# 파이토치로 배우는 자연어 처리
# 2장. NLP 기술 빠르게 훑어보기

박채원

# 목차

# 말뭉치, 토큰, 타입

모든 NLP 작업은 말뭉치라 부르는 텍스트 데이터로 시작.

텍스트 + 메타 데이터 = 샘플 or 데이터 포인트
말뭉치(데이터셋) : 원시텍스트 (ASCII, UTF-8 등의 문자 시퀀스) + 메타 데이터(텍스트와 관련된 부가 정보)
-> **토큰**이라는 연속 단위로 묶었을 때 유용 (*토큰 - 공백 문자나 구두점으로 구분되는 단어와 숫자)

토큰화 : 텍스트를 토큰으로 나누는 과정
* 교착어는 공백과 구두점으로 나누는 것으로는 충분하지 않다. 더 전문 기술이 필요 -> 텍스트를 바이트 스트림으로 신경망에 표현.

오픈소스 NLP 패키지는 대부분 기본적인 토큰화를 제공.

ex) NLTK 토크나이저

```
from nltk.tokenize import TweetTokenizer
tweet = u"Snow White and the Seven Degrees #MakeAMovieCold@midnight:-)"
tokenizer = TweetTokenizer()
print(tokenizer.tokenize(tweet.lower()))
```

```
['snow', 'white', 'and', 'the', 'seven', 'degrees', '#makeamoviecold', '@midnight', ':-)']
```

# 유니그램, 바이그램, 트라이그램, … , n-그램

n-그램 : 텍스트에 있는 고정 길이(n)의 연속된 토큰 시퀀스
즉 유니그램은 토큰 한 개, 바이그램은 토큰 두 개.
spaCy와 NLTK 같은 패키지에서도 n-그램을 편리하게 만드는 도구 제공

```
def n_grams(text, n):
  return [text[i:i+n] for i in range (len(text)-n+1)]

cleaned = ['mary', ',', "n't", 'slap', 'green', 'witch', '.']
print(n_grams(cleaned,3))
```

```
[['mary', ',', "n't"], [',', "n't", 'slap'], ["n't", 'slap', 'green'], ['slap', 'green', 'witch'], ['green', 'witch', '.']]
```

부분 단어 자체가 유용한 정보를 전달한다면 **문자** n-그램을 생성할 수 있다.
ex) methanol의 접미사 -ol은 알코올 종류를 나타냄. 이는 유용하게 사용될 수 있다.

# 표제어와 어간

표제어 : 단어의 기본형
ex) flow, flew, flies, flown, flowing 의 표제어 : fly

표제어 추출 : 토큰을 표제어로 바꾸는 것. 벡터 표현의 차원을 줄일 수 있다.

```
import spacy

nlp = spacy.load('en')
doc = nlp("he was running late")
for token in doc:
  print('{} --> {}'.format(token, token.lemma_))
```

```
he --> -PRON-
was --> be
running --> run
late --> late
```

spacy는 사전에 정의된 WordNet 사전 사용하여 표제어 추출
하지만 표제어 추출은 언어 형태론을 이해하려는 머신러닝의 문제로 나타낼 수 있다.

-> 어간 추출 사용.
: 수동으로 만든 규칙을 사용해 단어의 끝을 잘라 어간이라는 공통 형태로 축소한다.
잘라진 단어는 현존하는 단어가 아닐 수 있다.
각 알고리즘 마다 결과가 다르다.

# 문장과 문서 분류하기

앞장에서 소개한 TF와 TF-IDF 표현이 문서나 문장 같은 긴 텍스트 뭉치를 분류하는 데 유용하다.

# 단어 분류하기 : 품사 태깅

문서에 레이블을 할당하는 개념을 단어나 토큰으로 확장할 수 있다.
이 또한 패키지에서 제공한다.

```
import spacy
nlp = spacy.load('en')
doc = nlp(u"Mary slapped the green witch.")
for token in doc:
  print('{} - {}'.format(token, token.pos_))
```

```
Mary - PROPN
slapped - VERB
the - DET
green - ADJ
witch - NOUN
. - PUNCT
```

# 청크 나누기와 개체명 인식

연속된 여러 토큰으로 구분되는 텍스트 **구**에 레이블을 할당해야 한다.

ex) "Mary slapped the green witch" 문장을 명사구와 동사구로 구별해야 한다면?
-> 청크 나누기(부분 구문 분석, 구 단위를 식별) 사용, 고차원의 단위를 유도해 내는 것
이 또한 패키지에서 제공된다.

```
import spacy
nlp = spacy.load('en')
doc = nlp(u"Mary slapped the green witch.")
for chunk in doc.noun_chunks:
  print('{} - {}'.format(chunk, chunk.label_))
```

