Software Requirement Specification for Film Review Dematerializer

Project Team
Team #4
201411273 박재범
201411275 박진호
201411293 이상민
201511244 김민우

Date
2020-09-13

Table of Contents

1. Introduction

1.1. Purpose

본 문서는 다음 프로젝트에 대한 요구사항, 개발 및 설계 관련 명세서이다.

"문화체육부와 국립국어원에서 공동 주최하는 국어 정보처리 시스템 경진대회의 2020년도 주제인 '딥러닝 기반 영화평 감성 분석' 주제를 바탕으로, Pre-Trained KoELECTRA 모델을 기반으로 하여 영화평 분석 Binary Classification 모델을 개발한다. 한국어로 작성된 영화평을 읽어 들여 해당 영화평의 긍정/부정을 분류하는 것이 주요 목적이며, KoELECTRA에 독자적으로 개발한 NN 레이어 아키텍처를 융합하여 Fine-Tuning을 통해 성능 및 정확도를 향상시키고, 해당 모델을 기반으로 구성한 다양한 기능을 가진 사용자 및 관리자 웹 인터페이스 통해 편리하게 활용할 수 있게 한다. [프로젝트명: 리뷰 해체 분석기(Film Review Dematerializer)]"

해당 문서를 통해 개발자와 관리자, 사용자의 프로젝트 이해를 돕는 것이 목적이다.

1.2. Scope

프로젝트명 '리뷰 해체 분석기(Film Review Dematerializer)'는 크게 인터페이스와 언어 모델 두 가지 파트로 구분할 수 있는 소프트웨어의 개발을 목표로 하며, 프로젝트명과 소프트웨어명은 동일하다.

[언어 모델]

한국어로 쓰여진 영화평을 읽어들여 해당 영화평의 평가 의도가 '긍정'인지 '부정'인지 구별 및 판단(Classification)할 수 있는 언어 모델을, 딥러닝 인공 신경망을 활용하여 설계한다. 2020년 8월 기준 Classification 언어 모델 분야에서 State-Of-Art 중 하나인 ELECTRA를 한국어에 맞도록 Pre-Trained한 모델 KoELECTRA의 기술을 적극 활용하되, Pure KoELECTRA-Small 모델 기준으로 NSMC에 대해 보여주는 88.76의 정확도를 데이터 전처리 및 NN Layer 아키텍쳐 구성 등 다양한 아이디어를 접목하여 90 이상으로 개선하는 것을 최우선 과제로 둔다.

이에 적용한 아이디어를 소개하면 다음과 같다.

우선 입력의 형태가 기본적으로 '한국어로 작성된 영화평'이나, 사람이 작성하는 것이기 때문에 오타, 띄어쓰기, 이모티콘 등이 존재할 수 있다고 가정한다. 이러한 요소들은 모델 분석의 정확도를 감소시키는 요인이 되므로 Raw-Data를 모델에 그대로 입력하지 않고 전처리를 거쳐 불필요한 부분을 제거하고 분석에 최적화 된 포맷으로 재구성한다.

두 번째로, 재구성한 입력 문장을 Pre-Tranied/Fine-Tuned KoELECTRA 모델의 입력 규칙에 맞게 Word-Embedding하여 토큰으로 변환한다. 이는 [CLS], [SEP] 토큰을 포함하며, 해당 토큰들을 KoELECTRA의 입력으로 집어넣는다.

세 번째로, KoELECTRA의 출력인 Hidden State를 활용한다. 이 때, [CLS] 토큰의 Hidden State만으로도 긍정/부정 Classification이 가능하며 해당 토큰만 사용할 때의 정확도가 88.76이다. 하지만 정확도 향상이 프로젝트 언어 모델 개발의 목표이므로 아래 (1)~(4)과 같은 추가적인 과정들을 거친다.

(1) 14.000여개의 한국어 문장/단어들의 감성 정보를 매우 부정부터 매우 긍정까지 –2, -1, 0, 1, 2의 5가지의 Label로 분류해 놓은 한국어 감성 사전을 활용하여, 입력 문장과 비교하고 토큰 개수에 맞추어 가중치 벡터 형태로 만든다. [CLS] 토큰과 무관하게 문장 자체, 그 중에서도 각 어절의 의미를 대상으로 독립적인 Classification을 도출하기 위한 작업이다. 단, 감성 사전에는 최대 8-gram의 문장이 존재하고, KoELECTRA의 Vocabulary는 WordPiece 단위로 관리되기 때문에 토큰 별 1:1 대응은 불가능하다. 따라서 이에 대한 해결 방안으로 한국어 감성사전을 KoELECTRA의 Vocabulary 토큰으로 변환한 뒤 Bi-gram 등을 활용한 비교를 적용하여 최적의 가중치를 부여한다. 이 때, 토큰들이 감성사전의 n-gram에 대응되는 것으로 결정되면 가중치 벡터에 1/n을 스칼라곱 하여 다중 단어 문장에 대한 가중치를 희석시킨다.

(2) 한국어 감성사전은 일반적인 한국어 표현들을 대상으로 감성 분석을 진행하였

기 때문에 '영화평'에서의 활용과 의미가 상반되거나 하는 문제, 즉 오차가 발생할 수 있다. 예를 들어, 영화를 보고 감동을 받아 긍정적으로 작성된 "상영시간 내내 울었다 ㅠㅠ" 라는 영화평 문장에 대해서 감성사전을 적용하면 '울었다', 'ㅠㅠ' 모두 매우 부정의 의미를 담고 있기 때문에 우리가 기대하는 최종 분석 결과로 '긍정'을 도출하는 데 불리하거나 학습(Fine-Tuning) 시 올바르지 않은 가중치 반영이 일어날 수 있다. 그렇다고 14000여개의 감성 사전 분석 결과를 예외적인 상황을 고려하여 모두 수정하는 것도 불가능하다. 따라서 문맥 전체의 의미를 내포하는 [CLS] 토큰을 통해 각 단어에 보정을 가하여 원래 의도와 상반되는 토큰의 영향력은 적게, 일치하는 토큰의 영향력은 강하게 만들어준다. 이 때, Scaled Dot-Product Attention 기법을 적용한다.

(3) SDA를 적용한 벡터에 (1)로부터 도출된 가중치 벡터들을 Concatenate하고 Bi-LSTM에서 정확도는 유지하고 성능을 향상시킨 Bi-GRU에 입력으로 집어넣는다. 문장의 감성 정보가 앞에 치우친 경우, 뒤에 치우친 경우 모두를 고려하여 단방향이 아닌 양방향 GRU를 사용한다. 해당 레이어를 거쳐 출력 벡터를 뽑아내고 이후 활용을 위해 Projection을 진행한다.

(4) 최초 KoELECTRA의 [CLS]와, Bi-GRU에서 출력된 벡터들을 FFNN에 통과시켜 2차원의 벡터로 도출한다. 해당 벡터를 통해서 긍정/부정을 판단할 수 있다.

이러한 과정들의 적용을 통해서 기존 모델보다 향상된 90 이상의 정확도를 도출해 낼 수 있음을 기대한다.

[인터페이스]

아무리 뛰어난 모델이라고 해도 사용이 불편하면 의미가 없다. 따라서 딥러닝에 대한 지식이 없는 일반인도 손쉽게 영화평 감성 분석 기능을 사용할 수 있도록 인터페이스를 구성한다.

