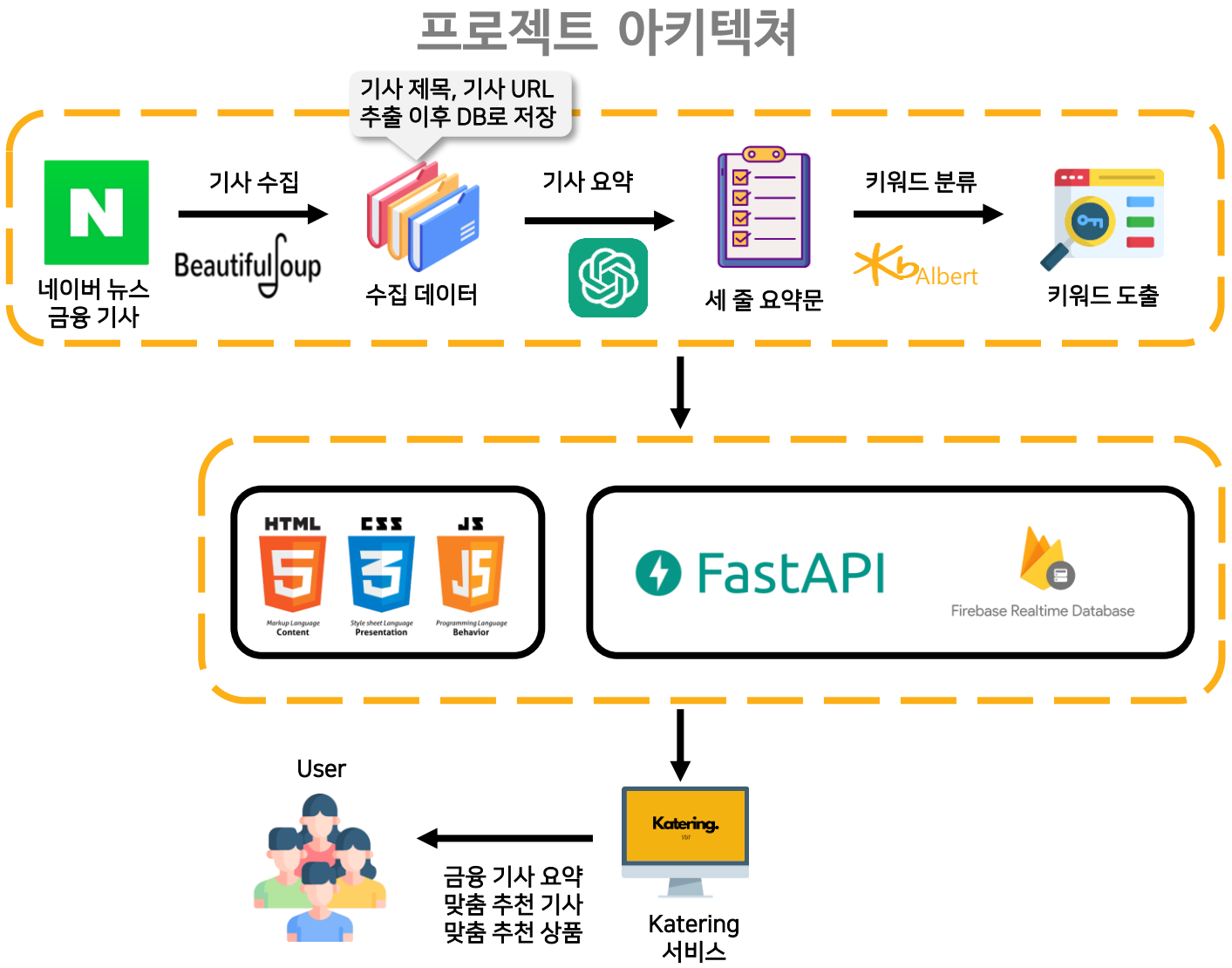


프로젝트 개요	◆ 금융 관련 기사와 금융 상품 정보를 요약 ◆ 기사와 상품을 카테고리별로 분류 ◆ 사용자가 관심 있는 기사와 상품 정보 제공
프로젝트 기간	2023.08~2023.08
프로젝트 진행 인원	3명(개발 2 & 디자인 및 기획 1)
프로젝트 역할	데이터 처리 파이프라인 구현, 웹페이지 구현
사용 기술 스택	CSS, HTML, JavaScript, FastAPI, Python, PyTorch
관련 링크	깃허브 저장소 시연 영상

