Maximum Entropy Markov Models

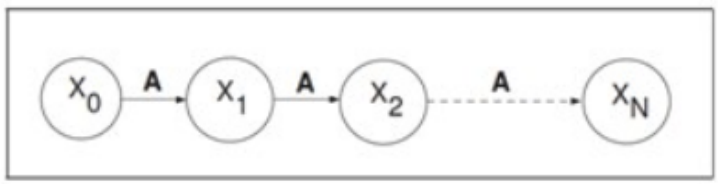
> Maximum Entropy model + Hidden Markov Model

먼저 Hidden Markov Model(이하 HMM), Maximum Entropy Markov Model(이하 MEMM) 을 왜 사용하는가??
- sequence classifier 로써, 각 unit 에 label/class 를 부여하는 작업을 진행하는 모델

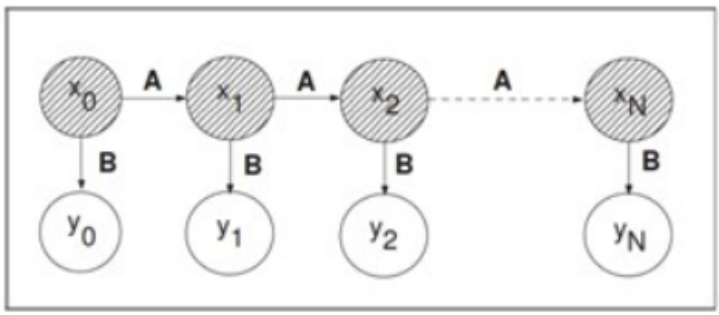
Hidden Markov Model(HMM)

- HMM 은 두가지 요소로 구성되어 있는데, Hidden State 와 관찰가능한 결과로 이루어져 있다.

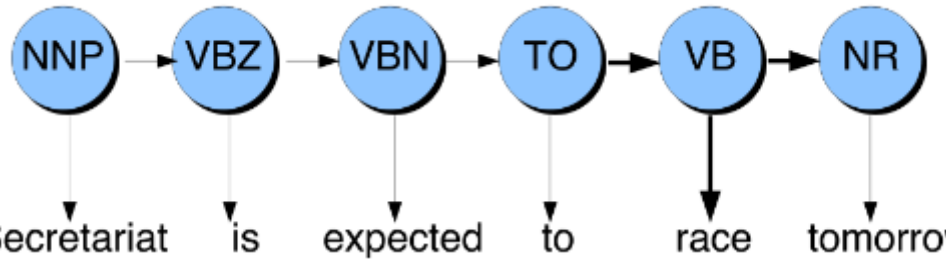
A Markov Model



A Hidden Markov Model



- NLP로 바라보면?



- 단어가 observation 이고, 품사 정보가 hidden state 인 것을 도식화 한 것으로, Time t 의 POS 에서 t+1 로 넘어가는 확률인 transition probability 와 특정 POS 에서 단어가 나올 확률인 Emission probability 를 통해 sequence 내에서 recursive 하게 prediction 을 하고자 하는 방법이다.

- ```
  ![[week9-fig17.png]]
  ```
    - $P(T)$: transition probability, $P(W|T)$: emission probability
    - 즉, 관측된 단어 Sequence W가 주어졌을 때 가장 확률이 높은 hidden state 의 Sequence T를 찾는다.
- 단점 : transition probability 와 emssion probability 만 활용하므로 문맥의 다양한 feature 들을 활용할 수 없다.

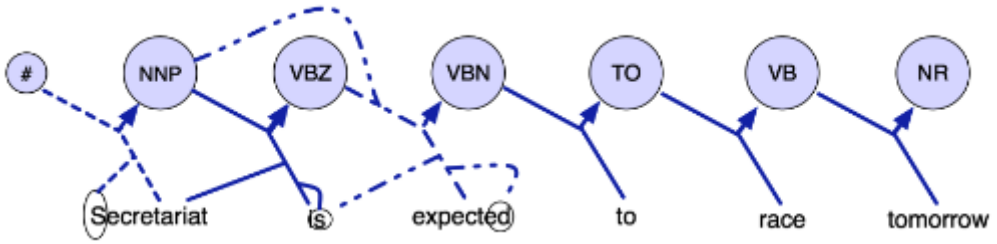
Maximum Entropy Model

- Multinominal logistic regression 이라고 할 수 있음.(자연어 처리에서 MEM 이라고 부름) 즉, 여러개의 데이터를 바탕으로 하나의 데이터를 예측하는 모델이다.
    - 단어 w 가 주어졌을 때, 범주 t (e.g., 품사) 가 나타날 확률
    - $$P(t|w) = \frac{exp\left\{\overrightarrow{w_t}^T \overrightarrow{f}(w)\right\}}{\sum_{t' \in T} exp\left\{\overrightarrow{w_{t'}}^T \overrightarrow{f}(w)\right\}}$$
    - feature vector f = 단어 w 에 해당하는 feature 들의 모음
        - 즉, f_0 = 직전 단어가 명사이면 1, 아니면 0, f_1 현재 단어가 동사이면 1, 아니면 0 등등의 feature 이 모아져 있는 vector 라고 할 수 있다.
- Maximum Entropy Model 은 feature vector 를 매우 유연하게 설정할 수 있어, 사전지식을 모델링에 활용 할 수 있는 장점이 있음.
- 단점 : sequence 가 아닌 single observation에 대해서만 예측이 가능

