

딥러닝 기반의 청와대 국민청원 주제분류

서울시립대학교 도시빅데이터융합학과 빅데이터딥러닝 paring 연구결과 보고

도시빅데이터융합학과 박사과정 G20214604 장홍진
통계학과 석박사통합과정 G201958013 신지은

1. 연구주제 선정

- 1-1. 연구목적 및 필요성
- 1-2. 선행연구

2. 국민청원 데이터

- 2-1. 데이터 수집
- 2-2. 데이터 전처리
- 2-3. EDA

3. 딥러닝 기반의 주제분류

- 3-1. CNN 기반의 카테고리 분류
- 3-2. BERT 기반의 카테고리 분류

4. 결론

1. 연구목적 및 필요성

연구배경

- 국민청원은 현 사회의 이슈를 실시간으로 반영하기 때문에, 특정 시대 국민의 관심사가 요약된 데이터로 볼 수 있다.
- 민원 데이터 성격인 국민청원 데이터를 분석함으로써 향후 민원 데이터에서의 응용이 가능할 것이다.
- 실제 데이터에 대한 딥러닝 기반의 분류 모델링 적용 가능성을 연구하고자 한다.

연구목적

- 사전 훈련을 한 모델과 사전훈련을 하지 않은 모델의 성능을 비교한다.
- CNN과 BERT의 구현을 시도해 본다.
- CNN과 BERT를 이용한 국민청원 주제분류 모델링과 그 성능을 비교한다.

1. 연구목적 및 필요성

선행연구

1. 우윤희, & 김현희. (2019). K-means 클러스터링과 토픽 모델링을 기반으로 한 국민청원 사이트의 카테고리 재구성. *한국정보처리학회 학술대회논문집*, 26(1), 302-305.

- 추천순 상위 1,500개의 청원글 수집
- K-평균 알고리즘으로 유사한 글을 그룹화하여 군집별 핵심단어를 추출함 → 상위 카테고리에 올 수 있는 주제를 확인
- 토픽모델링 (LDA; Latent Dirichlet Allocation)으로 원래 17개의 주제를 14개의 카테고리로 재분류

1. 연구목적 및 필요성

선행연구

2. 우윤희, & 김현희. (2020). 국민청원 주제 분석 및 딥러닝 기반 답변 가능 청원 예측. *정보처리학회논문지/소프트웨어 및 데이터 공학*, 9(2), 2.

- 추천순 상위 1,500개의 청원글 수집
- 청원 동의 수가 20만 이상인 청원 글을 답변 가능한 청원으로 정의하여 LSTM으로 분류
- 변수에 본문, 글의 길이, 카테고리, 품사 (체언, 용언, 독립언, 수식언)의 비율을 추가
- 예측 성능으로 0.9 이상의 F1-score를 보임

1. 연구목적 및 필요성

선행연구

3. 황상흠, & 김도현. (2020). 한국어 기술문서 분석을 위한 BERT 기반의 분류모델. *한국전자거래학회지*, 25(1), 203-214.

- SKTBrain에서 제공하는 한국어 BERT모델을 활용하여 기술 문서로부터 자동적으로 문서 특징을 추출
- 국가 R&D 과제 중분류 코드 분류

1. 연구목적 및 필요성

선행연구

4. 하지은, 신현철, & 이준기. (2017). RandomForest 와 XGBoost 를 활용한 한국어 텍스트분류: 서울특별시 응답소 민원 데이터를 중심으로. *한국빅데이터논문지 제, 2(2)*.

- 서울특별시 응답소 민원데이터의 17,700건을 대상으로 민원 카테고리 분류
- 카테고리가 업무 처리 부서로 나뉘어져 있어 처리부서 명칭이 변경될 때마다 적절한 카테고리를 새로 부여해야 하는 어려움이 있음

