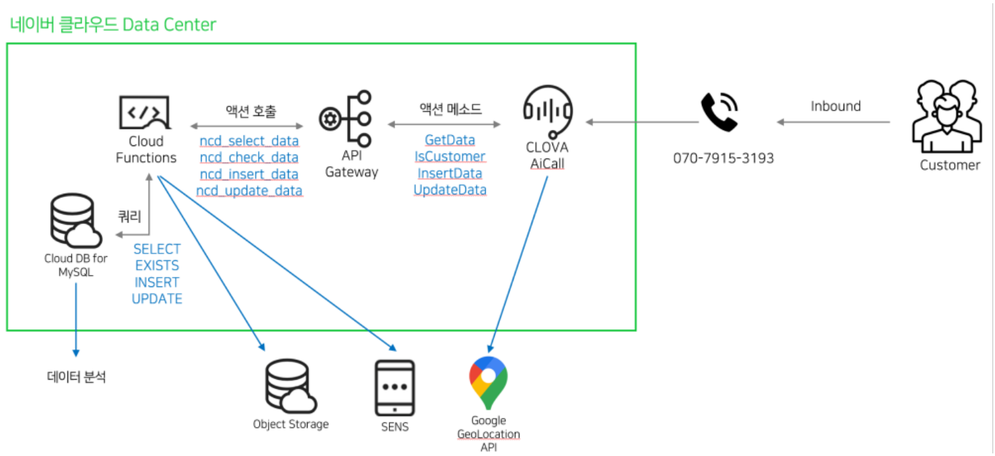
Work Experience

주제: 대리 운전 AiCall

컨택센터 인력이 부족한 곳은 많은 채용 공고를 올렸을 것이라고 생각함 -> 채용 사이트의 공고 1660개를 크롤링함

1	㈜블루포스	[중랑구
2	㈜케이에스한국고용	[유통업체/급여협의]
3	일본	유명쇼핑몰
4	NS홈쇼핑	이매
5	마르테디디	(영업x/월
6	#####	그룹영앤리치
7	SK렌터카	고객센터
8	JB우리캐피탈	[연4800만/정규직전환가능]JB우리캐피탈
9	나이스신용정보㈜	[월230만원/난이도下]기업평가
10	풍성FA	[놀은
11	㈜우리엔유	-
12	㈜윌앤비전	2023년
13	[대전시청역-KB국민	10-5시]
14	㈜	더블루밍
15	삼성생명	[내부콜/65만추가]삼성생명
16	㈜큐앤에스컴퍼니	[콜센터]
17	㈜기업금융센터	5시칼퇴/단순TM/영업성절대x/월300이상/신입경력무관/즉시
18	㈜트랜스코스모스코	[을지로]
19	하나손해보험(관리	[종로/정규/연평균3400만이상]
20	유니에스	[상담30%/사무70%월
21	㈜해피꽃배달	꽃주문
22	LG유플러스	[중앙동/일4시간야간/입사축하금/투잡가능]
23	KT	CS
24	효성ITX-현대카드오	[영등포역타임스퀘어/주5일/면접없음]
25	케이앤웍스	[카카오광고]
26	메가박스	드림센터
27	#NS홈쇼핑	#모란역
28	#NS홈쇼핑	#주야간
29	교육강사/채팅상담	[신도림]티오더
30	채팅상담위주.센터	[채팅상담위주]티오더

[업체 선정 과정]



[AICC 시스템 구성도]

근무 회사: 네이버 클라우드
부서: Digital Customer Experience
직무: AICC IT Architect 인턴
기간: 22.12 - 23.02

프로젝트 설명

- 인턴 기간 동안 받은 교육을 바탕으로 자신만의 AICC를 구축하기

*AICC란? Artificial Intelligence Contact Center의 약자로 기존의 상담원 기반의 컨택센터(콜센터)를 AI기반으로 전환한 것입니다.

