이경찬 nahcklee@gmail.com / 010-4105-7460

Portfolio

A. ML/DL 추천 모델 개발 & 배포

1. 개인화 추천 모델 개발 및 학습 최적화: Meta-BERT4Rec
2. 함께 많이 본 상품 추천 모델 개발 : Meta-Prod2Vec

B. AI 추천 모델 학습 최적화

1. Mixed Precision 적용 및 Max length 조정
2. Exponential Decay scheduler 적용
3. ReduceLROnPlateau 적용

C. 자연어처리 모델 개발

1. 개체명인식 / 문장 분류 모델 개발
2. 자연어 처리 모델 성능 최적화 작업

D. LLM 기반 추천 시스템 개발

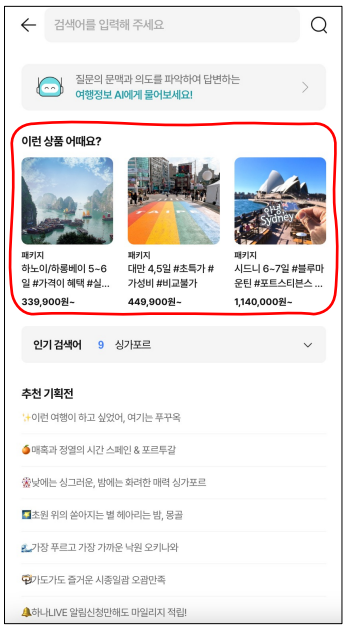
1. 빅콘테스트 2024 참가 : LLM 활용 제주도 맛집 추천 대화형 AI 서비스 개발
2. 여행플래너 서비스 내 LLM을 이용한 패키지 상품 추천

A. ML/DL 추천 모델 개발 & 배포

1. 개인화 추천 모델 개발 및 학습 최적화: Meta-BERT4Rec

프로젝트 개요

- **목표** : 클릭 로그 기반 개인화 추천 모델을 설계하고 성능을 개선
- **기술 활용**
  - BERT4Rec 기반 Sequential 추천 모델
  - 상품 메타 데이터를 활용한 Meta-BERT4Rec 설계
  - MLFlow로 모델 버전 관리 및 Mixed Precision 적용을 통해 학습 최적화



하나투어 앱 내 검색홈 개인화 추천 영역

- **학습 최적화**
  - Mixed Precision 적용 및 데이터셋의 시퀀스 길이와 배치 크기를 최적화하여 학습 시간을 50% 단축
  - Exponential Decay scheduler, ReduceLROnPlateau를 적용하여 학습 시간 단축 및 간헐적 학습 정체 현상 해결
- **Meta-BERT4Rec 모델 개발**
  - 순수 BERT4Rec의 추천 결과는 상품 카테고리, 국가, 도시가 혼재되어 있고, 인기 아이템 위주로 추천된다는 문제가 존재. 또한 BERT4Rec은 아이템의 메타 정보를 활용하지 않고 시퀀스 정보만 사용한다는 한계
  - 아이템 메타 정보를 추가로 사용한 모델인 [1] Meta-BERT4Rec을 설계하고, 상품 도시 코드를 추가로 임베딩함으로써 추천 성능 nDCG 0.015 향상
- **BERT4Rec 모델 개발**
  - 클릭 로그 데이터를 기반으로 Sequential 추천 모델 개발
  - NCF, BERT4Rec, DeepFM 모델의 오프라인 nDCG 성능 및 샘플 추천 결과 비교 후 BERT4Rec 선정
- **배포**
  - Pytorch를 기반으로 학습하여 젠킨스, AWS EC2, S3를 통해 모델 저장
  - Docker를 이용한 Flask API를 ECR로 Push 후 ECS에서 서빙