데이터 수집

	청원 진행	참여인 원	카테 고리	청원시작	청원마감	제목	내용
0	청원 종료	299	1	2018-05-01	2018-05-31	인천 초동생 살해범 징역 13년이라니요..	네이버 뉴스를 보다 너무 어이가 없어서 청원을 올립니다. 하루가 멀다하고 올라오는 ...
1	청원 종료	1676	2	2018-12-13	2019-01-12	우리 거북이 아이들의 미래를 부디 열어주세요	안녕하세요 4세 발달지연 아이를 키우고있는 대한민국의 평범한 어머니입니다 \n\n...
2	청원 종료	384	3	2018-03-19	2018-04-18	단역배우자매를 성폭행한 가해자 12명을 다시 수사하고 담당경찰관을 파면해주세요	10년전의 일이고 고소를 취한한 내용이지만\n이자가매가 그일로 인해 자살하고 아버지마저...
3	청원 종료	188	3	2018-08-26	2018-09-25	은혜로 독사와 그 주변의 관계자들을 인터넷에 문의하여 체포해 주세요	은혜로 독사인 신복사에게 포임을 당해 끌려간 사람들을 구하고 친근하고 그 주변에 ...
4	청원 종료	177	2	2017-11-17	2017-12-17	대구시 우등기 교육감을 증도에 교체할 수 있는 제도보완을 청원,제안합니다	최근 1~2주 사이 우등기 대구시교육감에 대한 생각을 적지 않을 수 없어 몇 자 ...
...
7842	청원 종료	195	2	2017-09-27	2017-10-27	대입 수능 성적 정보의 구체적이고 투명한 공개를 요청 드립니다.	대입 수험생을 둔 가정에서 많은 분들이 아쉬워하는 교육과정평가원의 수능 및 모의평가...
7843	청원 종료	222	4	2018-04-18	2018-05-18	국회의원전주소사	이번국회의원피검거관지침과관련전주소사하여잘못된관행을바로잡았으면합니다이렇게해야정지개혁...
7844	청원 종료	211	1	2018-02-22	2018-03-24	안전하지 않는 원전 운영허가 취소를 해 주세요	울진 한울원전 4호기는 상업운전 초기부터 심각한 전열판 손상 이 있었습니다. 그리고...
7845	청원 종료	3343	2	2018-01-04	2018-02-03	서남대학교 특별편입 반대 단국대학교 천안캠퍼스.	여러분 혹시 지금 서남대학교 폐교에대한 기사를 보셨나요? \n\n최근전캠본문에겐 해...
7846	청원 종료	145	3	2018-03-15	2018-04-14	고 장자영님 사건이나 단역 여배우 자살 사건에 대해 청원 커트라인을 낮춰주시기를 청...	이런 귀중한 문제는 20만이 넘어가야 청원 승락되는 것에서 좀 커트라인을 낮춰서 다...

8

2. 국민청원 데이터

데이터 전처리

1. 1음절 명사 제외

: 형태소 분석 후 명사만 선별하여 학습하되 의미 파악이 어려운 1음절 명사 제외

```
1 from pykosspacing import Spacing
2
3 save_text = []
4 for idx, contents in enumerate(train_data['내용']):
5     pre_text = re.sub("[^가-힣ㄱ-ㅎㅏ-ㅣ#s]", "", contents)
6     spacing = Spacing()
7     save_text.append(spacing(pre_text))
8
9 train_data['내용2'] = save_text
10 train_data
```

[코드 1] 명사 추출 코드 예시

```
1 okt = Okt()
2 noun_list = []
3
4 for idx, content in enumerate(train_data['내용2']):
5     noun = okt.nouns(content)
6
7     for i, word in enumerate(noun):
8         if len(word) <= 1 :
9             noun.pop(i)
10
11     noun_list.append(noun)
12
13 train_data['내용3'] = noun_list
```

[코드 2] 1음절 명사 제외 코드 예시

2. 국민청원 데이터

데이터 전처리

2. 단어 수가 10개 이하인 청원글은 제외
: 의미 파악이 어려운 글은 제외함
3. 청원 동의 100건 미만의 청원글은 제외
: 동의 수가 낮은 글은 의미를 파악하기 어렵다고 판단함

```
word_cnt = train_data['내용3'].astype(str).apply(lambda x: len(x.split(' ')))
train_data['word_cnt'] = word_cnt

drop_ind=[]
for ind, cnt in enumerate(train_data['word_cnt']):
    if cnt <= 10:
        drop_ind.append(ind)

train_data.drop(train_data.index[drop_ind], inplace=True)
train_data
```

