



# 누구인가? 누가 내 험담을 하였는가!

#악플수집 #악플 #멈춰

NLP-12조

AI-it



boostcamp aitech

# 목차

1. 팀 소개
2. 서비스 기획 의도
3. Dataset 및 EDA
4. 모델링
5. 모델 최적화
6. 서비스 시스템 구조

# 1. 팀 소개

# 1. 팀 소개



- Project Management  
- Service Dataset Crawling  
- Front-end & Back-end



- EDA  
- Service Dataset Crawling



- Modeling & Model Optimization  
- Service Architecture Design  
- Application Cloud Release (GKE)



- Front-end (Streamlit)  
- Back-end (FastAPI)  
- MongoDB  
- EDA



- EDA  
- Modeling



- EDA  
- Baseline Coding  
- Modeling & Model Optimization

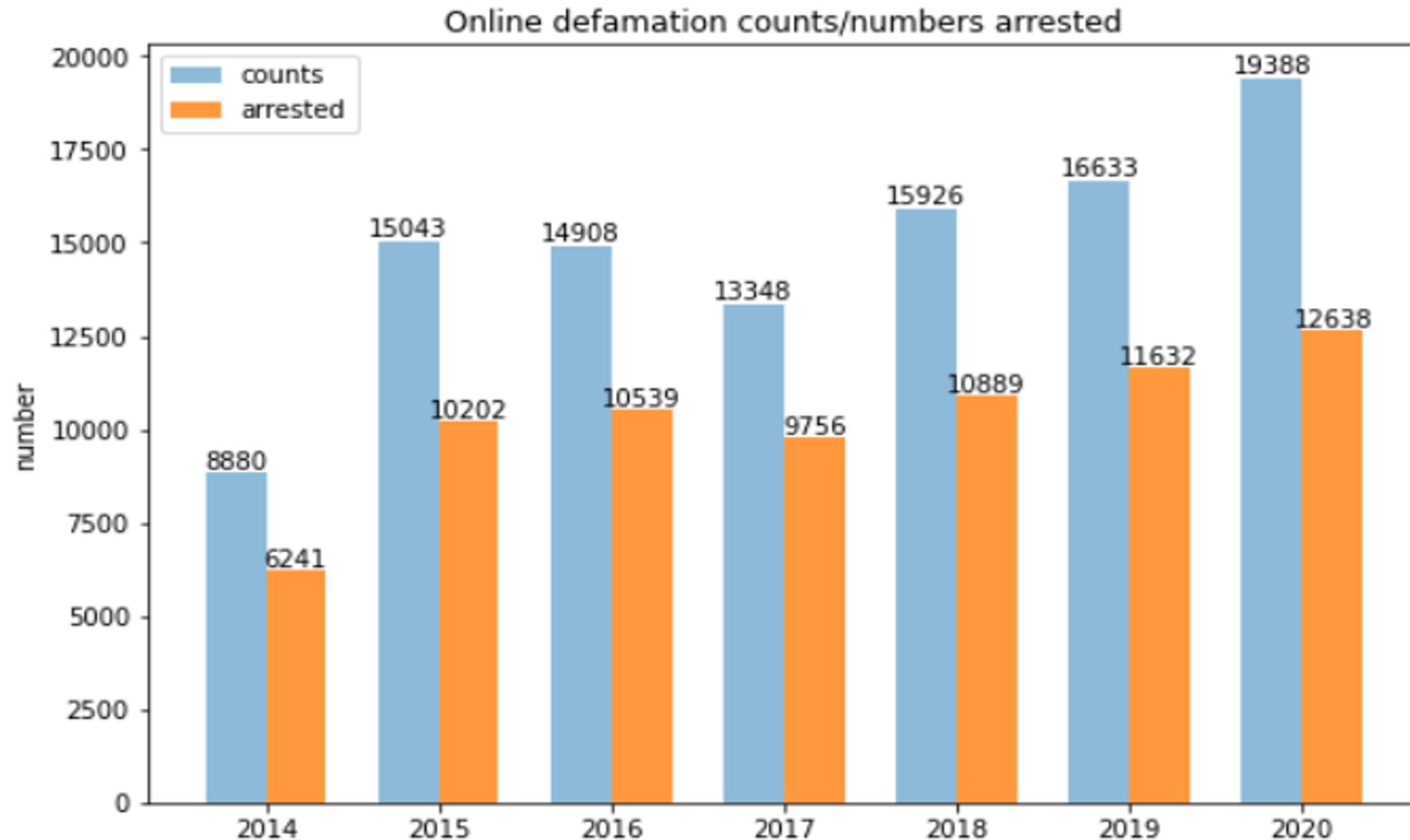


- EDA  
- Baseline Coding  
- Modeling & Model Optimization  
- AutoML (NLP)

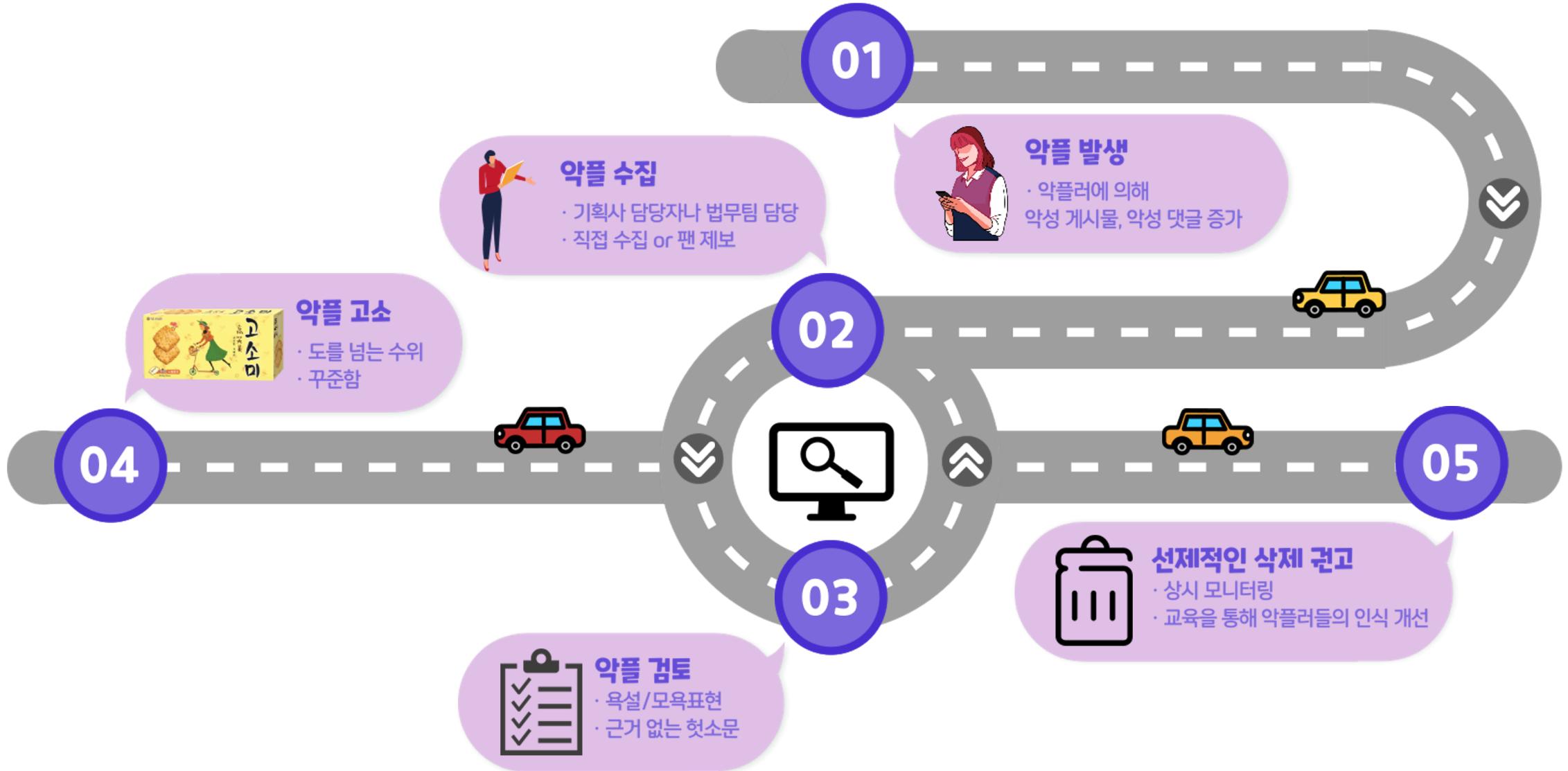
## 2. 서비스 기획 의도

## 2-1. 악성 댓글 범죄 발생 현황

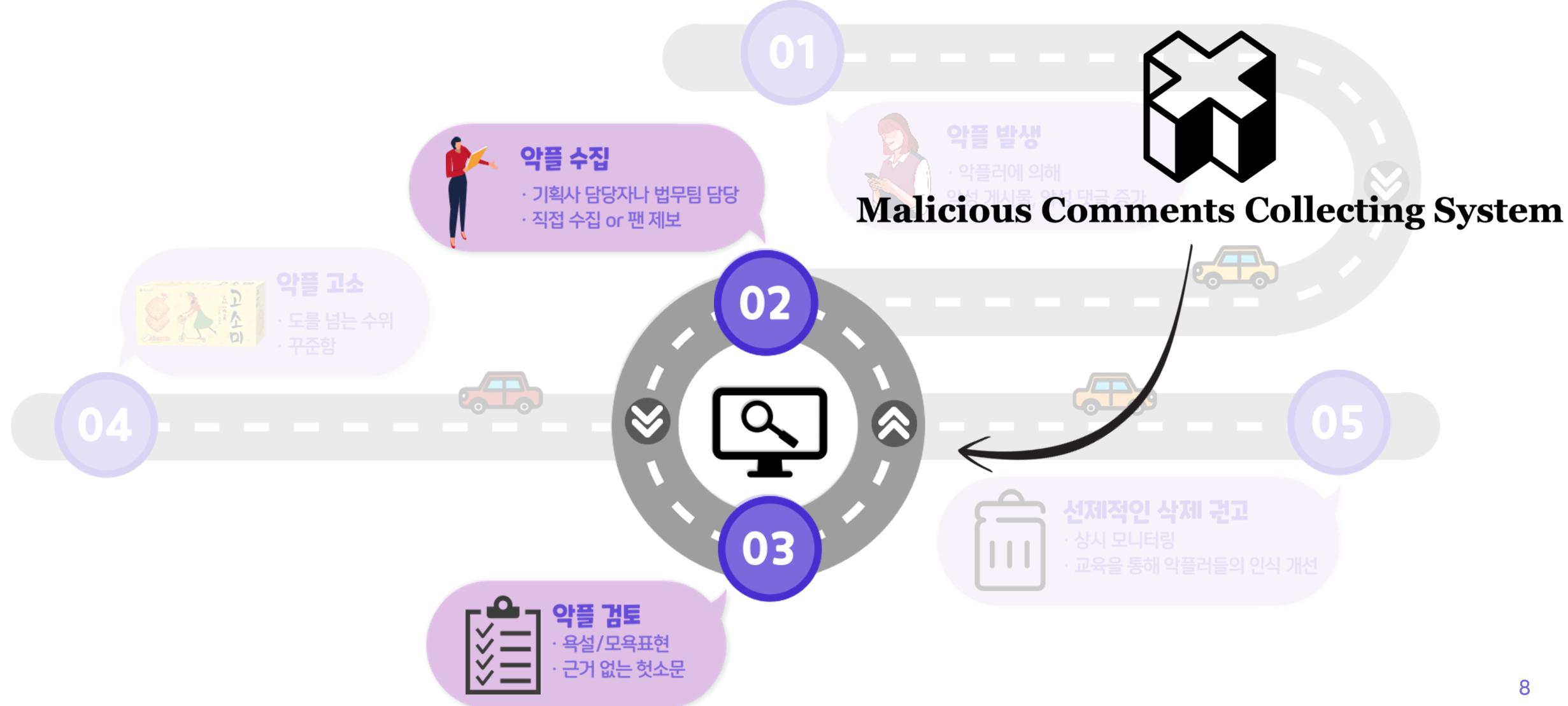
(출처: 경찰청)



## 2-2. 기존 악플 대응 프로세스

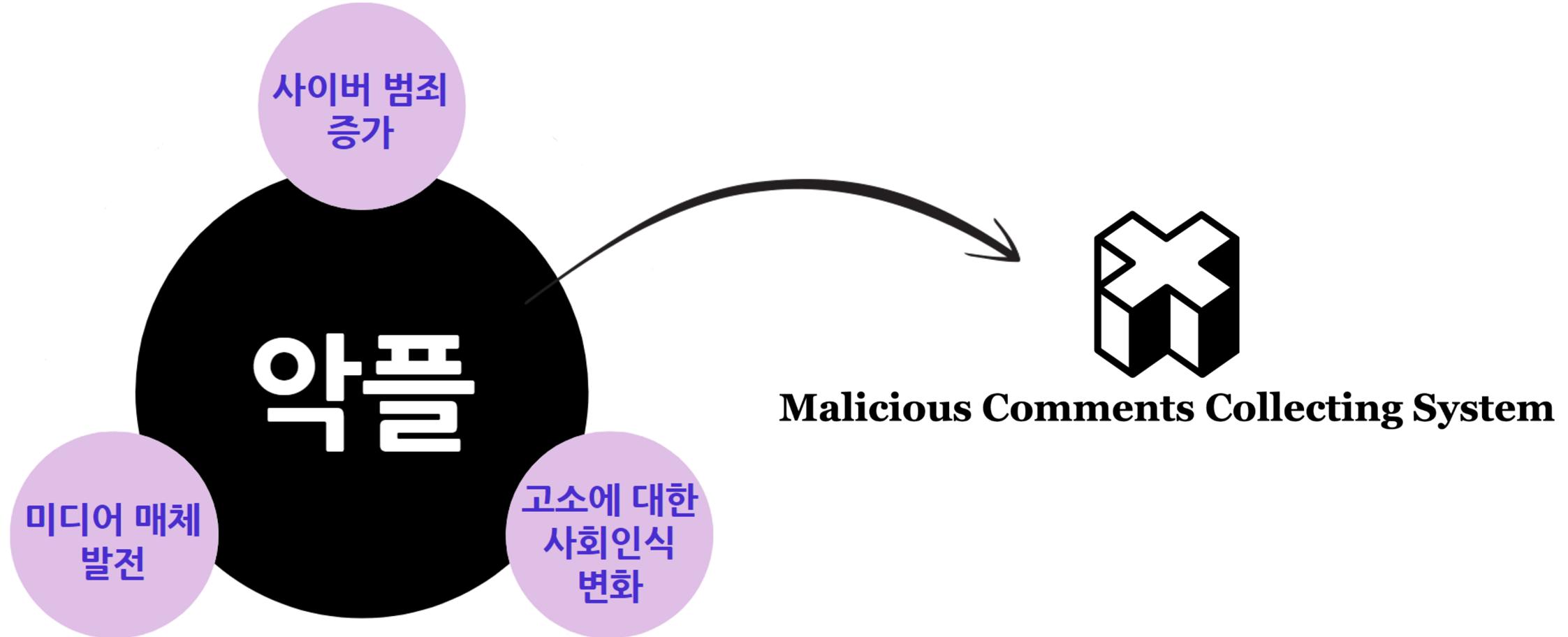


## 2-2. 기존 악플 대응 프로세스



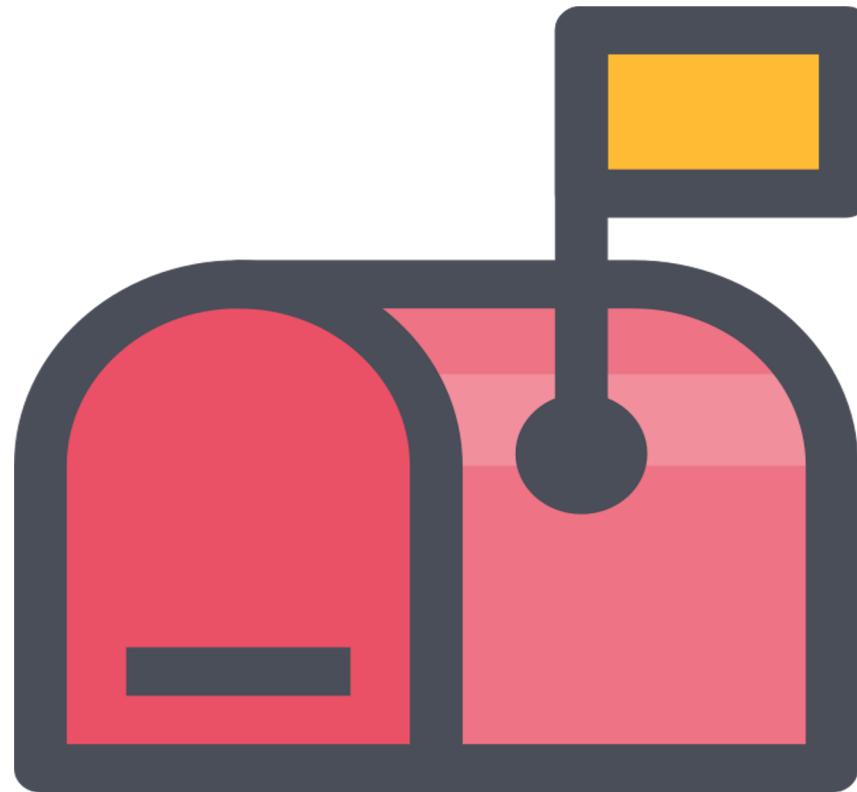
## 2-3. 서비스 소개

---



## 2-4. 설문조사

---



## 2-4. 설문조사

---



안녕하세요 :)

인공지능 기술 기반으로 "근거 없는 비인격적 악성 댓글을 수집해주는 서비스"를 제작 중인 연구팀입니다.

