

# Corpus based Part-of-Speech Tagger (HMM, CRF)

Hyunjoong Kim

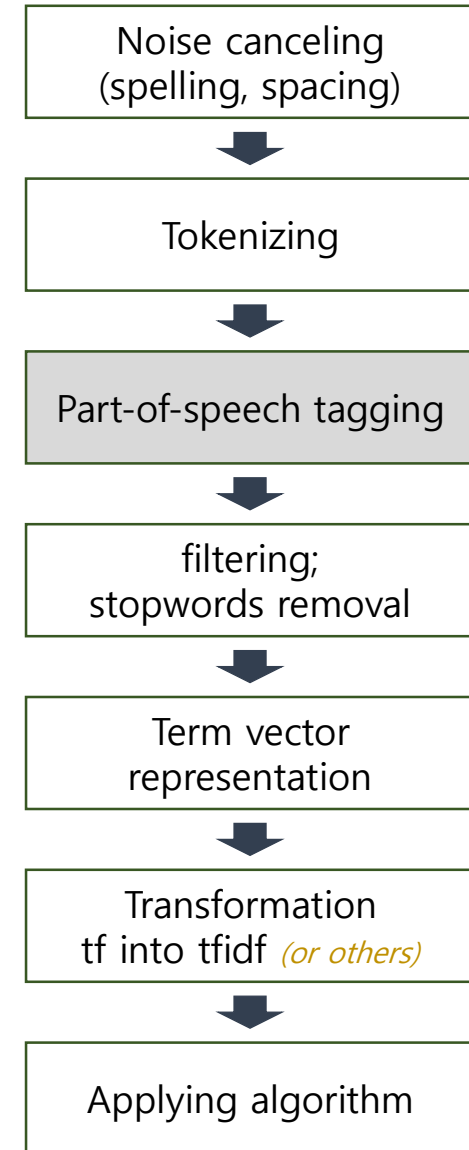
[soy.lovit@gmail.com](mailto:soy.lovit@gmail.com)

[github.com/lovit](https://github.com/lovit)

# Part-of-Speech tagging

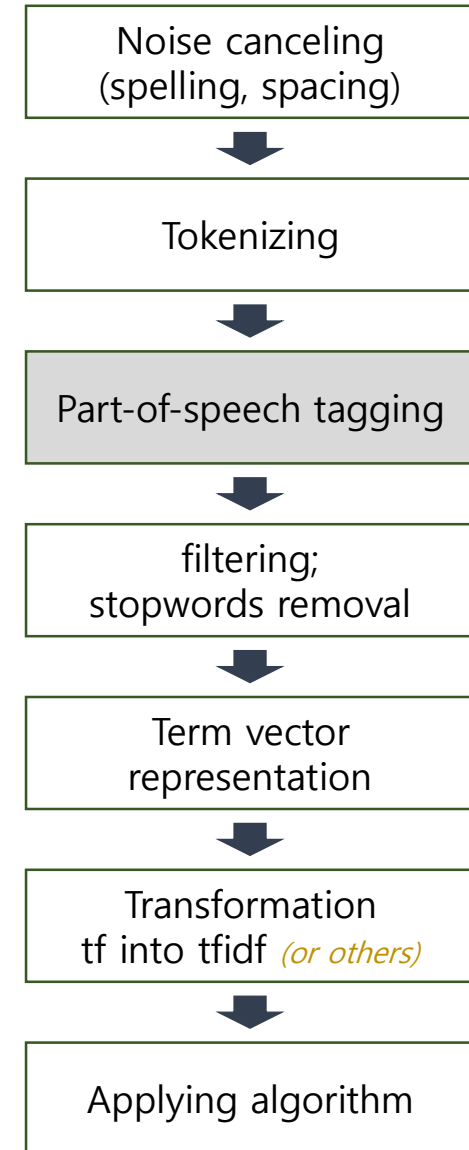
- 품사 판별은 주어진 단어의 품사를 구분합니다

- [토크나이징, 은, 어절, 에서, 단어,를, 나누는, 것, 입니다] →  
[ (토크나이징, 명사),  
(은, 조사),  
(어절, 명사),  
(에서, 조사),  
(단어, 명사),  
(를, 조사),  
(나누는, 동사),  
(것, 명사),  
(입니다, 형용사) ]



# Morphological analysis

- 형태소 분석은 단어의 형태소를 인식합니다.
  - 형태소는 단어를 구성하는 최소단위 입니다.
  - 품사 판별: "입니다" → 형용사
  - 형태소 분석: "입니다"  
→ 이/형용사어근 + ㅂ니다/어미



- 
- 품사 판별과 형태소 분석의 과정은 비슷합니다.
    - 형태소 분석이 품사 판별보다 조금 더 세분화된 단위로 문장의 구성요소를 인식하는 것 뿐입니다.
  - 품사 판별을 위하여 형태소 분석이 이용될 수도 있습니다.
    - 먹는 = '먹다/verb' + 는/eomi' 로 분해하여 '먹는/verb' 로 판단할 수도 있으며,
    - '먹는/verb' 자체가 동사 사전에 포함되어 있어도 됩니다.
    - 품사 판별을 위하여 반드시 형태소 분석을 해야 하는 것은 아닙니다.

- 
- 품사 판별, 형태소 분석의 과정은 두 단계로 구성됩니다.
    - (1) 주어진 문장으로부터 단어/형태소 열의 후보들을 만들고 (generation),
    - (2) 그 중 가장 높은 점수를 얻는 후보를 선택합니다 (evaluation).

