

NLP WITH PYTORCH

PART I

KOREAN MORPHEME ANALYER

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지

- 수많은 형태소 라이브러리가 [Open소스로 배포](#)
- 직접 사전 준비, 형태소 분석 알고리즘 구현없이 분석 가능
- 대표적인 형태소 분석 패키지
 - **NLTK(Natural Language Toolkit)** : 교육용 개발된 자연어 처리 및 문서 분석용
 - **genism** : 토픽 모델링 분야에서 두각 나타내는 패키지, Word2Vec 구현 기능 제공
 - **spacy** : 뛰어난 수행 능력으로 최근 가장 주목 받는 NLP
 - **KoNLPy(Korean NLP in Python)** : 한국어 정보처리를 위한 파이썬 패키지

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 - 한국어

▪ Genism

- Python 기반의 Text mining library이며, 토픽 모델링, word2vec 지원
- 주제 모델링, 문서 인덱싱, 유사성 검색 및 기타 자연어 처리 위한 오픈 소스 라이브러리

• KoNLPy(Korean NLP in Python)

- 한국어 정보처리를 위한 파이썬 패키지로 **한국어 형태소 분석기 대표**
- **다양한 한국어 형태소 분석기를 내장**하고 있음
- 미등록 단어 인식 잘 되지 않음 ➔ 사용자 직접 사전에 미등록 단어 등록 필요

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 - 한국어

- **Pororo (Platform Of neuRal mOdelS for natuRal language prOcessing : 뽀로로)**

- 카카오 브레인에서 개발한 자연어 및 음성 처리 패키지
- 한국어, 영어, 중국어, 일본어 등 여러가지 언어로 30가지 이상의 자연어 처리모델 구현
- 설치가 까다로움 → konlpy 설치 → pororo 설치 → python-mecab-ko 설치

- **Soynlp KoNLPy(Korean NLP in Python)**

- Konlpy의 단점인 미등록 단어 인식 부분 처리 기능 제공
- 비지도 학습으로 단어 토큰화 진행하며 자주 등장하는 단어들을 분석함
- 내부적으로 단어 점수 표를 생성하여 cohesion 기반으로 토큰화를 할 수 있는 기능 제공
- 패키지 내 말뭉치 미제공으로 추가 설치 필요

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 - 한국어

▪ Kiwi (Korean Intelligent Word Identifier)

- 세종 품사 태그 체계 기반으로 C++ 지능형 한국어 형태소 분석기 → Kiwipiepy 파이썬
- 세종 계획 말뭉치와 모두의 말뭉치를 학습 모델로 사용
- 한국어 신조어 및 고유명사 등의 분류도 원할
- <https://bab2min.github.io/kiwipiepy/v0.8.0/kr/>

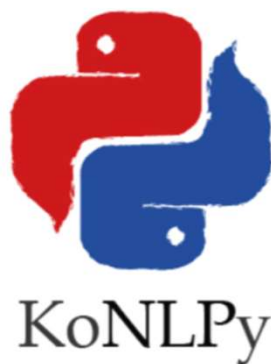
• Bareun 바른

- 웹사이트에서 간단하게 형태소 분석을 해볼 수 있는 서비스를 지원
- 형태소 분석이 어떤 것인지 시각적으로 바로 이해할 수 있는 유용
- <https://bareun.ai/>

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 준비 - KoNLPy

- Java기반의 형태소 분석기
- <https://konlpy.org/ko/latest/index.html>



KoNLPy: 파이썬 한국어 NLP

Build status docs passing

KoNLPy("코엔엘파이"라고 읽습니다)는 한국어 정보처리를 위한 파이썬 패키지입니다. 설치 방법은 [이 곳을](#) 참고해주세요.

NLP를 처음 시작하시는 분들은 [시작하기](#) 에서 가볍게 기본 지식을 습득할 수 있으며, KoNLPy의 사용법 가이드는 [사용하기](#), 각 모듈의 상세사항은 [API](#) 문서에서 보실 수 있습니다.

```
>>> from konlpy.tag import Kkma
>>> from konlpy.utils import pprint
```

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 준비 - KoNLPy

① JDK 설치

- OS 비트 수와 일치 하는 JDK 다운로드
- <https://www.oracle.com/java/technologies/downloads/>

Linux	macOS	Windows
Product/file description	File size	Download
x64 Compressed Archive	172.47 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip (sha256)
x64 Installer	153.55 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe (sha256)
x64 MSI Installer	152.34 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi (sha256)

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 준비 - KoNLPy

① JDK 설치

- 시스템 환경변수 JAVA_HOME 설정

환경 변수

The screenshot shows the 'Environment Variables' window for a user named 'anece'. It is divided into two sections: 'User variables for anece' and 'System variables'. In the 'System variables' section, the 'JAVA_HOME' variable is highlighted. Its value is 'C:\Program Files\Java\jdk-17.0.2\'. The 'Path' variable is also visible, showing the path to the Java bin directory. At the bottom, there are buttons for '확인' (OK) and '취소' (Cancel).

