

노가다 없이 한국어 뉴스/댓글 데이터 분석하기

Hyunjoong Kim

soy.lovit@gmail.com

<https://github.com/lovit/soynlp>

품사 판별과 형태소 분석

- 품사 판별은 텍스트 데이터 분석을 위한 전처리 과정 중 하나입니다

```
from konlpy.tag import Kkma
```

```
kkma = Kkma()
```

```
kkma.pos('오류보고는 실행환경, 에러메세지와함께 설명을 최대한상세히!^^')
```

```
[(오류, NNG), (보고, NNG), (는, JX), (실행, NNG), (환경, NNG), (,, SP),  
 (에러, NNG), (메세지, NNG), (와, JKM), (함께, MAG), (설명, NNG), (을, JKO),  
 (최대한, NNG), (상세히, MAG), (!, SF), (^^, EMO)]
```

품사 판별과 형태소 분석

- 품사 판별을 위하여 형태소 분석이 이용될 수 있습니다

SENT: 재공연을 했어요

POS: (재공연, 명사), (을, 조사), (했어요, 동사)

MORPHEMES: (재, 관형사), (공연, 명사), (을, 조사), (하, 동사), (었, 선어말어미), (어요, 종결어미)

- 형태소 분석은 단어의 구성 요소들을 분해하여 인식하는 과정입니다

품사 판별과 형태소 분석

- 품사 사전이 잘 구축된다면, **사전기반으로도 품사판별**을 할 수 있습니다

SENT: 재공연을 했어요

POS: (재공연, 명사), (을, 조사), (했어요, 동사)

명사사전: { ... 재공연, ... }

동사사전: { ... 했어요, 했엉, 해써용, ... }

- 품사 판별이 목적이라면 형태소분석 과정이 필수는 아닙니다

미등록단어 문제

- 하지만 형태소분석은 **새롭게 만들어진 단어**를 인식하기 어렵습니다

```
from konlpy.tag import Kkma, Twitter
kkma = Kkma()
kkma.pos('너무너무너무는 아이오아이의 노래예요')
```

너무/MAG, 너무너무/MAG, 는/JX, 아이오/NNG, 아이/NNG, 의/JKG, 노래/NNG, 예/JKM, 요/JX

```
twitter = Twitter()
twitter.pos('너무너무너무는 아이오아이의 노래예요')
```

너무/Noun, 너무/Noun, 너무/Noun, 는/Josa, 아이오/Noun, 아이/Noun, 의/Josa, 노래/Noun, 예요/Josa

미등록단어 문제

- 하지만 형태소분석은 **새롭게 만들어진 단어**를 인식하기 어렵습니다

```
from konlpy.tag import Kkma, Twitter
kkma = Kkma()
kkma.pos('최순실 게이트로 인해 비선실세가 드러났다')
```

최/NNP, 순/NNG, 실/NNG, 게이트/NNG, 로/JKM, 인하/VV, 어/ECS, 비선/NNG, 실세/NNG, 가/JKS, 드러나/VV, 었/EPT, 다/EFN

```
twitter = Twitter()
twitter.pos('최순실 게이트로 인해 비선실세가 드러났다')
```

최순실/Noun, 게이트/Noun, 로/Josa, 인해/Verb, 비/Noun, 선/Verb, 실/PreEomi, 세/PreEomi, 가/Eomi, 드러났/Verb, 다/Eomi

미등록단어 문제

- 형태소 사전 기반으로 형태소 분석을 수행할 경우, 미등록단어를 **알려진 형태소들로 분해**할 가능성이 있습니다

SENT: 집에가용

Kkma: [('집', 'NNG'), ('에', 'JKM'), ('가용', 'NNG')]

Twitter: [('집', 'Noun'), ('에', 'Josa'), ('가용', 'Noun')]

Hannanum: [('집에가용', 'N')]

SENT: 비선실세

Kkma: [('비선', 'NNG'), ('실세', 'NNG')]

Twitter: [('비', 'Noun'), ('선', 'Verb'), ('실', 'PreEomi'), ('세', 'Eomi')]

Hannanum: [('비선실세', 'N')]

미등록단어 문제

- 사전 기반으로 작동하는 형태소/품사 분석은 **사전 구성이 핵심**입니다
 - 이 과정은 주로 **노동집약적인 수작업**으로 이뤄집니다.
- 사전에 단어를 추가하는 과정을 자동화 할 수 있다면,
 - 효율적인 데이터 정제로, 데이터 분석에 집중할 수 있고,
 - 수작업이 적으므로, **다양한 도메인**의 분석에 **유연하게 적용될** 것입니다.

우리가 다룰 이야기

- 사용자 사전을 사람이 만들지 말고, 데이터 기반으로 추출할 것입니다
- 보강된 사전으로 품사 판별을 수행합니다

우리가 다룰 이야기

문장: 아이오아이는이번공연에서좋은것모습을보였습니다이빠이빠

단어 추출을 통한
미등록 단어 인식

단어: [아이오아이]는이번공연에서좋은것모습을보였습니다이빠이빠

품사 추정을 통한
품사 사전 업데이트

명사 사전 += [아이오아이, ...]

동사 사전 += [잘했어용, ...]

...

품사 사전을 이용한
품사 판별

품사열: [(아이오아이, 명사), (는, 조사), (이번, 명사), (공연, 명사),
(에서, 조사), (좋은, 형용사), (것, 명사), (모습, 명사), (을, 조사),
(보였습니다, 동사), (이빠, 형용사), (이빠, 형용사)]

후처리

우리가 다룰 이야기

문장: 아이오아이는이번공연에서좋은것모습을보였습니다이빠이빠

단어 추출을 통한
미등록 단어 인식

단어: [아이오아이]는이번공연에서좋은것모습을보였습니다이빠이빠

품사 추정을 통한
품사 사전 업데이트

명사 사전 += [아이오아이, ...]

동사 사전 += [잘했어용, ...]

...

품사 사전을 이용한
품사 판별

품사열: [(아이오아이, 명사), (는, 조사), (이번, 명사), (공연, 명사),
(에서, 조사), (좋은, 형용사), (것, 명사), (모습, 명사), (을, 조사),
(보였습니다, 동사), (이빠, 형용사), (이빠, 형용사)]

후처리

단어 추출

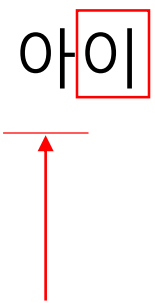
단어 추출

- 다양한 비지도학습 기반의 단어 추출 방법이 제안되었습니다
 - 그 중 character n-gram을 이용하여 단어를 추출하는 방법을 이용합니다

Cohesion (Character n-gram)

- 맥락이 충분히 주어지지 않으면 다음에 등장할 글자의 확률이 작습니다
 - 한글자 ('아')는 매우 모호한 문맥입니다

아이



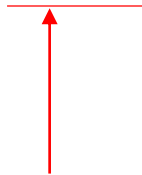
한글자는 특별한 문맥을
가지기가 어렵습니다

- > 아니 17.15 %
- > **아이 14.86 %**
- > 아시 8.06 %
- > 아닌 4.74 %
- > 아파 4.43 %
- > 아직 3.85 %
- ...

Cohesion (Character n-gram)

- 맥락이 충분히 주어지지 않으면 다음에 등장할 글자의 확률이 작습니다

아이오



어떤 경우는 두 글자라 하더라도
다양한 맥락에서 등장하기도 합니다

- > 아이폰 16.60 %
- > 아이들 13.37 %
- > 아이디 9.66 %
- > 아이돌 6.77 %
- > 아이뉴 6.77 %
- > **아이오 6.53 %**

...

Cohesion (Character n-gram)

- Subword 다음에 등장할 글자가 쉽게 예상된다면 (확률이 높다면) 아직 단어가 끝나지 않았다는 의미입니다

아이오아

문맥이 명확해 질수록
이전 단어 → 다음 글자 확률이
높아집니다

> **아이오아 87.95 %**

> 아이오닉 7.49 %

> 아이오와 3.26 %

> 아이오빈 0.65 %

> 아이오페 0.33 %

> 아이오케 0.33 %

Cohesion (Character n-gram)

- Subword 다음에 등장할 글자가 쉽게 예상된다면 (확률이 높다면) 아직 단어가 끝나지 않았다는 의미입니다

> 아이오아이 100.00 %

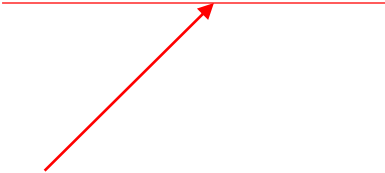
아이오아이

문맥이 확실하면 다음글자의
등장 확률이 높습니다

Cohesion (Character n-gram)

- 단어의 경계를 넘으면 다음 글자에 대한 확률이 다시 작아집니다

아이오아이는



단어 경계 뒤에는 다양한
조사/어미 들이 등장합니다

- > 아이오아이의 31.97 %
- > **아이오아이는 27.21 %**
- > 아이오아이와 13.61 %
- > 아이오아이가 12.24 %
- > 아이오아이에 9.52 %
- > 아이오아이까 1.36 %
- ...

