KU KONKUK UNIVERSITY

KAIST

형태소 분석
(Morphological Analysis)

건국대학교 컴퓨터공학부 /
KAIST 전산학부 (겸직)

김학수

자연어처리연구실

Core Layers of NLU

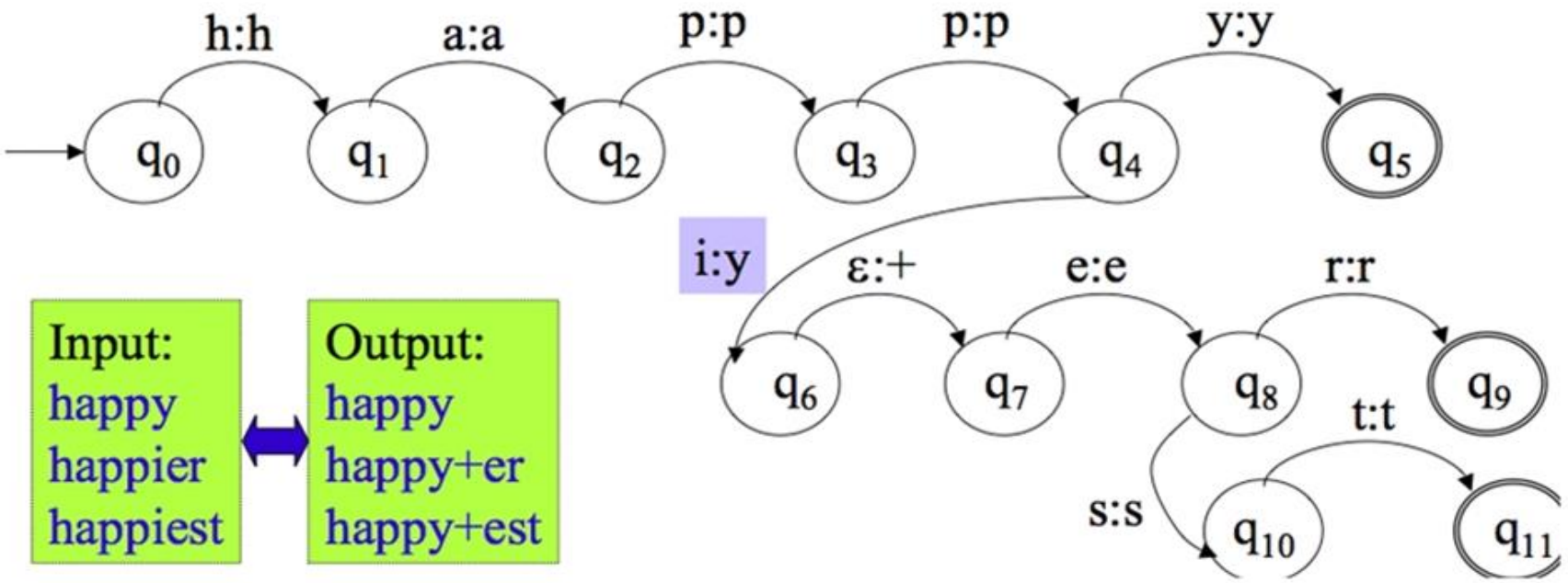
단계	설명	예제: 나는 그 과자를 먹었다.
형태소 분석	문장을 형태소열로 분리하고 품사를 부착하는 단계	나/대명사+는/조사 그/대명사 과자/명사+를/조사 먹/동사+었/선어말어미+다/어미+./기호
구문 분석	문장의 문법적 적합성과 어절의 구문적 역할(주어, 목적어 등)을 찾는 단계	[SUBJ: 나는 [[MOD: 그 [OBJ: 과자를]] 먹었다]]
의미 분석	문장을 구성하는 술어와 논항들 사이의 의미적 적합성을 분석하는 단계	PREDICATE: 먹다 AGENT: 나/ANIMATE OBJECT: 그 과자/EATABLE
담화 분석	대화 문맥을 파악하여 상호참조를 해결하고 의도를 파악하는 단계	SPEECH ACT: STATEMENT PREDICATE: 먹다 AGENT: 홍길동/ANIMATE OBJECT: 꼬깔콘/EATABLE

형태소 분석

- 형태소 분석이란?
  - 입력 어절을 모든 가능한 형태소의 조합으로 분리하는 과정
  - 형태소 분석 = 형태소 분리 + 품사 부착
- 어절
  - 띄어쓰기 단위
  - 영어에서는 1어절이 1단어
  - 한국어에서는 1어절이 1~n 단어
    - 예) 감기는, 나는
- 형태소
  - 의미를 갖는 최소 단위
  - 사전에 등록되어 있는 단일 품사를 갖는 단위

영어에서 형태소 분석

- FST (Finite State Transducers) : A FST is a two-tape automaton that recognizes or generates pairs of strings.



영어에서 형태소 분석

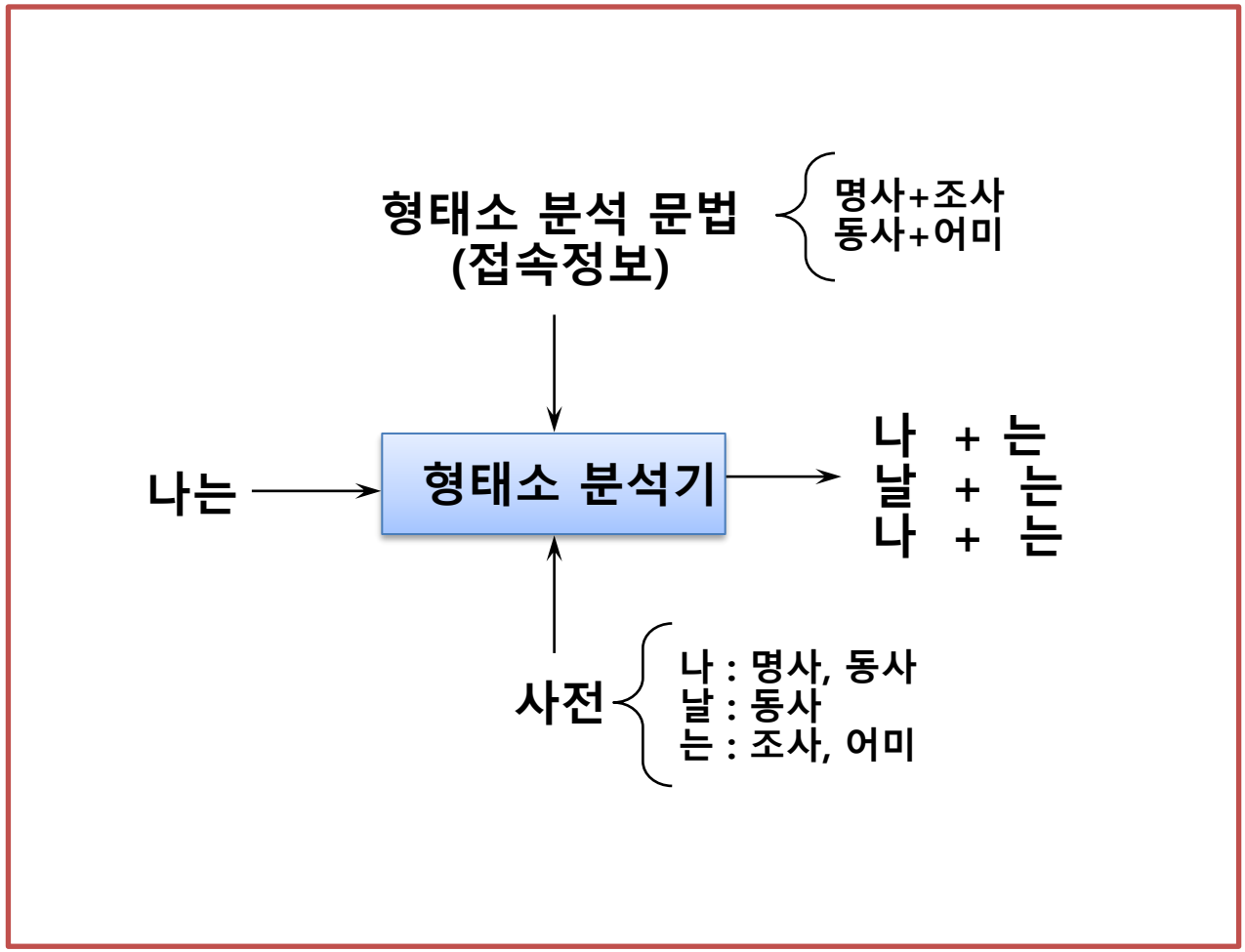
FST Table

	Input									
State	h:h	a:a	p:p	y:y	i:y	ε:+	e:e	r:r	s:s	t:t
0	**1**	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
1	Ø	2	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
2	Ø	Ø	3	Ø	Ø	Ø	Ø	Ø	Ø	Ø
3	Ø	Ø	**4**	Ø	Ø	Ø	Ø	Ø	Ø	Ø
4	Ø	Ø	Ø	**5**	**6**	Ø	Ø	Ø	Ø	Ø
5:	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø
6	Ø	Ø	Ø	Ø	Ø	**7**	Ø	Ø	Ø	Ø
7	Ø	Ø	Ø	Ø	Ø	Ø	**8**	Ø	Ø	Ø
8	Ø	Ø	Ø	Ø	Ø	Ø	**10**	**9**	Ø	Ø
9:	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

한국어의 특성

- 음절의 특징으로 한글은 모아 쓰기
  - 한글 = 초/중/종성의 조합
  - 완성형 코드 사용 불가능
  - 풀어쓰기 코드, 조합형 코드
- 띄어쓰기가 일정하지 않고 복잡(맞춤법)
  - 띄어쓰기 오류가 포함된 경우가 빈번(할 수 있다)
- 첨가어로 첨용과 활용이 자유로움
  - 조사나 접사가 붙어서 문법적 관계를 형성
  - 들 : 낱말 + 들, 나무 + 들
  - 이 : 본인 + 들 + 이, 처리 + 하 + ㅁ + 이
- 불규칙 용언 활용과 음운 현상이 발달
  - 고맙다(ㅂ불규칙 활용) : 고맙게도, 고마운, 고마워서
  - 오다 : 오면, 온, 올, 와서

한국어에서 형태소 분석



형태소 분석과 사전

- 형태소 분석에서 사전의 정의
  - 언어의 어휘적 정보를 저장하는 장소
  - 표제어 + 어휘정보(품사)
  - 표제어: 형태소 수준

- 형태소 분석에서 사전의 쓰임
  - 어절 내에 있는 모든 형태소(사전 표제어)를 사전을 이용하여 찾아내고, 찾아낸 형태소들 사이의 결합관계의 적합성을 검사

접속 정보 (1/2)

- 현재 형태소의 왼쪽이나 오른쪽에 붙을 수 있는 품사에 대한 정보

- 좌접속, 우접속을 분리
  - 좌 접속 범주 : 좌측에 붙을 수 있는 것을 기준
  - 우 접속 범주 : 우측에 붙을 수 있는 것을 기준

접속 정보 (2/2)

- 좌접속 범주의 예

대범주	소범주	세부범주	예	좌측에 접속가능
명사고유명사			김유신	접속불가
	보통명사	한자형	분석, 설계	한자형접두사
		순한글형	자랑, 노래	순한글형접두사
	의존명사	단위성	개, 분, 마리	숫자, 양수사
		기타	것, 등, 뿐	접속불가
접미사	인간고유명사형		씨, 군, 양	인명고유명사
	지명고유명사형		도, 시, 군	지명고유명사
	한자보통명사형		국, 실, 군	한자고유명사
숫자			1,2,3,4	숫자
기호	(특수문자)		@, #, $	용언어간,.. 제외

세종 품사 태그 (1/3)

대분류	소분류	세분류
(1) 체언	명사NN	일반명사NNG 고유명사NNP 의존명사NNB
	대명사NP	대명사NP
	수사NR	수사NR
(2) 용언	동사VV	동사VV
	형용사VA	형용사VA
	보조용언VX	보조용언VX
	지정사VC	긍정지정사VCP 부정지정사VCN
(3) 수식언	관형사MM	
	부사MA	일반부사MAG
		접속부사MAJ
(4) 독립언	감탄사IC	감탄사IC

45개

세종 품사 태그 (2/3)

(5) 관계언	격조사JK	주격조사JKS
		보격조사JKC
		관형격조사JKG
		목적격조사JKO
		부사격조사JKB
		호격조사JKV
		인용격조사JKQ
	보조사JX	보조사JX
	접속조사JC	접속조사JC
(6) 의존형태	어미E	선어말어미EP
		종결어미EF
		연결어미EC
		명사형전성어미ETN
		관형형전성어미ETM
	접두사XP	체언접두사XPN
	접미사 XS	명사파생접미사XSN
		동사파생접미사XSV
		형용사파생접미사XSA
	어근XR	어근XR

세종 품사 태그 (3/3)

(7) 기호	마침표,물음표, 느낌표	SF
	쉼표, 가운뎃점, 콜론, 빗금	SP
	따옴표, 괄호표, 줄표	SS
	줄임표	SE
	붙임표(물결, 숨김, 빠짐)	SO
	외국어	SL
	한자	SH
	기타 기호(논리 수학기호, 화폐 기호) 등)	SW
	명사추정범주	NF
	용언추정범주	NV
	숫자	SN
	분석불능범주	NA

형태소 분석 방법 (1/2)

- 최장 일치법
  - 부분 문자열 중에서 가장 긴 형태소를 우선적으로 선택하는 방법
    - 결혼식에서만 → 결혼식/명사 vs. 결혼/명사+식/접사
  - 좌 최장 일치
    - 좌방향으로 최장 일치 지점을 찾은 후에 품사를 부여
    - 감기는 → 감기 + 는 → 감기(명사/동사) + 는(조사/어미)
      마이크로는 → 마이크+로는 → 마이크(명사) + 로는(조사)
  - 우 최장 일치
    - 우방향으로 최장 일치 지점을 찾은 후에 품사를 부여
    - 마이크로는 → 마이크로 + 는 → 마이크로(명사) + 는(조사)
  - 좌우 최장 일치
    - 좌방향 최장일치의 끝을 구한 후에 이 위치값과 중첩 정보를 이용하여 우방향 최장일치의 끝 위치를 계산

한국어 형태소 분석 방법 (2/2)

- Tabular Parsing
  - 가정
    - 음소의 결합 관계 규명이 곧, 형태소 분석 결과
    - 예: 'ㄱㅏㅁㄱㅣㄴ-ㄴ' => 'ㄱㅏㅁㄱㅣ + ㄴ-ㄴ', 'ㄱㅏㅁ + ㄱㅣ + ㄴ—ㄴ'
  - 결합 제한 규칙
    - 음소간 결합 관계를 기술
  - 원형 복원 규칙
    - 음소간 결합 관계에 따른 복원 규칙을 사용
  - bottom-up 방식
  - 장점
    - 구현이 용이
    - 높은 정확도
  - 단점
    - 사전 탐색 부담
    - 옳은 분석 결과들을 생성하기 위해 백트래킹(backtracking)을 수행하는 비효율성

Tabular Parsing Algorithm(Right -> Left)

		1	2	3	4	5	6	7	8
ㄱ (초)	1								
ㅏ (중)	2								
ㅁ (종)	3								
ㄱ (초)	4								
ㅣ (중)	5								
ㄴ (초)	6								
ㅡ (중)	7								
ㄴ (종)	8	"ㄴ"사전검색							

접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

		1	2	3	4	5	6	7	8
ㄱ (초)	1								
ㅏ (중)	2								
ㅁ (종)	3								
ㄱ (초)	4								
ㅣ (중)	5								
ㄴ (초)	6	"ㄴ"사전검색							
ㅡ (중)	7								
ㄴ (종)	8	"ㄴ"사전검색							

접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

		1	2	3	4	5	6	7	8
ㄱ (초)	1								
ㅏ (중)	2								
ㅁ (종)	3								
ㄱ (초)	4								
ㅣ (중)	5								
ㄴ (초)	6	"ㄴ"사전검색	"느"사전검색						
ㅡ (중)	7								
ㄴ (종)	8	"ㄴ"사전검색							

접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

		1	2	3	4	5	6	7	8
ㄱ (초)	1								
ㅏ (중)	2								
ㅁ (종)	3								
ㄱ (초)	4								
ㅣ (중)	5								
ㄴ (초)	6	"ㄴ"사전검색	"느"사전검색	"는"사전검색 ->는:보조사					
ㅡ (중)	7								
ㄴ (종)	8	"ㄴ"사전검색							

접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

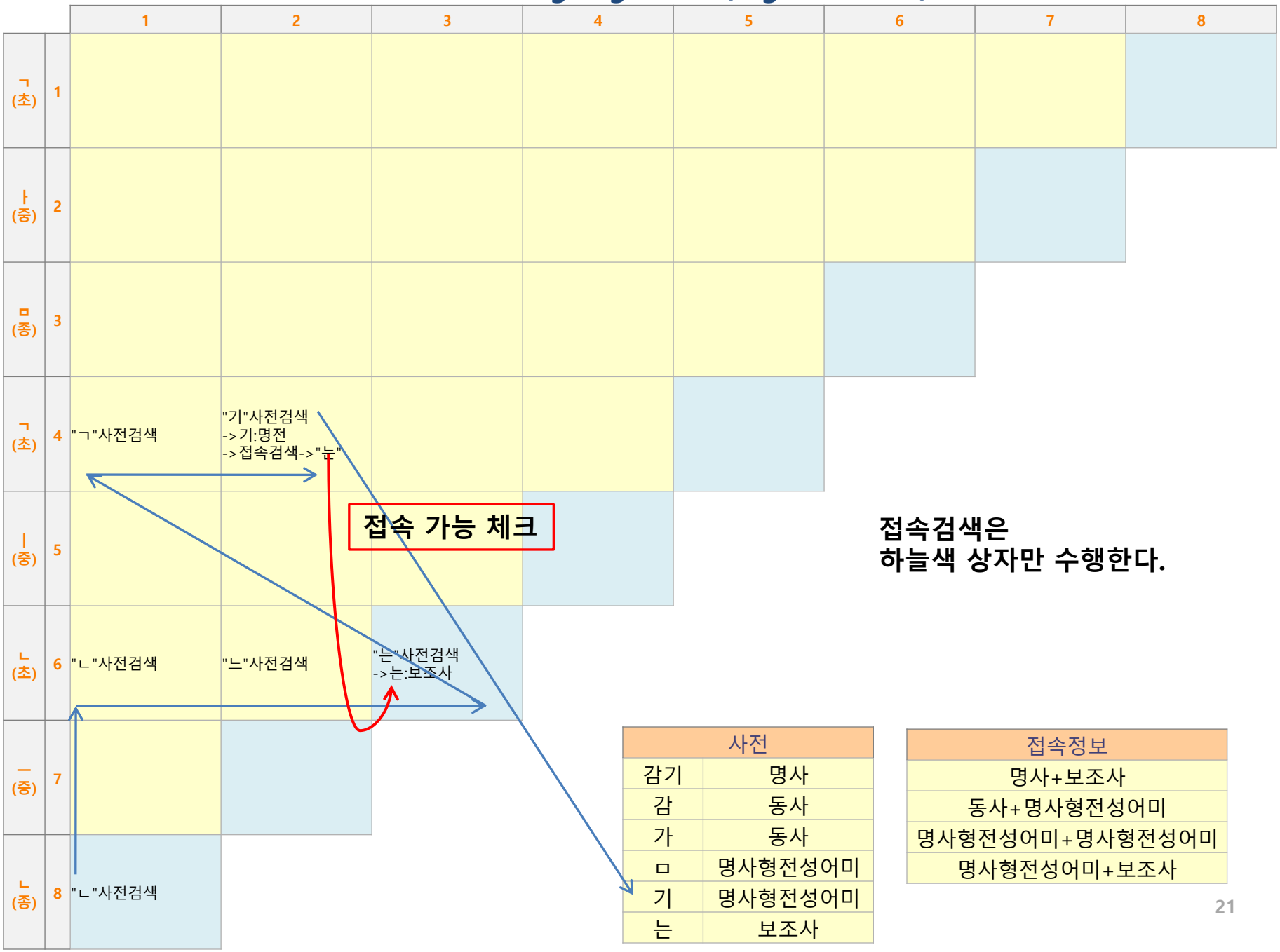
		1	2	3	4	5	6	7	8
ㄱ (초)	1								
ㅏ (중)	2								
ㅁ (종)	3								
ㄱ (초)	4	"ㄱ"사전검색							
ㅣ (중)	5								
ㄴ (초)	6	"ㄴ"사전검색	"느"사전검색	"는"사전검색 ->는:보조사					
ㅡ (중)	7								
ㄴ (종)	8	"ㄴ"사전검색							