접근 및 관리가 용이하도록 웹 인터페이스를 적용하며, 사용자 페이지와 관리자 페

이지로 구분하여 목적에 따라 활용할 수 있도록 한다.

사용자 페이지에서는 불필요한 기능을 최소화하고 명확성과 직관성을 중요시하여 구성한다. 메인 페이지에서는 영화평 입력, 제출, Excel 파일 연동을 통한 여러 개의 영화평 동시 입력이 가능하며, 결과 페이지에서는 입력에 대한 분석 결과에 대해 다양한 통계 및 그래프 형태로 보여준다.

관리자 페이지에서는 기본적으로 모델 버전 별 정확도를 시각적으로 보여주어 버전 관리가 편리하도록 하였으며, 적용시킬 모델 버전을 선택할 수 있어 항상 높은 성능의 버전을 기준으로 유지 관리가 용이하도록 한다. 또한 언제든지 선택된 모델을 기반으로 DB의 누적 Data Set을 통한 Fine-Tuning을 진행하고 새로운 버전을 생성할 수 있다. 이를 통해 모델의 지속적인 정확도 향상 또한 기대할 수 있다.

인터페이스 및 입출력 데이터는 기본적으로 서버에 탑재된 언어 모델 및 DB와 연동되어 유지 관리된다.

1.3. Definition, Acronyms, and Abbreviations

1.3.1. Acronyms

NN: Neural Network

NLP: Natural Language Processing

RNN: Recurrent Neural Network

FFNN: Feed Forward Neural Network

NSMC: Naver Sentiment Movie Corpus

CLS: Classification

LSTM: Long Short-Term Memory models

GRU: Gated Recurrent Units

SDA: Scaled Dot-Product Attention

1.3.2. Definition

Binary Classification:

머신러닝 모델에서 Classification이란 입력들을 속성에 따라 여러 Class로 분류하는 것을 의미하며, 해당 프로젝트의 목적은 긍정/부정의 도출이므로 긍정과 부정 2가지의 Class로 분류하는 Binary Classification을 적용한다.

Fine-Tuning:

주로 일반 대용량 데이터를 통해 Pre-Trained된 모델(ex. KoELECTRA)을 기반으로 새로운 목적에 맞는 추가 데이터를 입력하여 모델의 Parameter, Weight를 미세 조정하는 것을 의미한다. 학습이 이루어지지 않은 모델을 Training하는 데에는 방대한 시간과 자원이 소요되기 때문에 본 프로젝트에서는 그 방향성과 규모에 맞추어 Fine-Tuning을 통해 추가 학습을 시킨다.

Word-Embedding:

단어를 밀집 벡터(Dense Vector)의 형태로 표현하는 방법으로, 주로 Word2Vec 기법을 적용하여 학습 데이터 기반의 실수 타입 벡터를 구성한다. One-Hot-Vector 등의 희소 표현과 비교되는 장점은 비슷한 의미의 단어는 비슷한 공간에 위치하게 되기 때문에 단어의 의미적 유사성이 중요한 자연어 처리에 부합하는 임베딩 방법이다.

WordPiece:

단어 토큰화 기법으로, 접두사, 접미사 등 자주 이용되는 Unit과, 자주 등장하지 않는 Subword Unit으로 나누어 Vocabulary를 구성한다. 즉, 빈도수 기반으로 사전을 구성하기 때문에 제한된 크기의 사전을 구성할 때 효율적이다. OOV 이슈가 존재하는 워드 임베딩과 학습 난이도가 높은 캐릭터 임베딩의 장점을 합쳐놓은 것으로, ELECTRA에서 사용되는 토큰화 방법이다.

Bi-Gram:

N-Gram의 일종으로, 문장을 이루는 여러 단어들을 연속한 N개씩 묶어서 이 단위로 여러가지 처리를 수행하는 기법을 말한다. 본 프로젝트에서는 한국어 감성사전을 활용할 때 입력 문장과 가장 일치도가 높은 감성 문장을 탐색하는 용도로 사용된다.

Scaled Dot-Product Attention:

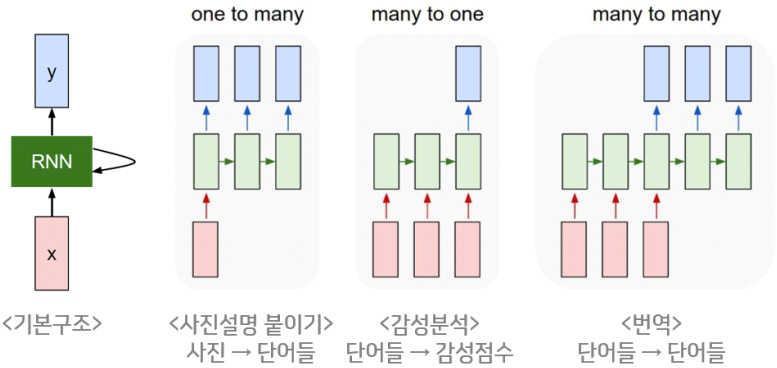
$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V$$

여기서 Q,K,V는 각각 dimention K의 Query와 Key, dimention V의 Value이다. Key와 Value는 같은 단어를 가진다. Attention 개념에 Scailing을 적용하여 행렬 연산 효율을 높인 것이다. 본 프로젝트에서의 사용을 예로 들어 간단히 설명하자면, 각 단어가 Classification 결과 도출에 얼마나 영향을 주었는지를 총 합이 1인 확률 형태로 도출하여 가중치로 곱해줌으로써 영향을 많이 준 단어에 집중할 수 있도록 하는 것이다.

Projection:

N차원의 벡터를 M차원의 벡터로 변환하고자 할때 M차원의 Projection Layer와의 행렬곱을 수행하는 것이다. 행렬 연산에 있어서 차원 일치가 필요한 경우가 있기 때문에 사용한다. 별도의 활성화 함수가 존재하지 않는 선형 변환이다.

Recurrent Neural Network:



RNN은 히든 노드가 방향을 가진 엣지로 연결돼 순환구조를 이루는(Directed Cycle) 인공신경망의 한 종류이다. 이전 토큰에 대한 정보를 기억하고 있기 때문에 음성, 문자 등 순차적으로 등장하는 데이터 처리에 적합한 모델이다. Image Labeling, Sentiment Classification, Translation에 주로 사용된다.

1.4. Reference

1. IEEE Std. 830-1998

2. [논문] 박천음, 김건영, 황현선, 이창기, "문맥 표현 기반 한국어 영화평 감성 분석 (Contextualized Embedding-based Korean Movie Review Sentiment Analysis)", 2018

3. [논문] 박천음, 이창기, "BERT 기반 Variational Inference 와 RNN 을 이용한 한국어 영화평 감성 분석 (Sentimental Analysis of Korean Movie Review using Variational Inference and RNN based on BERT), 2019

4. KNU 한국어 감성 사전

5. PyTorch Tutorial

6. KoELECTRA Github

7. KoNLPy 공식 홈페이지

1.5. Overview

해당 SRS에서는 IEEE Std. 830-1998 표준에 맞추어 KoELECTRA 기반 Binary Classification 모델과 인터페이스 관점 모두에 대한 기능, 특징, 요구사항, 소프트웨어 설계 등의 전반적인 내용을 다룬다.