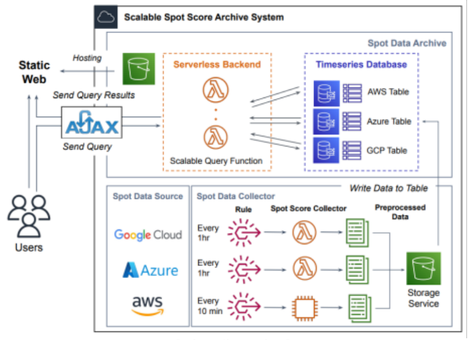
프로젝트 수행 내용

- AICC를 적용할 업체를 선정
  - 현재 채용을 활발히 하는 업체를 AI 기반으로 전환하면 이점이 많을 것이라 생각
  - 따라서, 컨택센터 채용 공고를 수집 후 자연어처리 및 단어 출연 빈도 분석을 통해 업체 선정
- 대리운전 AICC 시나리오 제작
  - NCP의 AiCall 기반의 통화 시나리오와 클라우드 컨택센터인 Bright Pattern 기반의 시나리오 제작
- AiCall 시스템 구축
  - 시나리오 기반으로 CLOVA Chatbot을 학습시킴
  - 액션 메소드를 활용해 서버리스 컴퓨팅 기반 API와 연동하여 고객 요청에 맞는 정보를 실시간으로 조회하고 응답하는 구조를 구현
- Bright Pattern 시스템 구축
  - AiCall과의 상호 전환이 가능하도록 TTS 기반의 클라우드 컨택센터 구현
  - REST API 호출을 통해 서버리스 컴퓨팅 기반 API와 연동하여 고객 데이터를 조회하고 확인하는 구조를 구현

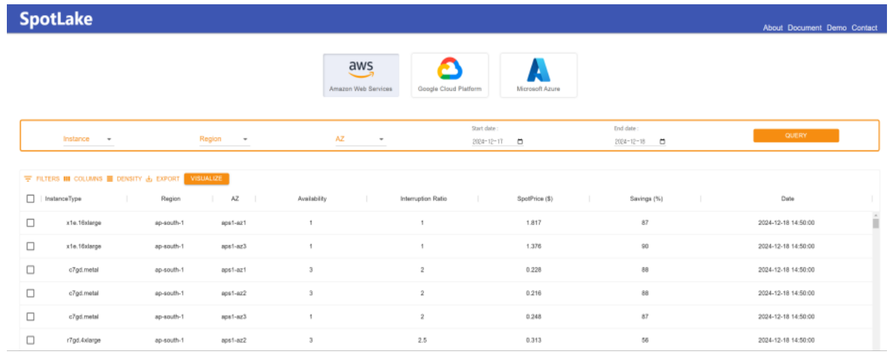
배운점

- 간단한 구성이지만, 클라우드 기반의 완전한 서비스를 직접 구축해봄
- 프로젝트 외에도 파트너사와의 미팅에 참여하고, 고객사를 방문하여 실제 직무를 체험하며 실무가 진행되는 과정을 파악함

Research Experience



[시스템 구성도]



[서비스 화면]

[GitHub issue를 통한 연구 내용 트래킹]

연구실: 국민대학교 DDPS Lab
직무: 학부연구생
기간: 22.03 - 22.12

프로젝트 설명

- SpotLake: 다양한 퍼블릭 클라우드 벤더사에서 수집한 스팟 인스턴스 데이터를 통합 제공하는 데이터 아카이브 서비스 서비스 링크

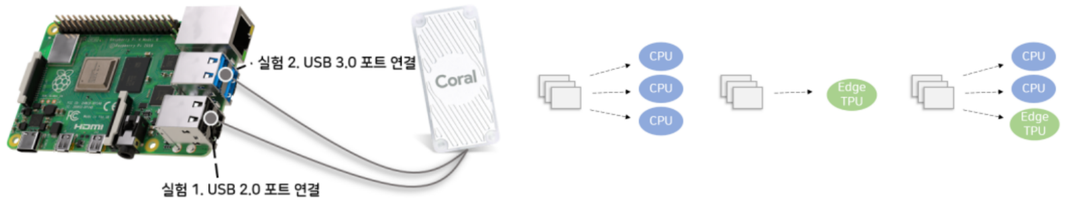
프로젝트 수행 내용

- AWS 온디맨드 인스턴스 데이터 수집 및 검증
  - AWS에서 제공하는 boto3 SDK를 활용하여 데이터 수집
  - 공식 홈페이지에서 제공되는 데이터와 수집된 데이터 간 일치성 검증 및 이상 데이터 탐지
  - 기존에 수집한 스팟 인스턴스 가격과 savings를 기반으로 계산한 예상 온디맨드 인스턴스 가격과 실제 수집 데이터 비교 분석
- Azure 스팟 인스턴스 데이터 수집
  - REST API 요청을 통해 데이터 수집
  - AWS Lambda를 활용하여 한 시간마다 수집 후 가공하여 Amazon S3에 저장
  - 멀티스레딩과 API 필터링을 활용해 수집 시간을 1400초 대에서 43초로 단축
- 수집 이상 모니터링 시스템 구축
  - 지수 백오프를 활용하여 api 응답이 API 응답이 제대로 이루어지지 않는 경우를 처리
  - 데이터 수집에 이상이 발생하면 Slack으로 메시지를 보내는 모니터링 시스템 구축