접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

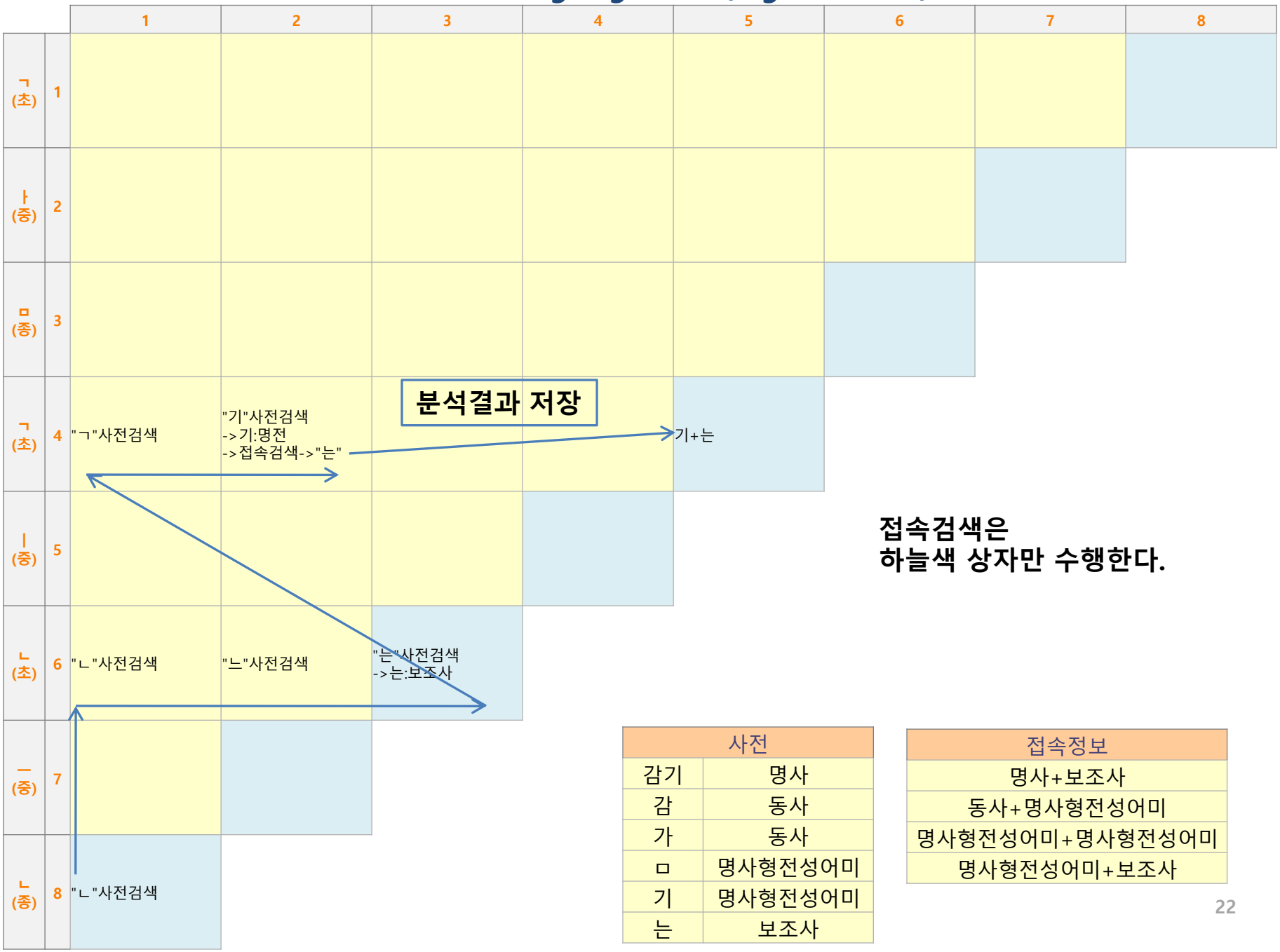


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

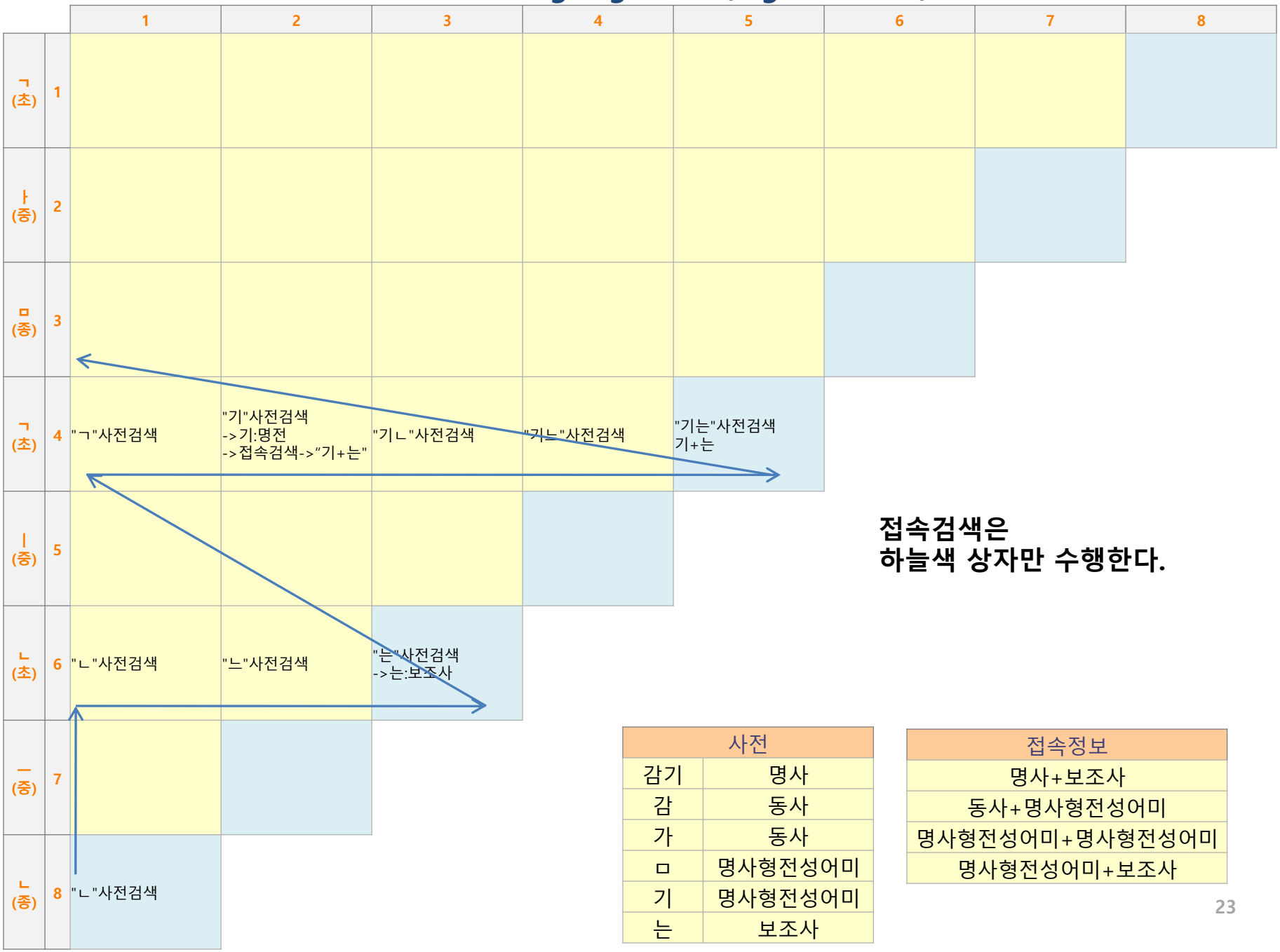


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

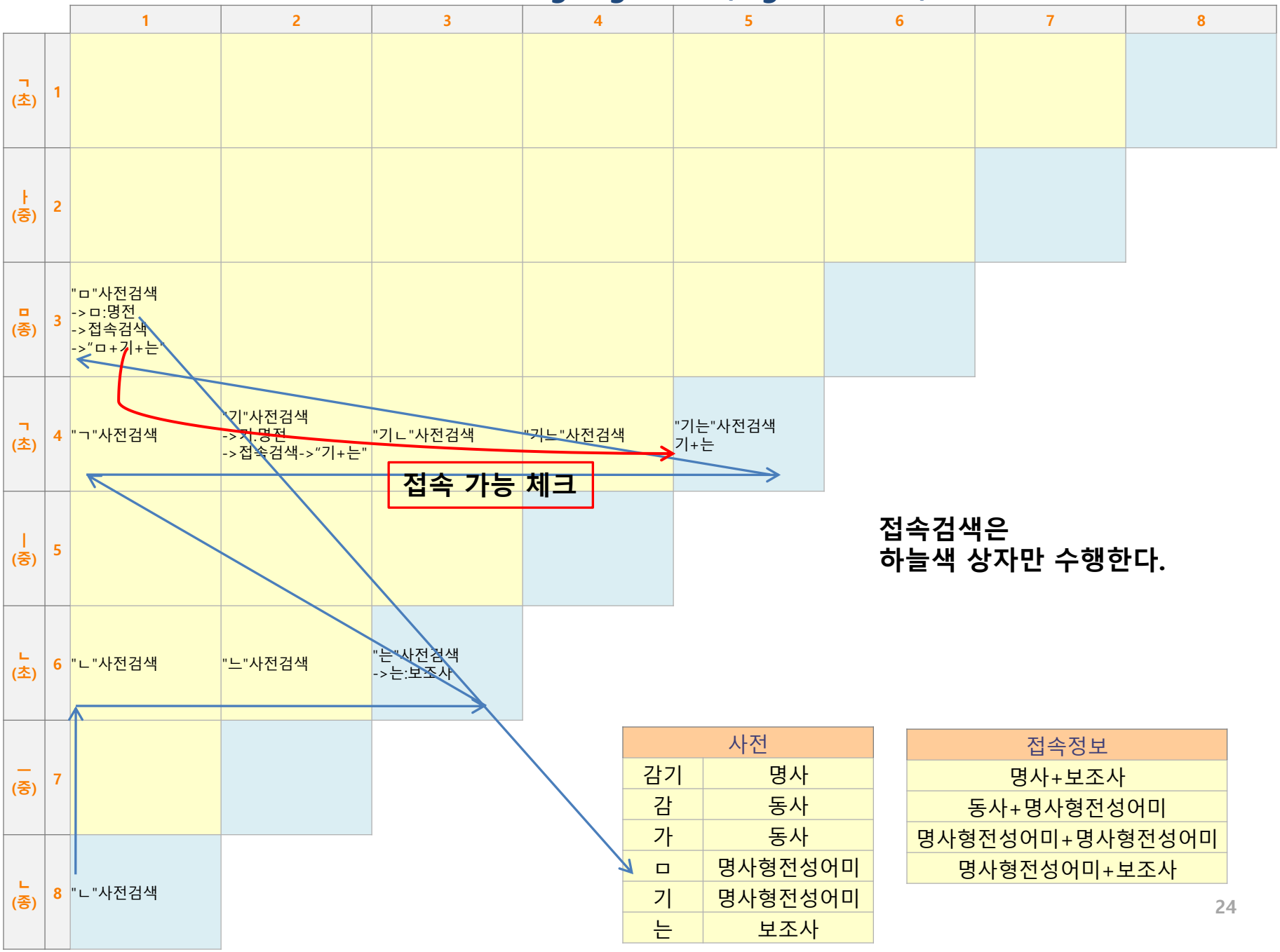


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

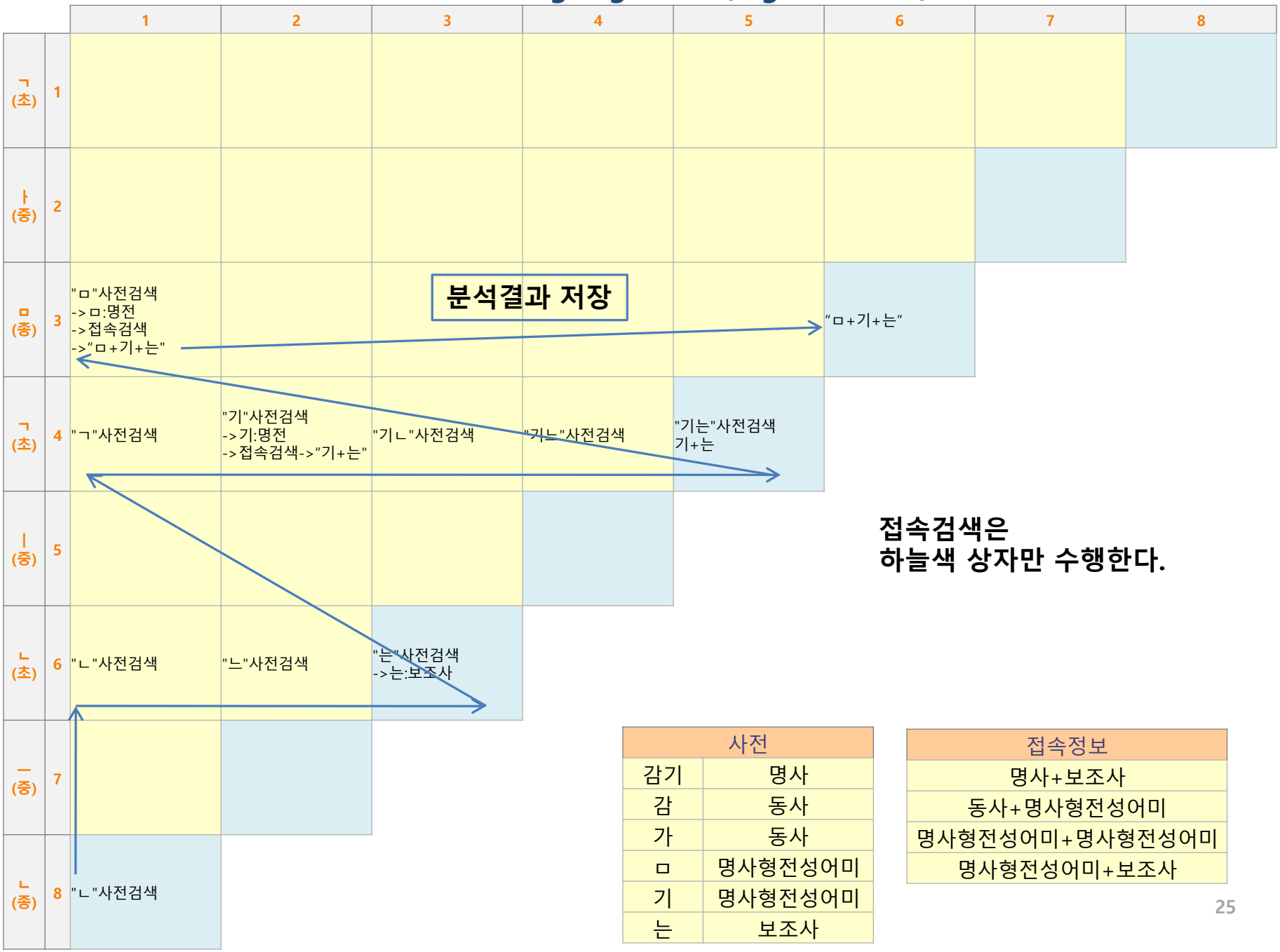


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

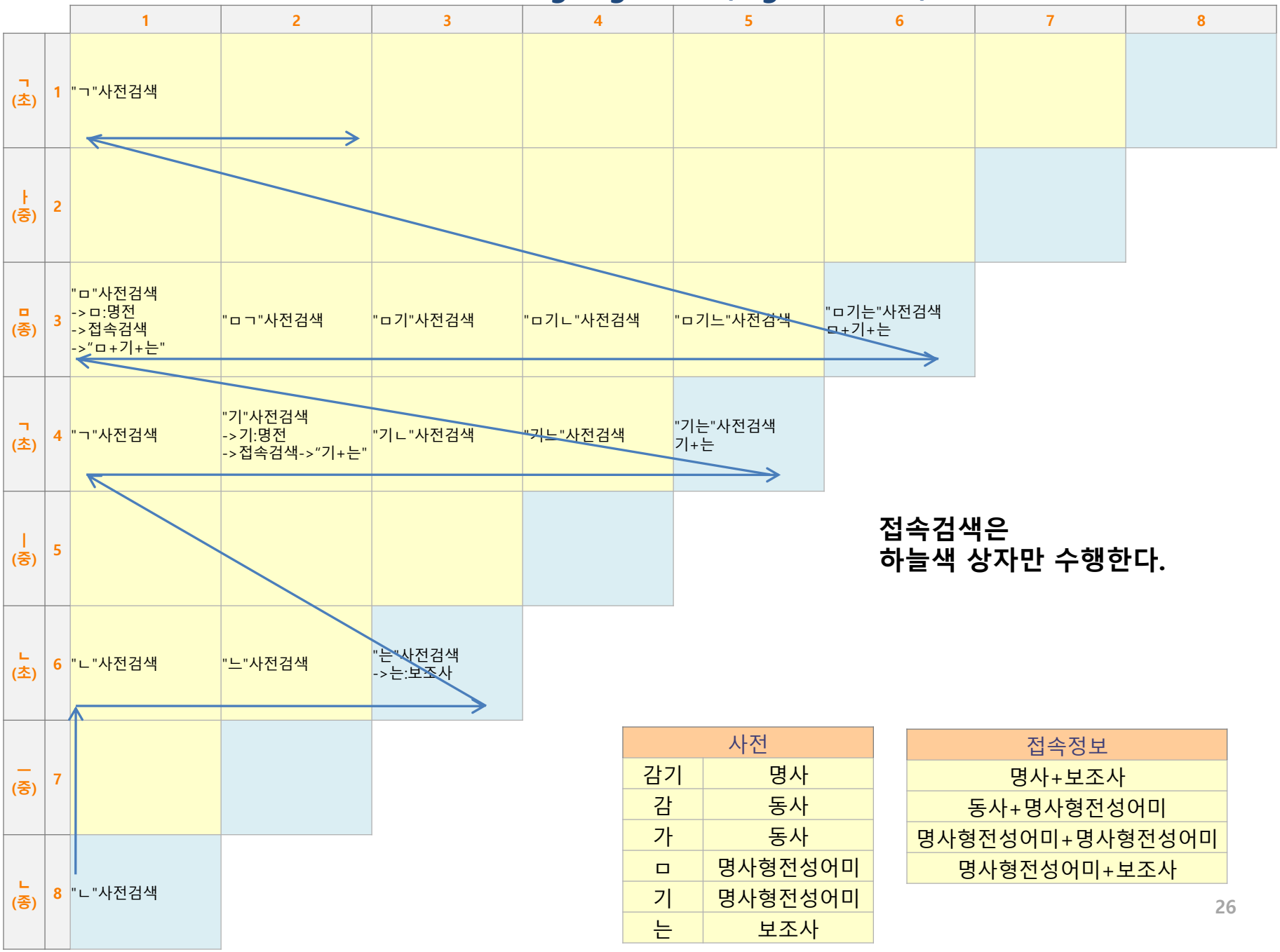


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

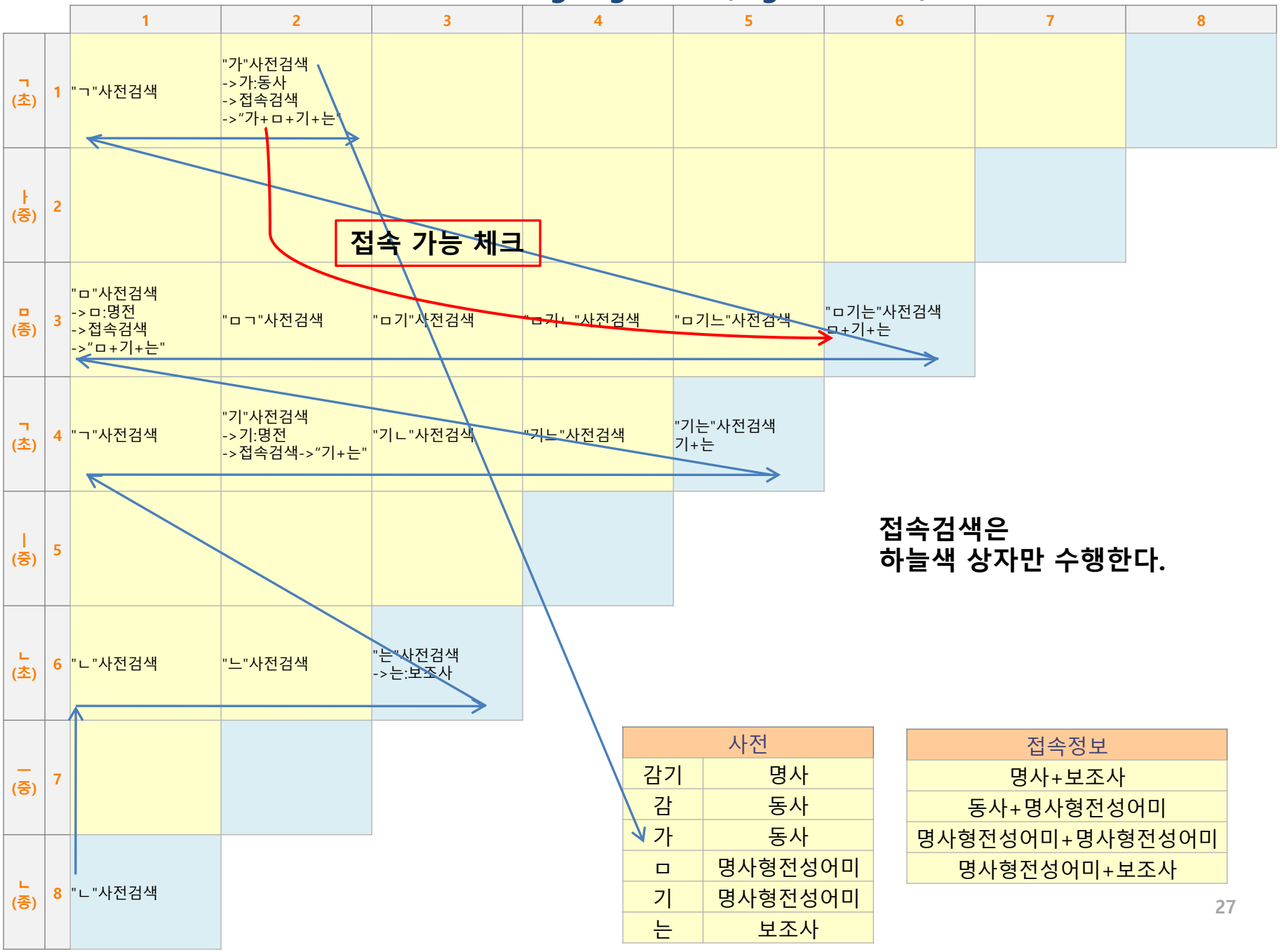


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

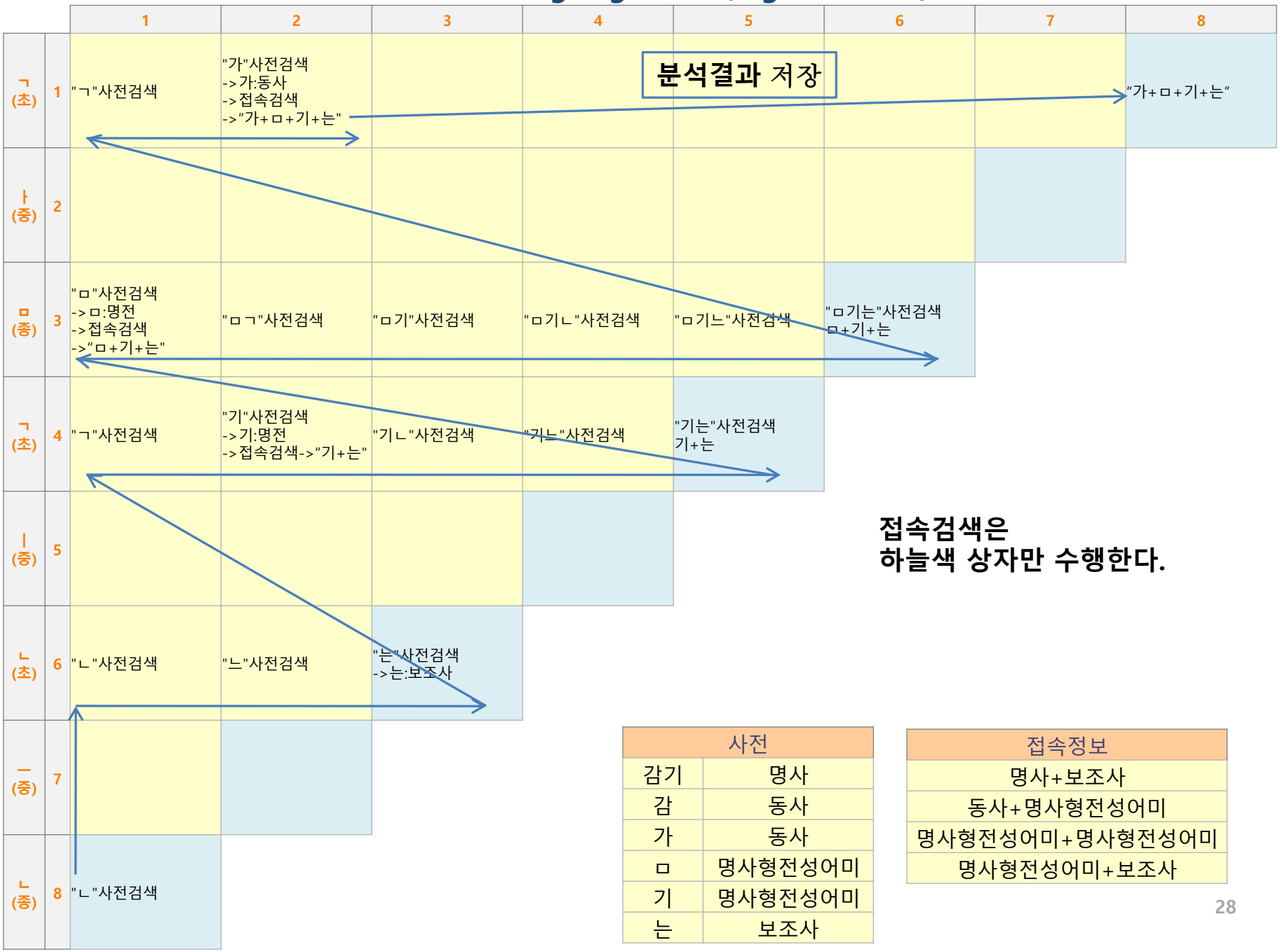


접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)