Chapter 2에서는 시스템에 대한 Overall Description을 설명하며, Chapter 3에서는 시스템에 대한 Specific Requirement를 설명한다.

2. Overall Description

2.1. Product Perspective

전체 제품은 KoELECTRA 기반 Binary Classification 모델(이하 모델)과 웹 인터페이스(이하 인터페이스)로 구분할 수 있다.

모델은 Pre-Processing을 통해 Input(Test Set)을 학습에 최적화된 형태로 수정하고, KoELECTRA를 통과시킨 뒤 CLS의 Hidden State와 나머지 Hidden State들의 Scaled Dot-Product Attention하고, 한국어감성사전을 활용한 가중치를 각 벡터에 Concatenation하여 Bi-GRU를 통과시킨다. 그리고 CLS의 Hidden State와 Bi-GRU의 모든 각 벡터들(차원을 일치시키기 위한 Projection Layer 통과)을 FFNN의 입력으로 하여 최종 Binary Classification 결과를 도출한다.

가중치의 최적화는 Back-Propagation을 통해서 모든 Layer에 적용된다. 즉, 모델 전체에 대한 Fine-Tuning이 이루어진다.

인터페이스는 웹을 통해 제공되며 Input의 입력과 출력의 확인이 가능한 사용자 페이지와 모델 버전 별 통계 확인 및 설정, Fine-Tuning이 가능한 관리자 페이지로 나뉜다.

사용자 페이지에서는 비교적 적은 수의 Input을 블록 단위로 입력하거나, 많은 수의 Input들을 Excel 파일 형태로 입력할 수 있다. 입력이 완료되면 데이터들은 서버로 전송되어 Fine-Tuning 모델을 거쳐 결과를 도출하며, 그 결과를 DB에 저장하고 필요한 데이터는 다시 클라이언트에게 전송한다. 클라이언트에서는 각 영화평에 대한 분석 결과, 확률 등을 확인할 수 있으며, 전체 입력들에 대한 긍정/부정 및 정확도(정답률) 또한 그래프를 통해 시각적으로 확인할 수 있다.

관리자 페이지에서는 DB를 바탕으로 모델 버전 별 분석 결과들에 대한 통계적 자료들을 표와 그래프를 통해 볼 수 있으며, 현재 적용시키거나 Fine-Tuning할 모델 버전을 선탁하고, 실제로 Fine-Tuning하는 기능을 제어할 수 있다.

참고적으로 Fine-Tuning을 수행하지 않을 때의 Input들은 서버에 탑재된 모델의 가중치에 실시간으로 영향을 미치지 않고, 관리자가 원할 때 DB에 쌓여 있는 데이터들을 통해 Fine-Tuning을 진행할 수 있는 구조이다.

2.1.1. System Interfaces

Web Application:

클라이언트 인터페이스로부터의 요청을 받아 DB에 접근 또는 모델을 작동시키거나 정보들을 화면에 출력하도록 클라이언트에게 요청하는 등 중간 단계에서의 데이터 처리 전반을 담당한다.

Model:

Fine-Tuning, Processing을 담당하는 핵심 언어 모델로, 모델 서버에서 버전 관리가 이루어진다. 서버의 Web Application과 연동되어 작동한다.

2.1.2. User Interfaces

User Page:

메인 페이지에서 영화평을 입력 및 제출할 수 있으며, 결과 페이지에서 입력 데이터에 대한 통계적 결과를 출력한다. 서버의 Web Application과 통신한다.

Admin Page:

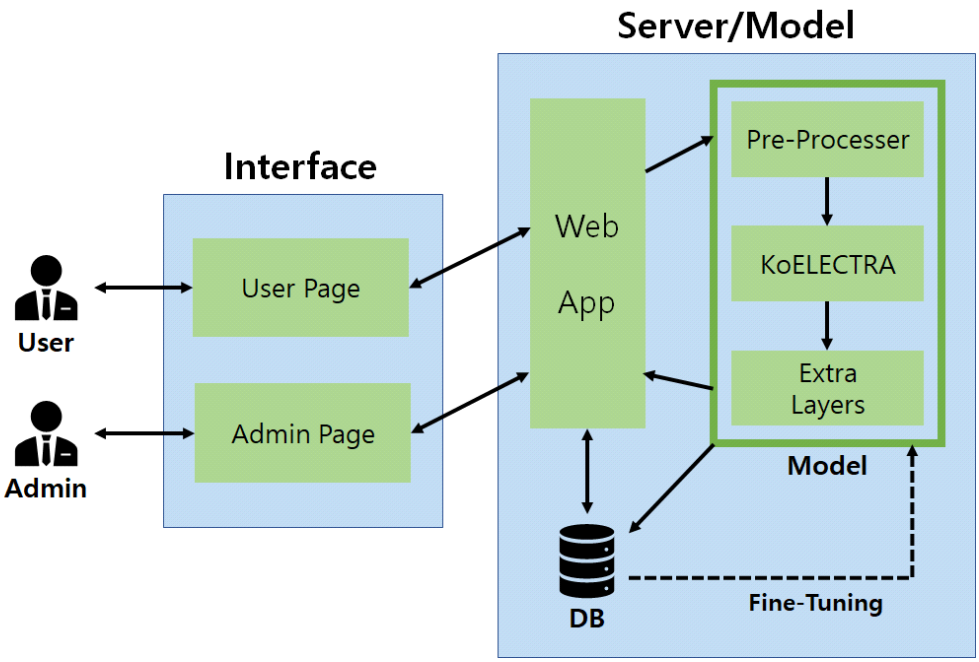
관리자 페이지에서 모델 버전 별 통계 자료를 확인하거나, 적용시킬 모델 버전을 선택할 수 있고 Fine-Tuning 요청을 할 수 있다. 사용자 페이지와 마찬가지로 서버의 Web Application과 통신한다.

2.1.3. Software Interfaces

사용자 관점에서 웹 페이지 접근이 가능한 환경이라면 별도의 제약 사항 없음.

2.2. Product Functions

2.2.1. Interface



2.2.1.1. Input

사용자가 글자 수 1000자 이하의 영화평 및 긍정/부정 입력을 할 수 있다.

2.2.1.2. Batch Input

Excel 파일을 통해 여러 개의 Input을 한번에 처리할 수 있다.

2.2.1.3. Submit

완료 버튼을 눌러 입력을 제출한다.

2.2.1.4. Display Output

분석 결과(표, 그래프)가 웹 인터페이스를 통해 출력된다.

2.2.1.5. Save Result

분석을 진행하고 사용자의 의도, 결과 등을 포함한 데이터를 DB에 저장한다.

2.2.1.6. Check Statistics

관리자 페이지에서 분석 결과에 대한 모델 버전 별 통계를 확인할 수 있다.

