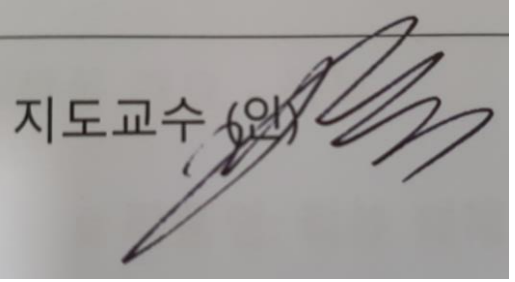


리뷰 해체 분석기

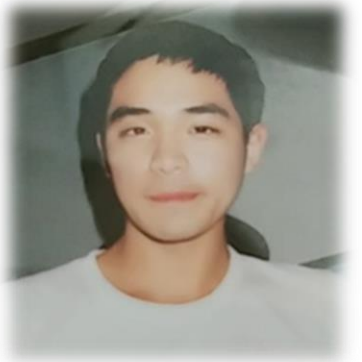
[딥러닝 기반 영화평 감성 분석]

2nd Implementation

Team #4

201411273 박재범

201411275 박진호



201411283 이상민

201511244 김민우

 # 작품명: 리뷰 해체 분석기

 # 소프트웨어의 목적:

한국어로 작성된 영화평을 입력받아 해당 영화평의 긍정/부정 여부를 판단하고 통계를 확인 및 관리 가능한 웹 어플리케이션을 서비스한다. 또한 신규 데이터셋에 대해서 추가적인 학습을 진행함으로써 새로운 버전을 생성하고 선택하여 적용할 수 있으며 버전별 통계도 함께 제공하여 버전 관리와 모델 정확도 향상이 가능하고 이를 통해 지속적으로 서비스의 질을 개선할 수 있도록 한다.

해당 소프트웨어를 통해 1차적으로는 영화 산업에서 관람객 리뷰 분석을 통해 전반적인 긍/부정 비율을 파악해 상영관 확대, DVD/Blue-Ray 발매 등 영화 개봉 전후 사업 확장 또는 축소에 도움을 주는 등 다양한 분야의 시장 반응 분석에 활용할 수 있으며,

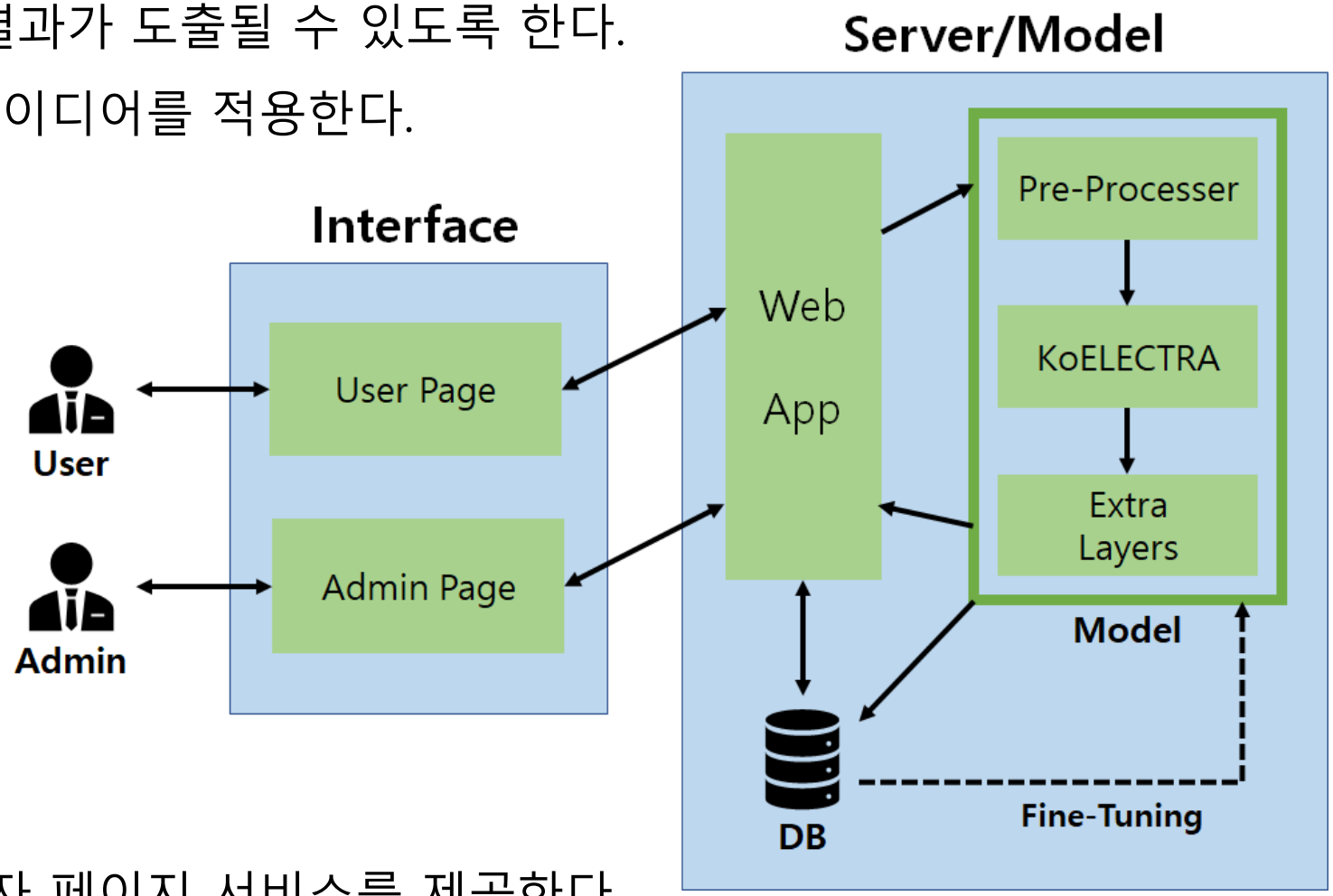
2차적으로는 한국어 감성 분석 정확도를 향상시킨 언어 모델을 얻을 수 있어 다른 분야에서도 유용하게 활용할 수 있도록 한다.

 # 언어 모델(Model)

전처리한 입력 문장에 대해 KoELECTRA 기반 인공신경망을 거쳐
긍정/부정의 Binary Classification 결과가 도출될 수 있도록 한다.
또한 정확도 향상을 위한 다양한 아이디어를 적용한다.

 # 웹 서버(Back-End)

웹 인터페이스를 통한 사용자 또는
관리자의 요청을 처리하거나
모델 서버와의 데이터 처리 및
DB 관리를 담당한다.



 # 웹 인터페이스(Front-End)

웹 환경에서 사용자 페이지와 관리자 페이지 서비스를 제공한다.