접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Tabular Parsing Algorithm(Right -> Left)

		1	2	3	4	5	6	7	8
ㄱ (초)	1	"ㄱ"사전검색	"가"사전검색 ->가:동사 ->접속검색 ->"가+ㅁ+기+는"	"감"사전검색 ->감:동사 ->접속검색 ->"감+기+는"	"감ㄱ"사전검색	"감기"사전검색 ->감기:명사 ->접속검색 ->"감기+는"	"감기ㄴ"사전검색	"감기느"사전검색	"감기는"사전검색 가+ㅁ+기+는 감+기+는 감기+는
ㅏ (중)	2								
ㅁ (종)	3	"ㅁ"사전검색 ->ㅁ:명전 ->접속검색 ->"ㅁ+기+는"	"ㅁㄱ"사전검색	"ㅁ기"사전검색	"ㅁ기ㄴ"사전검색	"ㅁ기느"사전검색	"ㅁ기는"사전검색 ㅁ+기+는		
ㄱ (초)	4	"ㄱ"사전검색	"기"사전검색 ->기:명전 ->접속검색->"기+는"	"기ㄴ"사전검색	"기느"사전검색	"기는"사전검색 기+는			
ㅣ (중)	5								
ㄴ (초)	6	"ㄴ"사전검색	"느"사전검색	"는"사전검색 ->는:보조사					
ㅡ (중)	7								
ㄴ (종)	8	"ㄴ"사전검색							

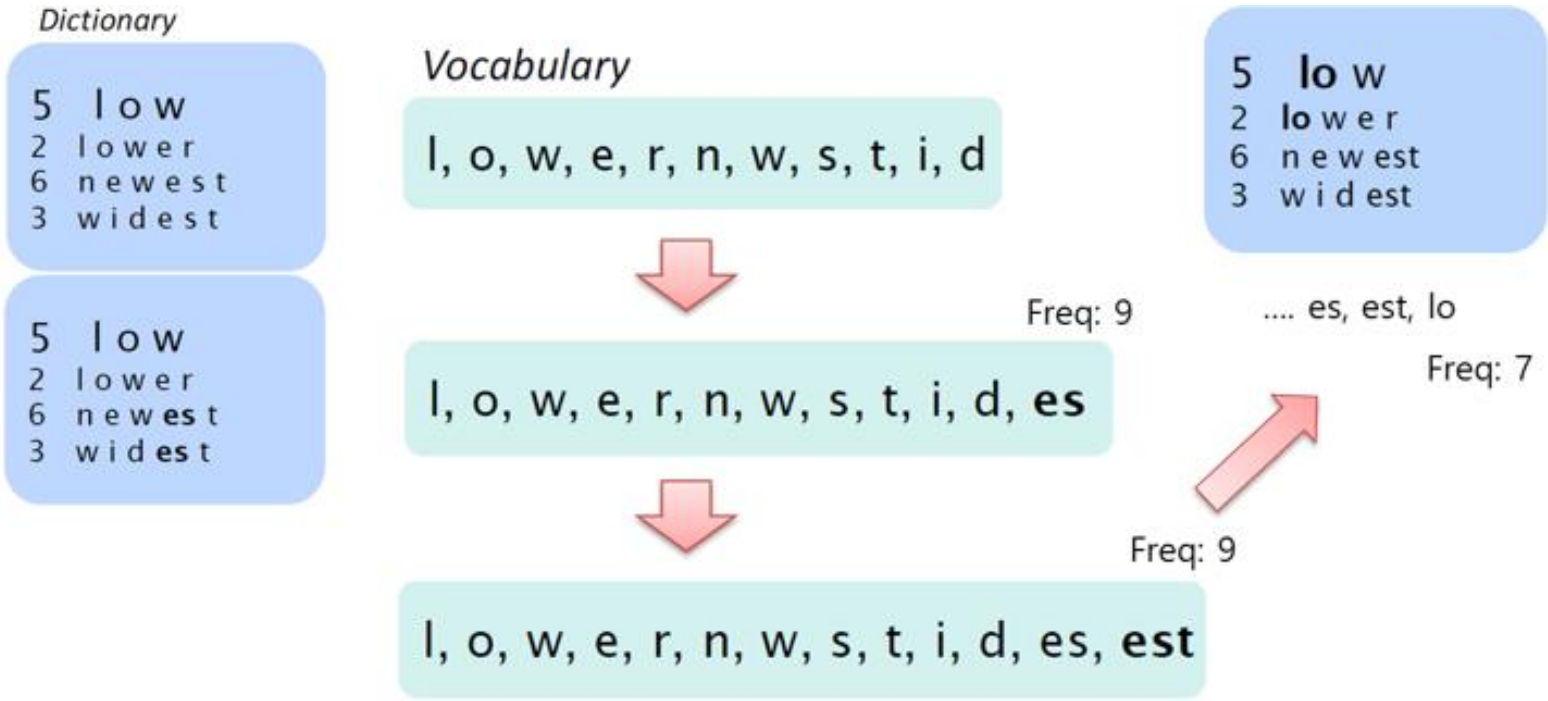
접속검색은
하늘색 상자만 수행한다.

사전	
감기	명사
감	동사
가	동사
ㅁ	명사형전성어미
기	명사형전성어미
는	보조사

접속정보
명사+보조사
동사+명사형전성어미
명사형전성어미+명사형전성어미
명사형전성어미+보조사

Byte Pair Encoding

- 워드피스(WordPiece) 단위: Byte Pair Encoding



확인 문제: Byte Pair Encoding

다음과 같은 문자열이 주어졌을 때 7개의 subunit이 될 때까지 BPE를 수행하시오.

5 l o w
2 l o w e r
6 n e w e s t
3 w i d e s t

?

?

?

?

?

?

KU KONKUK UNIVERSITY

KAIST

Sequence Labeling

HMM (Hidden Markov Model)

- What is the HMM?
  - Hidden + Markov Model
- Markov Model: Example
  - Training

		Tomorrow State (내일 상태)		
		Rainy	Cloudy	Sunny
Today State (오늘 상태)	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8

  - Assumption: Tomorrow weather depends only on today one.
  - Problem: P(Rainy, Rainy, Sunny, Cloudy)?

마코프 모델 (Markov Model)

- Markov Model: Sequence Probability
  - 결합 확률(joint probability) 계산 모델
  - 연쇄 규칙과 마코프 가정을 이용하여 결합 확률을 단순화하여 결합 확률을 근사화시키는 모델

$$\mathrm{P}(y_1, y_2, \ldots, y_t)$$

$$= \mathrm{P}(y_1)P(y_2|y_1)P(y_3|y_1, y_2) \ldots P(y_t|y_1, y_2, \ldots, , y_{t-1})$$

Chain Rule

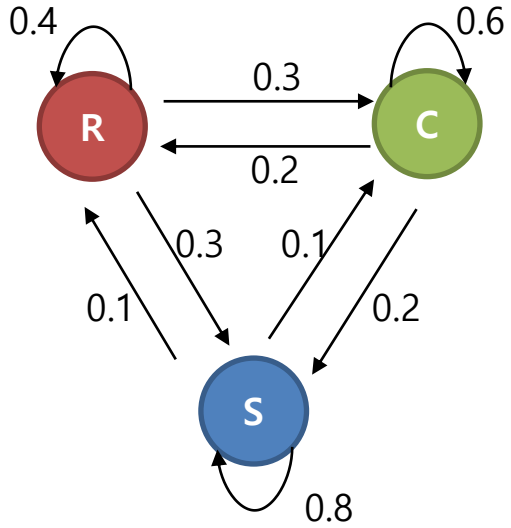
$$= \mathrm{P}(y_1)P(y_2|y_1)P(y_3|y_2) \ldots P(y_t|y_{t-1})$$

1'st-order Markov Assumption

마코프 모델 (Markov Model)

마코프 모델: 학습데이터로 부터 얻어진 전이 확률분포 (상태 간 이동 확률 분포)

		Tomorrow State (내일 상태)		
		Rainy	Cloudy	Sunny
Today State (오늘 상태)	Rainy	0.4	0.3	0.3
	Cloudy	0.2	0.6	0.2
	Sunny	0.1	0.1	0.8



State = $\{S_1: Rainy, S_2: Cloudy, S_3: Sunny\}$

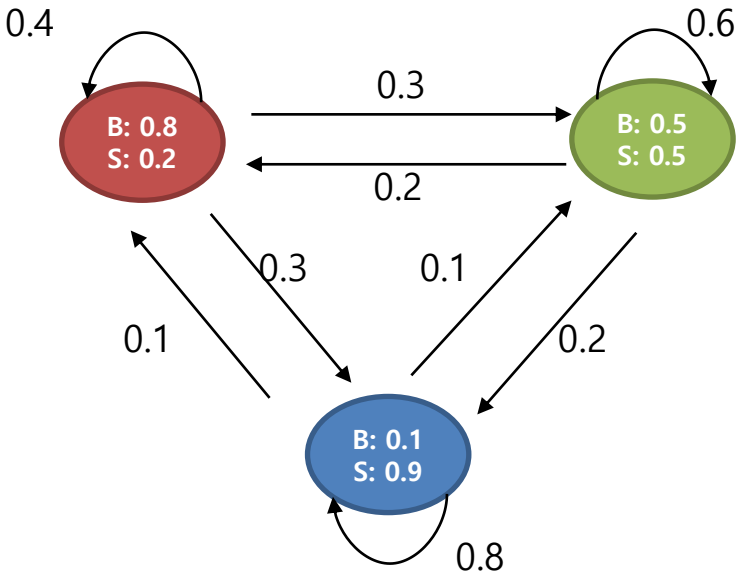
$P(S_1, S_1, S_3, S_2 | model)$
$= P(S_1)P(S_1|S_1)P(S_3|S_1)\ P(S_2|S_3)$
=1 * 0.4 * 0.3 * 0.1
=0.012

What is Hidden?

- Hidden Markov Model
  - 상태(풀고자 하는 레이블)를 직접 관측할 수 없고 상태를 예측하는데 도움이 되는 특징(자질)만을 관측할 수 있음
  - 상태가 감춰져 있고(직접 관찰할 수 없고) 관측에 대한 확률로만 존재
  - 예제
    - 상태(State): Rainy, Cloudy, Sunny
    - 관측(Observation): B (Rain Boots), S (Sports Shoes)
    - 문제(problem): P(B, B, S, S, Rainy, Rainy, Sunny, Cloudy)?

What is Hidden?

- 상태(State): Rainy, Cloudy, Sunny
- 관측(Observation): B (Rain Boots), S (Sports Shoes)
- 문제(problem): "B, B, S, S"를 관측했을 때 날씨를 어떻게 예측하는 게 최적일까?
  - P(Rainy, Rainy, Sunny, Cloudy) vs. P(Rainy, Rainy, Sunny, Sunny) vs. …



HMM (Hidden Markov Model)

$$\text{Let } \mathrm{P}(x_{1,t}) = P(x_1, x_2, \dots, x_t)$$

$$\mathrm{HMM} = \underset{y_{1,t}}{\operatorname{argmax}}\, \mathrm{P}(x_{1,t}, y_{1,t})$$

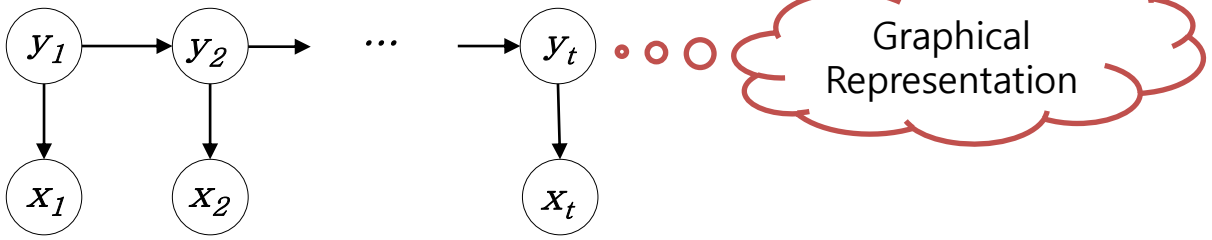
Chain Rule

$$= \underset{y_{1,t}}{\operatorname{argmax}} \mathrm{P}(y_{1,t})\mathrm{P}(x_{1,t}|y_{1,t})$$

$$= \underset{y_{1,t}}{\operatorname{argmax}} \prod_{i=1}^{t} \mathrm{P}(y_i|y_{i-1})\mathrm{P}(x_i|y_i)$$

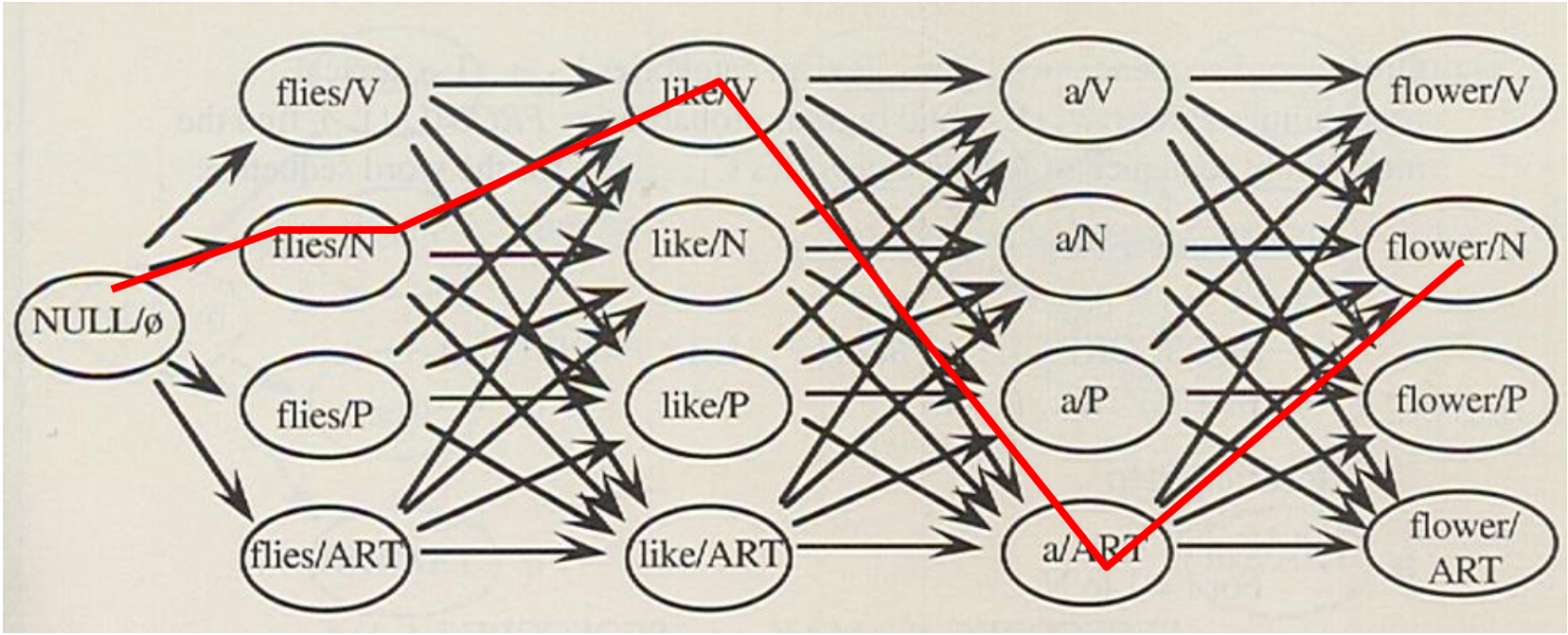
1'st-order Markov Assumption
→ 전이 확률 (Transition Probability)

Independent Assumption
→ 관측 확률 (Observation Probability)



Sequence Labeling Problem

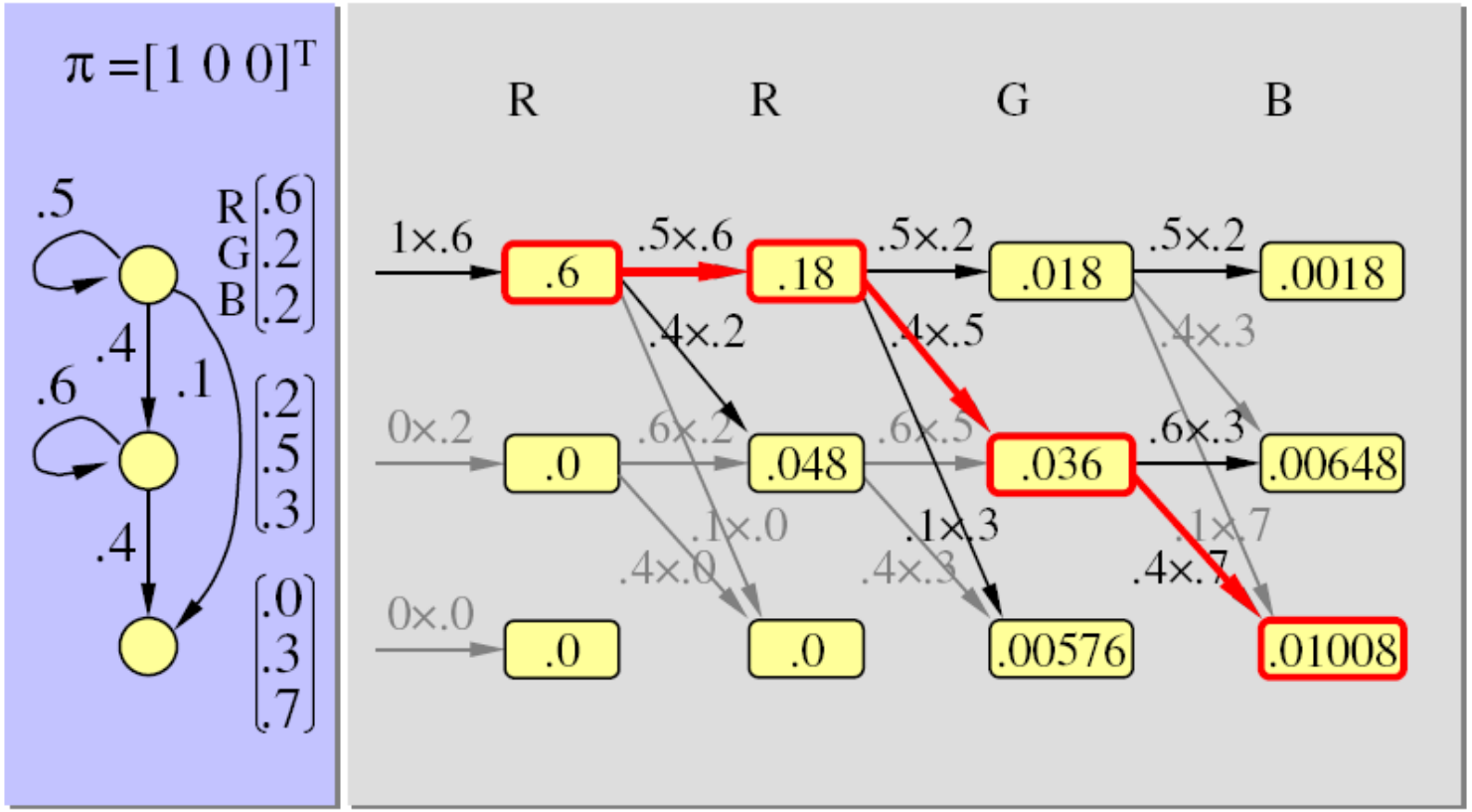
- Segmentation or path analysis problem
  - Application: Part-of-speech tagging



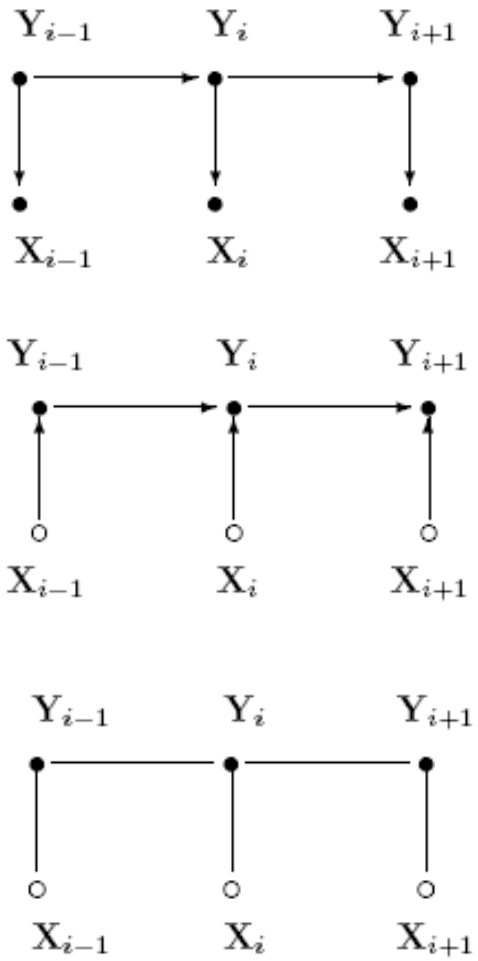
Viterbi Algorithm

- 모든 경로를 고려하지 않고도 빠른 시간 내에 최적의 경로를 찾는 알고리즘

Output: 1, 1, 2, 3



Graphical Models for Sequence Labeling



Relaxation of the Independence assumption (Avoid observation bias problem)

Avoid the label bias problem

HMM: 결합확률 모델

$$p(x,y) = \frac{p(x \mid y)p(y)}{\sum_y p(x \mid y)p(y)}$$

1st order Markov Model

$$\hat{y} = \arg\max_{y \in S} \prod_{t=1}^{T} p(x_t \mid y_t)p(y_t \mid y_{t-1})$$

MEMM: 방향성 조건부 확률 모델

Feature function

$$p(y \mid x) = \frac{1}{Z(x)} \exp\left[\sum_{i=1}^{k} \lambda_i f_i(x,y)\right]$$

$$Z(x) = \sum_y \exp\left[\sum_{i=1}^{k} \lambda_i f_i(x,y)\right]$$

CRFs: 무방향성 조건부 확률 모델

$$F_j(\boldsymbol{y},\boldsymbol{x}) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, \boldsymbol{x}, i)$$

Global feature function

$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\lambda}) = \frac{1}{Z(\boldsymbol{x})} \exp\left(\sum_j \lambda_j F_j(\boldsymbol{y},\boldsymbol{x})\right)$$

Global normalization

Uniform Model

- Two possible distributions that Aldo enjoys sports are:

	Yes	No	
Sunny	1/10	3/10	
Cloudy	1/10	1/10	
Rainy	3/10	1/10	
Total	우리가 계산하고 싶은 것		1.0

우리가 아는 것

	Yes	No	
Sunny	1/6	1/6	
Cloudy	1/6	1/6	
Rainy	1/6	1/6	
Total			1.0

- Intuitively, we think the right one is better
  - Since the probability distribution is more uniform than the left one

Uniform Model

- Now we re-examine the data and find that Aldo enjoys sports 70% of the days
- So, we can add a new constraint to the model:
  - $P_{SY} + P_{CY} + P_{RY} = 0.7$

	Yes	No	
Sunny	7/30	1/20	
Cloudy	7/30	1/20	
Rainy	7/30	1/5	
Total	0.7		1.0

	Yes	No	
Sunny	7/30	1/10	
Cloudy	7/30	1/10	
Rainy	7/30	1/10	
Total	0.7		1.0

- Again, which one is better this time?
  - The right one

Uniform Model

- Now we find that 50% of the days are Sunny
- So, we can add a new constraint to the model:
  - $P_{SY} + P_{SN} = 0.5$

- What is the best distribution this time?