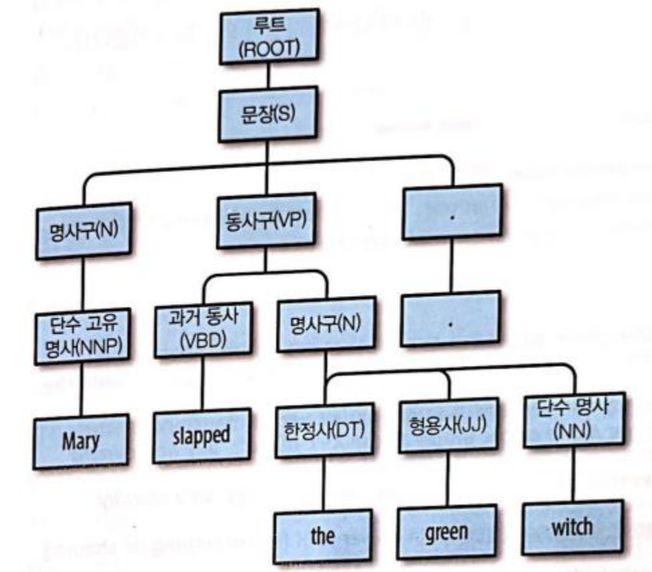
```
Mary - NP
the green witch - NP
```

또 다

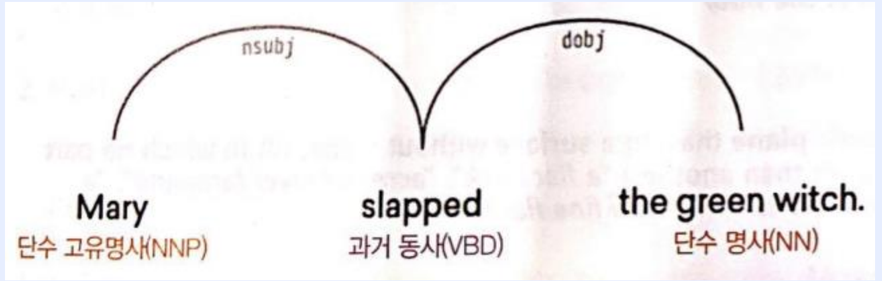
개체명 : 실제 세상의 개념을 의미하는 문자열. ex) person(인물), gpe(나라 및 도시), org(기관) 등등...
nltk에서 ne_chunk를 이용해 개체명 인식할 수 있음.