 # 언어 모델(Model)

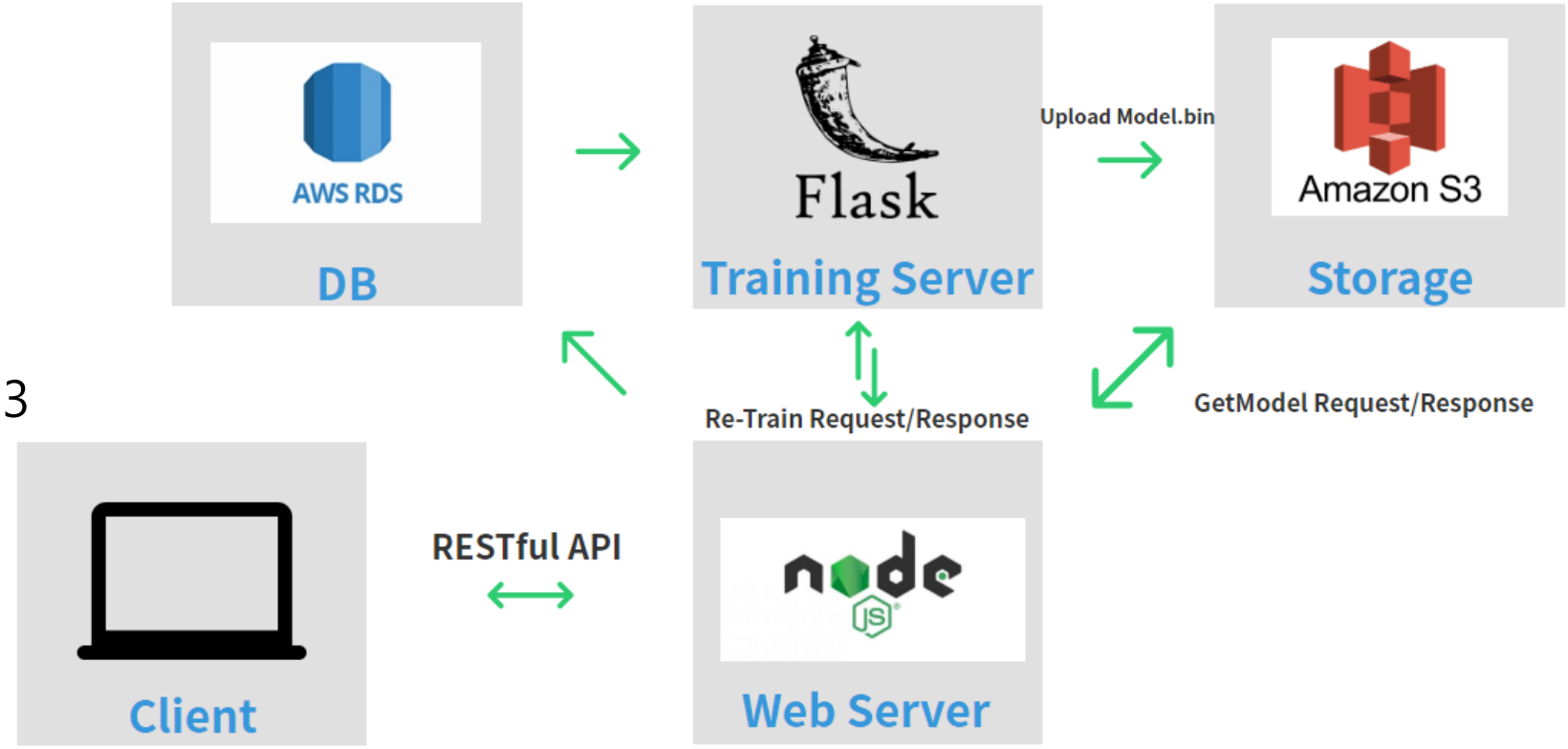
- Python 3.8.6
- Pytorch 1.6.0

 # 웹 서버(Back-End)

- Flask(Model Server) 2.0
- Node.js(App Server) 12.18.3
- Express.js 4.17.1
- MongoDB / AWS

 # 웹 인터페이스(Front-End)

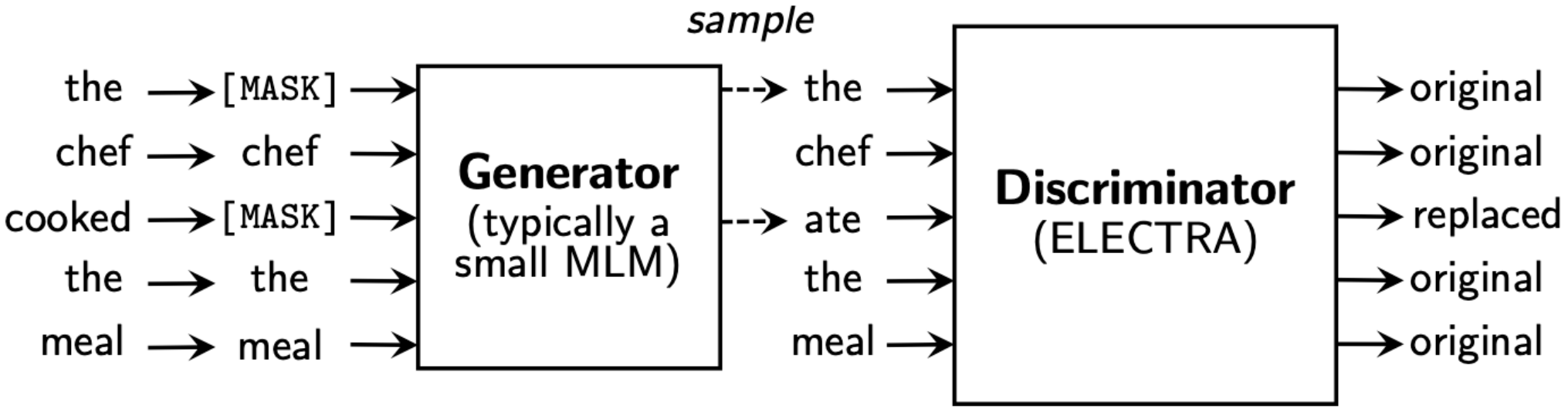
- React 16.13.1
- styled-components 5.1.1



 # Why KoELECTRA?

2020년 3월, Google Research 팀에서 발표한 Pre-Trained 언어 모델 ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately)를 한국어에 맞게 학습시킨 모델.

