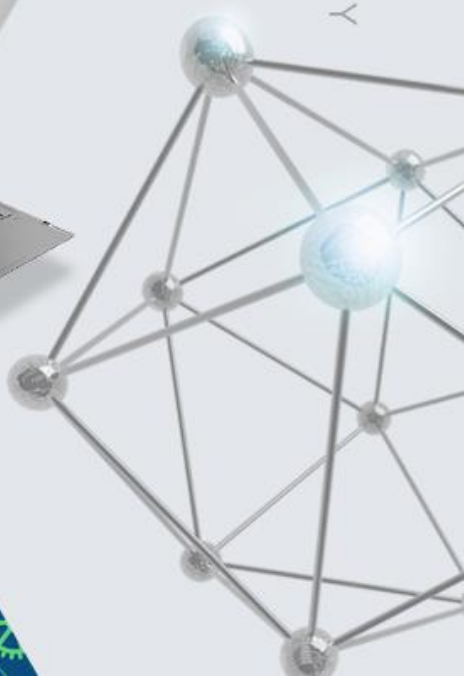
TECHNOLOGY

인공지능 자연어 처리

인공지능 자연어 처리를 위한 형태소 분석해 보기



The 4th Industrial Revolution is characterized by super connectivity and super intelligence, where various products and services are connected to the network, and artificial intelligence and information communication technologies are used in 3D printing, unmanned transportation, robotics, Of the world's most advanced technologies.

인공지능 자연어 처리를 위한 형태소 분석해 보기

학/습/목/표

1. 형태소 분석에 대해 설명할 수 있다.
2. 영문 형태소 분석에 대해 수행할 수 있다.
3. 한글 형태소 분석에 대해 수행할 수 있다.

학/습/내/용

1. 형태소 분석 개요
2. 영문 형태소 분석
3. 한글 형태소 분석

1. 형태소 분석 개요

1) 형태소 분석 개념

(1) 형태소(Morphome)란?