변수	값
ChocolateyLastPath...	133432261476427650
OneDrive	C:\Users\Wanece\OneDrive
Path	C:\Program Files\WRWR-4.0.2\bin\wx64;C:\Pr...
PyCharm Communi...	C:\Program Files\JetBrains\PyCharm Comm...
SCOOP	C:\scoop

변수	값
JAVA_HOME	C:\Program Files\Java\jdk-17.0.2\
NUMBER_OF_PRO...	4
OS	Windows_NT
Path	C:\Program Files (x86)\Intel\iCLS Client\;C...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;...

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 준비 - KoNLPy

② Jpype1 설치

- Python에서 Java 라이브러리 사용할 수 있도록 설치 필요
- <https://www.lfd.uci.edu/~gohlke/pythonlibs/#jpype>

- anaconda powershell prompt 관리자 권한으로 실행
- 자신의 가상환경 실행

[설치] (My_38) > **conda install -c conda-forge jpype1**

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 준비 - KoNLPy

③ KoNLPy 설치

[설치] (My_38) > pip install **konlpy**

```
(My_38) PS C:\Users\Wanece> pip install konlpy
Collecting konlpy
  Downloading konlpy-0.6.0-py2.py3-none-any.whl (19.4 MB)
    ----- 19.4/19.4 MB 22.6 MB/s eta 0:00:00
Requirement already satisfied: JPype1>=0.7.0 in c:\wprogramdata\Wanaconda3\Wenvs\Wmy_38\lib\site-packages (from konlpy) (1.4.1)
Requirement already satisfied: lxml>=4.1.0 in c:\wprogramdata\Wanaconda3\Wenvs\Wmy_38\lib\site-packages (from konlpy) (4.9.3)
Requirement already satisfied: numpy>=1.6 in c:\wprogramdata\Wanaconda3\Wenvs\Wmy_38\lib\site-packages (from konlpy) (1.24.3)
Requirement already satisfied: packaging in c:\wprogramdata\Wanaconda3\Wenvs\Wmy_38\lib\site-packages (from JPype1>=0.7.0->konlpy) (23.1)
WARNING: Error parsing requirements for jupyter-highlight-selected-word: [Errno 2]
tory: 'c:\wprogramdata\Wanaconda3\Wenvs\Wmy_38\lib\site-packages\jupyter_highlight
0.dist-info\METADATA'
Installing collected packages: konlpy
Successfully installed konlpy-0.6.0
(My_38) PS C:\Users\Wanece>
```

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 준비 - KoNLPy

④ KoNLPy 설치 확인

[설치 확인] (My_38) > Python

```
(My_38) PS C:\Users\Wanece> python
Python 3.8.18 (default, Sep 11 2023, 13:39:12) [MSC v.1916 64 bit (AMD64)]
Type "help", "copyright", "credits" or "license" for more information.
>>> import konlpy
>>> konlpy.__version__
'0.6.0'
>>>
```

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 - KoNLPy

- KoNLPy 제공 말뭉치(corpus)

대한민국 헌법 말뭉치	kolaw
국회법안 말뭉치	kobill

KOREAN MORPHEME ANALYZER

◆ 형태소 분석 패키지 - KoNLPy

- 사전/분석기 : 말뭉치를 이용해서 구축된 형태소 분석 및 품사 태깅에 활용

Hannanum 한나눔	KAIST 개발	KAIST 말뭉치 이용해 생성된 사전
Kkma 꼬꼬마	SUN 개발	세종 말뭉치 이용해 생성된 사전 http://kkma.snu.ac.kr/
Komoran 코모란	Shineware 개발	Java로 구현된 사전 https://github.com/shineware/KOMORAN
Mecab 메캅	Kyoto 대학 개발	일본어용 분석기를 한국어에 적용한 사전 https://bitbucket.org/eunjeon/mecab-ko/src/master/
Okt(구 Twitter)	OpenSource	Opoen Korea Text 약자 https://github.com/open-korean-text/open-korean-text

PART I

TEXT PREPROCESSING PROGRAMMING

PREPROCESSING PROGRAMMING

◆ Konlpy 다양한 형태소 분석기

❖ 사용법 및 공통점

Hannanum 한나눔	<ul style="list-style-type: none">- konlpy.tag 패키지 내 클래스로 존재- 공통 메서드<ul style="list-style-type: none">* 명사 추출 메서드 → nouns()* 형태소 추출 메서드 → morphs()* 품사 부착 메서드 → pos()
Kkma 꼬꼬마	
Komoran 코모란	
Mecab 메캅	
Okt(구 Twitter)	