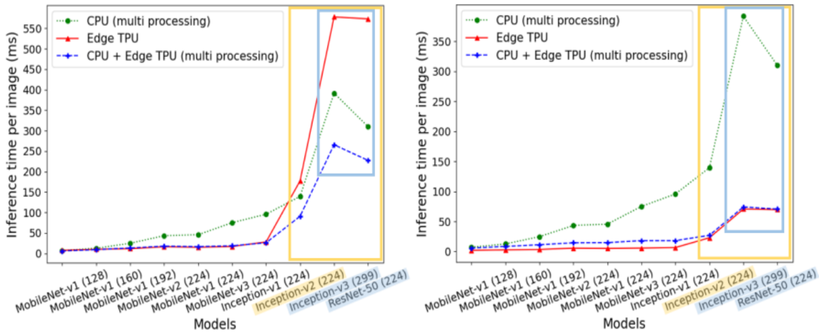
배운 점

- 크롤링뿐만 아니라, 필요에 따른 다양한 데이터를 수집 방법을 알게 됨
- Azure 데이터 수집의 경우, 멀티스레드 환경에서 마지막 페이지 도달 시 모든 스레드를 안전하게 종료하는 것이 중요했음. 데이터 양에 따라 마지막 페이지가 변하기에 이 방법에 대한 고민이 컸는데, event와 함수의 인자를 적절히 사용해서 해결함
- 데이터 수집 이후 검증 과정에서 데이터에 대한 수많은 조사와 표준화, 이상 데이터 판별 과정을 거쳐야 함
- 데이터 변화 주기에 따른 수집 주기 및 수집 방법을 선택하는 것이 중요함
  - 예를 들어, 변화가 잦은 데이터는 서버 기반 환경에서 수집 주기를 짧게 설정했으며, 변화가 드문 데이터는 서버리스를 활용해 비용 효율적으로 운영함

Research Experience



[실험 환경 구성]

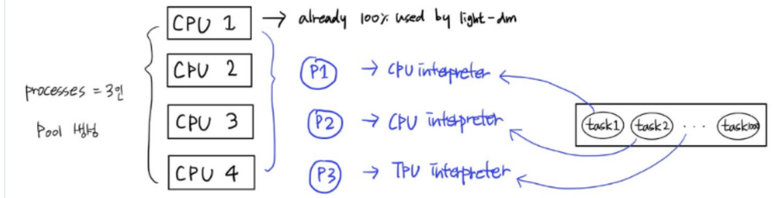


[Edge TPU를 2.0 포트에 연결 후 실험한 결과]

[Edge TPU를 3.0 포트에 연결 후 실험한 결과]

CPU	ResNet-50	2	224x224x3	25.0MB	0.757	0.132	21.394	356.507	0.357	1.403	2.645	2.805	378.021
EdgeTPU	ResNet-50	2	224x224x3	25.0MB	0.756	3.203	21.435	599.778	0.6	0.599	1.601	1.667	624.421
CPU+EdgeTPU	ResNet-50	2	224x224x3	25.0MB	0.756	3.919	21.535	228.581	0.229	0.9	3.996	4.675	250.25
CPU	InceptionV4	1	299x299x3	42.9MB	0.904	0.268	23.53	1036.613	1.037	4.105	0.943	0.965	1060.308
EdgeTPU	InceptionV4	1	299x299x3	42.9MB	0.903	3.263	23.594	1207.604	1.208	1.207	0.81	0.828	1234.465
CPU+EdgeTPU	InceptionV4	1	299x299x3	42.9MB	0.903	3.242	23.425	574.49	0.574	2.275	1.672	1.741	598.086

링크: https://docs.google.com/spreadsheets/d/17CN4zTTtzIDtX-VVn10b3eP5pNY3jAks/edit?usp=sharing&ouid=113624236219002214736&rtpof=true&sd=true



EdgeTPU와 CPU가 위와 같은 방식으로 동시에 추론을 진행할 수 있도록 해보았습니다.
모델의 크기가 커질수록 둘을 같이 사용했을 때 성능 개선이 컸으며, 작은 모델에서는 CPU나 EdgeTPU를 단독으로 사용했을 때와 비슷한 경우가 있었습니다.
위의 그림과 같이 3개의 프로세스를 만들어 하나는 EdgeTPU 추론, 두개는 CPU 추론에 이용한 경우 4개의 CPU를 모두 다 100%로

[GitHub issue를 통한 연구 내용 트래킹]

연구실: 국민대학교 DDPS Lab
직무: 학부연구생
기간: 22.03 - 22.12

프로젝트 설명

- **Edge Inference**: Raspberry Pi 4의 CPU와 Google Coral USB Accelerator의 Edge TPU를 사용하여 CNN 모델 추론 속도 최적화 연구 [코드 링크] [논문 링크]

*Edge TPU란? Google에서 개발한 저전력 AI 가속기로, 딥러닝 추론 작업을 빠르고 효율적으로 처리하도록 설계되었습니다.