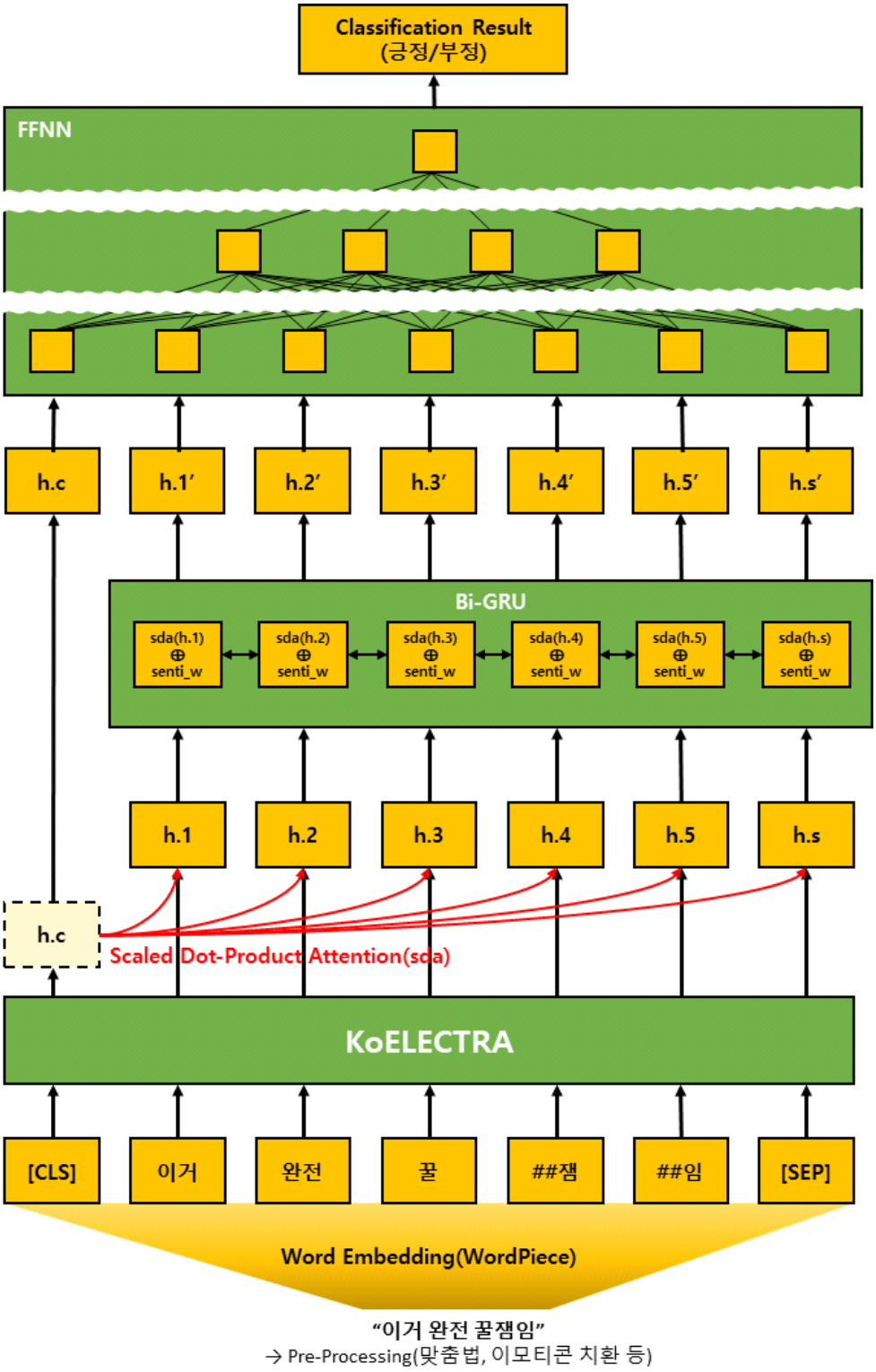
2.2.1.7. Apply Version

관리자 페이지에서 서비스에 적용할 모델 버전을 선택할 수 있다.

2.2.1.8. Activate Fine-Tuning

현재 선택된 모델 버전을 바탕으로 누적 DB 데이터를 통해 Fine-Tuning을 진행한다. 또한 결과 가중치가 반영된 새로운 모델 버전을 생성한다.

2.2.2. Model



- Scaled Dot-Product Attention: 문장 전체의 긍정/부정 속성을 각 단어 토큰에 보정해주는 효과
  → 또는 h.c 벡터를 argmax()한 긍정/부정 속성을 192차원(25%)의 random initialized vector로 변환하여 각 단어 벡터 뒤에 concat하는 방법으로 대체하거나 동시에 적용
- senti_w: sentiment weight, 한국어감성사전 적용 가중치(선처리)

2.2.2.1. Pre-Processing

입력에 대해서 오타, 띄어쓰기, 이모티콘에 대한 전처리가 모델 분석에 최적화된 포맷으로 정상적으로 이루어져야 한다.

2.2.2.2. Word-Embedding

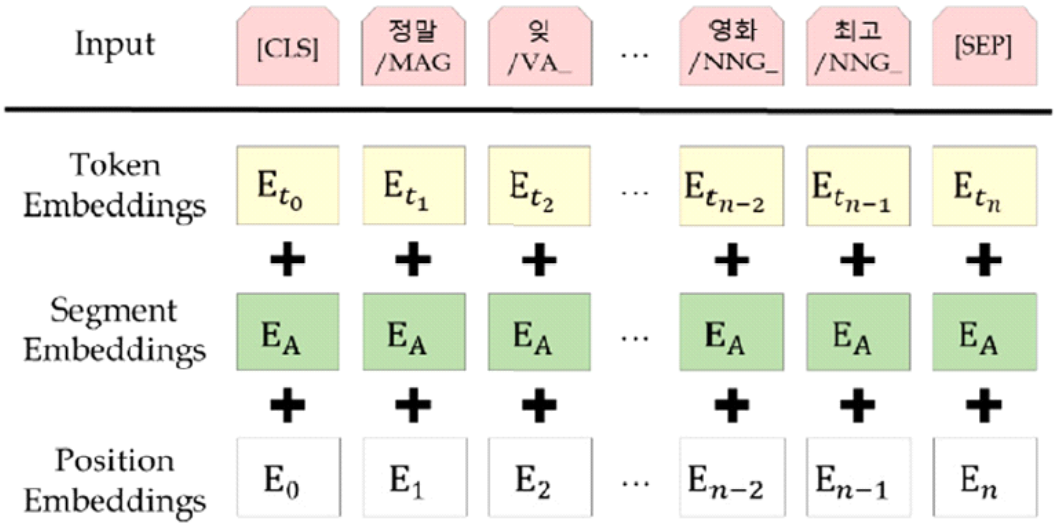
전처리된 입력에 [CLS], [SEP] 토큰을 부착시키고 각 토큰들을 아래 3가지 방식으로 임베딩(벡터화)한다.

1) Token Embedding: 각 토큰을 Vocabulary의 Index에 대응되는 Vector로 변환

2) Segment Embedding: 문장 구분을 위한 벡터, 하나의 문장인 경우 모두 0 할당

3) Position Embedding: 토큰의 위치를 의미하는 벡터, [CLS] 토큰에 0이 할당

이후 모든 임베딩 벡터들에 대한 Element-Wise-Sum을 수행하여 KoELECTRA의 최종 입력 벡터를 만든다.



2.2.2.3. KoELECTRA Layer

해당 Layer를 거침으로써 각 토큰들은 768차원의 Hidden State(벡터)로 변환되어 출력된다. [CLS] 토큰은 문장 전체에 대한 문맥 정보를 내포하고 있으며, 그 외의 모든 단어 벡터들은 Vocabulary 내 단어의 의미에 문맥 정보가 반영된 벡터가 된다.