Maximum Entropy Markov Model

- Maximum Entropy Model 의 장점 중 하나인 feature vector 활용 능력을 바탕으로 sequence classify 를 가능하게 하는 Model 이다.
    - NER(Named Entity Recognition) 에 사용됨
    - Word 정보를 기반으로 POS 를 예측하는 방식이다.
        - 간단하게 말하자면 현재의 POS 를 계산하기 전 Time step(t)의 hidden state 와 현재의 feature 를 정보로 주고 현재의 POS 를 계산하는 방식이다.
  ```

- 주어진 observation 들을 바탕으로 hidden state 의 조건 분포들을 모델링 하는 것



이다.

" MEMM-fig4.pn

  - Hidden state는 Markov chain 을 따르되, sequence prediction 에 다양한 feature 들을 활용하는 방식.
    - 여기서는 단어 뿐만 아니라 대소문자, 글자 수, 마지막 알파벳과 같은 여러

$$\hat{T} = arg\max_{T} P(T|W)$$
$$= arg\max_{T} \prod_{i} P\left(T_i|W_i, T_{i-1}\right)$$

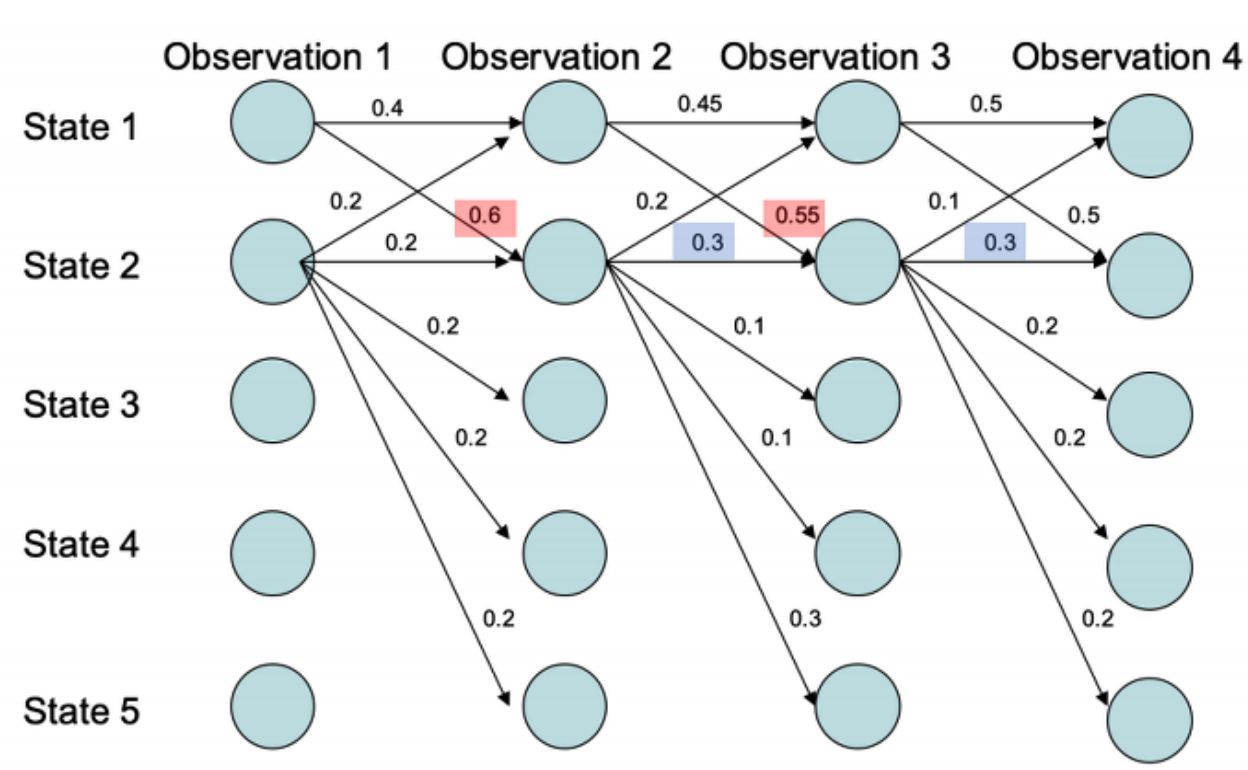
      가지 feature 들을 활용했다.

      " MEMM-fig6.pn
    - MEMM 이 sequence T 를 예측하기 위해 i번째 단어의 feature 와 직전 hidden state(품사)를 바탕으로 가장 확률이 높은 현재 hidden state 의 품사를 반환하는 것이다.
    - $P(T_i|W_i, T_{i-1})$ 은 MEM 을 가리키고, 이를 식으로 적으면 아래와 같다.

$$P(T_i|W_i, T_{i-1}) = \frac{exp\left\{\overrightarrow{w_t}^T \overrightarrow{f}\left(W_i, T_{i-1}\right)\right\}}{\sum_{t' \in T} exp\left\{\overrightarrow{w_{t'}}^T \overrightarrow{f}\left(W_i, T_{i-1}\right)\right\}}$$