현재, 목표로 하는 해당 서비스의 이용 대상은 셀럽, 인터넷 방송 <sup>b)</sup> 혹은 그와 관계된 소속사 직원입니다.

서비스 타당성 조사를 위해 엔터테인먼트 관계자분들의 의견 및 조언을 듣고자 간단한 설문조사를 진행하고 있습니다.

당사자 혹은 관계자분들께 아래 링크를 전달하고, 짧은 설문을 진행해주시면 저희가 보다 나은 서비스를 제작하는데 많은 도움이 될 것입니다.

서비스의 기능적 목적은 "악성 댓글"로 인한 정신적 피 입증하는 근거자료를 제공하는 것이고,

궁극적으로는 익명성에 기댄 "악성 댓글 문화"를 "근절"하는데 있습니다.

저희 연구팀의 취지에 함께 해주신다면 진심으로 감사하겠습니다.

- 설문조사 링크 : <https://forms.gle/26pQJsgcA6vKzzXC7>

- 연구팀 소개 링크 : [MCCS-Malicious-Comments-Collection-System](#)



## 2-4. 설문조사

여러 배우들을 봤을 때  
카더라 혹은 대수롭지않게 악플을 방치했  
다가역이지도 않은 일에 휘말리는 경우를  
종종 볼 수 있었다.

그리고 정신상태만 건강하다면 누군가의  
악플을 비판으로 보고 나아갈 수도 있다  
고 생각한다.

악플 멈춰!

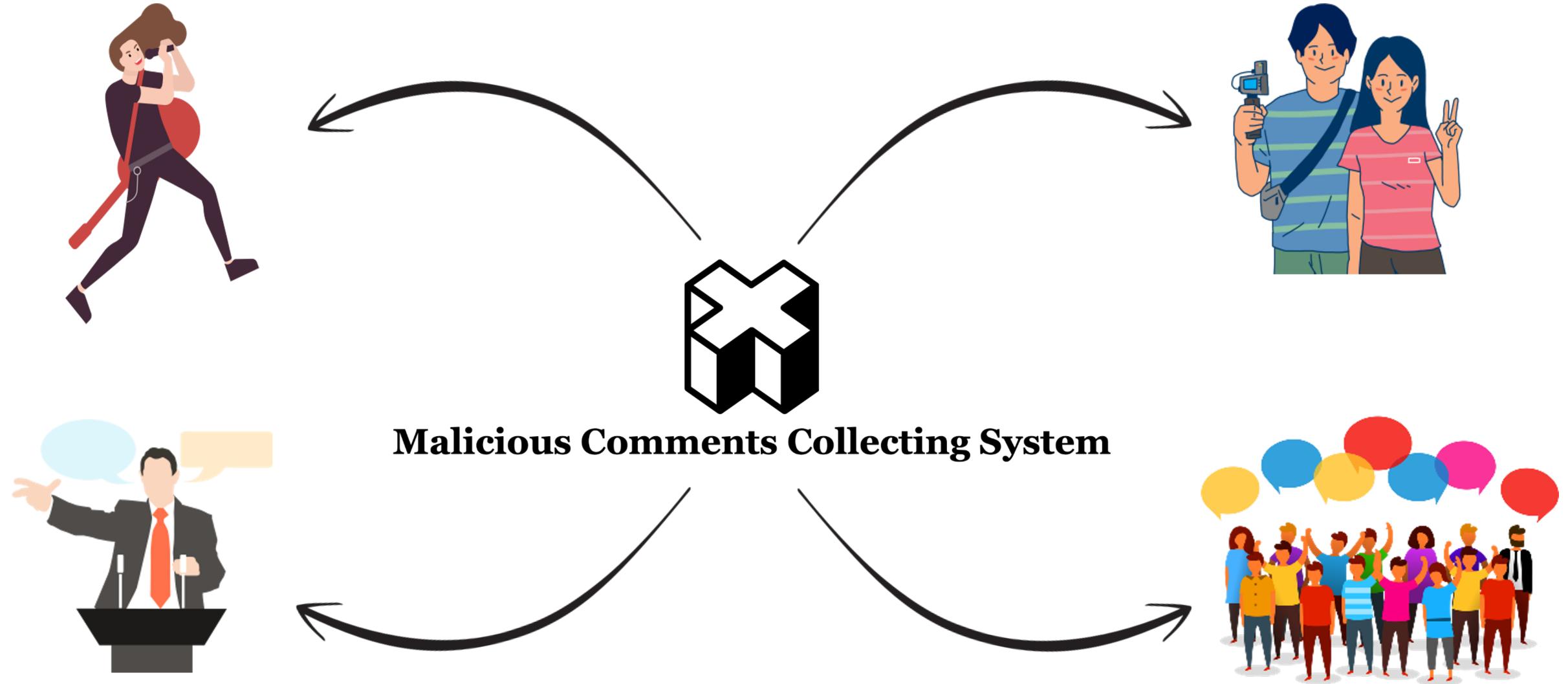
악플은 맴찢 ㅠ

마음의 상처를  
치유하고 싶어요

악플은  
초장에 잡아야함

악플때문에 피해를 보고  
정신건강에 힘들어하는  
동료들이 많습니다.  
부디 개선되길 바랍니다

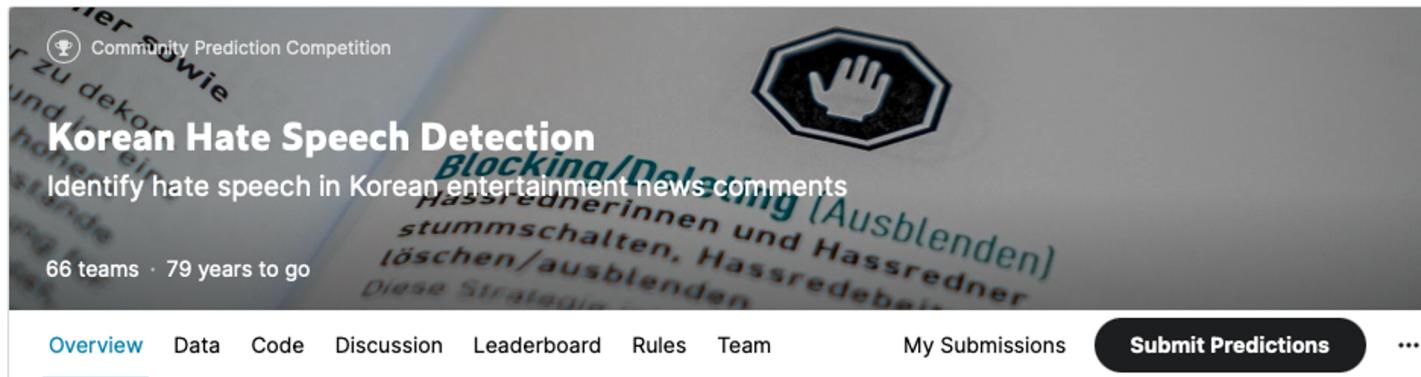
## 2-5. 시장수요



### 3. Dataset 및 EDA

## 3-1. Dataset과 Metric

### Kaggle Competition 참가 - Korean Hate Speech Detection



### 신뢰성 있는 Dataset

[Dataset에 대한 논문](#) SocialNLP@ACL에 등재

[Submitted on 26 May 2020]

### BEEP! Korean Corpus of Online News Comments for Toxic Speech Detection

### 본 서비스와 Kaggle Competition의 목표 유사

- 서비스 목표: 각 댓글에 대해 악플 여부 판별 후 악플 수집
- Competition Leaderboard: 각 댓글에 대해 hate, offensive, none 예측

### Metric : F1 score

#### Metric

The evaluation metric for this competition is [Macro F1-Score](#). The F1 score, commonly used in information retrieval, measures accuracy using the statistics precision  $p$  and recall  $r$ . Precision is the ratio of true positives ( $tp$ ) to all predicted positives ( $tp + fp$ ). Recall is the ratio of true positives to all actual positives ( $tp + fn$ ). The F1 score is given by:

$$F1 = 2 \frac{p \cdot r}{p + r} \text{ where } p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}$$

The F1 metric weights recall and precision equally, and a good retrieval algorithm will maximize both precision and recall simultaneously. Thus, moderately good performance on both will be favored over extremely good performance on one and poor performance on the other.

#### 참고자료

[Kaggle korean-hate-speech](#)

## 3-2. EDA: 가이드라인 분석

### 1. 성별/성적 지향성/성 정체성에 대한 편견

댓글 내용에 1) 성 역할 혹은 성별 2) 성적 지향성 혹은 성 정체성 3) 이 밖의 모든 성에 대한 편견을 가진 내용이 있는지 확인합니다.

- 성 관련 편견이 존재함 (아)

- 성별에 따른 역할이나 능력에 대한 편견

- 예시: 남자는 능력이고 여자는 외모지 / 가정을 지키고싶으면 요리배우고 살빼야할듯. / 여자는 집에 서 살피는게 죄고, 남자가 무잔가? / 이 외모면 회사해도된다 / 성폭행 몇번 했다고 이런 난리를 치면 오징어들은 뭐하고 사냐. 평생 휴지랑 함께하는 독거연성들이네. ㄷ ㄷ

- 성별과 나이에 대한 편견

- 예시: 여자는 어린 여자가 갑이지 / 남자가 나이많은 여자를 만날 이유가있나..? 1. 돈줄 2. 성욕해소 달고는.. 글자 결혼은 어린여자랑 하는게 맞지

- 그 외에 특정 성별, 성적 지향성, 성 정체성, 성에 관련된 사상을 가진 집단에 대한 편견

- 예시: 성전환자가 여자니? / 모래도살았네 게이가 / 메미는 레즈비언이 선돌한다던데? / 레즈비언들이 좋아하는만한... / 여자랑 여자랑 가능할? / 성 정체성을 잊어가는 병자들이 많은 시대네...병은 고쳐 야지 자랑이라고 떠들어 대나? / 테이블 쿠이 도나온다

- 그 외의 편견이 존재함 (△)

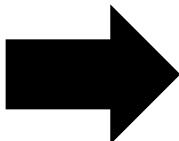
성별/성적 지향성/성 정체성 외에 인종이나 출신 지역, 국가 또는 민족, 정치색, 피부색, 종교, 장애, 나이, 키, 군복무 여부, 재산, 직업 등 개인의 특성을 성급하게 집단의 특성으로 확장시키고, 그 집단의 특성에 대한 편견이 드러난 경우에 해당합니다. 댓글의 내용에 "대상의 <인종이나 출신 지역, 국가 또는 민족, 정치색, 피부색, 종교, 장애, 나이, 키, 군복무 여부, 재산, 직업 등> 이 이러이러하므로 아마 이러이러할 것이다" 는 뉴앙스가 담겨있는 경우에 해당합니다. 그 예시를 몇 가지 보면 다음과 같습니다.

- 나이에 대한 편견

- 예시: 나이에 걸맞게 서 놀이라, 이제 미혼이 넘었는데 언지하지나 귀여운 책 할려 쭈

- 출신 지역 및 국가에 대한 편견

- 예시: 스테이크에 와인한잔하면서 가족끼리 희의했겠지 상도같 라도는 인생살면서 걸려야한다 / 댓글 알바풀었나 티아라 평체알바 ㅋ



**💡 hate** : 부정적인 정서를 함유한 댓글 중에서 대상에 대해 근거없이 **비난**하거나 **깎아내리는** 경우, 대상이 **모욕감 혹은 수치감**을 느낄 수 있는 발언

**offensive** : 대상을 일정한 특성에 근거해서 설불리 판단한 후 대상에 대해 **적대감**을 드러내는 발언; 표현의 대상에게 **정신적인 고통**과 같은 감정 상태를 야기하는 경우

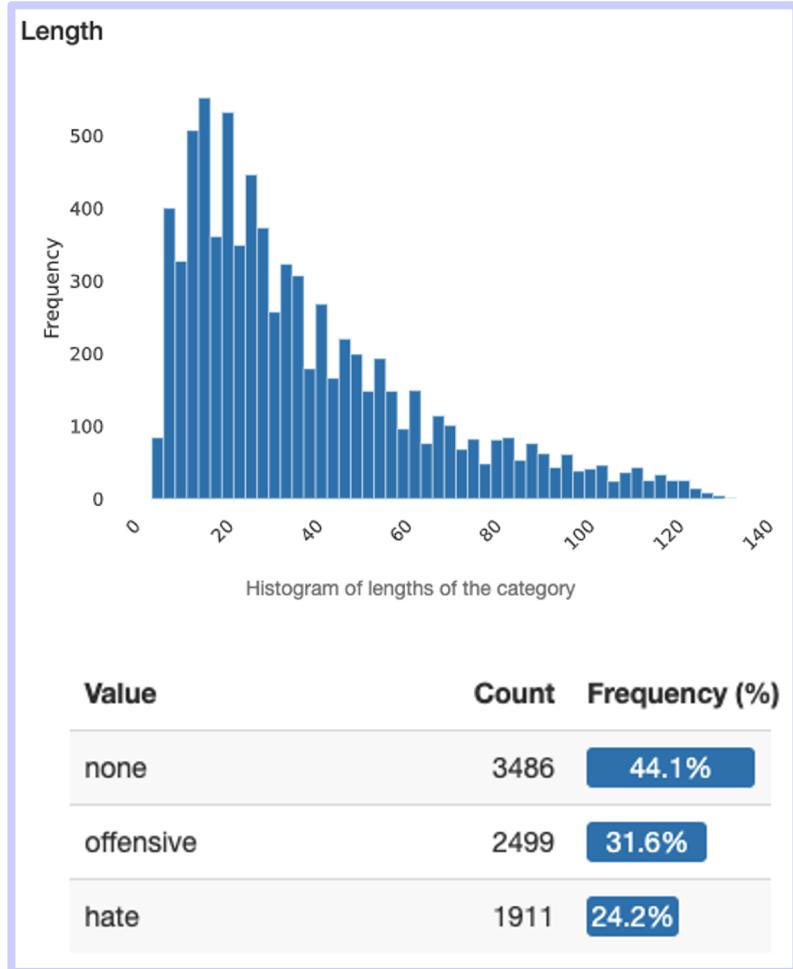
합리적인 비판은 해당되지 않으며, 단순히 욕설이 등장한다고 해서 모욕이나 혐오가 느껴지는 것은 아닐 수 있으니 주의