→ 기존 언어모델(BERT, RoBERTa, ALBERT 등)에 비해 높은 정확도



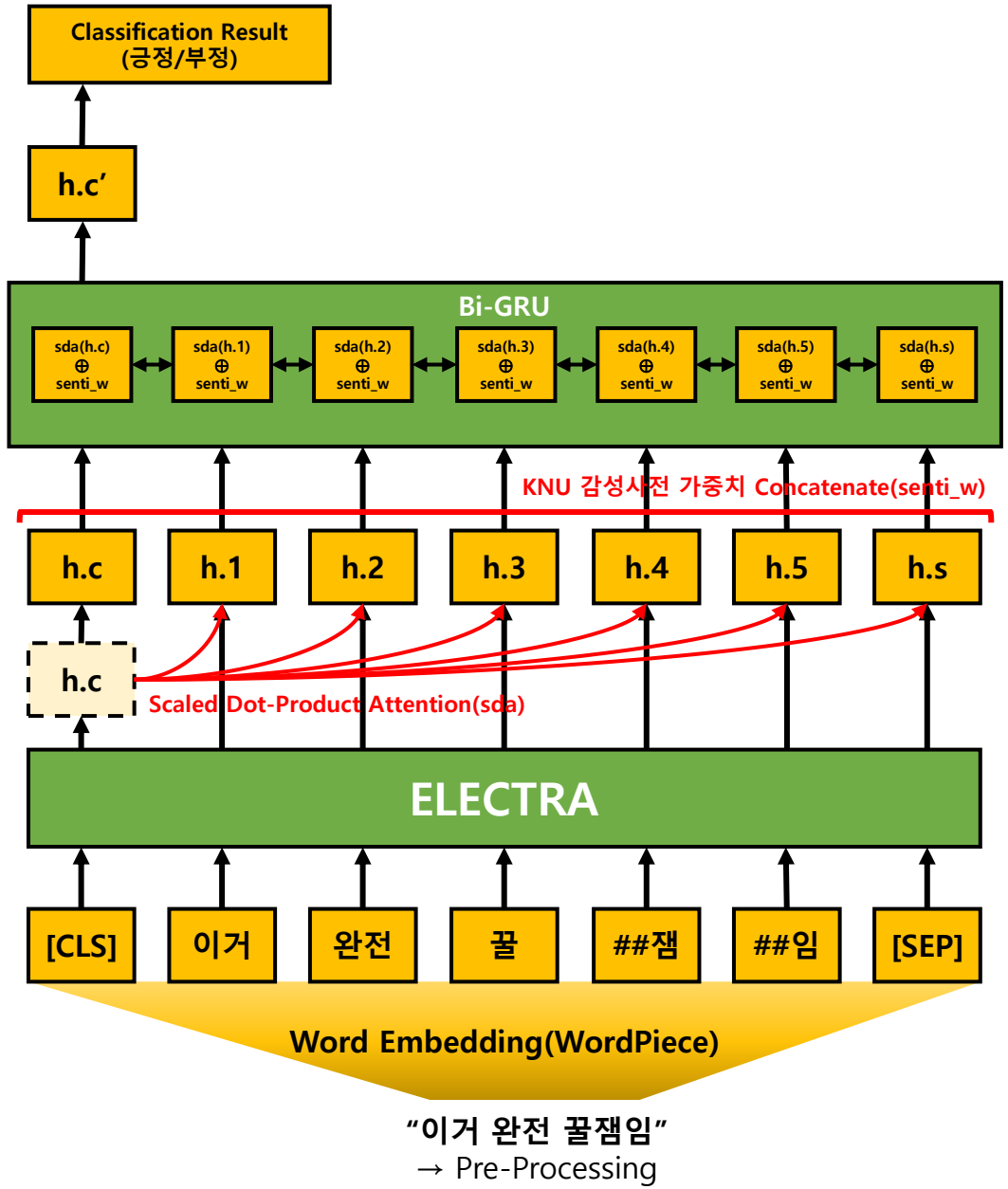
Source: github.com/monologg/KoELECTRA

 # 모델 학습(Fine-Tuning)

로컬 환경의 Google Colab에서 Fine-Tuning을 진행

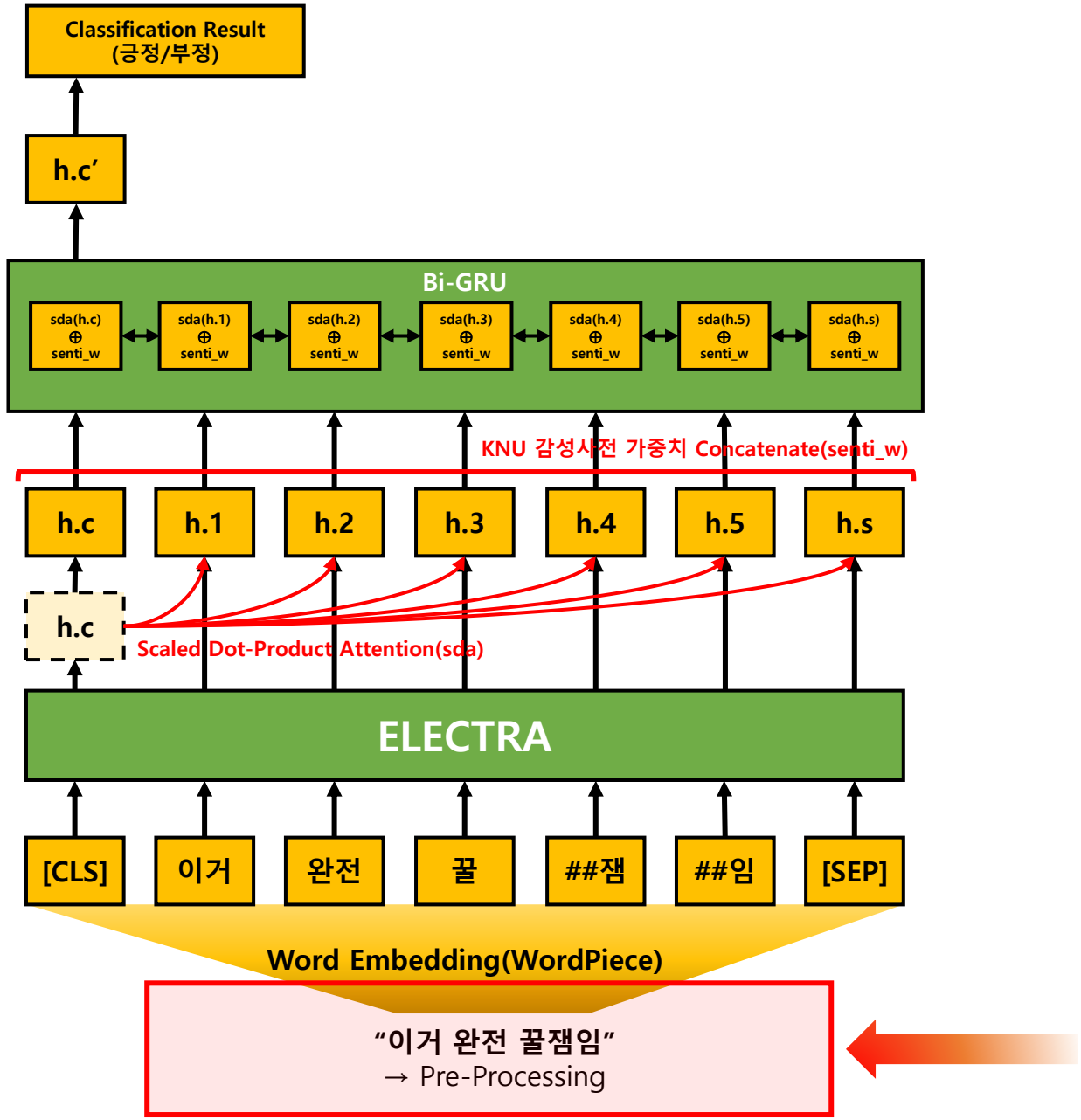
 # 최종 모델 구조 및 전체 프로세스

1) 입력 영화평을 전처리
2) Word Embedding 수행
3) ELECTRA 모델에 입력
4) 출력에서 CLS 벡터를 나머지 벡터에 Scaled Dot-Product Attention
5) KNU 한국어 감성사전 가중치 Concatenate
6) Bi-GRU에 입력
7) 좌측 결과 벡터를 활용하여 Classification



