커뮤니티
정치 분란글 필터 제작기
(삽질기)

숭실대학교 SSUML 장원준

Who am i?

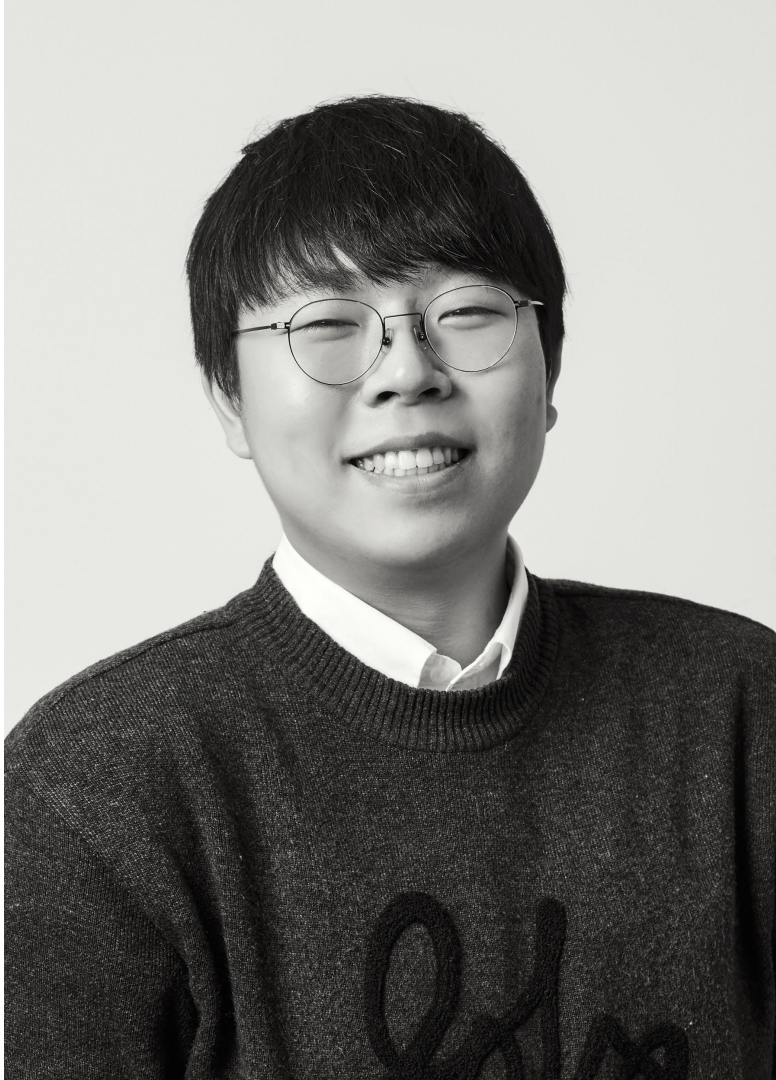
장원준

숭실대 소프트웨어학부 17학번

소프트웨어 마에스트로 9기

숭실대학교 커뮤니티 유어슈 - 리서치팀

피플펀드에서 Backend Engineer로 병역특례중



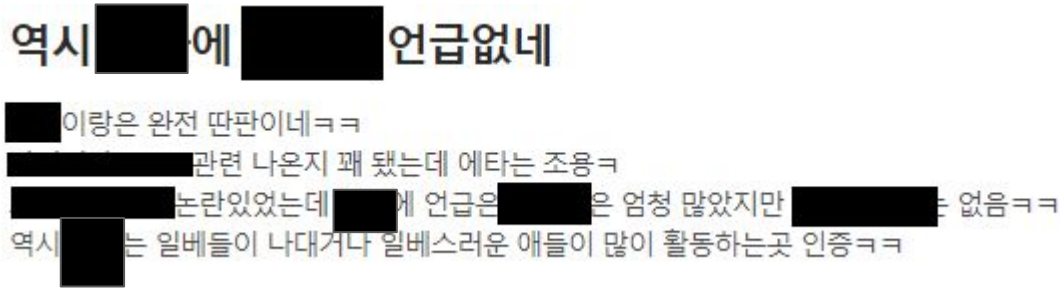
커뮤니티 정치 분란글이란?

좌 우 정치 성향을 드러내며 상대편을 비꼬는 글들.

하나 하나의 단어로는 분란성 단어가 아니며, 욕설도 거의 없음.

추가적으로, 커뮤니티에서 정치 이야기가 나올 경우 대부분 분란글이됨.

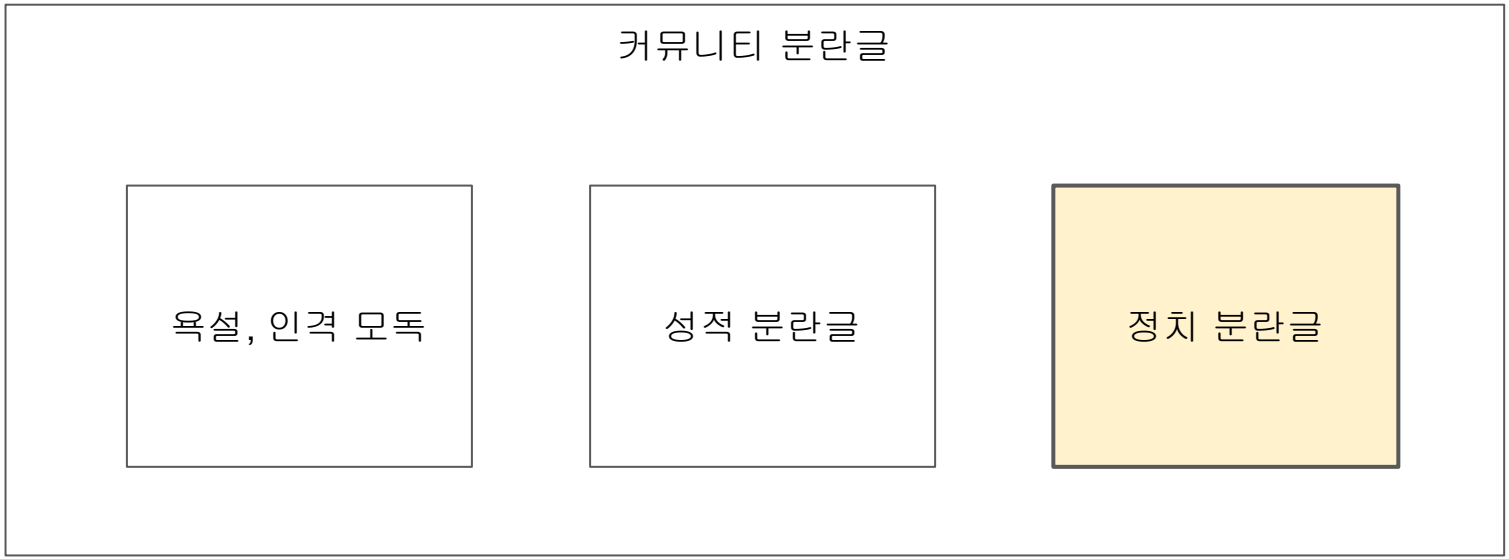
예시)



커뮤니티 분란글 데이터 특성

사실, 커뮤니티 분란글은 특성이 매우 다양합니다.

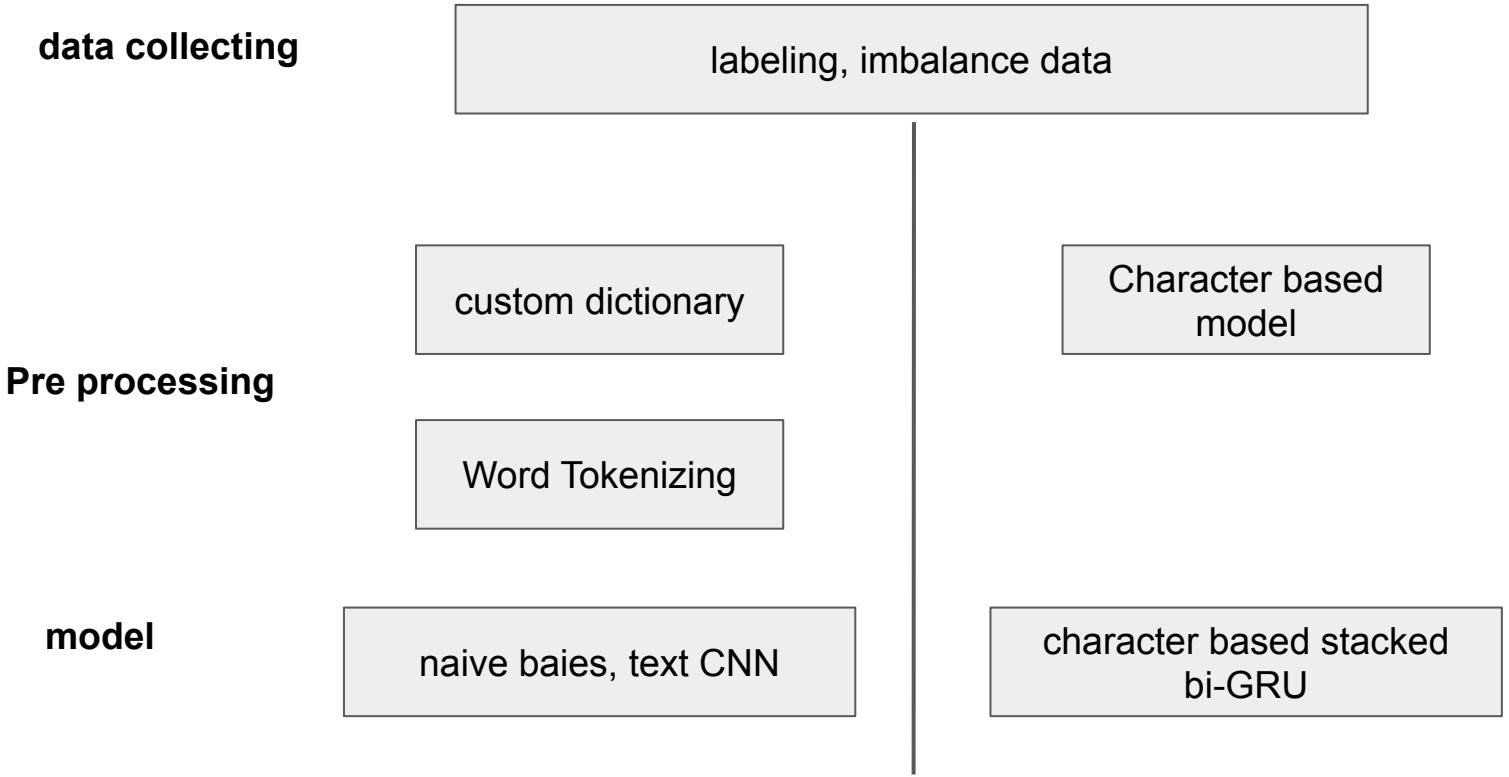
처음에는 정의심에 불타 아래 모든 분란글을 잡아보자! 하면서 호기롭게 도전했습니다.



