Korean Sentence Compression Model

: Language Scoring with Morphological Analysis and Utilization of Perplexity

(한국어 문장 압축 모델: 형태소 분석을 통한 언어 점수 부여 및 perplexity 활용)

소프트웨어융합캡스톤디자인
2023.06.09
소프트웨어융합학과 2020105714 오유솔

구성

CONTENTS

01 과제 설계 배경 및 필요성

■ 과제 설계 배경 및 필요성

국내 정보 탐색 방법

: 네이버 블로그, 유튜브 콘텐츠, SNS 게시글 등

모바일 기기를 통한 웹, 모바일 콘텐츠

↓

작은 화면을 통해 사용자가

“중요한 정보를 **빠르게** 획득”할 수 있도록 하는 것이,

각 모바일 콘텐츠의 경쟁력이 됨

■ 과제 설계 배경 및 필요성

문장 요약 VS 문장 압축

주어진 문장에 대해 의미 파악

→ 중요 요소를 추출

→ 핵심 내용을 포함한 간결한 문장 “**생성**”

- “생성” (X)
- 주어진 문장에서 중요하지 않은 정보를 **“제거”**하여 문장의 길이를 줄임

■ 과제 설계 배경 및 필요성

문장 요약 VS 문장 압축

주어진 문장에 대해 의미 파악

→ 중요 요소를 추출

→ 핵심 내용을 포함한 간결한 문장 “**생성**”

- “생성” (X)
- 주어진 문장에서 중요하지 않은 정보를 **“제거”**하여 문장의 길이를 줄임

생성 모델의 경우, 논리적으로 맞다고 생각되면,
사실이 아님에도 사실이라고 문장을 생성해내는
Hallucination 문제 존재

⇒ ‘문장 압축 모델’ 구현

02

문장 압축 모델

■ 문장 압축 모델

(EX) 주어진 문장 :

'수상한 파트너'가 한순간도 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다

1. 압축 문장 후보군 생성 : 주어진 문장에서 연속된 n개의 단어를 어절단위로 [MASK]화하여 생성

```
n = 1
[MASK] 한순간도 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다
'수상한 파트너'가 [MASK] 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다
'수상한 파트너'가 한순간도 [MASK] 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다 ...
n = 2
[MASK] [MASK] 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다
'수상한 파트너'가 [MASK] [MASK] 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다 ...
n = 3
[MASK] [MASK] [MASK] 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다
'수상한 파트너'가 [MASK] [MASK] [MASK] 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다 ...
```

2. KoBERT 모델이 [MASK] 토큰 위치에 적합한 단어를 예측

: 예측의 정확도가 높을수록 (예측률이 좋을수록) 해당 문장은 문맥적으로 올바른 문장이다

2. KoBERT 모델이 [MASK] 토큰 위치에 적합한 단어를 예측

: 예측의 정확도가 높을수록 (예측률이 좋을수록) 해당 문장은 문맥적으로 올바른 문장이다

3. Perplexity(문장의 혼잡도) 계산 : [MASK] 토큰 위치에 대한 예측 값의 확률 분포를 계산하여 구함

(perplexity가 낮을수록 문맥적으로 더 옳은 문장)

2. KoBERT 모델이 [MASK] 토큰 위치에 적합한 단어를 예측

: 예측의 정확도가 높을수록 (예측률이 좋을수록) 해당 문장은 문맥적으로 올바른 문장이다

3. Perplexity(문장의 혼잡도) 계산 : [MASK] 토큰 위치에 대한 예측 값의 확률 분포를 계산하여 구함

(perplexity가 낮을수록 문맥적으로 더 옳은 문장)

4. 각 압축 문장 후보에 언어정보점수 부여

: 한글 문장의 특성에 맞게, 주요 형태소 및 문장 주성분에 언어정보점수 부여

압축 문장 후보 : '수상한 파트너'가 한순간도 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 [MASK] [MASK] 만들었다 perplexity score : 3.4239870728924 87e-05. 언어정보점수 : 0.012800999999999995