Cohesion (Character n-gram)

- 단어의 점수(cohesion)를 아래처럼 정의해 봅니다

$$cohesion(c_{1:n}) = \sqrt[n-1]{\prod_{i=1}^{n-1} P(c_{1:i+1} | c_{1:i})}$$

$$P(c_{1:2} | c_1) = \frac{\#c_{1:2}}{\#c_1}$$

$$\begin{aligned} cohesion('아이오아이') = & \{ p(\text{아} \rightarrow \text{아이}) * \\ & p(\text{아이} \rightarrow \text{아이오}) * \\ & p(\text{아이오} \rightarrow \text{아이오아}) * \\ & p(\text{아이오아} \rightarrow \text{아이오아이}) \\ & \}^{1/(5-1)} \end{aligned}$$

학습은 오로지
string count

Cohesion (Character n-gram)

- 이 단어 점수는 이후 품사 판별기에서 이용됩니다 (2016-10-20 뉴스 예시)

subword	frequency	$P(AB \mid A)$	Cohesion score
아이	4,910	0.15	0.15
아이오	307	0.06	0.10
아이오아	270	0.88	0.20
아이오아이	270	1.00	0.30
아이오아이는	40	0.15	0.26

품사 추정

우리가 다룰 이야기

문장: 아이오아이는이번공연에서좋은것모습을보였습니다이빠이빠

단어 추출을 통한
미등록 단어 인식

단어: [아이오아이]는이번공연에서좋은것모습을보였습니다이빠이빠

품사 추정을 통한
품사 사전 업데이트

명사 사전 += [아이오아이, ...]

동사 사전 += [잘했어용, ...]

...

품사 사전을 이용한
품사 판별

품사열: [(아이오아이, 명사), (는, 조사), (이번, 명사), (공연, 명사),
(에서, 조사), (좋은, 형용사), (것, 명사), (모습, 명사), (을, 조사),
(보였습니다, 동사), (이빠, 형용사), (이빠, 형용사)]

후처리

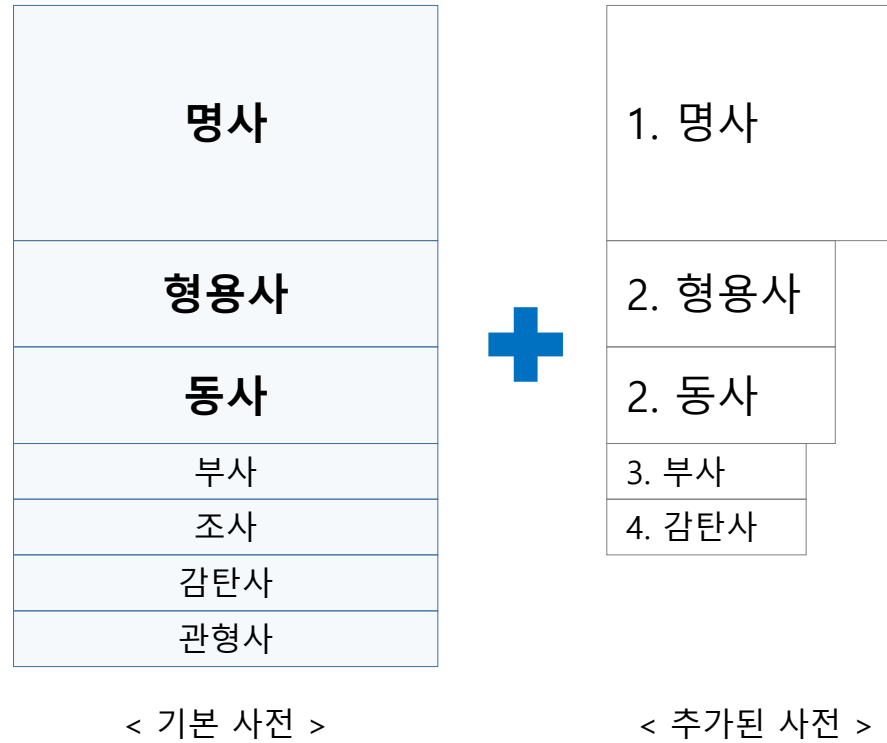
품사 추정

- 새롭게 만들어지는 단어(정확히는 형태소)는 **명사**와 **어미**입니다
 - 명사는 **새로운 개념**을 표현하기 위해서, (eg: 아이오아이)
 - 어미는 다양한 말투를 위해서 만들어집니다 (eg: 하지**말라**궁)
 - 어미에 의하여 새로운 동사/형용사가 만들어집니다

품사 추정

- 새롭게 만들어지는 단어(정확히는 형태소)는 **명사**와 **어미**입니다
- 새로운 형용사/동사의 어근은 “명사 + 이다/되다/하다”의 결합이 많습니다
(eg: 덕질/명사 + 하다/동사)

- 다음의 순서대로 품사 판별에 이용할 사전을 보완합니다



명사 추출

A는 명사일까?