[코드 3] 단어 수 10개 이하 청원글 제외 코드 예시

2. 국민청원 데이터

데이터 전처리

4. 빈도 기준 상위 5개의 카테고리를 대상으로 함

: 데이터 학습량의 과부하로 인한 데이터 축소와 상위 5개 카테고리에 대해 학습을 시도함

■ 전처리 전 카테고리

정치개혁	2,449	행정	907
인권/성평등	2,355	경제민주화	759
기타	1,880	일자리	691
육아/교육	1,858	반려동물	433
안전/환경	1,683	미래	416
보건복지	1,464	저출산/고령화대책	185
교통/건축/국토	1,384	성장동력	136
문화/예술/체육/언론	1,086	농산어촌	78
외교/통일/국방	908	합계	18,672

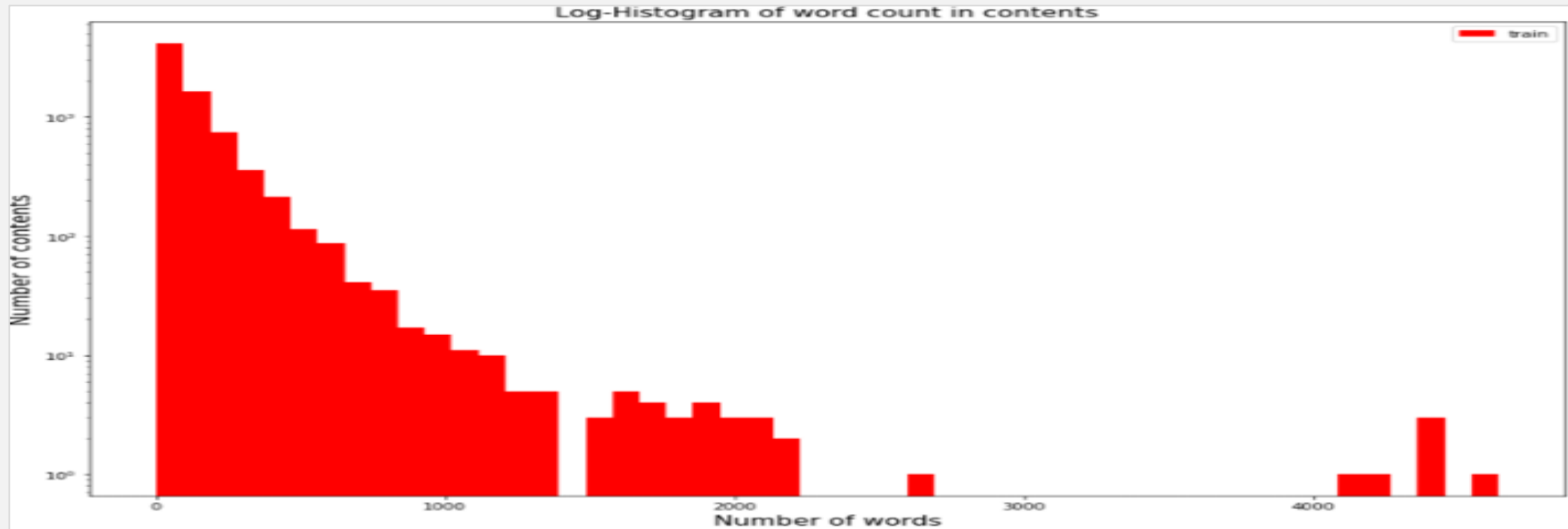
■ 전처리 후 상위 5개 카테고리

정치개혁	2,220
인권/성평등	2,219
육아/교육	1,823
안전/환경	1,644
보건복지	1,440
합계	9,346

훈련 데이터 : 시험 데이터
= 8 : 2

2. 국민청원 데이터

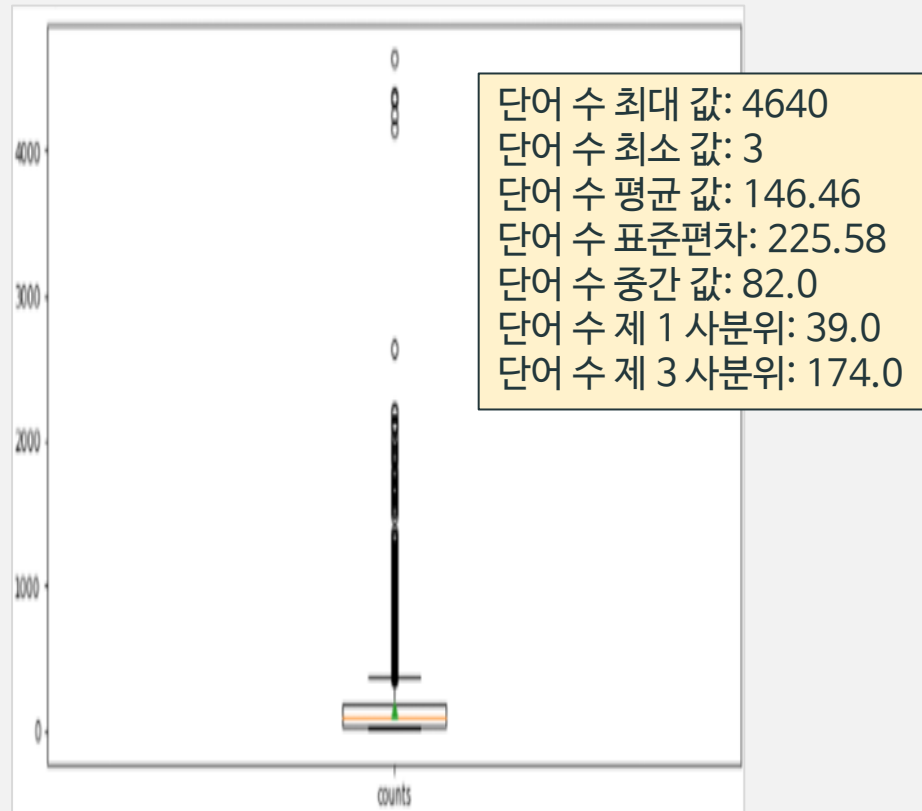
EDA



[그림 1] 훈련 데이터에 대한 단어 수 분포

2. 국민청원 데이터

EDA



[그림 2] 훈련 데이터에 대한 단어 수 box plot



[그림 3] 훈련 데이터에 대한 word cloud

3. 딥러닝 기반의 주제분류

CNN

- 합성곱 신경망 (CNN) 모듈과 한국어 처리를 위한 KoNlpy 활용함.
- Tensorflow의 keras기반 모델 적용함.
- 사전에 OKT 토큰화를 적용함.

```
class CNNClassifier(tf.keras.Model):  
    def __init__(self, **kwargs):  
        super(CNNClassifier, self).__init__(name=kwargs['model_name'])  
        self.embedding = layers.Embedding(input_dim=kwargs['vocab_size'],  
                                           output_dim=kwargs['embedding_size'])  
        self.conv_list = [layers.Conv1D(filters=kwargs['num_filters'],  
                                         kernel_size=kernel_size,  
                                         padding='valid',  