- 
- 품사 판별, 형태소 분석의 과정은 두 단계로 구성됩니다.

(1) 주어진 문장으로부터 단어/형태소 열의 후보들을 만들고 (generation),

(2) 그 중 가장 높은 점수를 얻는 후보를 선택합니다 (evaluation).

**문장:** '이것은 예문이다'

**후보 1:** [이것/Noun, 은/Josa, 예문/Noun, 이다/Adj]

**후보 2:** [이것/Noun, 은/Josa, 예문/Noun, 이/Noun, 다/Noun],

**후보 3:** [이/Noun, 것/Noun, 은/Josa, 예문/Noun, 이/Noun, 다/Noun],

...

- 
- 품사 판별, 형태소 분석의 과정은 두 단계로 구성됩니다.

(1) 주어진 문장으로부터 단어/형태소 열의 후보들을 만들고 (generation),

(2) 그 중 가장 높은 점수를 얻는 후보를 선택합니다 (evaluation).

**문장:** '이것은 예문이다'

**후보 1:** [이것/Noun, 은/Josa, 예문/Noun, 이다/Adj]

→ prob = 0.57

**후보 2:** [이것/Noun, 은/Josa, 예문/Noun, 이/Noun, 다/Noun],

→ prob = 0.23

**후보 3:** [이/Noun, 것/Noun, 은/Josa, 예문/Noun, 이/Noun, 다/Noun],

→ prob = 0.02

...

- 
- 품사 판별, 형태소 분석의 과정은 두 단계로 구성됩니다.
  - Evaluation 과정은 sequential labeling 을 이용할 수 있습니다.



# Sequential labeling

---

- 길이가  $n$  인  $x = [x_1, x_2, \dots, x_n]$  에하여 category sequence 인  $y = [y_1, y_2, \dots, y_n]$  을 출력하는 문제를 sequential labeling 이라 합니다.
- Word sequence  $x$  에 대하여 tag sequence  $y$  를 출력하면 품사 판별을 할 수 있습니다.

# Sequential labeling

---

- One hot representation 을 이용하는 sequential labeling 이 있습니다.
  - HMM → MEMM → CRF 순으로 이용되었으며,  
CRF 가 품사 판별에서 가장 좋은 성능을 보였습니다.
- 이들은 확률 모형으로, 이들의 목적은 다음과 같습니다.

$$\operatorname{argmax}_{y_{1:n}} P(y_{1:n} | x_{1:n})$$

<sup>^1</sup> Hidden Markov Model (HMM)

<sup>^2</sup> Maximum Entropy Markov Model (MEMM)

<sup>^3</sup> Conditional Random Field (CRF)

# Sequential labeling

---

- 최근에는 word embedding 정보를 이용하여 품사 판별을 하기 위하여 Recurrent Neural Network (RNN) 을 기반으로 하는 품사 판별 알고리즘도 이용되고 있습니다.

# Brief review of Hidden Markov Model

# Hidden Markov Model

---

- HMM 은  $P(y_{1:n}|x_{1:n})$  을 다음처럼 정의합니다.

- $P(y_{1:n}|x_{1:n}) = \frac{P(x_{1:n} | y_{1:n}) \times P(y_{1:n})}{P(x_{1:n})}$  Bayes rule

- 우리가 풀 문제는  $x_{1:n}$  이 고정이므로  $P(x_{1:n})$  는 모든  $y_{1:n}$  에 대하여 동일합니다. 즉 이를 무시합니다.

- $\operatorname{argmax}_{y_{1:n}} P(y_{1:n}|x_{1:n}) = \operatorname{argmax}_{y_{1:n}} P(x_{1:n} | y_{1:n}) \times P(y_{1:n})$  입니다.

# Hidden Markov Model

---

- HMM 은  $P(y_{1:n}|x_{1:n})$  을 다음처럼 정의합니다.
  - $\operatorname{argmax}_{y_{1:n}} P(y_{1:n}|x_{1:n}) = \operatorname{argmax}_{y_{1:n}} P(x_{1:n} | y_{1:n}) \times P(y_{1:n})$ 
    - $P(x_{1:n} | y_{1:n}) := \prod_i P(x_i | y_i)$
    - $P(y_{1:n}) := \prod_i P(y_i | y_{i-1})$
    - $\rightarrow P(x_{1:n} | y_{1:n}) \times P(y_{1:n}) = \prod_i P(x_i | y_i) P(y_i | y_{i-1})$

# Hidden Markov Model

---

- $P(y_{1:n}|x_{1:n}) = \prod_i P(x_i|y_i)P(y_i|y_{i-1})$  이 되는 것은 **HMM** 의 정의입니다.
- 다른 sequential labeling model 은  $P(y_{1:n}|x_{1:n})$  을 다른 방식으로 정의합니다.