**💡** 댓글의 내용이 미완성이거나 불분명하여 이해되지 않는 경우, 댓글의 내용만으로는 판단되지 않는 경우는 넘어가는 것을 용인합니다. 또한 한글이 아닌 경우에는 넘어가주세요.

Data Annotation 가이드라인 파악

각 데이터들의 labeling 기준 분석으로 EDA 방향 수립

## 3-2. EDA: 통계적 EDA



**초성 및 기호 반복 문장**

Aa 구분	# 갯수	특이사항
ㅋㅋ 연속 2개 이상	764	전체 약 10%
?? 연속 2개 이상	936	
!! 연속 2개 이상	236	
. 포함된 문장	1740	

## 3-2. EDA: 데이터 뜯어보기

### 띄어쓰기 및 맞춤법 문제

py-hanspell을 이용, 7896개 문장에서 19844개 맞춤법/띄어쓰기 error 파악



'180넘어갈정도의 기력지는 아님78에서79180넘는애들은 의심 의구심을 안하고안듬그래도 비율좋아 커보이긴  
함',  
'19에서15로 바꿔고 망트리 탄거',  
'근데 불친절한알바들보면 때리고싶긴하던데',

### Unknown\_token 확인 by beomi tokenizer

사용하는 tokenizer의 vocab에 모두 들어가 [UNK] 토큰

### 이모티콘

●, ★, 💕, ❤️, ♥️, ♡, !?, 😊, ✓ ⇒ 96문장

^^, , --, ;, ( °᷄°) ᶥ, ^-^, -, -, π.π ⇒ 306문장

### 욕설 관련

' 10+8 진짜 이승기랑 비교된다',  
'(현재 호텔주인 심정) 아 18 난 마른하늘에 날벼락맞고 호텔망하게생겼는데 누군 계속 추모받  
네....',  
가만히 있던 스엠 머리채 잡지마 ㅅ1발 언플 안해줘서 지금 다들 개빡쳐 있으니깐느느  
' 걔oooo새oooo끼 가 19살? 장난하는것도아니구. 먹는 음식 가지고 장난좀 하지 말자.',  
'명치 브이넥 실화냐고ㅋㅋㅋ 개토쏠리네 씩5발',

' ㅈㄹ 염병..ㅋㅋ재산이 1조인데 천억안내고 머리굴리다가 꼬시고 오지다',  
공산당 교육밑에서 자랐으니 공산당 말쓰지...한국말 쓰겠니... ㄷ신 들...너나 잘하세요..,

### URL & 기타

'블로그도 있던데... 요기... 다른 사람 블로그에 올린 후기에 댓글도 다시던데...  
<https://blog.naver.com/myungi7>'

comment : 김홍국— 1959–2018, label : hate ⇒ 김홍국이 2018에 죽었다는 의미인가..?

## 3-2. EDA: 데이터 뜯어보기

### 띄어쓰기 및 맞춤법 문제

py-hanspell을 이용, 7896개 문장에서 19844개 맞춤법/띄어쓰기 error 파악



'180넘어갈정도의 기력지는 아님78에서79180넘는애들은 의심 의구심을 안하고안듬그래도 비율좋아 커보이긴 함',  
'19에서15로 바뀌고 망트리 탄거',  
'근데 불친절한알바들보면 때리고싶긴하던데',

### Unknown\_token 확인 by beomi tokenizer

사용하는 tokenizer의 vocab에 모두 들어가 [UNK] 토큰

### 이모티콘

●, ★, 💕, ❤️, ♥️, ♡, !?, 😊, ✓ ⇒ 96문장

^^, , --, ;, ( °᷄° ) ᶥ, ^-^, -, -, π.π ⇒ 306문장

### 욕설 관련

' 10+8 진짜 이승기랑 비교된다',

'(현재 호텔주인 심정) 아 18 난 마른하늘에 날벼락맞고 호텔망하게생겼는데 누군 계속 추모반

### 패키지 적용 후

180 넘어갈 정도의 기력지는 아님 78에서 79180 넘는 애들은 의심 의 구심을 안하고 안듬그래도 비  
율 좋아 커 보이긴 함,  
19에서 15로 바뀌고 망트리 탄 거,

맞춤법이 틀리거나 고유명사(이름 등)에 대한 룰이  
명확하지 않아 가지고 있는 데이터에 대해 성능이  
좋지 못함

'블로그도 있던데... 요기... 다른 사람 블로그에 올린 후기에 댓글도 다시던데...'

<https://blog.naver.com/myungi7>

comment : 김홍국— 1959–2018, label : hate ⇒ 김홍국이 2018에 죽었다는 의미인가..?

### 3-3. Preprocessing

#### 1. 한글 및 영어, 특수문자, 그리고 이모지(😊)까지!

정규표현식을 통해 한글, 영어, 특수문자를 포함해 Emoji까지 학습 대상에 포함했습니다.

한편, 한글 범위를 ㄱ-ㅎ가-힝 으로 지정해 ㄱ-힝 내의 한자를 제외했습니다.

#### 2. 댓글 내 중복 문자열 축약

ㅋㅋㅋㅋㅋ 와 같이 중복된 글자를 ㅋㅋ 와 같은 것으로 합쳤습니다.

#### 3. Cased Model

KcBERT는 영문에 대해서는 대소문자를 유지하는 Cased model입니다.

#### 4. 글자 단위 10글자 이하 제거

10글자 미만의 텍스트는 단일 단어로 이뤄진 경우가 많아 해당 부분을 제외했습니다.

#### 5. 중복 제거

중복적으로 쓰인 댓글을 제거하기 위해 완전히 일치하는 중복 댓글을 하나로 합쳤습니다.

#### 6. 000 제거

네이버 댓글의 경우, 비속어는 자체 필터링을 통해 000 로 표시합니다. 이 부분을 공백으로 제거하였습니다.

```
def preprocess(sents):
    preprocessed_sents = []

    emojis = set()
    for k in emoji.UNICODE_EMOJI.keys():
        emojis.update(emoji.UNICODE_EMOJI[k].keys())

    punc_bracket_pattern = re.compile(r'[\\"\\\[\\]\\\\]')  
base_pattern = re.compile(r'^[^!@#$%^&()\\x00-\\x7F]+[{}]+'.format(emojis))  
url_pattern = re.compile(  
    r'(http|ftp|https)?://((www\.)?[-a-zA-Z0-9@:%._+~#=]{1,256}\.([a-zA-Z0-9()]{1,6}\b([-a-zA-Z0-9()@:%_+~#?&/=]*))')

    for sent in sents:  
        sent = punc_bracket_pattern.sub(' ', sent)  
        sent = base_pattern.sub(' ', sent)  
        sent = url_pattern.sub('', sent)  
        sent = sent.strip()  
        sent = repeat_normalize(sent, num_repeats=2)

    preprocessed_sents.append(sent)

    return preprocessed_sents
```

#### KcELECTRA 의 PLM 학습 전처리 코드 수정 후 사용

- emoji
- bracket

#### 참고자료

[Beomi/KcELECTRA](#)

## 4. 모델링

## 4-1. Backbone Model

---

### KcELECTRA (Korean comments ELECTRA)

- 한국어 Transformer 계열 모델 → 한국어 위키, 뉴스 기사, 책 등 잘 정제된 데이터 기반 모델 (e.g. KoELECTRA)
- User-Generated Noisy Text (e.g. 정제되지 않은 글, 신조어 등) dataset에 학습
  - 뉴스 기사의 댓글과 대댓글 데이터; 약 17.3GB, 1억 8천만개 이상의 문장
- 기존 KcBERT 대비 train data 증가 및 vocab 확장을 통해 상당한 수준으로 성능 향상

	Size (용량)	NSMC (acc)	Naver NER (F1)	PAWS (acc)	KorNLI (acc)	KorSTS (spearman)	Question Pair (acc)	KorQuaD (Dev) (EM/F1)
KcELECTRA-base	475M	<b>91.71</b>	86.90	74.80	81.65	82.65	<b>95.78</b>	70.60 / 90.11
KcBERT-Base	417M	89.62	84.34	66.95	74.85	75.57	93.93	60.25 / 84.39

참고자료

[Monologg/KcELECTRA](#)

## 4-2. Pseudo Labeling with Unlabeled Data

### Unlabeled Data

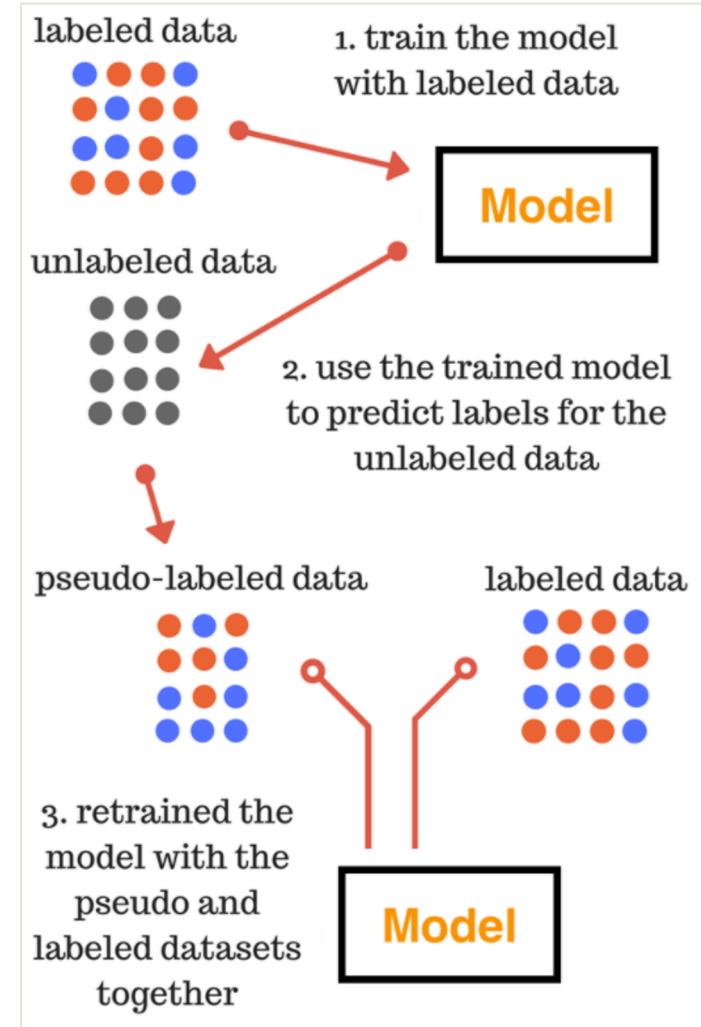
- 약 7K train dataset 은 학습에 부족
- 약 2M unlabeled data 를 train data 로 활용

### Pseudo Labeling

- 5% 의 unlabeled data (약 100K) pseudo labeling 후 model 학습
- [baseline] LB f1-score 61.412 → LB f1-score 62.343
- 성능이 높아진 모델로 추가 pseudo labeling 진행하며 train data 충당

### 참고자료

[korean-hate-speech](#)  
[Pseudo-labeling](#)

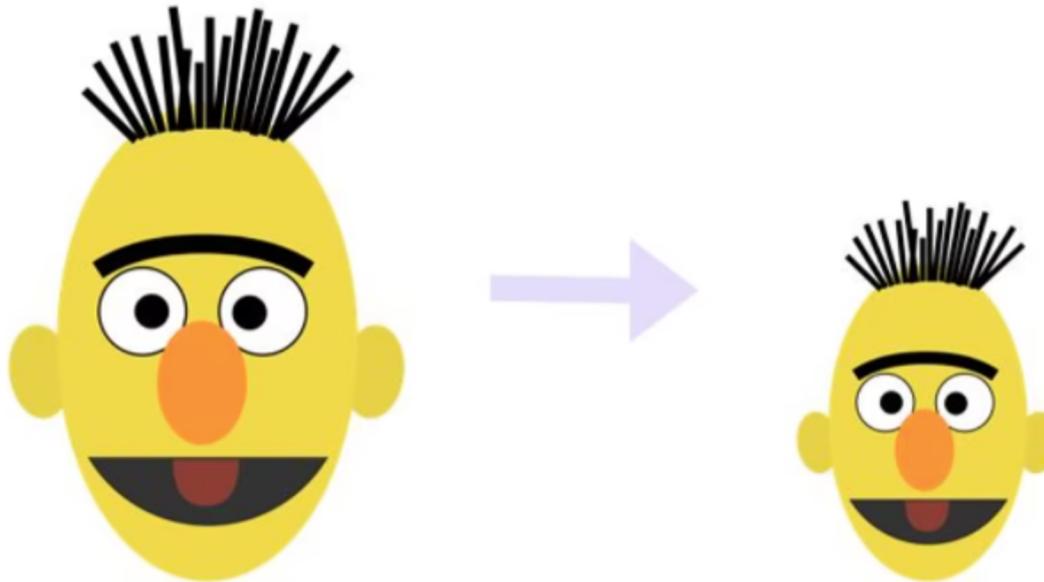


## 4-4. Light-weight Modeling

---

### Performance Limitation

- KcELECTRA 를 활용, 다양한 modeling 시도에도 성능 향상에 한계를 보임
- 다른 transformer backbone model 사용 → 성능 하락
  - KcELECTRA 가 현재 data domain 에 가장 fit 한 model임을 확인
- 주어진 task 와 data 에 비해 model 이 클 가능성 존재 → model size 를 줄이자!