PREPROCESSING PROGRAMMING

◆ Konlpy 다양한 형태소 분석기

❖ 품사 - Okt 품사 태그

태그	품사	태그	품사	태그	품사
Nonu	명사	Josa	조사	Number	숫자
Verb	동사	PreEomi	선어말어미	KoreanParticle	자음/모음
Adjective	형용사	Eomi	어미	URL	웹 주소
Determiner	관형사	Suffix	접미사	Hashtag	해시태그(#)
Adverb	부사	Punctuation	구두점	Email	이메일
Conjunction	접속사	Foreign	외국어	ScreenName	아이디 표기(@)
Exclamation	감탄사	Alpha	알파벳	Unkonwn	미등록

PREPROCESSING PROGRAMMING

◆ Konlpy 다양한 형태소 분석기

❖ 품사 - Kkma 품사 태그

태그	품사	태그	품사	태그	품사
NNG	보통명사	VXA	보조 형용사	EFQ	의문형 종결어미
NNP	고유명사	VCP	긍정 지정사	EFO	명령형 종결어미
NNB	일반의존명사	VCN	부정 지정사	EFA	청유형 종결어미
NNM	단위의존명사	MDN	수 관형사	EFI	감탄형 종결어미
NR	수사	MDT	일반 관형사	EFR	존칭형 종결어미
NP	대명사	EPH	존칭 선어말어미	ECE	대 등 연결어미
VV	동사	EPT	시제 선어말어미	ECS	보조적 연결어미
VA	형용사	EPP	공손 선어말어미	ECD	의존적 전성어미
VXV	보조동사	EFN	평서형 종결어미	ETN	명사형 전성어미

PREPROCESSING PROGRAMMING

◆ Konlpy 다양한 형태소 분석기

❖ 코퍼스 준비

```
# 모듈 로딩
from konlpy.corpus import kolaw, kobill

# 데이터 파일명
filename=kolaw.fileids()[0]
print(f'kolaw filename : {filename}')

# 코퍼스 저장
kolaw_corpus = kolaw.open(filename).read()

print(f'길이: {len(kolaw_corpus)}')
print(kolaw_corpus[:100])
```

PREPROCESSING PROGRAMMING

◆ Konlpy 다양한 형태소 분석기

❖ 형태소 분석

```
# 모듈로딩
from konlpy.tag import *
from konlpy.corpus import kolaw

## 데이터 준비
law_corpus = kolaw.open(kolaw.fileids()[0]).read()
data = law_corpus[:100]

## Hannanum 형태소 분석기
hannaum = Hannanum()
```

PREPROCESSING PROGRAMMING

◆ Konlpy 다양한 형태소 분석기

❖ 형태소 분석

```
# - 명사추출
hannaum.nouns(data)

# - 형태소 추출
hannaum.morphs(data)

# - 품사 태깅
hannaum.pos(data)

# -- 품사태그 기호와 의미
hannaum.tagset
```

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ Mecab 설치

(1) 개발환경 체크

→ 운영체제 비트 수 : 64bit (AMD64)