어제 A라는 가게에 가봤어

A에서 보자

A로 와줘

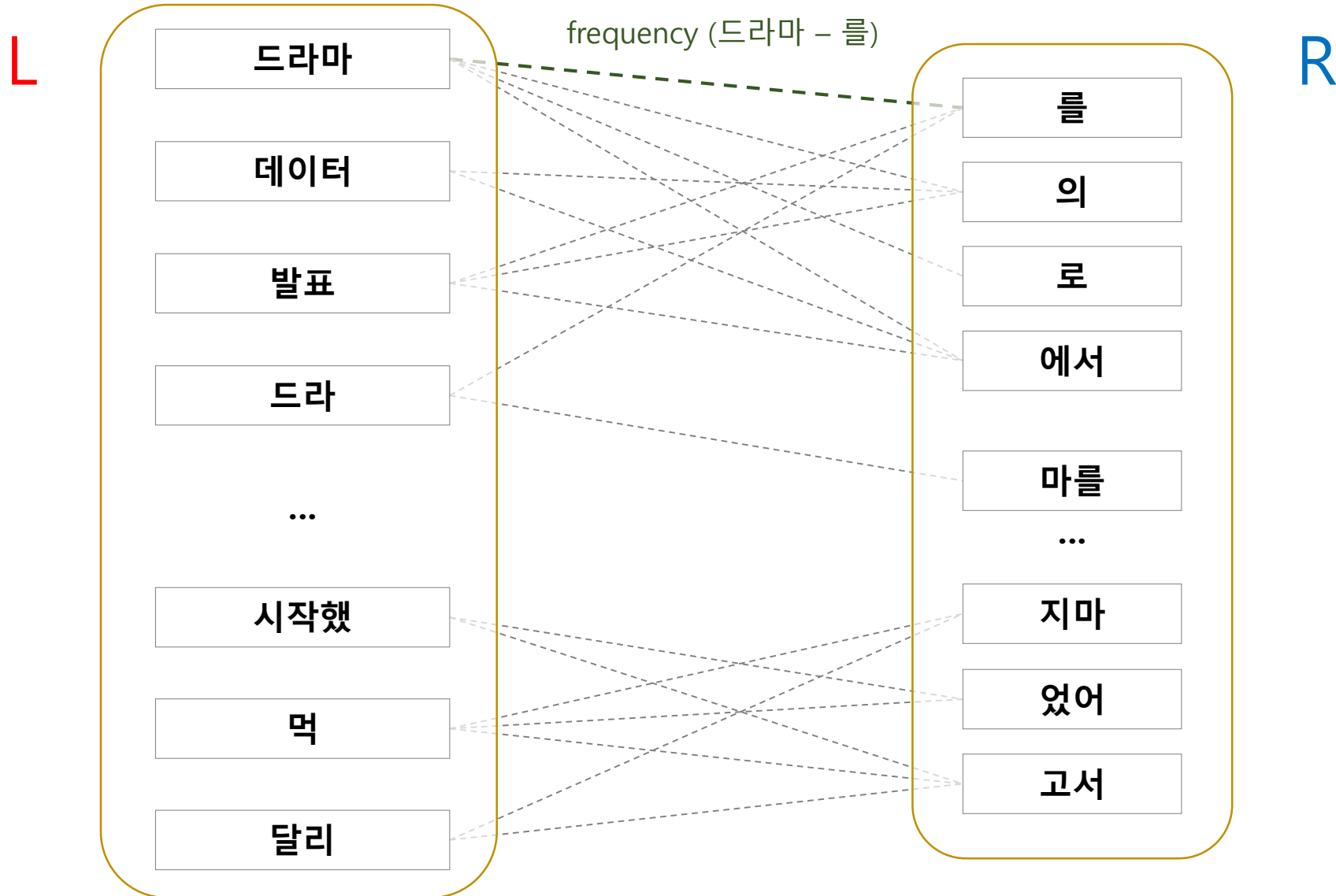
명사 우측에 등장하는 글자 분포를 이용하여

A가 명사임을 유추할 수 있습니다

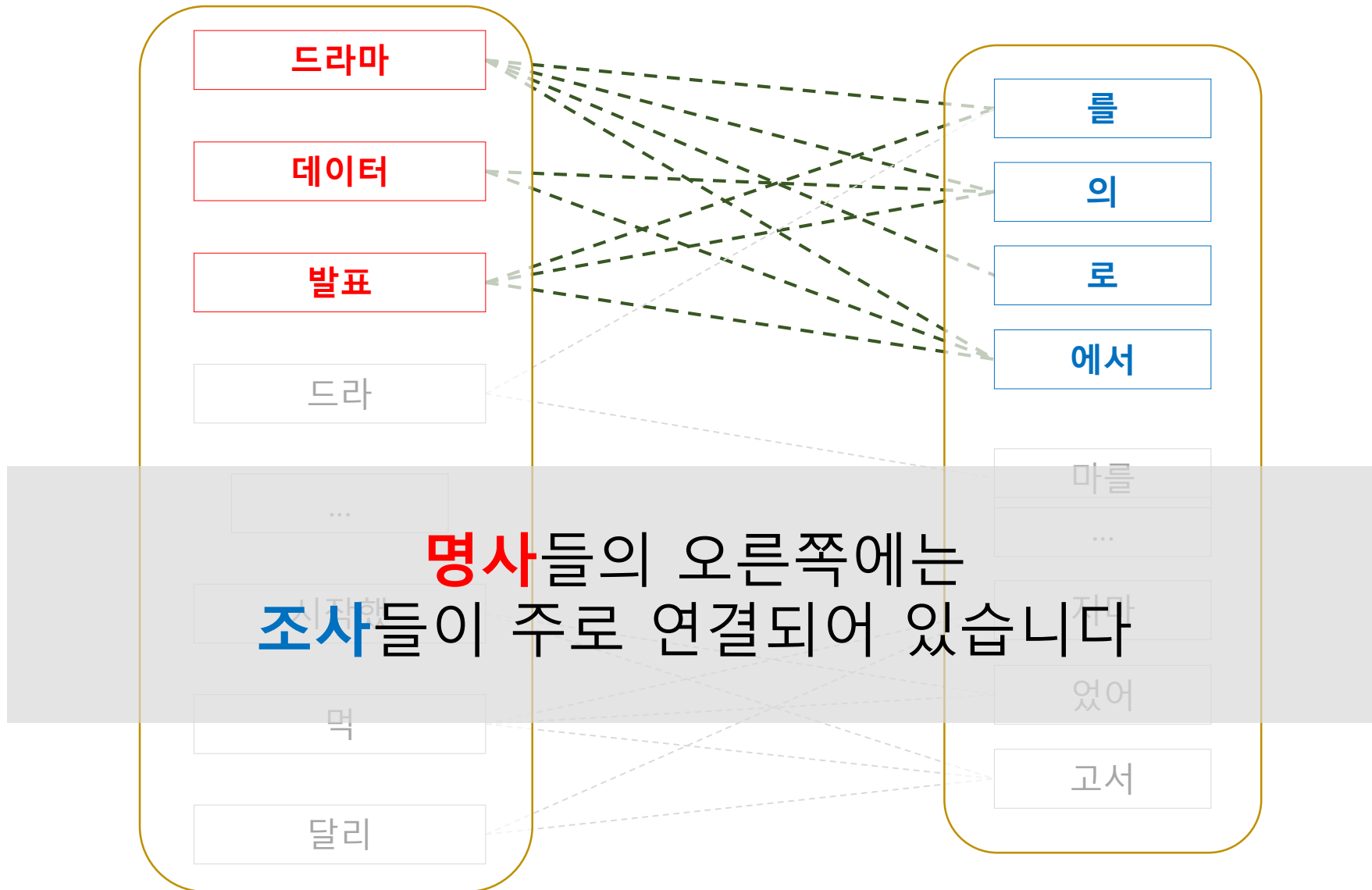
L – R graph

- 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

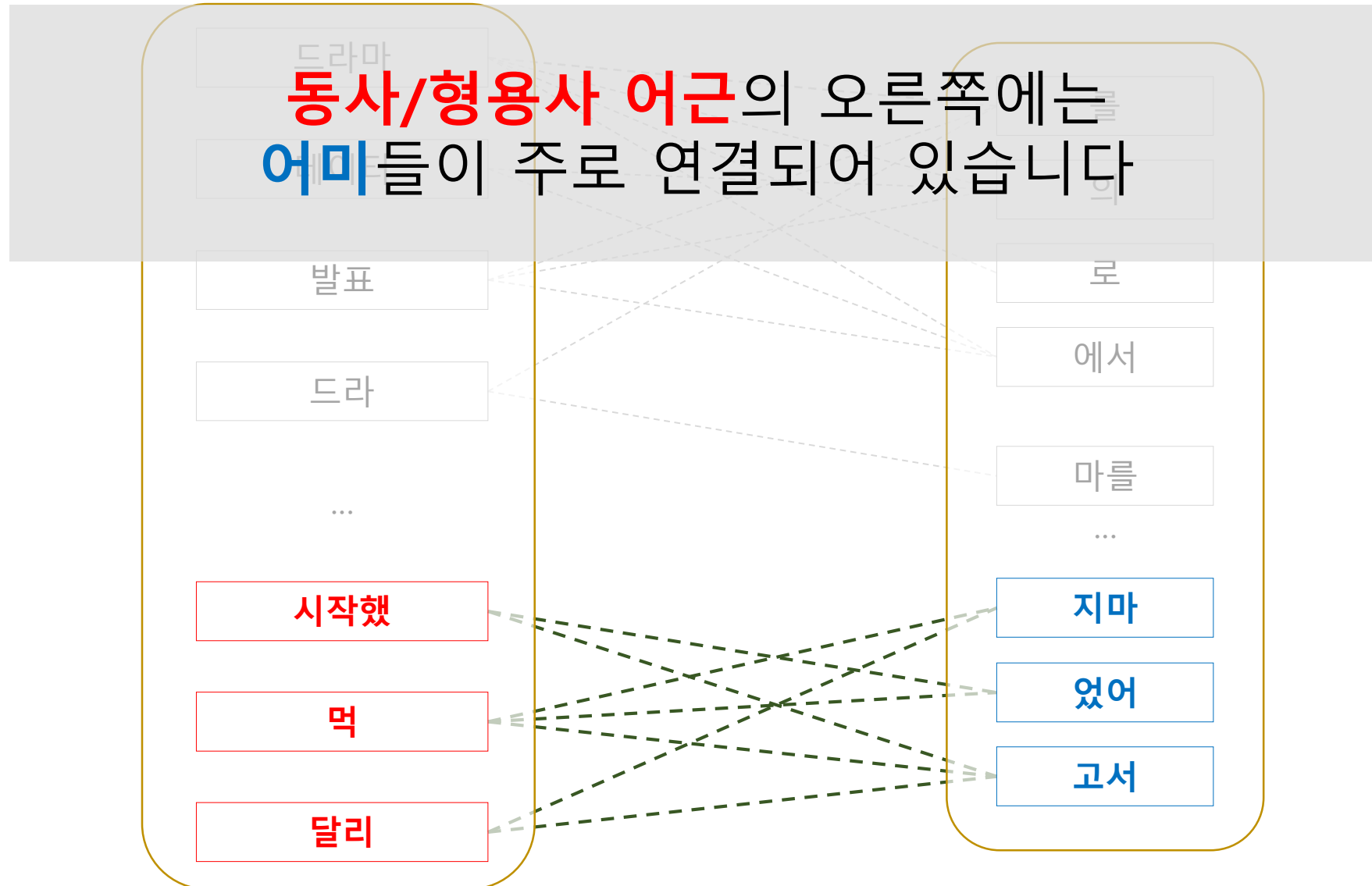
L - R graph



L – R graph

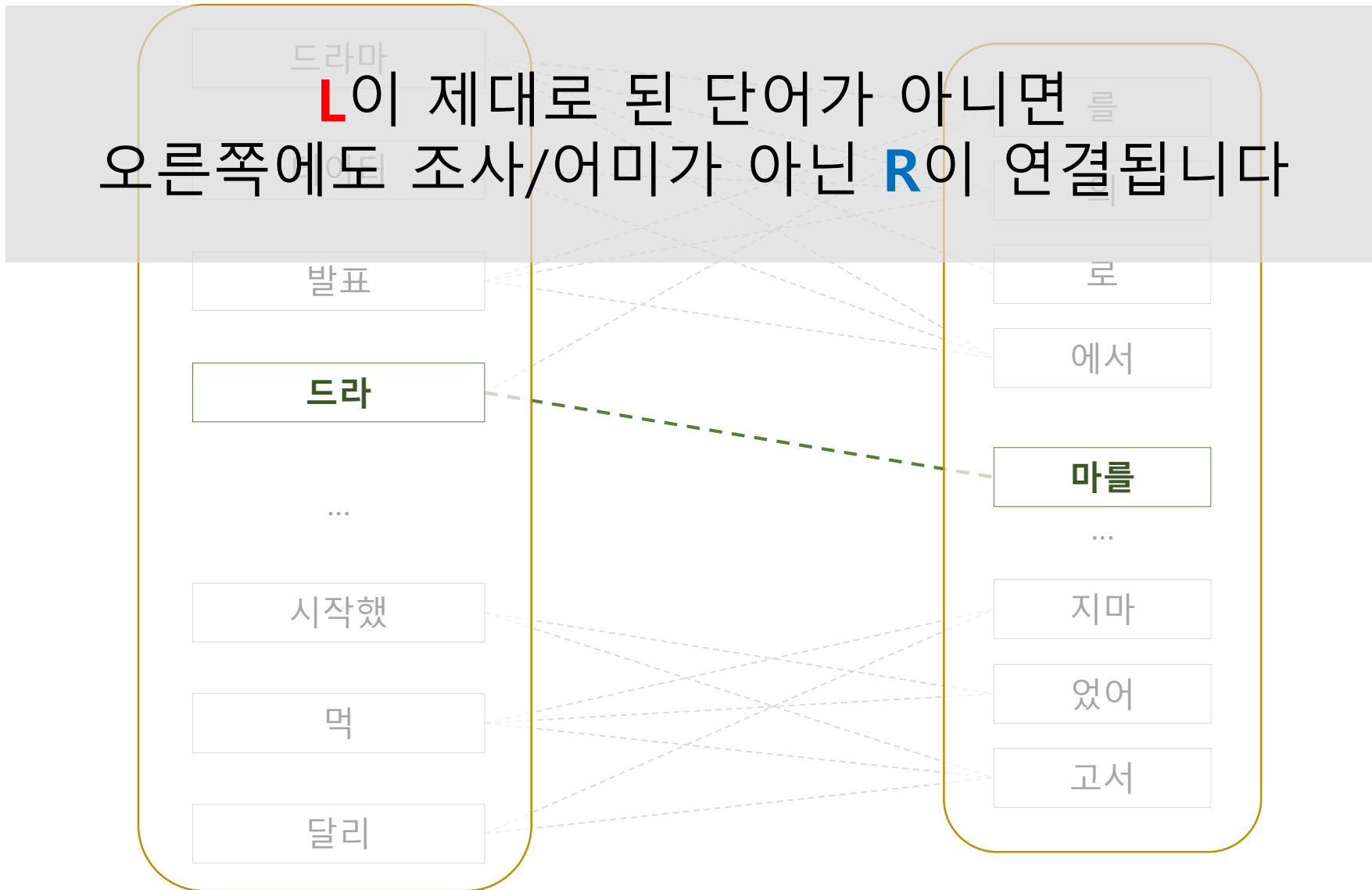


L – R graph



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),  
 ... ]
```

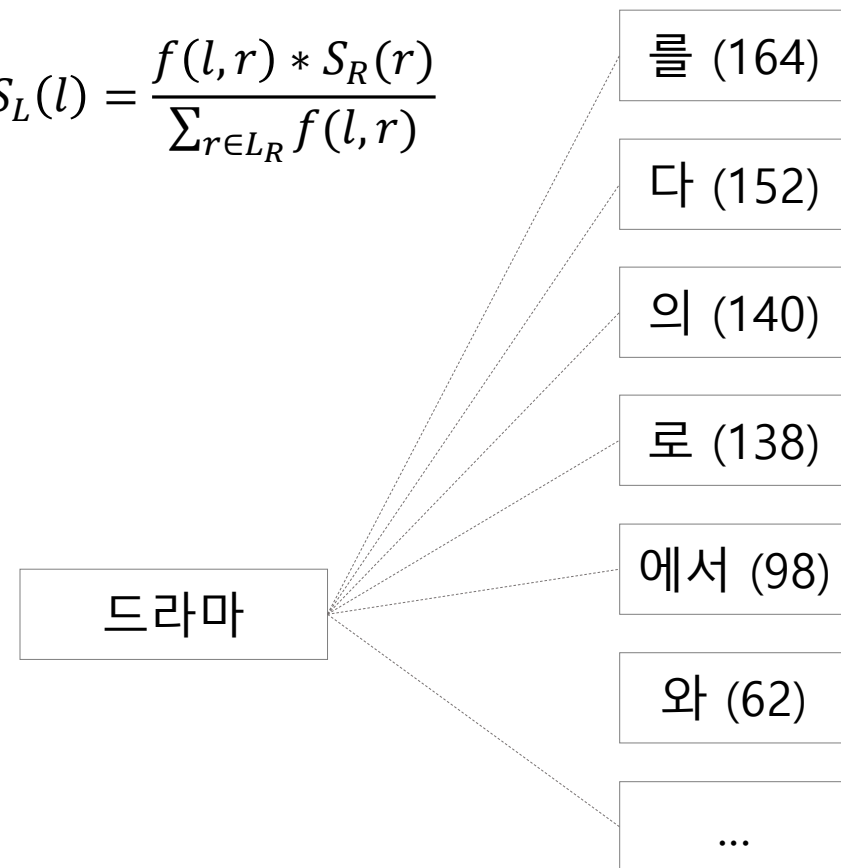
Noun Extraction

- 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: 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

- $N = N_{sub} + J$
 - 떡볶이 + 이
 - '-이'는 대표적 조사이며, '떡볶이' 자체로 어절로 이용기도 하여 '떡볶' 이 명사로 잘못 판단될 수 있습니다
- $N + J_{sub}$
 - 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

soynlp

- 후처리과정이 추가된 명사 추출기를 이용중입니다