최종 점수 : 4.383045852009671e-07

압축 문장 후보 : '수상한 파트너'가 한순간도 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 [MASK] [MASK] perplexity score : 3.3794618502724916e-05. 언어정보점수 : 0.013800999999999996

최종 점수 : 4.663995299561064e-07

압축 문장 후보 : [MASK] [MASK] [MASK] 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다 perplexity score : 3.332997584948316e-05. 언어정보점수 : 0.0038

최종 점수 : 1.2665390822803601e-07

5. 각 압축 문장 후보의 최종 점수 계산 : 최종점수는 ‘perplexity X 언어정보점수’로 계산

5. 각 압축 문장 후보의 최종 점수 계산 : 최종점수는 ‘perplexity X 언어정보점수’로 계산

6. 최종 압축 문장 선택 : 최종 점수가 가장 낮은 문장 선택

	압축 문장 후보	최종 점수	n-gram
0	눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다	0.000000114081554	2
1	한순간도 눈을 뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다	0.000000118026275	1
2	뗄 수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다	0.000000126653908	3
3	수 없는 전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다	0.000000145820014	4
4	전개 속 충격 반전을 펼치며 안방극장을 쑥대밭으로 만들었다	0.000000146963838	6
...	...	...	...
90	'수상한 파트너'가 한순간도 눈을 뗄 수 만들었다	0.000000552655131	8
91	'수상한 파트너'가 한순간도 눈을 뗄 만들었다	0.000000553194578	9
92	'수상한 파트너'가 한순간도 눈을 만들었다	0.000000555698274	10
93	'수상한 파트너'가 한순간도 눈을 뗄	0.000000556669450	10
94	'수상한 파트너'가 한순간도 눈을 뗄 수	0.000000558039736	9

95 rows × 3 columns

03

모델 성능 평가

■ 모델 성능 평가

1. 문장 압축 정확도가 좋은 모델 선택

(EX) 주어진 문장 : 신바람 농구가 살아난 **SK**가 **오리온**에 **승리**를 거뒀다

다국어로 학습된 BERT 모델

"신바람 농구가 살아난 승리를 거뒀다"

```
압축 문장 후보 : [MASK] 농구가 살아난 SK가 오리온에 승리를 거뒀다
Perplexity : 6.712080224957617e-09, 언어정보점수 : 0.0026000000000000003
최종 점수 : 0.0026000067120802253

압축 문장 후보 : 신바람 [MASK] 살아난 SK가 오리온에 승리를 거뒀다
Perplexity : 2.8498345350236320-10, 언어정보점수 : 0.0016
최종 점수 : 0.0016000002849834536

압축 문장 후보 : 신바람 농구가 [MASK] SK가 오리온에 승리를 거뒀다
Perplexity : 3.1396540029514026e-09, 언어정보점수 : 0.0026000000000000003
최종 점수 : 0.0026000031336540033

압축 문장 후보 : 신바람 농구가 살아난 [MASK] 오리온에 승리를 거뒀다
```

	압축 문장 후보	최종점수
0	신바람 살아난 SK가 오리온에 승리를 거뒀다	0.0016000000284983
1	신바람 농구가 살아난 오리온에 승리를 거뒀다	0.0020000000504691
2	신바람 농구가 살아난 SK가 오리온에 거뒀다	0.0021000000028024
3	신바람 농구가 살아난 SK가 승리를 거뒀다	0.0021000000467064
4	신바람 농구가 살아난 SK가 오리온에 승리를	0.0026000000854868
5	신바람 농구가 SK가 오리온에 승리를 거뒀다	0.0026000003136540
6	농구가 살아난 SK가 오리온에 승리를 거뒀다	0.0026000006712080