	Yes	No	
Sunny	?	?	0.5
Cloudy	?	?	
Rainy	?	?	
Total	0.7		1.0

- Two questions:
  - What does uniform mean?
  - How can we find the most uniform model subject to a set of constraints like the above example?

Maximum Entropy Model

- We can approach the modeling problem from an entirely different point of view. *Begin* with some fixed feature expectations:

$$\sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}) = \alpha_i$$

- Assuming expectations are consistent, there may exist many distributions which satisfy them. Which one should we select? The most uncertain or flexible one:
  i.e. the one with *maximum entropy*.
- This yields a new optimization problem:

Objective Function

$$\max \ \mathcal{H}[p(\mathbf{x})] = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x})$$

$$\text{subject to} \ \sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}) = \alpha_i$$

$$\sum_{\mathbf{x}} p(\mathbf{x}) = 1$$

Constraints

★ 주어진 제약하에서 목적함수의 최대값을 구하는 문제 ??

Maximum Entropy Model

- To solve the maxent problem, we use Lagrange multipliers:

$$L = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x}) - \sum_i \theta_i \left(\sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}) - \alpha_i \right) - \mu \left(\sum_{\mathbf{x}} p(\mathbf{x}) - 1 \right)$$

$$\frac{\partial L}{\partial p(\mathbf{x})} = 1 + \log p(\mathbf{x}) - \sum_i \theta_i f_i(\mathbf{x}) - \mu$$

최대값 = 정점
→ x에 대해서 편미분하였 때 기울기가 0인 곳!

$$p^*(\mathbf{x}) = e^{\mu - 1} \exp \left\{ \sum_i \theta_i f_i(\mathbf{x}) \right\}$$

$$Z(\theta) = e^{1-\mu} = \sum_{\mathbf{x}} \exp \left\{ \sum_i \theta_i f_i(\mathbf{x}) \right\}$$

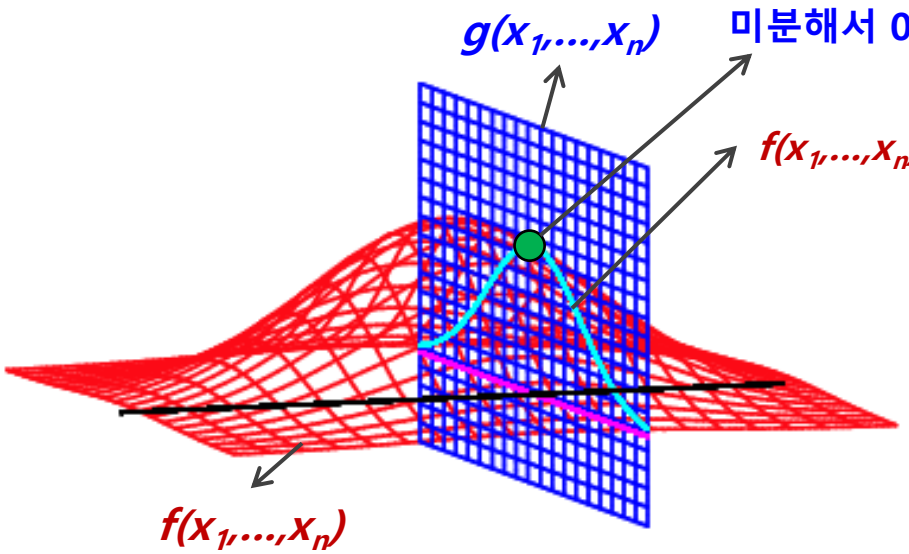
모든 x에 대한 P*(x)=1

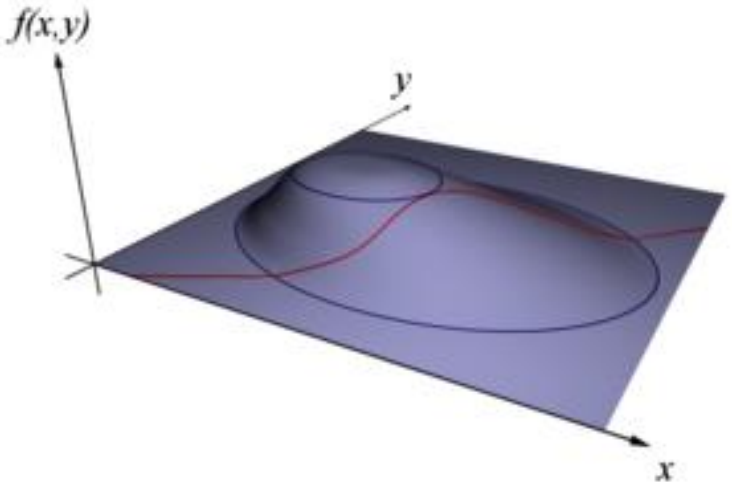
$$p(\mathbf{x}|\theta) = \frac{1}{Z(\theta)} \exp \left\{ \sum_i \theta_i f_i(\mathbf{x}) \right\}$$

- So feature constraints + maxent implies exponential family.
- Problem is convex, so solution is unique.

Maximum Entropy Model

- Can be used to find the extremum (maximum, minimum) of a multivariate function $f(x_1,\ldots,x_n)$ subject to the constraint $g(x_1,\ldots,x_n)=0$





Maximum Entropy Model

- Constrained Optimization Problem
  - Finding a set of parameters $\lambda=\{\lambda_1,\ldots,\lambda_k\}$ on an exponential model which maximizes its log likelihood
  - There exists many algorithms
    - ✓ Generalized Iterative Scaling (Darroch and Ratcliff, 1972)
    - ✓ Improved Iterative Scaling (Della Pietra et al., 1997)
    - ✓ L-BFGS

우리가 계산하고 싶은 것

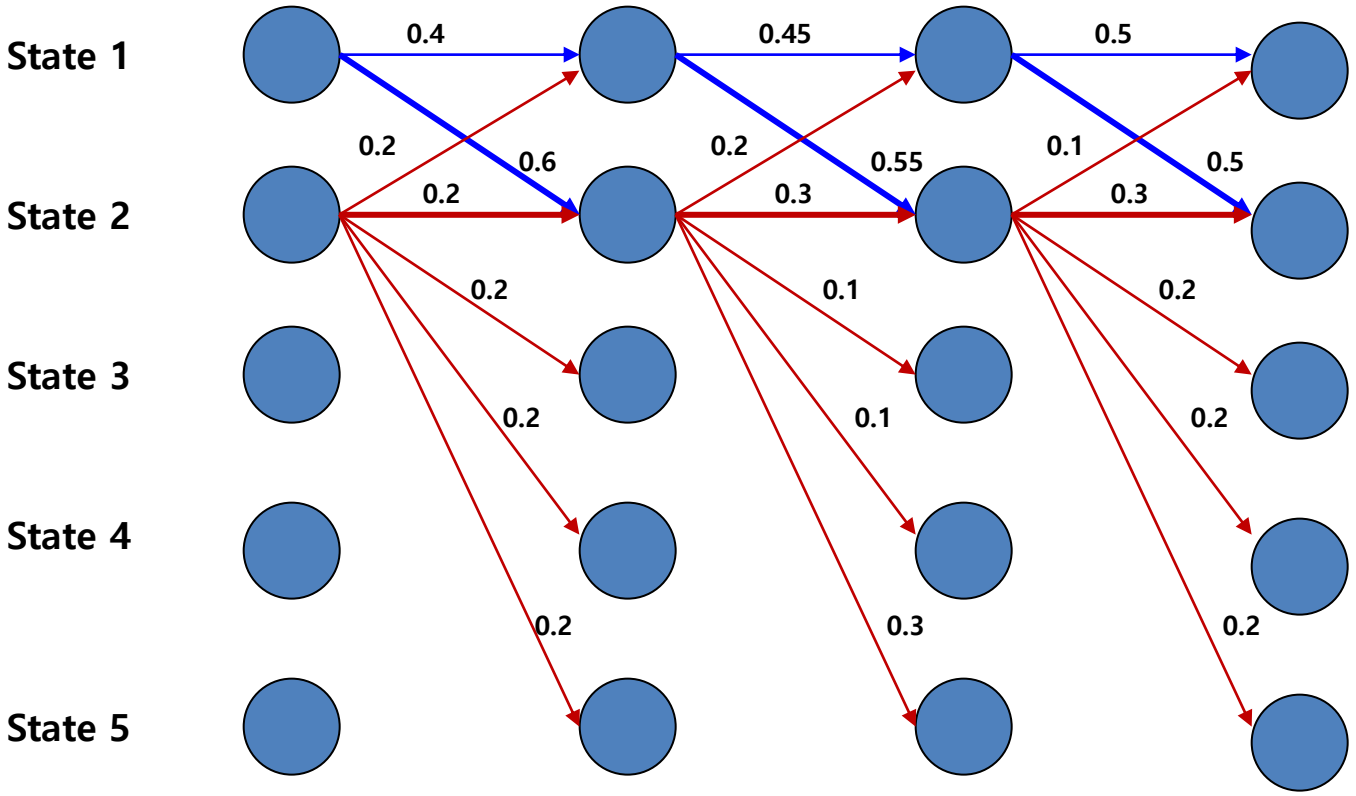
$$p(y \mid x) = \frac{1}{Z(x)} \exp\left[\sum_{i=1}^{k} \lambda_i f_i(x, y)\right]$$

우리가 아는 것

$$Z(x) = \sum_{y} \exp\left[\sum_{i=1}^{k} \lambda_i f_i(x, y)\right]$$

$|\Omega|$

Label Bias Problem of MEMM



WHY?

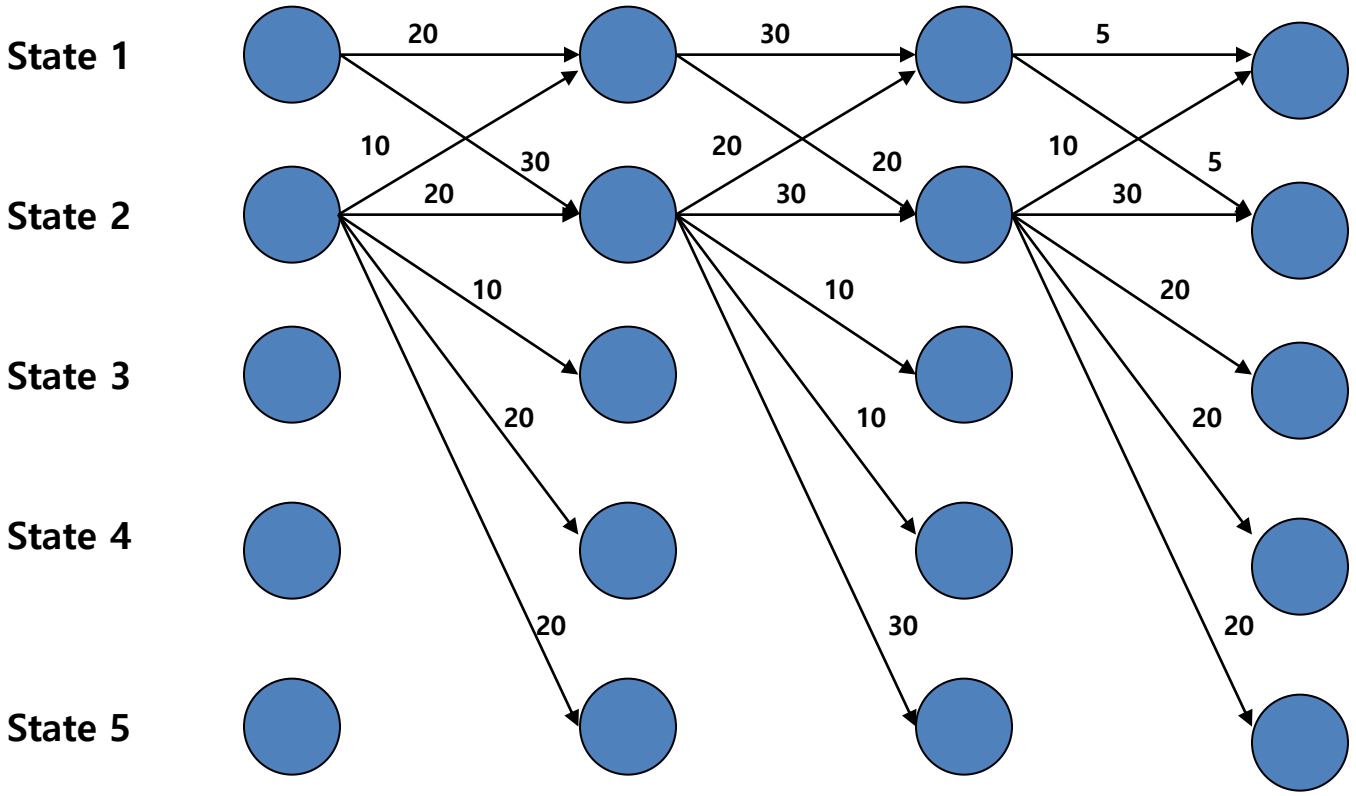
What the local transition probabilities say:

State 1 almost always prefers to go to state 2

State 2 almost always prefer to stay in state 2

1 → 2 → 1 → 2: 0.06
2 → 2 → 2 → 2: 0.018
1 → 1 → 1 → 1: 0.09

Global Normalization of CRFs



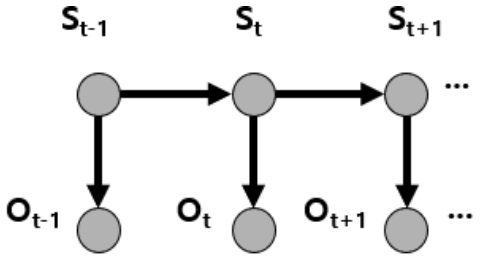
Local normalization: 전이가 많은 상태는 항상 불리

→ Global normalization (모든 상태를 기준으로 정규화)

Graphical Models

HMM

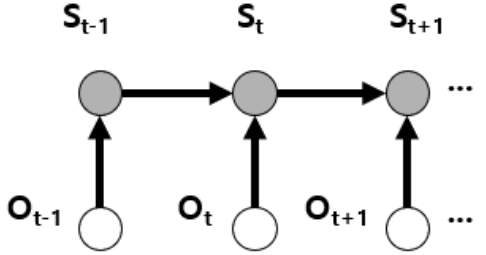
$$P(\vec{s},\vec{o}) \propto \prod_{t=1}^{|\vec{o}|} P(s_t \mid s_{t-1}) P(o_t \mid s_t)$$



MEMM

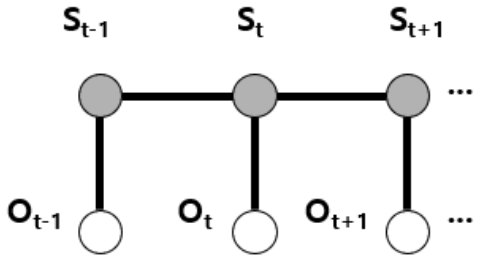
$$P(\vec{s} \mid \vec{o}) \propto \prod_{t=1}^{|\vec{o}|} P(s_t \mid s_{t-1}, o_t)$$

$$\propto \prod_{t=1}^{|\vec{o}|} \frac{1}{Z_{s_{t-1},o_t}} \exp\left(\sum_j \lambda_j f_j(s_t, s_{t-1}) + \sum_k \mu_k g_k(s_t, x_t) \right)$$



CRF

$$P(\vec{s} \mid \vec{o}) \propto \frac{1}{Z_{\vec{o}}} \prod_{t=1}^{|\vec{o}|} \exp\left(\sum_j \lambda_j f_j(s_t, s_{t-1}) + \sum_k \mu_k g_k(s_t, x_t) \right)$$

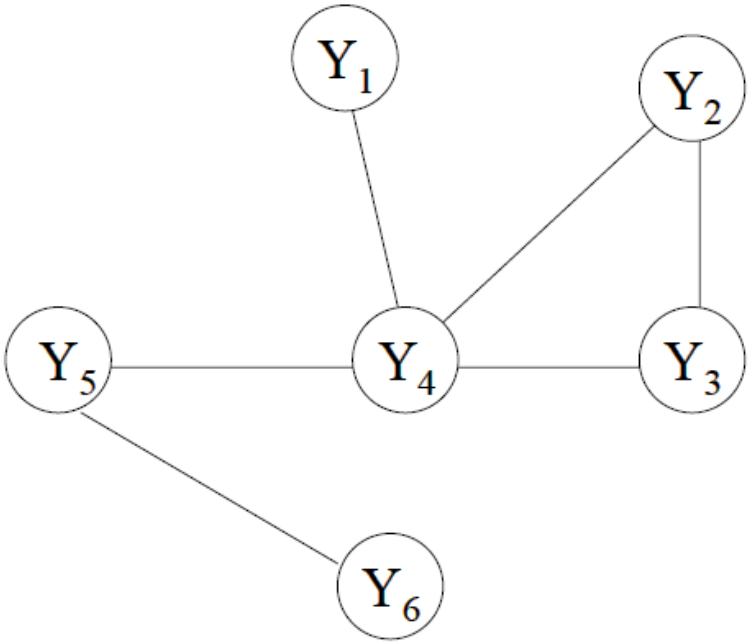


Random Fields

Let G = (Y, E) be a graph where each vertex Y_v is a random variable

Suppose $P(Y_v \mid \text{all other } Y) = P(Y_v \mid \text{neighbors}(Y_v))$ then Y is a random field

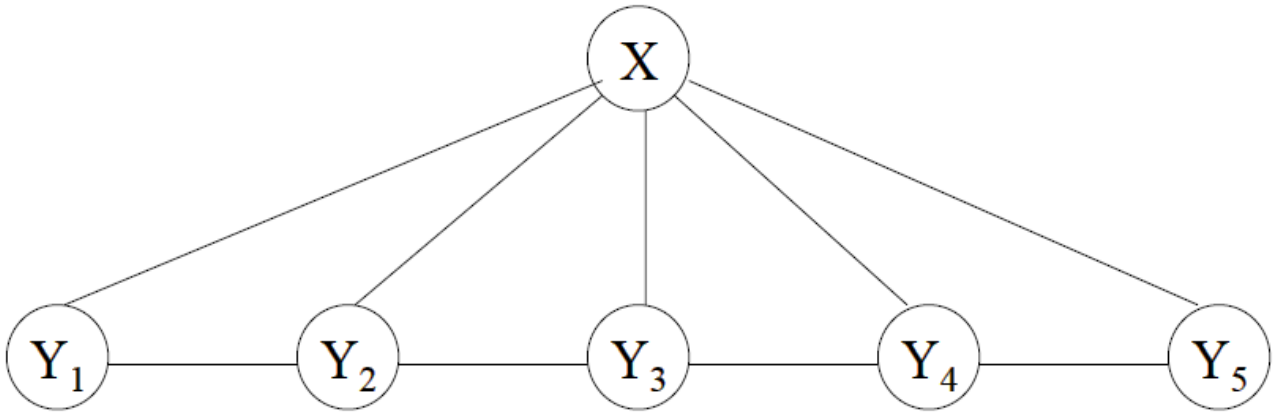
Example:



- $P(Y_5 \mid \text{all other } Y) = P(Y_5 \mid Y_4, Y_6)$

Conditional Random Fields

Suppose $P(Y_v \mid X$, all other $Y) = P(Y_v \mid X, \text{neighbors}(Y_v))$
then X with Y is a **conditional** random field



- $P(Y_3 \mid X$, all other $Y) = P(Y_3 \mid X, Y_2, Y_4)$
- Think of X as observations and Y as labels

Empirical Results of Graphical Models

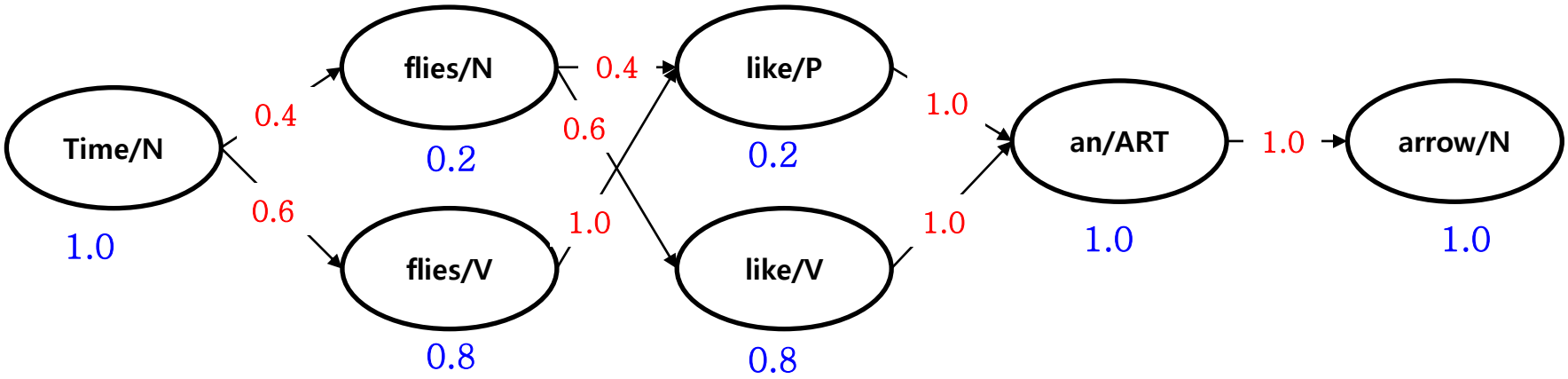
- Part-Of-Speech Tagging
  - Using same set of features: HMM >=< CRF > MEMM
  - Using additional overlapping features: CRF+ > MEMM+ >> HMM

model	*error*	*oov error*
HMM	5.69%	45.99%
MEMM	6.37%	54.61%
CRF	5.55%	48.05%
MEMM+	4.81%	26.99%
CRF+	4.27%	23.76%

+Using spelling features

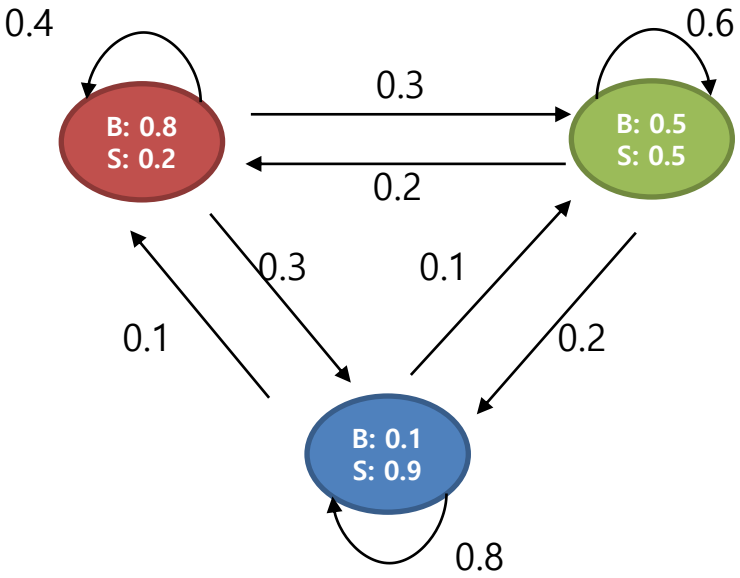
확인 문제

- 다음 HMM을 바탕으로 "Time files like an arrow"라는 문장의 품사를 결정하는 최적의 경로를 비터비 알고리즘을 이용하여 구하시오.