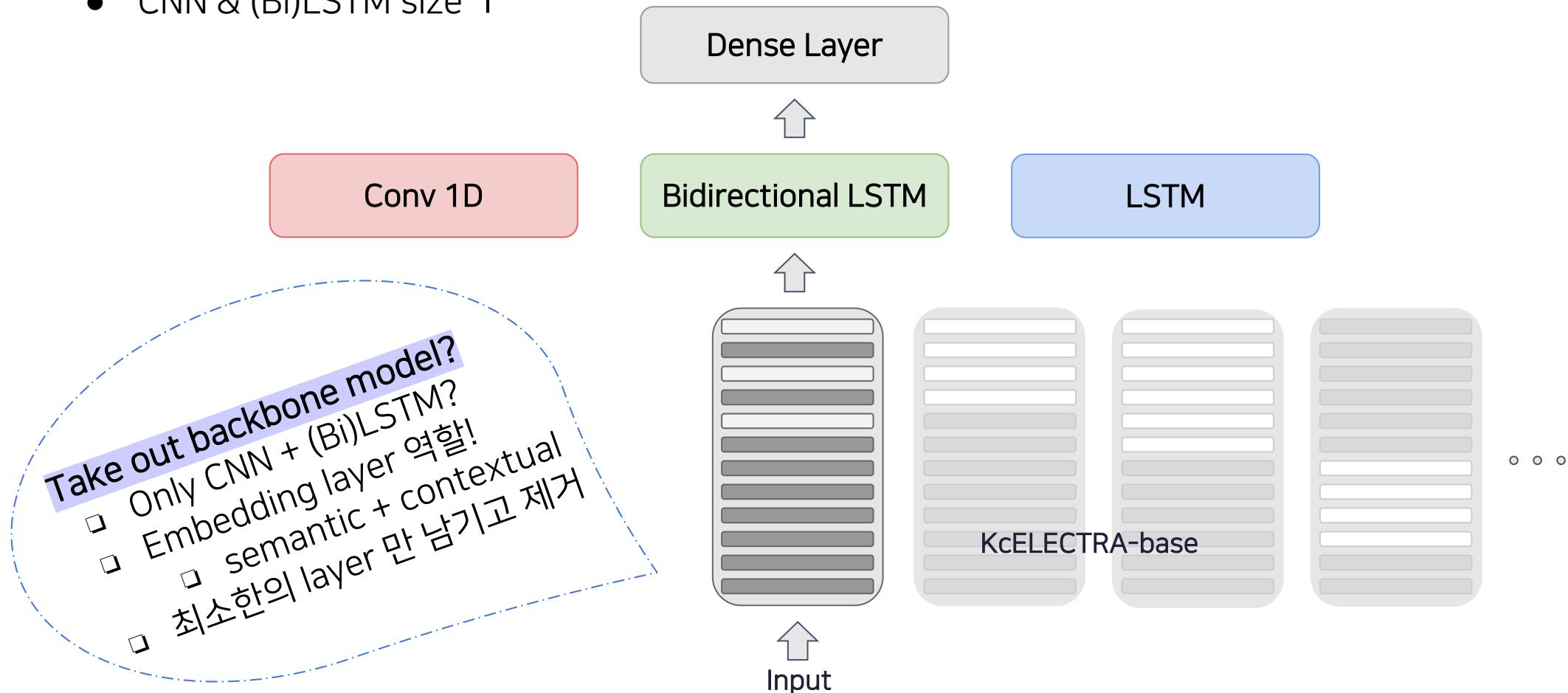
참고자료

[Compressing BERT for faster prediction](#)

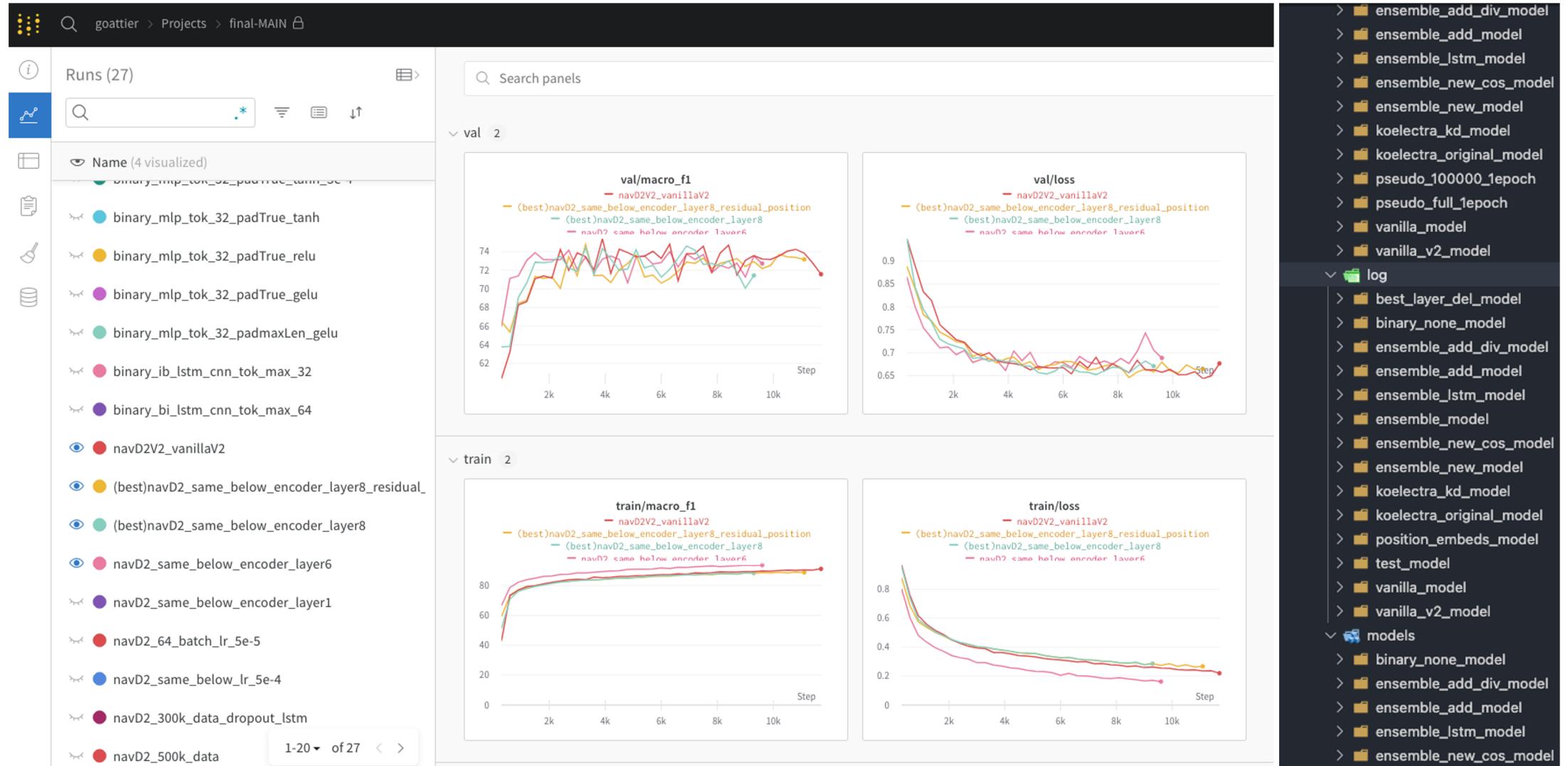
## 4-4. Light-weight Modeling

### KcELECTRA Encoder Layer Deletion + CNN + (Bi)LSTM

- KcELECTRA size ↓
- CNN & (Bi)LSTM size ↑



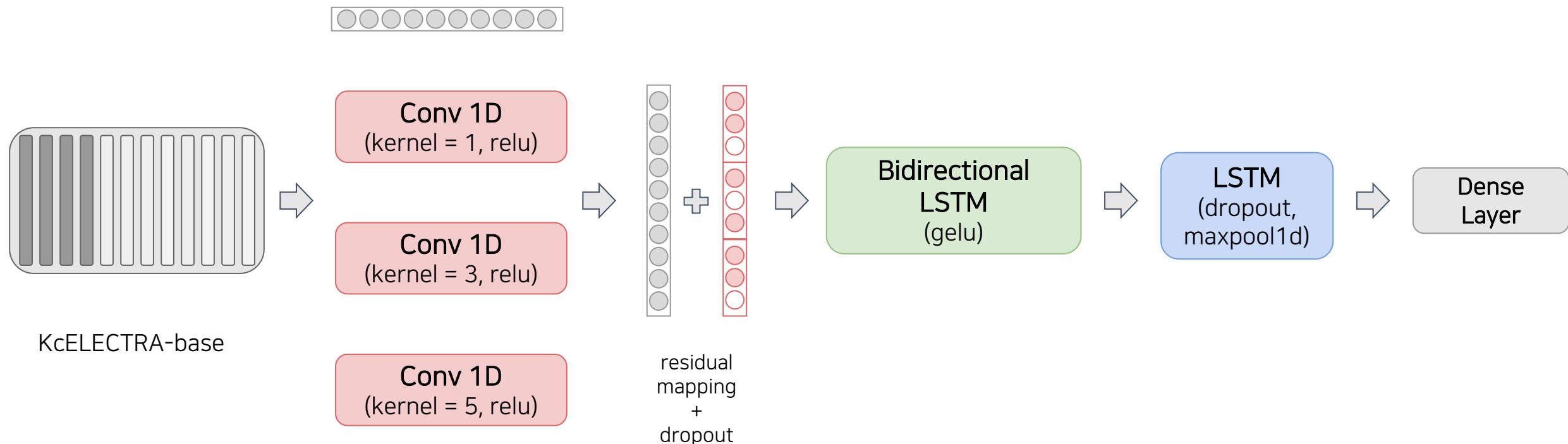
## 4-4. Light-weight Modeling



## 4-4. Light-weight Modeling

### Best Model

- Heuristic & empirical modeling 진행
- [baseline] LB f1-score 61.412 → LB f1-score 64.856



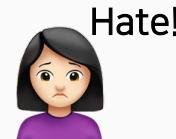
## 4-5. Clustering with Triplet Loss & KNN

### Performance Limitation

- 다양한 layer architecture addition & deletion 실험 → 성능 향상 X
- 다양한 hyperparameter tuning 실험 → 성능 향상 X

### Labeling Bias

“**걍 폐지해라 요즘 무한도전 노잼**”



Hate!



Offensive!



Labeling Bias 를 해결할  
방법이 없을까?

## 4-5. Clustering with Triplet Loss & KNN

### Triplet Loss

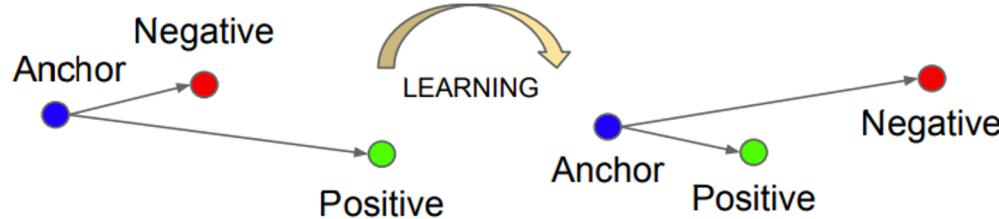
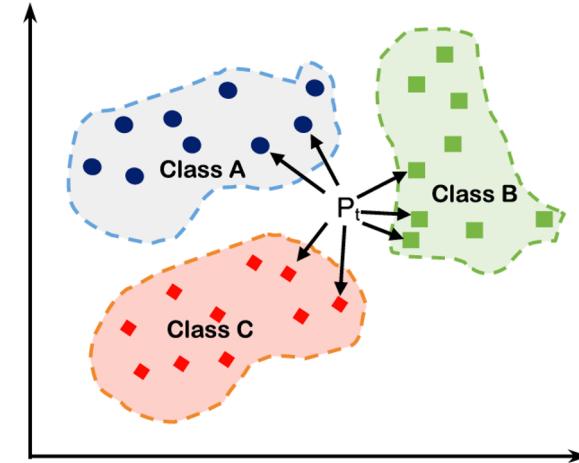


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

### KNN (K-Nearest Neighbors)



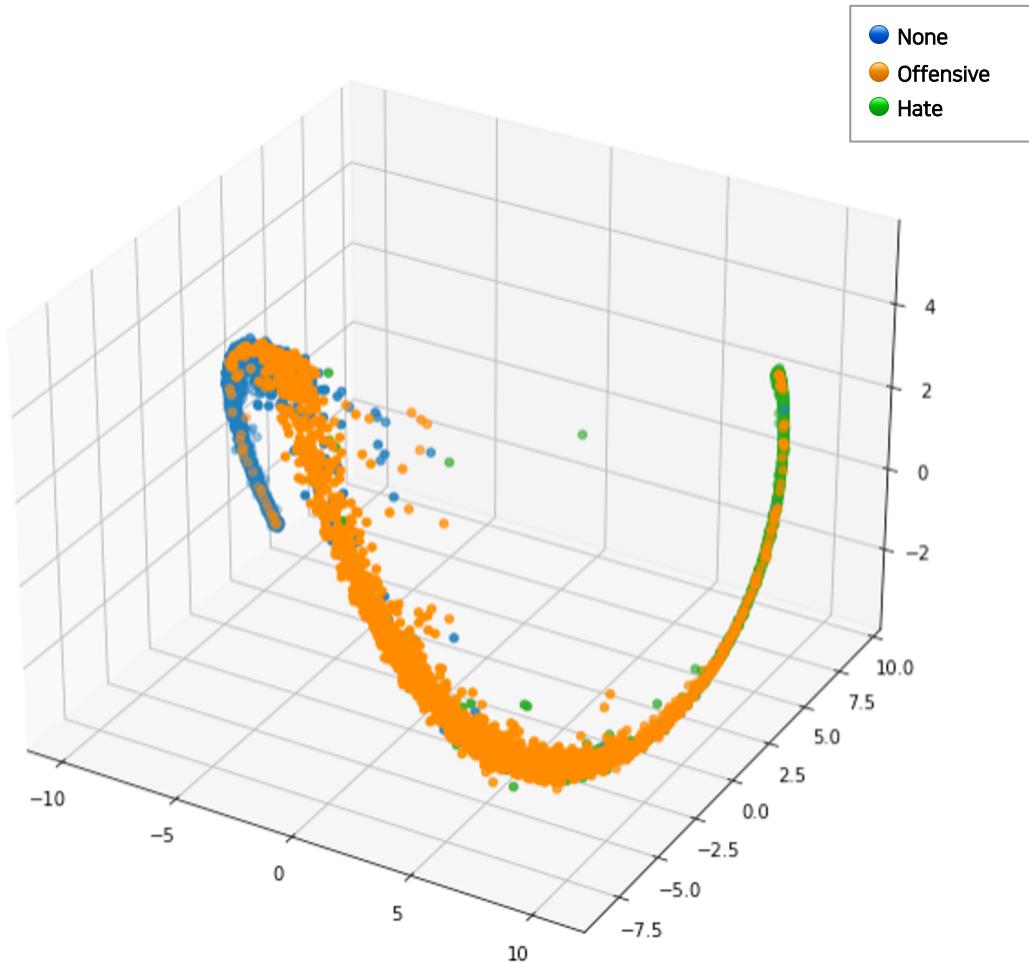
### 참고자료

[\[paper\]FaceNet](#)  
[KNN 정리글](#)

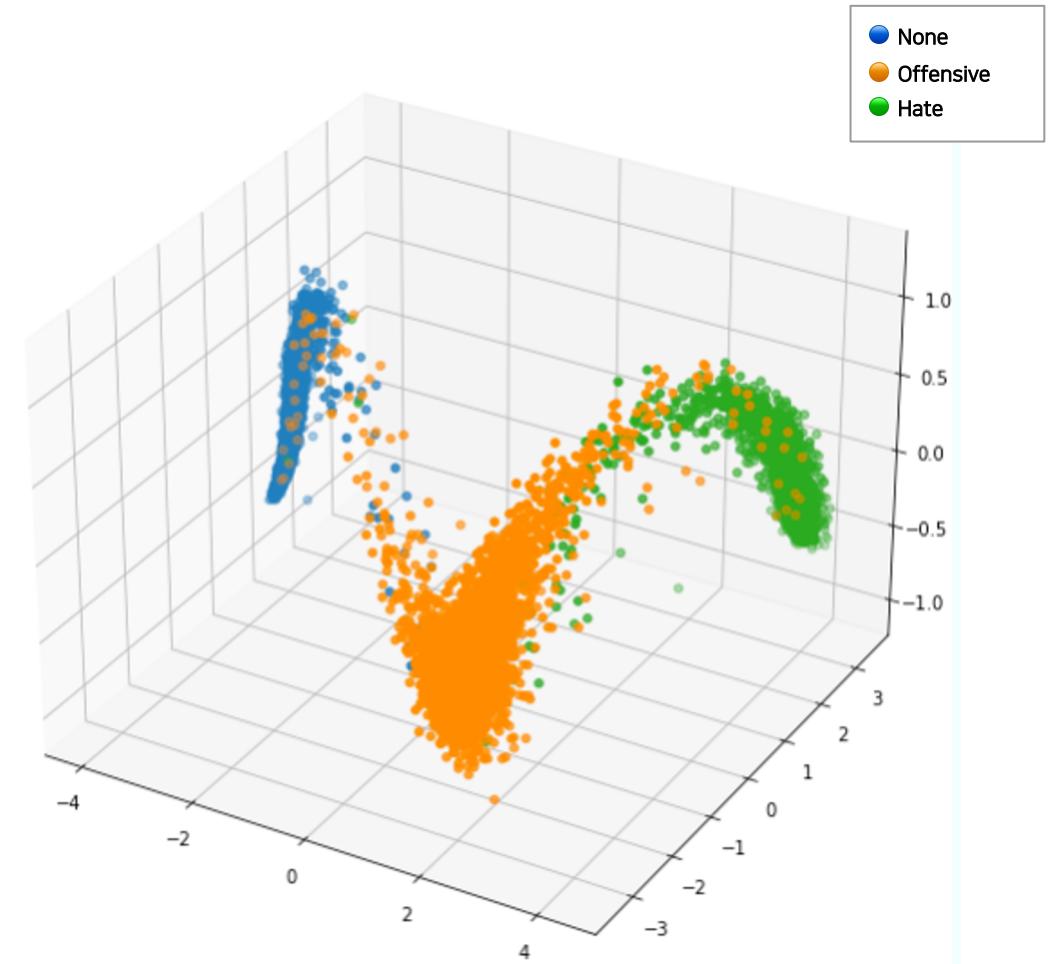
## 4-5. Clustering with Triplet Loss & KNN