Project

Katering

KB국민은행 제5회 Future Finance A.I. Challenge 특별상 수상작

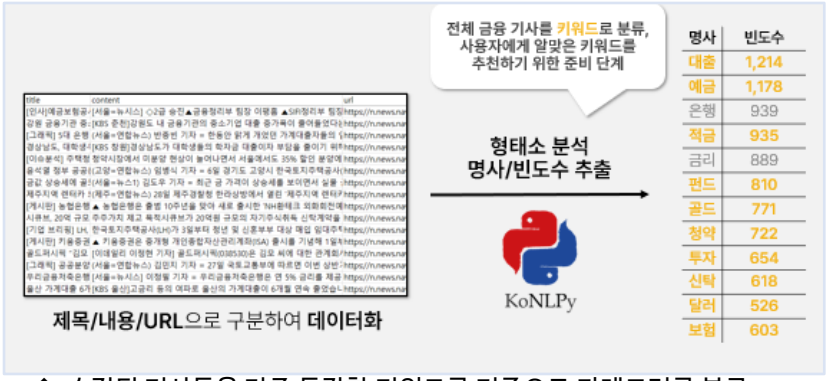


Project Katering

KB국민은행 제5회 Future Finance A.I. Challenge 특별상 수상작

데이터 처리 파이프라인 구현

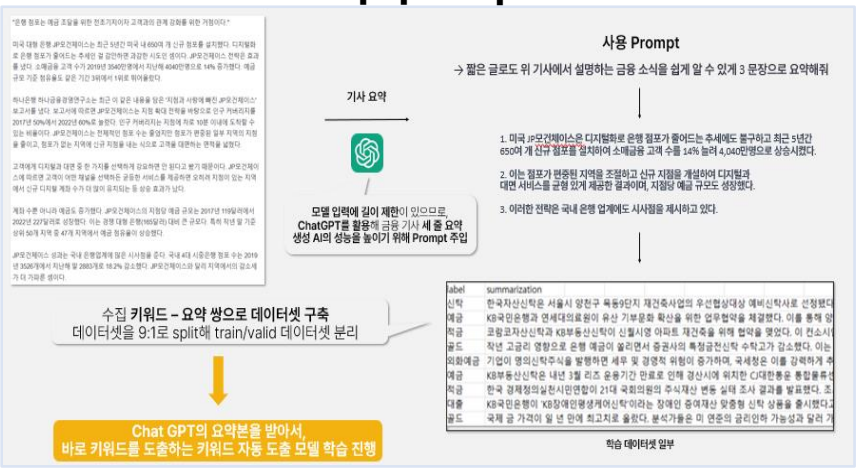
카테고리 분류



명사	빈도수
대출	1,214
예금	1,178
은행	939
적금	935
금리	889
펀드	810
골드	771
청약	722
투자	654
신탁	618
달러	526
보험	603

- 수집된 기사들을 자주 등장한 키워드를 기준으로 카테고리를 분류
- 빈도수를 추출하는 과정에서 기사의 문자 포맷이 달라 같은 단어가 서로 다른 단어로 인식되어 빈도수가 부정확함
  ⇒ 문자 포맷을 통합시키는 방식으로 정확한 빈도수 추출