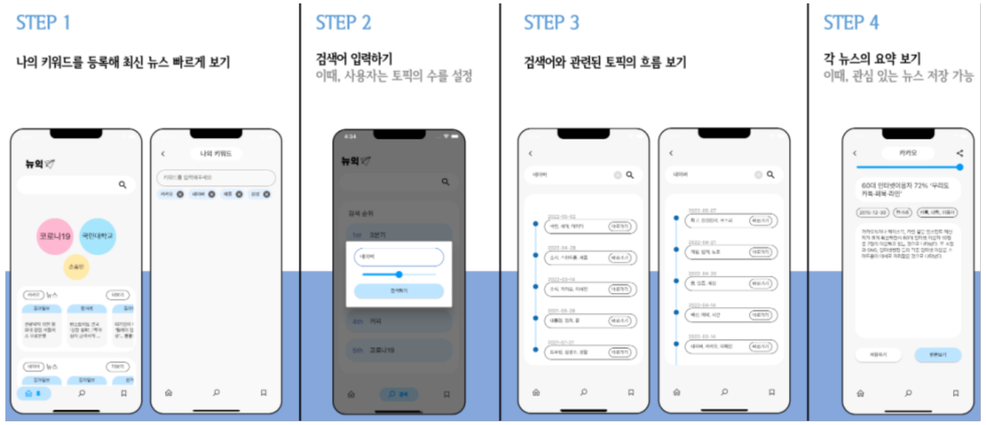
프로젝트 수행 내용

- 배치 추론
  - CPU 배치 추론: 다양한 CNN 모델을 CPU에서 배치 추론을 해보았지만, 추론 속도에 큰 차이가 없었음
  - Edge TPU 배치 추론: Edge TPU에서는 배치 크기가 1인 추론 또는 채널 수가 1인 이미지 배치 추론만이 가능
- 멀티 프로세서 (CPU와 Edge TPU) 추론
  - CPU 단독 멀티프로세싱, CPU와 Edge TPU를 병렬로 사용하는 멀티프로세싱, Edge TPU 단독 추론 시나리오를 설계하고, 각각의 추론 속도를 비교 및 분석
  - 실험 중 특정 프로세스가 다른 프로세스에 비해 더 많은 작업을 처리해 늦게 종료되는 문제가 발생
    - Python의 Pool.map()으로 각 프로세스에 이미지를 보내주는데, 기본 chunk 단위인 8에서 1로 수정
  - 멀티 프로세스로 추론 시 각 이미지를 IPC로 전달하는데, 여기서 약 4ms 병목현상 발생
    - 연구 이후에 나온 유사한 논문 중 멀티스레딩을 활용한 사례를 확인 [유사 논문 링크]
      - 이를 통해 IPC 병목현상을 제거할 가능성을 검토하였으나, Python GIL 고려 필요
- USB 대역폭에 따른 추론 성능 분석
  - Coral USB Accelerator는 8MB SRAM을 사용하며, 모델 크기가 SRAM 용량을 초과하는 경우, 파라미터를 지속적으로 호스트 디바이스와 교환해야 함
  - 이로 인해 MobileNet과 같이 크기가 작은 모델에 비해, Inception 모델과 같이 큰 모델에서 3.0포트를 이용하면 큰 성능 개선을 볼 수 있음