# Hidden Markov Model

---

- HMM 은  $P(\text{품사열} \mid \text{단어열})$  의 확률을 다음처럼 계산합니다.

$$P([\text{Noun}, \text{Josa}, \text{Noun}, \text{Adj}] \mid [\text{이것}, \text{은}, \text{예문}, \text{이다}])$$

$$\begin{aligned} &= P(\text{이것} \mid \text{Noun}) \times P(\text{은} \mid \text{Josa}) \times P(\text{예문} \mid \text{Noun}) \times P(\text{이다} \mid \text{Adj}) \\ &\quad \times P(\text{Noun}) \times P(\text{Josa} \mid \text{Noun}) \times P(\text{Noun} \mid \text{Josa}) \times P(\text{Adj} \mid \text{Noun}) \end{aligned}$$

emission probability

: state ( $y_i$ ) 에서 observation ( $x_i$ ) 가 발생할 확률  
: 품사 "Noun" 에서 "이것"이 등장할 확률

transition probability

: state ( $y_i$ ) 가 변하는 확률  
: "Noun  $\rightarrow$  Josa" 일 확률



# Training

---

- 말뭉치를 이용하여 HMM 을 학습할 때에는 두 종류의 parameter 를 학습합니다. 학습은 counting 입니다.

- emission:  $P(x_i|y_i) = \frac{\#(x_i, y_i)}{\# y_i}, \frac{\#(‘이것’, Noun)}{\# Noun}$

- transition :  $P(y_i|y_{i-1}) = \frac{\#(y_i, y_{i-1})}{\# y_{i-1}}, \frac{\#(Noun \rightarrow Josa)}{\# Noun}$

# Decoding

---

- HMM 에  $x_{1:n}$  이 주어졌을 때 확률이 가장 큰  $y_{1:n}$  를 찾기 위해서 주로 Viterbi style decoding 이 이용됩니다.
  - 이는 shortest path 의 계산 과정으로도 설명할 수 있습니다.

# HMM as Shortest path

---

- HMM 의 목적식에  $-\log$  를 취한 뒤,  $\operatorname{argmax}$  를  $\operatorname{argmin}$  으로 바꿉니다.

- $\operatorname{argmax}_{w,t} P(w_{1:m}, t_{1:m} | S) = \prod_{i=1}^m P(t_i | t_{i-1}) \times P(w_i | t_i)$

- $\operatorname{argmin}_{w,t} -\log \prod_{i=1}^m P(t_i | t_{i-1}) \times P(w_i | t_i)$

- $= \sum_{i=1}^m -\log(P(w_i | t_i) \times P(t_i | t_{i-1}))$   
shortest path edge weight

- $(w_i, t_i)$  를 그래프의 마디로,  $\log(P(w_i | t_i) \times P(t_i | t_{i-1}))$  를 마디의 weight 로 설정하면 HMM 의 최적해는 shortest path 로 찾을 수 있습니다.