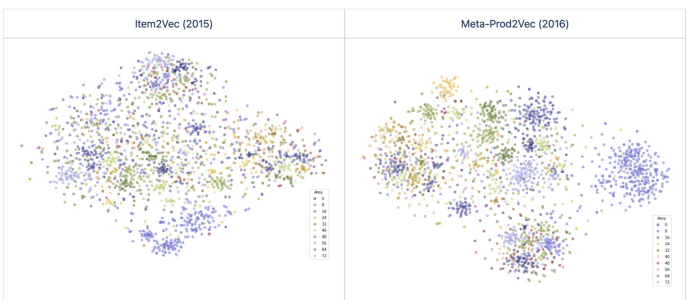
2. 함께 많이 본 상품 추천 모델 개발 : Meta-Prod2Vec

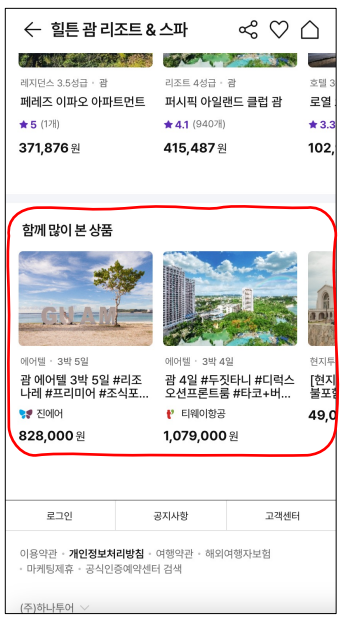
프로젝트 개요

- **목표** : 클릭 로그를 활용해 사용자들이 함께 많이 본 상품을 추천하는 모델 개발
- **기술 활용**
  - Word2Vec에서 영감을 받은 Prod2Vec 모델을 기반으로 설계
  - 손실 함수를 개선하여 메타 데이터를 반영한 Meta-Prod2Vec 구현

주요 내용

- **Prod2Vec 모델 설계**
  - Prod2Vec을 통해 상품의 벡터 표현을 얻을 수 있으며 상품 벡터간 코사인 유사도를 이용해 유사 상품을 추천
- **Meta-Prod2Vec 개선**
  - 하지만 Prod2Vec은 유저가 구매한 상품들의 시퀀스만 고려할 뿐, 아이템의 메타데이터는 사용하지 못하므로 Meta-Prod2Vec을 개발하여 도시코드 정보를 추가적으로 임베딩
  - 손실함수에 카테고리컬 데이터 관련 항을 추가하여 아래와 같이 더욱 명확한 상품별 임베딩을 얻을 수 있었음(색깔 : 도시를 의미)





하나투어 앱 내 상품 하단 '함께 많이 본 상품' 추천 영역

[1] MetaTransformer4Rec-Sequential Recommendation using Meta Information and Transformers

B. AI 추천 모델 학습 최적화

1. Mixed Precision 적용 및 Max length 조정

프로젝트 개요

- **목표** : BERT4Rec 모델의 학습 속도 개선을 위해 Mixed Precision 기법을 적용하고 Maximum Sequence Length를 최적화
- **기술 활용**
  - Mixed Precision : Float16 기반 연산으로 메모리 사용량 감소 및 학습 속도 증가
  - 추천 모델의 최적의 Max Length와 Batch Size 조합을 탐색.

주요 내용

- **Mixed Precision 요약**
  - 클릭 로그 데이터를 기반으로 Sequential 추천 모델 개발
  - 논문의 제안 방식 : 모델의 파라미터는 float32를 유지한 채 forward, backward, gradient를 float16으로 사용하는 것
- **Maximum Sequence Length가 성능과 학습시간에 미치는 영향 조사**
  - BERT4Rec의 원 논문에 의하면 성능 및 학습시간은 Maximum Sequence Length(N)의 영향을 많이 받음
  - 논문에 의하면 데이터셋 Beuty, ML-1m은 평균 시퀀스 길이가 다르며, 이에 모델의 최적 Max length는 매우 의존적임
- **실험 결과**
  - max length와 batch size의 여러 조합으로 nDCG와 Recall의 성능을 실험함
  - 에폭당 시간이 적고, nDCG와 Recall 성능이 높은 Max Length와 batch size의 최적값을 찾아 배포된 모델에 적용
  - 성능 유지 하에 학습 시간을 50% 이하로 단축