커뮤니티 분란글 데이터 특성

그렇게 호기롭게 도전했다가 1년동안 고통받았습니다.

학부생의 호기로운 생각으로, '분명히 CNN RNN 하면 될꺼같았는데!' 에서 시작한 고통들이였습니다.


data collecting
labeling, imbalance data
Pre processing
custom dictionary
Character based model
Word Tokenizing
model
naive baies, text CNN
character based stacked bi-GRU

Pre processing: Word Tokenizing

1안으로, 단어 단위 Tokenizing. Out of Vocabulary problem 을 풀기 위해 custom dictionary를 L Tokenizer를 통해 생성.

보통 한국어는 L+[R] 구조.

숭실대는
명사 / 조사

소프트는
명사 / 조사

띄어쓰기가 잘 되어있다면 어절 왼쪽에 의미를 가지는 단어들이 등장. 이 성질을 이용하여 L-Tokenizer를 구함.

Pre processing: Word Tokenizing

P(소프|소) = 0.5

P(소프트|소프) = 0.9

P(소프트는|소프트) = 0.1

p(xy|x) 값이 급격히 줄어드는 구간에서

L + [R] 중 L part을 발견 할 수 있음.

이 score를 cohesion score라고 함.

전체 dataset중 cohesion score가 top N 개인 단어들을 custom dictionary 에 추가.

$$cohesion(c_{0:n}) = \left(\prod P(c_{0:i+1}|c_{0:i})\right)^{n-1}$$

https://lovit.github.io/nlp/2018/04/09/cohesion_ltokenizer/

Word Tokenizng과 함께 시도했었던 모델들

Naive Bayesian Classifier

가장 빠르게 시도해볼 수 있는 머신 러닝 모델.

가장 빠른게 해본 만큼 성능도 최악.. (**test Accuracy : 43.2%**)

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x}|C_k)}{p(\mathbf{x})}.$$

$$argmax\mathrm{P}(c|x) = argmax\ \frac{\mathrm{P}(x|c)\mathrm{P}(c)}{\mathrm{P}(x)}$$

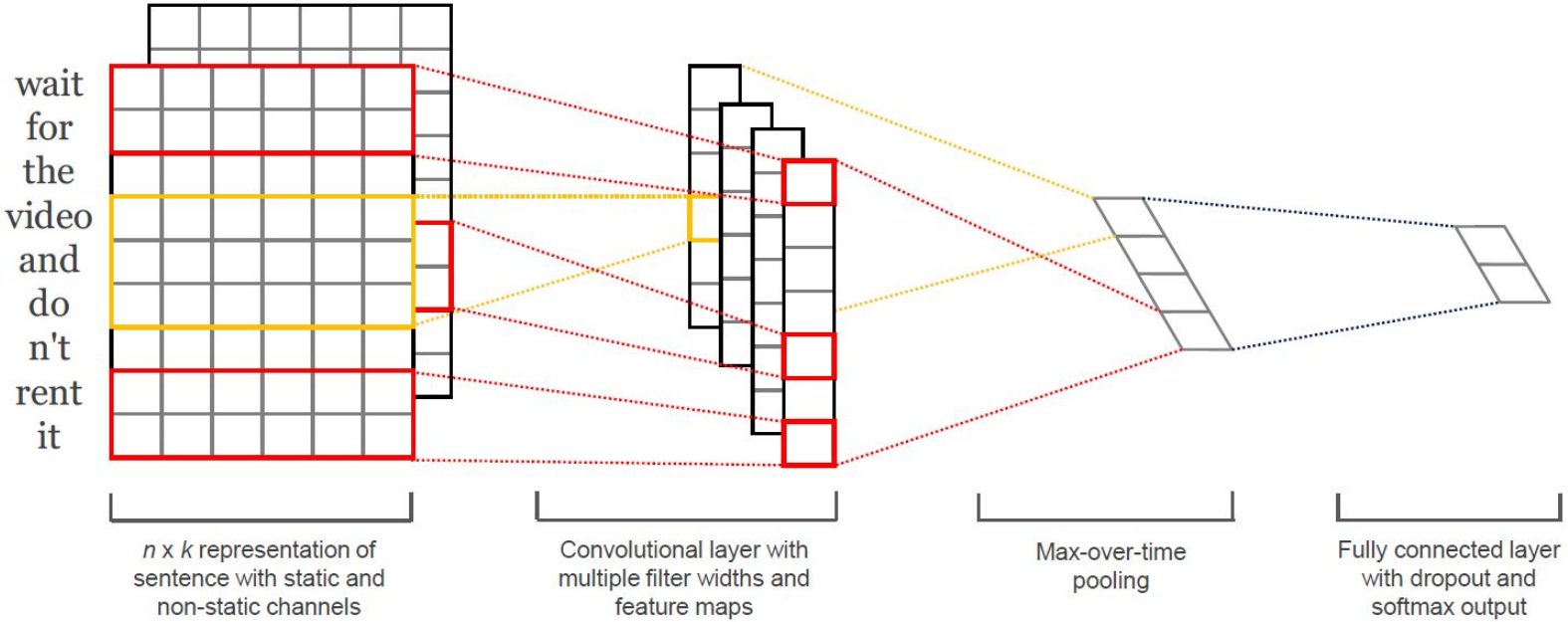
$$argmax\ \mathrm{P}(x|c)\mathrm{P}(c)$$

특정 텍스트에서 개별 단어의 출현 빈도를 모두 기록하고 특정 keyword에 대한 labeling 이 필요했음.

Feature(각 토큰)들간의 독립성이 있어야 한다는 단점이 있다.

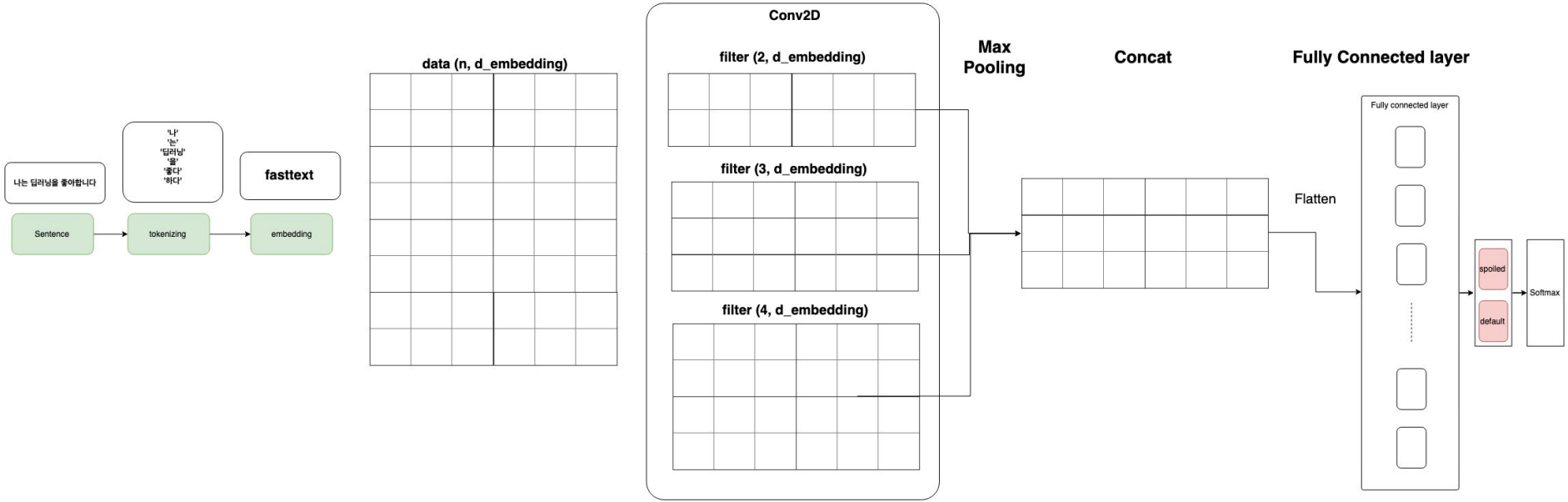
Word Tokenizng과 함께 시도했었던 모델들

textCNN



Word Tokenizng과 함께 시도했었던 모델들

textCNN



Word Tokenizng과 함께 시도했었던 모델들

accuracy: 43% -> 65%

Word Tokenizng과 함께 시도했었던 모델들

잘 안되었던 이유

- long sequence 에 대해서 정확도가 많이 떨어짐.
- sequence 간 dependency를 Convolution 으로 학습하려하니 noisy한 커뮤니티 데이터에서 문맥 의존성을 학습하기 어렵다고 판단함.