# HMM as Shortest path

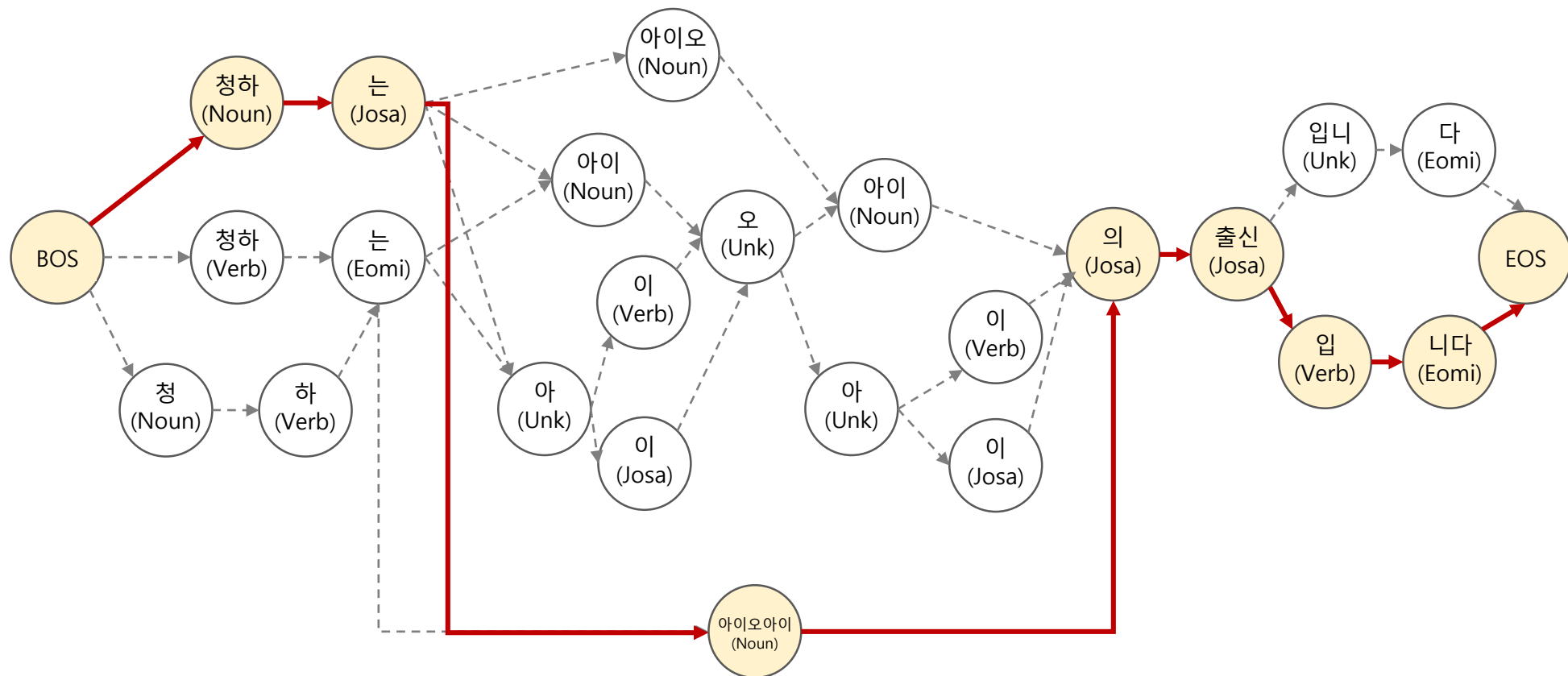
---

- 사전에 등록되어 있는 단어를 lookup 하여 마디 후보를 만듭니다.
- 마디 (U, V) weight 는  $-\log(P(V_t|U_t) \times P(V_w|V_t))$  를 이용하면 Hidden Markov Model (HMM) 을 이용하는 품사 판별기가 됩니다.

- $w(BOS \rightarrow \text{청하} / Noun) = -\log(P(Noun | BOS) \times P(\text{청하} | Noun))$
- $w(\text{청하} / Noun \rightarrow \text{는} / Josa) = -\log(P(Josa | Noun) \times P(\text{는} | Josa))$

# HMM as Shortest path

- (예문) 청하는 아이오아이의 출ship입니다



# Issues on HMM

---

- Issue #1. 학습데이터에 등장하지 않은  $x$  는  $P(x|y) = 0$  입니다.
  - 학습데이터의 '명사' 집합에 '아이오아이'가 존재하지 않았다면, '아이오아이' 라는 단어를 인식할 수 없습니다.
- Solution #1. 사용자 사전을 추가할 수 있습니다.
  - $P(x|y)$  에 새로운 단어를 추가하고,  $\sum_x P(x|y) = 1$  이 되도록 scaling 합니다.

# Issues on HMM

---

- Issue #2. Unguaranteed Independency Problem
  - 단어열은 앞, 뒤 단어 간에 상관성이 있습니다.
  - HMM 은 오로지 품사열  $(y_{i-1}, y_i)$  에 대해서만 상관성을 학습할 수 있습니다.
  - $(x_{i-1}, x_i)$  간의 상관성을 학습할 수 없기 때문에, 문맥을 고려하여  $x_i$  의 품사  $y_i$  를 추정할 수 없습니다.

# Issues on HMM

- Issue #3. Number of words
  - HMM 은  $P(word | tag)$  의 확률을 학습합니다.
  - 단어의 종류가 작은 품사의 emission prob. 는 상대적으로 큽니다. Bias 가 발생합니다.
    - $P(O | Noun) < P(O | Josa)$

Tag	Number of unique words
Noun	63968
Verb	3598
Adverb	3190
Eomi	1460
Adjective	849
Exclamation	464
Josa	158
Determiner	123



# Issues on HMM

---

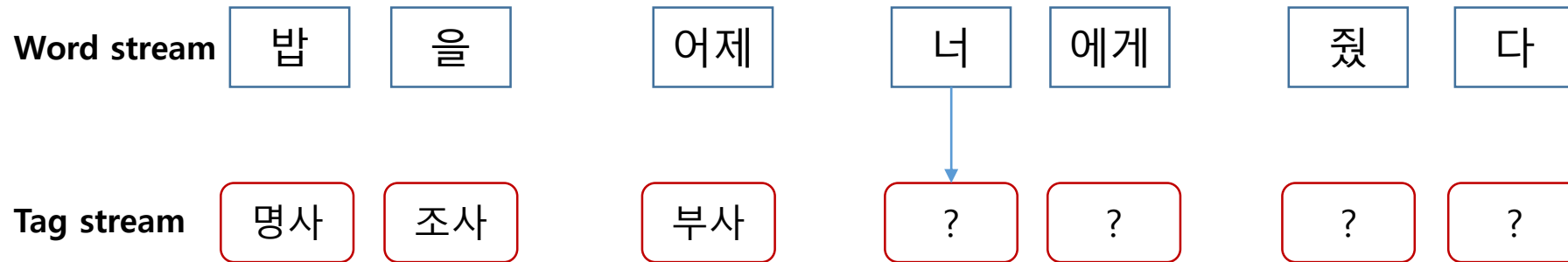
- Issue #4. Length bias

- HMM 은 길이가  $n$  인 문장을  $m$  개의 단어/품사열로 decode 합니다.
- 각 단어 품사열의  $P(y_{1:m}, x_{1:m}) = \prod_i P(x_i | y_i) \cdot P(y_i | y_{i-1})$  값을 계산합니다.
- 여러 번 확률을 곱할수록 그 값은 작아지기 때문에 HMM 은 적은 수의 단어/품사로 구성된 후보를 선호합니다.