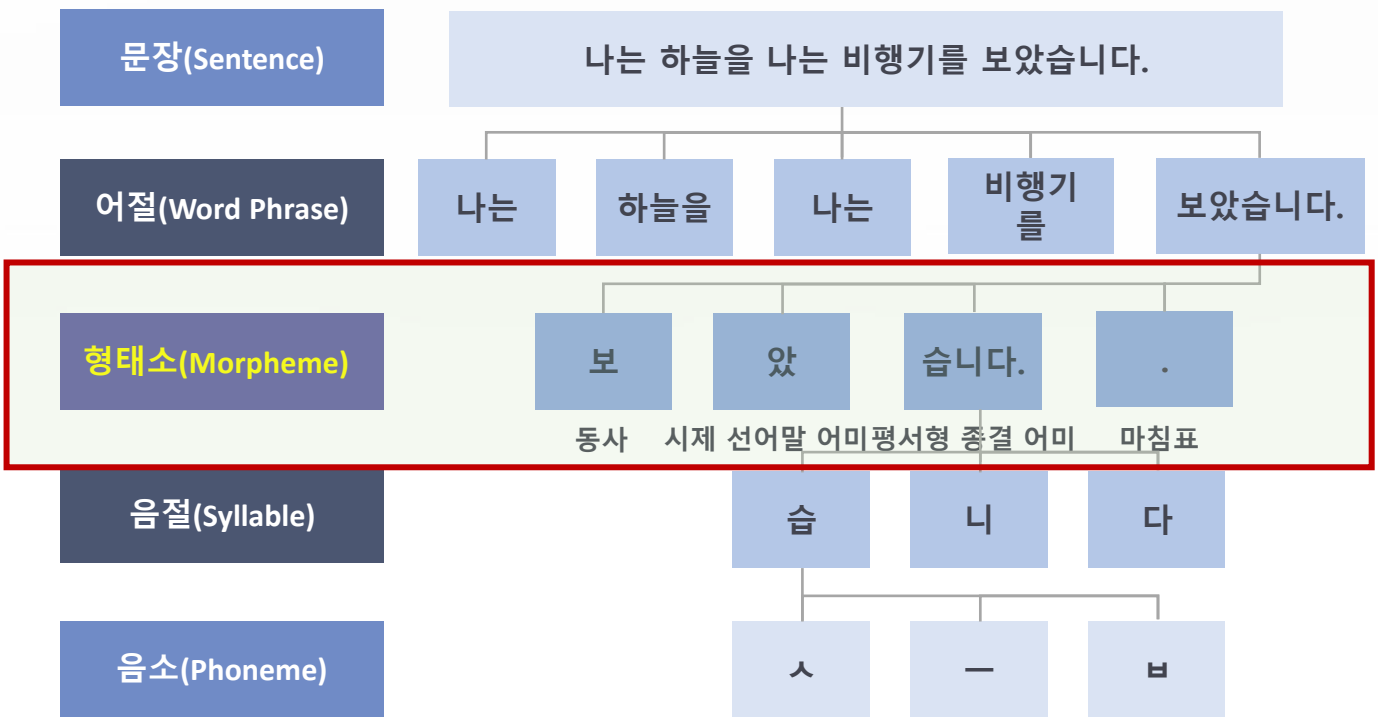
- 자연어에서 최소의 의미 단위

(2) 형태소 분석(Morphological Analysis)이란?

- 자연어 문장에서 의미를 가진 최소 단위인 형태소 분석

☞ 형태소 : 명사, 동사, 형용사, 부사, 조사, 어미 등

자연어 처리를 위해 수행해야 하는 가장 첫 단계의 분석

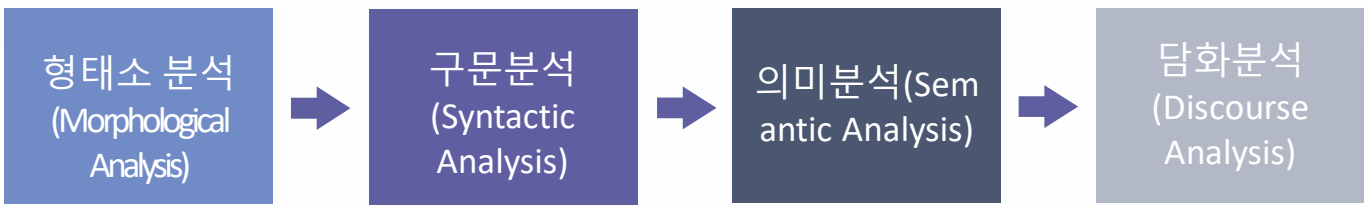


1. 형태소 분석 개요

2) 자연어 처리 분석 과정

(1) 형태소 분석(Morphological Analysis)

- 자연어 처리를 위한 분석 과정 중 가장 기본적인 시작 과정
- 자연어 처리는 일반적으로 형태소 분석, 구문 분석, 의미분석, 담화 분석 등의 과정으로 수행



(2) 형태소 분석 과정

- 형태소 분석기를 이용하여 형태소 분석 수행
- 파이썬을 이용한 형태소 분석

  → 영문 형태소 분석기 : NLTK(http://www.nltk.org)

  → 한글 형태소 분석기 : KoNLPy:(http://konlpy.org/ko/latest)

형태소 분석기

```
from eunjeon import Mecab
tagger = Mecab()
```

형태소 분석 문장

```
pos = tagger.pos('나는 하늘을 나는 비행기를 보았다')
```

형태소 분석 결과

```
pos
```

```
[('나', 'NP'),
 ('는', 'JX'),
 ('하늘', 'NNG'),
 ('을', 'JKO'),
 ('나', 'NP'),
 ('는', 'JX'),
 ('비행기', 'NNG'),
 ('를', 'JKO'),
 ('보', 'VV'),
 ('았', 'EP'),
 ('다', 'EC')]
```

1. 형태소 분석 개요

2) 자연어 처리 분석 과정

(3) 품사 태깅표 예시[Mecab 형태소 분석기]

NNG	일반 명사	MAG	일반 부사	JX	보조사
NNP	고유 명사	MAJ	접속 부사	EP	선어말어미
NNB	의존 명사	IC	감탄사	EF	종결 어미
NNBC	단위를 나타내는 명사	JKS	주격 조사	EC	연결 어미
NR	수사	JKC	보격 조사	ETN	명사형 전성 어미
NP	대명사	JKG	관형격 조사	ETM	관형형 전성 어미
VV	동사	JKO	목적격 조사	XPN	체언 접두사
VA	형용사	JKB	부사격 조사	XSN	명사 파생 접미사
VX	보조 용언	JKV	호격 조사	XSV	동사 파생 접미사
VCP	긍정 지정사	JKQ	인용격 조사	XSA	형용사 파생 접미사
VCN	부정 지정사	JC	접속 조사	XR	어근
MM	관형사				

