국어 문자열 감성 분석 프로젝트

이건뭐사람아니냐고

Electra 기반의 영화평 감성 분석

김주언 류한길 유병찬 최보라

알파쿠

프로젝트 개요

팀 구성 (건국대학교 컴퓨터공학과 교수) 김학수, (건국대학교 컴퓨터공학과 4학년) 김주언, 류한길, 유병찬, 최보라

프로젝트 명 국어 문자열 감성 분석

프로젝트 목표 입력 받은 국어 문자열의 감성을 긍정 또는 부정으로 분류하는 딥러닝 모델을 설계하고, 기존 시스템보다 정확도를 개선하는 것:

- 최신 논문[1]의 최종 NSMC[2] Task 의 최종 정확도인 90.51% 를 넘기는 것을 목표로 한다.

사용한 딥러닝 언어 모델: Electra

이 프로젝트에선, Electra 모델을 기반으로, 사전학습과 미세조정[3]을 거쳐 성능을 향상시킨다. Electra 는 BERT 의 학습 방식을 바꾸어서, 좀 더 가벼우면서도 성능을 개선한 언어 모델이다:

- 가중치 공유를 통해 파라미터를 감소시켜서 가볍다
- Dynamic Masking 으로 학습 효과를 증가
- Is Next Prediction 을 Sentence Order Prediction 으로 변경하여 학습 효과 증가
- Generator - Discriminator 로 기존처럼 사전학습 후에, 옳게 추측했는지 여부를 바탕으로 한 번 더 학습시킨다. 학습 효과가 증가한다

사용된 데이터 셋

Electra 사전 학습 데이터: 뉴스기사, 나무위키, 위키피디아 문자열 데이터 (총 50GB)

미세조정 학습 데이터: NSMC 리뷰 15만 개 (공개), 자체 구축 네이버 영화평 정제 데이터 115만 개

Feature 추가 학습 데이터 1: 군산대 한국어 감성사전 (공개)

Feature 추가 학습 데이터 2: 네이버 영화평 word-piece 별 감성 레이블링 데이터

논문[1] : <BERT 기반 Variational Inference와, RNN을 이용한 한국어 영화평 감성 분석(박천음, 이창기 2019)>
NSMC[2] : 네이버에서 구축하여 공개한 네이버 영화평 데이터로, 학습용 15만 건, 성능 평가용 5만 건의 리뷰로 구성되어 있다.
미세조정[3] : 미세 조정(Fine-tuning)이란, 기반이 되는 특정 딥러닝 모델 위에 별도의 신경층을 설계하거나, 입력 데이터를 정제하는 등의 수단으로 모델을 변경하여 성능을 향상시키는 것을 의미한다.

알파쿠

딥러닝 모델 설명 1

개발 과정

이 프로젝트의 알고리즘은, 다음과 같은 단계로 개발되었다:

1. 공개되어 있는 Electra 를 기반으로 사전학습한, 자체 개발 Electra 모델을 개발
2. 웹크롤링을 이용하여 학습에 사용되는 자체구축 데이터의 크기를 늘리고, 별도의 처리 과정을 거쳐 양질의 데이터가 남도록 데이터 정제
3. 그 모델에 군산대 감성사전을 활용하여, 각 단어의 감성을 별도로 추가 학습시키는 방식으로 감성사전 적용
4. 그 모델 위에 Self-Attention 과 Bi-LSTM 계층을 쌓고, CLS 토큰과 Bi-LSTM 의 출력 양단의 토큰을 Affine 하여 최종 예측 값을 산출

1 - Electra 모델 사전 학습 (약 40일 소요, 0.17% 성능향상)

연구실에서 자체구축한 데이터로 Electra 모델을 사전학습시킨다.

- Electra 는 최고 성능의 언어 모델인 BERT 의 학습 방식을 바꾸어서, 좀 더 가벼우면서도 성능을 개선한 언어 모델이다.
- Electra 사전 학습 데이터: 뉴스기사, 나무위키, 위키피디아 문자열 데이터 (총 50GB)
- Word-Piece[1] 단위로 학습: 비정형 문자열을 처리하기 위함

2 - 학습용 데이터 자체 구축 (약 3일 소요, 1.97% 성능향상)

- 네이버에서 공개한 NSMC 데이터 15만건 외에도, 웹크롤링을 사용하여 350만건의 영화평 데이터를 구축하였다.
- 2020 ~ 2010 년간의 모든 영화평 리뷰, 1100만 건을 크롤하고, 리뷰 별점을 기준으로 긍정/부정을 레이블링하였다.
- 긍정과 부정 문자열이 1:1 의 비율로 구성되도록 하였고
- 빈 문자열이나 의미 없는 특수문자 또는 크롤시에 변환된 ASCII 코드 등을 제거하거나 수정하여 데이터의 품질을 높였다.
- 그 과정에서 불필요한 영화평 데이터를 제외하고, 정제된 350만 건을 남겨 학습에 사용하였다.

Word-piece[1] : 단어를 구성하는 일종의 하위어(sub-word)로, 자연어 처리 분야에서 어절도 음절도, 형태소도 아닌 단위로 문장을 분석하기 위해 사용한다. 사용하는 모델에 따라 별도의 Vocab.txt 파일에 저장되어 있다.

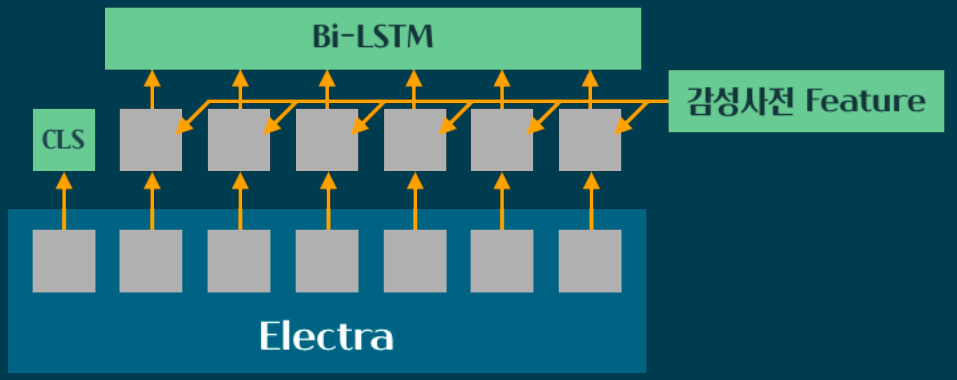
알파쿠

딥러닝 모델 설명 2

3 - 감성 사전 적용 (약 5일 소요, 0.04% 성능향상)

감성 사전 토크나이징

군산대 감성사전은 해당 1 ~ 8 어절의 단어 쌍을, 긍부정 정도에 따라 -2, -1, 0, 1, 2 의 5 개의 점수로 구분하였다. 그런데 우리 모델은 어절이 아닌 word-piece[1] 단위로 문자열을 분석하기 때문에 그대로는 사용할 수가 없어서, 기존 감성 사전의 인덱스를 전부 Electra 토크나이저를 사용해 우리가 개발하는 모델과 호환되는 word-piece 단위로 변경하였다.



[그림2] Feature 추가 방식의 감성 사전 적용 구조도

감성 사전 적용 방법 - 1

첫 번째 적용 방법은, 감성 사전을 통해 얻은 값을 **각 토큰에 concatenate 방식으로 결합**하는 것이다. Electra 에 입력되는 각 토큰에 대응되는 word-piece 쌍을 감성사전에서 검색하여, 해당하는 감성 값을 추출하여 word-piece 쌍을 구성하는 토큰의 갯수로 나누어 각 토큰에 결합한다. 이때, -2~2의 값을 가지는 감성 값을 0~1 사이의 값으로 변환한다.