HMM 만들기

- 상태(State): Rainy, Cloudy, Sunny
- 관측(Observation): B (Boots), S (Shoes)
- 문제(problem): "B, B, S, S"를 관측했을 때 날씨를 어떻게 예측하는 것이 최적일까?



$$\pi = \{0.2, 0.5, 0.3\}$$

HMM 만들기

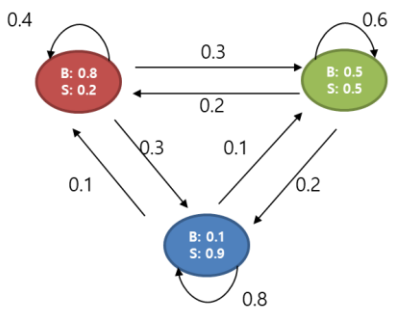
구글 colab 연결

```
from google.colab import drive
drive.mount("/gdrive", force_remount=True)
```

라이브러리 설치

```
[7]  !pip install hmmlearn
```

```
Collecting hmmlearn
  Downloading hmmlearn-0.2.6-cp37-cp37m-manylinux_2_5_x86_64.manylin
     |████████████████████████████████| 374 kB 5.2 MB/s
Requirement already satisfied: scipy>=0.19 in /usr/local/lib/python3
Requirement already satisfied: scikit-learn>=0.16 in /usr/local/lib/
Requirement already satisfied: numpy>=1.10 in /usr/local/lib/python3
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python
Installing collected packages: hmmlearn
Successfully installed hmmlearn-0.2.6
```



$\pi = \{0.2, 0.5, 0.3\}$

확률 설정

```
import numpy as np
from hmmlearn import hmm

states = ["Rainy", "Cloudy", "Sunny"]
n_states = len(states)

observations = ["Boots", "Shoes"]
n_observations = len(observations)

# 시작 확률
start_probability = np.array([0.2, 0.5, 0.3])

# 전이 확률
transition_probability = np.array([
  [0.4, 0.3, 0.3],
  [0.2, 0.6, 0.2],
  [0.1, 0.1, 0.8]
])

# 관측 확률
emission_probability = np.array([
  [0.8, 0.2],
  [0.5, 0.5],
  [0.1, 0.9]
])
```

HMM 만들기

모델 구성 및 디코딩

```
# 모델 만들기
model = hmm.MultinomialHMM(n_components=n_states)
model.startprob_ = start_probability
model.transmat_ = transition_probability
model.emissionprob_ = emission_probability

# 관측 입력
input = [0, 0, 1,1]

# HMM 모델 호출
hmm_input = np.atleast_2d(input).T
logprob, sequence = model.decode(hmm_input)

print("Input :", ", ".join(map(lambda x: observations[x], input)))
print("Output:", ", ".join(map(lambda x: states[x], sequence)))
print("Prob. :", logprob)
```

2차원으로 만들어서 트랜스포즈

```
Input : Boots, Boots, Shoes, Shoes
Output: Rainy, Rainy, Sunny, Sunny
Prob. : -4.609853133892472
```

RNN 응용 구조

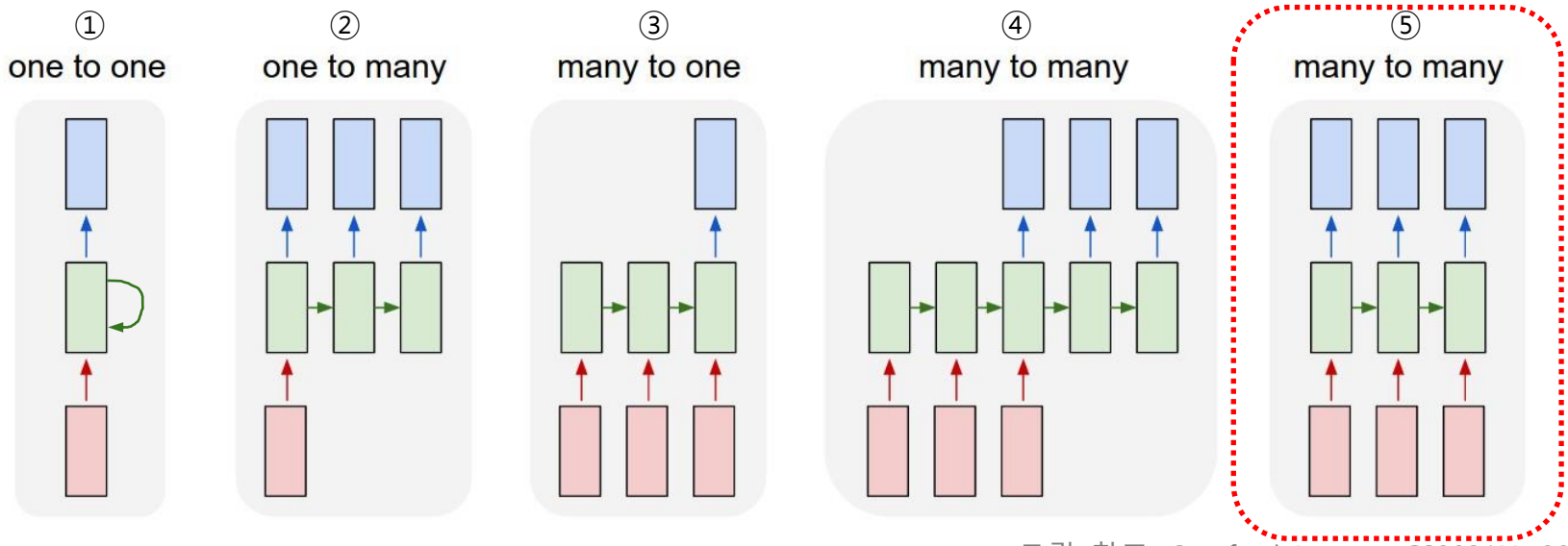
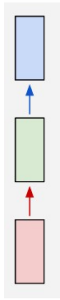


그림 참조: Stanford Lecture CS2031N, 2017
(Fei-Fei Li & Justin Johnson & Serena Yeung)



Blue : output vector

Green : RNN's state

Red : input vector

[활용 예]

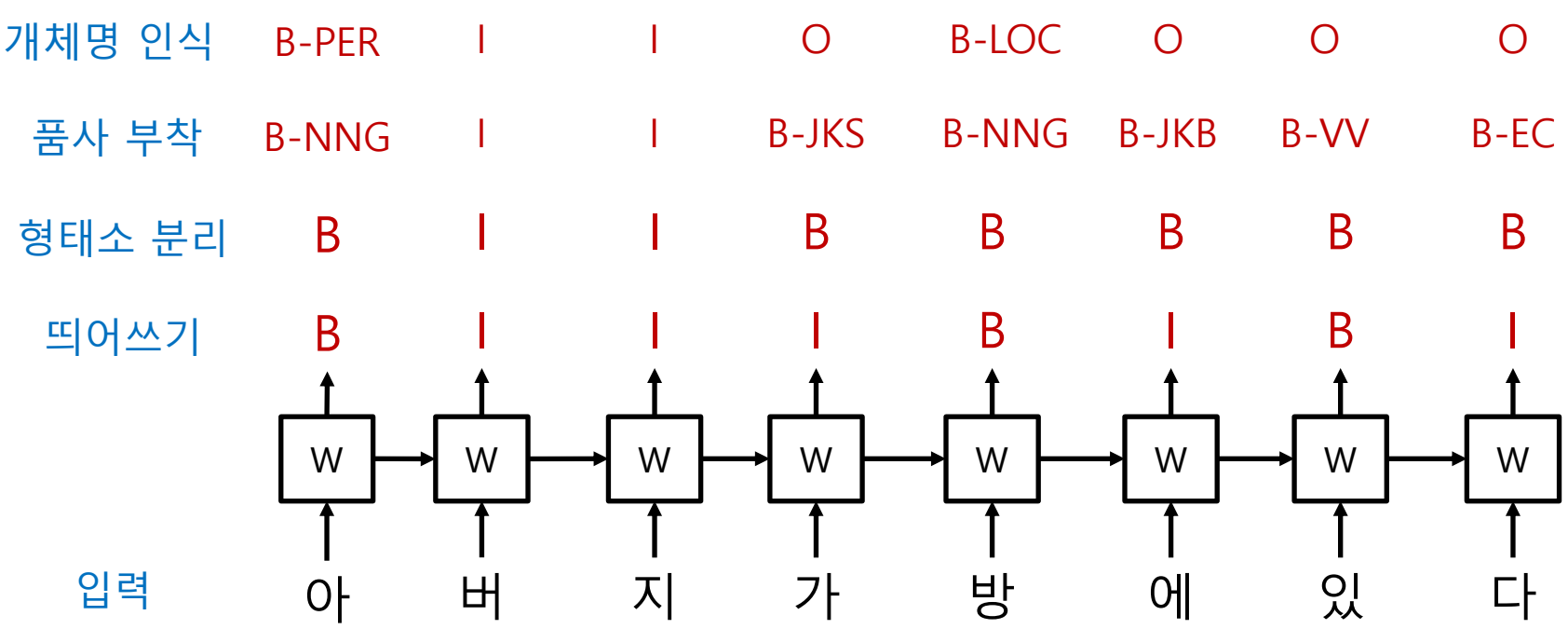
① 일반적인 NN (FNN)
② 자막 생성 (image captioning)
③ 분류 및 예측 (prediction)
④ 기계 번역
⑤ 순차 표지 부착

Input sequence → Label sequence
(자동 띄어쓰기, 개체명 인식 등)

Many-to-Many Model (Sequence Labeling)

- 순차적 레이블 부착
  - 연속된 입력에 대해 문맥을 반영하여 분류를 수행하는 것

BIO Notation for Segmentation: B(Beginner), I(Inner), O(outer)



KU KONKUK UNIVERSITY

KAIST

Transformer

Transformer Figures by Bgg's Blog.

Many-to-Many Model (Sequence-to-Sequence)

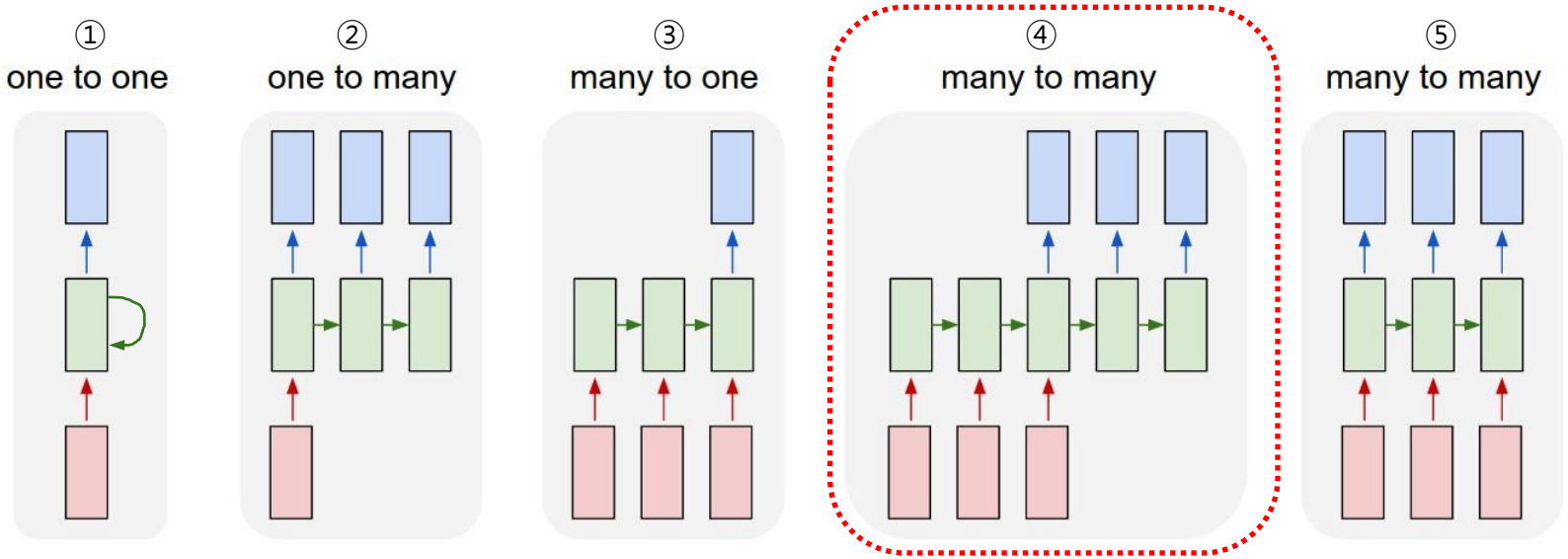
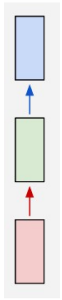


그림 참조: Stanford Lecture CS2031N, 2017
(Fei-Fei Li & Justin Johnson & Serena Yeung)



Blue : output vector

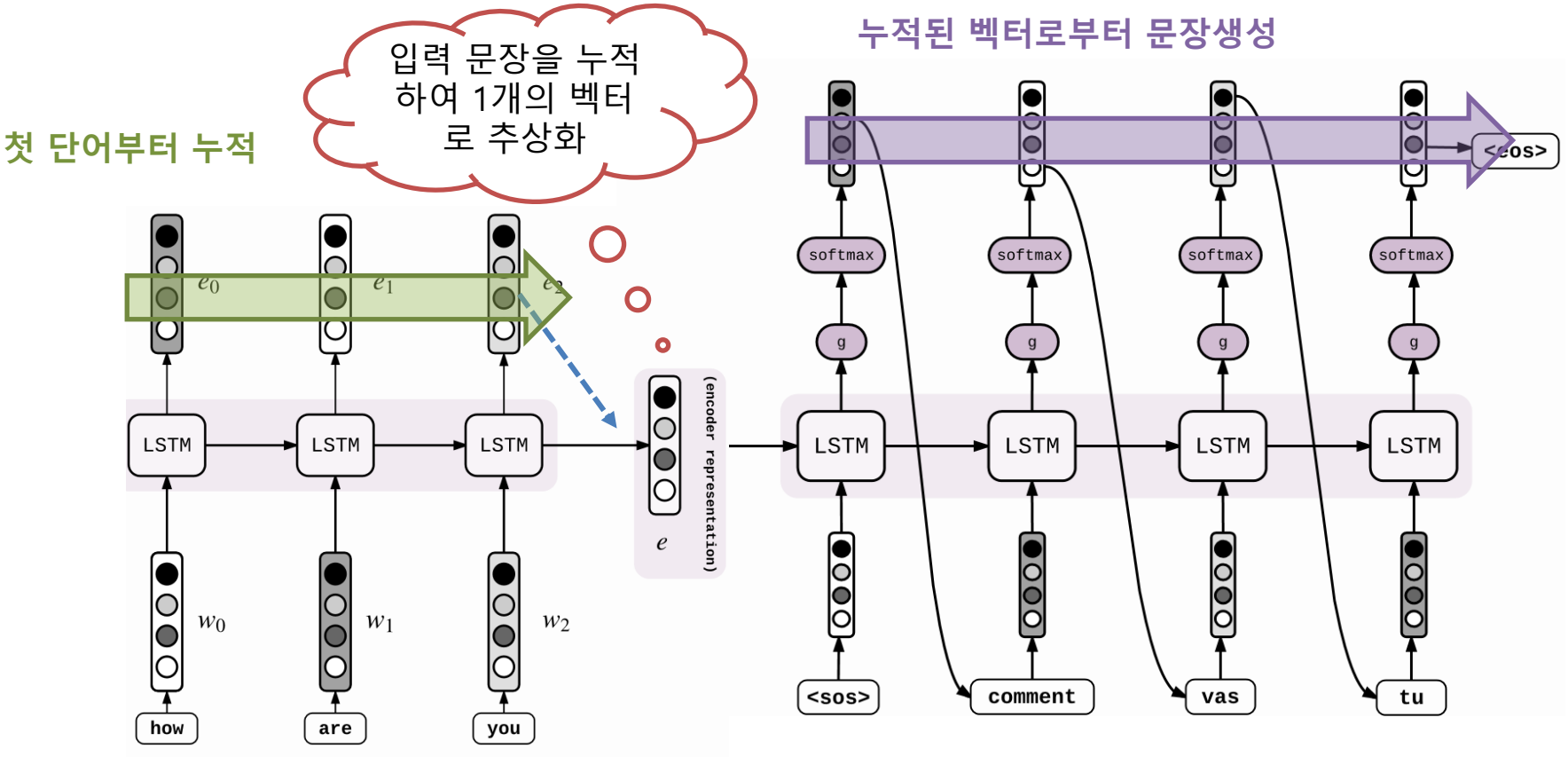
Green : RNN's state

Red : input vector

[활용 예]
① 일반적인 NN (FNN)
② 자막 생성 (image captioning)
③ 분류 및 예측 (prediction)
④ 기계 번역
⑤ 순차 표지 부착

Korean sentence (word sequence)→
English sentence (word sequence)

Seq2Seq for Machine Translation



Encoder: Many-to-One Model — Decoder: Many-to-Many Model (Language Model)

Encoder-Decoder Model (Seq2Seq Model)

