

Classifier based Noun Extractor

Hyunjoong Kim

soy.lovit@gmail.com

github.com/lovit

Noun Extraction

- Word extractor & unsupervised tokenizers 에서는 분석할 데이터를 기반으로, 그 데이터에서 사용되는 단어를 추출하는 방법을 다뤘습니다.
- 이번에는 {'파스타', '좋아'} 와 같은 단어에서 '파스타'를 명사로 인식하는 방법을 다뤄봅니다.

A는 명사일까?

어제 A라는 가게에 가봤어

A에서 보자

A로 와줘

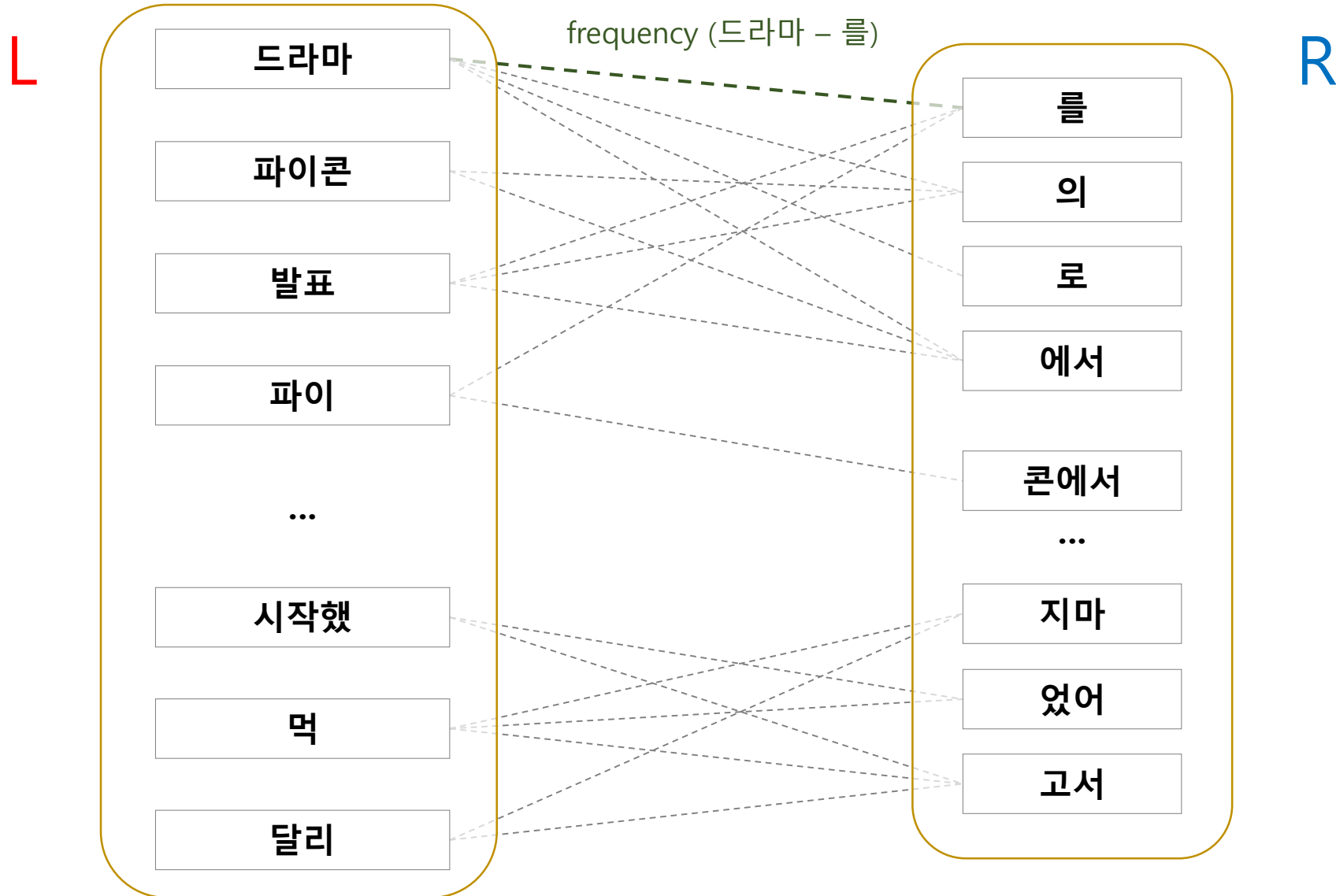
명사 우측에 등장하는 글자 분포를 이용하여

A가 명사임을 유추할 수 있다

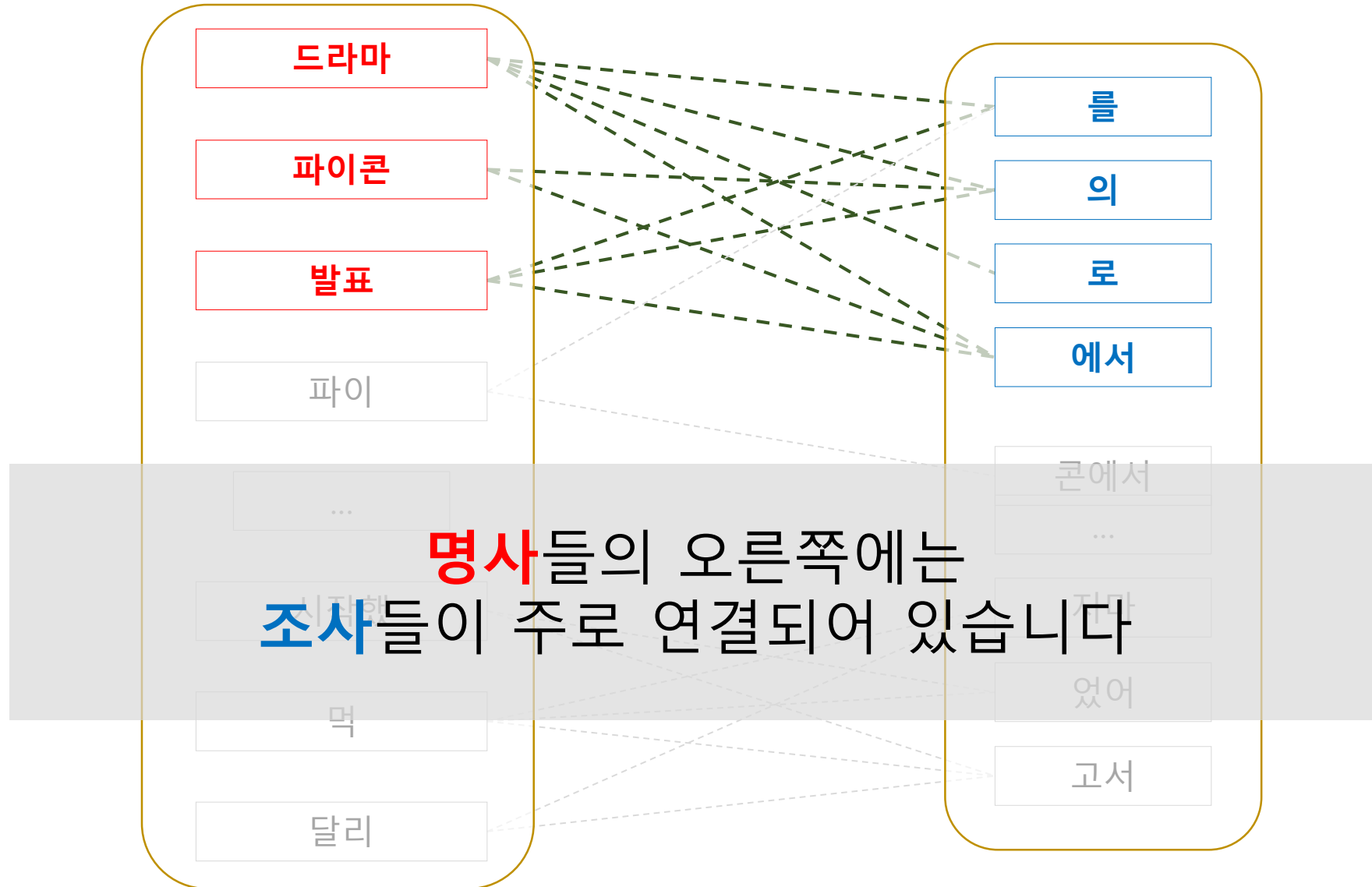
L – R graph

- 어절은 L + [R] 구조로 나눌 수 있습니다
 - 발표 + 를
 - 하 + 면서
- 데이터에서 모든 어절들을 두 개의 subwords로 나눈 뒤 연결하면 L – R graph를 만들 수 있습니다