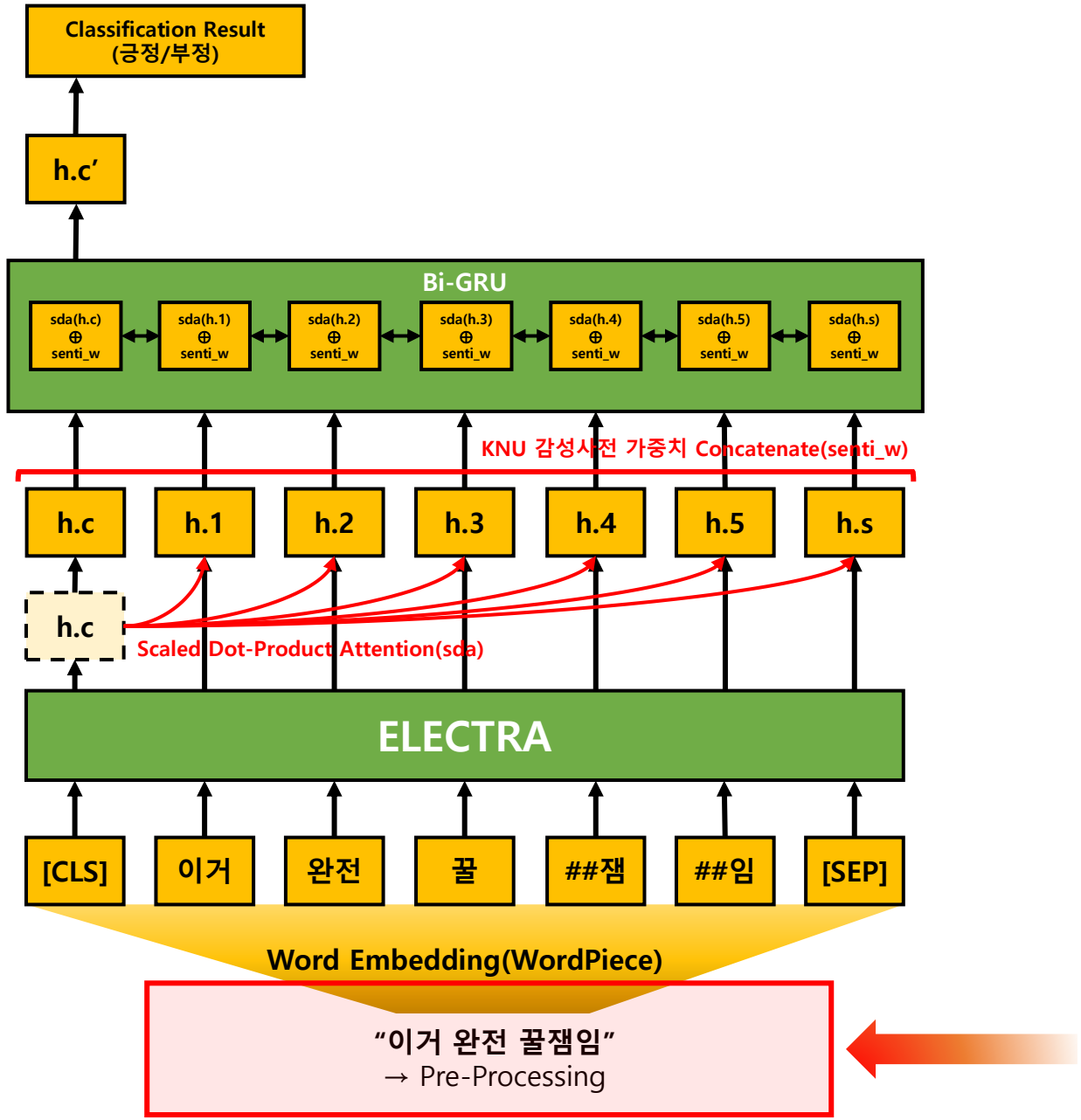
1) 학습 데이터 크롤링

네이버 영화 리뷰에서 크롤링을 진행,
특정 장르나 포맷에 치우치지 않도록 총 17개
(드라마, 판타지, 공포, 멜로/애정/로맨스, 모험,
스릴러, 느와르, 다큐멘터리, 코미디, 가족, 미스
터리, 전쟁, 애니메이션, 범죄, 뮤지컬, SF, 액션)
장르에 대해 대체적으로 관객들의 평가가
긍정적인 영화 10개, 부정적인 영화 10개를
직접 선별하여 총 340개의 영화에 대해
긍정(8~10점), 부정(1~3) 리뷰 비율이
1:1이 되도록 하였음. NSMC 데이터와 결합해
60만개의 Training Set, 20만개의 Test Set,
총 80만개의 데이터로 학습 및 평가를 진행함.



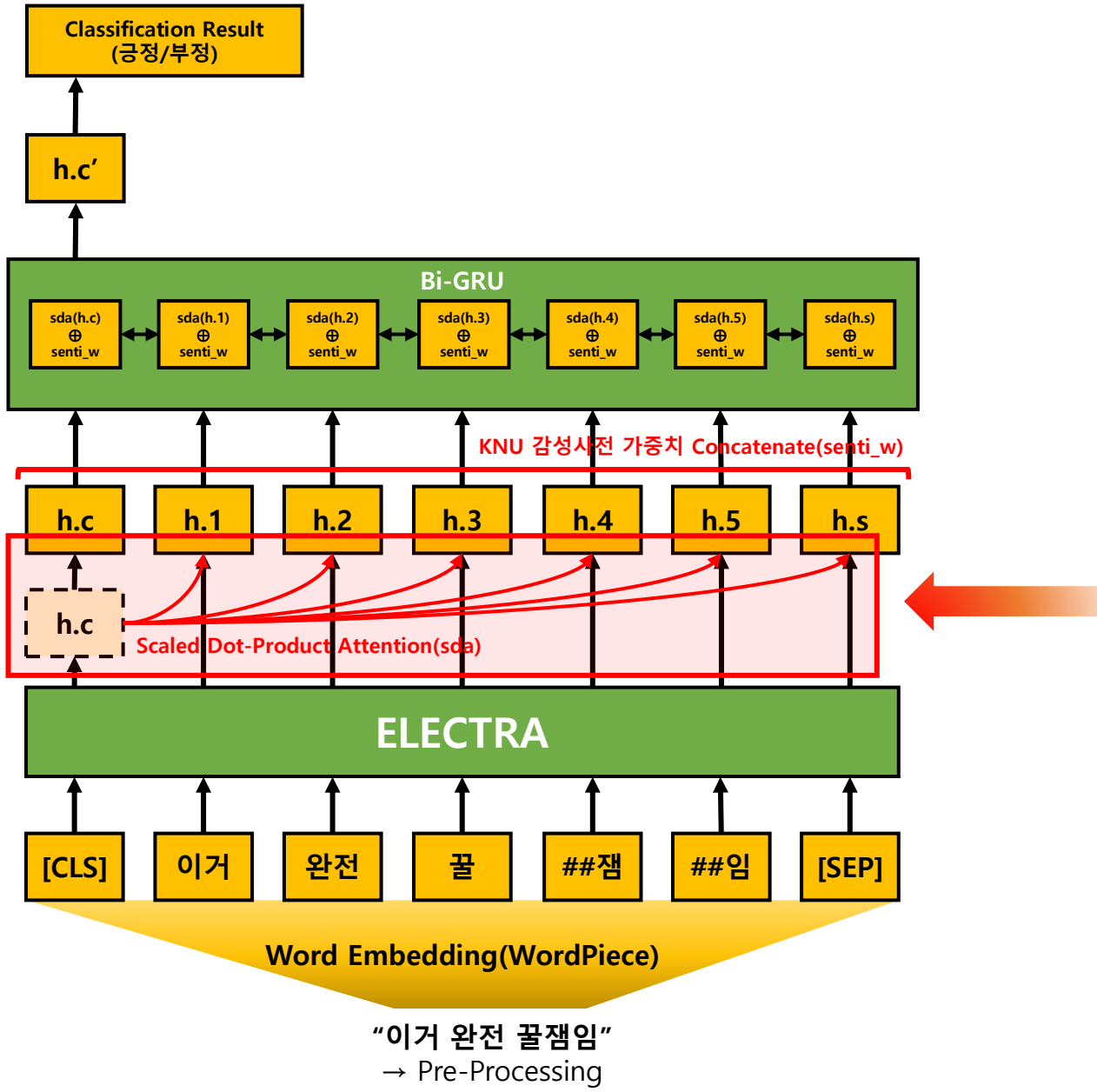
2) 데이터 전처리 작업

학습 시 정확도에 악영향을 끼칠 수 있는
요소들에 대한 전처리를 수행.
감성 정보에 영향을 미치지 않는
HTML 태그 기호나 한자 등을 제거하고,
한 문자가 3회 이상 반복되는 경우 2회로 축약.