# Brief review of Conditional Random Field

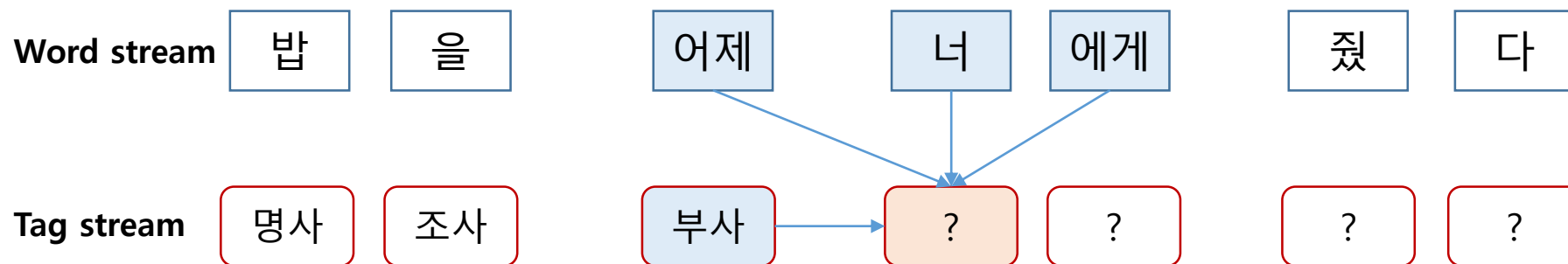
# Sequential labeling

- Unigram (independent classifier)
  - 단어  $x_i$  에 대하여 각각 품사  $t_i$  를 추정합니다.  $t_i = \operatorname{argmax} P(t_i|x_i)$
  - 한 단어는 여러 품사를 지니기 때문에 모호성이 발생합니다.
    - 이: 이빨(명사), 숫자(수사), 조사, 지시사, ...



# Sequential labeling

- 더 좋은 방법은 앞, 뒤의 단어와 품사 정보를 모두 활용하는 것입니다.
  - 문맥을 반영할 수 있습니다.
  - 이전 단어의 품사를 반영하면 큰 도움이 됩니다.
    - $(y_{i-1}, y_i) = (\text{조사}, \text{조사})$  인 경우를 방지할 수 있습니다.



# Potential function as Representation

---

- 길이가 3 인  $x = [3.2, 2.1, -0.5]$  에 두 개의 필터를 적용할 수도 있습니다.
  - 길이가 3 인 2 차원 벡터열이 만들어집니다.

$$F_1 = \text{1 if } x_i > 0 \text{ else } 0$$

$$F_2 = \text{1 if } x_i > 3 \text{ else } 0$$

$$v = [(1, 1) (1, 0), (0, 0)]$$

# Potential function as Representation

---

- 단어열도 필터를 적용하여 벡터열로 표현할 수 있습니다

$x = [\text{이것}, \text{은}, \text{예문}, \text{이다}]$

$F_1 = \text{1 if } x_{i-1}=\text{이것} \ \& \ x_i=\text{은} \ \text{else } 0$

$F_2 = \text{1 if } x_{i-1}=\text{이것} \ \& \ x_i=\text{예문} \ \text{else } 0$

$F_3 = \text{1 if } x_{i-1}=\text{은} \ \& \ x_i=\text{예문} \ \text{else } 0$

$v = [(0, 0, 0), (1, 0, 0), (0, 0, 1), (0, 0, 0)]$

# Potential function as Representation

---

- $(x_{i-1}, x_i, y_{i-1})$  을 이용하는 품사 판별을 위하여  $x_i$  를  $k$  차원의  $F_i$  로 표현합니다.

$F_{i1} = 1$  if (  $x_{i-1} = \text{'이것'}$ ,  $x_i = \text{'은'}$ ,  $y_{i-1} = \text{'명사'}$ ) else 0

$F_{i2} = 1$  if (  $x_{i-1} = \text{'은'}$ ,  $x_i = \text{'예문'}$ ,  $y_{i-1} = \text{'조사'}$ ) else 0

...

$F_{ik} = 1$  if (  $x_{i-1} = \text{'이'}$ ,  $x_i = \text{'단어'}$ ,  $y_{i-1} = \text{'조사'}$ ) else 0

# Potential function as Representation

---

- Potential function 은  $x_i$  가  $F_{ij}$  와 같은지 Boolean 으로 표현하기 때문에 대부분의 값이 0 인 sparse vector 입니다.

$$F_{i2} = \textcolor{red}{1} \text{ if } (x_{i-1} = \text{'은'}, x_i = \text{'예문'}, y_{i-1} = \text{'조사'}) \text{ else } \textcolor{red}{0}$$

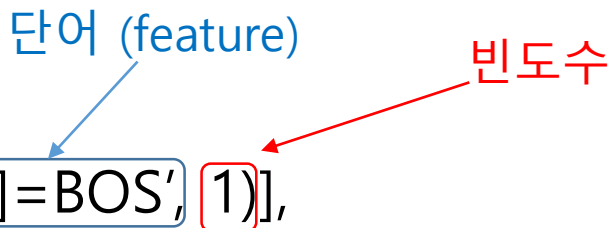


# Potential function as Representation

- 품사 판별을 위하여  $(x_{i-1}, x_i, y_{i-1})$  를 이용한다면,

- [이것, 은, 예문, 이다] 를 다음의 template을 이용

- $X[-1:0]$  : 앞 단어와 현재 단어
- $Y[-1]$  : 앞 단어의 품사

- [ [ ('x[-1:0]=BOS-이것', 1), ('y[-1]=BOS', 1)],  
[ ('x[-1:0]=이것-은', 1), ('y[-1]=Noun', 1) ],  
[ ('x[-1:0]=은-예문', 1), ('y[-1]=Josa', 1) ],  
[ ('x[-1:0]=예문-이다', 1), ('y[-1]=Noun', 1) ] ]
- 