Why word based model not worked?

잘 안되었던 이유

- L Tokenizer는 '띄어쓰기가 잘 되어있는 문서' 에만 잘 작동하는데, 커뮤니티 데이터의 경우 **띄어쓰기가 잘 되어있지 않아** 명사 추출이 생각보다 잘 안됨.
- 야민정음에 대해서 정확도가 뛰어나지 못함.

예시)

전남친나쁜새끼

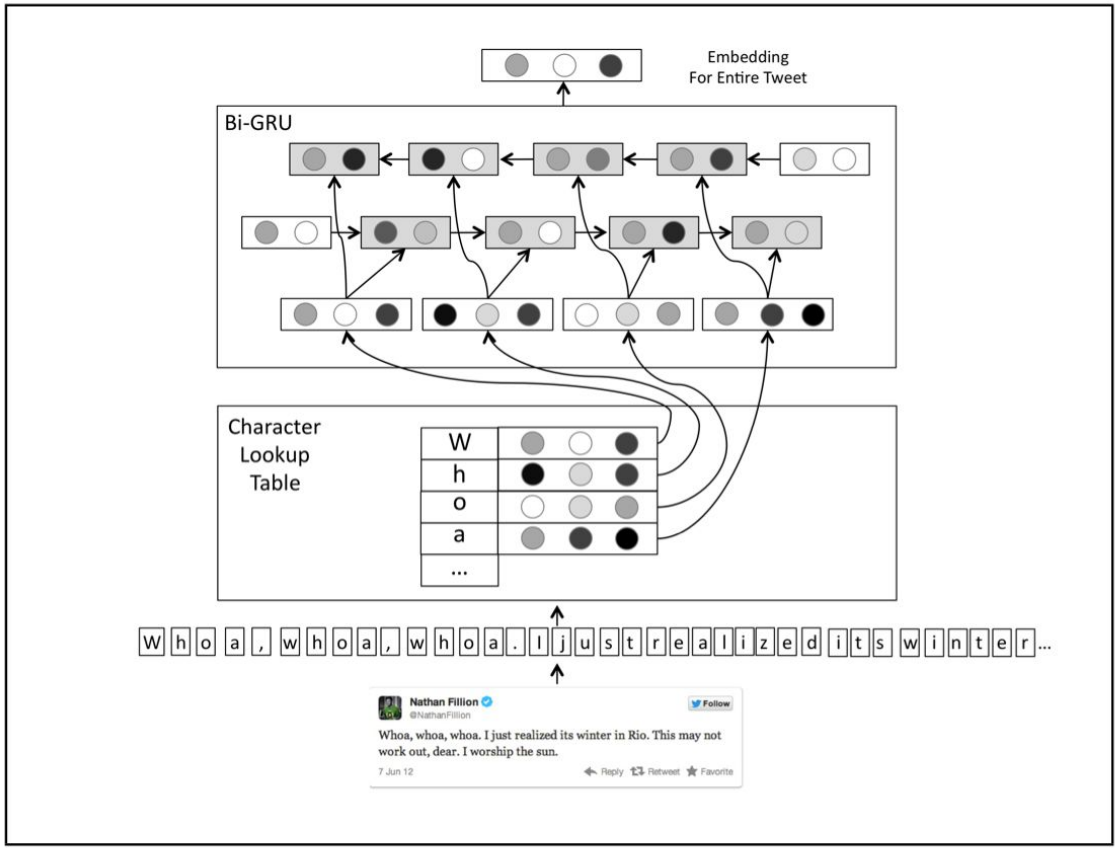
우리학교근처에 아마스빈_이나 공차 제발생겼으면좋겠다ㅜㅠ

모솔들공감함 눌러봐라 인원체크가즈아

Problem: Tokenizing

Tweet2vec 이라는 논문에서 noisy coummunity text data를 다루기 위해

character base tokenizing 활용함.



https://arxiv.org/abs/1605.03481

Problem: Tokenizing

Contribution

- SNS 텍스트 데이터의 맞춤법, 신조어, 변형(야민정음) 등의 문제를 풀기 위해 character-based 로 접근함.
- 위 character base encoding 을 위한 look up tabel 만 만들어두면 word segmentation, word dictionary등 추가적인 전처리가 필요 없음.

https://arxiv.org/abs/1605.03481

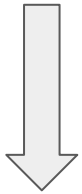
Pre processing: Character based one hot encoding

위 논문을 참고하여 열심히 시도해보았던 **word based tokenizing**을 버리고, **character based one hot encoding**을 시도함.

우리학교근처에 아마스빈_이나 공차 제발생겼으면좋겠다ㅜㅠ

character 단위로 **split.**

특수문자, whitespace 도 모두 ‘문자’ 취급.



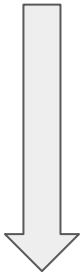
‘우’ ‘리’ ‘학’ ‘교’ ‘근’ ’처’ ‘에’ ‘ ’ ‘아’ ‘마’ ‘스’ ‘빈’‘ ’‘이’ ‘나’ ‘ ’ ‘공’ ‘차’

Pre processing: Character based one hot encoding

‘우’ ‘리’ ‘학’ ‘교’ ‘근’ ’처’ ‘에’ ‘ ’ ‘아’ ‘마’ ‘스’ ‘빈’ ‘ ’ ‘이’ ‘나’ ‘ ’ ‘공’ ‘차’ ----

각 **character one hot encoding**

unicode 안에 있는 모든 **character one hot encoding** 가능.



```
[array([ 11,  32,   5,  39,  18,  19,  40,   0,  31,   0,  37,  43,  14,
        23,  11,  24,  99,  11,  19,   6,  19,   9,  37,   7,  39,  43,
        99,  11,  39,   2,  19,  99,   0,  27,  60,  14,  19,  99,  12,
        24,   7,  19,  47,   9,  20,  60,   0,  25,  59,  11,  37,   6,
        25,  43,  12,  27,  66,   0,  24,  59,   3,  19, 238, 242,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   0,   0,   0,   0,   0,   0])]
```

RNN

기존에는 **textCNN**으로 **classification**을 진행하였음.

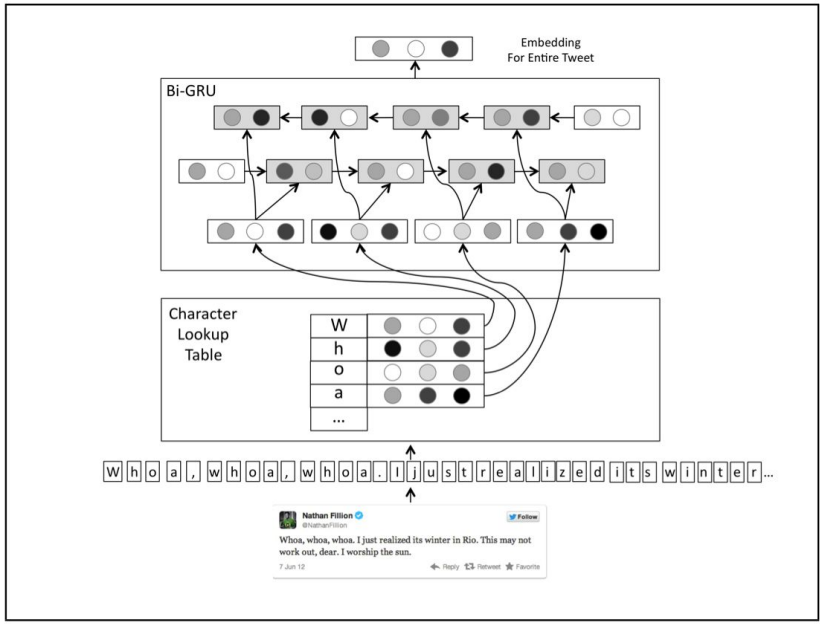
이제는 **Bi-GRU** 을 활용하여 **sequence data**를 학습.