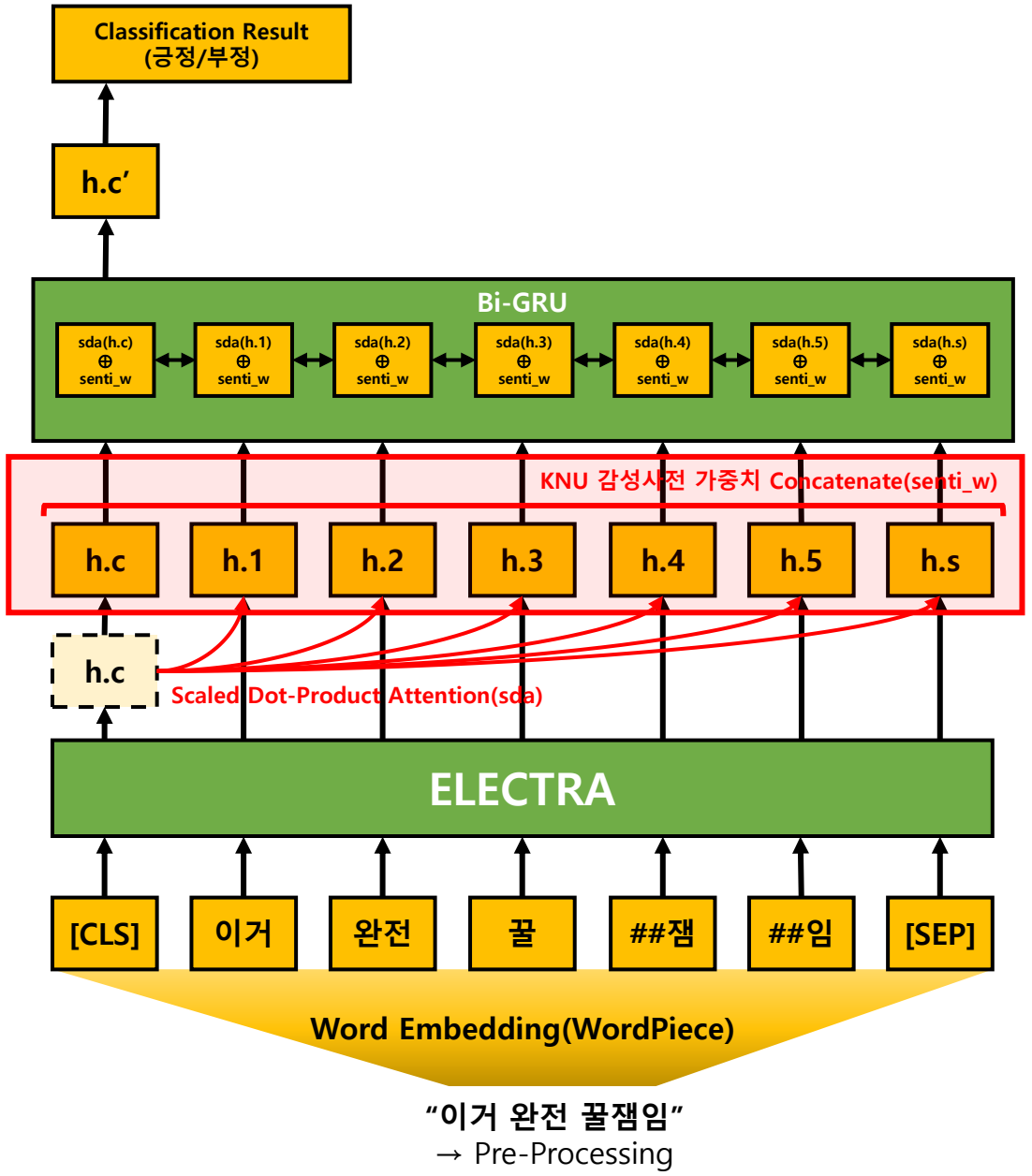
3) Scaled Dot-Product Attention

ELECTRA의 CLS 벡터를 나머지 벡터에 Attention하여 감성 분석에 영향을 많이 준 벡터일 수록 중요도를 높게, 적게 준 벡터일 수록 중요도를 낮게 차등 설정하는 효과를 줌.



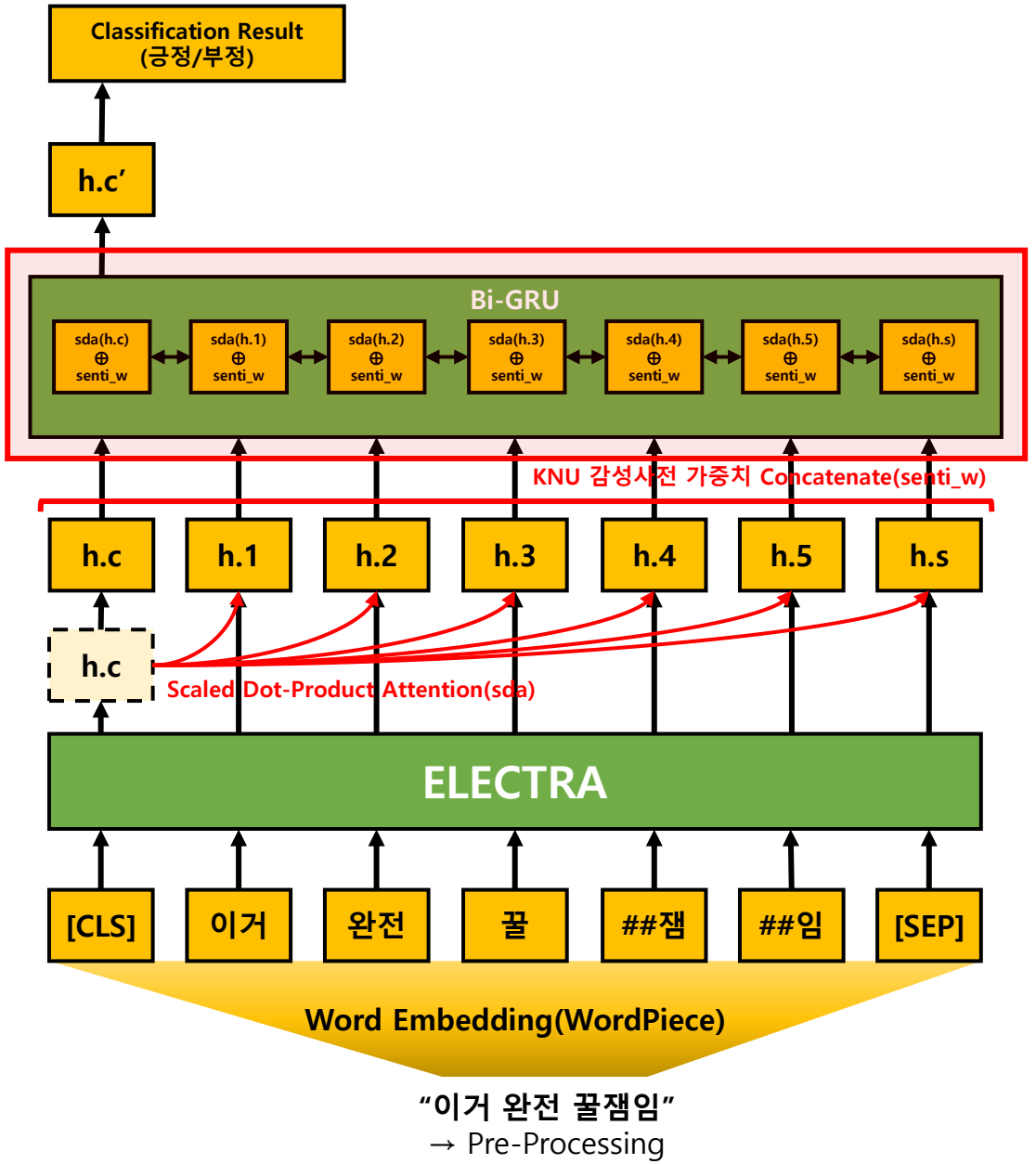
4) KNU 한국어 감성사전 가중치 적용

Attention을 마친 벡터들에
KNU 한국어 감성사전에 해당하는 가중치를
Concatenate하여 각 토큰의 감성 정보를 강조.
가중치는 96차원의 벡터이며, ELECTRA의
Vocabulary를 기반으로 감성사전의 말뭉치를
Tokenize하고, 각 토큰이 전체 말뭉치에서
-2부터 +2까지 5 단계의 가중치 중 어떤
감성값을 가진 말뭉치에 포함되었는지
빈도수를 계산하여, 빈도수의 평균값에 해당하
는 감성 값을 토큰과 1:1 맵핑하였음.
이 때, 감성 사전에서 사용되지 않은 토큰은
0(중립) 값을 부여함.