2.2.2.4. Scaled Dot-Product Attention(SDA)

한국어감성사전만을 적용하면 일반적인 용례와 영화평에서의 용례에서 단어의 긍정/부정이 상반되는 경우(예: '울다'는 부정적인 단어이나 영화평에서는 영화를 통해 감동을 받았을 때, 즉 긍정적으로 쓰이는 경우가 훨씬 많다)를 처리할 수 없으므로 이에 대한 보정을 위해 문맥 전체의 의미를 내포한 CLS 토큰의 Hidden State을 각 단어 Hidden State에 Scaled Dot-Product Attention 함으로써 문장 전체가 긍정인 경우 긍정 결과에 더 큰 영향을 주는 토큰의 중요도를 높게, 나쁜 영향을 주는 토큰의 중요도를 낮게 부여한다.

2.2.2.5. KnuSentiLex Weight Concatenation

한국어 문장(1-gram to 8-gram)들에 대한 감성정보(긍정/부정)를 매우 부정(-2), 부정(-1), 중립(0), 긍정(1), 매우 긍정(2) 5단계로 분석하여 정리한 한국어감성사전을 활용하여 각 단어 토큰들에 가중치를 부여한다. 이 때 토큰과 사전의 Unit이 다르기 때문에 여러 단어 토큰에 하나의 감성정보가 대응되는 경우가 존재할 수 있으며, 이 경우 가중치 벡터의 각 요소값을 토큰 개수로 나누어 부여하는 아이디어를 적용한다.

가중치 벡터는 Scaled Dot-Product Attention이 완료된 벡터에 Concatenation이 이루어진다.

2.2.2.6. Bi-GRU Layer

SDA와 한국어감성사전 가중치 적용이 끝난 토큰들을 Many-to-One RNN으로 쓰여 Sentiment Classification에 주로 사용되는 GRU(Gated recurrent unit)이 양방향으로 적용되는 Bi-GRU를 통과시킴으로써 KoELECTA의 CLS와 무관하게 문장 자체로써의 Classification을 학습시킨다. 모든 출력 벡터를 활용하며, 이후 FFNN을 원활히 통과시키기 위해 Projection Layer를 얹어 차원을 일치시킨다.

2.2.2.7. FFNN Layer

최초 KoELECTRA에서 도출한 CLS 토큰 벡터와, Bi-GRU로 별도 학습한 n개의 벡터

를 FFNN의 입력으로 넣어 최종적으로 2차원의 한 벡터로 변환한다. 최종 출력은 이 벡터를 Sigmoid 이후 Argmax하여 긍정/부정으로 나타낼 수 있도록 한다.

2.3. User Characteristics

머신 러닝에 대한 지식이 없어도 상관없는, 한국어를 주 언어로 영화평을 작성할 수 있는 일반적인 사용자.

2.4. Assumptions and Dependencies

전체 모델이 한국어 영화평에 최적화되어 있으므로 최대한 한국어로 표현되고 맞춤법에 맞는 영화평이 입력으로 들어온다고 가정한다(해당 기준을 준수하지 않으면 전처리를 하더라도 분석의 정확도가 떨어질 수 있음).

Vocabulary에 없는 단어는 모두 [UNK]으로 간주한다.

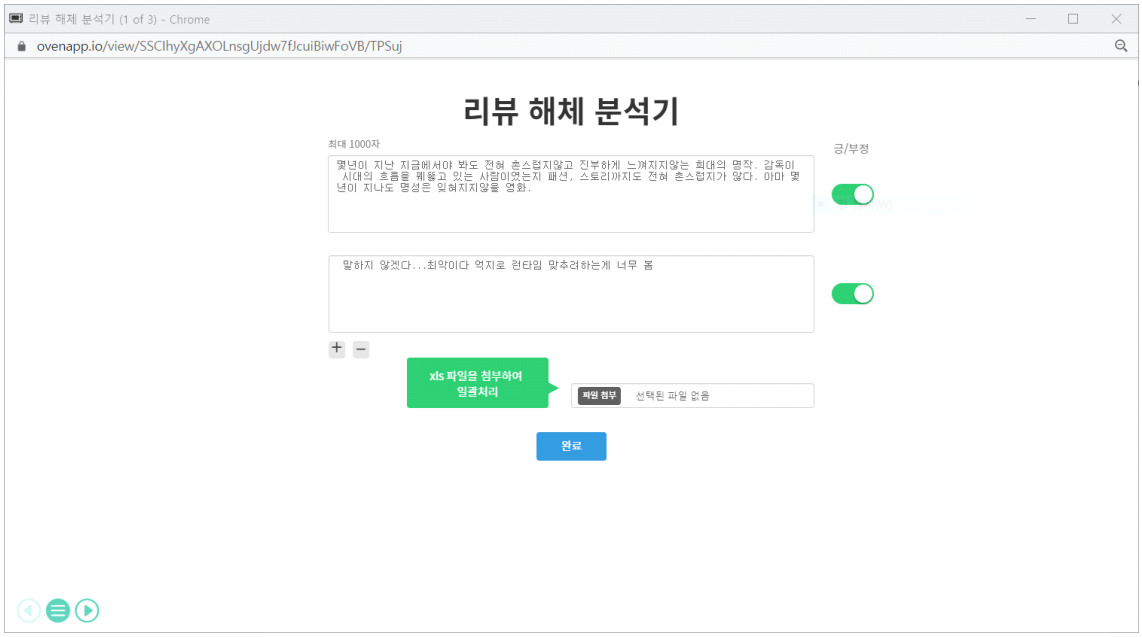
설정하지 않은 모든 테스트 입력은 모델의 학습 가중치에 영향을 주지 않는다.

3. Specific Requirements

3.1. External Interface

3.1.1. User Interface

3.1.1.1. 사용자 메인 페이지



1) 직접 입력 시

- 텍스트 입력 상자에 최대 1000자의 Input을 입력하고 우측에 해당 텍스트에 대한 긍/부정 의도를 선택할수 있다.

- 하단 + 버튼을 눌러 텍스트 입력 상자를 추가할 수 있다

- 하단 - 버튼을 눌러 텍스트 입력 상자를 삭제할 수 있다. - 버튼을 누를 시 삭제하려는 텍스트 입력 상자를 Radio 버튼으로 다수 선택할 수 있다.