기사 요약



- 기사들을 ChatGPT API를 이용해 요약하고 요약된 내용을 모델 학습에 사용할 수 있도록 학습용 데이터셋으로 변환
- 퓨샷러닝도 고려를 해보았으나 아래의 이유로 활용하지 않음
  1. 퓨샷러닝 없이도 요약의 질이 만족스러웠음
  2. 많은 기사의 양으로 인해 과다한 비용이 발생할 것으로 예상

요약 예시

하지만 지난 6월 초 시에서 금융사와 건설사, 신탁사, 설계사 등이 참석한 가운데 책임준공, 신탁계약 체결, 토지매매계약 등 의견을 청취하는 등 문제 해결에 힘을 쏟았으며, 결국 세부적인 이행시기와 방법 등의 협의를 이끌어 냈다.
민간사업자인 군산자동차무역센터는 이달 중 토지매매계약을 시작으로 10월까지 재원조달 등의 절차를 마치고 11월 중에 착공할 계획이다.
시도 시설 착공 후 인천 등 수도권 지역에서 사업설명회를 열고 새만금자동차 수출복합센터 시설 준공 후 잠정적 고객 확보에 노력해 나갈 계획이다.
아울러 완성차 제조 대기업들이 인증 중고차 시장 진출과 확대를 모색하고 있어 중고차의 수출·매매의 온라인 판매 기조에 맞춰 대기업과 상생 협력관계를 구축하고 수출·매매 통합 플랫폼을 통해 경쟁력을 확보해 나갈 계획이다.
시 관계자는 "사업이 지연된 만큼 민간사업자와 역량을 집중해 11월 착공과 2025년 상반기에 개장할 수 있도록 최선을 다하겠다"고 밝혔다.

→

새만금 자동차 수출복합센터가 군산시와 민간 투자자의 이견으로 중단될 위기에 직면했으나,
민간 사업자가 땅 확보와 자금 조달 계획을 마련하여 오는 11월에 착공될 것으로 예상된다.
이로써 수출복합센터 조성 사업은 지연되었던 것을 극복하고 2025년 상반기 개장을 목표로 하고 있다.

프로젝트 후기

팀 프로젝트 == 앙상블

이번 프로젝트는 제가 처음으로 개발에 대해 모르는 인원과 함께 진행한 프로젝트였습니다. 해당 인원과 의견을 교환하는 것이 쉽지 않았지만 서로 각자의 지식을 가지고 의견을 나눠가며 좋은 프로젝트를 만들기 위해 노력한 결과 수상이라는 쾌거를 거둘 수 있었습니다.
제가 제목에서 팀 프로젝트를 앙상블이라고 표현한 이유는 **팀 프로젝트도 앙상블처럼 서로 가지고 있는 지식과 사고 구조가 다른 인원들이 모였을 때 더 좋은 결과**가 나올 수 있다고 느꼈기 때문입니다.
현업에서도 마찬가지로 더 좋은 결과를 위해 **협업**으로 프로젝트가 진행됩니다. 미래에 그런 프로젝트를 진행할 저에게 이번 프로젝트에서의 경험은 아주 의미 있는 경험으로 남을 것 같습니다.

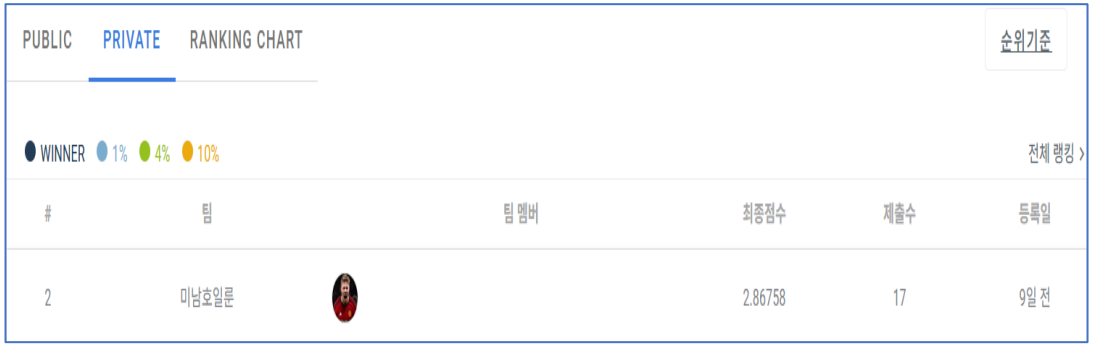
일단 해봤습니다!