<

KoBERT + 언어정보점수 반영X

"신바람 농구가 살아난 SK가 승리를 거뒀다"

```
압축 문장 후보 : [MASK] 농구가 살아난 SK가 오리온에 승리를 거뒀다
KoBERT_score : 3.5312848922330886e-05, 언어정보점수 : 0
최종 점수 : 3.5312848922330886e-05

압축 문장 후보 : 신바람 [MASK] 살아난 SK가 오리온에 승리를 거뒀다
KoBERT_score : 3.331164043629542e-05, 언어정보점수 : 0
최종 점수 : 3.331164043629542e-05

압축 문장 후보 : 신바람 농구가 [MASK] SK가 오리온에 승리를 거뒀다
KoBERT_score : 3.4105210943380370e-05, 언어정보점수 : 0
최종 점수 : 3.4105210943380370e-05
```

	압축 문장 후보	최종 점수
0	신바람 농구가 살아난 SK가 승리를 거뒀다	0.000032768086385
1	신바람 농구가 살아난 SK가 오리온에 거뒀다	0.000032963762351
2	신바람 농구가 살아난 SK가 오리온에 승리를	0.000033306270780
3	신바람 살아난 SK가 오리온에 승리를 거뒀다	0.000033311640436
4	신바람 농구가 살아난 오리온에 승리를 거뒀다	0.000033402557165
5	신바람 농구가 SK가 오리온에 승리를 거뒀다	0.000034105210943
6	농구가 살아난 SK가 오리온에 승리를 거뒀다	0.000035312848922

<

KoBERT + 언어정보점수 반영O

"신바람 살아난 SK가 오리온에 승리를 거뒀다"

```
압축 문장 후보 : [MASK] 농구가 살아난 SK가 오리온에 승리를 거뒀다
KoBERT_score : 3.5312848922330886e-05, 언어정보점수 : 0.0026000000000000003
최종 점수 : 0.002635312848922331

압축 문장 후보 : 신바람 [MASK] 살아난 SK가 오리온에 승리를 거뒀다
KoBERT_score : 3.331164043629542e-05, 언어정보점수 : 0.0016
최종 점수 : 0.001633311640436295

압축 문장 후보 : 신바람 농구가 [MASK] SK가 오리온에 승리를 거뒀다
KoBERT_score : 3.4105210943380370e-05, 언어정보점수 : 0.0026000000000000003
최종 점수 : 0.0026341052109433807

압축 문장 후보 : 신바람 농구가 살아난 [MASK] 오리온에 승리를 거뒀다
```

	압축 문장 후보	최종 점수
0	신바람 살아난 SK가 오리온에 승리를 거뒀다	0.001633311640436
1	신바람 농구가 살아난 오리온에 승리를 거뒀다	0.002033402557165
2	신바람 농구가 살아난 SK가 승리를 거뒀다	0.002132768086385
3	신바람 농구가 살아난 SK가 오리온에 거뒀다	0.002132963762351
4	신바람 농구가 살아난 SK가 오리온에 승리를	0.002633306270780
5	신바람 농구가 SK가 오리온에 승리를 거뒀다	0.002634105210943
6	농구가 살아난 SK가 오리온에 승리를 거뒀다	0.002635312848922

2. 정답 압축 문장과 문장 압축 모델을 통해 생성된 최종 압축 문장의 유사도 비교

- ‘축약 전 원본 문장.txt’ : 뉴스 기사의 가장 첫 문장 / ‘정답 축약 문장.txt’ : 뉴스 기사의 제목

- Random으로 50개의 문장에 대해 유사도를 비교하고, 이를 10번 반복하는 것을 ‘한 번의 시행’이라고 정함.