기존 논문에서는 오른쪽 그림처럼 **Bi-GRU 1 layer** 가 있고,
forward의 마지막 **unit**의 **output,**

backward 의 마지막 **unit**의 **output**을 가져와

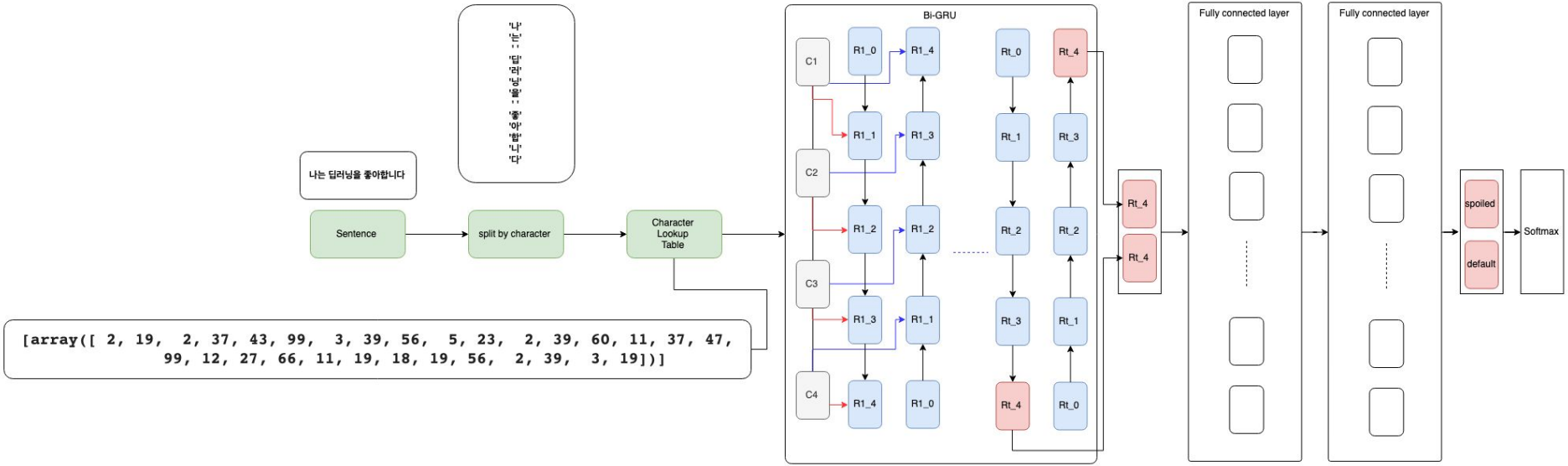
Fully connected layer로 데이터를 **merge**하고,

다시한번 **Fully connected layer**로 **classification**을
진행함.



RNN

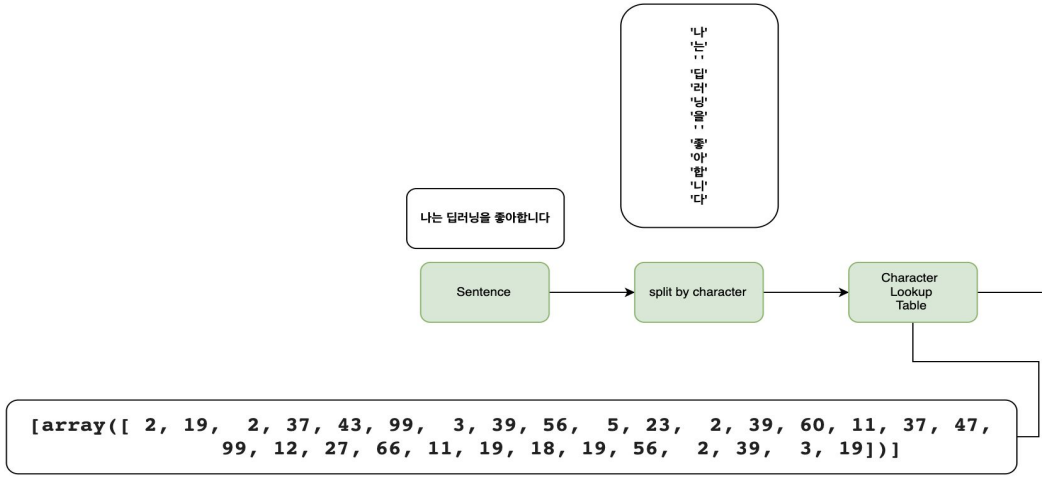
기존 논문의 모델을 조금 변경하여 사용함



RNN

논문에서는 **unicode**를 모두 커버한다하였지만,

전체 **unicode**를 하나하나 **lookup table**을 만들기에는 시간이 너무 오래걸려 한글 + 자주 사용되는 특수문자 + 영어 를 **one hot encoding** 으로 변환함.

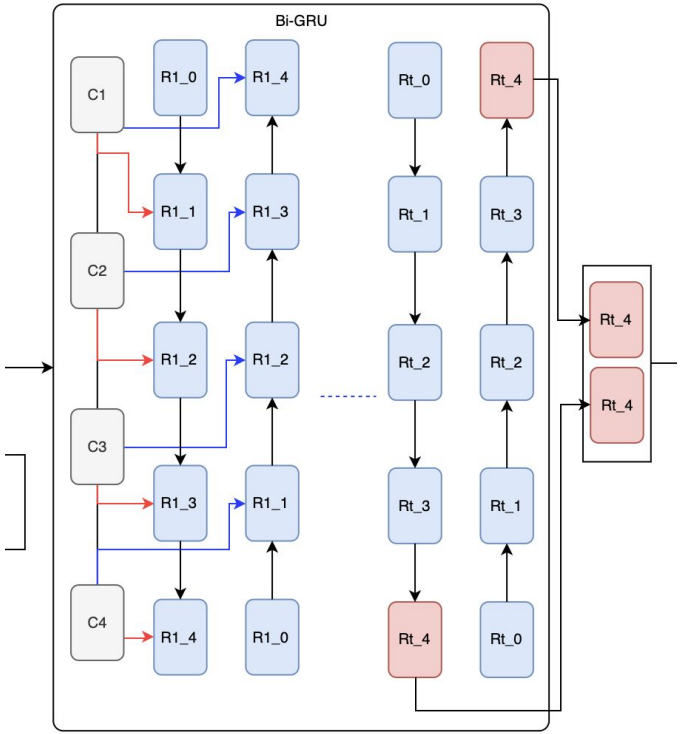


RNN

Tweet2vec 논문에서는 **Bi-GRU 1 layer** 를 사용했지만,

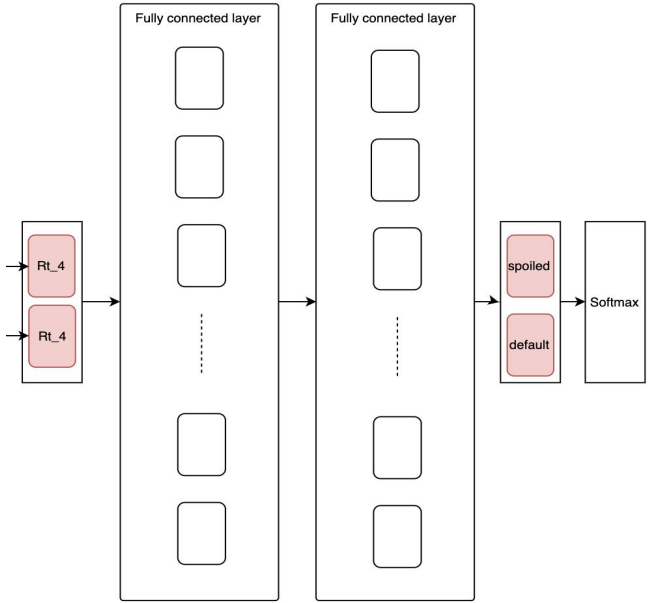
커뮤니티 게시글의 경우 **tweet** 보다 **length**가 긴 경우가 대다수라 **bi-gru**를 **stacking**하여 사용함.

마지막 **bi-gru layer**에서 **forward**의 마지막 **unit**의 **output, backward** 의 마지막 **unit** 의 **output**을 함께 **fully connected layer**로 전달하게됨.



RNN

마지막 **FF layer output**을 **2 dim** 으로 변경하여 **binary classification**을 할 수 있도록 모델을 조금 변경.



Tweet2vec

accuracy: 65% -> 79%

Tweet2vec

잘 안되었던 이유

- Bi-GRU 로 어느정도 문장 내부의 dependency를 학습을 하는것 같은데, 어떤게 문제일까?
  - Data imbalance 문제가 아닐까?
  - 과연 우리 데이터는 올바르게 labeling 되어있는가?

Data

Data imbalance

labeling 을 하면서 느낀점은, 대학 커뮤니티에서는 생각보다 분란글이 적음.

데이터 imbalance 문제를 풀어야함.

일반 글 데이터	분란글 데이터

Problem : Labeling

숭실대학교 커뮤니티 유어슈의 내부 커뮤니티 데이터 사용.

하지만 레이블링 전혀 안되어있음. 그저 망망대해에 텍스트 데이터하나만 들고 목표를 찾아가야함..

