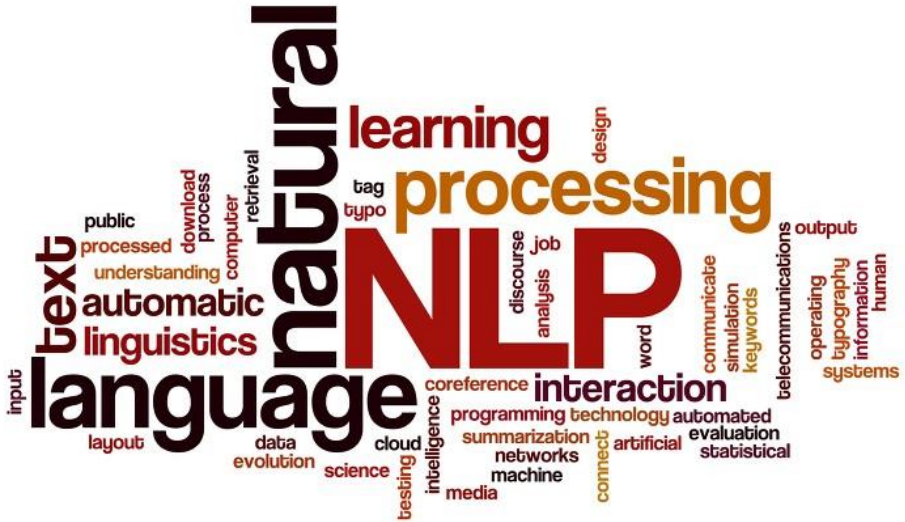
NLP

김찬영, 박지하, 송원진

목차


텍스트의 토큰화

NLP - 자연어 처리



SMARCLE

NLP - 자연어 처리

□ □

SMARCLE

사용 예시

1. Siri
2. Amazon Alexa
3. Google Assistant
4. Naver Clova

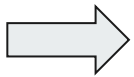
SMARCLE

텍스트의 토큰화

입력된 텍스트를 단어별, 문장별, 형태소별 등 잘게 나누는 과정 = “토큰화"

```
from tensorflow.keras.preprocessing.text import text_to_word_
sequence

text = '해보지 않으면 해낼 수 없다'
result = text_to_word_sequence(text)
print(result)
```



['해보지', '않으면', '해낼', '수', '없다']

SMARCLE

텍스트의 토큰화

먼저 텍스트의 각 단어를 나누어 토큰화합니다.

텍스트의 단어를 토큰화해야 딥러닝에서 인식됩니다.

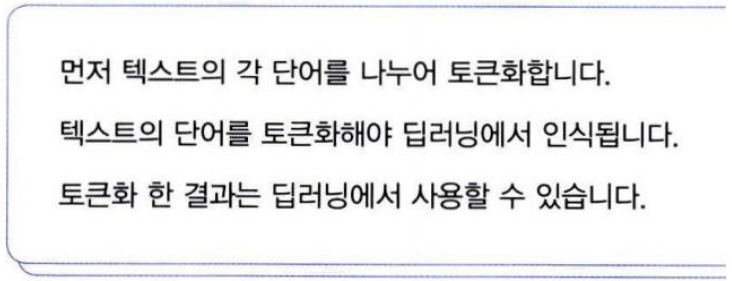
토큰화 한 결과는 딥러닝에서 사용할 수 있습니다.

텍스트의 토큰화

-텍스트 전처리 과정

Bag-of-Words

토큰화 3회
딥러닝에서 2회
텍스트의 2회



SMARCLE

텍스트의 토큰화

Tokenizer

```
from tensorflow.keras.preprocessing.text import Tokenizer
```

```
docs = ['먼저 텍스트의 각 단어를 나누어 토큰화합니다.',
        '텍스트의 단어로 토큰화해야 딥러닝에서 인식됩니다.',
        '토큰화한 결과는 딥러닝에서 사용할 수 있습니다.',
        ]
```

텍스트의 토큰화

word_counts

```
token = Tokenizer( )          # 토큰화 함수 지정
token.fit_on_texts(docs)      # 토큰화 함수에 문장 적용
print(token.word_counts)      # 단어의 빈도 수를 계산한 결과 출력
```

```
OrderedDict([('먼저', 1), ('텍스트의', 2), ('각', 1), ('단어를', 1), ('나누어', 1), ('토큰화', 3), ('합
니다', 1), ('단어로', 1), ('해야', 1), ('딥러닝에서', 2), ('인식됩니다', 1), ('한', 1), ('결과는', 1), ('사
용', 1), ('할', 1), ('수', 1), ('있습니다', 1)])
```

SMARCLE

텍스트의 토큰화

word_docs

```
print(token.word_docs)
```

```
{'한': 1, '먼저': 1, '나누어': 1, '해야': 1, '토큰화': 3, '결과는': 1, '각': 1, '단어를': 1, '인식됩니다': 1, '있습니다': 1, '할': 1, '단어로': 1, '수': 1, '합니다': 1, '딥러닝에서': 2, '사용': 1, '텍스트의': 2}
```

텍스트의 토큰화

word_index

```
print(token.word_index)
```

```
{'딥러닝에서': 3, '단어를': 6, '결과는': 13, '수': 16, '한': 12, '인식됩니다': 11, '합니다': 8, '텍스
트의': 2, '토큰화': 1,
'할': 15, '각': 5, '있습니다': 17, '먼저': 4, '나누어': 7, '해야': 10, '사용': 14, '단어로': 9}
```

원-핫 인코딩 (one-hot encoding)