감성 사전 적용 방법 - 2

첫 번째와 마찬가지로 입력 문자열을 구성하는 각 토큰 쌍에, 감성 사전을 바탕으로한 감성 값을 대응시킨다. 그 감성 값들의 벡터를 정답 시퀀스로, 각 단어의 감성 값을 추론하도록 **Electra 를 추가 학습**시킨다.

이 프로젝트에선, 감성 사전 적용방법 2를 사용하여 진행

Word-piece[1] : 단어를 구성하는 일종의 하위어(sub-word)로, 자연어 처리 분야에서 어절도 음절도, 형태소도 아닌 단위로 문장을 분석하기 위해 사용한다. 사용하는 모델에 따라 별도의 Vocab.txt 파일에 저장되어 있다.

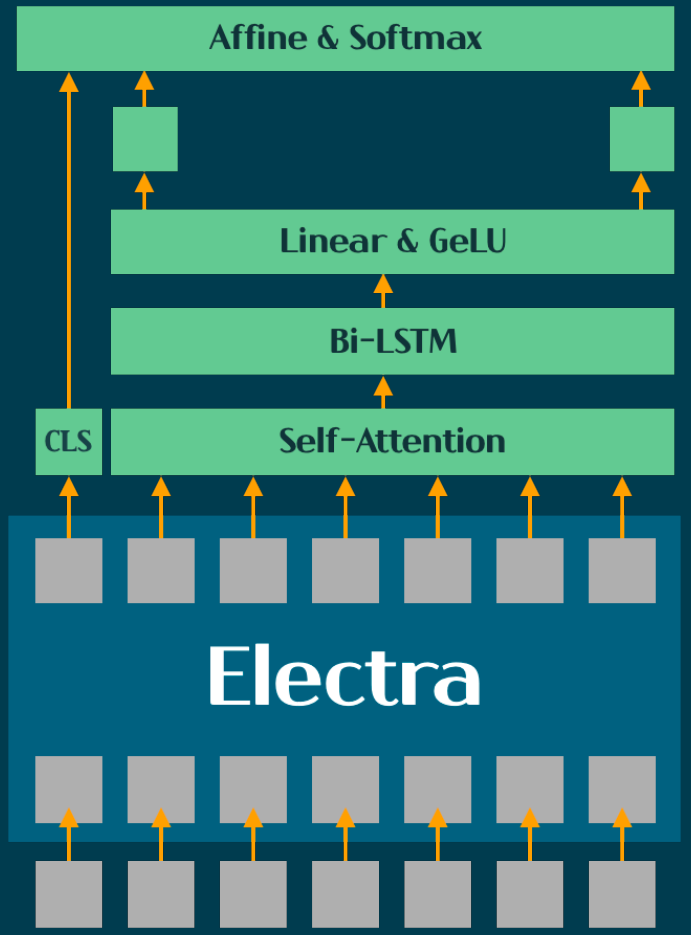
딥러닝 모델 설명 3

4 - 미세조정 레이어 설계 : Bi-LSTM 신경층 (약 60일 소요, 0.38% 성능향상)

- Electra 의 출력 벡터 중, 문장 전체의 정보를 갖고 있는 CLS 토큰을 따로 분리하고 나머지는 Self-Attention Layer[1] 와 Bi-LSTM[2] 을 거쳐 학습시킨다.
- 문장 내에서 각 단어의 연관 관계 정보가 감성 분석에 유효할 수 있으므로, Self-Attention Layer 를 삽입하였다.
- 또한, 문장을 구성하는 각 단어의 감성 정보는 앞 단어 뿐 아니라 뒤의 단어에도 영향을 받을 것이므로 문맥 정보를 포함하는 Bi-LSTM 을 선택하였다.
- Bi-LSTM 출력의 양단 토큰은 각각 앞에서 뒤 방향으로의 문맥 정보와 뒤에서 앞 방향으로의 문맥 정보를 갖고 있다. 따라서 이 두 토큰과 CLS 토큰을 Affine[3] 계층을 사용해 결합하면 문장 전체의 정보를 고려한 벡터를 얻을 수 있다.
- 최종적으로 얻은 이 토큰을 긍정(1)과 부정(0)의 두 클래스로 분류한다.

팀원간 역할 분담

이름	역할
김주언	팀장, 발표, 발표자료, 보고서, 학습데이터 350만개 구축, fintuning 최적화, 감성사전 토크나이징, 감성사전 학습로직 설계, 모델학습
류한길	보고서, self-attention layer 설계 감성사전 concat 로직 설계, 모델학습
유병찬	보고서, double bi-lstm with skip layer 설계, 감성 학습로직 설계, 모델학습
최보라	보고서, finetuning, masking 로직 설계, 감성사전 concat 로직 설계, 모델학습



[그림1] 사전학습된 Electra 모델과 미세조정 레이어의 구조

Self-Attention[1] : 각 단어가 서로에 대한 어떤 관계성을 갖고 있는지를 저장하는 신경층.
Bi-LSTM[2] :게이트 기법을 통해 순환 신경 회로망(RNN)의 한계를 극복한 모델인 LSTM을 순방향뿐 아니라 역방향의 결과를 함께 이용하는 모델이다. 문맥(Context)을 기반으로 하는 연관성 분석에 유리하여 속도를 고려한 다양한 NLP 문제에 널리 활용되고 있다.
Affine[3] : Affine 변환을 수행하는 신경층으로, Linear 계층이라고 보면 쉽다.

알고리즘 상세 1

감성 사전 토크나이징 & 문자열 라벨링 방법

1. 군산대 감성 사전의 키 값을 토크나이징 한다
2. 토큰 문자열과 감성 값을 매칭한 Dictionary 데이터를 저장한다
3. 중복되는 키 값 중에 최대 매칭 키 값을 찾기 위해, 같은 토큰으로 시작하는 키 값을 모아 둔 Dictionary 데이터를 또 만든다.
4. 이 두 Dictionary 를 사용하여 토큰 문자열에 최대 매칭되는 감성 사전 키 값을 찾아내고, 그 키 값으로 감성 값을 라벨링한다

ratings_test_sdc_2.txt × ratings_test_sd

```
122  가리지 않고 0
123  가마득하다 -1
124  가만있지 못하고 -1
125  가망 없는 -1
126  가망이 없게 -2
127  가벼운 마음으로 1
128  가벼운 말이나 -1
129  가벼운 상처 -1
130  가벼운 슬픔 -1
131  가벼운 행동으로 -1
132  가볍게 -1
133  가볍게 까부는 -1
134  가볍게 까불다 -1
135  가볍게 쑤시듯 -1
136  가볍게 웃다 1
```

ratings_test_sdc_2.txt × ratings_test_sdc.txt ×

```
122  24069 10817 0
123  31975 6384 10624 -1
124  19337 25669 11714 -1
125  153 6041 10733 -1
126  153 17216 15178 -2
127  18241 19828 1
128  18241 12213 5853 -1
129  18241 16045 -1
130  18241 20222 -1
131  18241 26769 -1
132  18959 -1
133  18959 508 10880 -1
134  18959 508 6275 5727 -1
135  18959 3282 5719 6477 -1
136  18959 3647 5727 1
```