이번 프로젝트에서 저의 역할을 데이터 처리 파이프라인 구현과 웹 페이지 구현이었습니다. 저는 사실 웹 개발에 대한 지식과 경험이 거의 없습니다.
하지만 팀에 웹 개발을 전문으로 하는 인원이 없었기 때문에 제가 웹페이지 구현을 맡게 되었습니다.
예상대로 웹 개발은 쉽지 않았고 기능을 하나 구현하려면 책과 스택 오버 플로우 사이트를 뒤지며 **일단 해봤습니다**. 결국 기한 내에 웹페이지를 완성할 수 있었고 수상이라는 쾌거를 거두었습니다.
만약 이것저것 모두 할 수 있는 실력자가 제 역할을 맡았다면 이런 고생이 없었을 것입니다. 하지만 저는 그런 사람이 아니기 때문에 이런 식으로 일단 해보는 것이 필요하다고 생각합니다.
능력 밖의 일이라고 생각되더라도 일단 해보면 결국은 해낼 수 있고 만약 실패하더라도 자신의 부족함에 대해 깨닫고 자신의 약점을 보완할 기회를 만들어 줄 것이라고 생각합니다.

Project

웹 로그 기반 조회수 예측 해커톤

프로젝트 정보

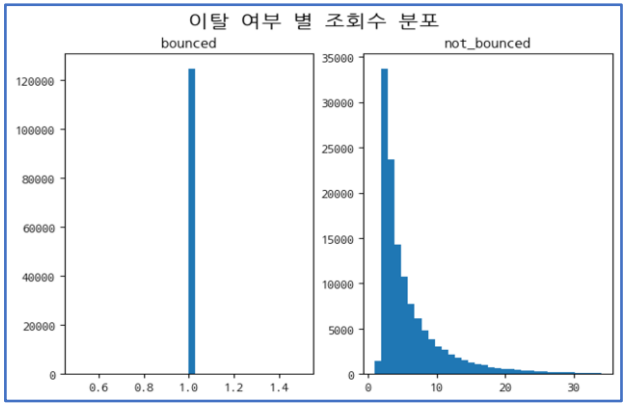


프로젝트 개요	◆ 데이터: 웹페이지 글 작성자의 브라우저, 사용 기기, 국가 등 19개의 피쳐 존재 ◆ 목표: 해당 글의 조회수를 예측 ◆ 평가 방식: RMSE
프로젝트 기간	2024.02~2024.03
프로젝트 진행 인원	개인 프로젝트
최종 결과	리더보드 기준 361명 중 2위
관련 링크	깃허브 저장소 대회 후기 및 정리