SMARCLE

단어의 원-핫 인코딩(one-hot encoding)

‘오랫동안 꿈꾸는 이는 그 꿈을 닮아간다’

(0인덱스) 오랫동안 꿈꾸는 이는 그 꿈을 닮아간다

[0 0 0 0 0 0 0]

오랫동안 = [0100000]

꿈꾸는 = [0010000]

이는 = [0001000]

그 = [0000100]

꿈을 = [0000010]

닮아간다 = [0000001]

단어 임베딩
(Word Embedding)

SMARCLE

단어 임베딩(Word Embedding)

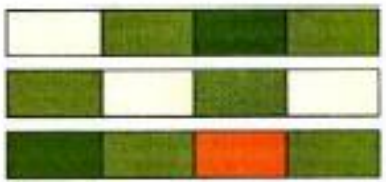
Ex) 나는 운동하고, 게임하고, 영화보고...(수천개)...하는 것이 취미이다.

원-핫 인코딩? **너무 길다! = 공간적 낭비 발생**

SMARCLE

단어 임베딩(Word Embedding)

▲ 원-핫 인코딩



▲ 단어 임베딩

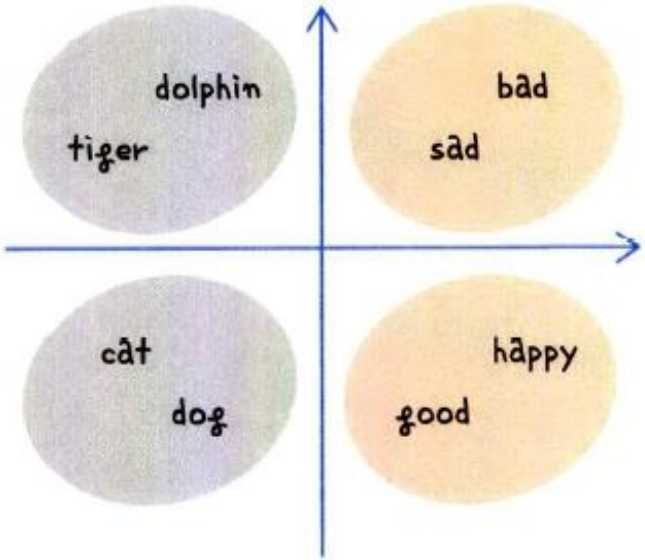
Ex) 16차원 벡터의 원-핫 인코딩과
단어 임베딩을 통해 4차원으로 바뀐 모습

SMARCLE

단어 임베딩(Word Embedding)

※ 단어 간의 유사도 파악

‘ Embedding() ’



SMARCLE

단어 임베딩(Word Embedding)

```
from keras.layers import Embedding

model = Sequential( )
model.add(Embedding(16, 4 , input_length = 2))
```

입력 출력

SMARCLE


긍정, 부정 예측하기

긍정, 부정 예측하기

-영화를 보고 남긴 리뷰를 딥러닝 모델로 학습해서,
각 리뷰가 긍정인지 부정인지 예측하기-

짧은 리뷰 10개를 불러와 각각 긍정이면 1이라는 클래스를,
부정이면 0이라는 클래스를 지정.

```
# 텍스트 리뷰 자료 지정
docs = ['너무 재밌네요','최고예요','참 잘 만든 영화예요','추천하고 싶
은 영화입니다.','한 번 더 보고싶네요','글쎄요','별로예요','생각보다 지
루하네요','연기가 어색해요','재미없어요']

# 긍정 리뷰는 1, 부정 리뷰는 0으로 클래스 지정
class = array([1,1,1,1,1,0,0,0,0,0])
```

토큰화 과정

```
# 토큰화
token = Tokenizer( )
token.fit_on_texts(docs)
print(token.word_index)    # 토큰화 된 결과를 출력해 확인
```

```
{'생각보다': 16, '만든': 6, '영화입니다': 10, '한 번': 11, '영화예요': 7, '싶은': 9, '보고싶네요':
13, '어색해요': 19, '재미없어요': 20, '더': 12, '추천하고': 8, '지루하네요': 17, '최고예요': 3, '
잘': 5, '참': 4, '재밌네요': 2, '별로예요': 15, '글쎄요': 14, '연기가': 18, '너무': 1}.
```

토큰에 지정된 인덱스로 새로운 배열 생성

```
x = token.texts_to_sequences(docs)
print(x)
```

```
[[1, 2], [3], [4, 5, 6, 7], [8, 9, 10], [11, 12, 13], [14], [15], [16, 17], [18, 19], [20]]
```

SMARCLE

패딩

```
padded_x = pad_sequences(x, 4)  # 서로 다른 길이의 데이터를 4로 맞추기
print(padded_x)
```

```
[[ 0  0  1  2]
 [ 0  0  0  3]
 [ 4  5  6  7]
 [ 0  8  9 10]
 [ 0 11 12 13]
 [ 0  0  0 14]
 [ 0  0  0 15]
 [ 0  0 16 17]
 [ 0  0 18 19]
 [ 0  0  0 20]]
```

SMARCLE

단어 임베딩

```
word_size = len(token.word_index) +1
```

```
Embedding(word_size, 8, input_length=4)
```

세 가지 파라미터 필요
(입력, 출력, 단어 수)

SMARCLE

최근 NLP 기술 동향

최근 NLP 개발 동향

kakaobrain

PORORO: Platform Of neuRal mOdels for natuRal language prOcessing

Pororo : https://github.com/kakaobrain/pororo

SMARCLE

감사합니다.