```
from soynlp.noun import NewsNounExtractor
```

```
noun_extractor = NewsNounExtractor(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)
```

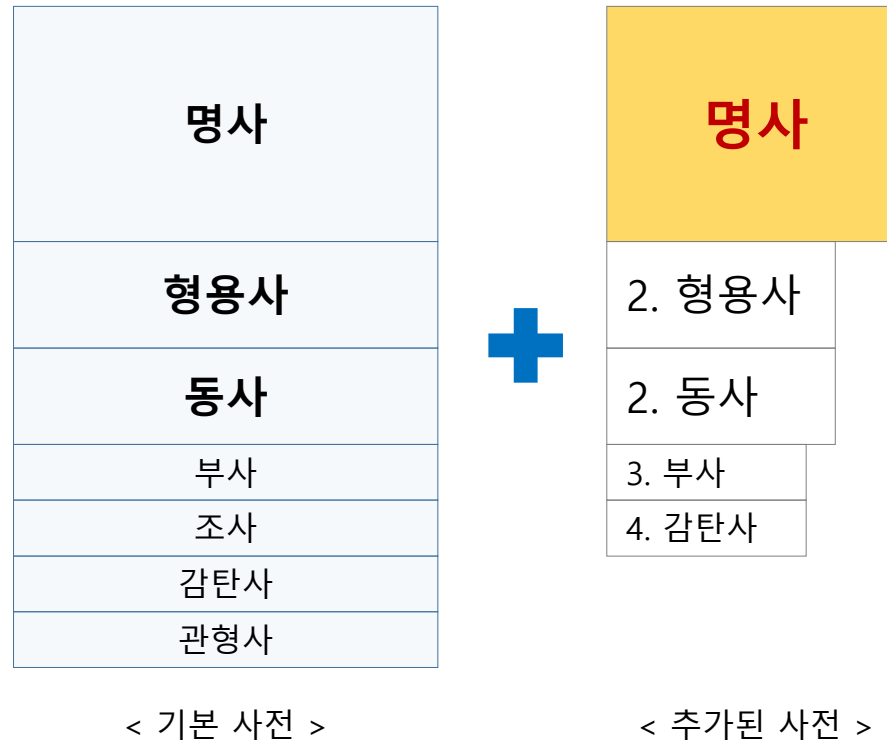
- 깔끔한 뉴스 데이터에서는 명사가 잘 추출됩니다 (2016-10-20)

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

- 댓글에서도 (좀 틀리지만) **관찰은 성능이 보입니다** (2016-10-20, 아이오아이 관련 뉴스의 댓글)

진짜	너무	트와이스	빅뱅	노래	엑소
정말	방탄	좋아	사랑	대박	생각
1위	유정	이번	많이	은근	장근석
예뻐	거짓	타이틀	가수들	티저	사고
운동	캐릭	덕분	다음주	어그로	학대
기획사	양현석	훈훈	휘파람	아미들	지지
기업	것들	만큼	성적	다이어트	관리
역대	시즌제	담주	신기	의미	직접
하차	우아하게	이간질	완전체	신선	연기력
놈들	성격	여전	모모	보상	ㅎㄷㄷ

- 데이터에 등장하는 **명사를 사전에 추가**하였습니다



용언 추출

단어 추출

- 한국어의 어미는 **어절 오른쪽**에 있습니다

먹/동사어근 + **었어**/어미

용언 추출

- 새롭게 만들어지는 단어(정확히는 형태소)는 명사와 **어미**입니다
 - 어미는 **다양한 말투**를 위해서 만들어집니다 (eg: 하지**말라**궁)
 - 어미에 의하여 새로운 동사/형용사가 만들어집니다
- 새로운 형용사/동사의 어근은 “명사 + 이다/되다/하다”의 결합이 많습니다
(eg: 덕질/명사 + 하다/동사)

용언 추출

- “명사 + 조사” 어절의 다양성은 새로운 명사 때문입니다
 - 조사는 잘 변하지 않기 때문에, 명사 추출의 feature가 될 수 있습니다
- “어근 + 어미”의 용언 어절의 다양성은 어미 때문입니다
 - 어근이 잘 변하지 않는다면, 어근은 어미 추출의 feature가 될 수 있습니다

용언 추출

- 명사와 용언이 결합된 어절에서 용언을 분리합니다
- 추출된 명사를 바탕으로 “**명사 + 용언**” 형태의 어절을 분리합니다
 - 덕질이지롱 → [**덕질/명사** + 이지롱]

용언 추출: 어미 추출

- 분리된 용언은 “**알려진 어근** + **새로운 어미**”로 구성되어 있습니다
- 명사가 분리되어 알려진 어근으로 용언 L – R graph를 만들 수 있습니다
 - **이** + [-구나, **-지롱**, **-에용**, **-구낭**, ...]
 - 덕질이 + [-구나, **-지롱**, **-에용**, **-구낭**, ...]