---

Before Clustering



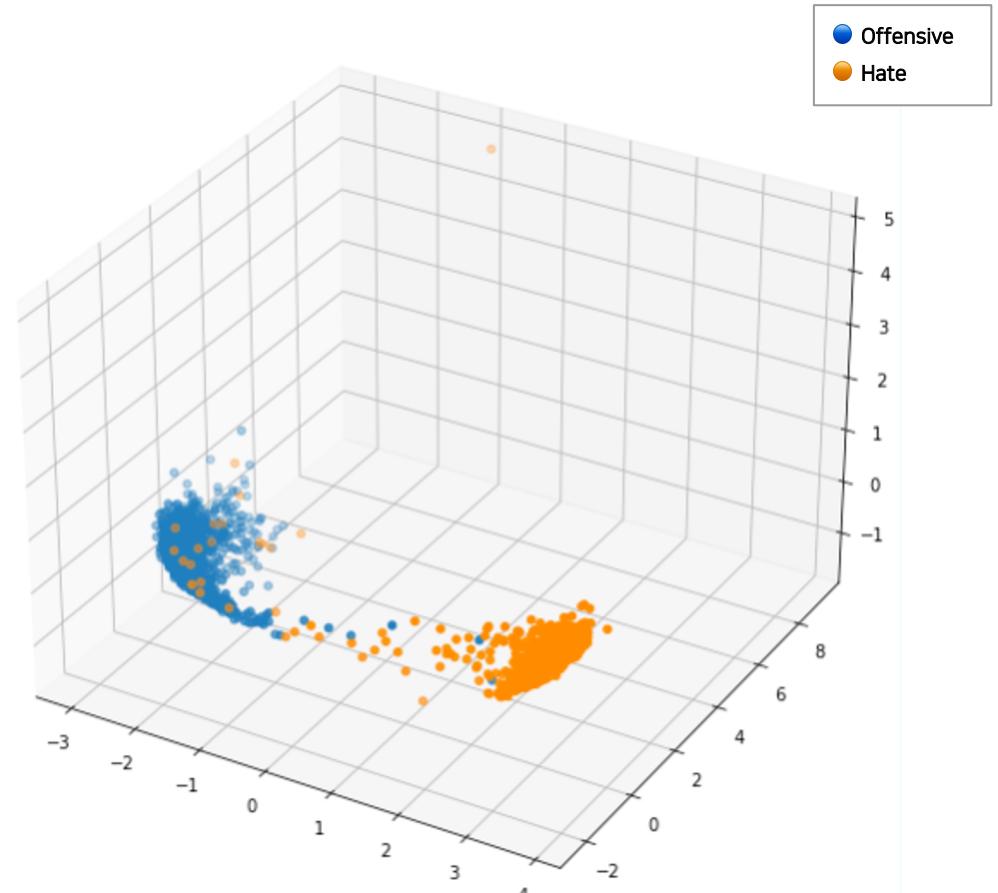
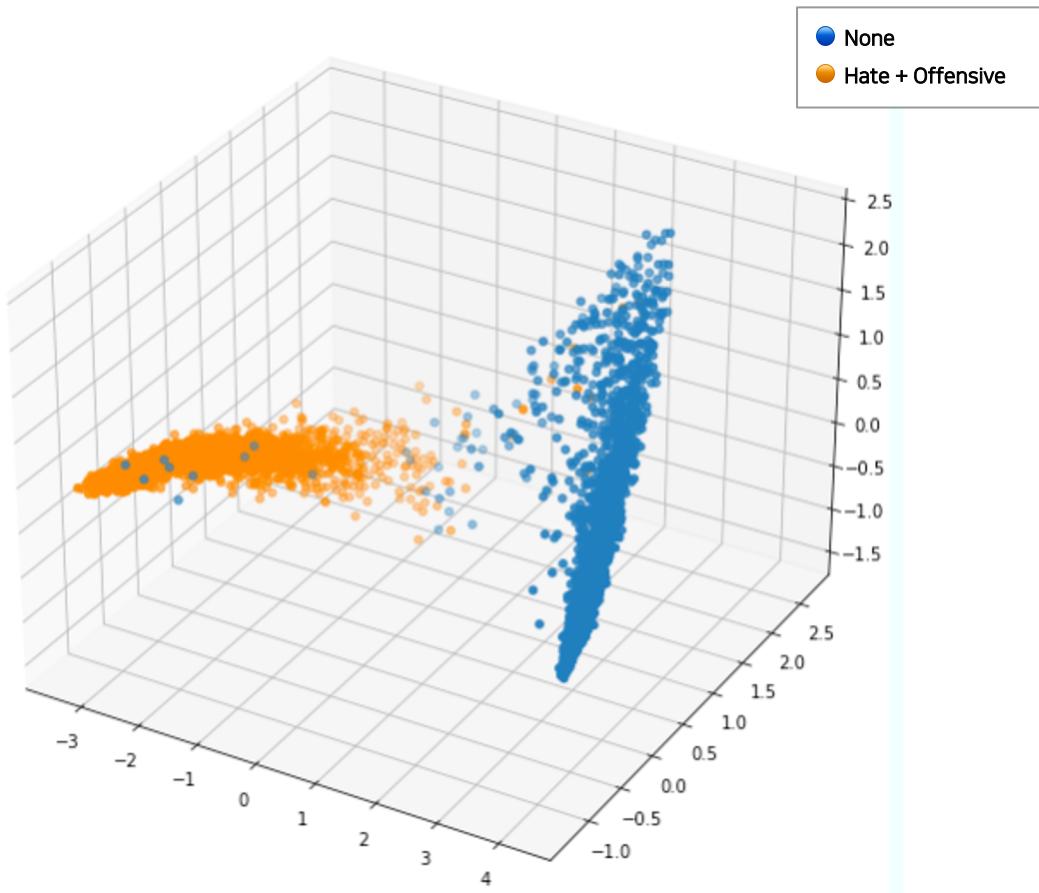
After Clustering



## 4-5. Clustering with Triplet Loss & KNN

Two-stage model: [None / Hate&Offensive], [Offensive / Hate]

- KNN(K=15, 23) LB f1-score: 62.452



## 4-5. Clustering with Triplet Loss & KNN

Two-stage model + pseudo labeled 300K data

- Validation f1-score
  - [None-Hate (K=19)]: 87.133, [Offensive-Hate (K=11)]: 82.072
- LB f1-score: 66.115

The screenshot shows the Kaggle interface for a competition. On the left, there's a sidebar with links like Home, Competitions, Datasets, Code, Discussions, Courses, More, Your Work, and Recently Viewed. The 'Competitions' link is highlighted. The main area is the Leaderboard page for the 'Korean Hate Speech Detection' competition. At the top, there are tabs for Overview, Data, Code, Discussion, Leaderboard (which is selected), Rules, Team, My Submissions, Submit Predictions, and an ellipsis. Below the tabs, it says 'This leaderboard is calculated with all of the test data.' There are buttons for Raw Data and Refresh. The leaderboard table has columns for #, Team Name, Notebook, Team Members, Score, Entries, and Last. The entries are:

#	Team Name	Notebook	Team Members	Score	Entries	Last
1	minjoon			0.67653	7	8mo
2	AI-it		+4	0.66115	44	now
3	Sang-geon Yun			0.65740	36	13d
4	b			0.65366	3	3mo
5	David Kim			0.65311	13	2h
6	April Kang			0.65026	27	8mo
7	Juhwan Choi			0.65013	14	7mo
8	Greenstar			0.64922	20	13d
9	offert100			0.64774	6	9mo

## 5. 모델 최적화

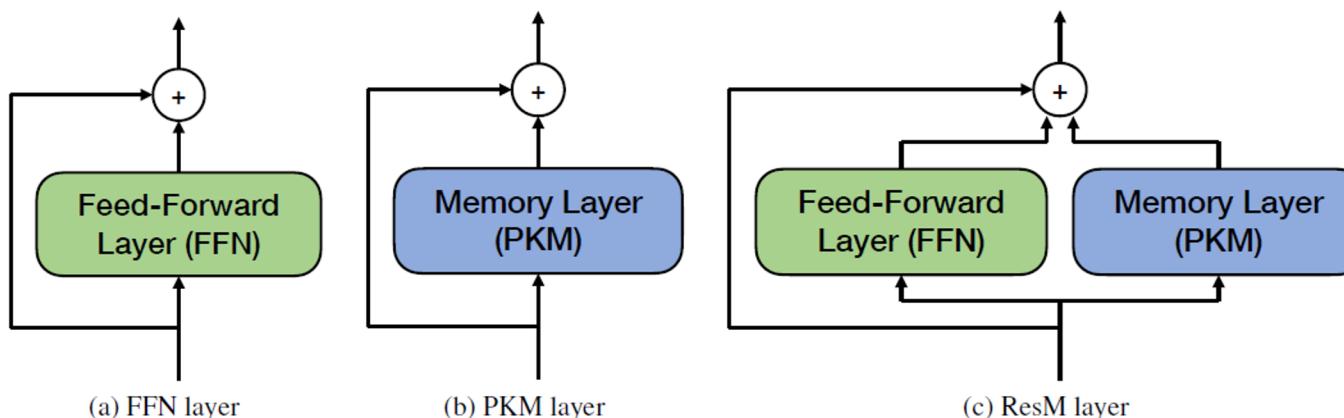
# 5-1. 최적화 : Applied PKM to PLM

## Goal

- 성능 향상 + Inference Time 유지
- 모델 최적화 논문 구현 실습

## Main Structure

- PLM Residual Block + PKM Product Key Memory
- 기존 Residual Block 보존 = Pretrained Weights 소실 문제 해결



Model	# Layers	# Params	Inference Speed (batch/sec)
BERT <sub>BASE</sub>	12	110M	79.8
BERT <sub>BASE</sub> +PKM	12	506M	61.4
BERT <sub>BASE</sub> +ResM	12	515M	59.3
BERT <sub>LARGE</sub>	24	340M	43.1
BERT <sub>LARGE</sub> +PKM	24	860M	37.2
BERT <sub>LARGE</sub> +ResM	24	876M	36.1

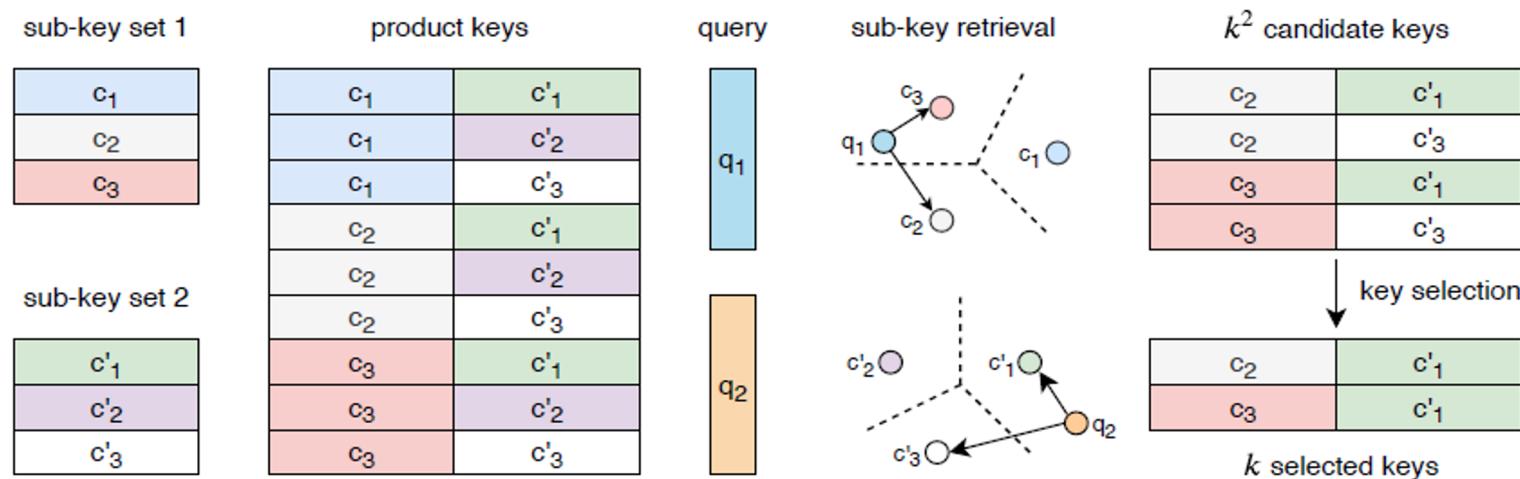
## 참고자료

[실험 상세내용](#)  
[실험 구현코드](#)

# 5-1. 최적화 : Applied PKM to PLM

## What is PKM

- Add Trainable Parameter
  - Train / Inference 과정에서 Input Data 별 연관성 높은 Top-K Parameter 활용 by KNN
    - Total # of Parameter 증가
    - Inference Time 유지
- Memory Module
  - Linear Layer
  - EmbeddingBag Layer

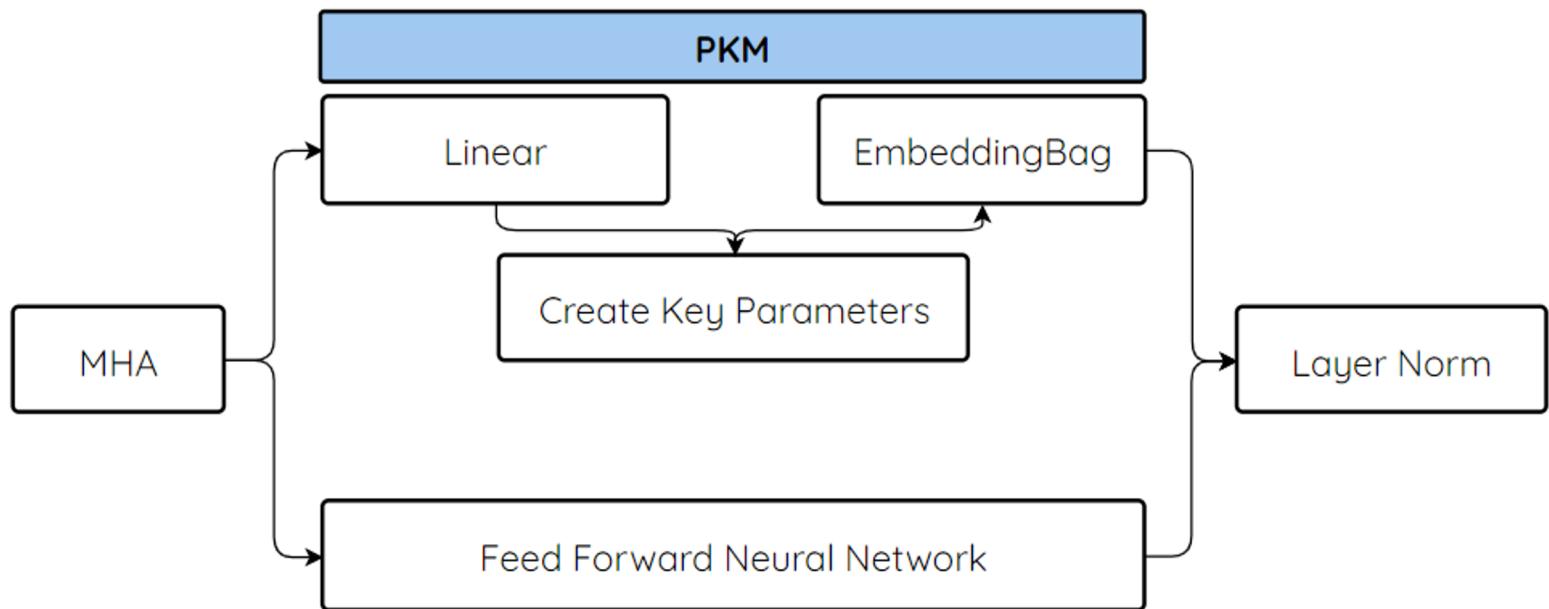
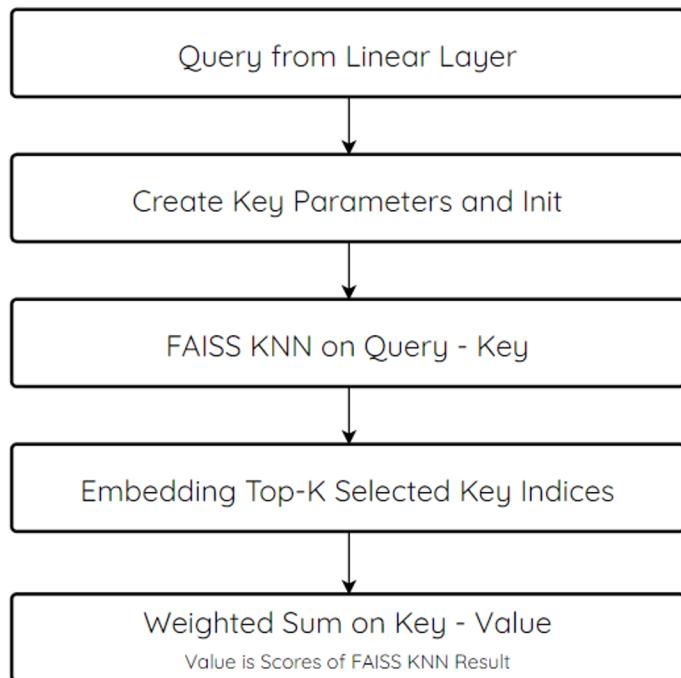