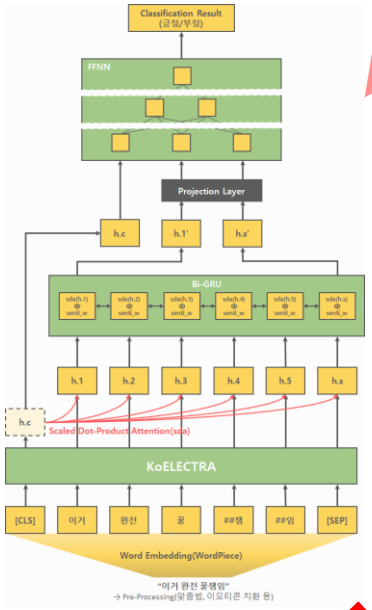
5) Bi-GRU

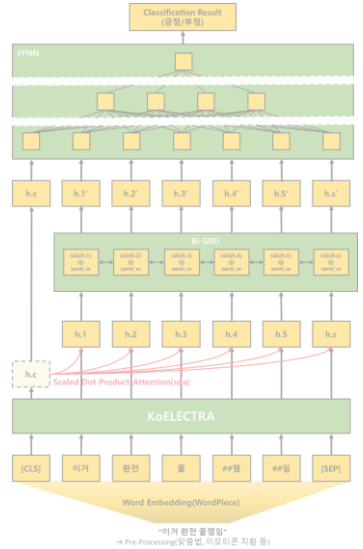
ELECTRA의 출력에 Scaled Dot-Product Attention을 거치고 감성사전 가중치를 부여한 토큰들을 RNN(Bi-GRU)에 통과시켜 재해석된 Classification을 유도하였고, LSTM에 비해 학습 속도가 빠른 GRU를 활용하였으며, 추후 성능 향상을 위한 유연한 구조 변경 및 활용을 위해 Bi-GRU를 적용.

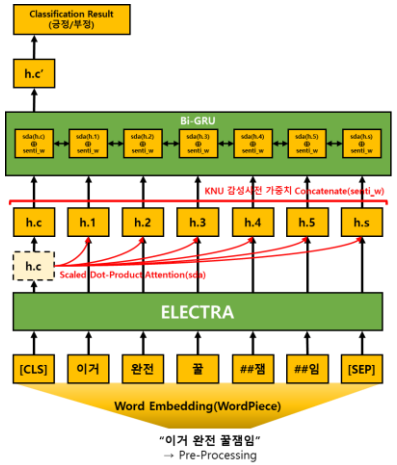


모델 버전(레이어 및 아이디어 적용) 별 정확도

모델 버전	정확도(acc)
KoELECTRA-base-cls(Default)	**90.210**
KoELECTRA-base-v1(token-ffnn)	**90.206**
KoELECTRA-base-senti-v2(cls-ffnn)	**90.292**
KoELECTRA-base-senti-v3(cls-only)	**90.310**
KoELECTRA-base-senti-crawling-v4(cls-only)	**90.534**



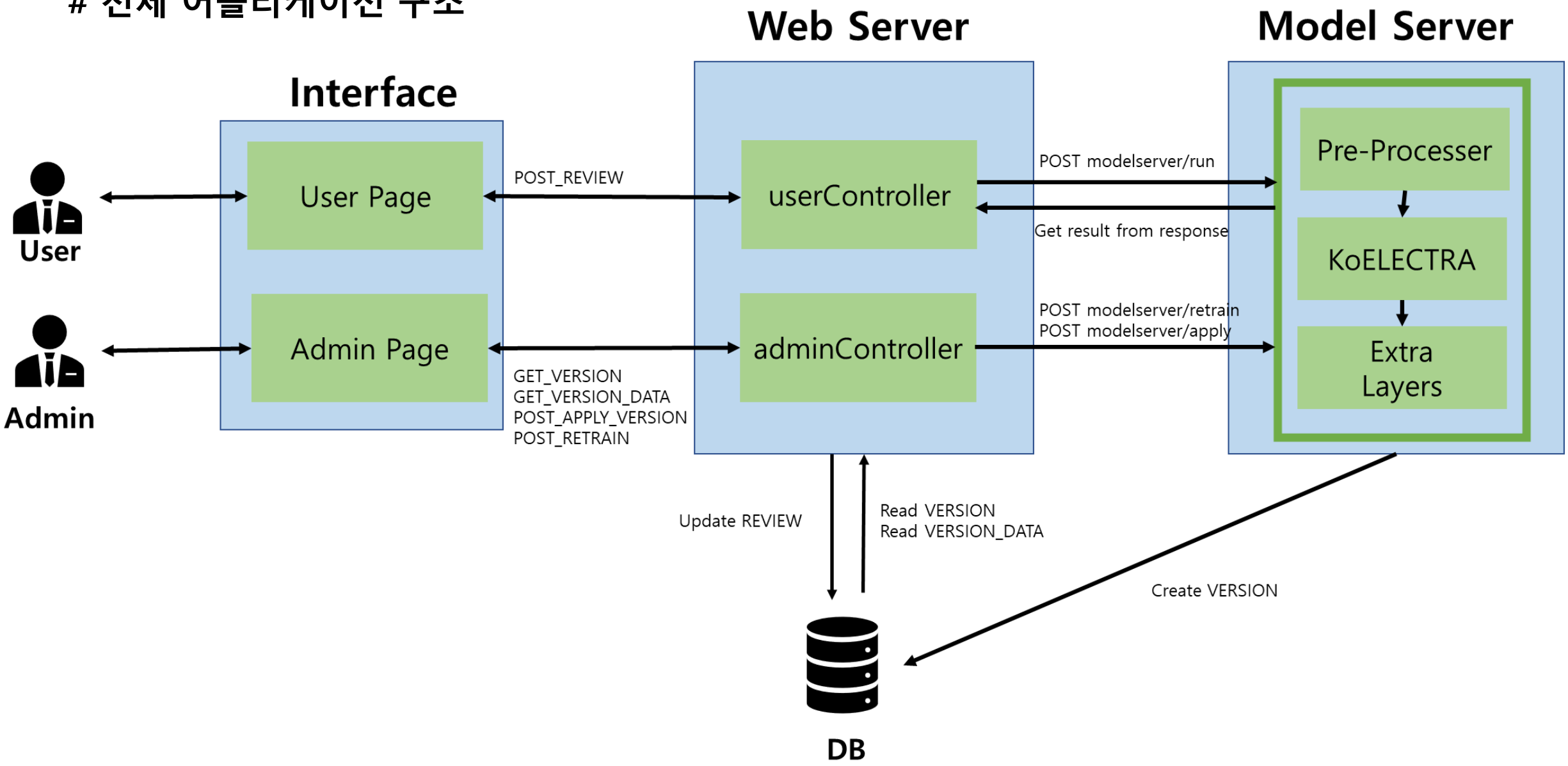




테스트 데이터(nsmc)에 오류가 많이 존재함.