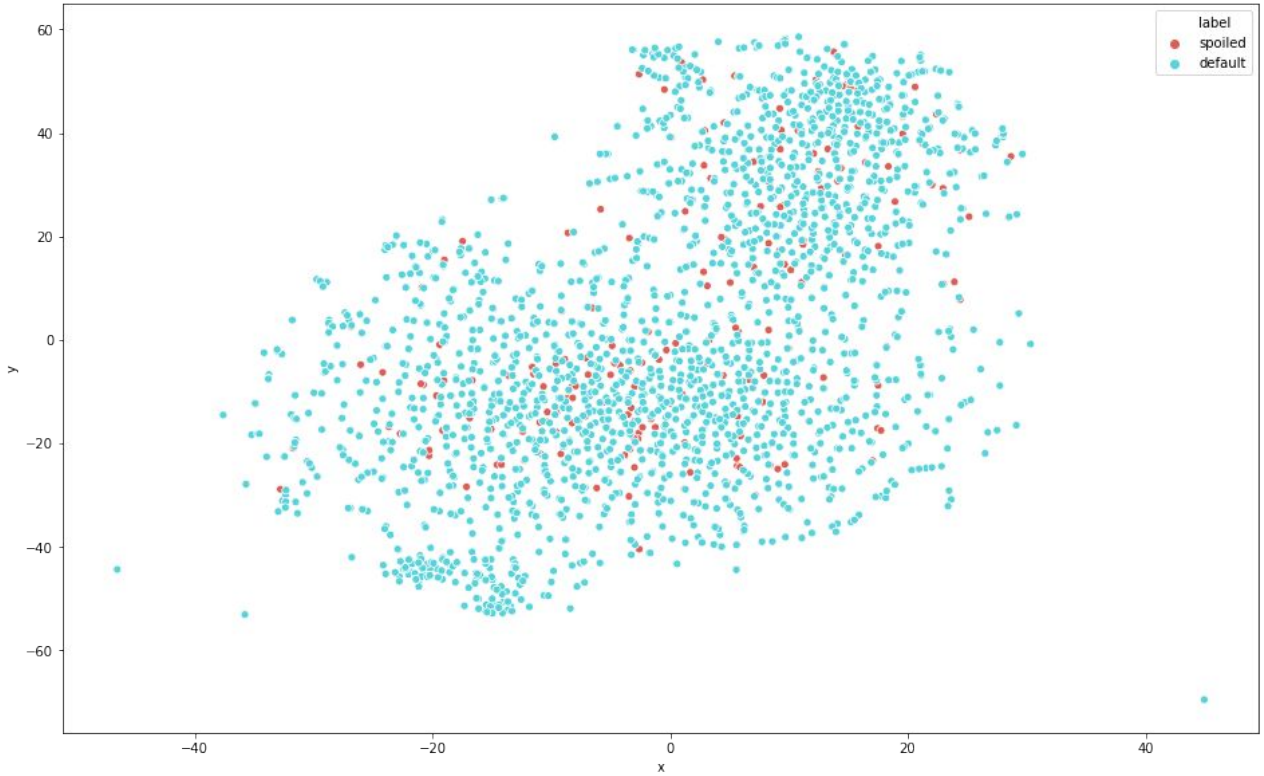
Problem : Labeling

초기에는 3명의 리서치팀 팀원이 자기가 분란글 이라고 생각하는것들을 분란글로 레이블링 하는 형식으로 진행함.

아래 처럼 레이블링 웹을 만들어서 레이블링을 진행.

_id	id	text	class
5c189d35ef5953009bef5190	36292040	텍스트 데이터	Spoiled Default

Problem 1: Labeling

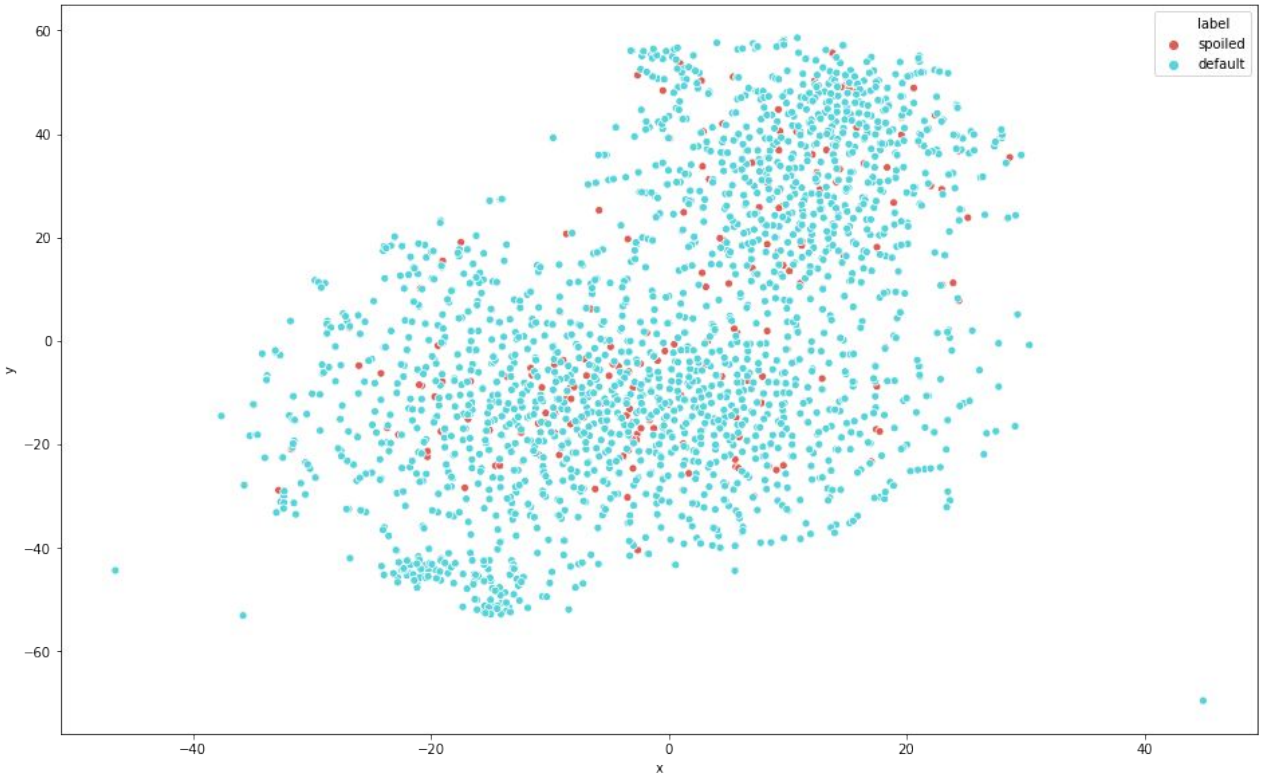


Problem 1: Labeling

실패 요인

1. 너무 적은 분란글 수
2. 서로 다른 분란글의 정의

30% 데이터를 labeling 한 후,
t-sne로 시각화한 데이터 분포.



Over sampling, under sampling

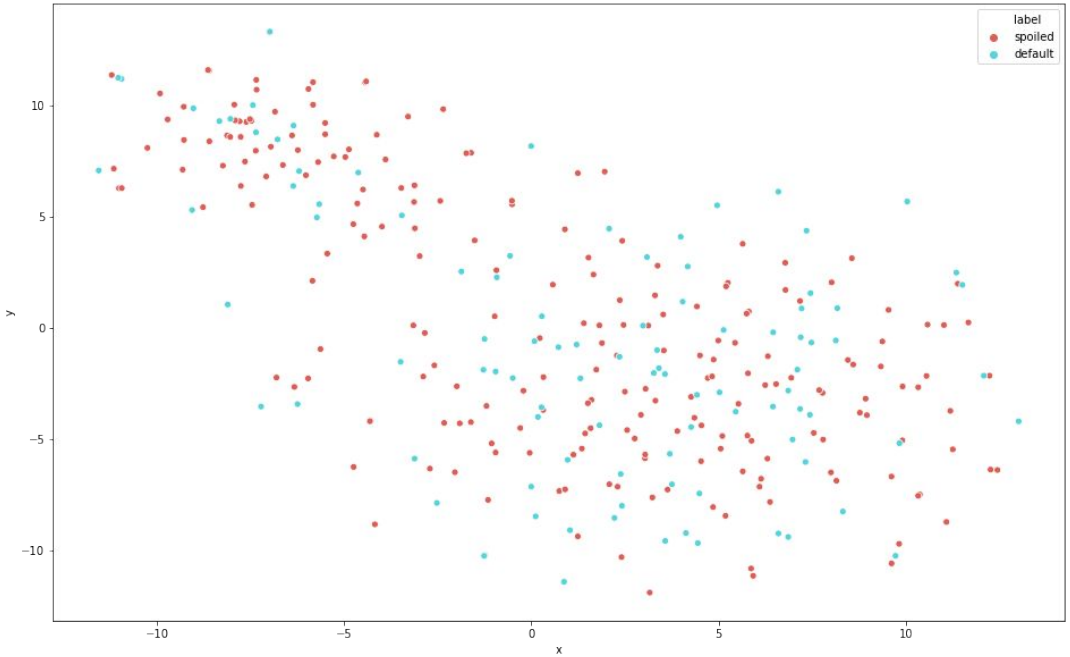
여전히 데이터 자체에 문제가 있어서

under sampling을 한다 해도 문제가

많았음.

애초에 labeling이 잘못된 상태.

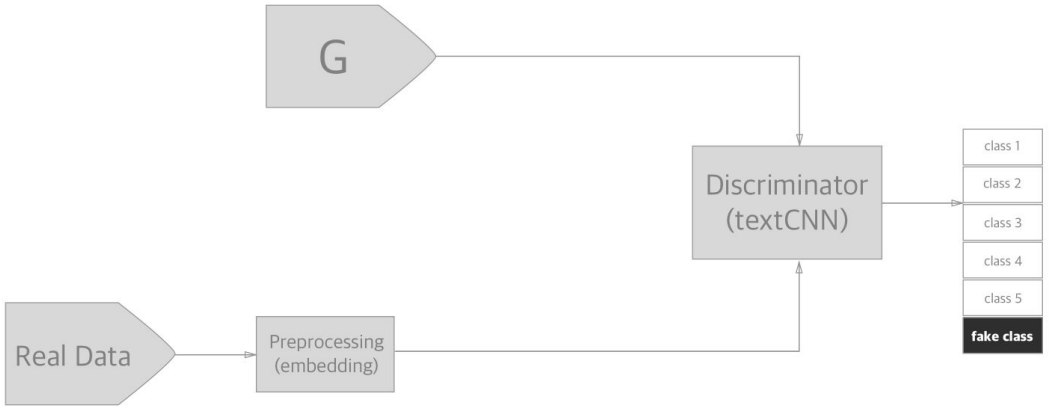
오른쪽 은 random under sampling한 사진



semi supervised learning

label을 자동으로 생성할 수는 없을까?