2) 자연어 처리 분석 과정

(4) 형태소 분석기의 한계

일반적인 형태소 분석기

주어 '나는' 술어 '나는'

구분 불가

정확도 높은 형태소 분석

딥러닝 '학습' 방법 사용을 통한 개선가능

1. 형태소 분석 개요

2) 자연어 처리 분석 과정

(5) 구문분석(Syntactic Analysis) 과정

문장의 구조적 성질을 규칙화한 문법 » 문장의 구조를 분석하는 과정

나	는	하늘	을	나	는	비행기	를	보	았	다
NP	JX	NNG	JKO	W	ETM	NNG	JKO	W	EP	EC
주어		목적어		술어		목적어		술어		

1. 형태소 분석 개요

2) 자연어 처리 분석 과정

(6) 의미분석(Semantic Analysis) 과정

구문 분석결과에 해석을 가미 » 문장의 의미를 분석하는 과정

나	는	하늘	을	나	는	비행기	를	보	았	다
NP	JX	NNG	JKO	W	ETM	NNG	JKO	W	EP	EC
주어		목적어		술어		목적어		술어		
행동주 (Agent)		피동주 (Agent)		술어 (날다, Fly)		피동주 (Agent)		술어 (보다, See)		

1. 형태소 분석 개요

2) 자연어 처리 분석 과정

(7) 담화분석(Discourse Analysis) 분석

대화의 맥락(Context)과 의도(Intent) 파악 » 문맥과 의도에 맞게 문장의 의미를 분석하는 과정

나는 하늘을 나는 비행기를 보았다

나는 하늘에서 날고 있는 비행기를 보았다

나는 아메리카노 그녀는 라떼

나는 아메리카노를 주문했고 그녀는 라떼를 주문했다

나는 무작정 걸었다

나는 무작정 (길을) 걸었다

나는 무작정 (전화를) 걸었다

나는 무작정 (옷을) 걸었다

2. 영문 형태소 분석

1) 영문 형태소 분석기 소개 및 설치(Python, NLTK)

(1) NLTK 패키지란?

- 교육용으로 개발된 자연어 처리 및 문서 분석용 파이썬 패키지

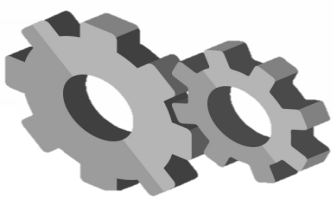
(2) NLTK 패키지 주요 기능

- 텍스트 분석에 유용한 다양한 메서드를 포함하는 Text 클래스 제공

코퍼스 (Corpus)

토큰화(Tokenizing)

형태소 분석



NLTK 패키지 설치

pip Install NLTK

- Windows에서 설치시 : pip install nltk
- Mac에서 설치시 : sudo pip install -U nltk

2. 영문 형태소 분석

2) 영문 형태소 분석기 사용방법

(1) NLTK 연구용 코퍼스 다운로드

- 저작권이 말소된 문학작품을 포함하는 gutenberg 코퍼스 다운로드

```
import nltk
nltk.download('gutenberg')
```

- gutenberg 코퍼스 목록

```
from nltk.corpus import gutenberg
print(gutenberg.fileids())
```

```
['austen-emma.txt',
 'austen-persuasion.txt',
 'austen-sense.txt',
 'bible-kjv.txt',
 'Bryant-stories.txt',
 'burgess-busterbrown.txt',
 'carroll-alice.txt',
 'chesterton-ball.txt',
 'chesterton-brown.txt',
 'chesterton-thursday.txt',
 'edgeworth-parents.txt',
 'melville-moby_dick.txt',
 'milton-paradise.txt',
 'shakespeare-caesar.txt',
 'shakespeare-hamlet.txt',
 'shakespeare-macbeth.txt',
 'whitman-leaves.txt']
```

2) 영문 형태소 분석기 사용방법

(2) 토큰화(Tokenizing)란?