Project
웹 로그 기반 조회수 예측 해커톤

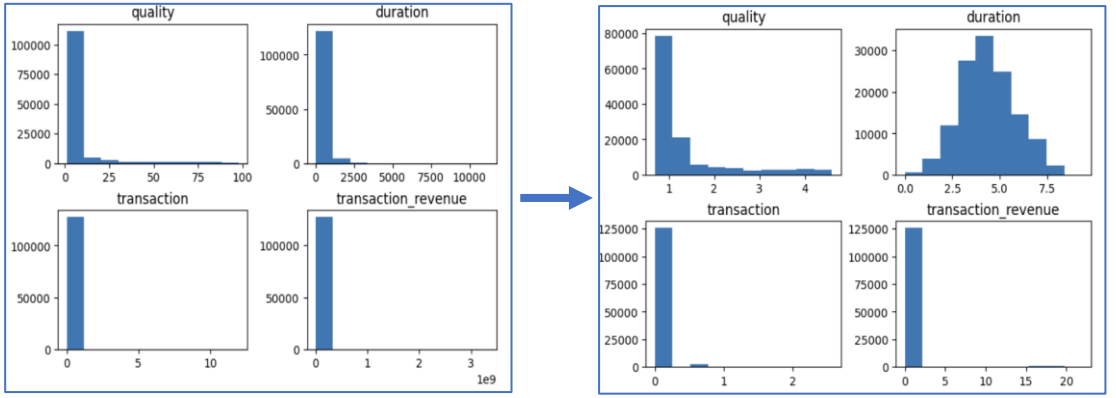
프로젝트 주요 내용

특정 피쳐에 의한 조회수 차이 발견



◆ EDA를 진행하는 과정에서 bounced에 따라 조회수의 분포가 완전히 달라진다는 점 발견
⇒ bounced인 데이터는 예측값을 1로 고정, not_bounced 데이터만 학습에 활용하고 예측을 진행
⇒ 자체 실험 기준 **2.59**에서 **2.53**으로 성능 향상

데이터 편향 해결



◆ 수치형 데이터들이 모두 좌측으로 편향되어 있다는 점 발견 ⇒ 로그 변환을 통해 편향을 어느정도 해결
◆ 데이터에 로그 변환을 적용하자 quality와 duration이 조회수와 매우 높은 상관관계를 가지게 되었음
◆ 로그 변환을 적용한 데이터를 통해 모델을 학습한 결과 리더보드 기준 **2.959**에서 **2.944**로 성능 향상

10	실험실 27번	2.9601453857	994399
11	리더보드 5번(실험실 9번)+리더보드 7번(실험실 13번)+리더보드9번(실험실 22번)+리더보드 10번(실험실 27번)	2.9424340348	994611
12	[최종 제출] 리더보드 9번(실험실 22번)+리더보드 10번(실험실 27번)	2.9347428794	994615
13	[최종 제출] 리더보드 7번(실험실 13번)+리더보드9번(실험실 22번)+리더보드 10번(실험실 27번)	2.938826049	994618

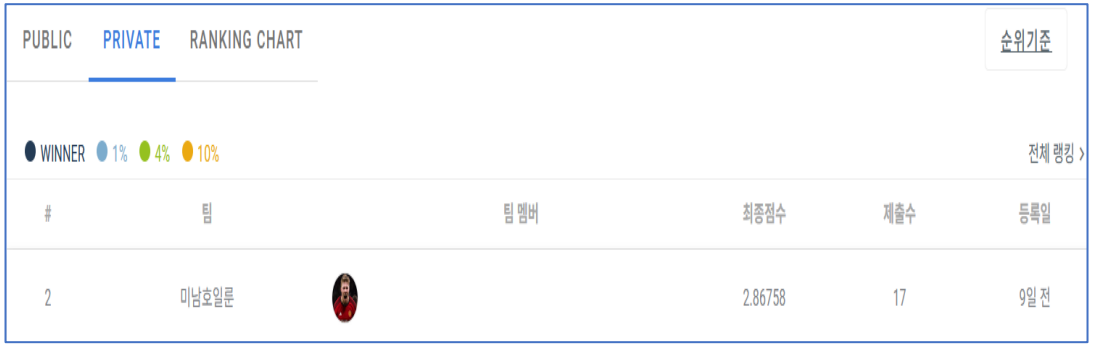
앙상블 진행

◆ 모델 성능 향상을 위해 K-Fold와 앙상블 적용
◆ 앙상블 과정에서 모델 간의 차이가 클수록 성능이 좋아진다는 점을 활용
⇒ 편향을 보정한 채 학습한 모델과 보정하지 않은 채 학습한 모델을 활용
⇒ 리더보드 기준 **2.936**에서 **2.934**로 성능 향상