2) 일괄 입력 시

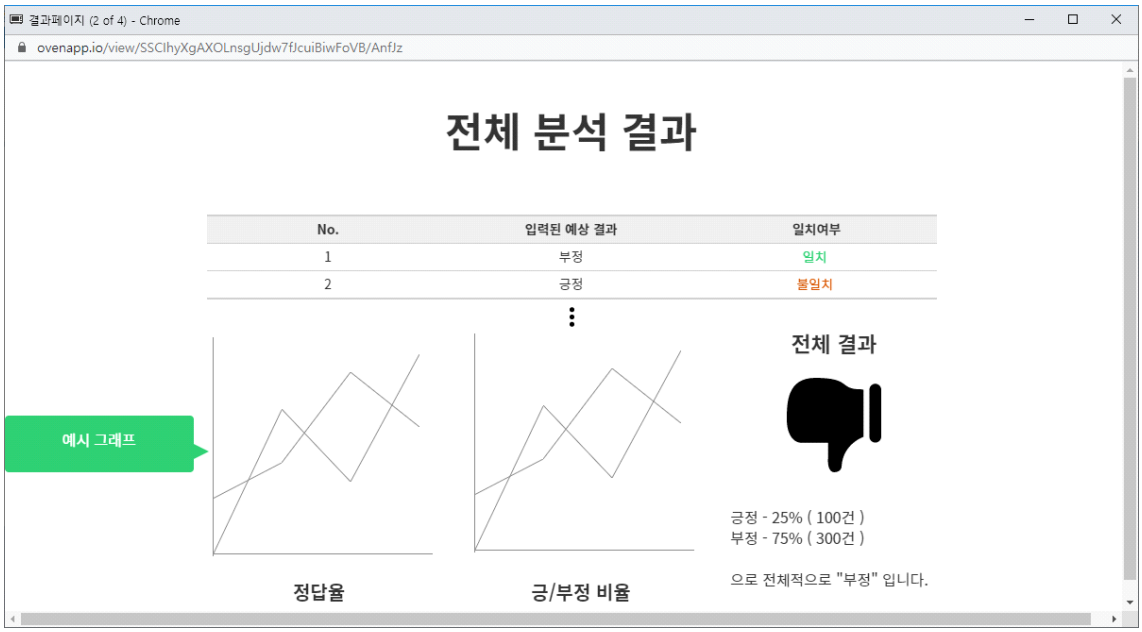
- 파일 첨부 버튼을 눌러 xls 파일을 업로드 할 수 있다.

- xls 파일은 첫 열에 입력하려는 텍스트, 두번째 열에 긍/부정을 1/0 으로 작성하여 업로드 하여야 한다.

3) 완료 버튼

- 완료 버튼을 누르면 서버로 텍스트, xls 파일이 전송되어 자연어 처리가 진행된다.

3.1.1.2. 사용자 결과 페이지



1) 결과 표

- 첫 열은 입력된 텍스트 순서에 따른 번호를 의미한다.

- 두번째 열은 사용자가 입력한 예상 결과 (긍/부정) 이다.

- 세번째 열은 학습 모델이 처리하여 도출해낸 결과와 일치하는지 여부이다.

- 사용자가 입력한 예상 결과와 학습 모델이 도출한 결과가 같을 시 세번째 열의 텍스트가 초록색으로 출력되고, 결과가 다를시 텍스트가 빨간색으로 출력된다.

- 분석 결과가 10개 이상일 시 스크롤 이동이 추가 된다.

2) 정답률 그래프

- 입력된 Input을 통계 내어 입력된 예상 결과와 학습 모델의 결과가 맞는지의 정답률을 그래프로 보여준다

3) 긍정/부정 비율 그래프

- 입력된 Input을 통계 내어 학습 모델의 결과중 긍/부정 비율을 그래프로 보여준

다

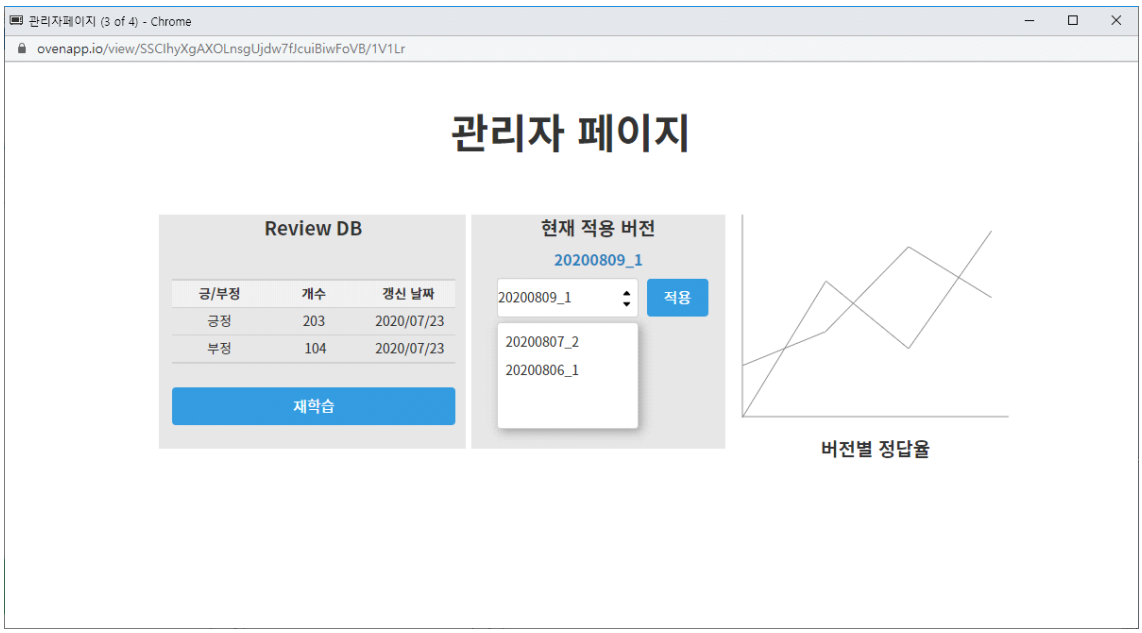
4) 전체 결과

- 전체 입력에 대해 학습 모델이 도출해낸 긍/부정중 더 많은 것을 전체 결과로 하여 엄지 이미지("긍정"일시 위, "부정"일시 아래 를 향함) 를 출력한다.

- 아래에 전체 결과중 "긍정" 인 결과의 비율과 그 개수를 출력한다

- 아래에 전체 결과중 "부정" 인 결과의 비율과 그 개수를 출력한다

3.1.1.3. 관리자 페이지



1) Review DB

- 현재 까지 시스템에 입력된 Data를 토대로 긍/부정의 개수와 제일 최근에 Update 된 날짜를 출력한다.

- "재학습" 버튼을 클릭 시 현재 까지 입력된 텍스트 들을 Training Set에 추가하여 재학습을 진행한다. 재학습을 완료한 모델은 버전별로 DB에 저장이 된다.

2) 적용 버전

- DB에 저장된 학습 모델 중 하나를 현재 감성 분석에 적용할 수 있다.

- 현재 적용된 버전을 텍스트로 출력해준다.

- Dropbox로 DB에 저장된 버전중 하나를 선택하여 적용할 수 있다.

2) 버전별 정답률

- 각 버전별 정답률을 그래프로 출력한다.