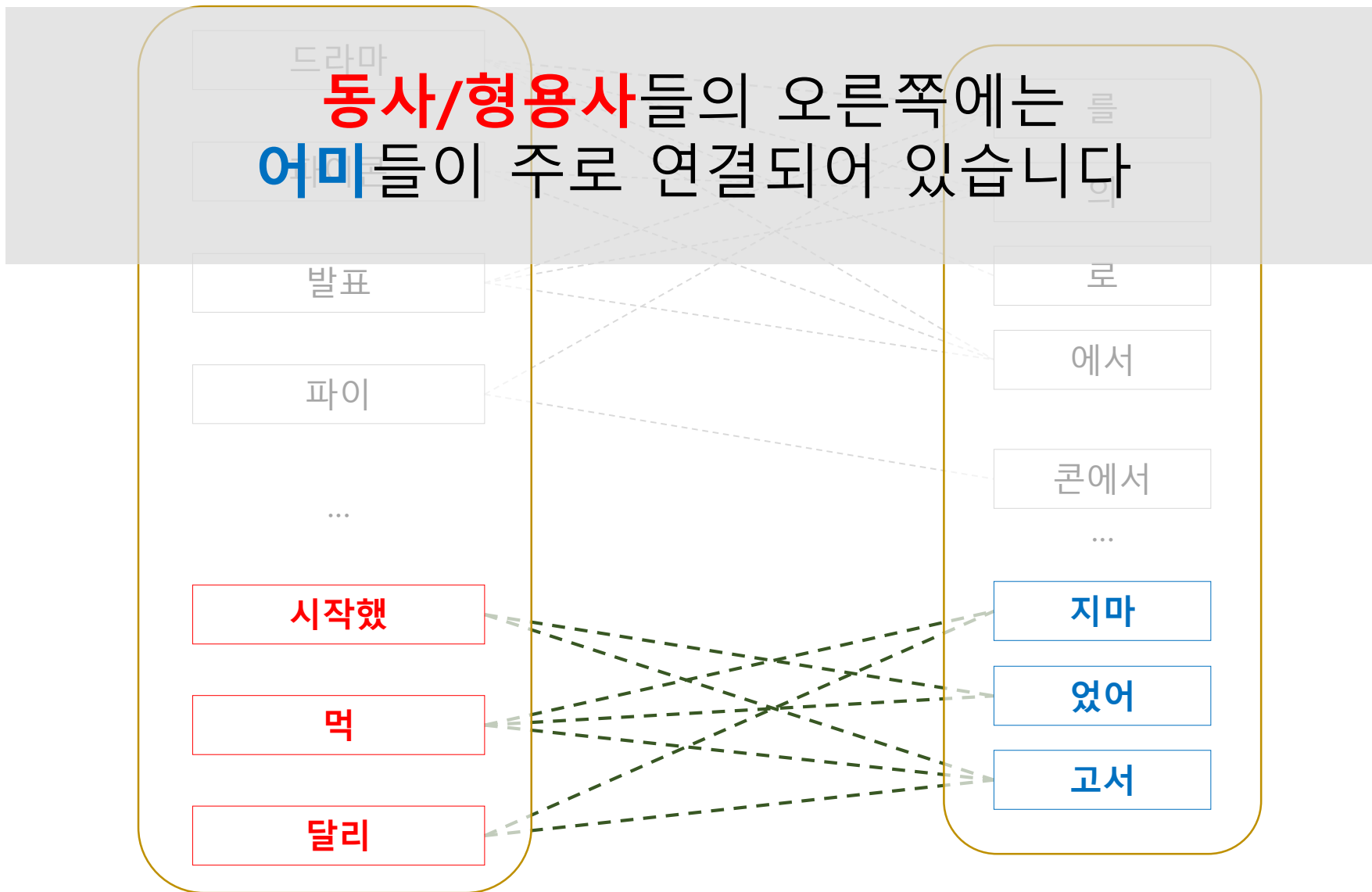
L - R graph



L – R graph

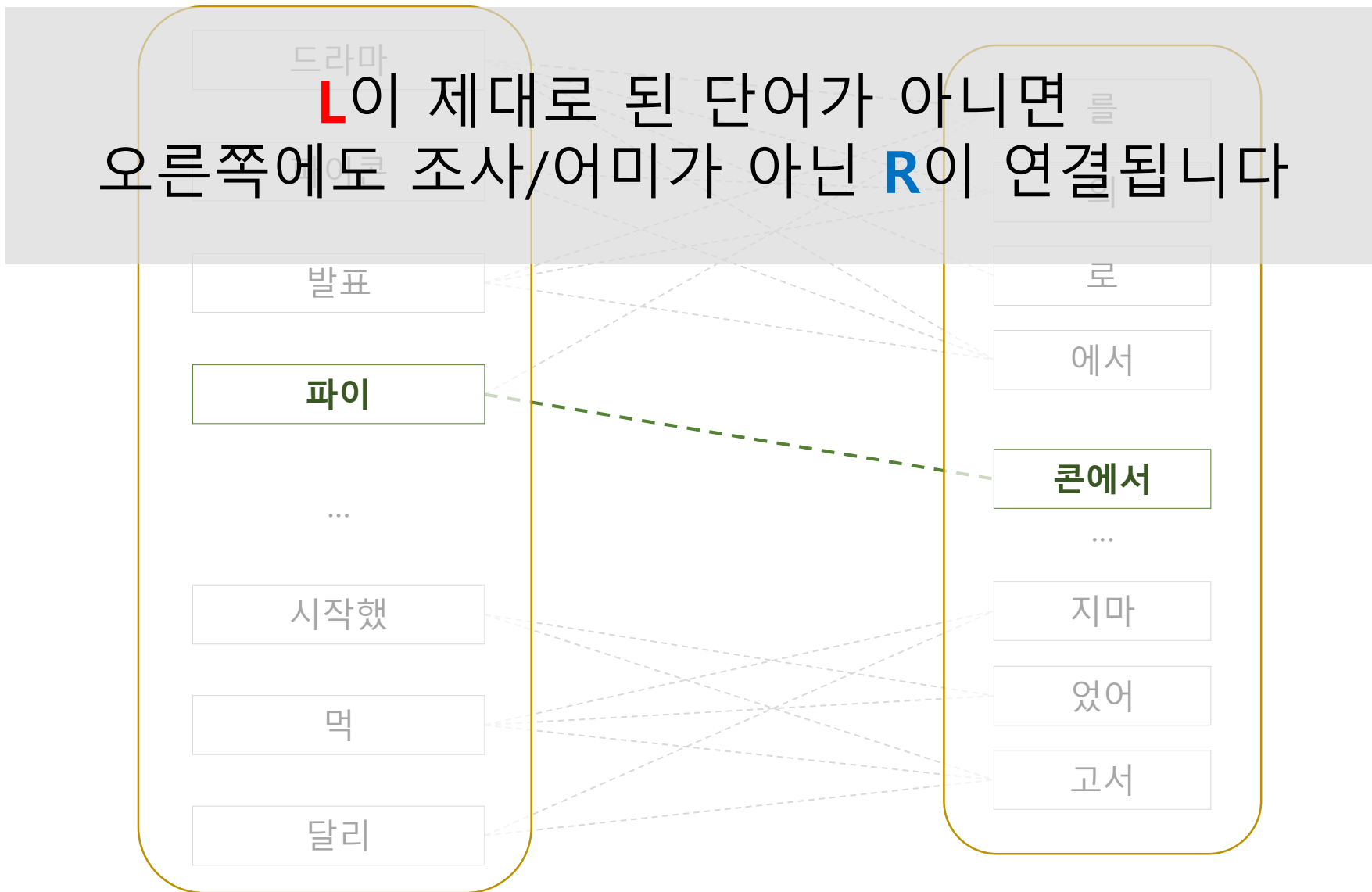


L – R graph



L - R graph

L이 제대로 된 단어가 아니면
오른쪽에도 조사/어미가 아닌 **R**이 연결됩니다



Noun Extraction

- L – R graph: **[명사 + 조사]**, [동사 + 어미], [틀린 단어 + 틀린 단어]

get_r('드라마')

```
[(' ', 1268),  
 ('를', 164),  
 ('다', 152),  
 ('의', 140),  
 ('로', 138),  
 ('에서', 98),  
 ('와', 62),  
 ('는', 55),  
 ('에', 55),  
 ('가', 48),  
 ('이다', 24),  
 ('인', 14),  
 ... ]
```

get_r('시작했')

```
[('다', 567),  
 ('고', 73),  
 ('다고', 61),  
 ('습니다', 42),  
 ('는데', 26),  
 ('으며', 16),  
 ('지만', 15),  
 ('던', 12),  
 ('어요', 10),  
 ('다는', 7),  
 ('으나', 5),  
 ('죠', 4),  
 ... ]
```

get_r('드라')

```
[('마', 1268),  
 ('마를', 164),  
 ('마다', 152),  
 ('마의', 140),  
 ('마로', 138),  
 ('마에서', 98),  
 ('기', 65),  
 ('마와', 62),  
 ('마는', 55),  
 ('마에', 55),  
 ('마가', 48),  
 ('이브', 28),  
 ... ]
```