그림 출처: Guillaume Genthial 발표자료

Seq2Seq with Attention

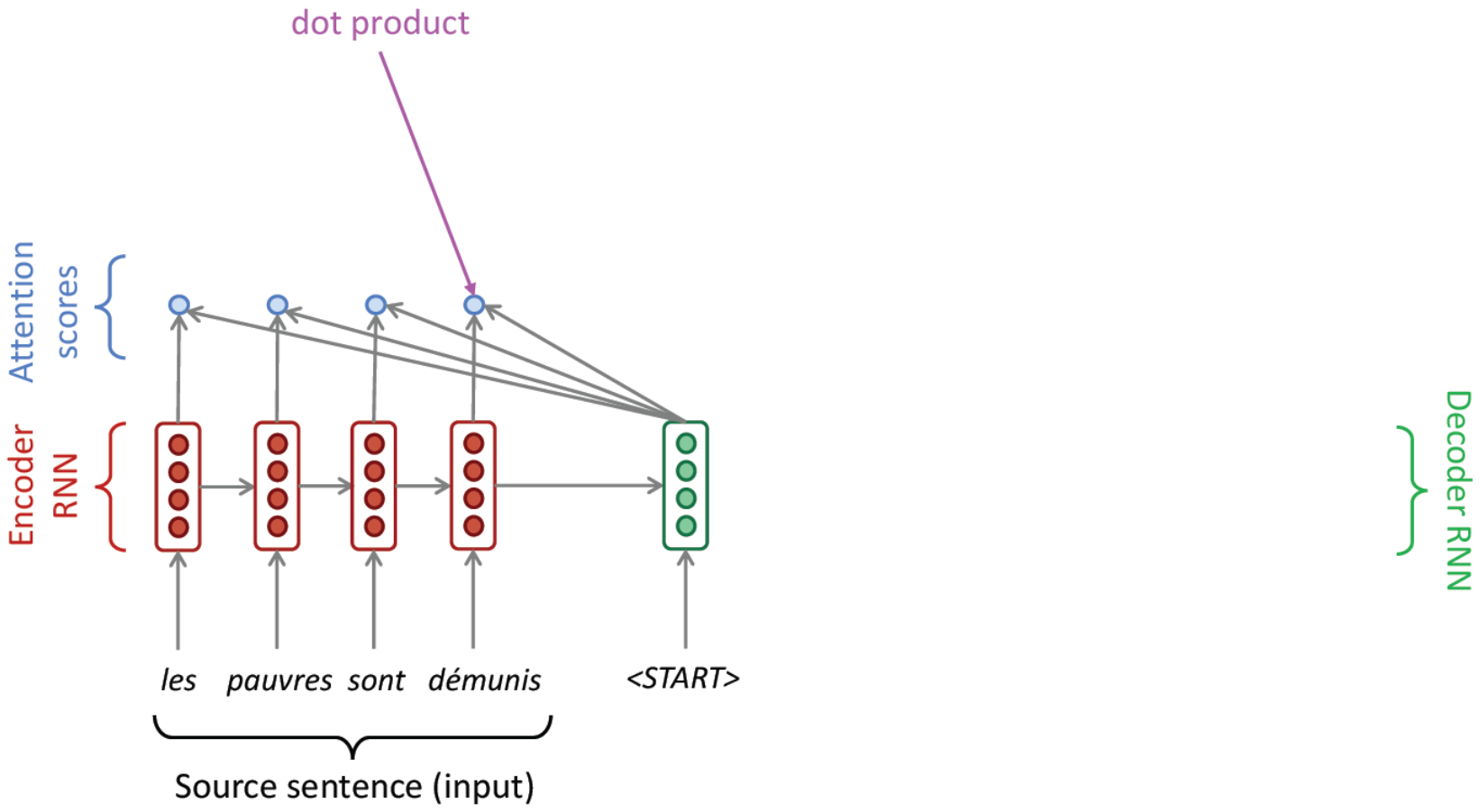
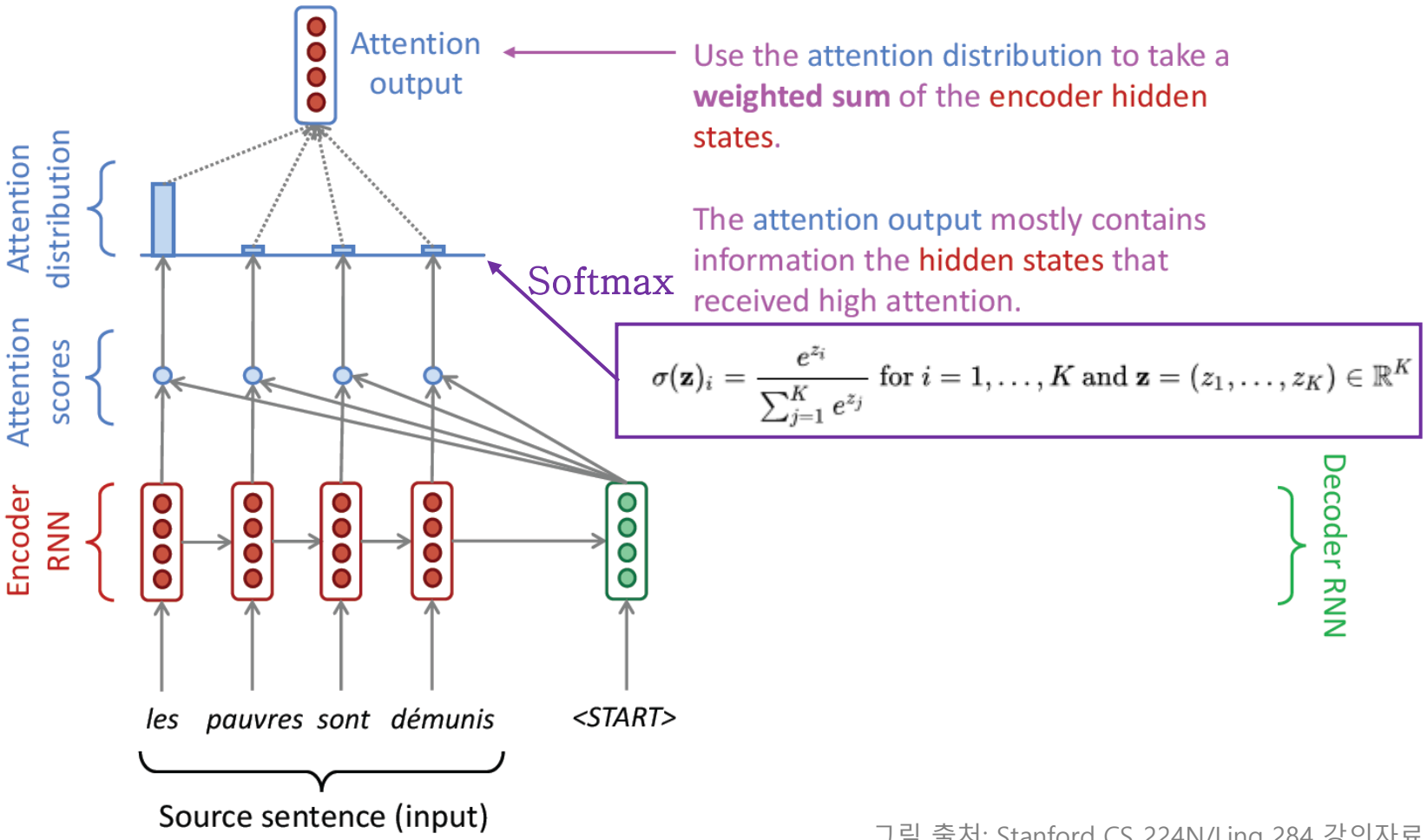


그림 출처: Stanford CS 224N/Ling 284 강의자료

Seq2Seq with Attention



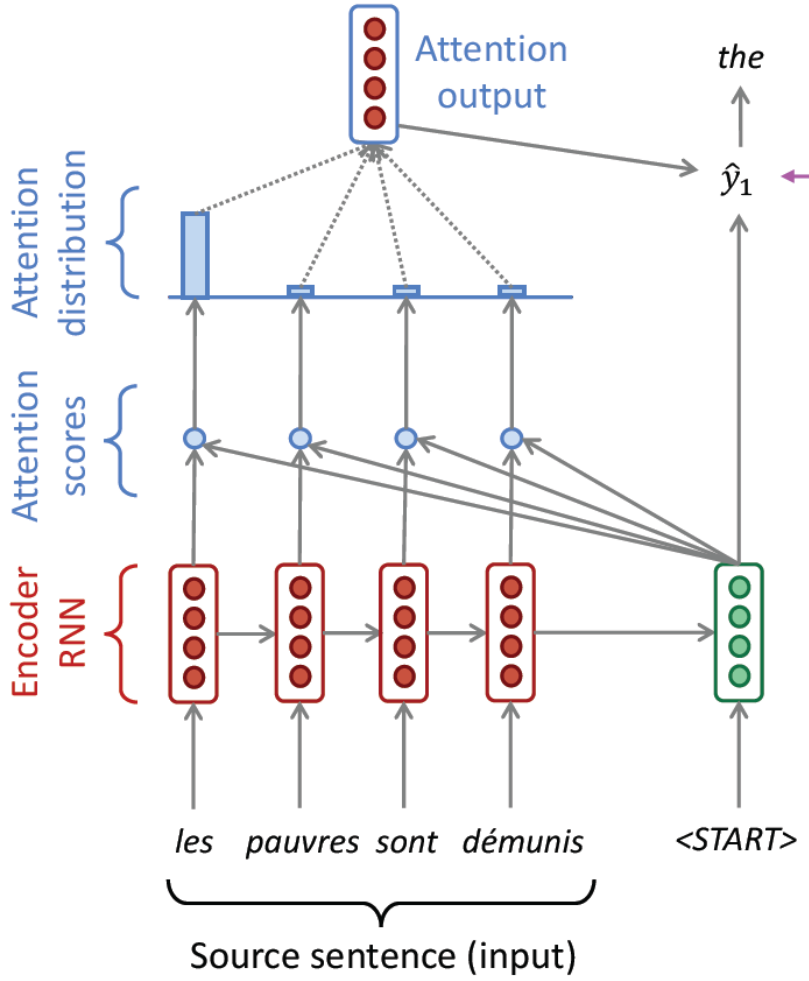
Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information the hidden states that received high attention.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

그림 출처: Stanford CS 224N/Ling 284 강의자료

Seq2Seq with Attention



Luong Attention for MT!

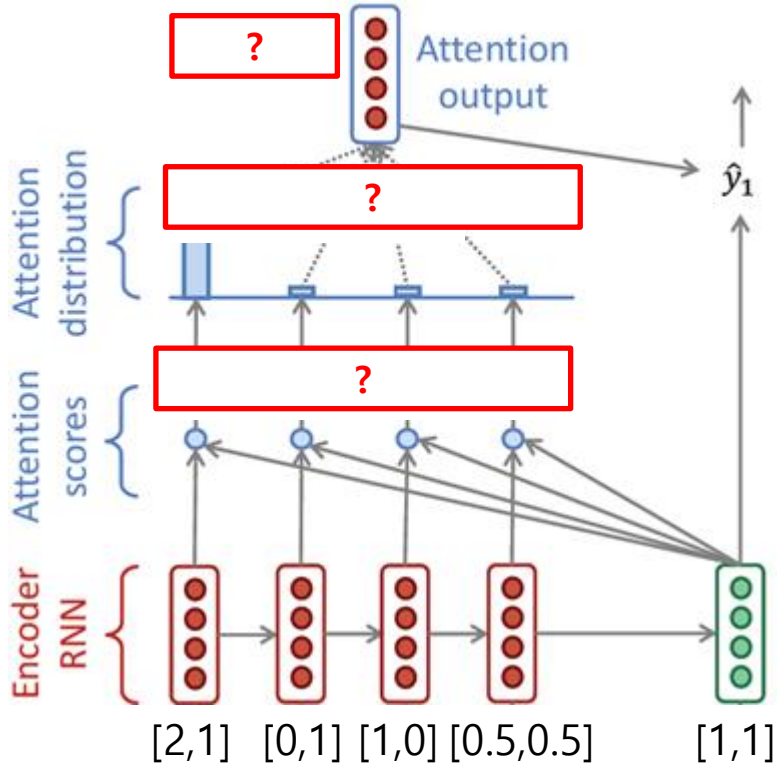
Concatenate attention output with decoder hidden state, then use to compute $\hat{y}_1$ as before

Decoder RNN

그림 출처: Stanford CS 224N/Ling 284 강의자료

확인 문제

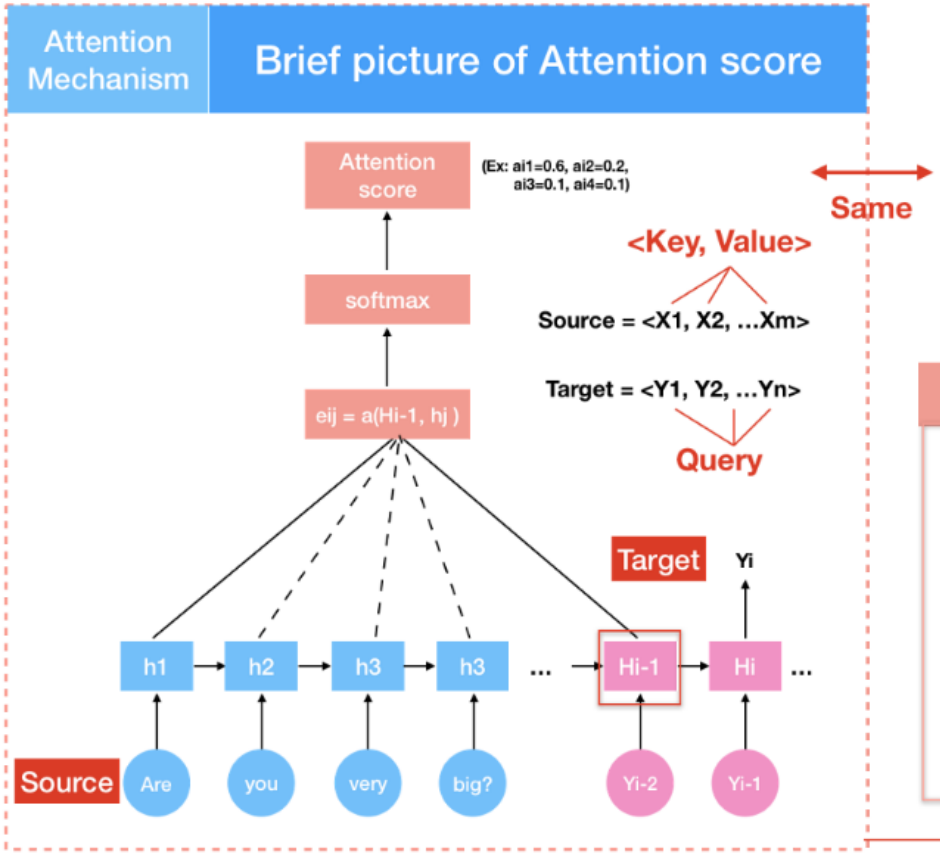
- Luong attention 수식에 기초하여 attention scores, attention distribution, attention output을 계산 하시오. (소수점 이하 2자리 반올림)



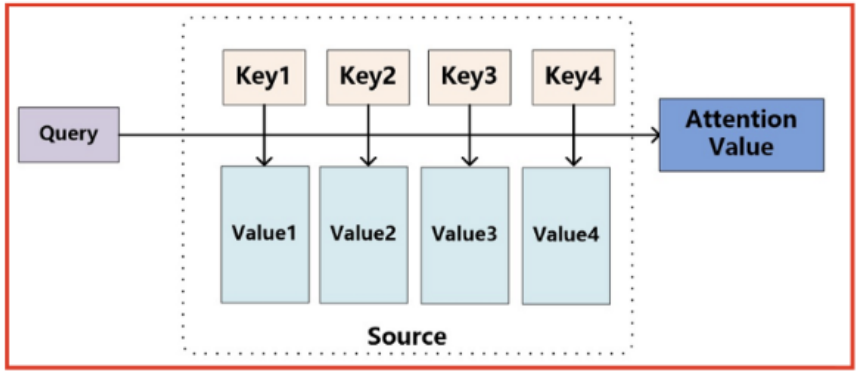
Softmax

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

Attention Models



Same



Hint

Given Query(Yi) in the target. Through calculating the attention score α of Query and each Key. We can get context vector, which is a sum of hidden states of the input sequence, weighted by attention scores α

In self attention, Key=Value.

$$\underbrace{\mathbf{Attention(Query, Source)}}_{\text{Context vector}} = \sum_{i=1}^{L_x} \underbrace{a_i}_{\text{Attention score}} \cdot \underbrace{Value_i}_{\text{Hidden state}}$$

그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Attention Models in Detail

Decoder Formula

The context vector c_i is, then, computed as a weighted sum of these annotations h_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \leftrightarrow \underbrace{\mathbf{Attention(Query, Source)}}_{\text{Attention vector}} = \underbrace{\sum_{i=1}^{L_x} a_i}_{\text{Attention score}} \cdot \underbrace{Value_i}_{\text{Hidden state}}$$ Stage 3

The weight α_{ij} of each annotation h_j is computed by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \leftrightarrow \underbrace{a_i}_{\text{Attention score}} = Softmax(Sim_i) = \frac{e^{Sim_i}}{\sum_{j=1}^{L_x} e^{Sim_j}}$$ Stage 2

where $e_{ij} = a(s_{i-1}, h_j) \leftrightarrow \mathbf{Similarity(Query, Key_i) = Query \cdot Key_i}$ *dot* Stage 1

is an *alignment model* which scores how well the inputs around position j and the output at position i match. The score is based on the RNN hidden state s_{i-1} (just before emitting y_i, Eq. (4)) and the j-th annotation h_j of the input sentence.

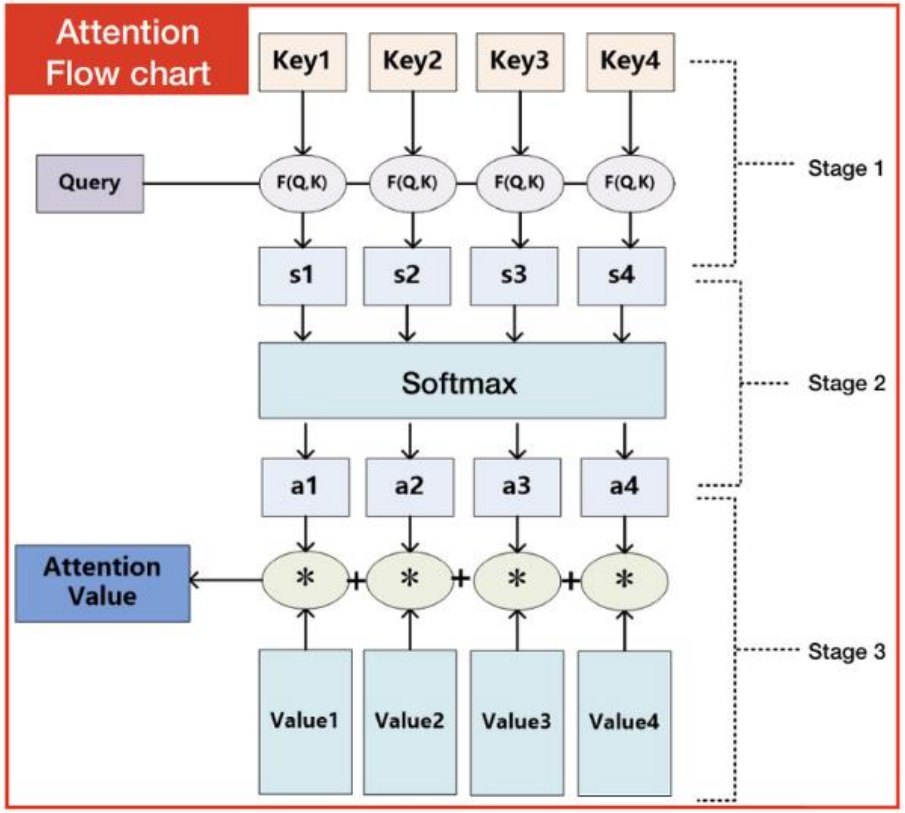
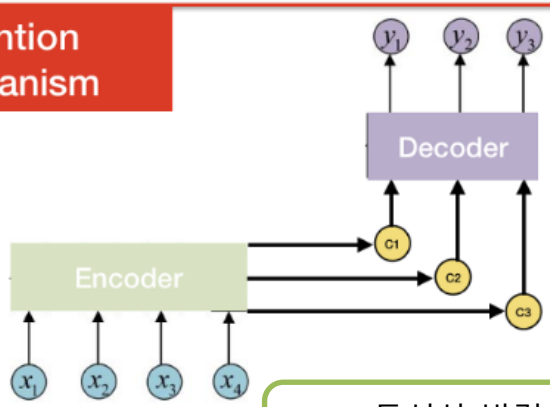


그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Problems of Attention-Based Models

Attention Mechanism



Still long dependency problem!

But

Source와 Target 사이 상호 관계만 고려!

C1、C2…Cn is calculated through the hidden state between source sentence and target sentence (h1、h2…, h1'、h2'…), leaving the attention in source sentence and target sentence itself ignored.

RNN 특성상 병렬 처리 어려움!

Mr. President, RNN is also hard to parallelize and not time-efficient. What should I do?

Self Attention

1. Why don't you take away RNN?
2. Why don't you self attention ?

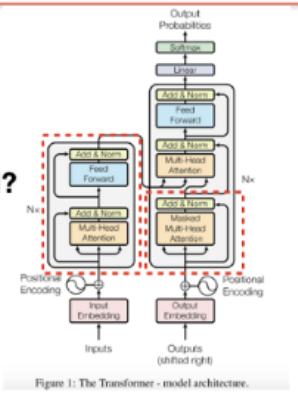


Figure 1: The Transformer - model architecture.