Project

코드 유사성 판단 시즌 2 AI 경진대회

프로젝트 정보



프로젝트 개요	◆ 데이터: 500개의 문제에 대한 풀이 코드 (문제 별 각 500개 씩 총 25,000개 코드 파일을 활용해 직접 학습 데이터 생성) ◆ 목표: 한 쌍의 코드가 같은 문제를 해결하는 코드인지 여부 예측 ◆ 평가 방식: Accuracy
프로젝트 기간	2024.03~2024.04
프로젝트 진행 인원	개인 프로젝트
최종 결과	리더보드 기준 133명 중 10위
관련 링크	깃허브 저장소 대회 후기 및 정리

Project

코드 유사성 판단 시즌 2 AI 경진대회

프로젝트 주요 내용

토큰 제거 위치 변경

```cpp
#include <algorithm>
#include <iostream>
#include <stack>
#include <string>
#include <vector>

using namespace std;

int N, idx;

bool finished[101];
int cost[101];
int parents[101];

stack<int> st;
vector<int> vec[101];
vector<vector<int>> SCC;

int dfs(int n)
{
    parents[n] = ++idx;
    st.push(n);

    int parent = parents[n];

    for(int city : vec[n])
```

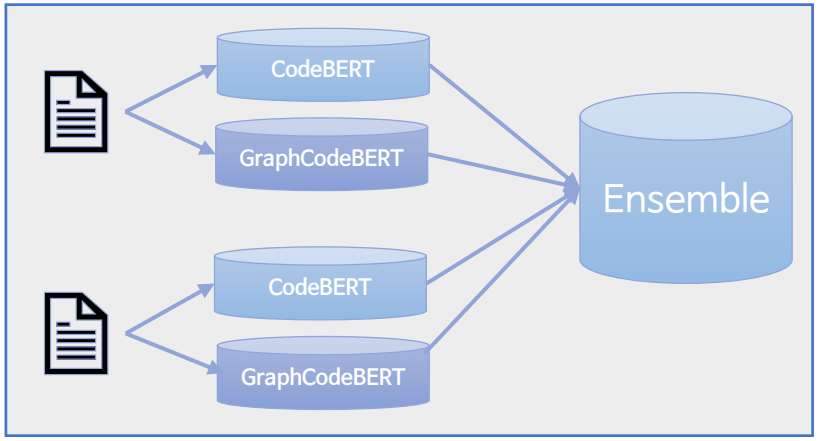
◆ 토큰 길이 제한 초과 시 뒤쪽 토큰을 살리는 방식을 활용
⇒ 보통 코드를 작성할 때 초반부에는 라이브러리 호출과 변수 선언이 등장
⇒ 라이브러리 호출과 변수 선언보다 코드의 내용이 더 많은 의미를 가짐
⇒ 대회 종료 후 다른 상위권팀들의 코드 확인 결과 대부분 이 방식을 사용

코드 데이터 전처리

```cpp
// if x is 1 then y is 2
if (x==1) y=2;
cout << y;

// x가 1이면 y는 2
if (x == 1) y = 2;
std::cout << y;
```

◆ 주석 삭제: 코드에 특화된 모델을 사용하였고 파일마다 주석의 언어가 달랐기 때문에 모두 삭제
◆ 띄어쓰기 방식 통합: 수식이나 논리 연산자가 띄어쓰기로 인해 다르게 인식되는 것을 방지
◆ 라이브러리 선언 문 제거: 사용하는 함수들을 통해 내용 파악에 문제가 없을 것이라 가정
◆ Std 선언문 삭제: ‘std::cout’와 ‘cout’ 같은 경우에 서로 다르게 인식되는 것을 방지
◆ 모두 적용한 결과 리더보드 기준 **0.967**에서 **0.970**으로 성능 향상