Noun Extraction

- L – R graph: [명사 + 조사], **[동사 + 어미]**, [틀린 단어 + 틀린 단어]

get_r(' 드라마')

```
[(' ', 1268),  
 ('를', 164),  
 ('다', 152),  
 ('의', 140),  
 ('로', 138),  
 ('에서', 98),  
 ('와', 62),  
 ('는', 55),  
 ('에', 55),  
 ('가', 48),  
 ('이다', 24),  
 ('인', 14),  
 ... ]
```

get_r(' **시작했**')

```
[('다', 567),  
 ('고', 73),  
 ('다고', 61),  
 ('습니다', 42),  
 ('는데', 26),  
 ('으며', 16),  
 ('지만', 15),  
 ('던', 12),  
 ('어요', 10),  
 ('다는', 7),  
 ('으나', 5),  
 ('죠', 4),  
 ... ]
```

get_r('드라')

```
[('마', 1268),  
 ('마를', 164),  
 ('마다', 152),  
 ('마의', 140),  
 ('마로', 138),  
 ('마에서', 98),  
 ('기', 65),  
 ('마와', 62),  
 ('마는', 55),  
 ('마에', 55),  
 ('마가', 48),  
 ('이브', 28),  
 ... ]
```

Noun Extraction

- L – R graph: [명사 + 조사], [동사 + 어미], **[틀린 단어 + 틀린 단어]**

get_r(' 드라마')

```
[(' ', 1268),  
 ('를', 164),  
 ('다', 152),  
 ('의', 140),  
 ('로', 138),  
 ('에서', 98),  
 ('와', 62),  
 ('는', 55),  
 ('에', 55),  
 ('가', 48),  
 ('이다', 24),  
 ('인', 14),  
 ... ]
```

get_r('시작했')

```
[('다', 567),  
 ('고', 73),  
 ('다고', 61),  
 ('습니다', 42),  
 ('는데', 26),  
 ('으며', 16),  
 ('지만', 15),  
 ('던', 12),  
 ('어요', 10),  
 ('다는', 7),  
 ('으나', 5),  
 ('죠', 4),  
 ... ]
```

get_r(' **드라**')

```
[('마', 1268),  
 ('마를', 164),  
 ('마다', 152),  
 ('마의', 140),  
 ('마로', 138),  
 ('마에서', 98),  
 ('기', 65),  
 ('마와', 62),  
 ('마는', 55),  
 ('마에', 55),  
 ('마가', 48),  
 ('이브', 28),  
 ... ]
```