토큰과 감성 사전의 키 값을 연결해주는 Dictionary

utils.py × ratings_train_sdc.txt × ratings_test_sdc_2.txt × ratings_test_sdc.txt × tokenized_sentdic.txt × SentiWord_Dict.txt × ke

```
31  13236  [[13236, 20119, 10899], [13236, 16542], [13236, 5980]]
32  24069  [[24069, 30645], [24069, 10817]]
33  31975  [[31975, 6384, 10624]]
34  19337  [[19337, 25669, 11714]]
35  18241  [[18241, 12213, 5853], [18241, 19828], [18241, 16045], [18241, 20222], [18241, 26769]]
36  18959  [[18959, 508, 6275, 5727], [18959, 3282, 5719, 6477], [18959, 4376, 5764, 11127], [18959, 508, 108
37  16304  [[16304, 5728, 25527, 6353, 5815], [16304, 5728, 2810, 6095, 10624], [16304, 5728, 2810, 24350], [1
```

알고리즘 상세 2

토크나이징 된 감성 라벨링 데이터

```
1498    왠만해선 영화평많이 안쓰는데 감동적이고 재미있네요^^   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0.625,0.625
1498    아는 그릇의 크기고 블랙코메디고 미국문화고 나발이고 한국의 김만봉씨가 미국가서 삽질하는 내용이면 과연 어떤 평점일까? 자국 문화를 풍자하기 위하 다른
1498    대상받을 만 하지않은가 .   0.75,0,0,0,0,0,0,0,0
1498    사람들이 아무리 그래도 그렇지 평점을 이렇게 박하게 줘서 쓰나 ㅉㅉㅉ~ 그래도 천녀유혼을 다시 떠올리게 한다는 점이 가상해서 원래 생각했던 평점에서 후
1498    어설픔을 살짝 벗어나다.    0,0,0,0,0,0.375,0.375,0
1498    늘리려고 아주 드라마를 ㅂㅅ삼류작으로 만들엇네 명불허전엠비씨      0,0,0,0,0,0.375,0.375,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1498    최고의 드라마이걸 이제야보게되다니머리가 아프긴합니다만 몰입하게되는 드라마!추천합니다:) 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
1498    맨온파이어 느낌. 하지만 더 좋음 0,0,0,0,0,0,0,0,0.75,0.75
1498    브루스 윌리스 이색히 ㅣ요즘에 떡밥 존ㄴ 날리네 ㅅㅂ ―― 브루스 나와서 뭔가 대작이다 싶어서 평점도 안보고 받았더니 ㅁㅊ 000기 베를린보다 더 구려
1498    시사회에서 보고 엄마랑 또 보고싶다는 생각을 했어요..가족과 함께 보면 좋을듯^^    0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0.625,0.625
1498    아 짜증나는 영화 재미없다 아무짝에 의미없는 정사들    0,0.25,0.25,0,0.375,0.375,0,0,0,0,0,0,0,0,0
1498    최대한 몰입해서 보려고 노력하고 자막이 많아 잠시 놓친 자막까지도 되돌려 보고 최대한 집중해서 봤지만 개인적으로 재미없는 영화. 1.2.3 전부 별로다.
1498    이거 촬영한 넘들도 대단하다ㅋㅋ  0,0,0,0,0.75,0.75,0,0
1498    아이들이 완전 초집중해서 보더라고요 ㅎㅎ 재밌었어요~   0,0,0,0,0,0,0,0,0.625,0.625,0,0,0
1498    평점 조절위해 10점 내가봤을땐 6~7점 보다 아래로 준애들은 문학적 소양이 전혀 없는애들 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
1498    이걸 극장에서 봤다. 한국영화만 조는 친구들때문에 나혼자 외국영화 본다고 했다가 다수결 때문에 이걸!! 정말 최악~~~~ 당시 내가 보고싶었지만 애들이
1498    결말이 만족스럽네요^^ 0,0,0,0,0,0.625,0.625
1498    피아프의 출생부터 유년기와 전성기 그리고 사망까지를 교차식구성으로 보여지면서 피아프의 인생만큼 슬프고 아름다운 그녀의 음악과 함께 보는 사람의 감정선
1498    슬프고 아프다. 영화가 수작이다. 0,0,0.25,0.25,0,0,0,0,0,0
1498    2014년에서야 이 영화를 보내요.좋다. 0,0,0,0,0,0,0,0,1.0,0
1498    어쩔수없는 줄거리의 한계    0,0,0,0,0,0,0.25
1498    마약을 팔아서 행복해진 남녀를 다룬 영화    0.25,0,0,0,0,0,0,0,0,0,0
1498    아름다운Andrew McCarthy^^*     0,0,0,0,0,0,0,0,0,0,0,0.625,0.625,0
```

알파쿠

알고리즘 상세 3

감성 사전 학습의 한계

감성 사전의 표제어가 충분하지 않아, 실제 리뷰에 사용되는 어휘를 커버하지 못했다.
그러한 이유로, 유의미한 성능향상이 없어 최종 모델에서는 제외되었다. (성능향상 0.04%)

tils.py × ratings_train_sdc.txt × ratings_test_sdc_2.txt × ratings_test_sdc.txt × tokenized_sentdic.txt × SentiWord_Dict.txt × keys_start_with.txt

```
========== Result ============
입력 문자열 :  주인공이 힘에 눌려 희미해져서 희뭇하고 가뿐하게 부끄러운 줄 모르고
tokens : [21449, 5650, 5664, 948, 6070, 5630, 5810, 18926, 5665, 5630, 6533, 10546, 153, 6039, 10571, 20900, 5749, 4029,
tot_labels : [0, -0.5, -0.5, -0.5, -0.5, -0.25, -0.25, -0.25, -0.25, 0, 0, 0, 0.3333333333333333, 0.3333333333333333, 0.
반복 횟수 (i) : 19
key[i:] : []
len(key) : 19
len(tot_labels) : 19
```

기대 했던 적용 모습

tils.py × ratings_train_sdc.txt × ratings_test_sdc_2.txt × ratings_test_sdc.txt × tokenized_sentdic.txt × SentiWord_Dict.txt × keys_start_with.txt

```
========== Result ============
입력 문자열 :  흠...포스터보고 초딩영화줄....오버연기조차 가볍지 않구나
tokens : [5620, 18, 18, 18, 21557, 11239, 4472, 6152, 14865, 6069, 18, 18, 18, 18, 17238, 5714, 5662, 12258, 16304, 5720
tot_labels : [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
반복 횟수 (i) : 22
key[i:] : []
len(key) : 22
len(tot_labels) : 22
```

실제 대부분 사례

20 만개의 NSMC 데이터 중 활용 가능한 것은 **고작 0.8%** 정도 뿐

알파쿠

성능 평가 및 분석

성능 평가

시스템 분류	Base (고성능 모델)	성능 향상 폭
koElectra (기존 시스템)	90.21%	
50GB 말뭉치 Electra 사전학습	90.38%	0.17%
+ Bi-LSTM 레이어 추가	90.76%	0.38%
+ 365만 개 영화평 감성 말뭉치	92.73%	1.97%

기존 시스템 대비 성능

기존 모델	우리 모델	성능 향상 폭
(koElectra base) 90.21%	92.73%	**2.52%**
(최신 BERT 논문) 90.51%	92.73%	**2.22%**

성능 평가 결과 해석

- 각 단계 별로 실제 성능이 향상되었음을 알 수 있다.
- 해결하려는 문제(여기서는 감성분석)에 맞게 미리 레이블링된 구축 데이터가 성능에 가장 큰 영향을 준다는 것을 알 수 있다.