                                         activation=tf.keras.activations.relu,  
                                         kernel_constraint=tf.keras.constraints.MaxNorm(max_value=3.))  
                           for kernel_size in [3,4,5]]  
        self.pooling = layers.GlobalMaxPooling1D()  
        self.dropout = layers.Dropout(kwargs['dropout_rate'])  
        self.fc1 = layers.Dense(units=kwargs['hidden_dimension'],  
                                 activation=tf.keras.activations.relu,  
                                 kernel_constraint=tf.keras.constraints.MaxNorm(max_value=3.))  
        self.fc2 = layers.Dense(units=kwargs['output_dimension'],  
                                 activation=tf.keras.activations.sigmoid,  
                                 kernel_constraint=tf.keras.constraints.MaxNorm(max_value=3.))  
  
    def call(self, x):  
        x = self.embedding(x)  
        x = self.dropout(x)  
        x = tf.concat([self.pooling(conv(x)) for conv in self.conv_list], axis=-1)  
        x = self.fc1(x)  
        x = self.fc2(x)  
  
        return x
```

[코드 4] CNN 모델 코드

3. 딥러닝 기반의 주제분류

CNN

- 하이퍼 파라미터 설정과 학습결과

- 하이퍼 파라미터 설정

Options	Values
input	국민청원 글
Label	카테고리
Activation	Softmax
Loss function	Multi-class cross entropy
Optimzier	Adam
Epoch	3
Batch size	128
Dropout	0.2
Hidden dimension	250
Padding size	200