# Potential function as Representation

- 마치 document – term frequency vector 처럼 해석할 수 있습니다.
  - [ [ ('x[-1:0]=BOS-이것', 1), ('y[-1]=BOS', 1)],  
[ ('x[-1:0]=이것-은', 1), ('y[-1]=Noun', 1) ],  
[ ('x[-1:0]=은-예문', 1), ('y[-1]=Josa', 1) ],  
[ ('x[-1:0]=예문-이다', 1), ('y[-1]=Noun', 1) ]]

char	Y	x[-1:0]=BOS-이것	x[-1:0]=이것-은	x[-1:0]=은-예문	x[-1:0]=예문-아다	..	y[-1]=BOS	y[-1]=Noun
이것	Noun	1	0	0	0	..	1	0
은	Josa	0	1	0	0		0	1
예문	Noun	0	0	1	1		0	0
이다	Adj	0	0	0	0		0	1

# Potential function as Representation

---

- $(x_{i-1}, x_{i+1}, y_{i-1})$  을 feature 로 이용한다면, 모르는 단어에 대한 품사 추정도 가능합니다.
  - (의/Josa,  $w$ , 는)  $\rightarrow$  'Noun' 의  $\lambda$  는 큰 값으로 학습됩니다  
한 단어  $w$  앞/뒤로 '의/Josa', '는' 이 위치하면  $w$  는 주로 'Noun' 입니다.

# Classification

---

- CRF는 문장 전체,  $x_{1:n}$ ,  $y_{1:n}$  를 이용하여 만든 features 를 이용하여 logistic regression 을 수행합니다.
  - $x_{1:n}$  가 될 수 있는 다양한  $y_{1:n}$  중에서 확률이 가장 큰  $y_{1:n}$  를 선택합니다.

**CRF**

$$P(y|x) = \frac{\exp(\sum_{j=1}^m \sum_{i=1}^n \lambda_j f_j(x, i, y_i, y_{i-1}))}{\sum_{y'} \exp(\sum_{j=1}^m \sum_{i=1}^n \lambda_j f_j(x, i, y_i, y_{i-1}))}$$

**HMM**

$$P(y|x) = \prod_i P(x_i|y_i)P(y_i|y_{i-1})$$

# Conditional Random Field

---

- CRF 는 HMM 의 정보를 학습할 수도 있습니다.
  - Potential function 을  $y_i, y_{i-1}$  성분이 있는  $g_j(y_i, y_{i-1}, i)$  와  $x_i, y_i$  성분이 있는  $f_j(x, y_i, i)$  로 나눌 수 있습니다.
  - $g_j$  는 transition 을,  $f_j$  은 emission 을 학습합니다.

$$P(y|x) = \frac{\exp(\sum_{j=1}^m \sum_{i=1}^n \lambda_j f_j(x, i, y_i, y_{i-1}))}{\sum_{y'} \exp(\sum_{j=1}^m \sum_{i=1}^n \lambda_j f_j(x, i, y_i, y_{i-1}))} = \frac{\exp(\sum_{j=1}^m (\sum_{i=1}^n \lambda_j f_j(x, y_i, i) + \sum_{i=1}^n \mu_j g_j(y_i, y_{i-1}, i)))}{\sum_{y'} \exp(\sum_{j=1}^m \sum_{i=1}^n \lambda_j f_j(x, i, y_i, y_{i-1}))}$$

# Conditional Random Field

---

- CRF 는 HMM 보다 다양한 features 를 학습할 수 있습니다.
  - 문맥 정보를 학습할 수 있으며,
  - potential function 을 설계하기에 따라 미등록 단어도 인식할 수 있습니다.
- 우리가 풀어야 하는 식은  $\operatorname{argmax}_y P(y|x)$  입니다.
  - CRF 는 직접적으로  $P(y|x)$  로부터  $y$  를 구하지만,  
HMM 은  $P(x|y) \times P(y) = P(x, y)$  로부터 간접적으로  $y$  를 구합니다.

Generating (word, pos) candidates  
from sentence

- 
- 품사 판별, 형태소 분석의 과정은 두 단계로 구성됩니다.

(1) 주어진 문장으로부터 단어/형태소 열의 후보들을 만들고 (generation),

(2) 그 중 가장 높은 점수를 얻는 후보를 선택합니다 (evaluation).

**문장:** '이것은 예문이다'

**후보 1:** [이것/Noun, 은/Josa, 예문/Noun, 이다/Adj]

**후보 2:** [이것/Noun, 은/Josa, 예문/Noun, 이/Noun, 다/Noun],

**후보 3:** [이/Noun, 것/Noun, 은/Josa, 예문/Noun, 이/Noun, 다/Noun],

...



- HMM, CRF 모두 사전을 기반으로 (단어, 품사) 후보를 만듭니다.

‘이것은 예문이다’

Word	Tag	b	e
이	Noun	0	1
이	Josa	0	1
이것	Noun	0	2
것	Noun	1	2
은	Josa	2	3
예문	Noun	3	5
이	Noun	5	6
이	Josa	5	6
이다	Adjective	5	7
다	Noun	6	7

품사	단어
Noun	이것, 예문, 집, 말, 이, 것, 다
Josa	은, 는, 에, 이
Adjective	이다,
Verb	가다, 하다

- HMM, CRF 모두 사전을 기반으로 (단어, 품사) 열 후보를 만듭니다.