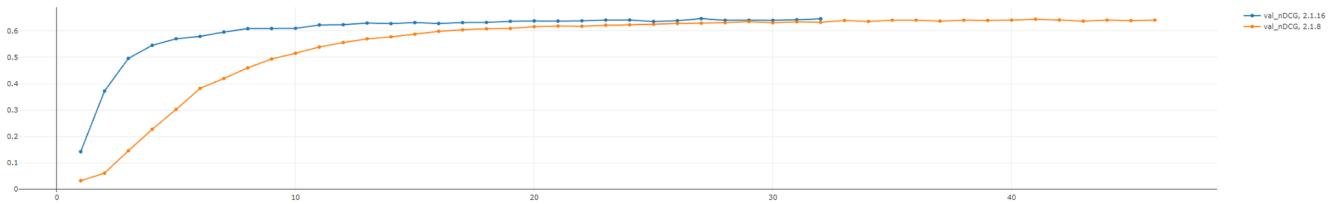
2. Exponential Decay scheduler 적용

프로젝트 개요

- **목표** : max epoch(=50)까지 학습되는 경우가 많음. Learning rate schedule을 적용하여 수렴속도를 향상시키고자 함
- **기술 활용**
  - 다양한 Learning Rate Scheduler 비교 후 적용(Exponential Decay, Multiplicative LR 등)
  - 최적 스케줄링을 통해 성능 안정화

주요 내용

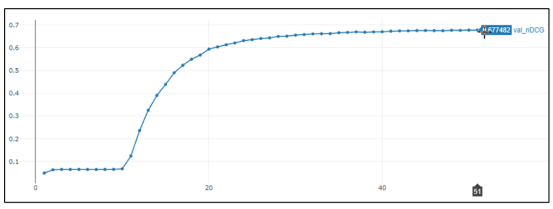
- **실험 설계**
  - Single Learning Rate와 Exponential Decay, Multiplicative LR, CosineAnnealingLR, OneCycleLR 4개의 scheduler를 비교
  - Learning rate는 기존에 사용 중이던 0.001과 함께, 각각 2배, 5배, 10배 증가시킨 0.002, 0.005, 0.01 등 다양한 값을 실험하여 비교
- **실험 결과**
  - 50번 에폭까지 가지 않고 30번대 에폭에서 수렴이 완료되는 가능성이 높아졌으며, nDCG도 소폭 향상됨
  - **Exponential Decay의 초기 Learning Rate의 최적값을 찾음**



As-is(주황)와 실험 Best 케이스(파랑)의 nDCG 그래프 비교. As-is에 비해 수렴 속도 향상

3. ReduceLROnPlateau 적용

- **목표** : 오른쪽과 같이 초기 에폭에서 nDCG가 올라가지 않는 현상이 간헐적으로 발생하여, 운영 배포 시 충분히 일어날 수 있기 때문에 예방책이 필요함.
- **해결 방법**
  - ReduceLROnPlateau라는 성능 향상이 없으면 learning rate를 줄이는 스케쥴러를 사용하여 해결



에폭에 따른 nDCG 그래프

C. 자연어처리 모델 개발

1. 개체명인식 / 문장 분류 모델 개발

프로젝트 개요

- **목표** : LLM 서비스인 여행정보AI에 들어오는 유저 질문에 대하여 Named-Entity Recognition(NER)과 Classification 모델 개발
- **기술 활용**
  - Huggingface Pretrained 모델 + Classifier 계층 추가로 데이터 fine-tuning
  - Flask API 사용 및 AWS ECS 배포

주요 내용

- **여행 질문 6,000건 이상 직접 라벨링**
  - 문장 분류 : 상품 추천, 일정 추천 등의 라벨 기준을 세우고 라벨링하여 데이터 관리
  - 개체명 인식 : 국가, 인원 등에 라벨링

Input : "일본 부모님과 갈만한 패키지 추천"

Output : {
"일본" : "국가",
"부모님" : "테마",
"패키지" : "상품 유형"
}

라벨링 수행 예시

- **BERT, ELECTRA, RoBERTa 성능 비교 실험**
  - DistilKoBERT, KcELECTRA, RoBERTa의 micro precision 성능 비교 후 best 모델 채택
- **AWS ECS 서버 부하테스트 수행 및 서빙**
  - 일정 TPS 이상 성공하기 위하여 Jmeter를 이용한 부하테스트 수행
  - GPU 인스턴스 도입 POC를 위한 부하테스트
  - CPU 서버와 GPU서버 부하테스트를 통한 안정성 확보