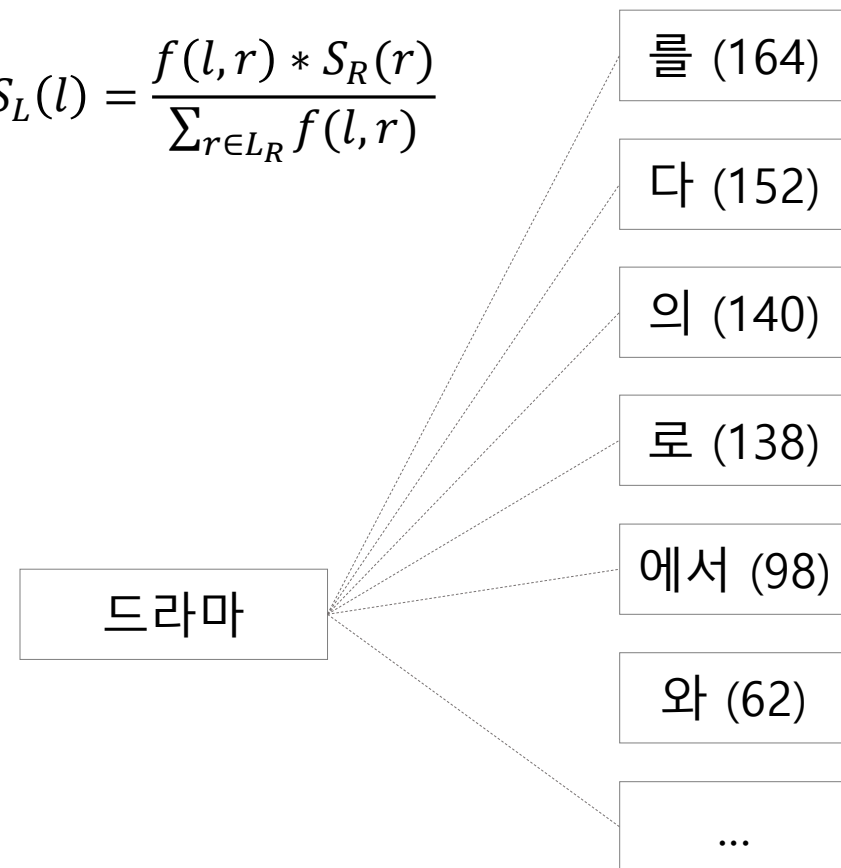
Noun Extraction

- R 의 분포를 이용하여 L의 명사 점수를 계산할 수 있습니다

```
r_scores = {'은': 0.5, '있다' : -0.9, ... }
```

```
def noun_score(L):  
    (norm, score, _total) = (0, 0, 0)  
  
    for R, frequency in get_r(L):  
        _total += frequency  
        if not R in r_scores:  
            continue  
        norm += frequency  
        score += frequency * r_scores[R]  
  
    score = score / norm if norm else 0  
    prop = norm / _total if _total else 0  
    return score, prop
```

$$S_L(l) = \frac{f(l, r) * S_R(r)}{\sum_{r \in L_R} f(l, r)}$$



Noun Extraction

- R 의 분포를 이용하여 L의 명사 점수를 계산할 수 있습니다

(명사 점수, 알려진 R 비율) = noun_score('Word')

```
noun_score('드라마')
```

```
(0.574, 0.921)
```

```
noun_score('시작했')
```

```
(-0.976, 0.999)
```

```
noun_score('드라')
```

```
(-0.661, 0.579)
```

틀린 단어 뒤에는 조사/어미 등이 아닌
글자들이 등장

Noun Extraction

- 세종 말뭉치의 품사가 태깅된 정보로부터 L – R 구조의 테이블을 만듭니다
 - R frequency vector를 이용하여 L이 명사인지 판단하는 Logistic Regression을 학습하여, 이의 coefficients를 r score table로 이용합니다.

...
 예술가의 예술가/NNG+의/JKG 113
 예술가는 예술가/NNG+는/JX 45
 예술가가 예술가/NNG+가/JKS 43
 예술가들의 예술가/NNG+들/XSN+의/JKG 30
 ...



단어 품사	단어 / R	- 는	- 의	- 고	- 었다	- 었던
명사	예술가	45	113	2	0	0
동사	먹	33	0	27	0	27
...

Noun Extraction

- – R 앞의 명사 빈도를 이용하여 R score 를 계산할 수도 있습니다.
- R의 명사 가능 점수 (s_R)를 $[-1, 1]$ 로 스케일링 한 효과가 있습니다

$$2 * (p(N|R) - 0.5)$$

$$s_R(-다/R) = 2 * (0.056 - 0.5) = -0.888$$

$$s_R(-은/R) = 2 * (0.73 - 0.5) = 0.46$$

Normalize R proportion by L distribution

R	Noun	Verb/Adj
-다/R	0.056	0.944
-은/R	0.73	0.265
...

Noun Extraction

- R 앞의 명사의 빈도를 이용하여 명사가 등장할 점수를 계산

품사	단어 / R	- 는	- 의	- 고	- 었다	-구나
명사	예술가	45	113	2	0	0
명사	나이	54	87	27	0	27
동사	들	0	0	0	255	0
형용사	춡	0	0	0	0	87
...



품사	빈도수 (비율)
명사	704,197 (0.838)
^명사	136,598 (0.162)

품사	빈도수	보정빈도수	점수
명사	980	980	0.810
^명사	20	$103.5 = 20 * (0.838/0.162)$	$= (980 - 103.5) / (980 + 103.5)$

Noun Extraction: post-processing

- 오류를 막기 위한 후처리 기능이 필요합니다
 - R frequency vector를 데이터로 이용하는 classifier는 다음의 경우 잘못된 판단을 할 수 있습니다.

Noun Extraction: post-processing

- $N = N_{\text{sub}} + J$
 - 떡볶이 + 이
 - '-이'는 대표적 조사이며, '떡볶이' 자체로 어절로 이용기도 하여 '떡볶' 이 명사로 잘못 판단될 수 있습니다
- $N + J_{\text{sub}}$
 - 대학생으 + 로
 - '-로'는 대표적 조사이기 때문에 '-으로'가 잘못 나뉘어질 경우, '대학생의' 역시 명사로 잘못 판단될 수 있습니다