‘이것은 예문이다’

Word	Tag	b	e
이	Noun	0	1
이	Josa	0	1
이것	Noun	0	2
것	Noun	1	2
은	Josa	2	3
예문	Noun	3	5
이	Noun	5	6
이	Josa	5	6
이다	Adjective	5	7
다	Noun	6	7



[ (이/Noun, 것/Noun, 은/Josa, 예문/Noun, 이/Josa, 다/Noun),  
 (이/Josa, 것/Noun, 은/Josa, 예문/Noun, 이/Josa, 다/Noun),  
 (이것/Noun, 은/Josa, 예문/Noun, 이/Josa, 다/Noun),  
 (이것/Noun, 은/Josa, 예문/Noun, 이다/Adjective),  
 ...  
 ]

- 중복 계산을 피하기 위하여 beam-search 나 shortest path 가 이용됩니다.

‘이것은 예문이다’

Word	Tag	b	e
이	Noun	0	1
이	Josa	0	1
이것	Noun	0	2
것	Noun	1	2
은	Josa	2	3
예문	Noun	3	5
이	Noun	5	6
이	Josa	5	6
이다	Adjective	5	7
다	Noun	6	7



[ (이/Noun, 것/Noun, 은/Josa, 예문/Noun, 이/Josa, 다/Noun),  
 (이/Josa, 것/Noun, 은/Josa, 예문/Noun, 이/Josa, 다/Noun),  
 (이것/Noun, 은/Josa, 예문/Noun, 이/Josa, 다/Noun),  
 (이것/Noun, 은/Josa, 예문/Noun, 이다/Adjective),  
 ...  
 ]

- HMM, CRF 모두 사전을 기반으로 (형태소, 품사) 후보도 만듭니다.

‘집에 간다고 말했다’

Word	Tag	b	e
집	Noun	0	1
에	Josa	1	2
간	Noun	2	3
가 + ㄴ다고	Verb + Eomi	2	5
다	Noun	3	4
고	Unknown	4	5
말	Noun	5	6
하+았다	Verb + Eomi	6	8

품사	단어
Noun	이것, 예문, 집, 말, 이, 것, 다, 간
Josa	은, 는, 에, 이
Adjective	이다,
Verb	가다, 하다
Eomi	-다, -ㄴ다고, -았다

형태소 사전에 어간의 원형과 어미가 있을 경우,  
“어간 + 어미 결합 규칙”을 이용하여 용언을 인식합니다.

- HMM 은 미등록 단어에 대한 인식이 불가능합니다.

'집에 간다고 말했다'

Word	Tag	b	e
집	Noun	0	1
에	Josa	1	2
간	Noun	2	3
가 + ㄴ다고	Verb + Eomi	2	5
다	Noun	3	4
고	Unknown	4	5
말	Noun	5	6
하+았다	Verb + Eomi	6	8

(집/Noun, 에/Josa, 간/Noun, 다/Noun, **고/?**, 말/Noun, 하+았다/Verb+Eomi)



$$P_{HMM}(y|x) = P(\text{집} | \text{Noun}) \times P(\text{에} | \text{Noun}) \times \dots \times P(\text{고} | \text{?}) \times \dots \\ \times P(\text{말} | \text{Noun}) \times \dots$$

- HMM 은 미등록 단어에 대한 품사 추정을 할 수 있습니다.
- 앞/뒤 품사의 transition probability 를 이용합니다.

‘집에 간다고 말했다’

Word	Tag	b	e
집	Noun	0	1
에	Josa	1	2
간	Noun	2	3
가 + ㄴ다고	Verb + Eomi	2	5
다	Noun	3	4
고	Unknown	4	5
말	Noun	5	6
하+았다	Verb + Eomi	6	8

(집/Noun, 에/Josa, 간/Noun, 다/Noun, **고**/? , 말/Noun, 하+았다/Verb+Eomi)



$$\operatorname{argmax}_t \sum_t P(t \mid \text{Noun}) + P(\text{Noun} \mid t)$$

- HMM 은 사용자 사전 등록이 쉽습니다.
- 분석 후보마다 상대 비교를 하기 때문에  $\sum_w P(w | t) = 1$  일 필요는 없습니다.

‘집에 간다고 말했다’

Word	Tag	b	e
집	Noun	0	1
에	Josa	1	2
간	Noun	2	3
가 + ㄴ다고	Verb + Eomi	2	5
다	Noun	3	4
고	Unknown	4	5
말	Noun	5	6
하+았다	Verb + Eomi	6	8

(집/Noun, 에/Josa, 간/Noun, 다/Noun, **고**/? , 말/Noun, 하+았다/Verb+Eomi)



set P(**고** | Eomi) = your\_preference

- CRF 는 생성한 (단어, 품사) 후보열을 이용하여 feature 를 생성합니다.  
이를 이용하여 evaluation 을 할 수 있습니다.

‘집에 간다고 말했다’

Word	Tag	b	e
집	Noun	0	1
에	Josa	1	2
간	Noun	2	3
가 + ㄴ 다고	Verb + Eomi	2	5
다	Noun	3	4
고	Unknown	4	5
말	Noun	5	6
하+았다	Verb + Eomi	6	8

단어열: (집, 에, 가+ㄴ 다고, 말, 하+았다),  
품사열: (Noun, Josa, Verb+Eomi, Noun, Verb+Eomi)



$F_1 = 1$  IF  $x_{i-1} = \text{'집'}$  &  $x_i = \text{'에'}$  ELSE 0

$F_2 = 1$  IF  $x_i = \text{'말'}$  ELSE 0

...



- CRF 는 beam-search 를 이용하여 최적의 후보를 찾을 수 있습니다.