→ 파이썬 버전 : 3.8.19

```
(TEXT_018_38) C:\Users\anece\TORCH_WORK\PROJECT_TEXT>python
Python 3.8.19 (default, Mar 20 2024, 19:55:45) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ Mecab 설치

(2) C:/mecab 폴더 생성

(3) mecab-ko-msvc 설치

→ 사이트 접속 : <https://github.com/Pusnow/mecab-ko-msvc/releases/tag/release-0.9.2-msvc-3>

→ 압축파일 다운로드 : mecab-ko-msvc-x64.zip

release-0.9.2-msvc-3

Pusnow released this Jul 1, 2017 · release-0.9.2-... · aef1d89

▼ Assets 4

 mecab-ko-msvc-x64.zip


 mecab-ko-msvc-x86.zip


 Source code (zip)


압축해제


mecab


 libmecab.dll


 mecab.exe


 mecab.lib

 mecab-cost-train.lib


 mecab-dict-gen.lib


 mecab-dict-index.lib


 mecab-system-eval.lib


 mecab-test-gen.lib


 libmecab.lib


 mecab.h

 mecab-cost-train.exe

 mecab-dict-gen.exe

 mecab-dict-index.exe

 mecab-system-eval.exe

 mecab-test-gen.exe

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ Mecab 설치

(4) 한국어 단어 사전 mecab-ko-dic-msvc 설치

→ 사이트 접속 : <https://github.com/Pusnow/mecab-ko-dic-msvc/releases>

→ 압축파일 다운로드 : **mecab-ko-dic-msvc.zip**

mecab-ko-dic-2.1.1-20180720-msvc-3 Latest

Update CI Pipeline

▼ Assets 3

 **mecab-ko-dic-msvc.zip**

71.2 Mb · Nov 23, 2022

 Source code (zip)




















Nov 23, 2022

 Source code (tar.gz)

Nov 23, 2022

압축해제

mecab

 mecab-ko-dic	 tools
 user-dic	 libmecab.dll
 libmecab.lib	 mecab.exe
 mecab.h	 mecab.lib
 mecab-cost-train.exe	 mecab-cost-train.lib
 mecab-dict-gen.exe	 mecab-dict-gen.lib
 mecab-dict-index.exe	 mecab-dict-index.lib
 mecabrc	 mecab-system-eval.exe
 mecab-system-eval.lib	 mecab-test-gen.exe
 mecab-test-gen.lib	

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ Mecab 설치

(5) Python에서 mecab 패키지 빌드 할 수 있도록 wheel 설치: python 버전, OS 선택

→ 사이트 접속 : <https://github.com/Pusnow/mecab-python-msvc/releases>

→ 압축파일 다운로드 : **mecab_python-0.996_ko_0.9.2_msvc-cp38-cp38-win_amd64.whl**

mecab_python-0.996_ko_0.9.2_msvc-3

• Add Python 3.8 support

▼ Assets 16

mecab_python-0.996_ko_0.9.2_msvc-cp27-cp27m-win32.whl	429 KB	Nov 15, 2019
mecab_python-0.996_ko_0.9.2_msvc-cp37-cp37m-win32.whl	432 KB	Nov 15, 2019
mecab_python-0.996_ko_0.9.2_msvc-cp37-cp37m-win_amd64.whl	485 KB	Nov 15, 2019
mecab_python-0.996_ko_0.9.2_msvc-cp38-cp38-win32.whl	432 KB	Nov 15, 2019
mecab_python-0.996_ko_0.9.2_msvc-cp38-cp38-win_amd64.whl	485 KB	Nov 15, 2019
Source code (zip)		Nov 2, 2019
Source code (tar.gz)		Nov 2, 2019

64비트 Window + Python3.8

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ Mecab 설치

(5) Python에서 mecab 패키지 빌드 할 수 있도록 wheel 설치: python 버전, OS 선택

→ 파일 이동 : mecab_python-0.996_ko_0.9.2_msvc-cp38-cp38-win_amd64.whl

→ 이동 위치 : 가상환경 아래

→ C:\Users\Wanece\anaconda3\envs\TEXT_018_38\Lib\site-packages

anaconda3 > envs > TEXT_018_38 > Lib > site-packages		
이름	수정한 날짜	유형
libmecab.dll	2024-08-20 오전 10:35	응용 프로그램 확장
MeCab.py	2024-08-20 오전 10:35	Python 원본 파일
mecab_python-0.996_ko_0.9.2_msvc-cp38-cp38-wi...	2024-08-20 오전 10:15	WHL 파일
nest_asyncio.py	2024-01-22 오전 12:26	Python 원본 파일

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ Mecab 설치

(6) Mecab 설치

- 가상환경 터미널 실행
- 설치 : **pip install mecab**

```
(TEXT_018_38) C:\Users\anece\TORCH_WORK\PROJECT_TEXT>pip install mecab
Collecting mecab
  Downloading mecab-0.996.3-cp38-cp38-win_amd64.whl.metadata (3.9 kB)
  Downloading mecab-0.996.3-cp38-cp38-win_amd64.whl (500 kB)
Installing collected packages: mecab
Successfully installed mecab-0.996.3
```

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ mecab-ko 설치

→ 윈도우 기반 한글 mecab 패키지 설치

mecab-ko 1.0.1

```
pip install mecab-ko
```



Python wrapper for the MeCab-ko morphological analyzer for Korean

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ mecab-ko 설치

→ 윈도우 기반 한글 mecab 패키지 설치