용언 추출: 어미 추출

- L – R graph: [형용사/동사 L + 어미]

get_r('아프')

[('면', 3357),
('지', 3068),
('고', 2389),
('다', 2360),
('다고', 1181),
('게', 947),
('네', 813),
('지마', 733),
('니까', 722),
('진', 423),
('겠다', 371),
('지말고', 348),
...]

get_r('갔')

[('어', 6450),
('다가', 6009),
('다', 4401),
('는데', 2446),
('어요', 1737),
('네', 1243),
('지', 892),
('으면', 882),
('나', 837),
('던', 750),
('을', 726),
('다와', 636),
...]

get_r('해')

[('서', 30932),
('요', 17947),
('도', 17383),
('야지', 13620),
('야', 9627),
('줘', 4259),
('주고', 2868),
('용', 2722),
('라', 2394),
('야겠다', 2341),
('줄게', 2266),
('봐', 2202),
...]

용언 추출: 어미 추출

- L – R graph: [형용사/동사 L + 세종말뭉치에 없는 어미]

get_r('아프')

[('지마', 733),
('진', 423),
('당', 222),
('긴', 187),
~~('리카', 187),~~
('지마요', 135),
('자나', 121),
('넝', 104),
('지말구', 77),
('기만', 71),
('지마라', 63),
('징', 59),
...]

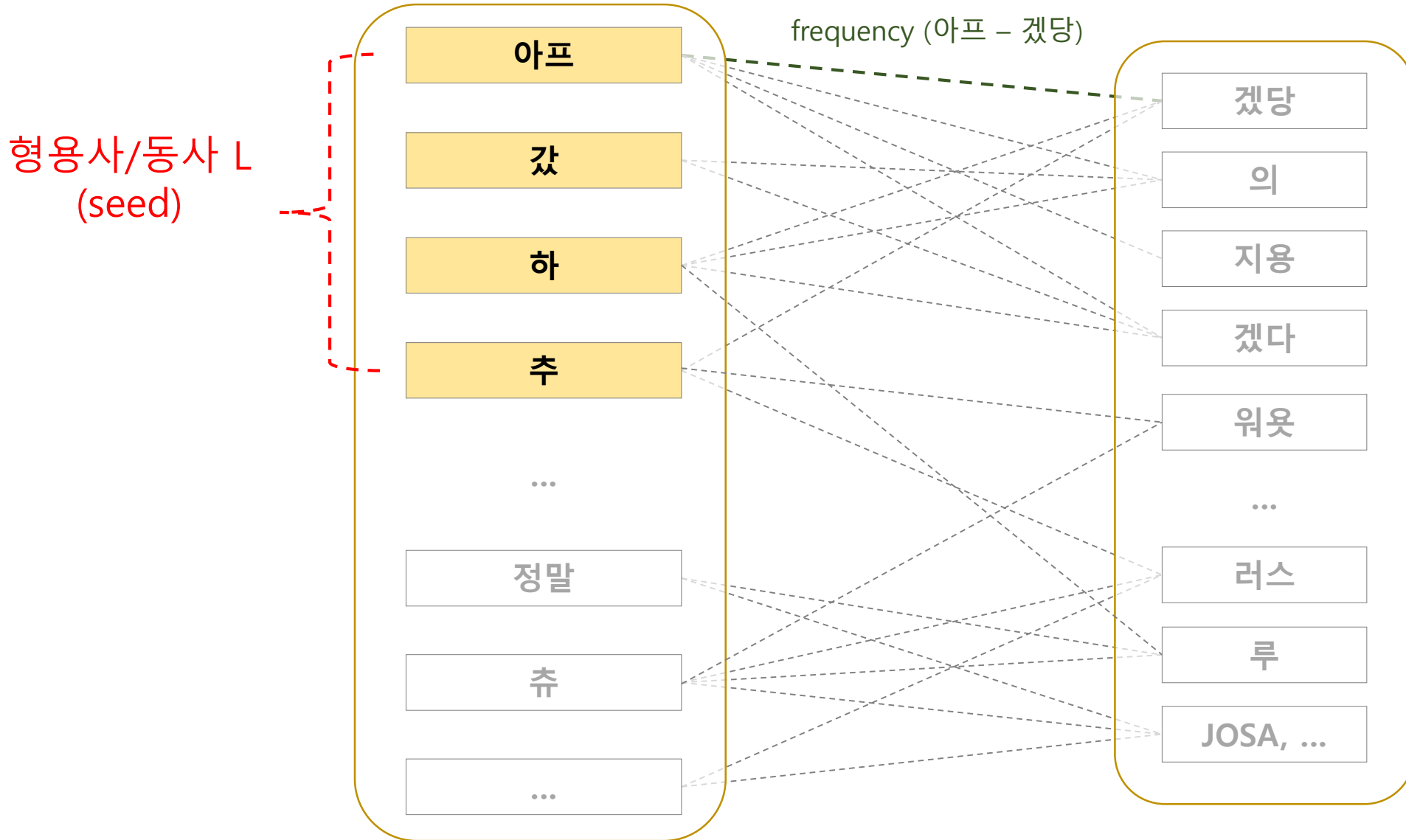
get_r('갓')

[('다와서', 612),
('엉', 466),
('다왔어', 414),
('다오면', 297),
('다올게', 288),
('다와요', 221),
('넝', 218),
('어용', 193),
('다오고', 167),
('나봐', 129),
('다올까', 108),
('다왔는데', 105),
...]

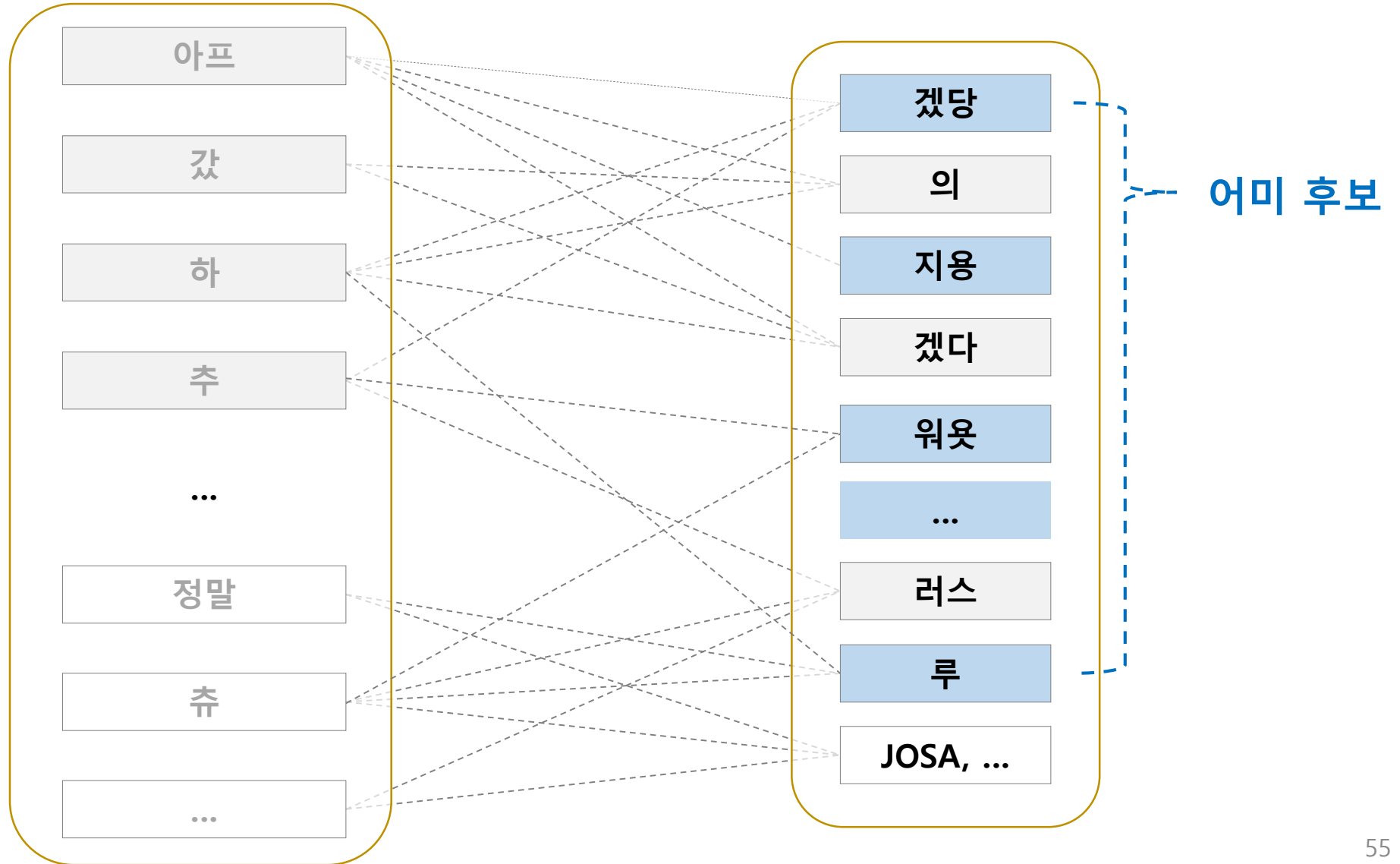
get_r('해')

[('줘', 4259),
('주고', 2868),
('용', 2722),
('야겠다', 2341),
('줄게', 2266),
('봐', 2202),
('놓고', 1582),
('야해', 1458),
('야징', 1446),
('써', 1334),
('주면', 1230),
('달라고', 1206),
...]

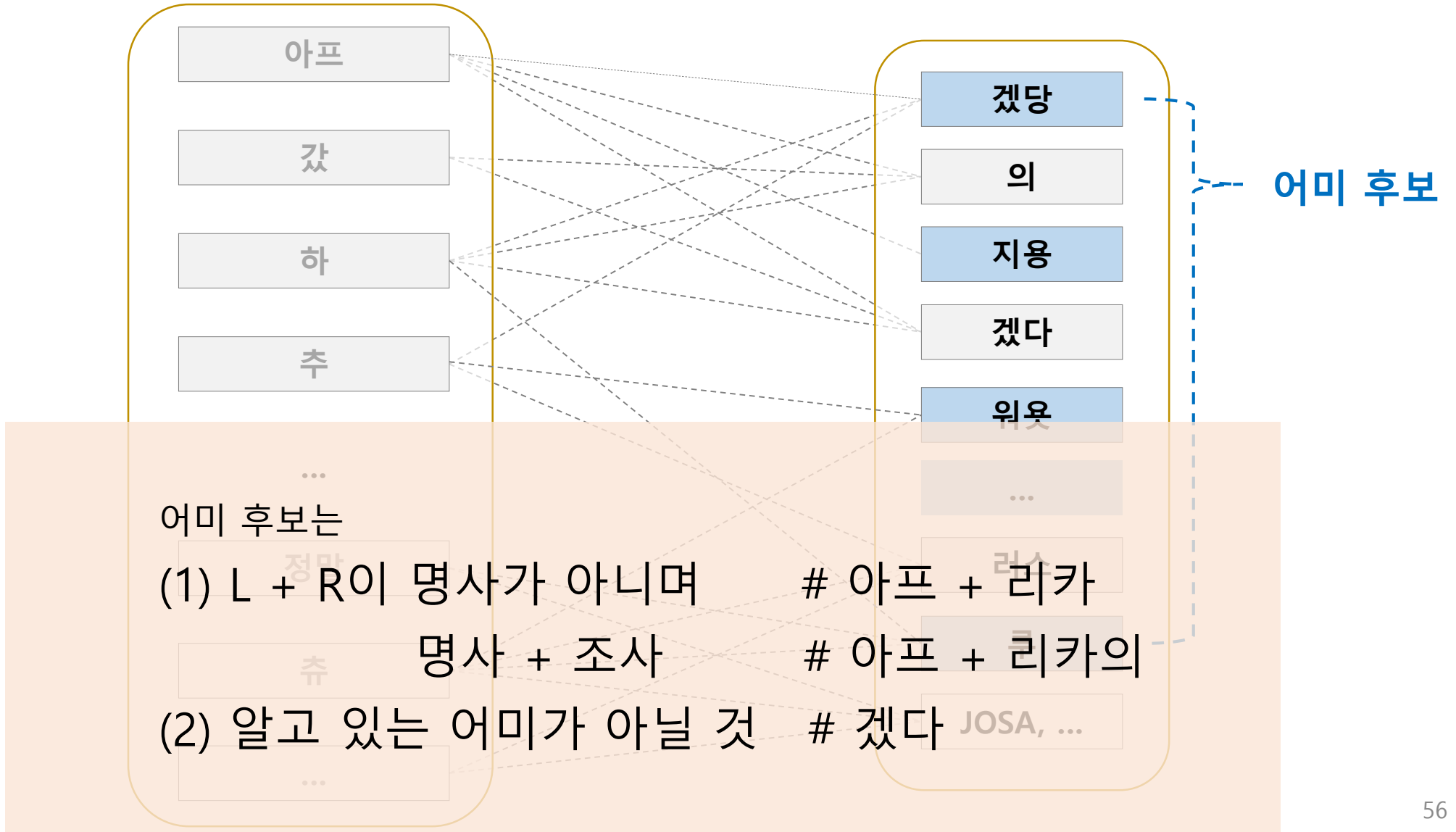
용언 추출: 어미 추출