2. 자연어 처리 모델 성능 최적화 작업

프로젝트 개요

- **목표** :
  - NER(Named-Entity Recognition) 모델의 성능 향상
  - 학습 데이터의 부족으로 인해 발생하는 잘못된 개체명 인식을 개선
- **기술 활용**
  - 학습 데이터 증강 및 WordPiece Tokenizer 개선
  - Gunicorn Worker 메모리 최적화 작업

주요 내용

- **LLM을 이용한 NER 데이터 증강 > precision 0.1 이상 향상**
  - 학습데이터가 부족하여 생소한 지명을 인식하지 못하는 문제가 존재.
    - ✓ ex) "파타야"를 "파타"로 인식
  - 학습데이터에서 지명 태그 자리에 생소한 지명으로 대체하여 학습데이터 증강
  - 예시
    - ✓ [지역명] 자리에 자사 데이터상의 여러 지역명으로 대체 : 예시) "[지역명] 날씨는?"
    - ✓ [국가명] 자리에 자사 데이터상의 여러 국가명으로 대체 : 예시) "가족끼리 [국가명] 여행지 추천해줘"
  - 결과적으로, "파타"로 인식되던 "파타야", "인터"로 인식되던 "인터라켄" 등이 잘 인식됨
- **WordPiece tokenizer의 UNK 토큰 방지를 위한 자사데이터 토큰 추가 및 재학습**
  - '홋카이도' '스케쥴' '하얏트' 등의 단어가 roberta 토크나이저에서 UNK로 출력됨을 확인
  - UNK 토큰때문에 분류가 이상해지는 현상 발견 → UNK로 출력되는 단어를 Roberta의 토크나이저에 추가하는 로직 추가 & 학습하여 해결
  - 모델 및 토크나이저 로드 시 아래와 같은 코드 추가
  - 결과
    - ✓ '홋카이도'가 UNK 토큰으로 인해 잘 분류되지 못하던 문제가 해결
    - ✓ 이 외에도 '하얏트', '스케줄' 등의 단어가 들어간 질문도 제대로 분류됨
- **Gunicorn worker 메모리 이슈 해결**
  - gunicorn worker를 3개로 사용 중에, 한 서버에서 worker가 메모리 문제로 kill되는 문제 발생
  - 컨테이너 메모리 소프트한도에 비해 worker의 메모리 사용률이 조금만 초과해도 도커가 worker를 kill하는 것 확인
  - 개별 컨테이너의 메모리 확인 후 로컬 실험 수행
  - 3개에서 2개로 변경하여 해결

D. LLM 기반 추천 시스템 개발

1. 빅콘테스트 2024 참가 : LLM 활용 제주도 맛집 추천 대화형 AI 서비스 개발

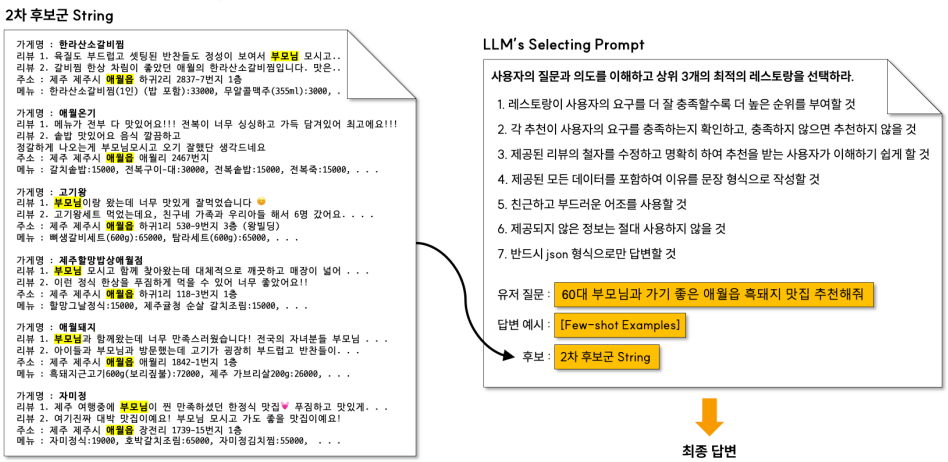
프로젝트 개요

- **목표** : 여행 그룹(가족, 커플, 개인 등)과 연령대별 맞춤형 맛집 추천 서비스를 개발
- **기술 활용**
  - LangGraph 기반 파이프라인 설계.
  - Text2Cypher, HyDE, PALR와 같은 추천 기법 적용.
  - Neo4j를 활용한 Graph 데이터베이스 구축.
  - Streamlit을 활용하여 웹 기반 사용자 인터페이스 구현 및 배포
- **결과** : **과학기술정보통신부 장관상(대상)** 수상