```
~  
# mecab = Mecab()  
%pip install mecab-ko  
i] ✓ 6.0s  
  
Collecting mecab-ko  
  Using cached mecab_ko-1.0.1-cp38-cp38-win_amd64.whl.metadata (6.0 kB)  
Collecting mecab-ko-dic<2.0,>=1.0 (from mecab-ko)  
  Using cached mecab_ko_dic-1.0.0-py3-none-any.whl  
Using cached mecab_ko-1.0.1-cp38-cp38-win_amd64.whl (500 kB)  
Installing collected packages: mecab-ko-dic, mecab-ko  
Successfully installed mecab-ko-1.0.1 mecab-ko-dic-1.0.0  
Note: you may need to restart the kernel to use updated packages.
```

PREPROCESSING PROGRAMMING

◆ Mecab 형태소 분석기

❖ mecab-ko 설치

→ <https://taku910.github.io/mecab/#format>

MeCab: Yet Another Part-of-Speech and Morphological Analyzer

MeCab(와후 쇼우)란?

MeCab은 교토대학 정보학연구과-일본전신전화주식회사 커뮤니케이션과학기초연구소 공동연구단위 프로젝트를 통해 개발된 오픈소스 형태소 분석기이고 ChaSen 이 채택한 숨겨진 마르코프 모델에 비해 성능이 향상되었습니다. 또한 평균적으로 ChaSen , Juman , KAKASI 보다 빠르게 작동합니다.

목차

- 특징
- 비교
- 신작 정보
- 개발까지의 경위
- 다운로드
- 설치
 - 유닉스
 - Windows

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

- ❖ 자연어 처리를 위한 Python 기반의 오픈 소스 라이브러리
- ❖ 처리 속도가 빠르고 메모리 사용이 효율적인 특징
- ❖ 다양한 언어에 대한 미리 훈련된 언어 모델을 제공
- ❖ 모델이 포함되어 있지 않으므로, 원하는 모델을 직접 다운로드 받아 사용
 - sm, md : 비교적 속도 느림, 성능 비슷
 - lg : 모델 크고 성능 좋음

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 설치

spaCy

🔴 New: Case study with S&P Global

USAGE

MODELS

API

GET STARTED

Installation

- Quickstart
- Instructions
- Troubleshooting
- Changelog

Models & Languages

Facts & Figures

spaCy 101

New in v3.7

Install spaCy

Operating
system

macOS / OSX

Windows

Linux

Platform

x86

ARM / M1

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 설치

```
$ conda create -n venv  
$ conda activate venv  
$ conda install -c conda-forge spacy  
$ python -m spacy download en_core_web_trf  
$ python -m spacy download ko_core_news_lg
```



PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 분석기 설치

```
conda install -c conda-forge spacy
```

```
(TORCH_NLP) PS C:\Users\Wanece> conda install -c conda-forge spacy
Channels:
- conda-forge
- defaults
- anaconda
- pytorch
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\Wanece\Wanaconda3\envs\WTORCH_NLP

added / updated specs:
- spacy
```

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 한국어 모델 설치

```
python -m spacy download ko_core_news_sm
```

```
(TORCH_NLP) PS C:\Users\Wanece> python -m spacy download ko_core_news_sm
Collecting ko-core-news-sm==3.7.0
  Downloading https://github.com/explosion/spacy-models/releases/download/ko_core_news_sm-3.7.0/ko_core_news_sm-3.7.0-py3-none-any.whl (14.7 MB)
    14.7/14.7 MB 23.1 MB/s eta 0:00:00
Requirement already satisfied: spacy<3.8.0,>=3.7.0 in c:\Users\Wanece\anaconda3\envs\torch_nlp\lib\site-packages (from ko-core-news-sm==3.7.0) (3.7.6)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in c:\Users\Wanece\anaconda3\envs\torch_nlp\lib\site-packages (from spacy<3.8.0,>=3.7.0->ko-core-news-sm==3.7.0) (3.0.12)
```

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 형태소 분석

```
### 모듈 로딩
import spacy

### 한국어 모델 설정
KO_MODEL = 'ko_core_news_sm'

### 한국어 분석기 생성
nlp = spacy.load(KO_MODEL)
```

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 형태소 분석

```
text = "일만 하고 놀지 않으면 바보가 된다."  
doc = nlp(text)  
  
for token in doc:  
    # 표제어, 단어 품사, 자세한 품사, 불용어 여부  
    print(f'{token.text:6} {token.lemma_:6}  
          {token.pos_:6} {token.tag_:6} {token.is_stop}')  
    )
```

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 형태소 분석

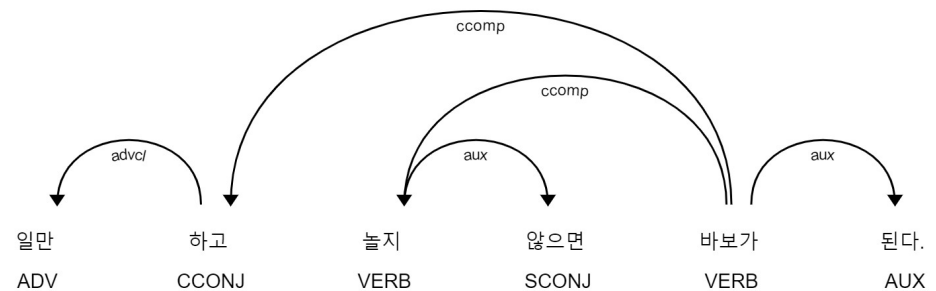
일만	일+만	ADV	ncpa+jxc	False
하고	하+고	CCONJ	pvg+ecc	False
놀지	놀+지	VERB	pvg+ecx	False
않으면	않+으면	SCONJ	px+ecs	False
바보가	바보+가	VERB	pvg+ecx	False
된다	되+ㄴ다	AUX	px+ef	False
.	.	PUNCT	sf	False

PREPROCESSING PROGRAMMING

◆ spaCy 형태소 분석기

❖ 문장 구조 시각화