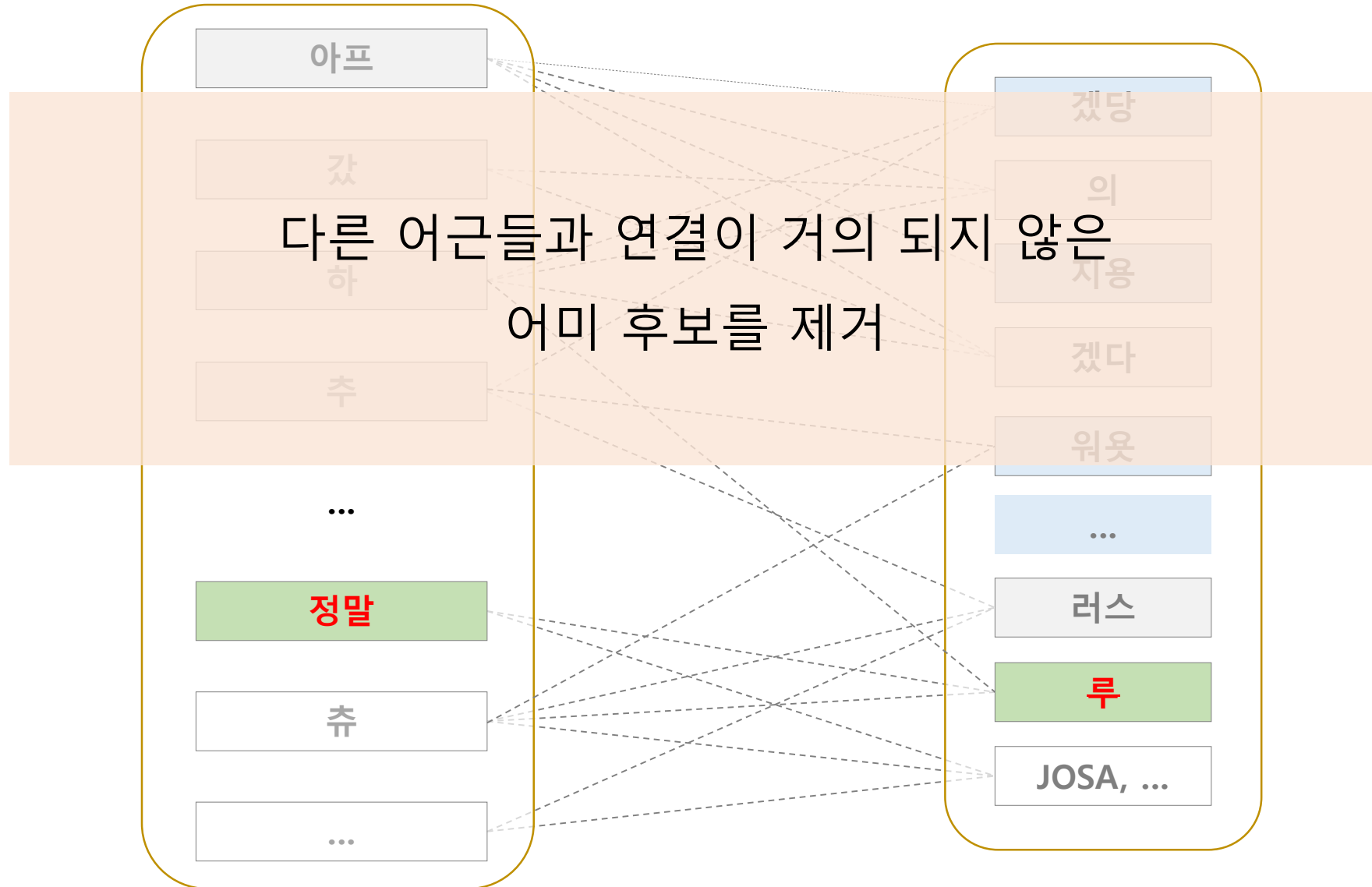
용언 추출: 어미 추출



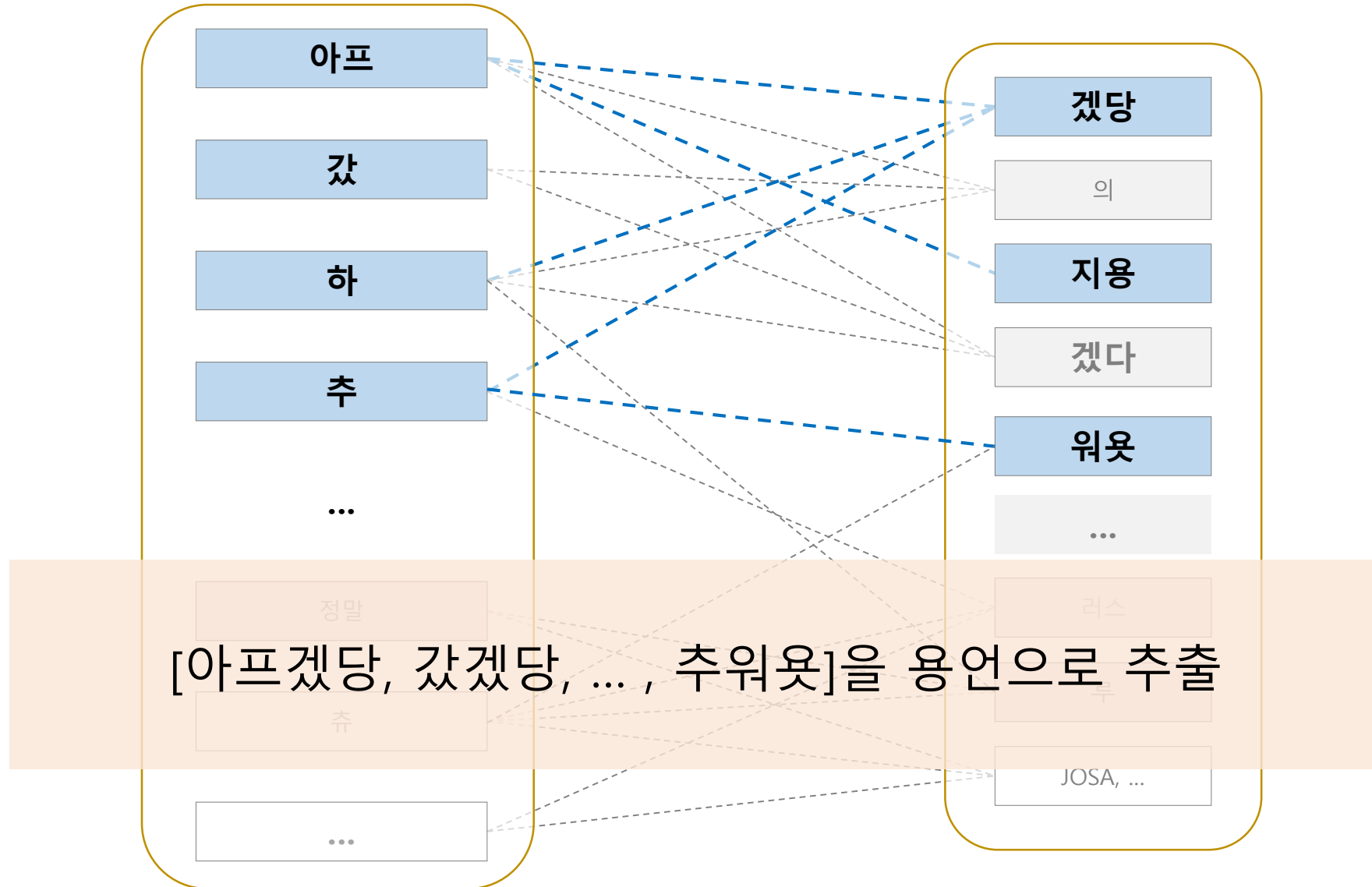
용언 추출: 어미 추출



용언 추출: 어미 추출



용언 추출: 어미 추출



용언 추출

• 추출된 어미와 용언입니다

어근		어미
[안, 사, 올라, 집 , 가져, 다, 내려, ...]	—	왔어
[잘자, 그래, 해, 아니, 와, 먹어, 고마워, ...]	—	요오
[어 , 나와, 몰, 와, 먹어, 해, 자, ...]	—	랏
[되는, 한, 하는, 있는, 사전 , 가는, 잘, ...]	—	거가
[올, 있을, 줄, 그러, 씻을, 해줄, 그럴, ...]	—	게여
[가, 해, 깨, 먹어, 잊어, 와, 나와, ...]	—	버렸어
[갈, 하, 나가, 잘, 먹을, 보, 씻으, ...]	—	려궁
[올, 틀, 그, 밀, 달, 버, 내, ...]	—	린
[온, 없, 먹는, 잔, 싫, 좋, 아프, ...]	—	다매

- 데이터에 등장하는 **형용사/동사를 사전에 추가**하였습니다



품사 판별

우리가 다룰 이야기

문장: 아이오아이는이번공연에서좋은것모습을보였습니다이빠이빠

단어 추출을 통한
미등록 단어 인식

단어: [아이오아이]는이번공연에서좋은것모습을보였습니다이빠이빠

품사 추정을 통한
품사 사전 업데이트

명사 사전 += [아이오아이, ...]

동사 사전 += [잘했어용, ...]

...

품사 사전을 이용한
품사 판별

품사열: [(아이오아이, 명사), (는, 조사), (이번, 명사), (공연, 명사),
(에서, 조사), (좋은, 형용사), (것, 명사), (모습, 명사), (을, 조사),
(보였습니다, 동사), (이빠, 형용사), (이빠, 형용사)]

후처리

품사 판별

- 확장된 품사 사전을 이용하여 품사 판별을 수행합니다



품사 판별

- 품사 판별을 위해 Finite State Model 방법으로 순차적으로 후보를 만들 수도 있습니다.

읽은 input
"아이오 아이는 이번 공연에서 좋은 것 모습을 보였습니다 아이빠이빠"

후보 1: "아/명사 + 이/조사 + 오/명사" : score -0.53
후보 2: "아이/명사 + 오/명사" : score -0.27
후보 3: "아이오/명사" : score -0.11

k - beam

품사 판별

- 알고 있는 단어부터 품사 판별을 수행하도록 하였습니다
 - 긴 문장이 주어지면 사람은 아는 단어부터 눈에 보입니다
 - 확신이 있는 단어부터 품사를 판별합니다

품사 판별

Step 1: 어절의 “명사/형용사/동사/부사”를 사전과 매칭 합니다

- 단어가 겹치더라도 가능한 모든 후보를 만듭니다