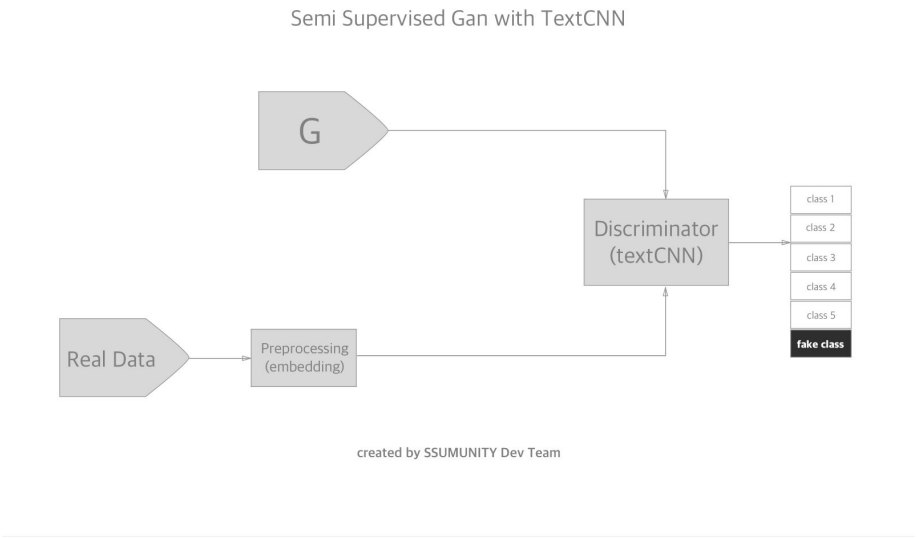
Semi-Supervised-GAN



unlabeled된 데이터가 labeled된 데이터보다 월등히 많아서 (약 1000배 차이) 이 문제를 이미지에서는 종종 사용하던 semi-supervised 방법론을 nlp에 적용해서 해결을 해 보고 싶었다

Semi-supervised learning

gan은 특성상 decision-boundary의 넓이가 좁은 편이고, 커뮤니티 데이터들은 discrete한 인풋들로 구성되어 있어서 NLP에는 적절하지 않은 방법

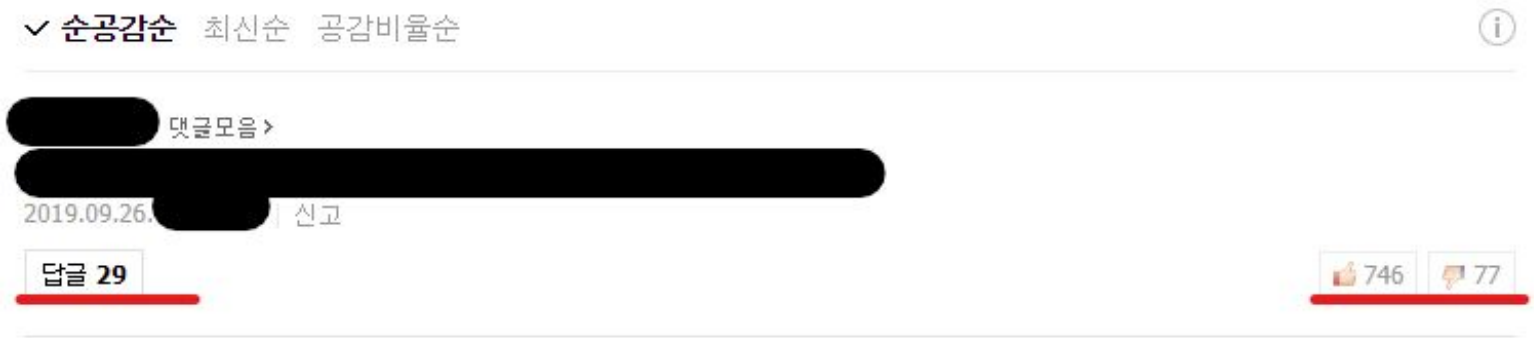


Data!

어떻게하면 분란글을 대량으로 얻을 수 있을까?

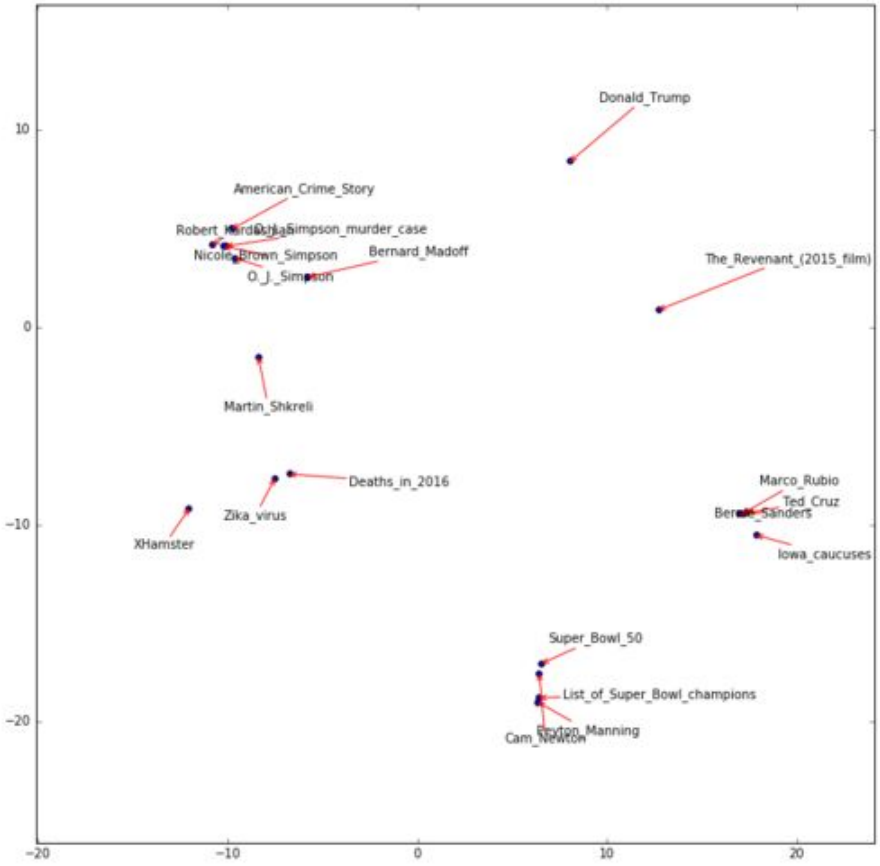
네이버 정치 뉴스 댓글

- 네이버 뉴스 댓글에는 좋아요, 싫어요가 존재.
- 정치뉴스의 경우 좌 vs 우 극명하게 의견이 대립
- 좋아요+싫어요 개수가 많은것은 정치 대부분 분란글



