**최종 프로젝트**

**기업 요구사항 기반의 문제해결 프로젝트**

**프로젝트 기간 : 8월 25일 ~ 10월 8일**

**일정**

PJT 기획

* 8월 25일(수) ~ 8월 27일(금)
* 프로젝트 기획
  + 프로젝트 주제 최종 확정 및 기획안
  + WBS 작성
* 산출물 : 기획안, WBS

PJT 수행 및 완료

* ~ 10월 7일(목)
* 프로젝트 수행
  + 프로젝트 설계, 구현 및 테스트
* 프로젝트 완료
  + 포트폴리오 작성
  + 마무리 및 발표 준비
* 산출물 : 수행일지

발표

* ~ 10월 8일(금)
* 프로젝트 발표 및 평가
  + 프로젝트 발표
  + 평과 결과, 시상
* 산출물 : 포트폴리오

**5조**

팀명 : OJO

팀원 : 김정명, 박성하, 최원희, 김태리, 이민성(조장) (DE DS)

**8월 25일 (수요일)**

1. 주제선정
2. 프로젝트 진행 정리
   1. 기획서
   2. WBS

**주제**

**교통 공공데이터를 활용한 교통사고 위험도 예측 모델 개발**

**프로젝트 진행 정리**

결과 산출물 : 코드정리, 시각화, ppt

**전체 과정**

1. 코드 정리 :
   1. 작업하는 도중 주석달기
   2. 작업 완료했으면 구글 드라이브에 올리기 (하나의 작업이 끝났을 시 문서화)
   3. troubleshooting 정리 필수
   4. 그렇게 코드를 짠 이유
2. 시각화 :
   1. DS 분석 시각화
   2. django 인터렉티브 시각화
3. PPT :
   1. 구글 드라이브에 정리한 내용 토대로 작성
   2. DE 부분은 DE가 DS 부분은 DS가

**DE & DS 세부 작업 내용**

DE : 배치, 실시간, 자동화, 인터렉티브 시각화(django) (툴을 왜 썼는지 이유)

* pipeline
  1. 배치 : spark
  2. 실시간 : kafka, spark
  3. 자동화 및 관리 : airflow
  4. 데이터베이스 : mongoDB(Hbase), mysql(HDFS) → python
* django 백엔드 부분 구현

DS : 데이터 분석 → 시각화 →  인사이트 도출 →  모델링 → 데이터베이스 연결 → 시각화 (plotly dash / etc...) 인터렉티브 시각화를 할 수 있는 종류

**분석**

spark  sparkSQL / spark streaming / spark MLib

spark를 두 부분 작업 공간 분리

1. 자동화에 맞는 스파크 (sparkSQL, spark streaming) ⇒ django
2. 분석에 쓸 수 있는 스파크 (sparkSQL, spark MLib)

aws → ubuntu → jupyter notebook → spark → Dataframe → To Pandas → 데이터분석

**단계**

ETL / ELT

배치 / 실시간

단일 아키텍처 / 람다 아키텍처

수집

**8월 26일 (목요일)**

DE

- 데이터 수집 – open api 인증키 발급✔

* 1. 서울시 교통정보(seoul TOPIS) open api
  2. 교통사고정보 개방시스템 (TAAS) open api
  3. 한국도로공사 고속도로 공공데이터 포털 open api

- 파이프라인 구축

: spark, airflow, kafka

- DE파트 기획안 / WBS 작성

DS

- 변수 선정 완료 ✔

- 실시간 데이터 모델링 방법

- y변수 접근 방법

- 위험도 지표 활용 여부

- DS파트 기획안 / WBS 작성

**8월 27일 (금요일)**

1. 기존 주제 토론

* DS : 예측 모델
* DE : DB 연동
* TAAS 벤치마킹 (시간대별, 지역별 구체적인 교통사고 발생 기록)

1. 새로운 주제 토론

1. 도시 재생 사업

2. collocation(연어)

1. 기획안 / WBS 작성

새로운 기법 도입 위한 주제 변경

기업요구사항 ⇒ 고객이 필요한 서비스에 대한 needs 파악, 충족 / 평가 고려

**2-1. 도시 재생 사업**

도시 개발 완료 / 안된곳 비교 분석하여, 도시 개발이 완료 된 지역의 특성을 추출

비슷한 특성을 가진 개발이 완료되지 않은 도시들의 사업 형태, 산업 추천 모델

* 기법 : 텍스트 마이닝 → 특징 추출
  + 뉴스, sns
  + 키워드 : 도시 재생 사업 지역명(장소명 기반), 쇠퇴 지역, 낙후 지역

**2-2. collocation**

서비스 대상자 : 영어가 모국어가 아닌 사용자

영어가 모국어가 아닌 사용자가 영작할 때 주로 맥락에 대한 이해가 부족함

이를 위해 보다 더 자연스러운 영작을 위해 높은 빈도로 나타나는 어휘의 쌍 추천

1. 크롤링을 통한 워드 카운트 (주로 formal한 문장 ex)영문으로 된 뉴스, 논문, 소설)
2. sns(트위터, 인스타그램, 페이스북 ···)과 유튜브 실시간 스트리밍 api를 이용한 informal한 문장 처리