- **정답 압축 문장과 유사도가 높을수록 모델의 문장 압축 정확도가 높은 것.**

유사도 평균 : 0.5048226637

```
def add_linguisitic_score(sentence):
  pos = komoran.pos(sentence)
  score = 0
  for p in pos:
    if p[1] == 'NNP' : # 고유명사
      score += 0.0001 # 임의로
    elif p[1] == 'NNG' : # 일반명사
      score += 0.0001 # 임의로
    elif p[1] == 'vv' : # 동사 -> 서술어
      score += 0.03 # 임의로
    elif p[1] == 'SL' : # 외국어
      score += 0.0003 # 임의로
    elif p[1] == 'JKS' : # 주격 조사 -> 주어
      score += 0.01
    elif p[1] == 'JKO' : # 목적격 조사 -> 목적어
      score += 0.001
  quoted_words = re.findall(r"'(.*?)'", sentence)
  for quoted_word in quoted_words:
      score += 0.000001
  return score
```

▶ 1차 시행 - 10번의 최종 평균: 0.5033522891298445

▶ 2차 시행 - 10번의 최종 평균: 0.5090985760178934

▶ 3차 시행 - 10번의 최종 평균: 0.5020171259988828

<

→

동사, 주어

가중치 올림

(더 작은 값 부여)

유사도 평균 : 0.531327939

```
def add_linguisitic_score(sentence):
  pos = komoran.pos(sentence)
  score = 0
  for p in pos:
    if p[1] == 'NNP' : # 고유명사
      score += 0.0001 # 임의로
    elif p[1] == 'NNG' : # 일반명사
      score += 0.0001 # 임의로
    elif p[1] == 'vv' : # 동사 -> 서술어
      score += 0.0003 # 임의로
    elif p[1] == 'SL' : # 외국어
      score += 0.0003 # 임의로
    elif p[1] == 'JKS' : # 주격 조사 -> 주어
      score += 0.003
    elif p[1] == 'JKO' : # 목적격 조사 -> 목적어
      score += 0.003
  quoted_words = re.findall(r"'(.*?)'", sentence)
  for quoted_word in quoted_words:
      score += 0.000001
  return score
```

1차 시행 - 10번의 최종 평균: 0.536003336291128

2차 시행 - 10번의 최종 평균: 0.5268287790376864

3차 시행 - 10번의 최종 평균: 0.5311517023484111

<

→

고유명사, 동사

가중치 올림

(더 작은 값 부여)

유사도 평균 : 0.539874001

```
def add_linguisitic_score(sentence):
  pos = komoran.pos(sentence)
  score = 0
  for p in pos:
    if p[1] == 'NNP' : # 고유명사
      score += 0.00001 # 임의로
    elif p[1] == 'NNG' : # 일반명사
      score += 0.0001 # 임의로
    elif p[1] == 'vv' : # 동사 -> 서술어
      score += 0.00001 # 임의로
    elif p[1] == 'SL' : # 외국어
      score += 0.0003 # 임의로
    elif p[1] == 'JKS' : # 주격 조사 -> 주어
      score += 0.003
    elif p[1] == 'JKO' : # 목적격 조사 -> 목적어
      score += 0.003
  quoted_words = re.findall(r"'(.*?)'", sentence)
  for quoted_word in quoted_words:
      score += 0.000001
  return score
```

1차 시행 - 10번의 최종 평균: 0.5349084841949296

2차 시행 - 10번의 최종 평균: 0.5381389225947976

3차 시행 - 10번의 최종 평균: 0.5465745968652412

04

활용 방안

■ 활용 방안

신문 기사의 제목 생성

자동 문서 요약기의 성능 향상

영상의 자막 생성

정보성 이메일의 내용 압축

…

효과적인 정보 전달을 필요로 하는 곳에 활용

05

결론 및 제언

■ 결론 및 제언

- 문제점
  - ‘우리카드’, ‘한국전력’ 등 일반 명사의 결합으로 생성된 고유명사를 인식하지 못하여, 최종 압축 문장에 꼭 포함되어야 하는 중요 단어임에도 포함되지 않은 경우 존재
    (한국어로 공개된 **개체명 인식기**가 존재하지 않아서 해결이 불가능하였음)
  - 언어정보점수 부여 시, 가중치를 주관적인 판단으로 부여함

- 앞으로 추가적으로 진행할 부분
  - 개체명 인식 관련 데이터를 통해 직접 개체명 인식기 제작
  - 언어 정보 점수 부여 시, 객관적인 기준 제시

감사합니다.