그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Transformer

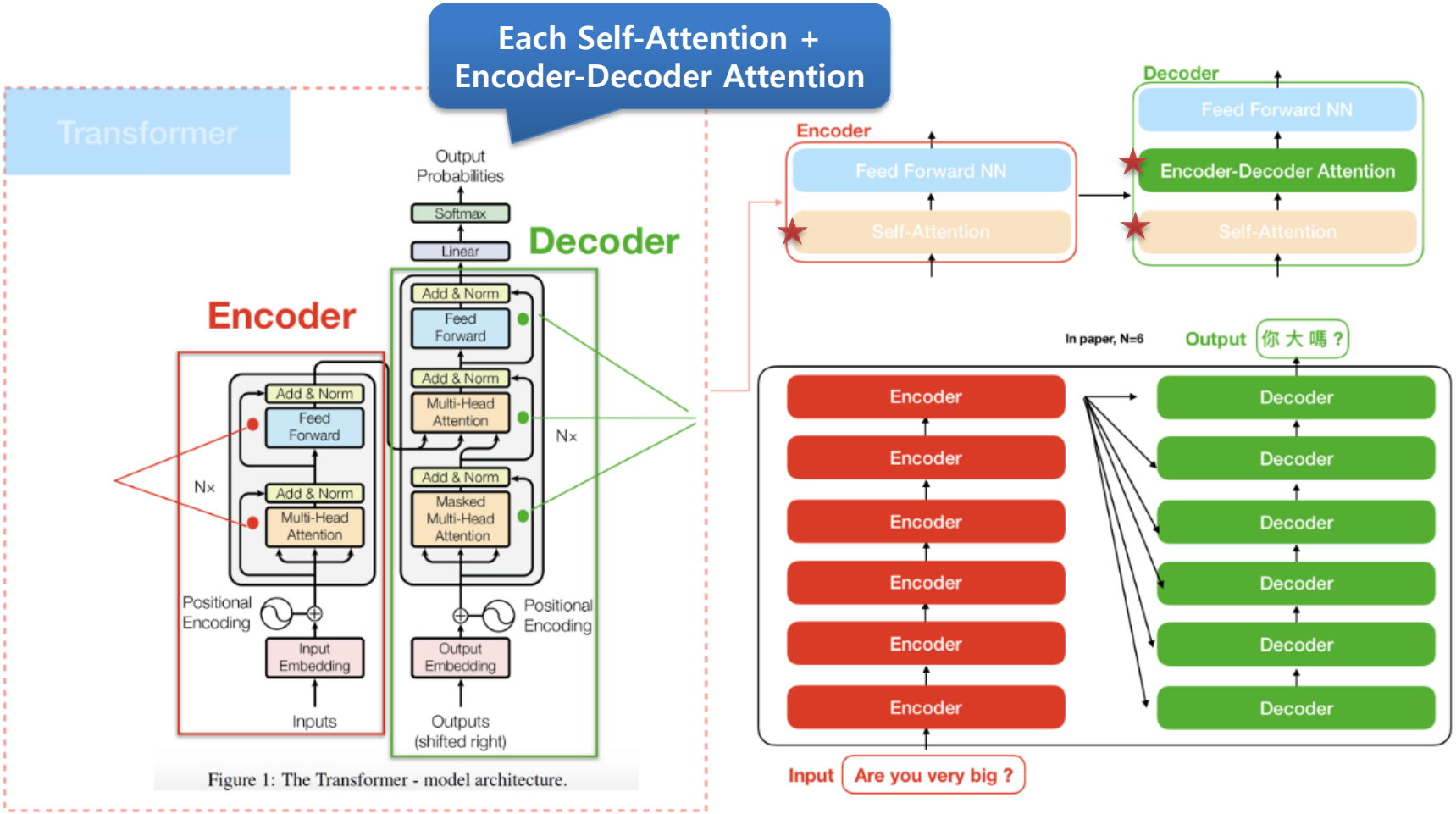


그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Attentions in Transformer

Transformer

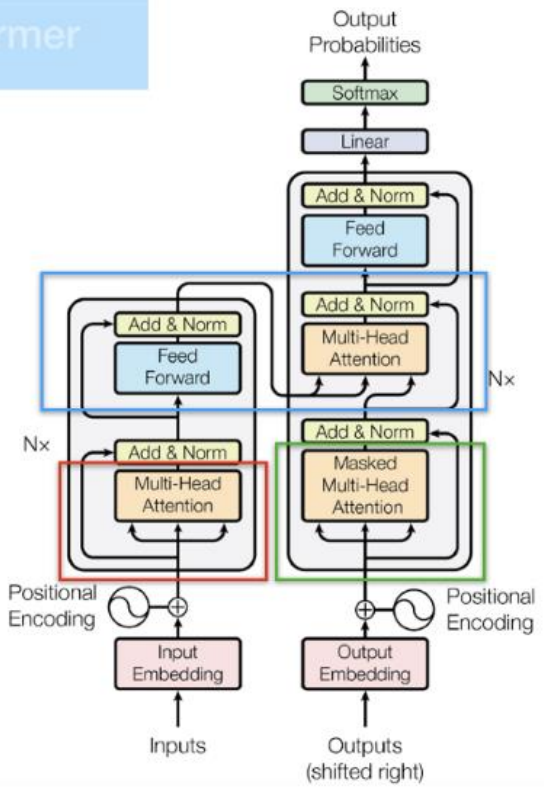


Figure 1: The Transformer - model architecture.

encoder self attention

1. Multi-head Attention
2. **Q**uery=**K**ey=**V**alue

decoder self attention

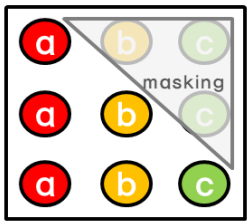
1. **Masked** Multi-head Attention
2. **Q**uery=**K**ey=**V**alue

encoder-decoder attention

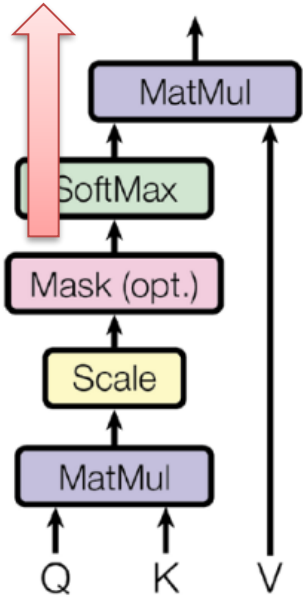
1. Multi-head Attention
2. Encoder Self attention=**K**ey=**V**alue
3. Decoder Self attention=**Query**

그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Scaled Dot-Product Attention



언어 모델
디코딩



비교대상, 입력, 히든벡터

Scaled Dot-Product Attention

Macro

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where $\boldsymbol{Q} \in \mathbb{R}^{n \times d_k}, \boldsymbol{K} \in \mathbb{R}^{m \times d_k}, \boldsymbol{V} \in \mathbb{R}^{m \times d_v}$

Sequence Q : $n * d_k \xrightarrow{attention\ layer} n * d_v$

Micro

$$Attention(\boldsymbol{q}_t, \boldsymbol{K}, \boldsymbol{V}) = \sum_{s=1}^{m} \frac{1}{Z} \exp\left(\frac{\langle \boldsymbol{q}_t, \boldsymbol{k}_s \rangle}{\sqrt{d_k}}\right) \boldsymbol{v}_s$$

Attention score

Recall Attention score $a_i = Softmax(Sim_i) = \frac{e^{Sim_i}}{\sum_{j=1}^{L_x} e^{Sim_j}}$

Hint

Q: Why divide by $\sqrt{d_k}$?

A:

$$\because \quad q \cdot k = \sum_{i=1}^{d_k} q_i \, k_i$$

Assume q_i , k_i are normal distribution,

$$\because \quad Var\left(\sum_{i=1}^{m} X_i\right) = \sum_{i=1}^{m} Var(X_i)$$

$q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.

Sum of the dot products will grow large in magnitude if d_k is large, pushing softmax into regions 0 or 1.

To counteract this effect, we scale the dot procuts by $\sqrt{d_k}$

그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Calculation of Attentions

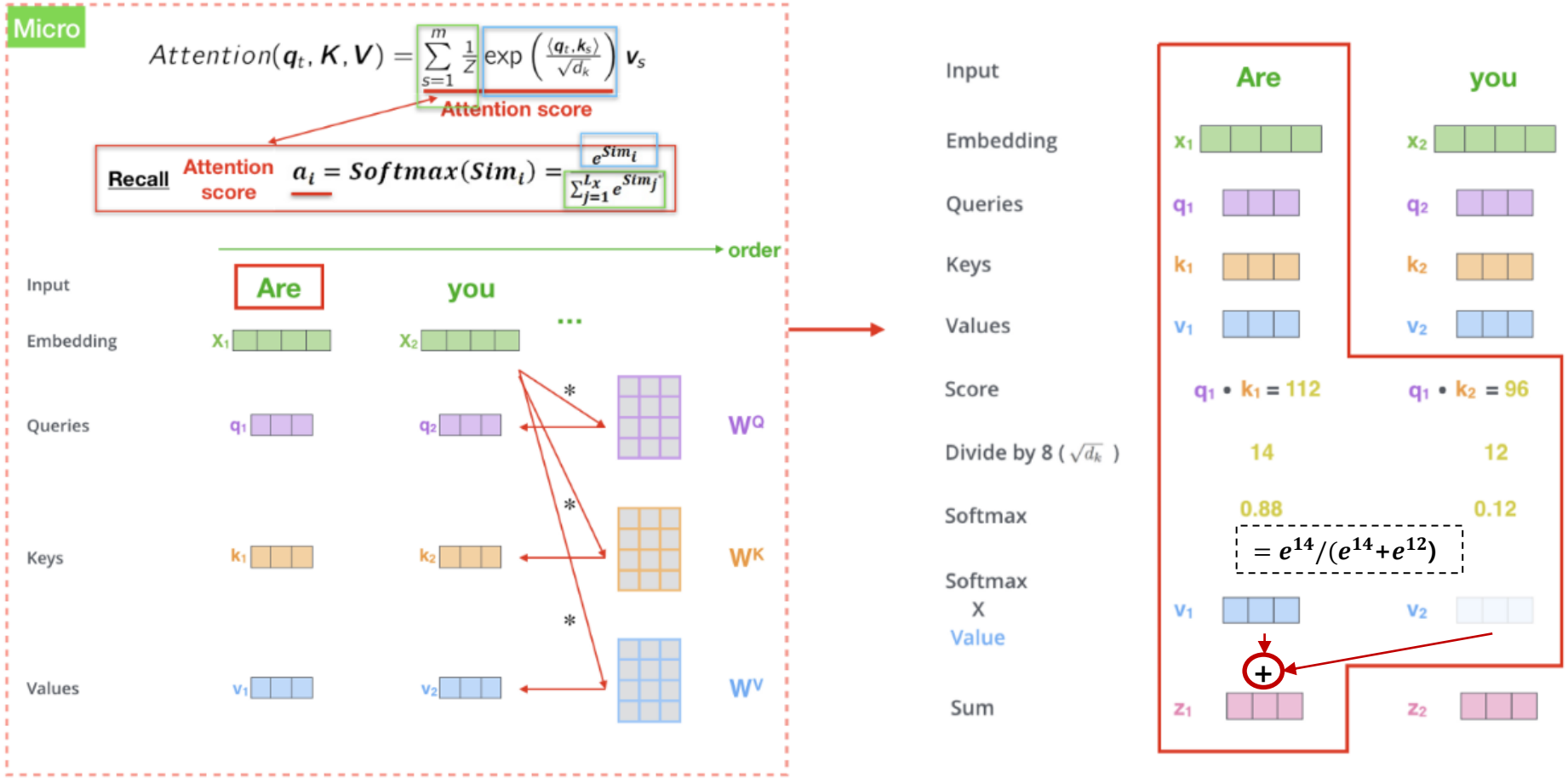


그림 출처: Ta-Chun (Bgg/Gene) Su's blog

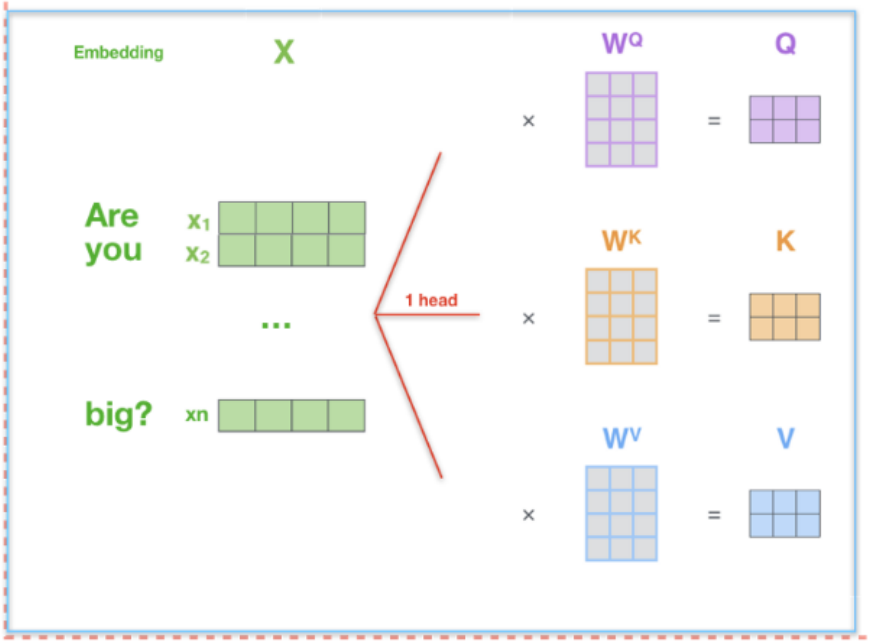
Calculation of Attentions

Macro

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where $\boldsymbol{Q} \in \mathbb{R}^{n \times d_k}, \boldsymbol{K} \in \mathbb{R}^{m \times d_k}, \boldsymbol{V} \in \mathbb{R}^{m \times d_v}$

SequenceQ : $n * d_k \xrightarrow{attention\ layer} n * d_v$



Self Attention

$$\text{Attention}(Q, K, V) = Attention(\boldsymbol{X}, \boldsymbol{X}, \boldsymbol{X})$$

$$\text{softmax}\left(\frac{Q \times K^T}{\sqrt{d_k}}\right) V = Z$$

Self Attention=
explore the correlation inside sequence itself

그림 출처: Ta-Chun (Bgg/Gene) Su's blog

확인 문제

- 다음과 같이 단어 임베딩이 주어졌을 때, self-attention score를 계산하시오. (소수점 이하 두 자리에서 반올림)
  - are: [1,1], you: [2,1]
  - root(2)=1.4로 계산

$$Attention(q_t, K, V) = \sum_{s=1}^{m} \frac{1}{Z} \exp\left(\frac{(q_t, k_s)}{\sqrt{d_k}}\right) v_s$$

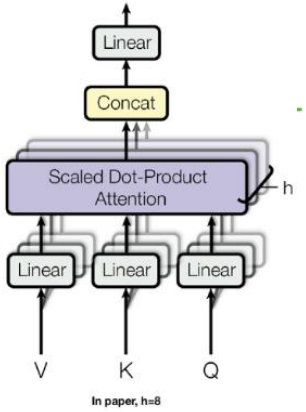
Scaled dot-product

	are	you
are	?	
you		

Self-attention score

	are	you
are	?	
you		

Multi-Head Attention



1. 작은 범위에서 softmax → 뚜렷한 특징만 살아남는 효과
2. 다양한 관점에서 문장을 바라보는 효과

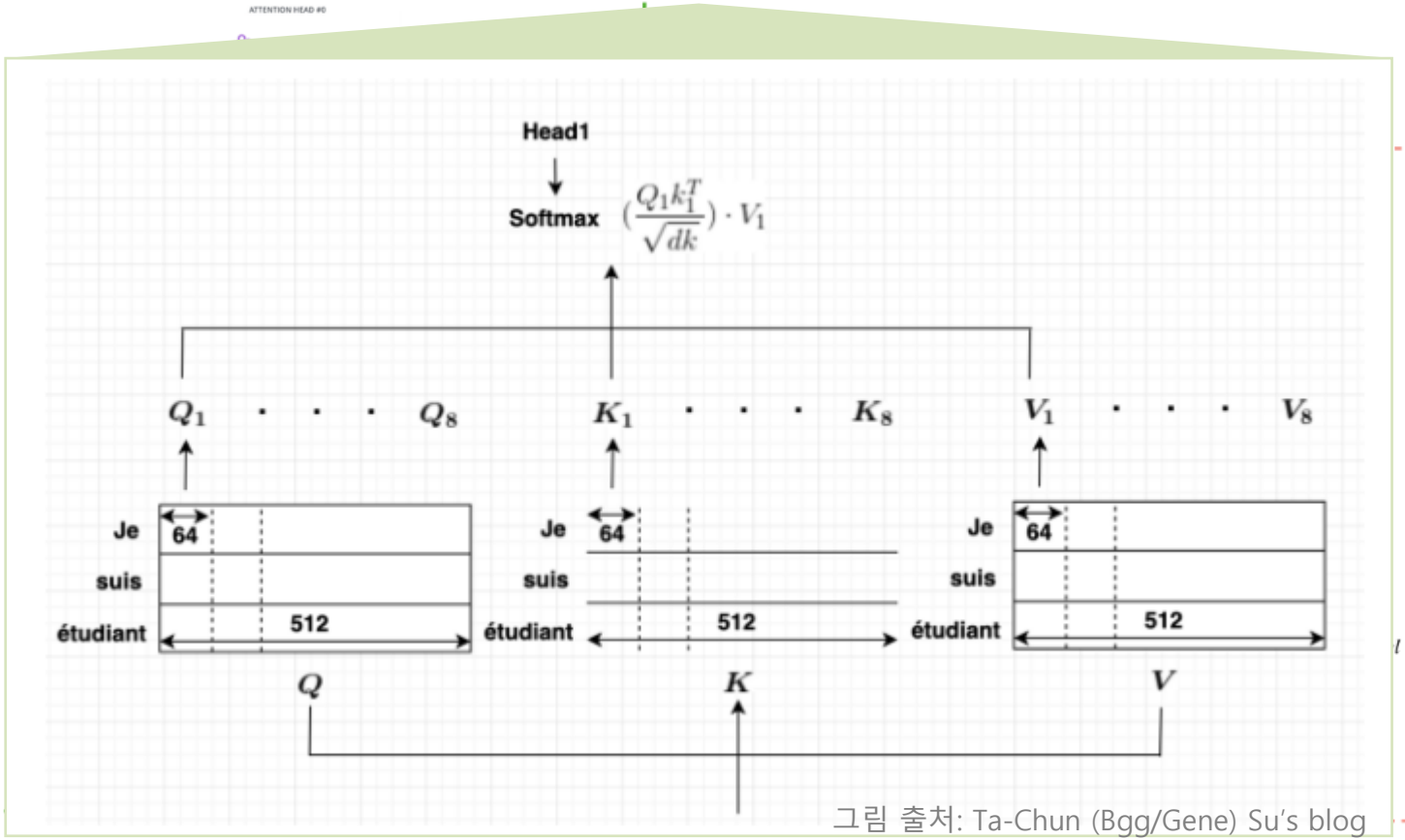


그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Layer Norm. & Residual Conn.

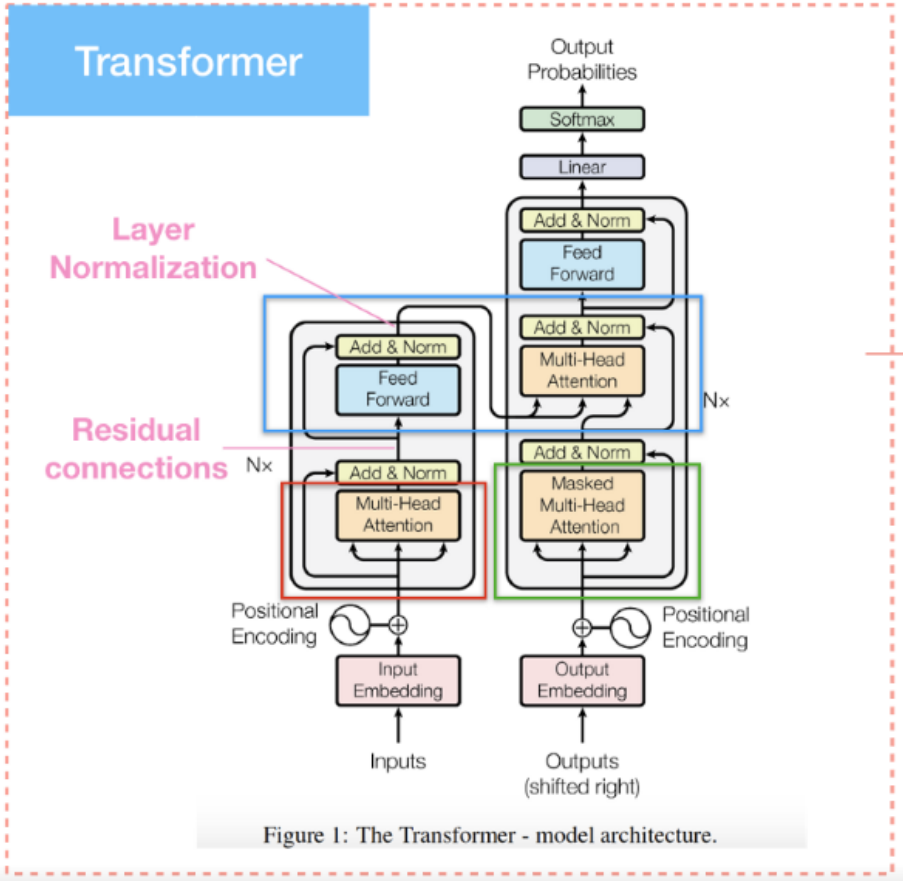
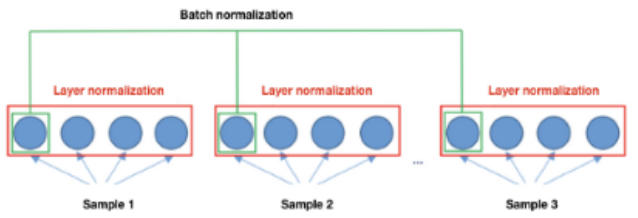


Figure 1: The Transformer - model architecture.

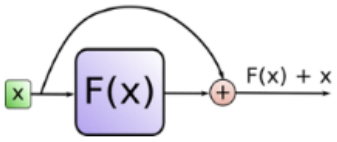
Layer Normalization



Advantage: The statistics are independent of other examples, therefore avoids Interval covariate shift (ICS).

Residual Connections

$$H(x) = g(x) = x + F(x)$$



Advantage:

fit a residual mapping $F(x)$, ∵ x=input

Easy to know how input changed

$$Ex: x = 2.9 \begin{cases} H_1(x) = 3,\ F(x) = 0.1 \\ H_2(x) = 3.1,\ F(x) = 0.2 \end{cases}$$

$$Plain\ layer = 3.3\,\%\ ,\ Residual\ block = 100\,\%$$

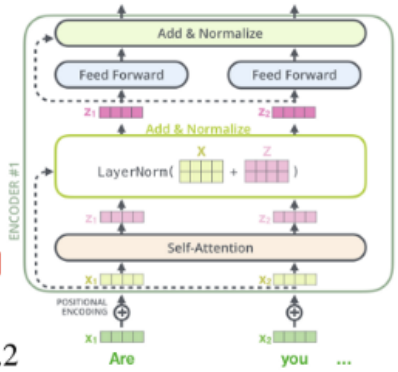
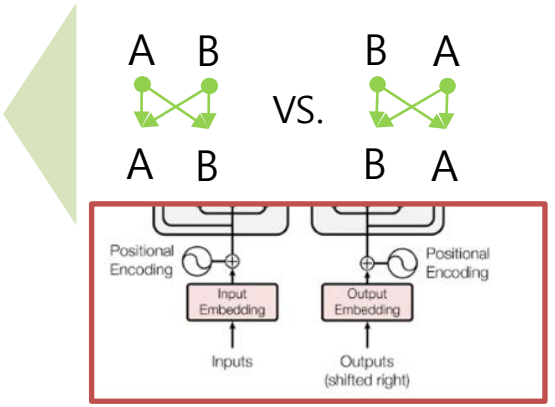


그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Position Encoding

Problem

The multi-head attention network **cannot** naturally **make use of** the position of the words in the input sequence.
The output of the multi-head attention network would be **the same** for the same sentences in different order.



Sol **Positional Encoding**

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$ ← 짝수 차원

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$ ← 홀수 차원

$if\ shape\ of\ enc, dec = [T, d_{model}]$ **pos: position of the word**

$\rightarrow then\ pos \in [0, T), i \in [0, d_{model})$ **i: Element i in** d_{model}

Advantage:

PE[pos+k] can be represented as a linear function of **PE[pos]**, so the relative positon between different embeddings can be easily inferred

$$sin(\alpha+\beta) = sin\alpha cos\beta + cos\alpha sin\beta$$
$$cos(\alpha+\beta) = cos\alpha cos\beta - sin\alpha sin\beta$$

$\because$ **Trigonometric Periodicity**

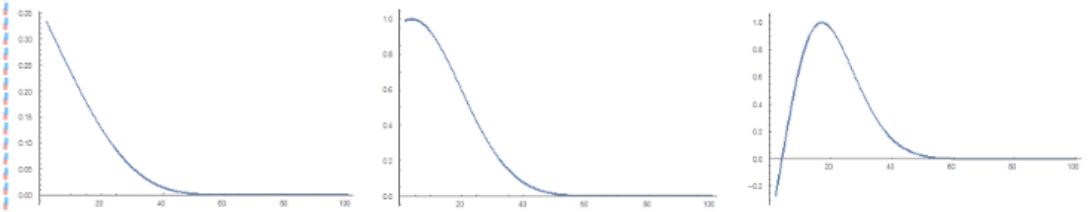
$\therefore$ **Learned the relation between relative and absolute position**

Then Word embedding + positional encoding

(sum, concat...)