```
sent = '아이오아이는이번공연에서좋은강모습을보였습니다이빠이빠'
```

```
candidates = _initialize_L(sent)
```

```
[['아이', 'Noun', 0, 2],
```

```
['아이오', 'Noun', 0, 3],
```

```
['아이오아이', 'Noun', 0, 5],
```

```
['이오', 'Noun', 1, 3],
```

```
['아이', 'Noun', 3, 5],
```

```
['이는', 'Verb', 4, 6],
```

```
['이번', 'Noun', 6, 8],
```

```
['공연', 'Noun', 8, 10],
```

```
...]
```

[단어, 품사, 시작 index, 종료 index]

품사 판별

Step 2: 포함 관계에 있는 **같은 품사의 단어**중, **가장 긴 것**만 남깁니다

```
sent = '아이오아이는이번공연에서좋은강모습을보였습니다이빠이빠'  
candidates = _remove_1_subsets(candidates)
```

```
[ ['아이아', 'Noun', 0, 2],  
  ['아이아오', 'Noun', 0, 3],  
  ['아이오아이', 'Noun', 0, 5],  
  ['아오', 'Noun', 1, 3],  
  ['아아', 'Noun', 3, 5],  
  ['이는', 'Verb', 4, 6],  
  ['이번', 'Noun', 6, 8],  
  ['공연', 'Noun', 8, 10],  
  ... ]
```

← ‘아이오아이’에 포함되는 모든 명사는 제거합니다

← “이는/Verb”는 ‘아이오아이’와 다른 품사이며, 포함되지 않습니다

품사 판별

Step 3: “조사/형용사/동사”를 확장합니다

- 조사, 어미보다 명사/형용사/동사/부사를 잘 인식하는 것이 중요합니다

```
sent = '아이오아이는이번공연에서좋은캠모습을보였습니다이빠이빠'  
candidates = _initialize_LR(sent, candidates)
```

```
[(['아이오아이', 'Noun'], ('', ''), 0, 5, 5],  
  ([('아이오아이', 'Noun'), ('는', 'Josa')], 0, 5, 6],  
  ([('이는', 'Verb'), ('', '')], 4, 6, 6],  
  ([('이번', 'Noun'), ('', '')], 6, 8, 8],  
  ([('공연', 'Noun'), ('', '')], 8, 10, 10],  
  ([('공연', 'Noun'), ('에', 'Josa')], 8, 10, 11],  
  ([('공연', 'Noun'), ('에서', 'Josa')], 8, 10, 12],  
  ...  
]
```

품사 판별

Step 4: 확장된 단어 중 같은 품사는 가장 긴 것만 남깁니다

```
sent = '아이오아이는이번공연에서좋은캠모습을보였습니다이빠이빠'
```

```
candidates = _remove_r_subsets(candidates)
```

```
[(['아이오아이', 'Noun'], ('', ''), 0, 5, 5],  
 (['아이오아이', 'Noun'], ('는', 'Josa'), 0, 5, 6],  
 (['이는', 'Verb'], ('', ''), 4, 6, 6],  
 (['이번', 'Noun'], ('', ''), 6, 8, 8],  
 (['공연', 'Noun'], ('', ''), 8, 10, 10],  
  (['공연', 'Noun'], ('에', 'Josa'), 8, 10, 11],  
  (['공연', 'Noun'], ('에서', 'Josa'), 8, 10, 12],  
  ...  
]
```

품사 판별

Step 4: 최종 후보입니다

```
sent = '아이오아이는이번공연에서좋은모습을보였습니다이빠이빠'  
candidates = _initialize(sent)
```

```
[('아이오아이', 'Noun'), ('는', 'Josa'), 0, 6, 6],  
[('아이오아이', 'Noun'), ("", ""), 0, 5, 5],  
[('이는', 'Verb'), ("", ""), 4, 6, 2],  
[('이번', 'Noun'), ("", ""), 6, 8, 2],  
[('공연', 'Noun'), ('에서', 'Josa'), 8, 12, 4],  
[('공연', 'Noun'), ("", ""), 8, 10, 2],  
[('좋은', 'Adjective'), ("", ""), 12, 14, 2],  
[('모습', 'Noun'), ('을', 'Josa'), 15, 18, 3],  
[('모습', 'Noun'), ("", ""), 15, 17, 2],  
[('보였습니다', 'Verb'), ("", ""), 18, 23, 5],  
[('다이', 'Noun'), ("", ""), 22, 24, 2],  
...]
```

품사 판별

Step 5: "L + R"을 고려하여 **scoring**을 합니다