비정형 문장도 처리할 수 있는가? – 비정형 문장 처리 예시

입력된 문자열	예측값	정답
콩콩 ~ ~ ~ ~ ㅋㄲㅈㅁ	부정	부정
- -	부정	부정
투니버스에서 보는것만으로도 만족한다 ㅡㅡ	부정	부정
굿굿구후구수구수수구구숫굿	긍정	긍정
ㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋㅋ	부정	부정
다 성지순례	부정	부정
재미읎다	부정	부정

입력된 문자열	예측값	정답
쎄이헬로우투마이리를프렌 ~ !	긍정	긍정
추남추녀의검 . . 이 두명 정말 쵝오의 감초다 ㅎㅎ	긍정	긍정
꿀잼 팽긴 커여워ㅇㅅㅇ	긍정	긍정
ㅉㅉㅉ	부정	부정
존스 : 존ㅡ나 쓰ㅡ레기	부정	부정
뻔한 영호ㅓ . . . . (십자)	부정	부정

알파쿠

프로젝트 총평

기술적 진보

- 이 프로젝트의 목표였던, 기존 시스템의 정확도인 90.51% 보다 2.22% 향상된 92.73%의 정확도를 얻음으로써, 프로젝트의 목표를 달성
- 기존에 시도되지 않은 Fine-tuning Layer 인, Self-Attention, Bi-LSTM, CLS 의 조합을 사용하여 0.38% 의 실제적인 성능 향상을 얻음

국립국어원 주관 자연어 처리 대회, 은상 수상

- 국립국어원에서 주관한, <2020 국어 처리 시스템 경진대회>에서 지정 분야인 '국어 감성 분석'을 주제로 은상을 수상했습니다.
- 한국어 감성사전 적용을 시도했다는 점에서 기술적 창의성 점수를 얻었고,
- 주최측 자체 성능 평가에서, 다른 시스템 대비 높은 정확도를 달성한 것이 수상에 결정적인 영향을 미쳤습니다.

활용 방안 : 소비자 피드백 실시간 분석

- 블로그나 댓글, 커뮤니티 포스팅 등 웹 상의 데이터를 크롤하여, 특정 제품이 언급된 문자열을 수집하고,
- 우리가 개발한 한국어 감성분석기로 감성을 분석한다.
- 이와 같은 방법으로, 다양한 기업들이 소비자의 피드백을 실시간으로 분석할 수 있게 된다.

기대 효과

- 빠르게 변하는 소비 트렌드를 신제품에 반영할 수 있다.
- 현재 제품에 대한 수요를 미리 예측하여 제품의 생산을 탄력적으로 변화시킬 수 있다.
- 소비 트렌드 분석, 소비자 피드백 분석을 자동화할 수 있기 때문에, 기존에 들어가던 분석 비용 및 시간을 절감할 수 있다.
- 이러한 효과들을 바탕으로, 특히 미래 신사업과 관련이 있는 소프트웨어 업종(AR/VR, IOT, 미래차, 딥러닝프로그램)에 속하는 기업들은 빠르게 변하는 4차 산업혁명에 기민하게 대응할 수 있다.

데모 웹앱 상위 디자인 1

데모 프로그램 소개

국어 문자열 분석 모델의 시연을 위해, 문장을 입력 받아 긍부정 여부를 출력해주는 웹프로그램을 설계한다. 이 문서의 앞 부분에서 설명하는 웹앱은 모델 시연을 위한 것이며, 이 프로젝트의 본질이 아님을 밝힌다.

시스템 요구사항 1: 기능적 요구사항

- 사용자로부터 감상평을 입력받는다
- 감성 분석 결과를 출력한다 (긍정/부정)
- 사용자가 감성 분석 결과가 맞는지 틀린지 정보를 입력한다

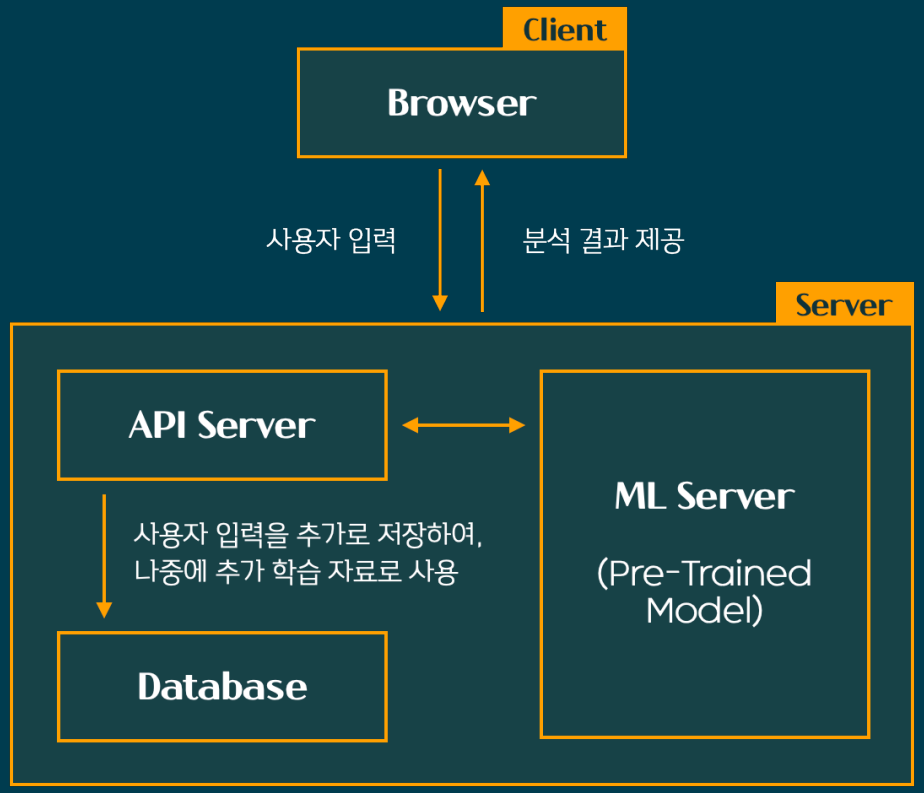
시스템 요구사항 2: 비기능적 요구사항

- 감성 분석 정확도가 91% 이상이 되도록 한다
- 사용자의 입력을 받은 순간부터 응답까지 3초 이내가 되도록 한다

HTTP Interface

GET / : 메인 페이지 호출
GET /analysis : 입력 문장 보내고 분석 결과 호출
POST /analysis : 리뷰 텍스트와 정오 판단 값 보내서 DB 에 Reivew 데이터 저장