3.1.2. SW Interface

3.1.2.1. Model

Name	Python
Version	3.7.6
Purpose	머신 러닝에서 주류로 사용되는 언어로 플랫폼에 독립적이며 인터프리터식, 객체지향적, 동적 타이핑 대화형 언어
Source	www.python.org

Name	Pytorch
Version	1.6.0
Purpose	기계학습과 관련한 기본적인 자료구조와 함수를 제공하는 기계학습 프레임워크
Source	https://pytorch.org/ https://github.com/pytorch/pytorch

Name	Transformers
Version	3.0.2
Purpose	Pytorch와 Tensorflow로 미리 작성된 모델을 쉽게 관리할 수 있게 하는 라이브러리. 본 프로젝트에서 HuggingFace에 등록된 KoELECTRA-small 모델을 사용하는데 활용
Source	https://github.com/huggingface/transformers

3.1.2.2. Backend(Server)

Name	MySQL
Version	8.0
Purpose	오픈 소스 관계형 데이터베이스, 다중 스레드, 다중 사용자 형식의 구조질의어 형식의 데이터베이스 관리 시스템
Source	https://www.mysql.com/

Name	Flask
Version	2.0
Purpose	파이썬 웹 프레임워크
Source	https://flask.palletsprojects.com

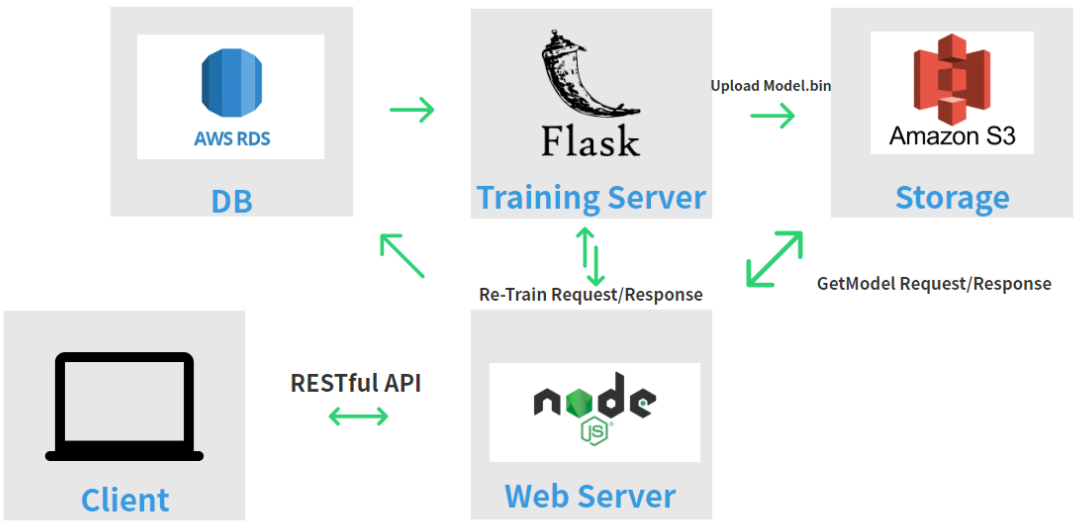
Name	aws-sdk
Version	2.0
Purpose	aws 애플리케이션 개발 및 서비를 위한 Javascript API
Source	https://aws.amazon.com/ko/tools/

Name	Express.js
Version	4.17.1
Purpose	Node.js 웹 애플리케이션 서버 프레임워크
Source	https://expressjs.com/ko/

3.1.2.3. Frontend(Client)

Name	React
Version	16.13.1
Purpose	자바스크립트기반 웹 프레임워크
Source	https://github.com/facebook/react

3.1.3. Communications Interfaces



3.2. Functional Requirements

3.2.1. Interface

3.2.1.1. User Page

(1) 사용자가 Data(200자 이하의 영화평 문장 및 긍정/부정 여부)를 입력할 수 있어야 함.

(2) '+, -' 버튼 클릭을 통해 추가 Data 입력 또는 삭제가 가능해야 함.

(3) 작성된 입력들을 제출할 수 있어야 함.

(4) 여러 개의 Data를 한번에 제출할 수 있도록 Excel 파일 연동 입력 기능이 제공되어야 함.

[결과 페이지]

(5) 입력을 바탕으로 각 Data에 대한 '긍정/부정 예측', '일치 여부' 리스트와 Data Set에 대한 '정답률(일치율)', '긍정/부정 비율' 그래프, 전체적인 긍정/부정 여부에 대한 결과가 출력되어야 함.

3.2.1.2. Admin Page

[메인 페이지]

(1) 관리자 페이지에 모델 버전 별 '정확도' 가 표 및 그래프로 출력되어야 함.

(2) 관리자가 활용 또는 Fine-Tuning할 모델 버전을 선택할 수 있어야 함.

(3) 현재 선택된 모델을 기반으로 누적 Data Set을 통해 Fine-Tunning하여 새로운 버전을 생성할 수 있어야 함.

3.2.2. Language Model

3.2.2.1. Pre-Processing

(1) Raw-Data의 오타, 띄어쓰기, 이모티콘 등을 작업에 최적화된 포맷으로 변환해야 함.

(2) 변환한 Data를 KoELECTRA의 Vocabulary에 대응시켜 Word-Embedding해야 함.

3.2.2.2. RNN Layer

(1) KoELECTRA의 Input Sentence에 한국어 감성사전을 적용하여 각 Position에 맞는 가중치 벡터(senti_w)들을 구성해야 함.

(2) 한국어 감성사전 적용에 관한 오차 보정을 위해 [CLS] 토큰의 Hidden State를 나머지 토큰들에 Scaled Dot-Product Attention(SDA)해 주어야 함.

(3) Position이 일치하는 'SDA를 마친 벡터'와 '가중치 벡터'들을 Contatenate하여 Bi-GRU를 통과시키고, 문맥 흐름 정보를 담고 있는 레이어의 출력 벡터를 FFNN Layer 입력 차원에 맞춰 Projection 해야 함.

3.2.2.3. FFNN Layer

(1) KoELECTRA의 [CLS] 토큰에 대한 Hidden State와 RNN Layer에서 나온 두 벡터를 FFNN에 통과시켜 2차원 Classification 벡터로 변환하고, 이를 바탕으로 긍정/부정 예측 결과를 출력해야 한다.