참고자료

[실험 상세내용](#)  
[실험 구현코드](#)

# 5-1. 최적화 : Applied PKM to PLM

## Detail



참고자료

[실험 상세내용](#)  
[실험 구현코드](#)

## 5-1. 최적화 : Applied PKM to PLM

### Result

- mem\_size 늘릴수록 Validation F1 향상 + Inference Time 유지
  - EmbeddingBag Layer Input Dimension =  $\text{mem\_size} * \text{mem\_size}$
  - Inference Time of mem\_size = 128  $2.299692 (s)$  batch\_size = 16
  - Inference Time of mem\_size = 768  $2.319346 (s)$  batch\_size = 16
- Applicable
  - Large Model 사용 시 Inference Time 증가 부담될 때
  - Base Model 대비 성능 향상 + Inference Time 증가폭 최대한 줄이고 싶을 때

mem\_size = 128 PKM

```
train/loss      : 0.24874999977958698
train/macro_f1 : 88.40376031438464
val/loss       : 1.1377064297596613
val/macro_f1   : 67.20628674730841
epoch          : 2
steps          : 900
```

# of Parameter x 10



mem\_size = 768 PKM

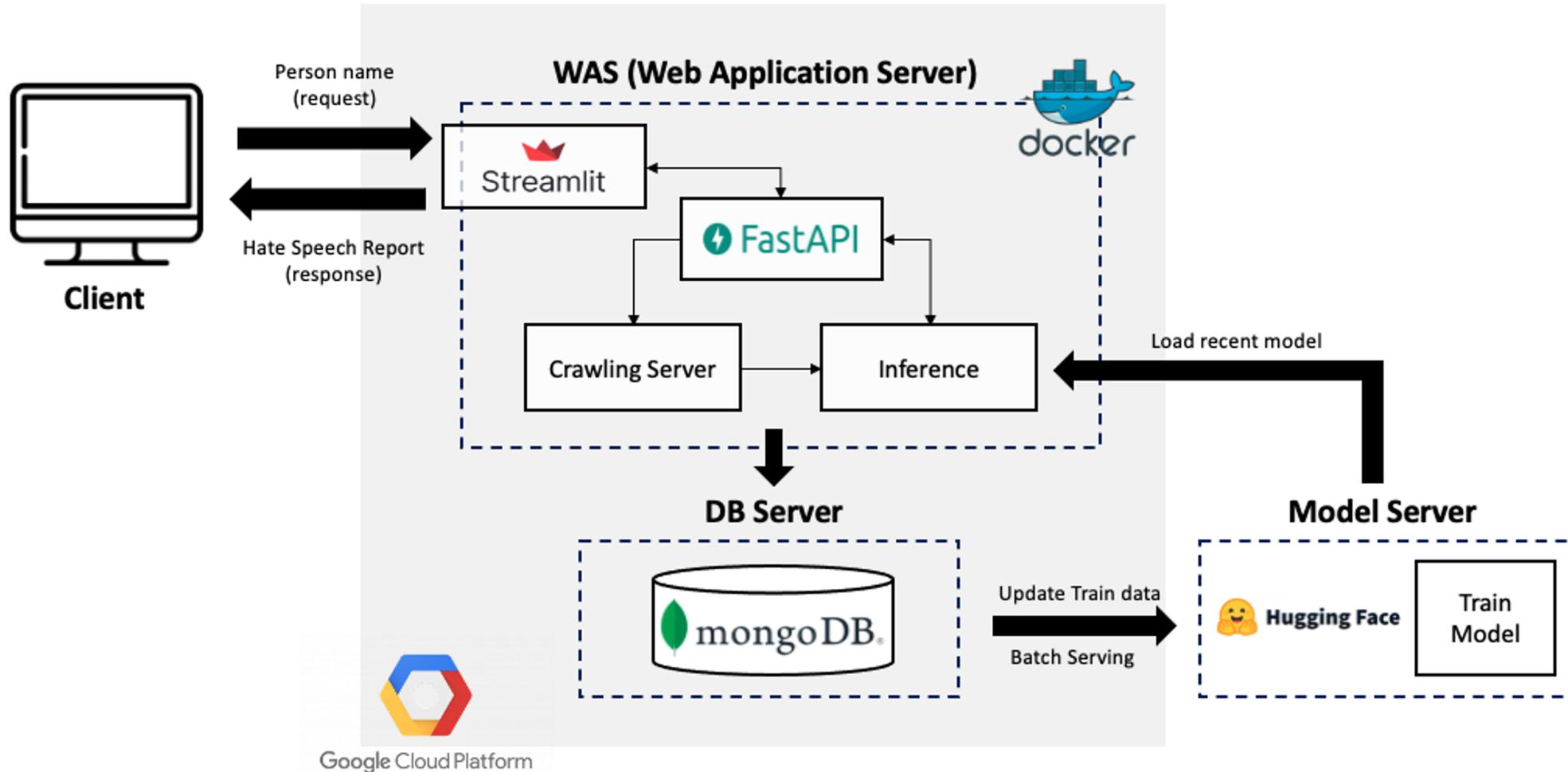
```
train/loss      : 0.2484590121017148
train/macro_f1 : 88.40990812524262
val/loss       : 1.133202891300122
val/macro_f1   : 67.78531265966765
epoch          : 2
steps          : 900
```

### 참고자료

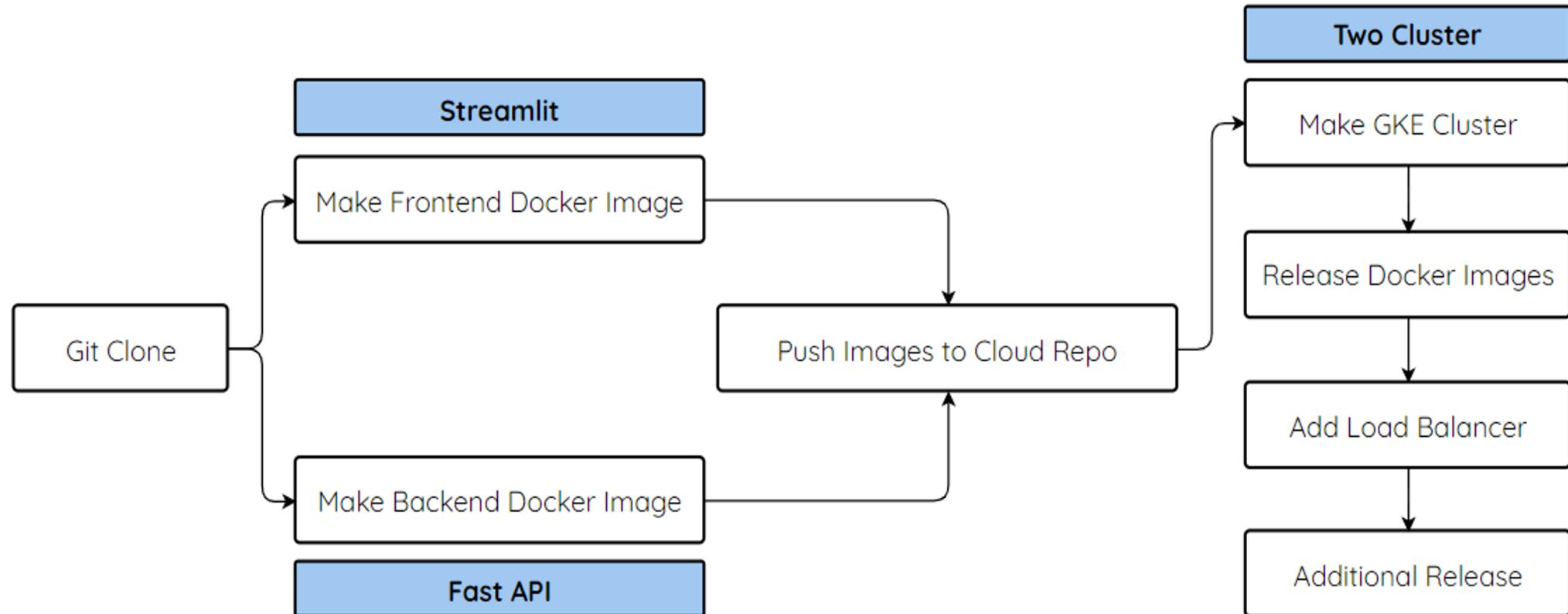
[실험 상세내용](#)  
[실험 구현코드](#)

## 6. 서비스 시스템 구조

## 6-1. 시스템 구조도



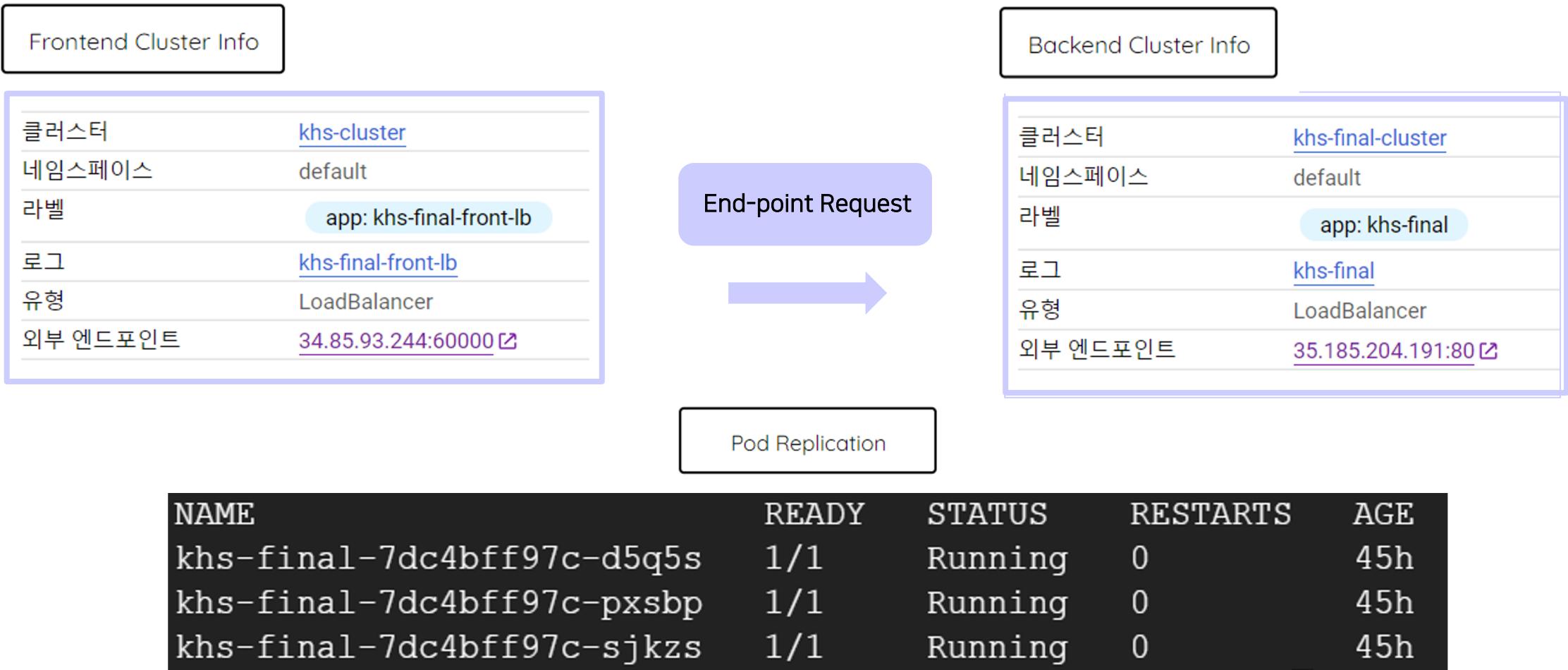
## 6-2. Release on Cloud



참고자료

[배포상세](#)

## 6-2. Release on Cloud



참고자료

배포상세

시연



Hello AI-it!!

## Malicious Comments Collecting Service

password

I



[Authenticate](#)

# 시연

Hello AI-it!!

## Malicious Comments Collecting Service

password

 ... o

You are authenticated!

Keyword you want to collect!!

 이승우

Done!