- 코퍼스를 문장(Sentence), 단어(Word) 단위 등의 작은 단위로 분리하는 과정
- 문장 토큰화 : 코퍼스를 문장 단위로 토큰화

→
```
from nltk.tokenize import sent_tokenize
print(sent_tokenize(hamlet[:100]))
```

- 단어 토큰화 : 코퍼스를 단어 단위로 토큰화

→
```
from nltk.tokenize import word_tokenize
print(word_tokenize(hamlet[:100]))
```

2. 영문 형태소 분석

2) 영문 형태소 분석기 사용방법

(3) 형태소 분석

- Tagger 다운로드

Tagger 다운로드

```
import nltk
nltk.download('averaged_perceptron_tagger')
```

- 품사 태깅

품사 태깅

```
from nltk.tag import pos_tag
sentence = "You come most carefully vpon your houre"
tagged_list = pos_tag(word_tokenize(sentence))
print(tagged_list)
```

```
[('You', 'PRP'),
 ('come', 'VBP'),
 ('most', 'RBS'),
 ('carefully', 'RB'),
 ('vpon', 'VB'),
 ('your', 'PRP$'),
 ('houre', 'NN')]
```

- 품사 태깅 제거

품사 태깅 제거

```
from nltk.tag import untag
untag(tagged_list)
```

```
['You', 'come', 'most', 'carefully', 'vpon', 'your', 'houre']
```

3. 한글 형태소 분석

1) 한글 형태소 분석기 설치(Python, KoNLPy)

(1) 한글 형태소 분석기 설치

Java JDK 설치	»	JDK 1.8 설치
		- 이미 설치되어 있는 경우 생략 가능
JPype1 설치	»	pip Install JPype1
KoNLPy 설치	»	pip Install KoNLPy

(2) 한글 품사 태깅 클래스 종류

- Kkma
- Komoran
- Hannanum
- Okt(previous Twitter)
- Mecab

Windows 환경에서는 Mecab 미지원 → 별도 설치 필요

3. 한글 형태소 분석

2) 한글 품사 태깅 클래스 종류 및 성능비교

(1) 품사 태깅 클래스 품사 태깅 성능 비교

Hannanum	Kkma	Komoran	Mecab	Twitter
아버지가방에 들어가 / N	아버지 / NNG	아버지가방에 들어가신다 / NNP	아버지 / NNG	아버지 / Noun
이 / J	가방 / NNG		가 / JKS	가방 / Noun
시ㄴ다 / E	에 / JKM		방 / NNG	에 / Josa
	들어가 / W		에 / JKB	들어가신 / Verb
	시 / EPH		들어가 / W	다 / Eomi
	s다 / EFN		신다 / EP+EC	

(2) 수행시간 성능 비교

품사 태깅 클래스	클래스 로딩 시간	10만 문자 품사 태깅 실행시간
Kkma	5.6988 secs	35.7163 secs
Komoran	5.4866 secs	25.6008 secs
Hannanum	0.6591 secs	8.8251 secs
Okt(previous Twitter)	1.4870 secs	2.4714 secs
Mecab	0.0007 secs	0.2838 secs

3) KoNLPy를 이용한 형태소 분석 방법

(1) 파이썬에서 KoNLPy 사용

- 형태소 분석 수행: tagger .morphs(txt)
- 명사만 추출: tagger . Nouns(txt)
- 품사 태깅: tagger .pos(txt)
- KoNLPy 클래스 Import

  → `from konlpy.tag import Okt, Kkma, Komoran, Hannanum`

- 사용하고자 하는 클래스 객체 생성

  →
```
# tagger = Kkma( )
# tagger = Komoran( )
# tagger = Hannanum( )
tagger = Okt( )
```

3) KoNLPy를 이용한 형태소 분석 방법

(2) 파이썬에서 KoNLPy 사용 예시

```
tagger.morphs(txt)
```

['나', '는', '하늘', '을', '나', '는', '비행기', '를', '보았습니다', '.']

```
tagger.nouns(txt)
```

['나', '하늘', '나', '비행기']

```
tagger.pos(txt)
```

[('나', 'Noun'),
('는', 'Josa'),
('하늘', 'Noun'),
('을', 'Josa'),
('나', 'Noun'),
('는', 'Josa'),
('비행기', 'Noun'),
('를', 'Josa'),
('보았습니다', 'Verb'),
('.', 'Punctuation')]