# 문장 구조

구문 분석 : 구 사이의 관계를 파악하는 작업




구성 구문 분석
문장 안의 문법 요소가 계층적으로 어떻게 관련되는지

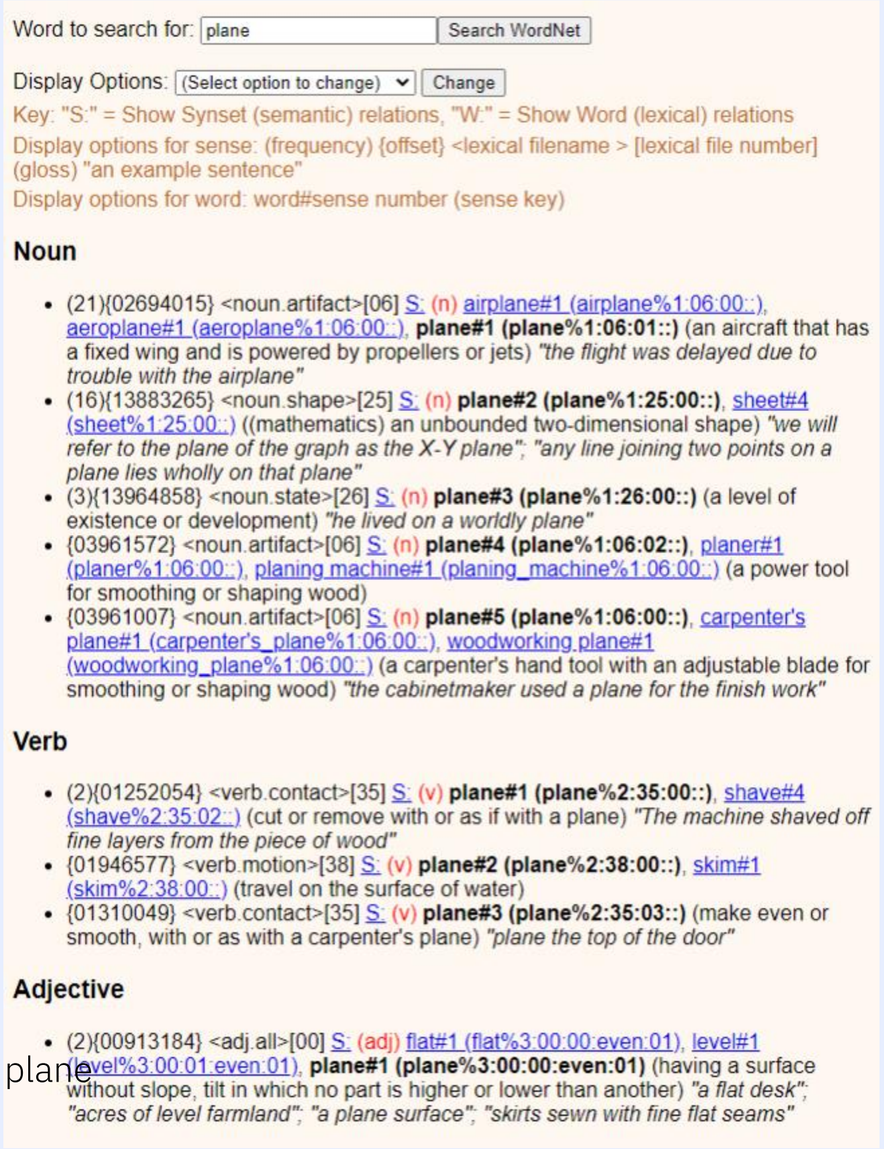



의존 구문 분석
각 성분이 서로 의존 관계를 이루어 하나의 구문을 이룬다고 본다.

# 단어 의미와 의미론

단어에는 의미가 하나 이상 존재한다.
프린스턴 대학교에서 장기간 진행 중인 어휘 사전 프로젝트
WordNet은 거의 모든 영단어의 관계와 의미를 수집하는 것이 목표.

예시) WordNet에서의 plane 검색 결과 ▶



Word to search for: plane Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (frequency) {offset} <lexical filename > [lexical file number] (gloss) "an example sentence"

Display options for word: word#sense number (sense key)

**Noun**

- (21){02694015} <noun.artifact>[06] S: (n) airplane#1 (airplane%1:06:00::), aeroplane#1 (aeroplane%1:06:00::), **plane#1 (plane%1:06:01::)** (an aircraft that has a fixed wing and is powered by propellers or jets) *"the flight was delayed due to trouble with the airplane"*
- (16){13883265} <noun.shape>[25] S: (n) **plane#2 (plane%1:25:00::)**, sheet#4 (sheet%1:25:00::) ((mathematics) an unbounded two-dimensional shape) *"we will refer to the plane of the graph as the X-Y plane"; "any line joining two points on a plane lies wholly on that plane"*
- (3){13964858} <noun.state>[26] S: (n) **plane#3 (plane%1:26:00::)** (a level of existence or development) *"he lived on a worldly plane"*
- {03961572} <noun.artifact>[06] S: (n) **plane#4 (plane%1:06:02::)**, planer#1 (planer%1:06:00::), planing machine#1 (planing_machine%1:06:00::) (a power tool for smoothing or shaping wood)
- {03961007} <noun.artifact>[06] S: (n) **plane#5 (plane%1:06:00::)**, carpenter's plane#1 (carpenter's_plane%1:06:00::), woodworking plane#1 (woodworking_plane%1:06:00::) (a carpenter's hand tool with an adjustable blade for smoothing or shaping wood) *"the cabinetmaker used a plane for the finish work"*

**Verb**

- (2){01252054} <verb.contact>[35] S: (v) **plane#1 (plane%2:35:00::)**, shave#4 (shave%2:35:02::) (cut or remove with or as if with a plane) *"The machine shaved off fine layers from the piece of wood"*
- {01946577} <verb.motion>[38] S: (v) **plane#2 (plane%2:38:00::)**, skim#1 (skim%2:38:00::) (travel on the surface of water)
- {01310049} <verb.contact>[35] S: (v) **plane#3 (plane%2:35:03::)** (make even or smooth, with or as with a carpenter's plane) *"plane the top of the door"*

**Adjective**

- (2){00913184} <adj.all>[00] S: (adj) flat#1 (flat%3:00:00:even:01), level#1 (level%3:00:01:even:01), **plane#1 (plane%3:00:00:even:01)** (having a surface without slope, tilt in which no part is higher or lower than another) *"a flat desk"; "acres of level farmland"; "a plane surface"; "skirts sewn with fine flat seams"*

2장. NLP 빠르게 훑어보기

# Thank you