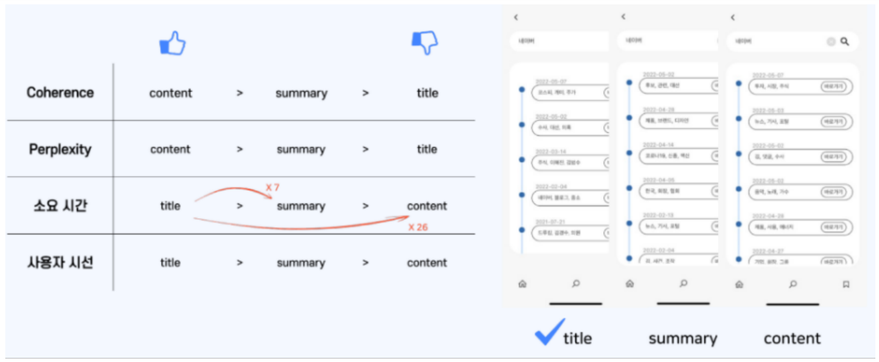
배운 점

- GPU와 같은 고가의 장비 없이도 Edge TPU와 같은 저비용 AI 가속기를 활용해, CPU만 사용할 때보다 뛰어난 추론 성능을 얻을 수 있음을 확인함
- 다만, 가속기의 한정된 메모리 자원으로 인해 기존 모델을 그대로 사용할 수 없었으며, 양자화가 필수적임을 알게 됨. 또한 이 과정에서 여러 제약 사항들이 있음을 알게 됨
- 멀티 프로세스를 활용하여 CPU와 Edge TPU에 작업을 할당하는 방법을 알게 됨

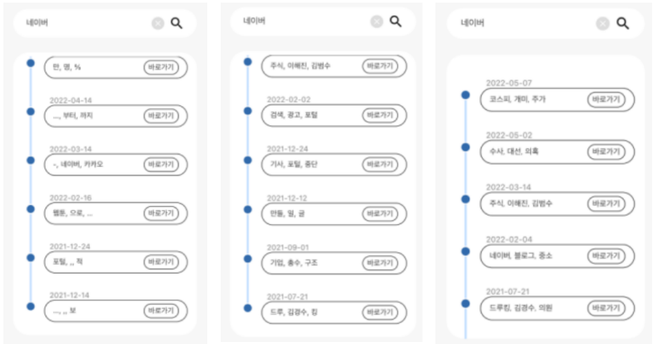
Projects



[주요 기능]



[인자값에 따른 토픽 타임라인]



[데이터 전처리에 따른 토픽 타임라인]
가장 왼쪽이 품사 필터링, 사용자 사전 적용 후의 모습

프로젝트: 뉴익 (캡스톤 프로젝트)
인원: 4명
기간: 22.03 - 22.06

프로젝트 설명

- 뉴익: 뉴스 기사 데이터 기반으로 검색어에 대한 일련의 토픽들을 키워드로 구성하여 시기별 흐름을 알려주는 앱 코드 링크

프로젝트 수행 내용

- 데이터 수집
  - 데이터 선정:
    - 크롤링이 허용된 언론사(국내 주요 일간지 등)를 선정하고, 열독률, 구독자 수 등을 고려하여 한겨레, 동아일보 선정
  - 수집 방식:
    - Python Scrapy를 활용하여 최근 10년간의 기사 제목, 기사 본문 등을 1차적으로 수집. 이후 매일 새롭게 올라오는 기사를 EC2 서버에서 cron 스케줄링을 통해 추가 수집.
    - 기사 본문을 토대로 요약 데이터도 생성.
- 데이터 전처리
  - 불필요 텍스트 제거:기자 이메일, [속보]와 같은 말머리 등 분석에 방해가 되는 요소들을 정규식으로 제거
  - 알파벳 소문자로 변환
  - 형태소 단위 토큰화: Mecab 사용하여 조사, 감탄사 등 필요치 않은 품사를 제거하고, 분석에 유의미한 핵심 어휘(명사 등)를 추출
  - 사용자 사전: 신조어, 고유명사 등이 형태소 분석에서 누락되지 않도록 사용자 사전을 구축
- 토픽모델링
  - coherence와 perplexity 지표를 참고하고, 사람들을 대상으로 설문조사를 진행해 모델의 인자값 결정
  - 입력 데이터: 기사 제목, 기사 본문, 기사 본문 요약본을 가지고 성능(모델링에 소요되는 시간, 결과에 대한 설문조사)을 평가해 기사 제목으로 결정
  - 토픽 타임라인
    - 검색어에 대한 토픽의 흐름을 타임라인 형태로 시각화하여 제공
    - 이렇게 구한 토픽 타임라인은 각 토픽에 해당하는 기사 정보와 함께 MongoDB에 적재

배운 점

- 성능 평가 기준이 명확히 없는 상황에서 정성적 평가가 도움이 됨. 추후에는 A/B 테스트가 가능하도록 서비스를 구축해보고 싶음.
- 토픽 모델링같이 실시간 처리가 어려운 문제의 경우 매일 토픽모델링을 만들어놓도록 스케줄링하는 방법이 있음
- MongoDB 쿼리 성능을 인덱스 설정으로 개선함