```
sent = '아이오아이는이번공연에서좋은강모습을보였습니다이빠이빠'  
scores = _scoring(candidates)
```

```
[[('아이오아이', 'Noun'), ('는', 'Josa'), 0, 6, 6, 3.39],  
 [('아이오아이', 'Noun'), ("", ""), 0, 5, 5, 2.54],  
 [('이는', 'Verb'), ("", ""), 4, 6, 2, 1.90],  
 [('이번', 'Noun'), ("", ""), 6, 8, 2, 2.24],  
 [('공연', 'Noun'), ('에서', 'Josa'), 8, 12, 4, 2.77],  
 [('공연', 'Noun'), ("", ""), 8, 10, 2, 1.73],  
 [('좋은', 'Adjective'), ("", ""), 12, 14, 2, 2.25],  
 [('모습', 'Noun'), ('을', 'Josa'), 15, 18, 3, 3.26],  
 [('모습', 'Noun'), ("", ""), 15, 17, 2, 2.01],  
 [('보였습니다', 'Verb'), ("", ""), 18, 23, 5, 2.21],  
 [('다이', 'Noun'), ("", ""), 22, 24, 2, 1.11],  
 ... ]
```

← '아이오아이/명사 + 는/조사' 점수

품사 판별

Step 5: 점수 계산 feature를 만든 뒤, **weight**를 곱하여 **scoring**을 합니다

```
profile = OrderedDict([  
  
    ('cohesion_1', 0.5),  
    ('P(AB|A)_1', 0.5),  
    ('log_count_1', 0.1),  
  
    ('prob_12r', 0.1),  
    ('log_count_12r', 0.1),  
    ('known_LR', 1.0),  
  
    ('R_is_syllable', -0.1),  
    ('log_length', 0.5)  
])
```

L의 cohesion score	* 0.5
+ L의 P(AB A)	* 0.5
+ L의 log 빈도수	* 0.1
+ 분석텍스트의 $P(L \rightarrow R)$	* 0.1
+ 분석텍스트의 $\text{Freq}(L \rightarrow R)$	* 0.1
+ L과 R이 모두 알려진 품사	* 1.0
+ 1음절 조사/어미	* -0.1
+ “L+R” 길이의 log	* 0.5

품사 판별

Step 6: 높은 점수의 어절부터 품사를 부여 / 겹치는 부분은 제거합니다

```
sent = '아이오아이는이번공연에서좋은강모습을보였습니다이빠이빠'  
words = _find_best(scores)
```

```
[('아이오아이', 'Noun'), ('는', 'Josa'), 0, 6, 6, 3.39],  
[('모습', 'Noun'), ('을', 'Josa'), 15, 18, 3, 3.26],  
[('공연', 'Noun'), ('에서', 'Josa'), 8, 12, 4, 2.77],  
[('아이오아아', 'Noun'), ('', ''), 0, 5, 5, 2.54],  
[('좋은', 'Adjective'), ('', ''), 12, 14, 2, 2.25],  
[('이번', 'Noun'), ('', ''), 6, 8, 2, 2.24],  
[('보였습니다', 'Verb'), ('', ''), 18, 23, 5, 2.21],  
[('모습', 'Noun'), ('', ''), 15, 17, 2, 2.01],  
[('이는', 'Verb'), ('', ''), 4, 6, 2, 1.90],  
[('공연', 'Noun'), ('', ''), 8, 10, 2, 1.73],  
[('다아', 'Noun'), ('', ''), 22, 24, 2, 1.11],  
... ]
```

품사 판별

Step 7: 사전에 등록되지 않은 단어는 아직 인식되지 않았습니다

```
sent = '아이오아이는이번공연에서좋은강모습을보였습니다이빠이빠'
```

```
[('아이오아이', 'Noun'),  
 ('는', 'Josa'),  
 ('이번', 'Noun'),  
 ('공연', 'Noun'),  
 ('에서', 'Josa'),  
 ('좋은', 'Adjective'),  
 ('모습', 'Noun'),  
 ('을', 'Josa'),  
 ('보였습니다', 'Verb'),  
 ('이빠', 'Adjective'),  
 ('이빠', 'Adjective')]
```

품사 판별

Step 7: 사전에 없는 단어로 구성된 sub-sentence를 **후처리** 합니다

```
sent = '아이오아이는이번공연에서좋은강모습을보였습니다이뻐이뻐'
```

```
[('아이오아이', 'Noun'),  
 ('는', 'Josa'),  
 ('이번', 'Noun'),  
 ('공연', 'Noun'),  
 ('에서', 'Josa'),  
 ('좋은', 'Adjective'),  
 ('강', None),  
 ('모습', 'Noun'),  
 ('을', 'Josa'),  
 ('보였습니다', 'Verb'),  
 ('이뻐', 'Adjective'),  
 ('이뻐', 'Adjective')]
```

품사 판별 성능: vs. 트위터 한국어 분석기

twitter.pos(sent)

Process time: 102 ms

```
[('아이오', 'Noun'),  
 ('아이', 'Noun'),  
 ('는', 'Josa'),  
 ('이번', 'Noun'),  
 ('공연', 'Noun'),  
 ('에서', 'Josa'),  
 ('좋', 'Adjective'),  
 ('은', 'Eomi'),  
 ('캉', 'Noun'),  
 ('모습', 'Noun'),  
 ('을', 'Josa'),  
 ('보였', 'Verb'),  
 ('습니다', 'Eomi'),  
 ('이빠', 'Adjective'),  
 ('이빠', 'Adjective')]
```

proposed.pos(sent)

Process time: 3.05 ms

```
[('아이오아이', 'Noun'),  
 ('는', 'Josa'),  
 ('이번', 'Noun'),  
 ('공연', 'Noun'),  
 ('에서', 'Josa'),  
 ('좋은', 'Adjective'),  
 ('캉', None),  
 ('모습', 'Noun'),  
 ('을', 'Josa'),  
 ('보였습니다', 'Verb'),  
 ('이빠', 'Adjective'),  
 ('이빠', 'Adjective')]
```

분석가를 위한 인터페이스

-
- 알고리즘은 예외가 발생하며, 사용자는 예외를 쉽게 수정하고 싶어합니다
 - 사전의 단어 추가 및 삭제
 - 반드시 보존하고 싶은 단어의 손쉬운 보호

사전의 단어 추가 및 삭제

- 컴파일을 다시 하지 않으면서 단어를 추가/삭제 해야 합니다

```
from soynlp.pos import LRMaxScoreTagger  
  
tagger = LRMaxScoreTagger()  
  
tagger.add_words_into_dictionary( ['아이오아이'], 'Noun')  
tagger.remove_words_from_dictionary( ['아이오아이'], 'Noun')
```

반드시 보존하고 싶은 단어의 손쉬운 보호

- 도매인의 키워드, 혹은 중요한 단어들은 선호도를 설정합니다

```
tagger.set_word_preference(['아이오아이', '너무너무너무'], 'Noun', 10)
```

[('아이오아이', 'Noun'), ('는', 'Josa'),	0, 6, 6,	3.39],
[('아이오아이', 'Noun'), ('', ''),	0, 5, 5,	2.54],
[('이는', 'Verb'), ('', ''),	4, 6, 2,	1.90],
...]



[('아이오아이', 'Noun'), ('는', 'Josa'),	0, 6, 6,	13.39],
[('아이오아이', 'Noun'), ('', ''),	0, 5, 5,	12.54],
[('이는', 'Verb'), ('', ''),	4, 6, 2,	1.90],
...]

다른 단어로 인식될 가능성을
원천 봉쇄

score += 10

연관어 분석 예시

Related Word Analysis

- 2016년 10월 20일의 뉴스를 바탕으로 연관어 분석을 수행함으로써 데이터로부터 추출된 단어 사전을 이용하는 효과를 살펴봅니다

Related Word Analysis

- 단어 간의 co-occurrence를 바탕으로 한 연관어 점수를 정의합니다
 - 평상시에 '1위' 라는 단어가 0.1 % 등장하는데,
 - '아이오아이' 가 포함된 문서에서 '1위' 라는 단어가 1 % 등장한다면
 - '1위'는 '아이오아이'가 등장하는 문서에서 평상시보다 10배 더 자주 등장
→ 두 단어는 연관이 있다

Related Word Analysis

- 단어 간의 co-occurrence를 바탕으로 한 연관어 점수를 정의합니다

$$S(w) = \frac{P(w|D_S)}{P(w|D_S) + P(w|\widetilde{D}_S)}$$

$P(w|D_S)$: 기준 단어 S 가 등장한 문서 집합 D_S 에서 단어 w 의 등장 비율

$P(w|\widetilde{D}_S)$: 기준 단어 S 가 등장하지 않은 문서 집합 \widetilde{D}_S 에서 단어 w 의 등장 비율

Related Word Analysis

- 2016-10-20, 하루치 뉴스에서의 연관어 (명사) 분석 결과입니다

Seed: 아이오아이	엠카운트다운	루나	상큼	쇼챔피언	프로듀스
빅브레인	박진영	잠깐	희현	본명	박현민
너무너무너무	백퍼센트	3집	포미닛	아미	유정
레이디스	완전체	몬스	장항	리듬	보컬
엠카	하이포투엔티	앵콜	일산동구	월드투어	김지연
신용재	중독성	대기실	음악방송	깜찍	겉모습
오블리스	정채연	드림센터	사나	방탄소년단	음원차트
하이포	프로듀스101	소녀들	귀요미	걸크러시	선배님들
갓세븐	팀명	온순	선율	몬스타엑스	에이핑크
다비치	꽃길	엠넷	엑스	키미	트로피
세븐	열창	금빛	타이틀곡	챔피언	그대
다이아	펜타곤	걸크러쉬	코드	수록곡	파워풀

마무리

정리

- 사전만 잘 구축된다면 품사 판별을 잘 할 수 있습니다
- 데이터 기반으로 “텍스트에서 실제 사용이 되는 단어”를 품사 사전에 추가할 수 있습니다
- 다양한 도메인의 텍스트 분석에 유연하게 적용/적응될 수 있는 품사 판별기(혹은 품사 판별기 builder)가 될 수 있습니다