Noun Extraction: post-processing

- '떡볶 + 이'의 경우,

- {'떡볶이'가 명사이고} and {끝 부분이 1음절 조사일 경우} '떡볶'을 명사에서 제외

- '대학생으 + 로'의 경우,

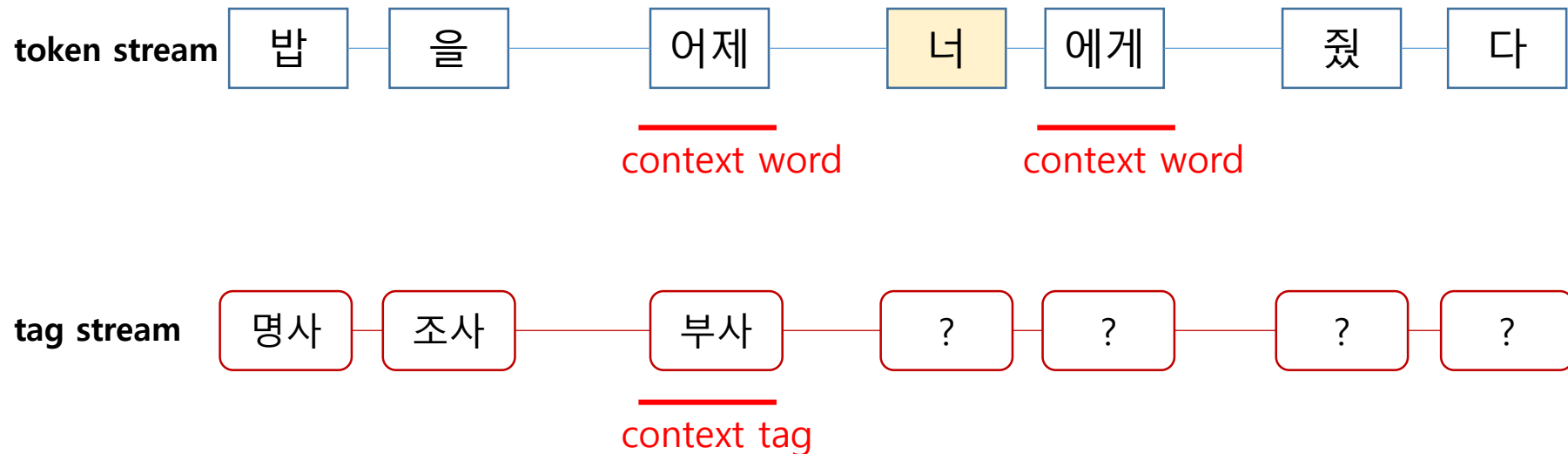
- {'대학생'이 명사이고} and { '-으 + 로 = -으로'가 조사일 경우} '대학생으'를 명사에서 제외

Noun extraction

- Topic tracking, topic modeling 에는 concepts 을 나타내는 명사만을 이용하는 것도 좋습니다.
- 더 좋은 명사인식 / 품사판별을 위해서는 각 문장을 독립적으로 보지 않고, 문서 전체의 경향을 볼 필요가 있습니다 (global information)