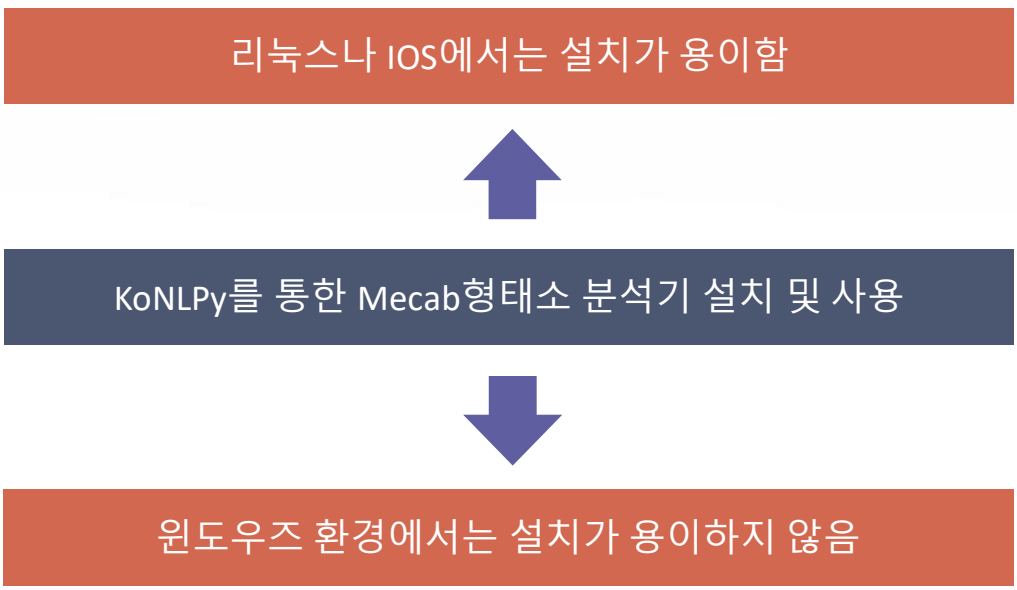
3. 한글 형태소 분석

3) KoNLPy를 이용한 형태소 분석 방법

(3) 윈도우즈 환경에서 설치 방법

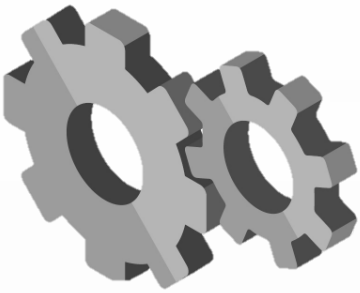
- 윈도우즈 환경에서는 별도의 eunjeon 패키지를 통해 Mecab 설치 및 사용 용이

- Pyeunjeon이란 은전한닢 프로젝트와 Mecab 기반의 한국어 형태소 분석기의 독립형 Python 인터페이스를 의미



3) KoNLPy를 이용한 형태소 분석 방법

(3) 윈도우즈 환경에서 설치 방법



윈도우즈의 환경에서 eunjeon
패키지 설치

pip Install eunjeon

1. 형태소 분석 개요

1) 형태소 개요

- 형태소 분석은 자연어 문장에서 의미를 가진 최소 단위인 형태소를 분석
- 자연어 처리를 위해 수행해야 하는 가장 첫 단계의 분석

2. 영문 형태소 분석

1) 영문형태소 분석

- 영문형태소 분석을 위한 NLTK 패키지는 교육용으로 개발된 자연어 처리 및 문서 분석용 파이썬 패키지

2) NLTK 패키지

- 코퍼스, 토큰화, 형태소 분석 등의 기능을 제공하고 텍스트 분석에 유용한 다양한 메서드를 포함하는 Text 클래스 제공

3) 토큰화(Tokenizing)

- 토큰화(Tokenizing)는 코퍼스를 문장(Sentence), 단어(Word) 단위 등의 작은 단위로 분리하는 과정으로 문장 토큰화나 단어 토큰화로 수행

3. 한글 형태소 분석

1) KoNLPy의 한글 품사 태깅 클래스 종류

- Kkma
- Komoran
- Hannanum
- Okt (previous Twitter)
- Mecab
- Windows 환경에서는 Mecab 미지원, 별도 설치 필요