## Report

### Label Description

- **Hate:** 혐오적인 표현
- **None:** 일반적인 표현

# Future Work

## Future Work

---

- Active Learning 구현
- 서비스 평가 지표
  - A/B Test
  - 유명인 컨택 및 서비스 만족도 평가
- 모델 경량화
- Report Visualization 고도화
- 악플 수집 사이트 다각화

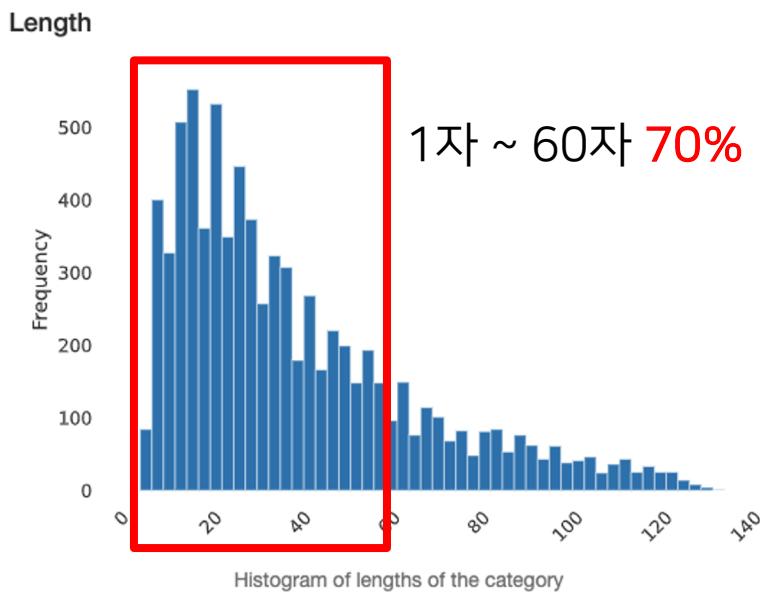
# Q&A

# Appendix. 실패한 실험

# Appendix. 실패한 실험

## EDA 분석 결과

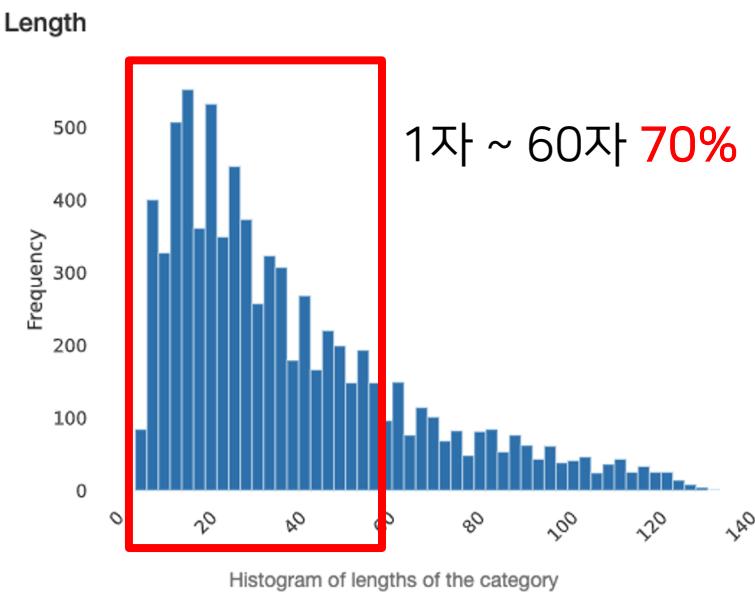
train data의 문장 최대 길이 : 135



# Appendix. 실패한 실험

## EDA 분석 결과

train data의 문장 최대 길이 : 135



문장이 짧을 수록 → 의미를 파악 **HARD**  
주어진 데이터 → 대부분 짧은 문장

가설 : 모델이 문장의 의미를 파악해 분류하는 것이 힘들 것  
짧은 문장을 잘 분류할 수 있는 방법이 무엇이 있을까 조사

# Appendix. 실패한 실험

---

## 1) DNN model + Knowledge Graph + similarity based on CNN

Knowledge + DNN = 더 포괄적, 이해력 있게 정보 계산 가능

short text classification은 크게 explicit representation / implicit representation 연구로 진행되어옴

### 1. explicit representation

- segmentation → POS tagging → syntactic analysis → text feature
- 많은 feature를 찾는건 가능하지만 ambiguity 문제는 여전히 풀지 못한다. & POS tagging은 매우 많은 자원 소모 & sparsity ↑

### 1. implicit representation

- 짧은 문장을 neural network를 통해 hyperspace로 매핑
- 보다 풍부한 semantic 정보 capturing
- Knowledge 무시

예) "Warrior won the NBA Championship"에서 *Warrior*

여기서 *Warrior*는 농구 팀이지만 implicit model은 사람이나 새로운 단어로 여긴다

"이 둘을 결합시켜 장점을 지키며 단점을 상쇄하고자 함"

### 참고자료

[paper] Short text classification via knowledge powered attention with similarity matrix based CNN

# Appendix. 실패한 실험

## 1) DNN model + Knowledge Graph + similarity based on CNN

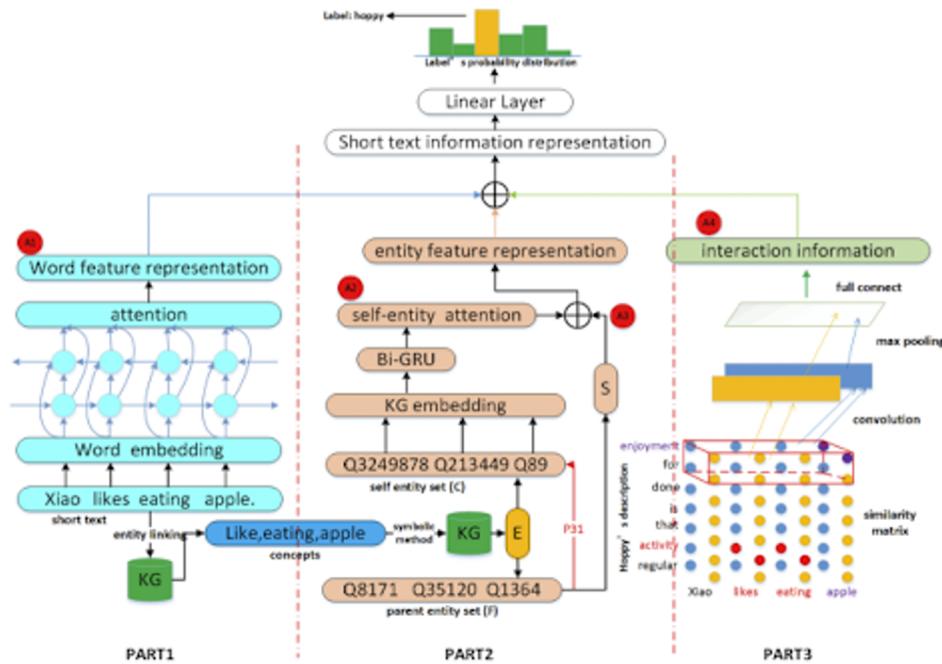
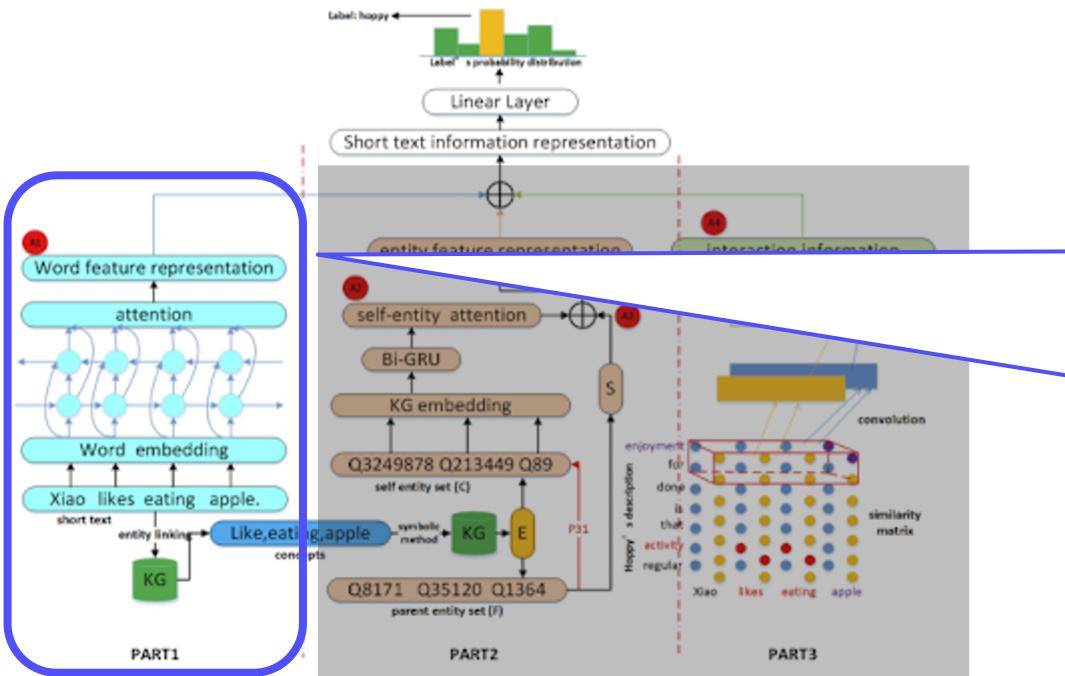


Figure 2: Our KASM model. The yellow box E refers to the function about how to choose the right entity. The box S refers to the parent-entity has the same operation as the self-entity, but has not the same attention mechanism.

# Appendix. 실패한 실험

## 1) DNN model + Knowledge Graph + similarity based on CNN



### DNN model

Bi-GRU 사용

기존에 사용했던 단일 모델로 학습시키듯 적용하면됨

$$u_j = \tanh(W_w h_j + b_w)v^T$$

$$\eta_j = \text{softmax}(u_j U_w)$$

$$A1 = \sum_{j=1}^l \eta_j h_j$$

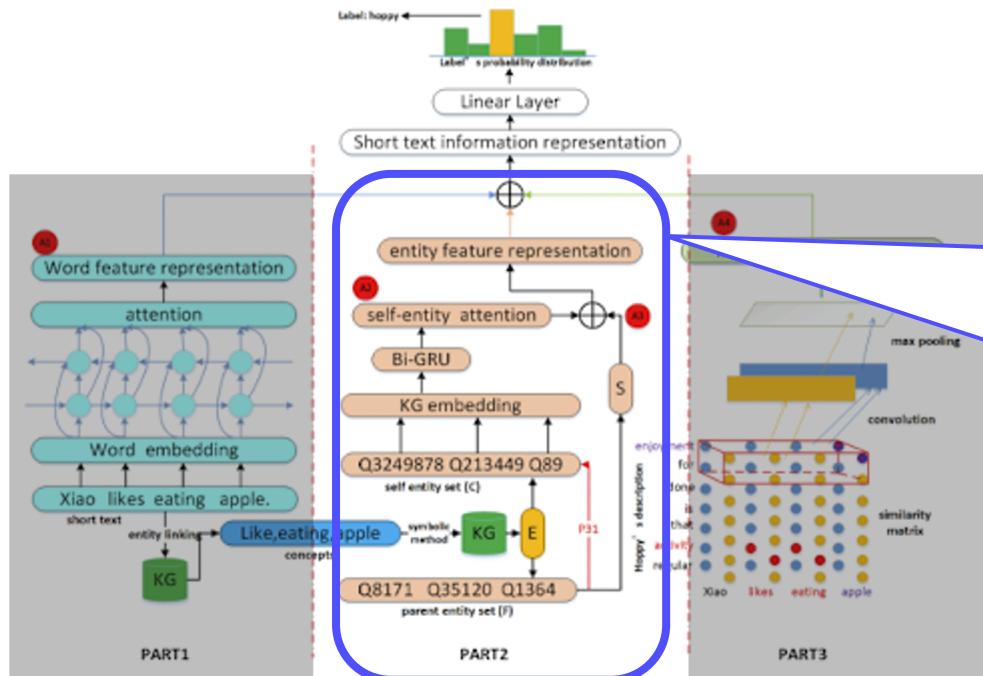
Figure 2: Our KASM model. The yellow box E refers to the function about how to choose the right entity. The box S refers to the parent-entity has the same operation as the self-entity, but has not the same attention mechanism.

### 참고자료

[paper] Short text classification via knowledge powered attention with similarity matrix based CNN

# Appendix. 실패한 실험

## 1) DNN model + Knowledge Graph + similarity based on CNN



### Knowledge Graph(KG) Model

KG 를 이용하여 entity 간의 정보를 통해 Text에서 보다 명확한 의미 받아오고자 하는 모델

#self-entity #parent-entity

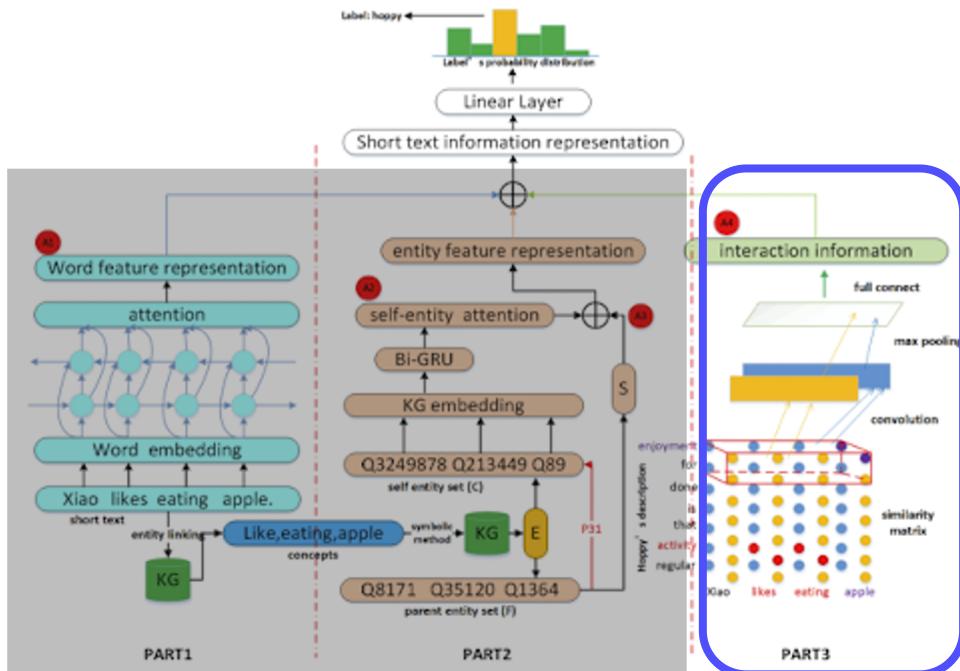
Figure 2: Our KASM model. The yellow box E refers to the function about how to choose the right entity. The box S refers to the parent-entity has the same operation as the self-entity, but has not the same attention mechanism.

### 참고자료

[paper] Short text classification via knowledge powered attention with similarity matrix based CNN

# Appendix. 실패한 실험

## 1) DNN model + Knowledge Graph + similarity based on CNN



### Similarity based on CNN

Short text와 Label간의 interaction information을 보여준다.

label과 text를 cosine  $\rightarrow$  CNN  $\rightarrow$  max pooling  $\rightarrow$  FC

Figure 2: Our KASM model. The yellow box E refers to the function about how to choose the right entity. The box S refers to the parent-entity has the same operation as the self-entity, but has not the same attention mechanism.