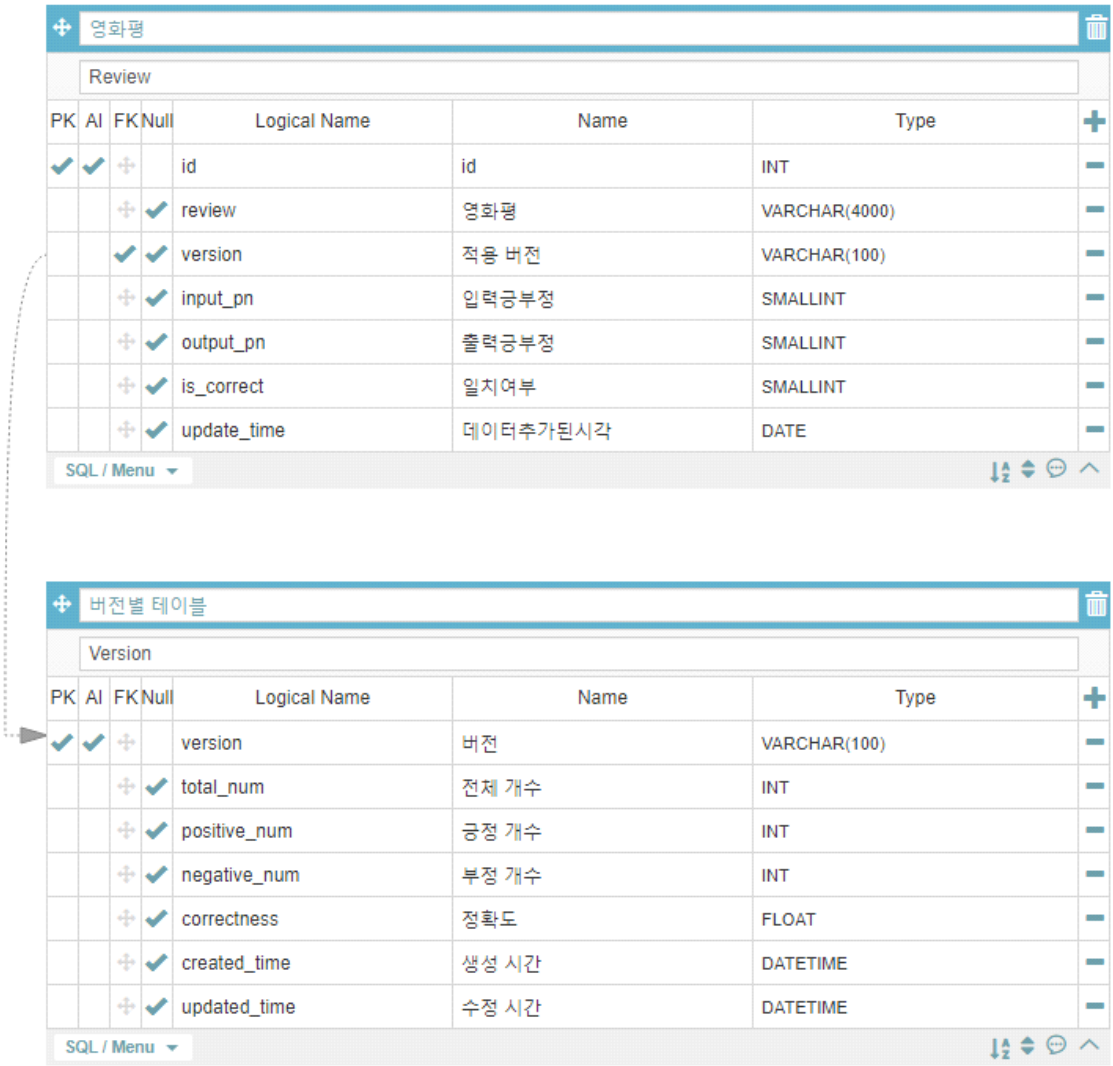
3.3. Performance Requirements

1. 인터페이스의 명료성과 직관성에 대해 60% 이상의 사용자가 만족해야 함.

2. 입력 제출 후 최종 결과가 출력될 때까지 Input 개수 당 2초 이하의 시간이 소요되어야 함.

3. 표본이 50개 이상일 때 모델로부터 도출된 평가와 사용자의 의도의 일치도가 90% 이상이어야 함.

3.4. Logical Database Requirements



영화평

Review

PK	AI	FK	Null	Logical Name	Name	Type
✓	✓			id	id	INT
			✓	review	영화평	VARCHAR(4000)
		✓	✓	version	적용 버전	VARCHAR(100)
			✓	input_pn	입력긍부정	SMALLINT
			✓	output_pn	출력긍부정	SMALLINT
			✓	is_correct	일치여부	SMALLINT
			✓	update_time	데이터추가된시각	DATE

SQL / Menu

버전별 테이블

Version

PK	AI	FK	Null	Logical Name	Name	Type
✓	✓			version	버전	VARCHAR(100)
			✓	total_num	전체 개수	INT
			✓	positive_num	긍정 개수	INT
			✓	negative_num	부정 개수	INT
			✓	correctness	정확도	FLOAT
			✓	created_time	생성 시간	DATETIME
			✓	updated_time	수정 시간	DATETIME

SQL / Menu

3.5. Design Constraints

3.5.1. 모델 Fine-Tuning에 필요한 GPU, TPU 자원과 서버

해당 제품이 제공하는 가장 중요한 서비스는 서버의 모델로부터 나온다. 모델은 웹 인터페이스의 입력으로 들어온 데이터를 통해 저장한 데이터베이스로부터 읽어 들여 Classification을 학습하는 Fine-Tuning을 주기적으로 실행할 수 있어야 한다.

서버에서 사용할 기초적인 Fine-Tuning 과정은 Google Colab에서 제공하는 GPU와 Google Cloud Platform에서 제공하는 TPU를 사용한다. 두 서비스는 무료이기 때문

에 제한적인 사용률과 제한시간을 가지고 있어, 클라이언트 서비스를 제공하는 동안에는 서버에서 성능이 낮은 CPU 자원을 이용해야 한다. 따라서 Fine-Tuning 시 CPU가 처리할 수 있는, 소량의 데이터 Batch만 수용 가능하다.

3.5.2. 웹 서비스를 제공할 서버 환경

본 서비스를 제공하는 웹 서버는 클라이언트의 요청을 처리하고, 내부적으로는 통계 결과와 모델 Fine-Tuning을 제공하기 위한 데이터베이스 instruction을 실행해야 한다. 이를 동시다발적으로 처리하기 위해 각각을 처리하는 서버를 다수 두거나, 자원이 충분한 서버가 필요하다.

3.5.3. 데이터 규모를 고려한 서버 용량과 DBMS 선택

웹 서비스가 지속됨에 따라 저장되는 유저 입력 데이터의 양과 Fine-Tuning 결과로 생성되는 모델 결과물들을 충분히 저장할 수 있는 서버 용량이 필요하다. 그에 따라 서버 용량의 증량이 필요하거나 데이터의 삭제를 고려해야 한다. 즉, 이를 처리할 수 있는 적절한 DBMS 선택이 필요하다.

3.5.4. Input 및 Fine-Tuning Data의 Scalability

자원이 제한된 환경에서 서비스가 제공되므로 Input 데이터와 Fine-Tuning Batch Size에 제한이 필요하다.

- 한 번에 받아들일 수 있는 Input 개수는 100개 이하
- Fine-Tuning 시 활용되는 Test Set의 최대 크기는 1000개 이하

3.6. Software System Attribute

1) 성능(Performance)

- 클라이언트의 요청을 2sec/Input 안에 처리해야 함
- 최대 5명의 요청을 동시에 처리할 수 있어야 함, 초과되는 인원은 queue를 통한 대기열로 관리
- 상기 Input 및 Fine-Tuning Data의 제약 사이즈 이하에 대한 처리가 원활해야 함

- 주기적으로 이루어지는 Fine-Tuning 과정이 서버가 제공하는 서비스에 영향을 주지 않아야 함

- 서버에서 처리한 클라이언트 입력에 대한 반응이 2초 이내에 웹페이지에 새로 반영되어야 함

2) 신뢰성(Reliability)

해당 문서에 기재된 제약 사항을 모두 준수했을 때 모든 서비스가 정상적으로 제공됨

3) 가용성(Availability)

기본적으로 여러 개의 요청들을 병렬적으로 처리할 수 있어야 함

서버의 복구, 업데이트를 위해 2개 이상의 서버(백업 서버 포함)를 사용함

4) 유지 보수성(Maintainability)

모델, 데이터베이스, 웹 인터페이스와에 대해 각 기능별로 관련된 모듈을 명확히 구분해 설계 및 관리

3.7. Other Requirements

1) 인터페이스가 사용자 관점에서 명료하고 직관적이어야 함.