**참고 자료**

NLP 이해하기

<https://hryang06.github.io/nlp/NLP/#nlinatural-language-inference-%EC%9E%90%EC%97%B0%EC%96%B4-%EC%B6%94%EB%A1%A0>

연어 이해

<https://www.fluentu.com/blog/english-kor/%EC%98%81%EC%96%B4-%EC%97%B0%EC%96%B4/>

딥러닝이 충분히 탐구하지 못한 언어 관련 5가지 태스크

<https://www.kakaobrain.com/blog/36>

Week 22 - 딥러닝 기계번역 모델 seq2seq 깊게 파보기

<https://jiho-ml.com/weekly-nlp-22/>

**각 파트별 수행해야 할 것**

DE 파이프라인 구축, 웹 크롤링 후 DB연동

DS 분석, 모델링, 사례 벤치마킹

**8월 28일 (토요일)**

**1차 멘토링**

- 범위 줄이기(도메인 영역) → formal / informal 중 영역은 informal한 일상대화(SNS) 기준으로 선택

- 제목 변경

기존 : collocation기반 영단어 검사 및 추천 시스템 구현

변경 : 비영어권 국가의 SNS사용자를 위한 문맥기반 유의어 추천 서비스 구현

- 단어 수준 추천만 진행

- 모델링 : CNN-LSTM 쪽으로 진행 예정

**8월 31일 (화요일)**

오전 - 주제 고도화를 위한 브레인스토밍 진행

**브레인스토밍 진행 결과**

- 영역 → SNS (Twitter, Reddit)

- 대상 → 영어를 어느정도 알고 있고, 자연스러운 영작을 원하는 사용자

- 상대방과의 관계(분위기)에 따른 collocation추천

- 실제 활용사례를 row data에서 문장단위로 추천

- 1차 기능 : 일상어

2차 기능 : formal 문장

- 부가기능 생각해볼 것

1. 영작 체크 고도화

2. 추천이 안되는 상황 고려 (단어 존재하지 않거나, 오탈자가 발생한 경우)

- 키워드 추출 방식

1. 수동 (사용자가 원하는 단어 직접 선택)

2. 자동 (사용자가 선택하지 않아도 모든 단어의 collocation 추천)

**참조 사이트**

- <https://wordsimilarity.com/>

**오후 일정 (산출물)**

1. 기획안 작성 ✔

2. 유스케이스 작성 ✔

3. DFD ✔

4. 아키텍처 ✔

5. WBS ✔

**수행 상세 내용**

DS

Usecase 작성 → 산출물

DE

**AWS 서버 오픈 이후 파이프라인 구축 예상**

Local ↔ Server

- (배치) 크롤링<crontab> → HDFS → Spark → MongoDB

- (실시간) Open API → Kafka → Spark Streaming → MongoDB → Spark → MySQL(정규화, ERD) → Django → DashBoard / Service

**파이프라인 구축 이후 수행할 것**

1. 데이터 수집

2. 데이터 정제

3. 데이터 셋 생성

4. 자동화

5. Django 백엔드 연결

**데이터 수집 방법**

1. 뉴스 크롤링

2. twitter, Reddit open api (불용어 제거 처리)

**Database (스키마?)**

1. RDBMS

2. NoSQL

3. 파일 분산 시스템 (HDFS)

**데이터 흐름도(DFD) 작성** → 산출물

**Conceptual Architecture 작성** → 산출물

**참조**

Twitter API를 활용하여 실시간 tweet을 Kafka로 보내기

<https://velog.io/@shinychan95/Twitter-API%EB%A5%BC-%ED%99%9C%EC%9A%A9%ED%95%98%EC%97%AC-%EC%8B%A4%EC%8B%9C%EA%B0%84-tweet%EC%9D%84-kafka%EB%A1%9C-%EB%B3%B4%EB%82%B4%EA%B8%B0>

hdfs, dfs 명령어 정리 및 설명

<https://blog.voidmainvoid.net/175>