### 참고자료

[paper] Short text classification via knowledge powered attention with similarity matrix based CNN

# Appendix. 실패한 실험

---

## 1) DNN model + Knowledge Graph + Similarity based on CNN

### 한계점 : Knowledge Graph 문제

이 모델의 가장 중요한 특징이자 성능을 끌어올린 것이 KG(Knowledge Graph)인데 이로 인한 한계가 발생

1. 한국어로 임베딩 된 KG 오픈소스 데이터가 없다.
2. [BERT와 지식 그래프를 이용한 한국어 문맥 정보 추출 시스템](#)을 보면 한국어를 NMT 를 통해 영어로 번역 후 KG를 사용했다고 하는데 우리가 쓰는 데이터는 욕설, 맞춤법 파괴, 초성인 데이터가 많아서 번역이 잘 이뤄지지 않음 → 효용가치에 의문

### 참고자료

[\[paper\]Short text classification via knowledge powered attention with similarity matrix based CNN](#)

[\[paper\]BERT와 지식 그래프를 이용한 한국어 문맥 정보 추출 시스템](#)

# Appendix. 실패한 실험

---

## 2) Pre-Trained Model + AutoEncoder + K-means

짧은 말뭉치들은 sparse, high-dimensional, noise 등의 특징을 가짐  
그렇기에 짧은 텍스트의 효과적인 학습 방법은 Clustering 이라 주장

또한 Pretrained model 모델들은 general한 상황에 사용할 목적으로 만들어졌기에 짧은 문장,  
말뭉치 Clustering 에 대해서 suboptimal

이를 보완하기 위하여 unsupervised autoencoder를 이용한 2가지 방법을 소개

1. STN-GAE (Structural Text Network Graph Autoencoder)
2. SCA-AE (Soft Cluster Assignment Autoencoder)

### 참고자료

[paper] [Representation learning for short text clustering](#)

# Appendix. 실패한 실험

## 2) Representation Learning for Short Text Clustering

짧은 말뭉치들은 sparse, high-dimensional, noise 등의 특징을 가짐  
그렇기에 짧은 텍스트의 효과적인 학습 방법은 Clustering 이라 주장

또한 Pretrained model 모델들은 general한 상황에 사용할 목적으로 만들어졌기에 짧은 문장,  
말뭉치 Clustering 에 대해서 suboptimal

이를 보완하기 위하여 unsupervised autoencoder를 이용한 2가지 방법을 소개

1. STN-GAE (Structural Text Network Graph Autoencoder)
2. SCA-AE (Soft Cluster Assignment Autoencoder)

Dataset	Metric	BoW	W2V	BERT	AE	STN-GAE	SCA-AE
MR	ACC	0.5188	0.526	<b>0.7848</b>	0.7492	0.6625	0.7662
	NMI	0.0037	0.002	<b>0.2511</b>	0.2013	0.1676	0.2198
AGnews	ACC	0.2809	0.3977	0.5748	0.6748	0.5742	<b>0.6836</b>
	NMI	0.0094	0.1193	0.2577	0.3299	0.2914	<b>0.3414</b>
SearchSnippets	ACC	0.2459	0.623	0.6675	0.6429	0.4144	<b>0.6871</b>
	NMI	0.0893	0.4782	0.4763	0.4619	0.3167	<b>0.5026</b>
Yahoo	ACC	0.1507	0.1355	0.4271	0.4803	0.3387	<b>0.5606</b>
	NMI	0.0413	0.0264	0.2765	0.3069	0.2529	<b>0.3472</b>
Tweets	ACC	0.6161	0.5771	0.8126	0.8424	0.4049	<b>0.8485</b>
	NMI	0.7284	0.6084	0.867	0.8875	0.3546	<b>0.8919</b>
StackOverflow	ACC	0.4615	0.2289	0.6253	0.6672	0.4049	<b>0.7655</b>
	NMI	0.5506	0.1905	0.5962	0.6156	0.4492	<b>0.6599</b>
Biomedical	ACC	0.1388	0.2296	<b>0.4043</b>	0.3894	0.1550	0.4025
	NMI	0.0887	0.2166	<b>0.3347</b>	0.3385	0.1468	0.3329

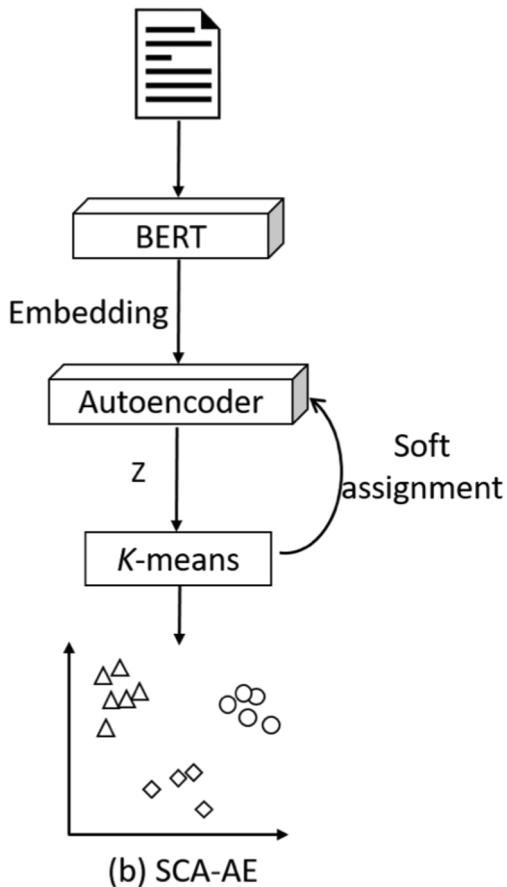
### 참고자료

[paper] Representation learning for short text clustering

# Appendix. 실패한 실험

## 2) Representation Learning for Short Text Clustering

Pre-Trained Model + AutoEncoder + K-means



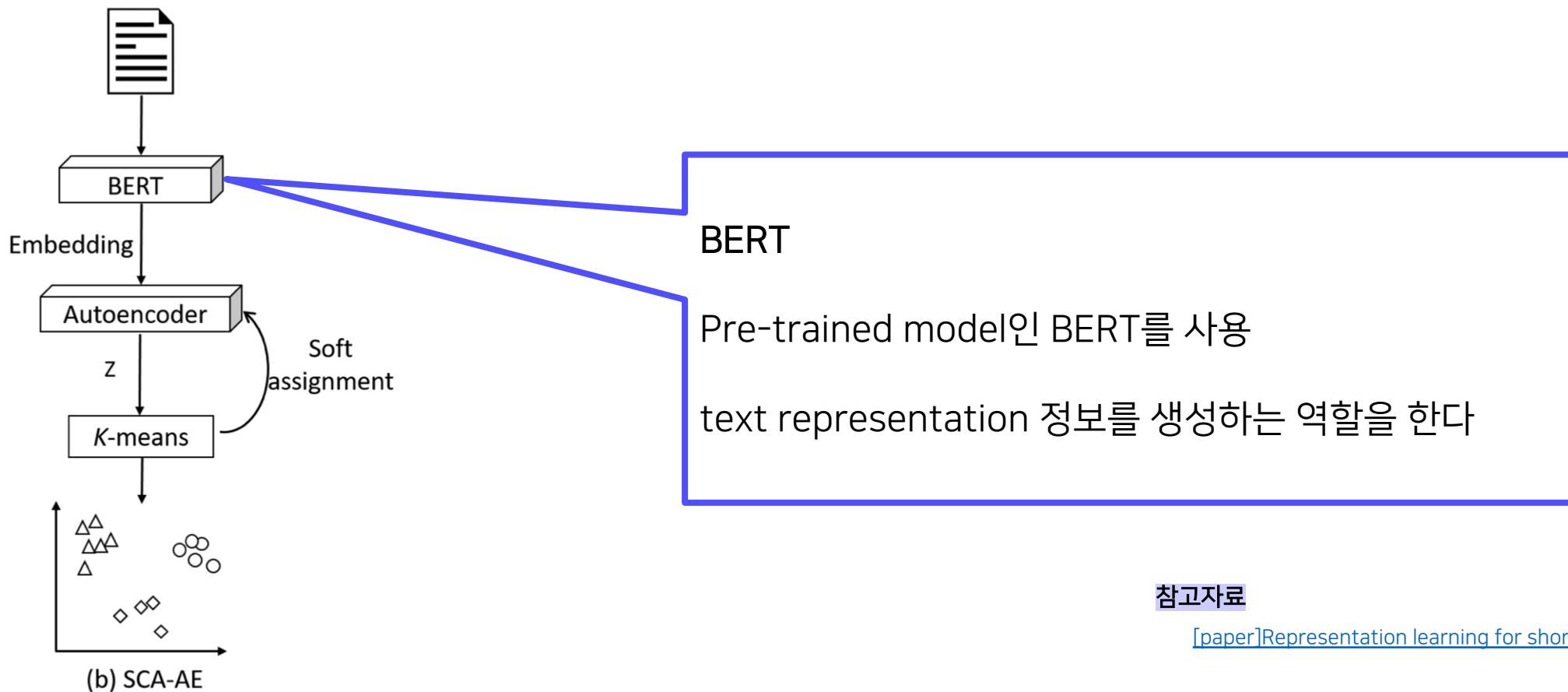
참고자료

[paper] [Representation learning for short text clustering](#)

# Appendix. 실패한 실험

## 2) Representation Learning for Short Text Clustering

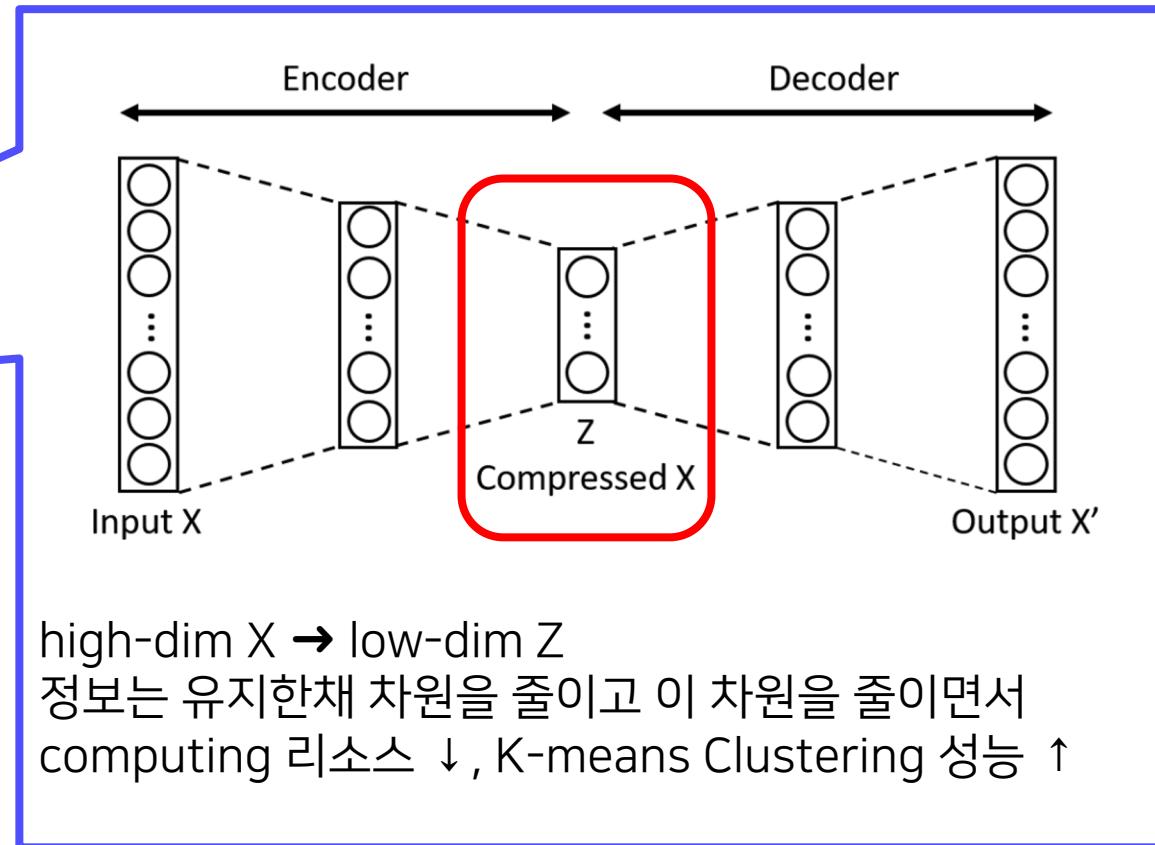
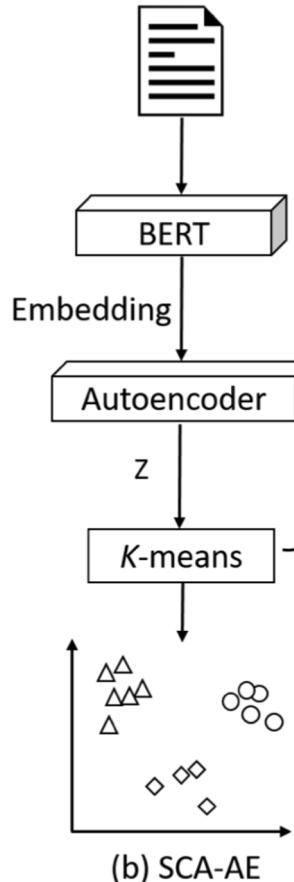
Pre-Trained Model + AutoEncoder + K-means



# Appendix. 실패한 실험

## 2) Representation Learning for Short Text Clustering

Pre-Trained Model + AutoEncoder + K-means



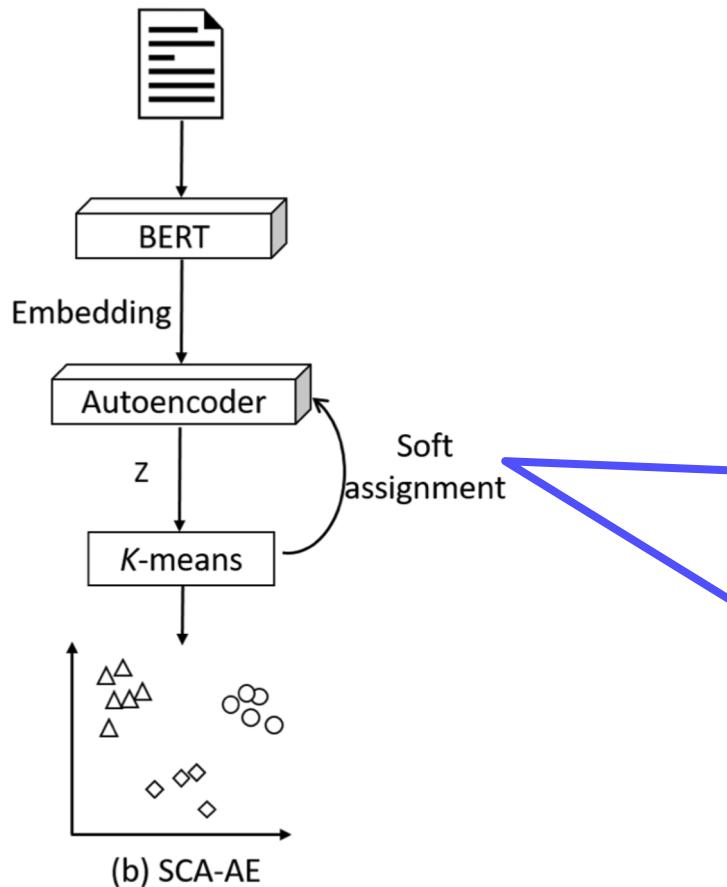
참고자료

[paper]Representation learning for short text clustering

# Appendix. 실패한 실험

## 2) Representation Learning for Short Text Clustering

Pre-Trained Model + AutoEncoder + K-means



### Soft Cluster assignment

보조 target distribution 으로 사용  
encoder weights와 clustering assignments를 묶어  
fine-tuning하여 clustering 성능을 향상시키려함

$$q_{ij} = \frac{(1 + \| z_i - u_j \|^2)^{-1}}{\sum_{j'} (1 + \| z_i - u_{j'} \|^2)^{-1}} \quad p_{ij} = \frac{q_{ij}^2 / \sum_{i'} q_{i' j}}{\sum_{j'} q_{ij'}^2 / \sum_{i'} q_{i' j'}}$$

$z$ : embedding node,  $u$ : centroid

$$L_{soft} = KL(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

참고자료

[paper]Representation learning for short text clustering

# Appendix. 실패한 실험

---

## 2) Representation Learning for Short Text Clustering

### 선정 이유

1)에서 언급한 Bi-GRU 모델 사용과 KG를 사용하는 것에 비해 Pre-trained Model을 이용한 모델  
논문에서는 BERT를 사용했지만  
task adapted model인 KcELECTRA 모델이 공개되어 있어 이를 사용하고자 함

→ 모델링 시간을 줄이고 task adapted model을 활용하기 때문에 좋은 성능 기대

### 한계점

제한된 기간 이내 구현이 이뤄지지 않음

→ 향후 구현을 완료하여 repo 업데이트를 할 예정

### 참고자료

[paper]Representation learning for short text clustering

# [추가자료] Tokenizer

---

## beomi/KcELECTRA-base tokenizer

- BertWordPieceTokenizer
- KcELECTRA vocab + KoELECTRA vocab → 50135 개

## Word Piece

- [Stage 1] space 기반 word tokenizing → [Stage 2] Byte Pair Encoding(BPE) 기반 subword tokenizing
- 악플 판별에 핵심이 될만한 (등장 빈도가 높은) word 를 tokenizer vocab 에 추가



(word in vocab) tokenizer: '후레자식' ⇒ ['후', '##레', '##자식']

(word not in vocab) tokenizer: '후레자식' ⇒ ['후레자식']

## Unused Token

- beomi/KcELECTRA-base tokenizer unused token → 1000 개
- 악플에 존재할 확률이 높은 욕 644 개 중 vocab 에 존재하는 105 개 제외, 539 개 추가

## 참고자료

[Beomi/KcELECTRA](#)  
[Unused Token](#)