- 훈련 데이터에 대한 모델 훈련 결과 (정확도)

```
Epoch 00001: val_accuracy improved from -inf to 0.16906, saving model to ./data_out/cnn_class
Epoch 2/30
46/46 [=====] - 54s 1s/step - loss: -3934.8025 - accuracy: 0.1799 -

Epoch 00002: val_accuracy did not improve from 0.16906
Epoch 3/30
46/46 [=====] - 53s 1s/step - loss: -34377.8438 - accuracy: 0.1799 -

Epoch 00003: val_accuracy did not improve from 0.16906
```

- 시험 데이터에 적용 결과 (정확도)

```
58/58 [=====] - 5s 85ms/step - loss: -439.2676 - accuracy: 0.1695
[-439.2676086425781, 0.16946399211883545]
```

→ 전체적인 정확도가 높지 않음

3. 딥러닝 기반의 주제분류

BERT

- 기본 코드는 SKT Brain의 KoBERT (<https://github.com/SKTBrain/KoBERT>)를 사용함.
- 작업 환경은 구글 코랩 (google colab)에서 GPU를 사용함.
- 사전에 OKT토큰화를 한 CNN과는 다르게 BERT 토큰라이저를 사용함.

```
# 기본 Bert tokenizer 사용
tokenizer = get_tokenizer()
tok = nlp.data.BERTSPTokenizer(tokenizer, vocab, lower=False)

using cached model

class BERTDataset(Dataset):
    def __init__(self, dataset, bert_tokenizer, max_len,
                 pad, pair):
        transform = nlp.data.BERTSentenceTransform(
            bert_tokenizer, max_seq_length=max_len, pad=pad, pair=pair)
        self.sentences = [transform(i) for i in dataset['내용']]
        self.labels = [np.int32(i) for i in dataset['카테고리']]
    def __getitem__(self, i):
        return (self.sentences[i] + (self.labels[i], ))
    def __len__(self):
        return (len(self.labels))
```

[코드 5] BERT 토큰라이저와 모델

```
# 모델 학습 시작
for e in range(num_epochs):
    train_acc = 0.0
    model.train()
    for batch_id, (token_ids, valid_length, segment_ids, label) in enumerate(tqdm(train_dataloader)):
        optimizer.zero_grad()
        token_ids = token_ids.long().to(device)
        segment_ids = segment_ids.long().to(device)
        valid_length= valid_length
        label = label.long().to(device)
        out = model(token_ids, valid_length, segment_ids)
        loss = loss_fn(out, label)
        loss.backward()
        torch.nn.utils.clip_grad_norm_(model.parameters(), max_grad_norm) # gradient clipping
        optimizer.step()
        scheduler.step() # Update learning rate schedule
        train_acc += calc_accuracy(out, label)
        if batch_id == batch_size :
            print("epoch {} batch id {} loss {} train acc {}".format(e+1, batch_id+1, loss.data.cpu().numpy(), train_acc / (batch_id+1)))
            print("epoch {} train acc {}".format(e+1, train_acc / (batch_id + 1)))
```

[코드 6] 훈련데이터 학습을 위한 코드

3. 딥러닝 기반의 주제분류

BERT

- 하이퍼 파라미터 설정과 학습결과

- 하이퍼 파라미터 설정

Options	Values
input	국민청원 글
Label	카테고리
Activation	Softmax
Loss function	Multi-class cross entropy
Optimier	Adam
Epoch	5
Batch size	124
Dropout	0.2
Hidden size	18
Maximum length	30

- 훈련 데이터에 대한 모델 훈련 결과 (정확도)

```
100%|██████████| 61/61 [00:26<00:00, 2.27it/s]
0%|          | 0/61 [00:00<?, ?it/s]epoch 1 train acc 21.950819672131146
100%|██████████| 61/61 [00:27<00:00, 2.24it/s]
0%|          | 0/61 [00:00<?, ?it/s]epoch 2 train acc 21.459016393442624
100%|██████████| 61/61 [00:27<00:00, 2.23it/s]
0%|          | 0/61 [00:00<?, ?it/s]epoch 3 train acc 21.459016393442624
100%|██████████| 61/61 [00:27<00:00, 2.23it/s]
0%|          | 0/61 [00:00<?, ?it/s]epoch 4 train acc 21.57377049180328
100%|██████████| 61/61 [00:27<00:00, 2.23it/s]epoch 5 train acc 21.770491803278688
```