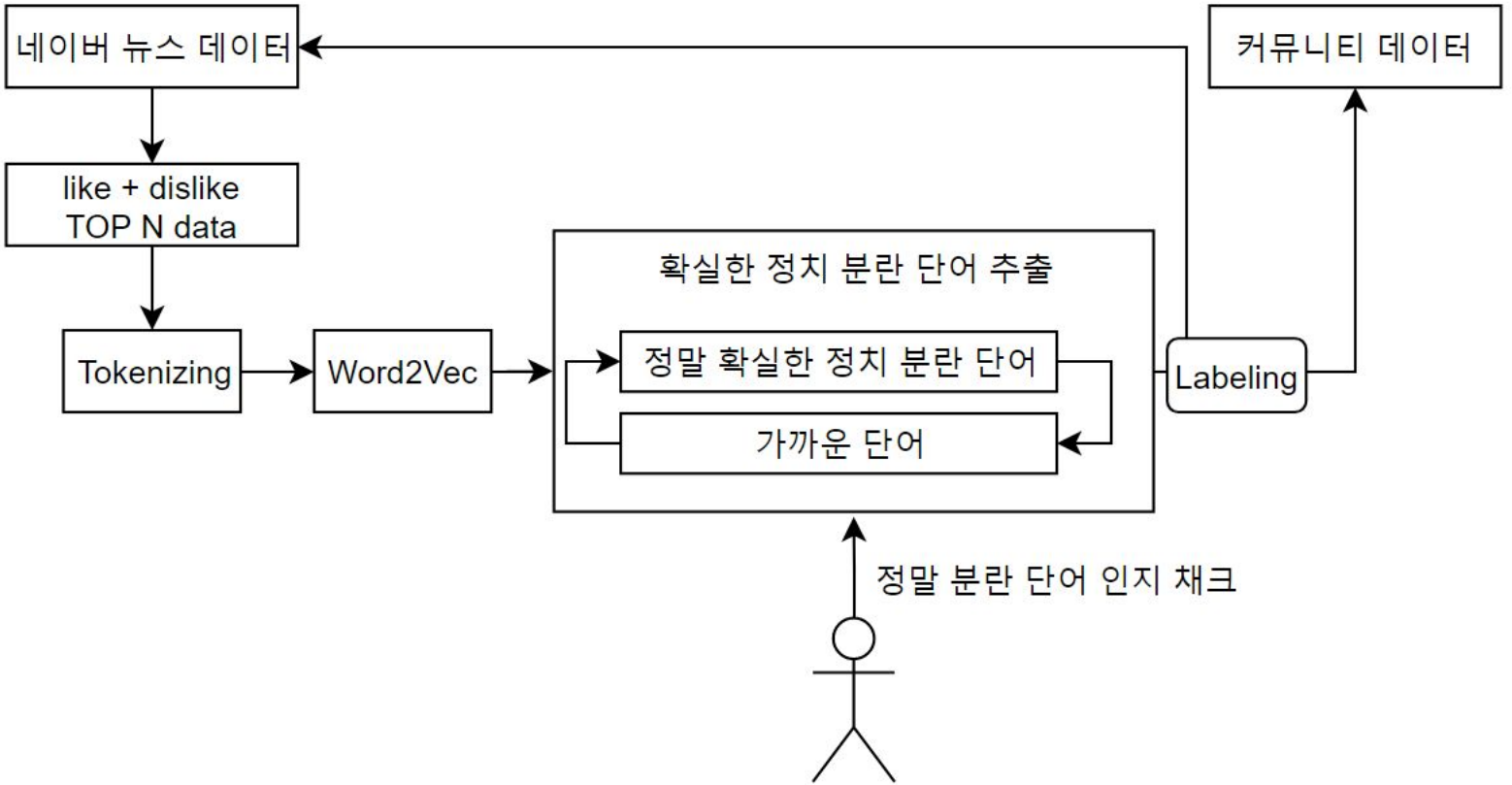
네이버 정치 뉴스 댓글

- 좋아요 + 싫어요 수 에서 많이 등장하는 단어들이 분란을 조장하는 글에 대부분 존재하지 않을까?
- ‘비슷한’ 단어들을 어떻게 모아 볼 수 있을까?
- Word2vec 으로 단어를 vectorizing 한 후, 빈도가 높은 단어들 중 무조건 분란글 이라고 생각되는 단어들로 부터 연관된 단어를 찾으면 되지 않을까?



출처: https://commons.wikimedia.org/wiki/File:2016_02_mini_embedding.png

Data pipeline



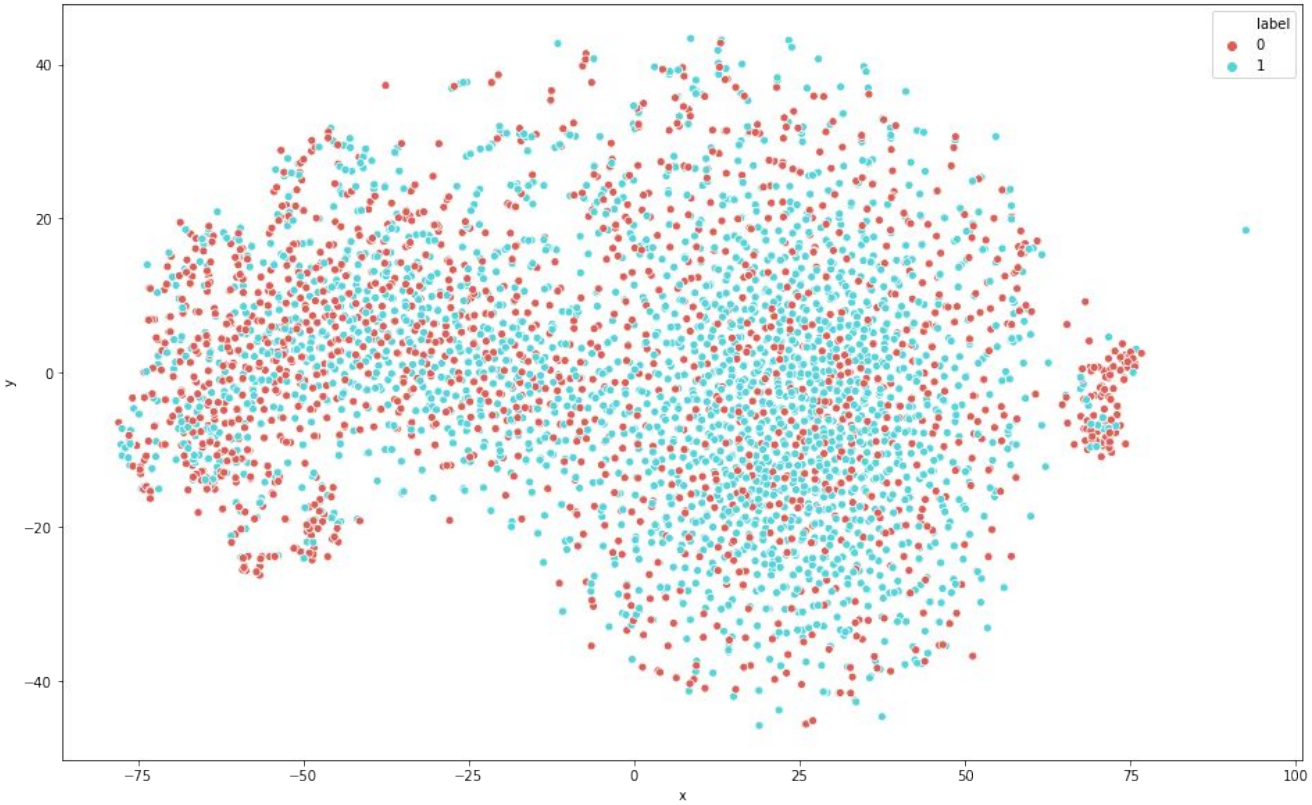
Scope 분리 - 정치 분란글만 먼저 해보자!

네이버 정치 뉴스 댓글

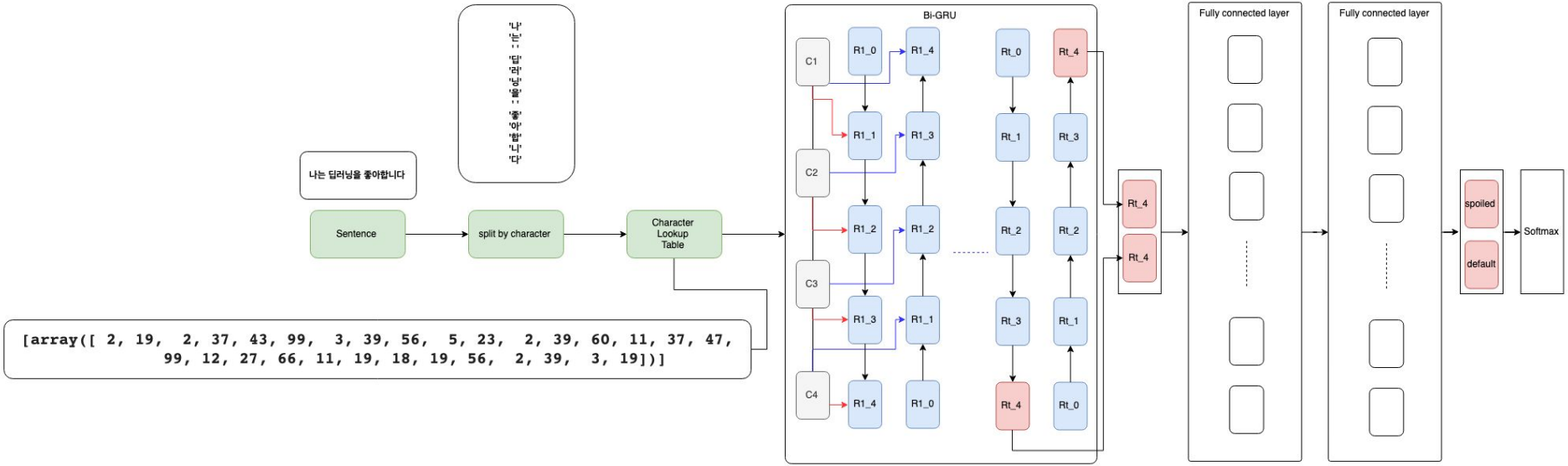
- 많이 본 뉴스의 특정 기간동안의 뉴스를 수집.