```
from IPython.display import SVG
SVG(spacy.displacy.render(doc))
```



PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

- ❖ 코퍼스 다운로드 : koNLPy와 달리 패키지 내에서 말뭉치를 제공하지 않음!

```
pip install soynlp
```

```
Collecting mecab-ko
```

```
Using cached mecab_ko-1.0.1-cp38-cp38-win_amd64.whl.metadata (6.0 kB)
```

```
Collecting mecab-ko-dic<2.0,>=1.0 (from mecab-ko)
```

```
Using cached mecab_ko_dic-1.0.0-py3-none-any.whl
```

```
Using cached mecab_ko-1.0.1-cp38-cp38-win_amd64.whl (500 kB)
```

```
Installing collected packages: mecab-ko-dic, mecab-ko
```

```
Successfully installed mecab-ko-1.0.1 mecab-ko-dic-1.0.0
```

```
Note: you may need to restart the kernel to use updated packages.
```


PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

- ❖ 한국어 처리 위한 파이썬 패키지 중 하나
- ❖ koNLPy 형태소분석기는 형태소 기반으로 문서 토큰화
 - ➔ 새롭게 만들어진 미등록 단어들은 인식이 잘 되지 않는 단점
- ❖ 사용자 사전과 형태소 분석 없이 **cohesion** 기반 토큰화 할 수 있는 기능 제공

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ 코퍼스 다운로드 : koNLPy와 달리 패키지 내에서 **말뭉치를 제공하지 않음!**

```
# 데이터 파일 다운로드
from urllib.request import urlretrieve

data_url = 'https://raw.githubusercontent.com/lovit/soynlp/master/
           tutorials/2016-10-20.txt'
data_filename = '../data/2016-10-20.txt'

urlretrieve(data_url, data_filename)
```

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ 문서 단위 형태소 분석

```
# 모듈로딩
from soynlp import DoublespaceLineCorpus

# 문서 단위 말뭉치 생성
doc_corpus = DoublespaceLineCorpus(data_filename)

print(f'문서의 갯수 : {len(doc_corpus)} 개')

# 앞 5개 문서 출력
i = 0
for idx, d in enumerate(doc_corpus):
    print(idx, d)
    if idx > 4: break
```

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ 문장 단위 형태소 분석

```
# 문장 단위 말뭉치 생성
sent_corpus = DoublespaceLineCorpus(data_filename, iter_sent=True)
print(f'문장의 갯수 : {len(sent_corpus)} 개')

# 앞 5개 문서 출력
i = 0
for idx, d in enumerate(sent_corpus):
    print(idx, d)
    if idx > 4: break
```

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ 단어 단위 형태소 분석

```
## 단어 추출
from soynlp.word import WordExtractor

word_extractor = WordExtractor()
word_extractor.train(sent_corpus)

# cohesion, branching entropy, accessor variety 등 통계 수치 계산
word_score = word_extractor.extract()
```

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ Cohesion

문자열을 글자단위로 분리하여 부분문자열(substring)을 만들 때 왼쪽부터 문맥을 증가시키면서 각 문맥이 주어졌을 때 그 다음 글자가 나올 확률을 계산하여 누적곱을 한 값

```
word_score["연합"].cohesion_forward
```

✓ 0.0s

0.1943363253634125

```
word_score["연합뉴스"].cohesion_forward
```

✓ 0.0s

0.5710254410737682

```
word_score["연합뉴"].cohesion_forward
```

✓ 0.0s

0.43154839105434084

```
word_score["연합뉴스는"].cohesion_forward
```

✓ 0.0s

0.1535595043355021

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ Branching Entropy

확률분포의 엔트로피값을 사용

하나의 단어를 중간에서 끊으면 다음에 나올 글자 쉽게 예측 가능 → 값은 0에 가까움

하나의 단어 완결되는 위치는 다양한 조사나 결합 → 여러가지 글자 확률 비슷 → 값이 높음