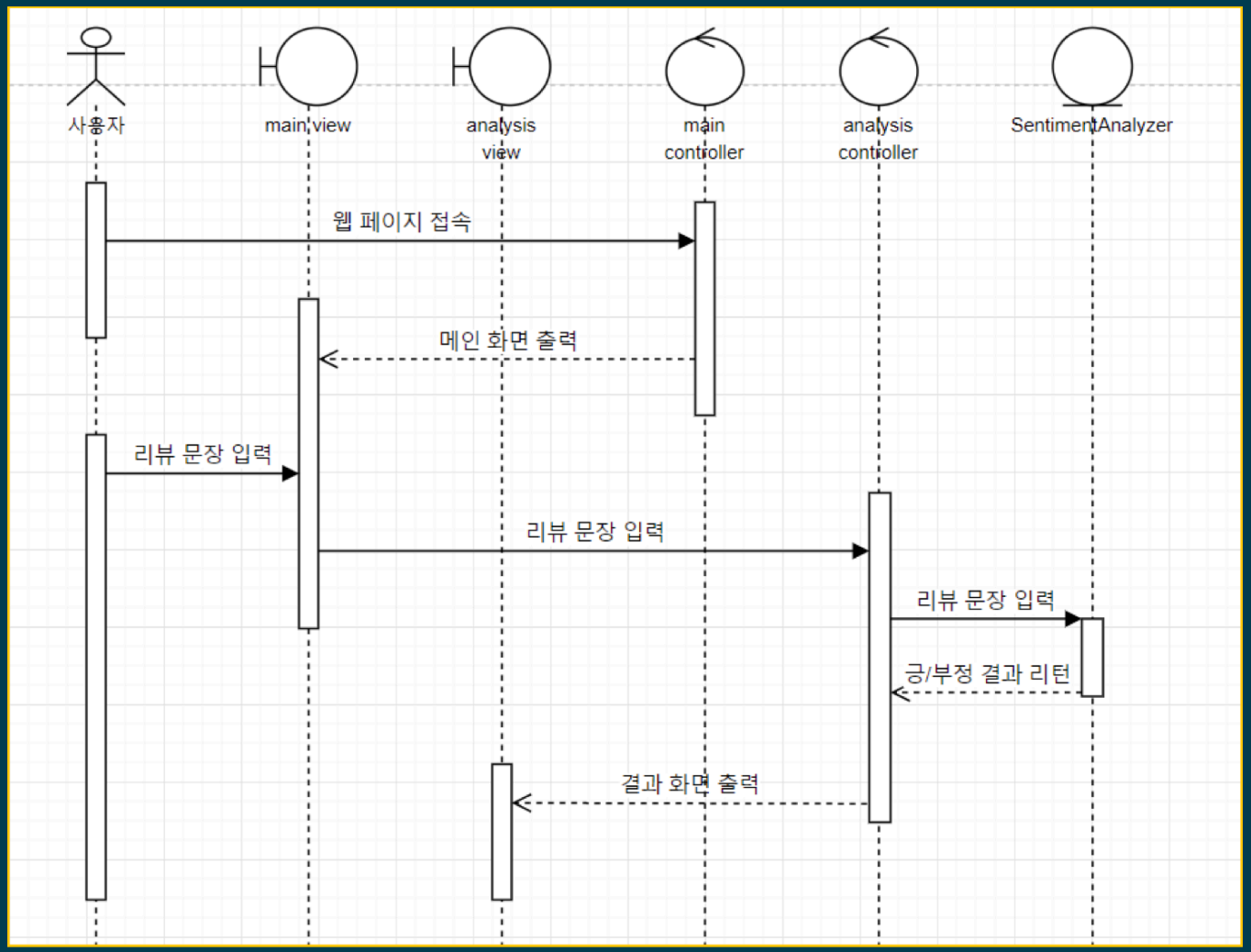
데모 웹앱 아키텍쳐 다이어그램



알파쿠

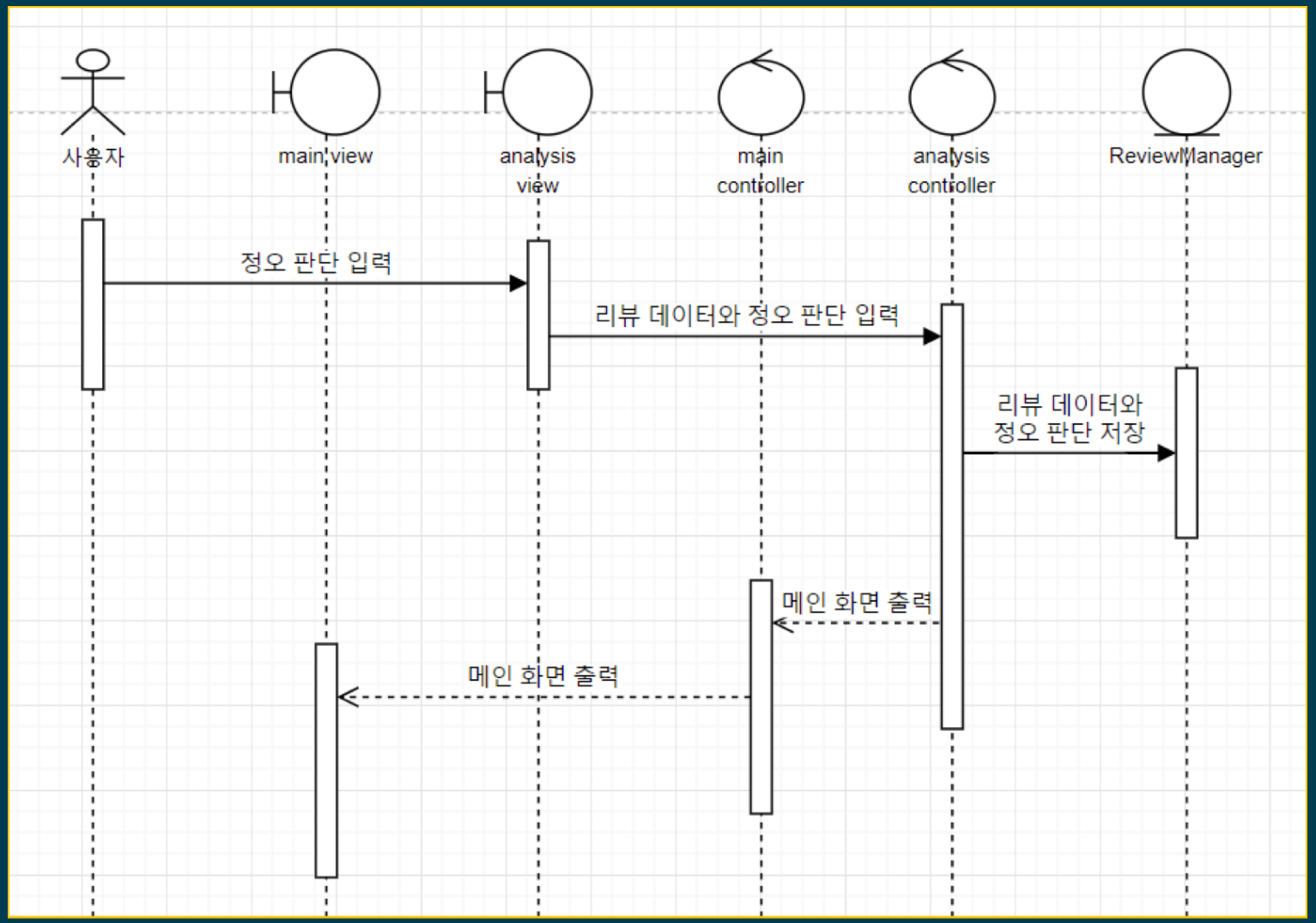
데모 웹앱 상위 디자인 2

시퀀스 다이어그램 1: 리뷰 문자열 입력 / 분석 / 분석 결과 출력



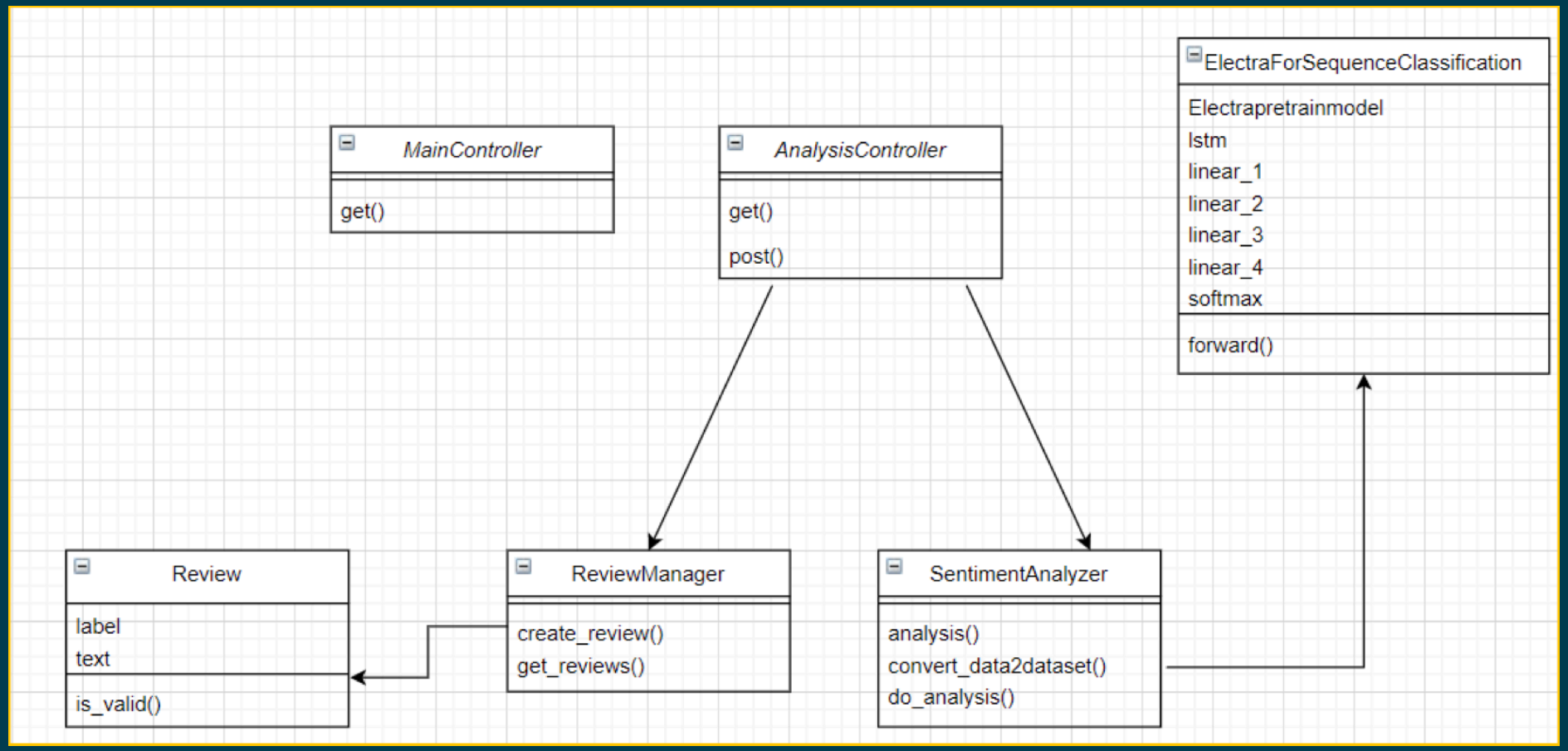
데모 웹앱 상위 디자인 3

시퀀스 다이어그램 2 : 분석 결과 정오 판단 입력 / 메인 화면 출력



데모 웹앱 상세 디자인

클래스 다이어그램



데모 웹앱 시스템 테스트

추적성 분석표

요구사항	상위디자인	상세디자인
사용자 → 리뷰 문자열 입력 (서버)	1.1 메인 화면 출력	MainController.get()
	1.2 리뷰 문자열 입력	AnalysisController.get()
서버 → 리뷰 분석 및 결과 출력 (클라이언트)	2.1 리뷰 분석	SentimentAnalyzer.analyze()
	2.2 분석 결과 화면 출력	AnalysisController.get()
사용자 → 분석 결과 정오 판단 입력 (서버)	3.1 분석 결과 정오 입력	AnalysisController.post()
	3.2 분석 결과 정오 DB 저장	ReviewManager.create_review()

시스템 테스트

Identifier	Input specification	Output specification	Result
KTSAS_STC_001_001	url '/' 에 입력해서 접속, 사용자가 분석 정오 판단 버튼 입력	리뷰 입력 창, 분석 요청 버튼이 화면에 나타난다.	Pass
KTSAS_STC_001_002	사용자가 분석 요청 버튼 입력	분석 결과 표시 창, 분석 정오 판단 사용자 입력 버튼이 화면에 나타난다.	Pass
KTSAS_STC_002_001	사용자가 분석 요청 버튼 입력	전송된 리뷰 문장으로 prediction을 진행하고 그 결과를 화면에 표시한다.	Pass
KTSAS_STC_003_001	사용자가 정오 판단 입력	입력한 정오 판단과 리뷰 문장을 DB에 저장.	Pass