3. 딥러닝 기반의 주제분류

BERT


- 시험 데이터에 대한 학습결과

- 시험 데이터에 적용 결과 (정확도)

```
[32] model.eval() # 평가 모드로 변경
test_acc = 0.0

for batch_id, (token_ids, valid_length, segment_ids, label) in enumerate(tqdm_notebook(test_dataloader)):
    token_ids = token_ids.long().to(device)
    segment_ids = segment_ids.long().to(device)
    valid_length= valid_length
    label = label.long().to(device)
    out = model(token_ids, valid_length, segment_ids)
    test_acc += calc_accuracy(out, label)
    print("test acc {}".format(test_acc / (batch_id+1)))
```

시험 데이터를 나누어 정확도 측정



```
test acc 27.0
test acc 24.0
test acc 25.666666666666668
test acc 27.5
test acc 26.2
test acc 25.5
test acc 25.285714285714285
test acc 25.625
test acc 25.444444444444443
test acc 26.1
test acc 26.454545454545453
test acc 26.0
test acc 25.76923076923077
test acc 25.571428571428573
test acc 25.266666666666666
test acc 23.875
```

시험 데이터의 정확도 평균: 25.70

4. 결론

분석 결과

- 실험 환경의 메모리 부족으로 인해 모델 학습을 충분히 시키지 못한 점이 아쉽음.
- 가능한 환경 상에서 사전 훈련을 하지 않은 CNN보다 사전훈련을 한 BERT의 정확도가 더 높았음.
- 같은 단어를 포함한 글이 다른 카테고리를 갖는 경우가 많아 분류의 정확도가 낮은 것으로 예상됨.

청원진행	참여인원	카테고리	청원시작	청원마감	제목	내용
청원종료	469.0	기타	2019-03-06	2019-04-05	코리아나 호텔 방용훈회장 그 자녀 범죄 행위 대해	피디수첩 통해 방송 코리아나 호텔 방용훈회장 그 자녀 범죄 또한 준 검 세력 대해...
청원종료	174.0	인권/성평등	2019-03-06	2019-04-05	코리아나 호텔 사모님 죽음 대한 수사 고인	수첩 시청 국민 라면 저 잠 고인 명복 극단 선택 이유 시청 이 사건 누가 존 담...
청원종료	598.0	기타	2019-03-06	2019-04-05	조선일보 방용훈 사장 처 고이 미란 죽음 제대로 수사	수첩 방용훈 회장 처 고이 미란 죽음 대한 내용 대한민국 사법부 권력 무를 국민 ...
청원종료	142.0	정치개혁	2019-03-06	2019-04-05	고 씨 죽음 내몬 패륜 씨 일가 사건 재 수사	수첩 보고 씨 일가 악행 경악 금치 고 씨 죽음 택할수 씨 일가 악행 재 수사 또...

보완점

- 충분한 학습이 부족하여 CNN와 BERT의 정확도 비교가 공정한가에 대한 문제를 고려할 필요가 있음.
- 카테고리 재분류 후 분류문제를 고려할 필요가 있음.

감사합니다.

1. 우윤희, & 김현희. (2019). K-means 클러스터링과 토픽 모델링을 기반으로 한 국민청원 사이트의 카테고리 재구성. *한국정보처리학회 학술대회논문집*, 26(1), 302-305.
2. 우윤희, & 김현희. (2020). 국민청원 주제 분석 및 딥러닝 기반 답변 가능 청원 예측. *정보처리학회논문지/소프트웨어 및 데이터 공학*, 9(2), 2.
3. 황상흠, & 김도현. (2020). 한국어 기술문서 분석을 위한 BERT 기반의 분류모델. *한국전자거래학회지*, 25(1), 203-214.
4. 하지은, 신현철, & 이준기. (2017). RandomForest 와 XGBoost 를 활용한 한국어 텍스트분류: 서울특별시 응답소 민원 데이터를 중심으로. *한국빅데이터논문지 제*, 2(2).
5. 유원준. (2021). 딥 러닝을 이용한 자연어 처리 입문.