최종 모델에 크롤링 데이터를 training, test set으로 포함시켰을 때 → acc 93.393

전체 어플리케이션 구조


Web Server
Model Server
Interface
User
Admin
User Page
Admin Page
POST_REVIEW
GET_VERSION
GET_VERSION_DATA
POST_APPLY_VERSION
POST_RETRAIN
userController
adminController
POST modelserver/run
Get result from response
POST modelserver/retrain
POST modelserver/apply
Pre-Processer
KoELECTRA
Extra Layers
Update REVIEW
Read VERSION
Read VERSION_DATA
Create VERSION
DB

 # 웹 서버(Back-End)

Python의 웹 프레임워크 중 하나인 Flask로 개발하였으며, 개발한 언어 모델을 서버에 올려서 인터페이스로 부터 REST API로 Json Body를 받고, Body를 Parsing 하여 Sentence를 추출한다. 추출한 Sentence들을 각 Token으로 나누어 모델을 통과시키고 긍/부정을 도출하여 인터페이스로 긍정 : 1, 부정 : 0 을 담은 리스트를 Json Body에 담아 Response를 전송한다. 구현한 서버는 AWS EC2에 탑재하여 운영한다.

```
 역시 놀란 감독이었습니다. 미리 내용에 관해 좀 찾아보시면 이해가 쉬울 것 같습니다.
generated tokens:  ['역시', '놀란', '감독이', '##었습니다', '.', '미리', '내용에', '관해', '좀', '찾아', '##보', '##시면', '이해가', '쉬', '##울', '것', '같습니다', '.']
결과:  1
 아이하고 같이 보기에 적당한 영화입니다.딱 그 정도.
generated tokens:  ['아이', '##하고', '같이', '보기', '##에', '적당한', '영화', '##입니다', '.', '딱', '그', '정도', '.']
결과:  0
낮에도 무서울 수가 있다
generated tokens:  ['낮', '##에도', '무서', '##울', '수가', '있다']
결과:  1
영화 기술자들(김우빈 주연)비슷하게 따라했으나....뭔가많이부족한영화
generated tokens:  ['영화', '기술', '##자들', '(', '김', '##우', '##빈', '주연', ')', '비슷하게', '따라', '##했으나', '.', '.', '.', '.', '뭔가', '##많', '##이', '##부족', '##한', '##영화']
결과:  0
[1, 0, 1, 0]
127.0.0.1 - - [09/Nov/2020 18:34:48] "POST /run HTTP/1.1" 200 -
```

모델 서버 구동

 # 웹 인터페이스(Front-End)

React.js 프레임워크로 개발하였으며, MVC 모델 구조를 적용하여 유지보수 및 관리가 수월하도록 설계하였다. 사용자 페이지에서는 유저가 리뷰를 입력하고 입력한 리뷰의 긍정/부정 여부를 계산할 수 있도록 구현하고 유저가 입력한 결과와 모델이 예측한 결과를 비교하고 긍정/부정 비율의 통계를 출력해 직관적으로 확인할 수 있도록 하였으며, 관리자 페이지에서는 현재까지 입력된 데이터의 수, 버전, 모델 별 정답율을 출력하고 원하는 버전을 선택할 수 있도록 하였다.

Requirement #	Scope	Description
1.1.1	Interface	사용자가 Data(200자 이하의 영화평 문장 및 긍정/부정 여부)를 입력할 수 있어야 함.

Test #	Description	Expected Result	State
1	Text Area에 200자를 초과하여 글자 입력을 시도했을 경우 (클립보드 붙여넣기 포함)	200자 이후의 입력은 무시됨, 클립보드 붙여넣기 시에는 Truncate 됨	Pass
2	긍정/부정 스위치가 Off(부정)인 상태에서 스위치를 클릭한 경우	스위치가 On(긍정) 상태로 Toggle	Pass
3	긍정/부정 스위치가 On(긍정)인 상태에서 스위치를 클릭한 경우	스위치가 Off(부정) 상태로 Toggle	Pass

Requirement #	Scope	Description
1.1.2	Interface	'+, -' 버튼 클릭을 통해 추가 Data의 입력 또는 삭제가 가능해야 함.

Test #	Description	Expected Result	State
4	Text Area가 1개일 때 '-' 버튼을 클릭해 삭제를 시도한 경우	삭제가 일어나지 않음	Pass
5	Text Area가 10개일 때 '+' 버튼을 클릭해 추가를 시도한 경우	추가가 일어나지 않음	Pass
6	Text Area가 9개 이하일 때 '+' 버튼을 클릭해 추가를 시도한 경우	Text Area 및 스위치가 최하단에 추가됨	Pass
7	Text Area가 2개 이상일 때 '-' 버튼을 클릭해 삭제를 시도한 경우	Text Area 및 스위치가 최하단에서 삭제됨	Pass

Requirement #	Scope	Description
1.1.3	Interface	작성된 입력들을 제출할 수 있어야 함.

Test #	Description	Expected Result	State
8	공백 또는 ₩t만 있는 Text Area가 하나 이상 존재할 때 제출하는 경우	제출 요청이 무시됨	Pass
9	비어있는 Text Area가 하나 이상 존재할 때 제출하는 경우	제출 요청이 무시됨	Pass
10	모든 Text Area의 입력이 정상일 때 제출하는 경우	입력들을 json으로 변환하여 서버에 Post 요청을 보냄	Pass
11	서버에서 정상 입력 요청을 받은 경우 (일반 입력)	전송받은 json을 파싱해 DB에 저장한 후 모델 입력에 적합한 text 파일 형태로 변환하여 모델 서버로 전송	Pass
12	입력 파일이 50mb 이상일 경우	"Too Large File" Error를 Response	Pass
13	입력 파일이 50mb 미만일 경우	Input을 txt로 변환한 후 모델 서버로 Request	Pass

Requirement #	Scope	Description
1.1.4	Interface	여러 개의 Data를 한 번에 제출할 수 있도록 Excel 파일 연동 입력 기능이 제공되어야 함.

Test #	Description	Expected Result	State
14	입력 포맷에 맞지 않는 Excel 파일을 업로드한 경우	'입력 포맷이 올바르지 않은 파일입니다.' 경고 메시지를 출력	Pass
15	입력 포맷에 맞는 Excel 파일을 업로드한 경우	엑셀 파일을 파싱하여 json 형태로 구성한 뒤 서버에 Post 요청을 보냄	Pass
16	서버에서 정상 입력 요청을 받은 경우 (Excel 입력)	전송받은 json을 파싱해 DB에 저장한 후 모델 입력에 적합한 text 파일 형태로 변환하여 모델 서버로 전송	Pass

Requirement #	Scope	Description
1.1.5	Interface	입력을 바탕으로 각 Data에 대한 '긍정/부정 예측', '일치 여부' 리스트와 Data Set에 대한 '정답률(일치율)', '긍정/부정비율' 그래프 등 전체적인 긍정/부정 여부에 대한 결과가 출력되어야 함.