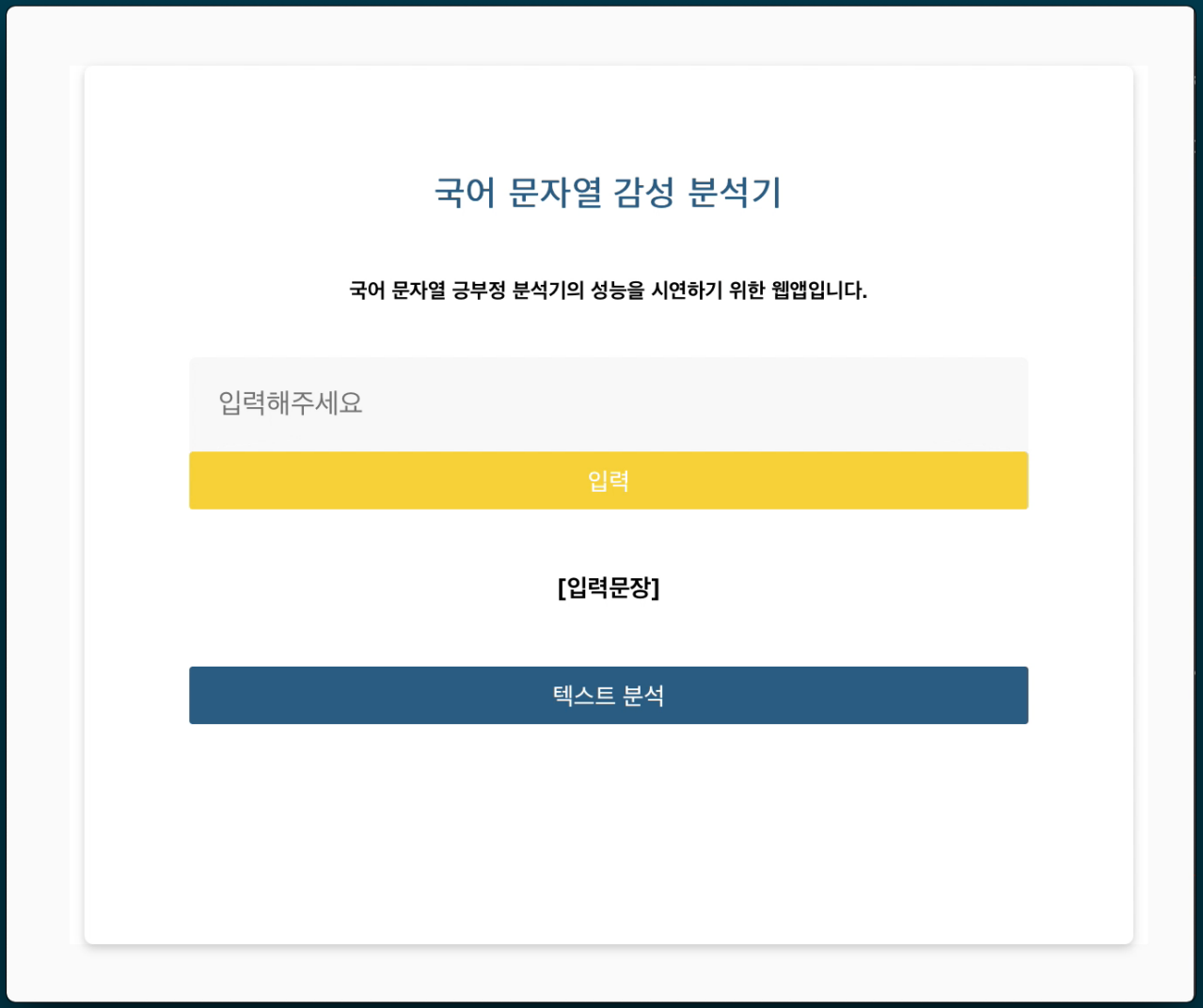
Pass / Fail 기준 및 결과

시스템 테스트의 모든 테스트를 Pass 하고, 모델의 NSMC Task 의 최종 정확도가 90.51% 를 넘기는 것 → **Pass**

알파쿠

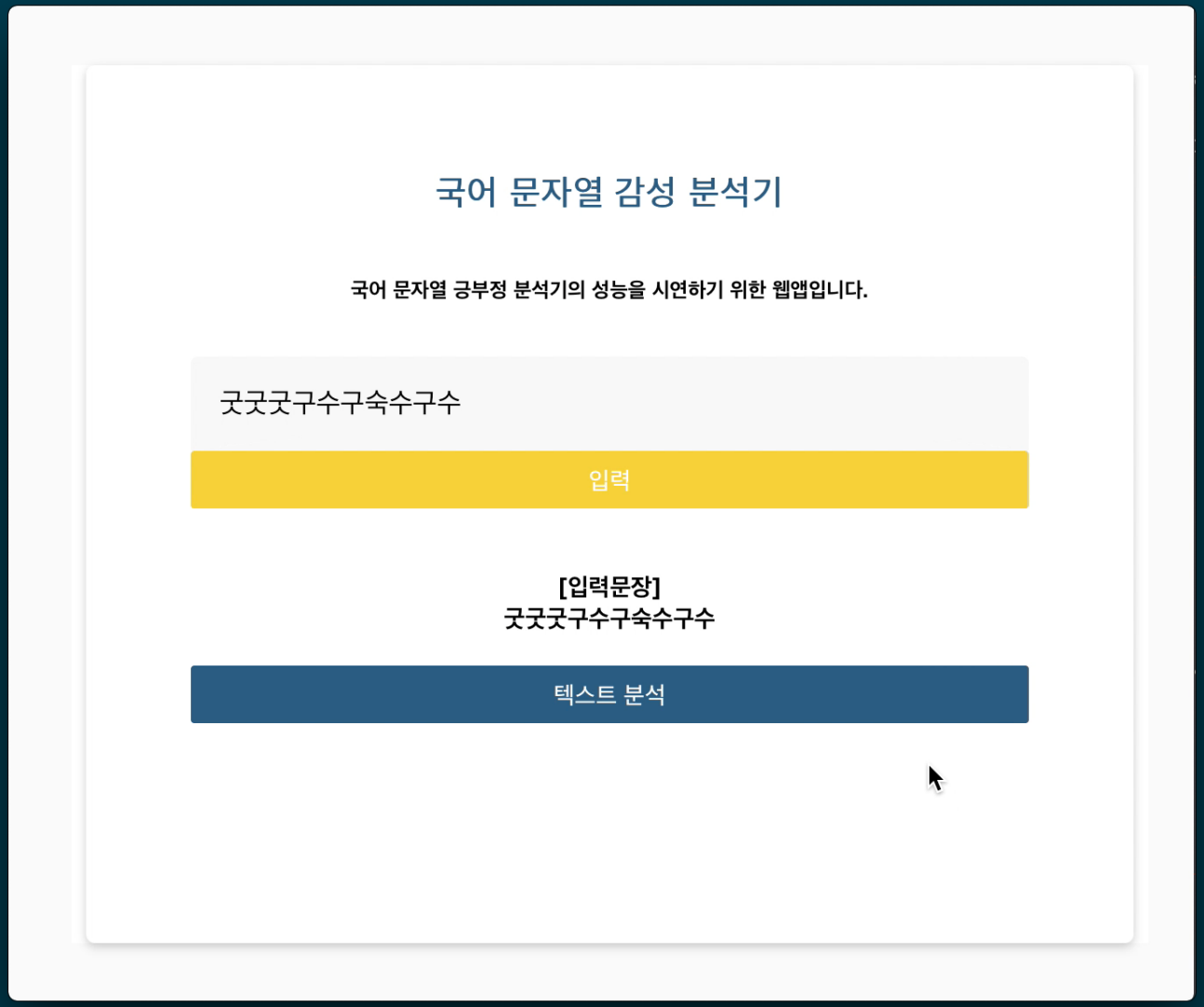
데모 웹앱 실행 화면

메인 페이지



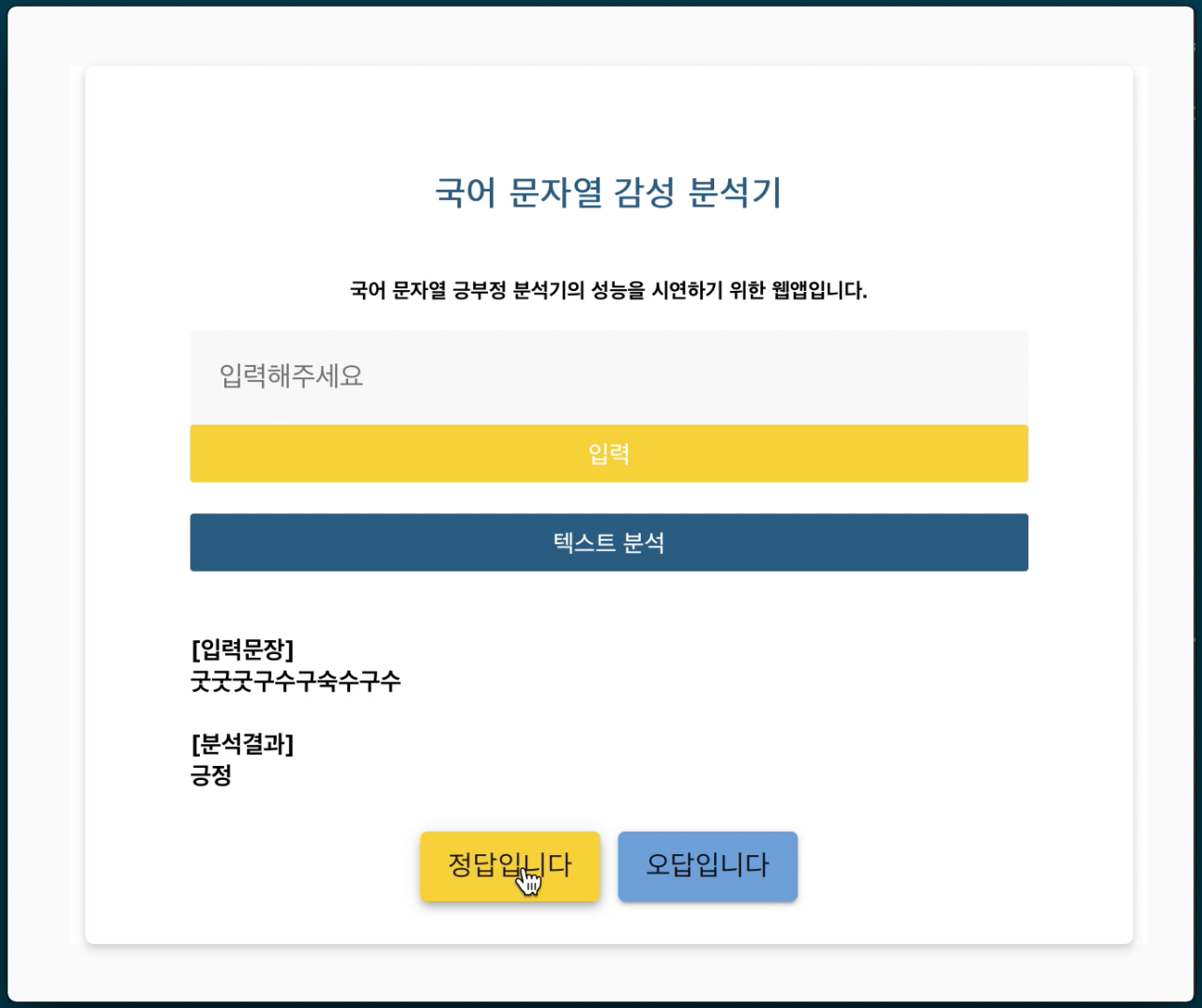
데모 웹앱 실행 화면

문자열 입력



데모 웹앱 실행 화면

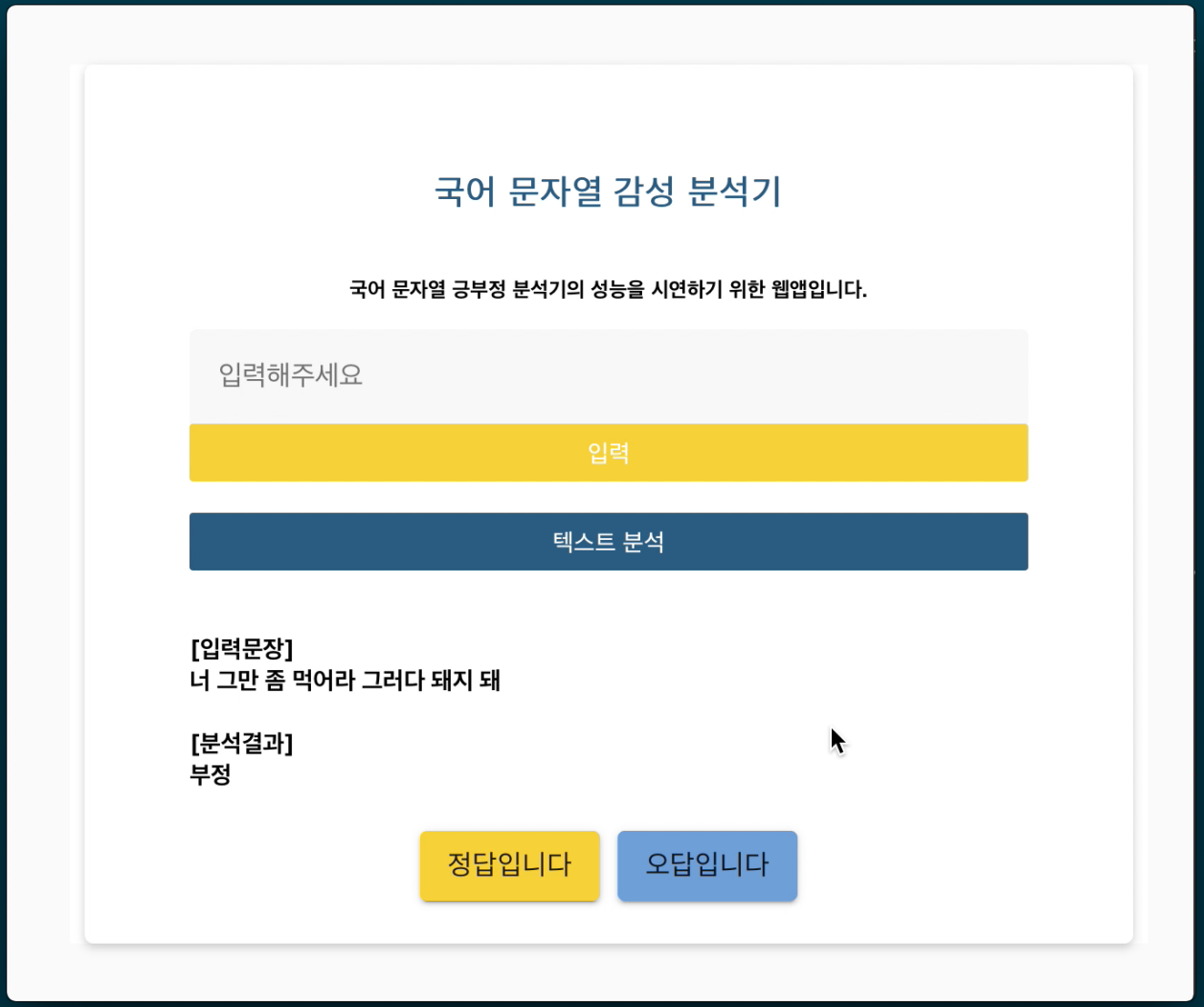
분석 결과 페이지 (긍정 결과)



알파쿠

데모 웹앱 실행 화면

분석 결과 페이지 (부정 결과)



알파쿠

데모 웹앱 데모 시나리오

데모 시나리오

- 메인 페이지 요청
- 텍스트 박스에 문자열 입력
- 입력한 문자열 분석 요청
- 분석 결과 확인 및 피드백
    - 문자열 분석이 맞았을 때 -> 정답 클릭 -> DB 에 그대로 저장