      - f 는 w_i 와 $i-1$ 번째 hidden state 에 해당하는 featue vector 가 된다.
        - 현재 단어의 모든 Feature W_i , $i-1$ 번째의 hidden state T_{i-1}
- 즉, MEMM 은 가장 가능성 있는 품사를 찾기 위해 Markov chain 의 상태로 모델링 하여 tag sequence 를 예측한다. 품사를 예측하기 위해 MEMM 은 현재의 word 와 이전 word 에 할당된 품사를 사용한다. 각 품사의 확률은 MEM 을 활용하여 계산한다.
- 단점 : 현재의 State 만 고려하기 때문에 전체 Sequence 의 확률이 높은것이 아닌 현재 확률만 고려하여 가장 높은 것을 고른다는 문제점이 있다. "**Label Bias**"

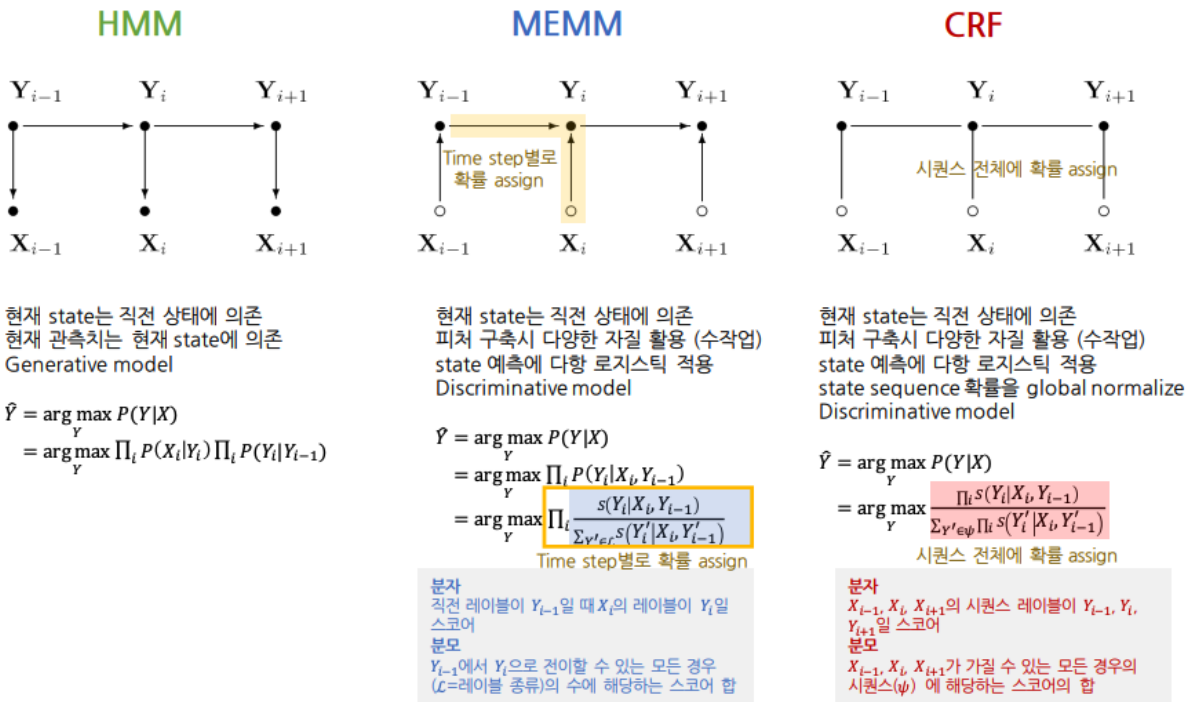
Label Bias



- 현재 state 만 바라보고 확률을 계산하게 되면 Stage1 -> Stage2 -> Stage2 가 될 텐데, 전체 확률을 놓고 보면 Stage1 -> Stage1 -> Stage1 이 가장 큰 확률을 가지게 된다.
    - 이것이 Label Bias 라고 할 수 있다.

CRF(Conditional Random Fields)

- Label Bias 를 Global Normalize 라는 방법을 통해 해결했다.
- Global Normalize?
    - 가능한 모든 조합의 label sequence 에 대한 확률을 구해야함
        - DP 이용하여 비효율성 개선



-

Reference.

https://ratsgo.github.io/machine%20learning/2017/11/04/MEMMs/
https://devopedia.org/maximum-entropy-markov-model
https://ratsgo.github.io/machine%20learning/2017/10/26/MEMs/
https://www.quantumdl.com/entry/Endtoend-Sequence-Labeling-via-Bidirectional-LSTMCNNsCRF
https://ko.wikipedia.org/wiki/%EC%A1%B0%EA%B1%B4%EB%B6%80_%EB%AC%B4%EC%9E%91%EC%9C%84%EC%9E%A5