앙상블 진행

◆ 학습 데이터를 여러 개 생성하여 각각 학습 후 앙상블 적용
⇒ K-Fold 학습 방식에서 아이디어를 얻어 작은 데이터로 학습한 모델들을 앙상블하는 방법 시도
⇒ 약 4개까지는 성능이 향상되지만 그 후로는 효과가 미미
⇒ 리더보드 기준 **0.971**에서 **0.979**로 성능 향상
◆ 모델의 다양성을 위해 CodeBERT와 GraphCodeBERT 2개의 모델을 학습에 사용하였음
⇒ 위의 데이터 다양화 방법과 결합하였고 최종적으로 리더보드 기준 **0.975**에서 **0.981**로 성능 향상

Project

부스트캠프 AI Tech 5기 NLP Track

문장 내 개체간 관계 추출

프로젝트 개요

◆ 데이터: 문장과 해당 문장 내 언급된 두 개체의 명칭과 관계
◆ 목표: 두 개체의 관계를 예측
◆ 평가 방식: F1 Score

프로젝트 상세

개발 기간: 2023.05~2023.05

프로젝트 진행 인원: 5명

깃허브 저장소

랩업 리포트

프로젝트 주요 내용

데이터 증강

◆ 원본 문장의 일부 요소가 삭제된 문장을 생성하는 방식으로 새로운 문장을 생성
⇒ EDA 논문에서 문장의 일부 요소가 삭제되어도 의미를 유지한다는 내용을 참고
⇒ 리더보드 기준 **80.4**에서 **80.7**로 성능 향상
◆ 문장의 일부 단어를 해당 위치에 어울리는 다른 단어로 변경하여 새로운 문장 생성
⇒ BERT의 MLM 학습 방식에서 모티브를 얻어서 시도
⇒ 리더보드 기준 **80.4**에서 **80.9**로 성능 향상

증강 데이터 검증

◆ 데이터 증강을 통해 생성된 문장 중 기존의 의미를 유지하는 문장만 학습에 사용
⇒ 기존의 의미를 유지하지 못하는 새로운 문장은 오히려 학습에 방해
⇒ 코사인 유사도를 활용하여 의미를 유지하는 지 검증
⇒ 실험 결과 코사인 유사도가 0.9 이상인 문장들만 사용하였을 때 가장 좋은 성능
⇒ 리더보드 기준 **80.9**에서 **81.1**로 성능 향상

문장 간 유사도 측정

프로젝트 개요

◆ 데이터: 한 쌍의 문장과 그 두 문장의 유사도 점수
◆ 목표: 두 문장의 유사도를 0 ~ 5 사이의 값으로 측정
◆ 평가 방식: 피어슨 상관계수

프로젝트 상세

개발 기간: 2023.04~2023.04

프로젝트 진행 인원: 5명

깃허브 저장소

프로젝트 주요 내용

Loss 함수 변경

◆ 학습에 사용되는 Loss 함수를 L1 Loss에서 L2 Loss로 변경
⇒ 결과값이 0에서 5사이로 정해져 있기 때문에 이상치의 영향이 적다고 가정
⇒ 따라서 Regularization 효과를 위해 L2 Loss가 학습에 적합하다고 가정
⇒ 리더보드 기준 **0.8925**에서 **0.8938**로 성능 소폭 상승

데이터 증강

◆ 문장 쌍의 순서를 바꿔 생성한 데이터를 학습에 사용
⇒ 문장의 순서가 바뀌어도 두 문장의 유사도는 동일하기때문에 학습에 사용 가능
⇒ 리더보드 기준 **0.8938**에서 **0.8959**로 성능 향상

앙상블 방식 변경

◆ 앙상블 과정에서 각 모델의 출력 값에 시그모이드 함수를 적용한 후 평균을 구하고 다시 logit 값으로 변환하는 방식으로 앙상블 수행
⇒ 모델들의 출력 값이 logit 값이었기 때문에 단순히 평균을 구하는 것보다는 확률 값으로 변환 후 확률의 평균을 구하는 방식이 더 좋을거라 가정
⇒ 리더보드 기준으로 성능 변화가 거의 없었음(0.01 상승)

감사합니다.

앞으로도 몰입하는 개발자
김주성이 되겠습니다.