    - 문자열 분석이 틀렸을 때 -> 오답 클릭 -> DB 에 라벨을 바꿔서 저장

데이터 베이스 입력 확인

```
In [3]: for r in Review.objects.all():
   ...:     print(r)
   ...:
라벨: 0, 문자열: 너 빨리 틀려 이제 안 그럼 내가 전원 꺼버린다
라벨: 0, 문자열: 영화 내용은 좋은데, 연기가 영...
라벨: 1, 문자열: 니까짓게 이런 문장도 해독할 수 있을까
라벨: 0, 문자열: 니까짓게이런문장도해독할쑤있을까
라벨: 0, 문자열: 니까짓게 이런 문장도 해독할 수 있을까?
라벨: 0, 문자열: 너 그만 좀 먹어 너 그러다 돼지 돼
라벨: 1, 문자열: 안녕? 나는 민수라고 해 친하게 지내자!!
라벨: 1, 문자열: 굿굿굿구수구숙수구수
라벨: 0, 문자열: 너 그만 좀 먹어라 그러다 돼지 돼
라벨: 1, 문자열: 안녕? 나는 수민이라고 해! 친하게 지내자
라벨: 0, 문자열: 하하 니까짓게 이런 문장도 해석할 수 있을까?
라벨: 0, 문자열: 오 저런 틀리는 것도 보여줘야 하는데... 큰일이네

In [4]:
```