```
word_score["연합"].right_branching_entropy
```

0.42721236711742844

```
# '연합뉴' 다음에는 항상 '스'만 나온다.  
word_score["연합뉴"].right_branching_entropy
```

-0.0

```
word_score["연합뉴스"].right_branching_entropy
```

3.8967810761022053

```
word_score["연합뉴스는"].right_branching_entropy
```

0.410116318288409

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ Accessor Variety

확률분포를 구하지 않고 단순히 특정 문자열 다음에 나올 수 있는 글자의 종류만 계산
글자의 종류가 많다면 엔트로피가 높아지리 것이라고 추정하는 것

```
word_score["연합"].right_accessor_variety
```

✓ 0.0s

42

```
# '연합뉴' 다음에는 항상 '스'만 나온다.  
word_score["연합뉴"].right_accessor_variety
```

✓ 0.0s

1

```
word_score["연합뉴스"].right_accessor_variety
```

✓ 0.0s

158

```
word_score["연합뉴스는"].right_accessor_variety
```

✓ 0.0s

2

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ L-토큰화

- 한국어의 경우 공백(띄어쓰기)으로 분리된 하나의 문자열은 'L 토큰 + R 토큰; 구조인 경우 많음
- 왼쪽에 오는 L 토큰 : 체언(명사, 대명사)이나 동사, 형용사 등
- 오른쪽 오는 R 토큰 : 조사, 동사, 형용사 등
- 여러가지 길이의 L 토큰의 점수를 비교하여 가장 점수가 높 L단어를 찾는 것

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ L-토큰화

```
from soynlp.tokenizer import LTokenizer

scores = {word:score.cohesion_forward for word, score in word_score.items()}
l_tokenizer = LTokenizer(scores=scores)

l_tokenizer.tokenize("안전성에 문제있는 스마트폰을 휴대하고 탑승할 경우에 압수한다",
flatten=False)
```

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ 최대점수 토큰화

- 띄어쓰기가 되어 있지 않는 긴 문자열에서 가능한 모든 종류의 부분문자열 추출
가장 점수가 높은 것을 하나의 토큰으로 정함
- 해당 토큰 제외, 전체 문자열이 다시 더 작은 문자열들로 나누어짐
- 더 작은 문자열들에 대해 다시 한번 가장 점수가 높은 부분문자열을 찾는 것을 반복

PREPROCESSING PROGRAMMING

◆ Soynlp 형태소 분석기

❖ 최대점수 토큰화