https://github.com/jason9693/NNST-Naver-News-for-Standard-and-Technology-Database

tsne 시각화 - 네이버 뉴스 댓글 추가



Model - Tweet2Vec - tensorflow 2.0 refactoring



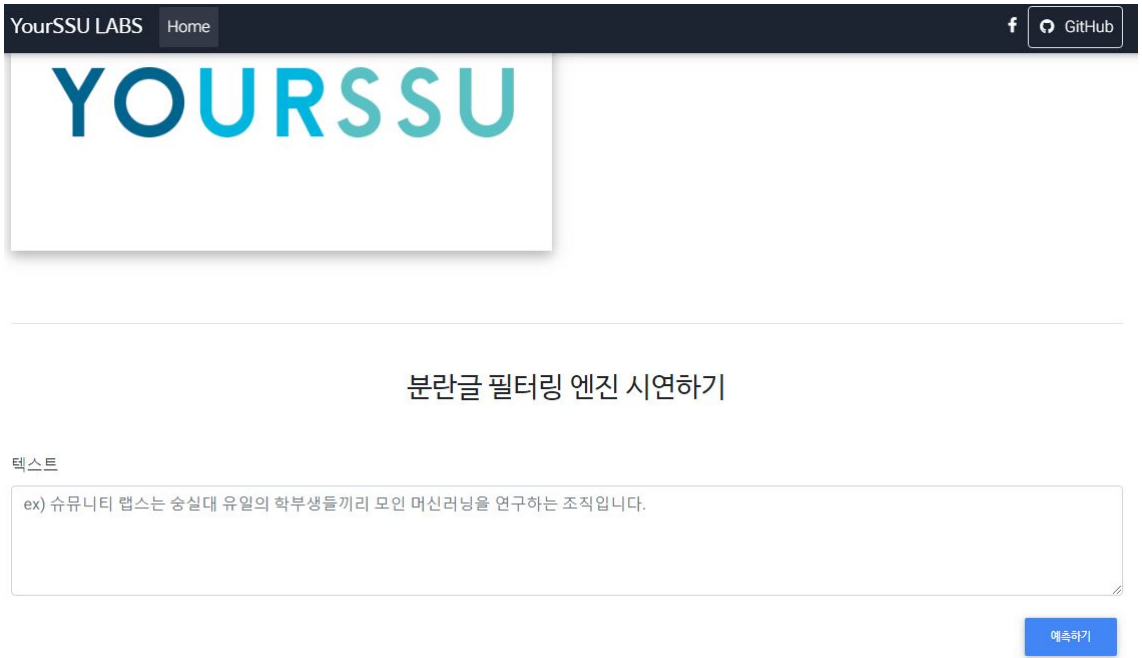
결과

Result

	Tweet2Vec	Tweet2Vec(PreTrained)	Tweet2Vec(with Naver Data)	Tweet2Vec(max len: 1000, with Naver Data)
accuracy	0.798658	0.845638	0.877751	0.848411
epoch	1700	1700	1700	1700
f1 score	0.821429	0.868571	0.888889	0.85446
loss	0.211897	0.319541	0.284464	0.334172
precision	0.821429	0.873563	0.896861	0.842593
recall	0.821429	0.863636	0.881057	0.866667

Web Example

http://coc-filtering.herokuapp.com/



Web Example


분란글 필터링 엔진 시연하기
"지금 숭실존 입장가능?? 자리 널널해요??" 에 대한 예측 결과:
99.44% 의 확률로
일반 글 일 가능성이 높습니다.

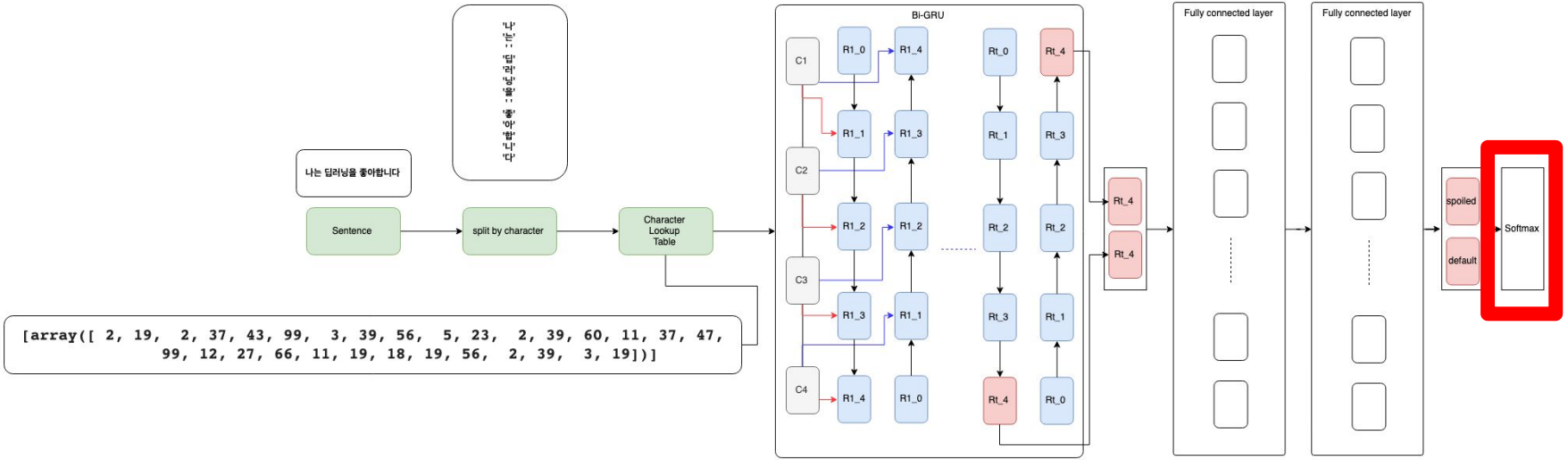
Web Example

"역시[redacted]에[redacted] 언급없네[redacted]완전 딴판이네ㅋㅋ[redacted]논문관련 나온지 꽤 됐는데[redacted]는 조용ㅋ[redacted]도 [redacted]있었는데[redacted]에 언급은[redacted]은 엄청 많았지만[redacted]는 없음ㅋㅋ 역시[redacted]는 일베들이 나대거나 일베스러운 애들이 많이 활동하는곳 인증ㅋㅋ" 에 대한 예측 결과:

98.31% 의 확률로

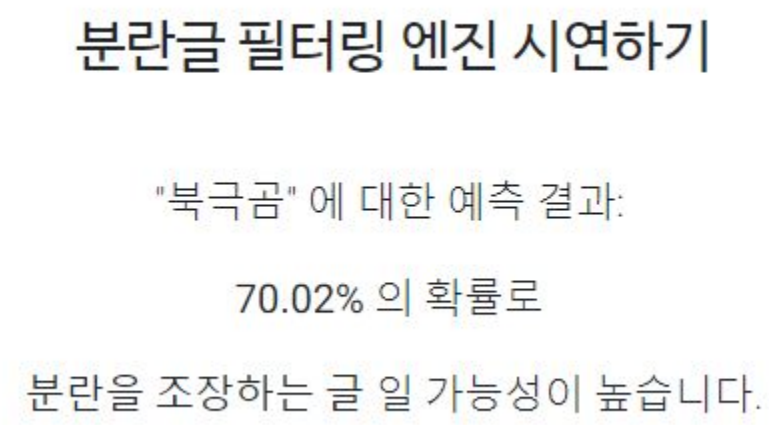
분란을 조장하는 글 일 가능성이 높습니다.

Web Example


나는 딥러닝을 좋아합니다
Sentence
split by character
Character Lookup Table
[array([2, 19, 2, 37, 43, 99, 3, 39, 56, 5, 23, 2, 39, 60, 11, 37, 47, 99, 12, 27, 66, 11, 19, 18, 19, 56, 2, 39, 3, 19])]
Bi-GRU
C1
C2
C3
C4
R1_0
R1_1
R1_2
R1_3
R1_4
Rt_0
Rt_1
Rt_2
Rt_3
Rt_4
Fully connected layer
Fully connected layer
spoiled
default
Softmax

Next step

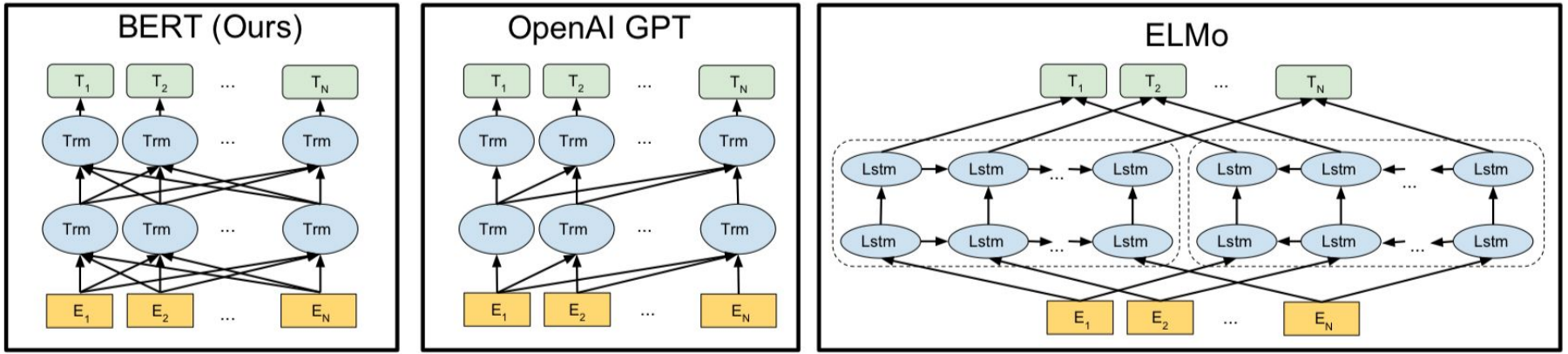
- 정치 댓글 + character based model 의 단점
- character가 비슷한 분란글이 다른 단어를 분란글로 분류.
- 해당 단어에 대한 데이터를 augmentation 하여 많은 데이터를 만들어 model에 넣어 해당 단어가 분란글로 분류되지 않도록 모델을 유도.



https://mk.kakaocdn.net/dn/if-kakao/conf2019/%EB%B0%9C%ED%91%9C%EC%9E%90%EB%A3%8C_2019/T10-S01.pdf

Next step

- language model with transformer based model + byte pair encoding



https://github.com/strutive07/transformer-tensorflow2.0

QnA

Thank you