(예시 k=3)

오	늘	의	날	씨
(오/Noun)				
(오/Verb)				
(오/Eomi)				

- CRF 는 beam-search 를 이용하여 최적의 후보를 찾을 수 있습니다.

(예시 k=3)

오	늘	의	날	씨
(오/Noun)				
(오/Verb)				
(오/Eomi)				
	(오/Noun, 늘/Noun)			
	(오/Verb, 늘/Noun)			
	(오/Eomi, 늘/Noun)			
	(오늘/Noun)			

- CRF 는 beam-search 를 이용하여 최적의 후보를 찾을 수 있습니다.

(예시 k=3)

오	늘	의	날	씨
(오/Noun)	(오/Noun, 늘/Noun)			
(오/Verb)	(오/Verb, 늘/Noun)			
(오/Eomi)	(오늘/Noun)			
		(오/Noun, 늘/Noun, 의/Josa)		
		<del>(오/Verb, 늘/Noun, 의/Josa)</del>		
		<del>(오/Verb, 늘/Noun, 의/Noun)</del>		
		(오늘/Noun, 의/Josa)		
		(오늘/Noun, 의/Noun)		

- CRF 는 beam-search 를 이용하여 최적의 후보를 찾을 수 있습니다.

(예시 k=3)

오	늘	의	날	씨
(오/Noun)	(오/Noun, 늘/Noun)	(오/Noun, 늘/Noun, 의/Josa)		
(오/Verb)	(오/Verb, 늘/Noun)	(오늘/Noun, 의/Josa)		
(오/Eomi)	(오늘/Noun)	(오늘/Noun, 의/Noun)		

- CRF 는 beam-search 를 이용하여 최적의 후보를 찾을 수 있습니다.

(예시 k=3)

오	늘	의	날	씨
(오/Noun)	(오/Noun, 늘/Noun)	(오/Noun, 늘/Noun, 의/Josa)	(오늘/Noun, 의/Josa, 날/Noun)	
(오/Verb)	(오/Verb, 늘/Noun)	(오늘/Noun, 의/Josa)	(오늘/Noun, 의/Josa, 날/Verb)	
(오/Eomi)	(오늘/Noun)	(오늘/Noun, 의/Noun)	(오늘/Noun, 의/Josa, 나+르/Verb+Emi)	
			(오늘/Noun, 의/Josa, 날/Noun, 씨/Noun)	
			(오늘/Noun, 의/Josa, 날/Verb, 씨/Noun)	
			(오늘/Noun, 의/Josa, 나+르/Verb+Emi, 씨/Noun)	
			...	
			(오늘/Noun, 의/Josa, 날씨/Noun)	

# Issues on CRF

- Issue #1. Occurrence bias

- CRF 는  $P(word | tag)$  대신  $Score(feature \rightarrow tag)$  를 학습합니다.
- 그러나 한 단어  $w$  가 품사  $t$  로 등장한 횟수는 여전히 차이가 납니다.
- $(w, t)$  의 빈도수가 높을수록 feature " $x[0] = w$ " 이나 " $x[-1,0] = w_{-1}w$ " 의 빈도수도 많아집니다.

Tag	Frequency / Unique numbers
Noun	49.38
Eomi	1492.17
Josa	14478.32
Verb	414.16
Modifier	176.73

# Issues on CRF

- Issue #1. Occurrence bias

- Gradient descent 를 이용하는 CRF solver 들은 확실한 features 에 대해서는 자주 등장할수록 그 방향으로 coefficient 의 크기를 증가시킵니다.
- " $s(x[0] = 0 \rightarrow Noun) \ll s(x[0] = 0 \rightarrow Josa)$ " 경향이 발생합니다.
- L2 regularization 을 적용하여도 전반적인 coef 의 크기가 작아질 뿐, 순위는 거의 동일합니다

Tag	Max coef of $x[0]=w$ (Frequency / Unique numbers)
Noun	4.739 (49.38)
Eomi	10.900 (1492.17)
Josa	13.669 (14478.32)

# Issues on CRF

---

- Issue #1. Occurrence bias
  - 명사는 가장 많이 등장하며, 그 종류도 많은 품사입니다.
  - Bias 를 보완하기 위하여 다른 품사보다 명사를 선호하는 preference score 를 부여할 수 있습니다.



# Issues on CRF

---

- Issue #2. Preference shorter words
  - 대부분의 글자는 단어로 등록되어 있으며, 긴 단어 (특히 명사)의 빈도수는 짧은 단어의 빈도수보다 작습니다.
  - 짧고 품사가 명확한 단어는 큰 coefficient 를 지닙니다.
  - '아이오' 보다 '아 + 이 + 오' 가 더 선호될 수 있습니다.

# Issues on CRF

---

- Issue #2. Preference shorter words
  - Heuristics 가 이용될 수 있습니다.
    - 길이가 1 인 경우에 penalty 를 부여합니다.
    - 길이가 긴 명사에 대하여 preference 를 부여합니다.

- 
- HMM <sup>^1</sup> 과 CRF <sup>^2</sup> 를 이용한 구현체를 github 에 올려두었습니다.
    - 세종 말뭉치를 이용한 학습된 모델도 제공합니다

[1] [https://github.com/lovit/hmm\\_postagger](https://github.com/lovit/hmm_postagger)

[2] [https://github.com/lovit/crf\\_postagger](https://github.com/lovit/crf_postagger)