```
from soynlp.tokenizer import MaxScoreTokenizer

maxscore_tokenizer = MaxScoreTokenizer(scores=scores)
maxscore_tokenizer.tokenize("안전성에문제있는스마트폰을휴대하고탑승할경우에압수한다")
```

PREPROCESSING PROGRAMMING

◆ 서브워드 형태소 분석

- 하나의 단어를 여러 서브워드로 분리해서 단어를 인코딩 및 임베딩하겠다는 전처리 작업
- OOV나 희귀 단어, 신조어와 같은 문제를 완화시킬 수 있음
- 언어의 특성에 따라 영어권 언어나 한국어는 서브워드 분리를 시도했을 때 어느정도 의미 있는 단위로 나누는 것이 가능
- 주요 알고리즘 ➔ **바이트 페어 인코딩(Byte Pair Encoding)**

PREPROCESSING PROGRAMMING

◆ 서브워드 형태소 분석

❖ BPE(Byte Pair Encoding) 알고리즘 기반 서브단위 분절

- 1994년에 제안된 데이터 압축 알고리즘
- 자연어 처리의 서브워드 분리 알고리즘으로 응용
- [기본] 연속적으로 가장 많이 등장한 글자의 쌍을 찾아 하나의 글자로 병합
 - aaabdaaabc → 가장 자주 등장 바이트 쌍은 'aa' → 'Z'로 치환
 - ZabdZabc → 가장 자주 등장 바이트 쌍은 'ab' → 'Y'로 치환
 - ZYdZYac → 가장 자주 등장 바이트 쌍은 'ZY' → 'X'로 치환
 - XdXac → 더 이상 병합할 바이트 쌍 없음 → 종료

PART I

KOREAN DATASET

KOREAN DATASET

◆ 분류 분석(감성/의도분류)

네이버 영화 리뷰	네이버 영화 리뷰 데이터에 대한 긍/부정 라벨 데이터 - 학습 15만건 / 테스트 5만건	github
Toxic Comment Data	네이버 영화 리뷰 데이터의 라벨을 상세화한 데이터 - toxic / obscene / threat / insult / identity_hate 분류	github
3i4k	의도분류 학습용 데이터셋 - 문장에 대해 7가지 클래스 라벨 부여 - 논문: https://arxiv.org/pdf/1811.04231.pdf	github
korean-hage-speech	한국어 혐오발언 분류 데이터셋 - 연예 뉴스 댓글에 대한 혐오 / 사회적 편견 유무 라벨 데이터 - 사회적 편견은 성별/ 기타/ 없음 세 가지로 분류 - 9,381건(7,896 / 471 / 974)	github

KOREAN DATASET

◆ 유사도 판별

KorNLI	두 문장의 관계를 entailment/neutral/contradiction 으로 분류 - 학습/ 검증/ 테스트 데이터로 분리되어 있음.	github
KoSTS	두 문장의 유사도 점수를 라벨링한 데이터 - 학습/ 검증/ 테스트 데이터로 분리되어 있음.	github
Question pair	두 개의 질문이 같은 질문인지 아닌지 레이블링한 데이터 - 학습 6,888건 / 테스트 688건 제공	github
ParaKQC	10개의 비슷한 문장에 대한 1,000개의 집합으로 구성 - 문장 유사도 데이터 494,500건 생성 가능 - 패러프레이징 데이터 45,000건 생성 가능	github

KOREAN DATASET

◆ 자연어 질의응답(기계독해 / MRC)

KorQuAD 1.0	한국어 기계독해를 위한 표준 데이터셋 - 리더보드 운영중	webpage
KorQuAD 2.0	구조를 가진 HTML 문서에 대한 기계 독해 데이터셋 - 리더보드 운영중	webpage
AI HUB 기계독해	한국어 기계독해를 위한 데이터셋 - SQuAD1.0 / 2.0(no-answer) 타입의 데이터 제공 - 질문 답변과 답변을 선택한 단서 제공	webpage

KOREAN DATASET

◆ 대화 모델

Chatbot_data	일상 챗봇 학습용 문답 페어 11,876건 일상(0) / 이별,부정(1) / 사랑,긍정(2) 라벨 부여	github
AI HUB 한국어대화	소상공인, 공공민원 관련 10개 분야 대한 1만건 이상 대화 데이터셋 - 대화 데이터와 함께 의도(Intent) 라벨링 포함	webpage
클로바 AI Call 데이터	자동 음성 인식을 위한 Goal-oriented 대화 음성 코퍼스 데이터셋 - 비영리/ 학계 소속 기관은 신청을 통해 다운로드 가능	github
웰니스 대화 스크립트	세브란스 상담 데이터를 기반으로 구축한 정신 상담 데이터셋 - 359개 대화의도에 대한 5,232개 사용자 발화 - 1,023개 챗봇 발화 포함	webpage
KETI 한국어 대화 데이터	멀티턴 대화 데이터 758개 - 각 대화는 3~12개의 턴으로 구성 - 문장 단위로는 4,975건의 발화문 존재	webpage

KOREAN DATASET

◆ 대화 모델

트위터기반 일상 대화	트위터상에 둘 이상의 화자가 대화한 내용 모음 - 1~17 턴의 데이터로 구성되어 있음 - 1차년도 데이터로 2,000 세트가 공개됨	webpage
대화형 한글 에이전트	영화/드라마/SNS등에 대한 멀티턴 대화 데이터 - 8,000개의 대화 세트 포함 - 각 대화는 2~10 턴의 대화 포함	webpage
한국어 감정 정 포함된 연속적 대화 데이터셋	크롤링으로 수집한 멀티턴 대화 데이터셋 - 각 발화문은 7가지 감정정보로 레이블링되어 있음 - 10,000개의 대화 세트/ 각 대화는 약 5.6개 턴으로 구성 - 문장 단위로는 55,627건의 발화문 존재	webpage

KOREAN DATASET

◆ RAW CORPUS

국립국어원 말뭉치	다양한 분야에 대한 방대한 한국어 raw 코퍼스	webpage
카이스트 코퍼스	1994~1997년 수집한 70,000,000 어절의 코퍼스	webpage
위키피디아 덤프	한국어 위키피디아 (추출기- 링크)	webpage
나무위키 덤프	나무위키 (추출기 - 링크)	webpage
한국 정치인 관련 뉴스	한국 정치인 19인에 대한 뉴스 수집 데이터셋	github
인공지능 윤리연구 위한 비정형 텍스트 데이터셋	윤리 연구를 위해 윤리/비윤리 데이터 코퍼스 구축 1차년도: 뉴스기사 댓글 7,000만 건, 트위터 3,000만 건 2차년도: 온라인커뮤니티 (일베저장소) 댓글 4,500만 건 3차년도: 온라인커뮤니티 (일베저장소) 댓글 2,000만 건	webpage