주요 내용

- **데이터 처리 및 추천 모델 설계**
  - 신한카드의 9,252개 가맹점명에 대해 카카오맵, 네이버지도, 구글맵에서 리뷰를 약 90만개 이상 수집하여 Neo4j에 적재
- **추천 프로세스**
  - Text2Cypher를 통한 1차 후보군 추출 > 리뷰 유사도를 이용한 2차 후보군 추출 > Graph Embedding을 통한 후보 보충
  - 2차 후보군(최대 6개)중에서 LLM이 최종 3개의 가맹점을 추천
- **개인 역할**
  - 팀장으로서 LangGraph 기반 파이프라인 설계 & 리뷰 Vector Search로 성능 검증 및 프롬프트 최적화 진행
  - HyDE, PALR, Few-shot Prompting, Text2Cypher 등 추천 기법 적용
  - Streamlit 인터페이스 개발 및 아웃풋 포맷팅

LLM 선택 프롬프트 ([2] PALR)



시연 화면

[2] PALR: Personalization Aware LLMs for Recommendation

2. 여행플래너 서비스 내 LLM을 이용한 패키지 상품 추천

프로젝트 개요

- **목표** : LLM을 이용한 서로 다른 DB상의 여행지 이름 매핑
- **기술 활용**
  - Few-Shot Prompting을 활용한 LLM 기반 매핑 기법 설계.
  - 2단계 필터링 알고리즘으로 매핑 정확도를 30% 개선.

주요 내용

- **매핑 파이프라인 설계**
  - 하나투어 데이터베이스와 구글 플레이스 데이터베이스 간 여행지 이름을 매핑하기 위해 LLM 기반의 Few-Shot Prompting 기법을 설계
  - 두 이종 DB의 여행지명을 1:N으로 여행지를 입력한 후 동일한 여행지를 찾으라 명령하여 1차적으로 매핑

- **매핑 불량 문제 개선**
  - 1차 매핑 결과에 대해 "근처에 있거나 같은 곳이면 yes, 아니면 no"라는 간단한 규칙으로 검증
  - 잘못 매핑된 것 중 87%가 삭제

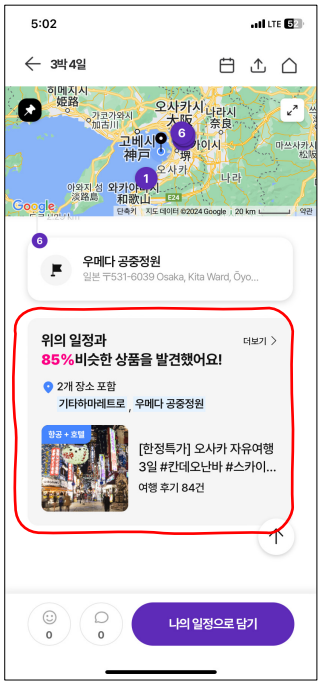
```
data['messages'][0]['content'] = "answer me 'yes' if A and B are located nearby or same places. if not, answer me 'no'"
data['messages'][1]['content'] = "A : Hozenji Yokocho\nB : Janjan Yokocho Alley"

'No'
```

'Yokocho'(골목)이 들어가서 잘못 매핑된 것을 1:1 필터링에서 제거 가능

- **배치 코드 개발**
  - 매칭 자동화 및 젠킨스를 이용한 신규로 추가된 여행지명에 대한 주기적 신규 매핑
  - OpenAI API 활용
  - 배치 Request 처리 진행 상태(validating, in_progress, completed)를 관리
  - 에러가 발생한 요청은 로깅하여 재처리 가능하도록 설계

- **추천 결과 예시**
  - 유저의 플래너 일정에 대하여, 매칭률이 가장 높은 패키지 상품을 추천



여행플래너 서비스 안의 일정 매칭률 높은 패키지 추천 영역