Test #	Description	Expected Result	State
17	서버에서의 처리가 완료되기 전 결과 페이지로 넘어간 경우	Loading indicator를 출력함(await)	Pass
18	결과 페이지에서 서버에서의 처리가 완료된 경우	서버로부터 전송받은 json을 파싱하여 영화평 Index 별 모델 예측 결과, 일치 여부를 표로 출력, 정답률과 긍정/부정 비율을 원그래프로 출력, 전체 결과를 아이콘과 텍스트로 출력	Pass

Requirement #	Scope	Description
1.2.1	Interface	관리자 페이지에 모델 버전 별 '정확도'가 표 및 그래프로 출력되어야 함.

Test #	Description	Expected Result	State
19	관리자 페이지에 진입한 경우	서버로부터 전송받은 json을 파싱하여 모델 버전 별 정확도를 꺾은선 그래프로 출력	Pass

Requirement #	Scope	Description
1.2.2	Interface	관리자가 활용 또는 Fine-Tuning 할 모델 버전을 선택할 수 있어야 함.

Test #	Description	Expected Result	State
20	관리자 페이지의 드롭다운 메뉴에서 임의의 모델 버전을 선택한 뒤 적용 버튼을 클릭한 경우	적용 버전을 텍스트로 출력하고 서버에 Post 요청을 보내 버전 변경을 알림	Pass
21	서버에서 버전 변경 요청을 받은 경우	모델 버전 상태값을 변경	Pass

Requirement #	Scope	Description
1.2.3	Interface	현재 선택된 모델을 기반으로 누적 Data Set을 통해 Fine-Tuning 하여 새로운 버전을 생성할 수 있어야 함.

Test #	Description	Expected Result	State
22	관리자 페이지에서 재학습 버튼을 클릭한 경우	서버에 Post 요청을 보냄	Pass
23	서버에서 재학습 요청을 전송받은 경우	DB의 데이터를 text 파일로 만들고 모델 서버에 학습 요청과 파일을 함께 전송	Pass
24	이미 재학습이 진행 중일 때 관리자 페이지에 진입한 경우	재학습 버튼이 있던 위치에 재학습 중 메시지를 출력	Pass
25	모델 재학습이 완료된 경우	서버에서 DB에 버전을 생성	Pass
26	유효하지 않은 Format의 버전명을 요청한 경우	"Unexpected Format" Error를 Response	Pass
27	존재하지 않는 버전을 요청한 경우	"Not Existing Version" Error를 Response	Pass
28	유효한 버전을 요청한 경우	모델 서버에서 예측 결과를 내고 결과 JSON을 웹서버에 전송	Pass

Requirement #	Scope	Description
2.1.1	Model	E-mail, URL, HTML tag와 괄호 제거 작업을 통해 Raw-Data를 최적화된 포맷으로 변환해야 함.

Test #	Description	Expected Result	State
29	입력에 E-mail이 포함되어 있는 경우	E-mail이 입력에서 제거됨	Pass
30	입력에 URL이 포함되어 있는 경우	URL이 입력에서 제거됨	Pass
31	입력에 HTML tag 또는 (content) 형식이 포함되어 있는 경우	HTML_tag와 (content)가 입력에서 제거됨	Pass

Requirement #	Scope	Description
2.1.2	Model	변환한 Data를 KoELECTRA의 Vocabulary에 대응시켜 Word-Embedding(Tokenizing)해야 함.

Test #	Description	Expected Result	State
32	전처리된 입력 문장을 Tokenizer에 입력한 경우	입력 문장과 Tokenizing 이후 토큰들이 정상적으로 대응됨	Pass

Requirement #	Scope	Description
2.2.1	Model	KoELECTRA의 Input Sentence에 한국어 감성사전을 적용하여 각 Position에 맞는 가중치 벡터(96-vector)들을 구성해야 함.

Test #	Description	Expected Result	State
33	KoELECTRA의 결과가 한국어 감성사전에 존재할 경우	-2,-1,0,1,2의 긍/부정 값과 매핑된 가중치 벡터(96-vector)가 정확히 대응되어야 함.	Pass

Requirement #	Scope	Description
2.2.2	Model	한국어 감성사전 적용에 관한 오차 보정을 위해 [CLS] 토큰의 Hidden State를 나머지 토큰들에 Scaled Dot-Product Attention(SDA)해 주어야 함.

Test #	Description	Expected Result	State
34	KoELECTRA의 결과가 정상 출력되었을 경우	SDA 적용을 위해 CLS 벡터와 다른 벡터의 차원(768-vector)이 일치해야 함.	Pass
35	SDA를 적용했을 경우	결과 벡터의 차원(768-vector)이 유지되어야 함.	Pass

Requirement #	Scope	Description
2.2.3	Model	Position이 일치하는 'SDA를 마친 벡터'와 '가중치 벡터'들을 Contatenate 하여 Bi-GRU를 통과시키고, 문맥 흐름 정보를 담고 있는 양 끝의 출력 벡터를 FFNN Layer 입력 차원에 맞춰 Projection 해야 함.

Test #	Description	Expected Result	State
36	Scaled Dot-Attention가 문장에 적용됐을 경우	'SDA를 마친 벡터'(768-vector)와 '가중치 벡터' (96-vector)의 concat 결과 정확히 두 차원이 더해져야 함	Pass
37	Bi-GRU 결과가 정상적으로 나왔을 경우	Bi-GRU의 결과 벡터를 Classification을 위한 차원으로 정확히 Projection 해야 함 (768*2-vector -> 768-vector)	Pass

Requirement #	Scope	Description
2.3.1	Model	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.

Test #	Description	Expected Result	State
38	CLS와 RNN Layer의 두 결과 벡터를 FFNN에 통과시켰을 경우	결과 벡터로 하나의 768-vector가 출력되어야 함	Pass
39	Classifier에 768-vector를 입력했을 경우	결과값으로 ['0','1']와 같은 포맷의 2차원 벡터가 출력되어야 함	Pass

C1. 사용자의 60% 이상이 편리함을 느낄 수 있도록 인터페이스를 구성한다.

→ Pass

C2. 웹 인터페이스를 통해 손쉽게 모델 버전 관리가 가능하도록 한다.

→ Admin Page에서 현재 적용 버전을 기반으로 새로운 모델 생성, 정확도 비교 가능: Pass

C3. 모델을 어플리케이션과 명확히 구분하여 독립적인 모델로써 다른 분야에서도 활용할 수 있도록 한다.

→ 독립적인 모델 코드, 가중치 파일을 통한 버전 관리 가능: Pass

C4. 모델의 입력 영화평 1개 당 처리 시간이 500ms 이하가 되도록 한다.

→ 50개의 임의 데이터 입력 기준 5s 소요 :Pass

C5. 다양한 아이디어를 적용하여 NSMC 기준 모델의 정확도를 KoELECTRA Baseline(90.21%) 보다 0.25%p 이상 향상시킨다.

→ nsmc 기준 정확도 0.324% 향상: Pass

Functional Requirement	Pass/Fail Criteria	System Test Case
1.1.1 영화평 입력	C1, C3	1-3
1.1.2 입력공간 추가/삭제	C1, C3	4-7
1.1.3 입력 제출	C1, C3	8-13
1.1.4 Excel File 제출	C1, C3	14-16
1.1.5 결과 출력	C1, C3	17-18
1.2.1 모델 버전 정확도 출력	C1, C3	19
1.2.2 모델 버전 선택	C1, C3	20-21
1.2.3 데이터 학습	C2, C3	22-28
2.1.1 데이터 전처리	C3, C4, C5	29-31
2.1.2 워드 임베딩	C3, C4, C5	32
2.2.1 감성 사전 적용	C3, C4, C5	33
2.2.2 Scaled-Dot-Attention	C3, C4, C5	34-35
2.2.3 FFNN	C3, C4, C5	36-37
2.3.1 긍/부정 결과 예측	C3, C4, C5	38-39