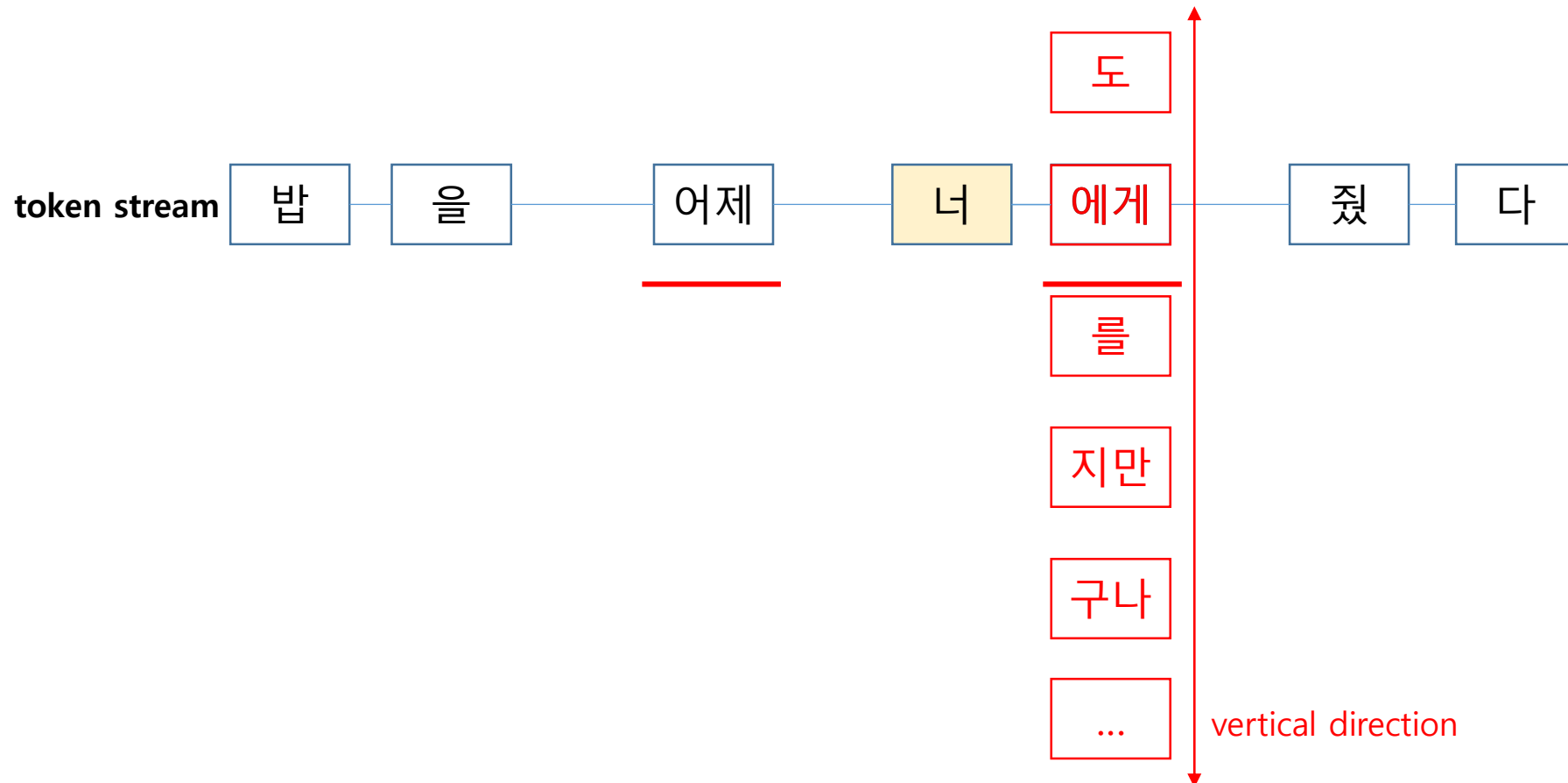
Noun extraction

- Sequential labeling 기반 품사 판별기는 각 문장을 독립적으로 다룹니다.
 - 앞/뒤 단어의 문맥 정보를 이용하여 품사 판별을 수행합니다.



Noun extraction

- 한 문장을 독립적으로 다루지 않고, 모든 문장에서의 “너/L” 의 오른쪽에 등장하는 R 의 분포를 이용하면 사전에 존재하지 않는 명사도 인식합니다.



- 깔끔한 뉴스 데이터에서는 명사가 잘 추출됩니다

덴마크	웃돈	너무너무너무	가락동	매뉴얼	지도교수
전망치	강구	언니들	신산업	기뢰전	노스
할리우드	플라자	불법조업	월스트리트저널	2022년	불허
고씨	어플	1987년	불씨	적기	레스
스퀘어	충당금	건축물	뉴질랜드	사각	하나씩
근대	투자주체별	4위	태권	네트웍스	모바일게임
연동	런칭	만성	손질	제작법	현실화
오해영	심사위원들	단점	부장조리	차관급	게시물
인터폰	원화	단기간	편곡	무산	외국인들
세무조사	석유화학	워킹	원피스	서장	공범

- 아직 노이지 데이터에서는 성능 개선이 필요합니다

변심	인시디어스	코디님	강만후	병원예약	전공필수
이런의미	바깥양반	프레시아	전리품상자	난심일기	근육몬
동피랑	상태이상	흐흐르	수입과자	모래사장	알엔에이
오션윙	옥매미	산넘어산	클럽하우스	물한번	손에다
따른사람	모진말	노들섬	한신아파트	괜차니	핵고생
다음장	아이도루	본사사람	마스크걸	브런치카페	클렌징워터
배아플라	일일미션	버스정거장	리장	안산도착	다리하나
빠른사과	용계	아홉시꺼	걸신	컴퓨터	필터빨
성큼성큼	서가엔쿱	경중	스카이워크	인형만	하루시작
판의점	신청하기	미모짤	재밌는시간	운동하규	간사들

soynlp

- 후처리과정이 추가된 명사 추출기를 이용중입니다

```
from soynlp.noun import LRNounExtractor
```

```
noun_extractor = LRNounExtractor(min_count=50)
```

```
nouns = noun_extractor.train_extract(corpus, minimum_noun_score=0.5)
```

```
nouns['설입']
```

```
NounScore(frequency=67, score=0.926, known_r_ratio=0.529)
```

```
nouns['드라마']
```

```
NounScore(frequency=4976, score=0.522, known_r_ratio=0.601)
```