$if\ d_{model} = 512:$

pos=1 pos=5 pos=10



→ **Three plots are the same by doing linear transformation**

그림 출처: Ta-Chun (Bgg/Gene) Su's blog

Linear & Softmax

Linear + Softmax

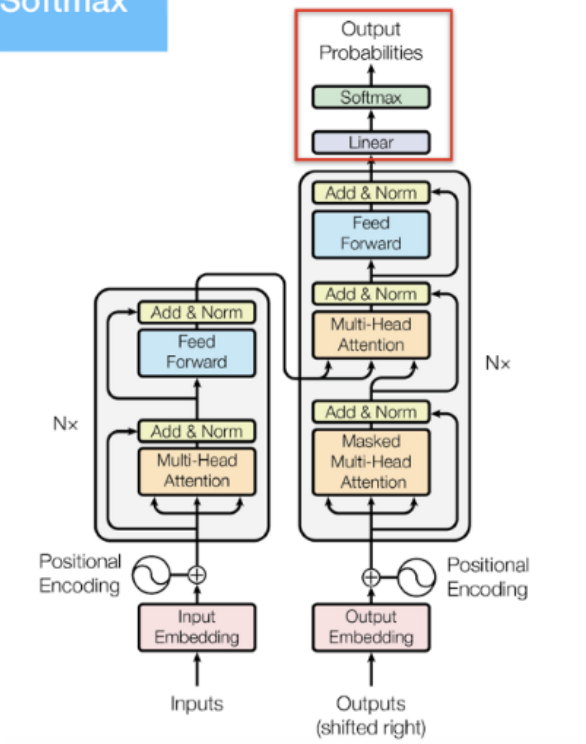


Figure 1: The Transformer - model architecture.

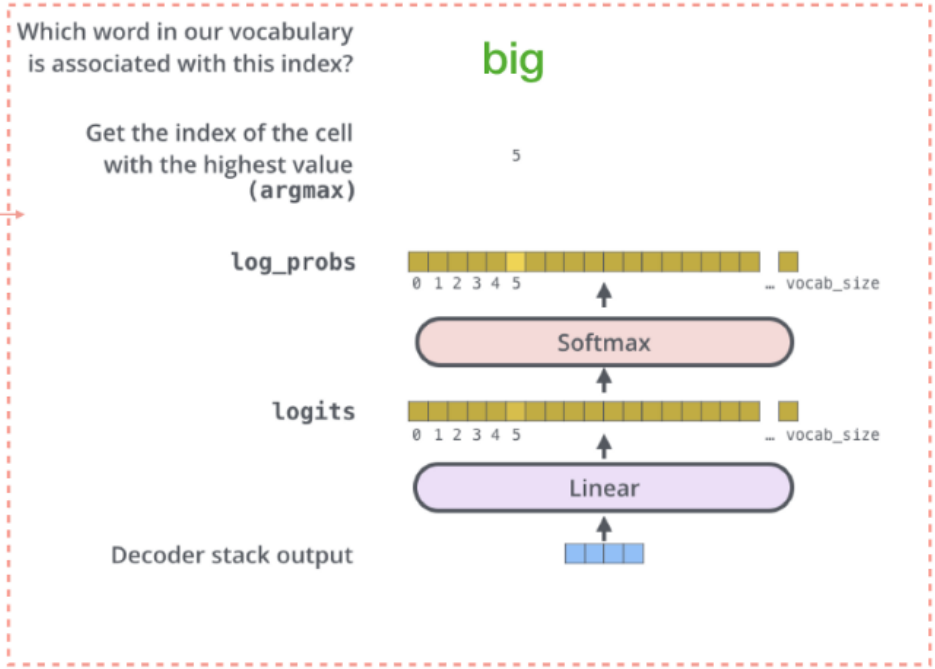


그림 출처: Ta-Chun (Bgg/Gene) Su's blog

질의응답

Q&A

Homepage: http://nlp.konkuk.ac.kr
E-mail: nlpdrkim@konkuk.ac.kr

KU KONKUK UNIVERSITY

KAIST

품사 부착
(Part-Of-Speech Tagging)

품사 부착 (Part-Of-Speech Tagging)

- 품사 부착이란?
  - 형태소 분석 결과에서 좌우 문맥을 보고 가장 적당한 후보를 1개 선택하는 과정
  - 예제
    - 어절 "감기는"의 품사 후보는?
      - 감기/명사+는/조사, 감기/동사+는/어미, 감/동사+기/명사형전성어미+는/조사
    - 문장 "감기는 자주 걸리는 병이다"에서 "감기는"의 품사는?
      - 감기/명사+는/조사

- 형태소 분리 vs 품사 태깅
  - 입력 단위: 어절 vs 문장
  - 출력 단위: 모든 가능한 후보 vs 정답 1개

HMM for POS tagging

- Hidden Markov Model for POS tagging

$$POS(S) = \arg_{T_{1,n}} \max P(M_{1,n}, T_{1,n})$$

Chain rule

$$POS(S) = \arg_{T_{1,n}} \max P(T_{1,n}) P(M_{1,n} | T_{1,n})$$

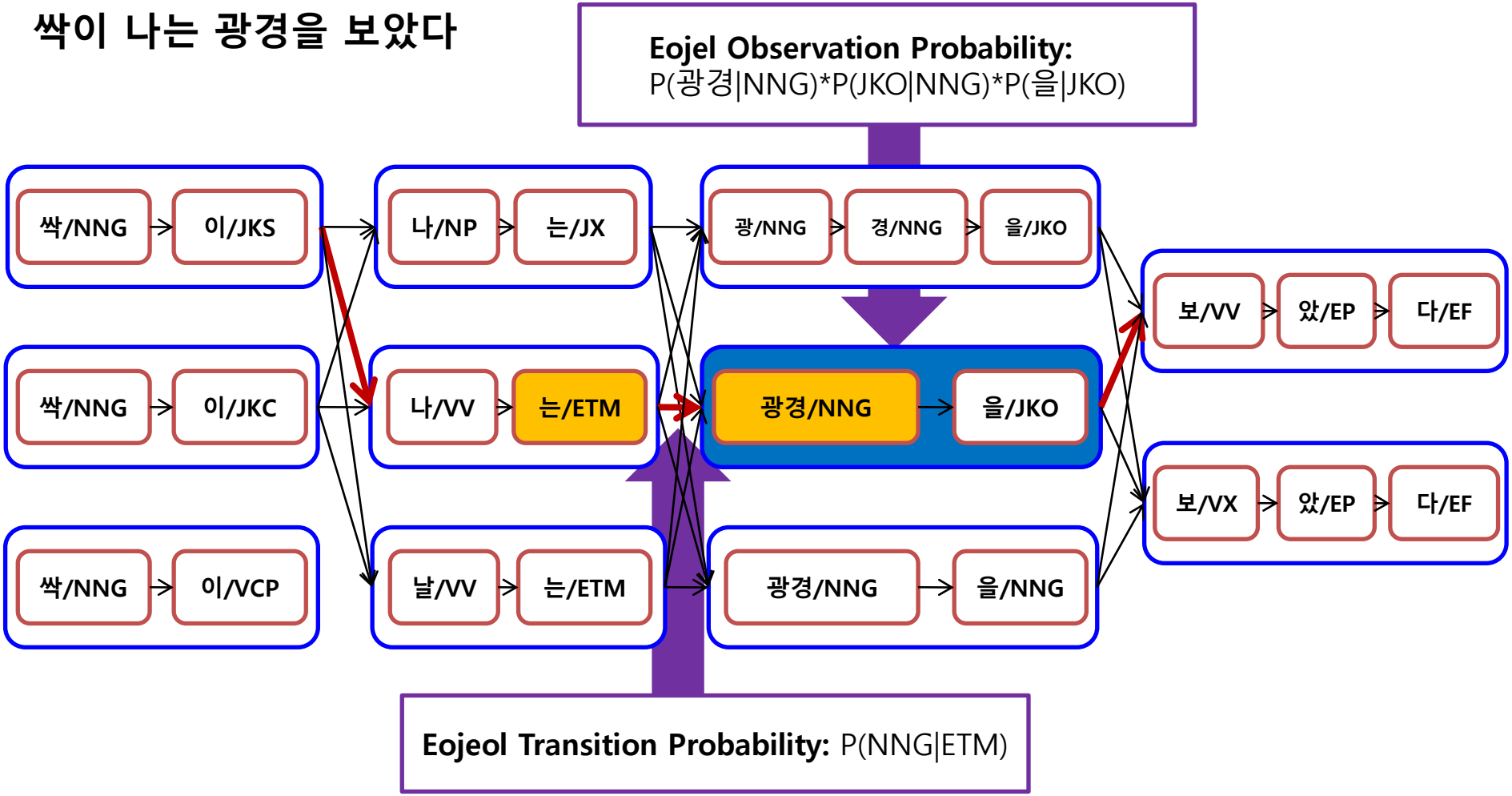
Independent Assumption

$$POS(S) = \arg_{T_{1,n}} \max \prod_{i=1}^{n} P(T_i | T_{i-1}) P(M_i | T_i)$$

1st Markov Assumption

한국어 품사 태깅 개념

싹이 나는 광경을 보았다



확률 계산

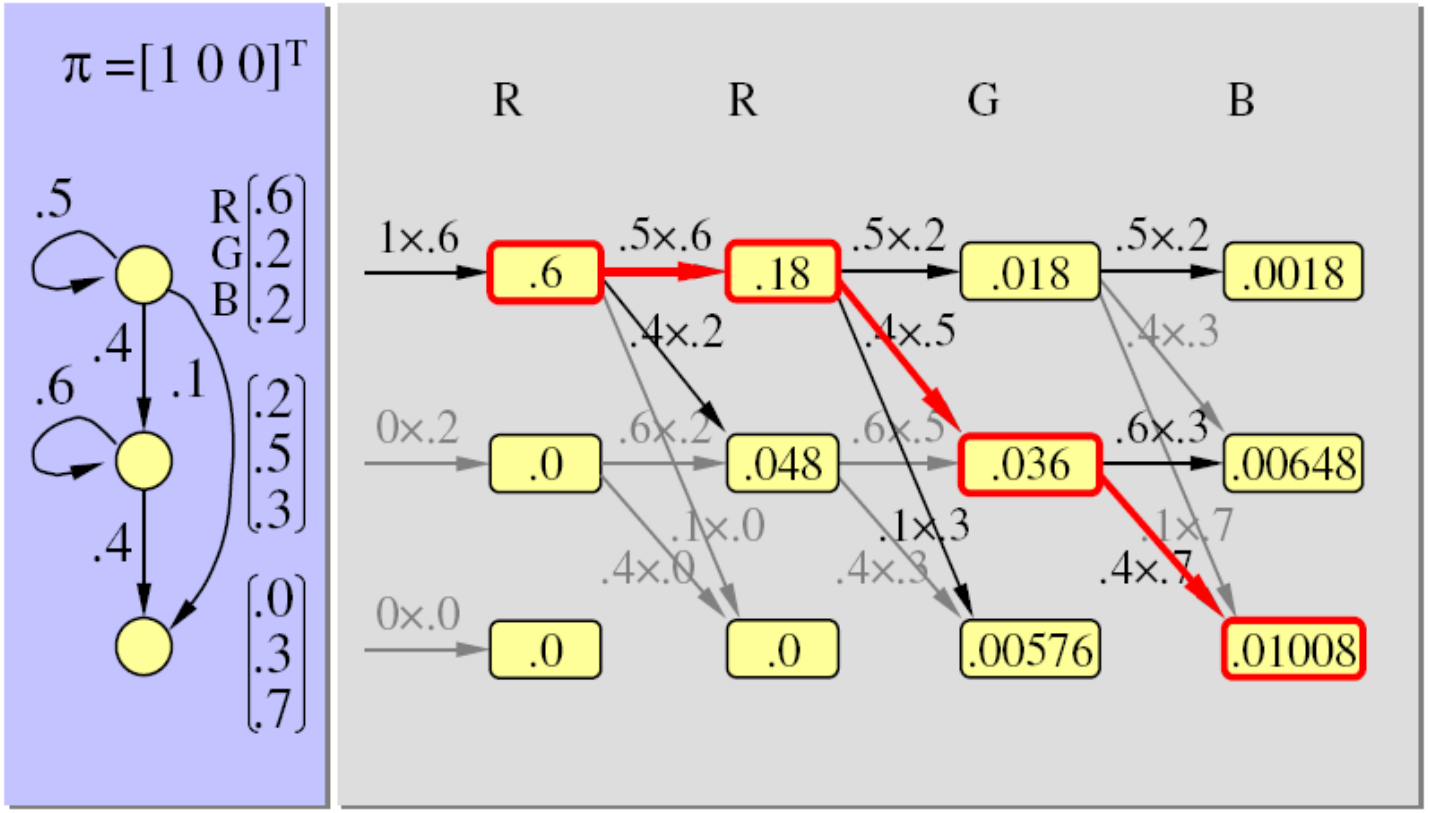
- 전이확률 $P(T_i \mid T_{i-1})$과 관측확률 $P(M_i \mid T_i)$의 계산
  - 말뭉치에서 출현 횟수를 계산
  - 데이터 희소성 문제를 줄이기 위해서 말뭉치에 나타나지 않는 경우에 매우 작은 확률(예: 0.0001)을 부여

$$P(V \mid N) = \frac{\#\ of\ \ "V"\ at\ position\ i}{\#\ of\ \ "N"\ at\ position\ i\text{-}1}$$

$$P(the \mid ART) = \frac{\#\ of\ \ "the/ART"}{\#\ of\ \ "ART"}$$

Viterbi Algorithm

- 비터비 알고리즘
  - 모든 경로를 다 고려하지 않고도 최대 확률값 경로를 찾는 알고리즘



Recent Morphological Analysis Models

- 형태소 분리, 품사 부착 동시 수행
  - 형태소 분리 → BIO Sequence Labeling
  - 품사 부착 → POS Sequence Labeling

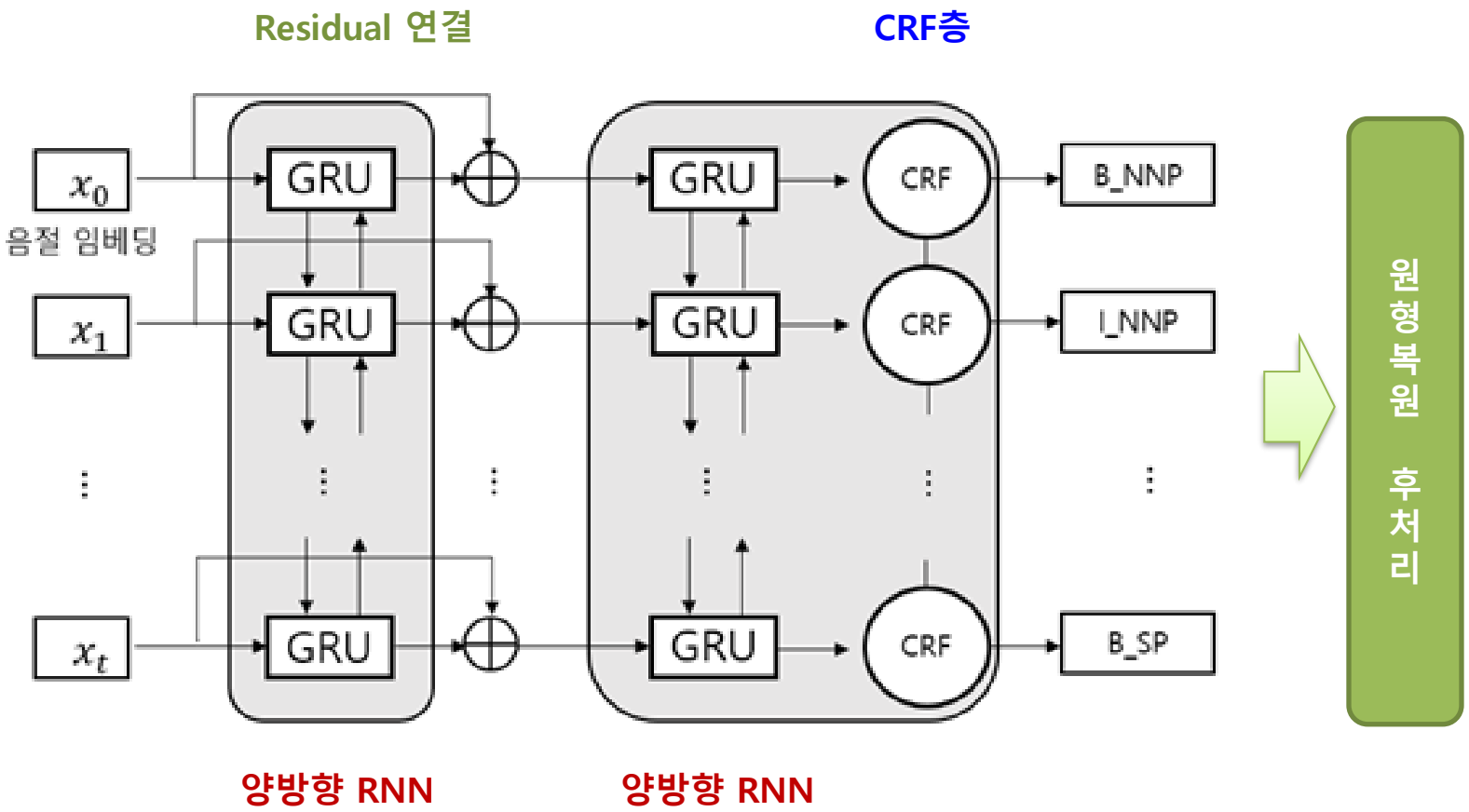
$$POS(S) = \arg_{L_{1,n}} \max P(L_{1,n} | C_{1,n})$$

+

규칙 기반 원형 복원
원형 생성 모델

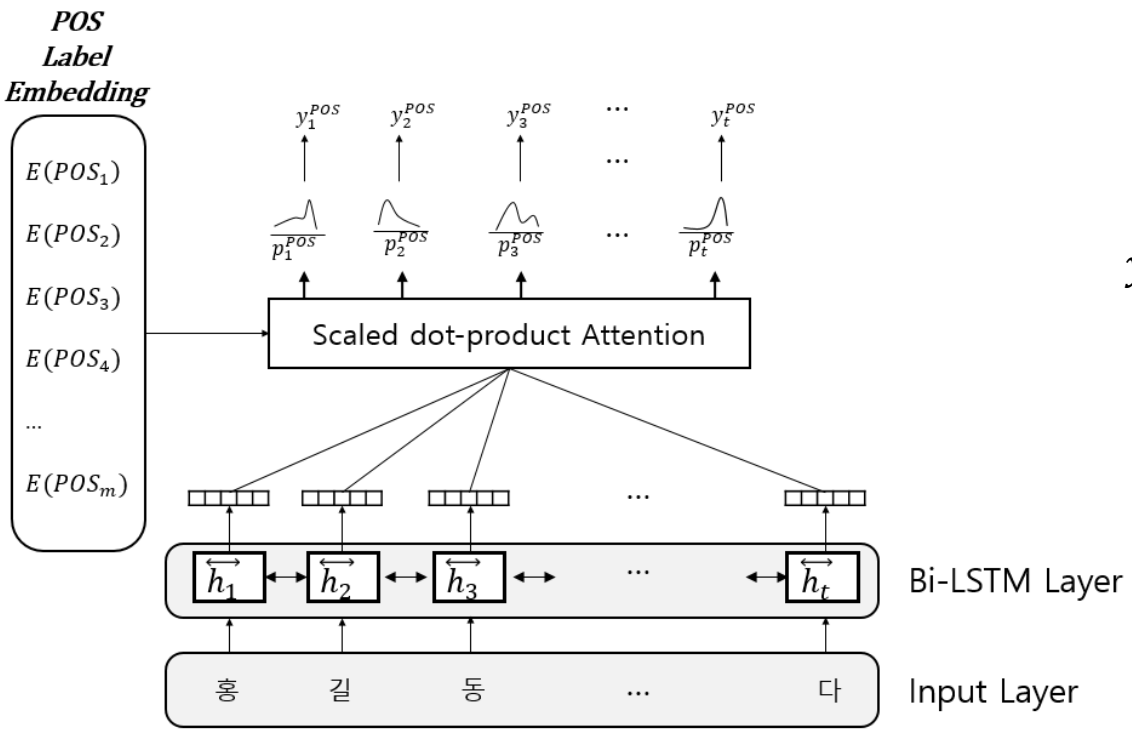
구분	표지	설명	예제
형태소 분석	B-POS	B: 형태소 경계의 시작 POS: 품사	=[청, 와, 대, 에, _, 갔, 다, .] =[B-NNP, I, I, B-JKB, O, B-VV+EP, B-EF, B-SF]
	I	I: 형태소 경계의 내부	
	O	O: 형태소 경계 바깥	

RNN+CRFs 형태소 분석 모델



LAN 기반 형태소 분석 모델

- LAN (Label Attention Network)



$$\overleftrightarrow{H} = (\overleftrightarrow{h_1}, \overleftrightarrow{h_2}, \overleftrightarrow{h_3}, \dots, \overleftrightarrow{h_t})$$

$$Q = \overleftrightarrow{H}, K = E(POS)$$

$$p^{POS} = softmax\left(\frac{Q * K^T}{\sqrt{d}}\right)$$

$$y^{POS} = argmax(p^{POS})$$

LAN 기반 형태소 분석 모델

- LAN 구조
  - $Q = \overleftrightarrow{H}, K = E(POS)$
    - Q는 양방향 문맥이 반영된 벡터, K는 가능한 모든 품사에 대한 임베딩
  - $p^{POS} = softmax\left(\frac{Q * K^T}{\sqrt{d}}\right)$
    - 현재 입력에 대한 품사 확률 분포
  - $y^{POS} = argmax(p^{POS})$
    - 최종 출력은 품사 확률 분포 중 가장 큰 값을 가지는 품사

DNN+CRFs vs LAN

- 메모리 사용량, 시간 복잡도
  - L : 레이블 개수(품사 개수), n : 시퀀스 길이(문장 길이)
  - CRFs의 시간 복잡도 : $O(|L|^2 * n)$
  - LAN의 시간 복잡도 : $O(|L| * n)$

Model	Memory usage (Megabyte)	Prediction time (Millisecond)
Bi-GRU-CRFs	4,946	8.0
Bi-LSTM-LAN	4,276	4.8

  - 메모리 사용량 측면에서 LSTM보다 효율적인 GRU를 사용했음에도, CRFs를 사용한 모델보다 LAN을 사용한 모델이 메모리, 시간 비용이 적음

형태소 분석 시연 영상

질의응답

Q&A

Homepage: http://nlp.konkuk.ac.kr
E-mail: nlpdrkim@konkuk.ac.kr