Conditional Random Fields cho bài toán VLSP 2018 NER

Giảng viên hướng dẫn: TS. Nguyễn Phi Lê

Sinh viên thực hiện: Nguyễn Thị Thắm 20183984

Content

- **NER Problem**
  - Problem Overview
  - Approaches
- Machine Learning Approaches for NER
  - Compare Generative vs Discriminative Models
  - Generative model : HMM
  - Discriminative model : MaxEnt Model
    - Maximum Entropy Markov Model (MEMM)
    - Conditional Random Fields (CRF)
- Reference

Named Entity Recognition Problem (NER)

- **Ký hiệu**:

Đầu vào: chuỗi từ(câu) $\mathrm{x} = (x_1, x_2, \dots x_T)$

Cần dự đoán chuỗi nhãn: $\mathrm{y} = (y_1, y_2, \dots y_T)$

Tập huấn luyện $D = \{(x^1, y^1), \dots . (x^N, y^N)\}$

Foreign	ORG
Ministry	ORG
spokesman	O
Shen	PER
Guofang	PER
told	O
Reuters	ORG
:	:

- **Nhiệm vụ của mô hình:**

Cần tính được $\boldsymbol{p(y|x)}$

Approaches

➢ **Các cách giải quyết**:

- **Rule-based model**: Pattern Matching, Dictionary (spacy)
  → Không khả thi với các bài toán phức tạp. Nhưng có thể sử dụng kết hợp với machine learning model

```
{"label": "GPE", "pattern": [{"LOWER": "san"}, {"LOWER": "francisco"}]}
```

Ví dụ về rule-based entity recognition

- **Machine Learning Model**:
  Classification: *SVM*
  Seq-to-seq: ***CRF, HMM***
  → Với NER, độ chính xác có thể đạt ~ mô hình Deep Learning

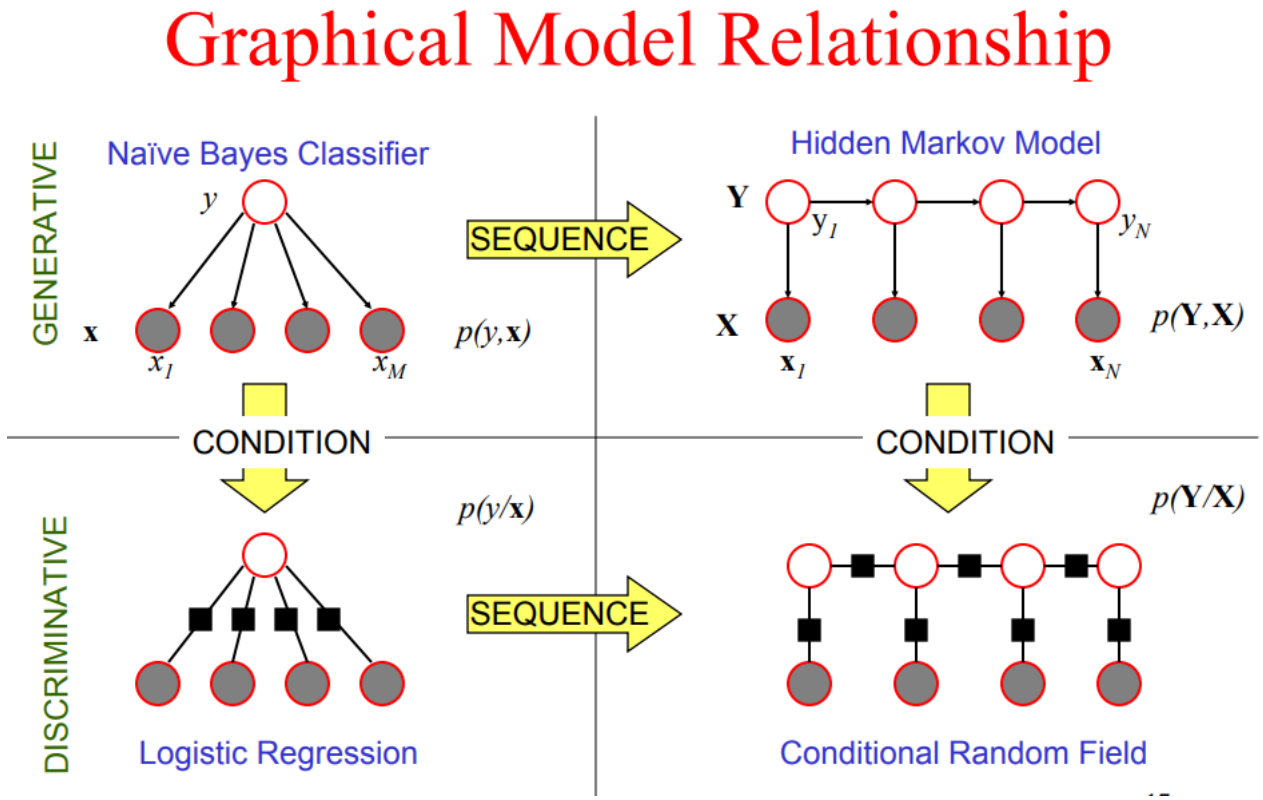
- **Deep Learning Model**: *CNN, LSTM, RNN, Bi-LSTM*

Content

- NER Problem
  - Problem Overview
  - Approaches
- Machine Learning Approaches for NER
  - Compare Generative vs Discriminative Models
  - Generative model : HMM
  - Discriminative model : MaxEnt Model
    - Maximum Entropy Markov Model (MEMM)
    - Conditional Random Fields (CRF)
- Reference

Machine Learning Model for Seq-to-Seq

- Chọn mô hình thuộc dạng nào? **Generative** or **Discriminative**?



Generative vs Discriminative Model

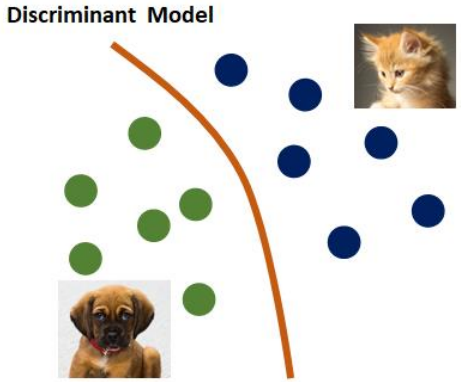
- **Generative Model** :
  - Quá trình học: Tính $\boldsymbol{p(x, y)}$ cho tất cả khả năng gán (x,y)

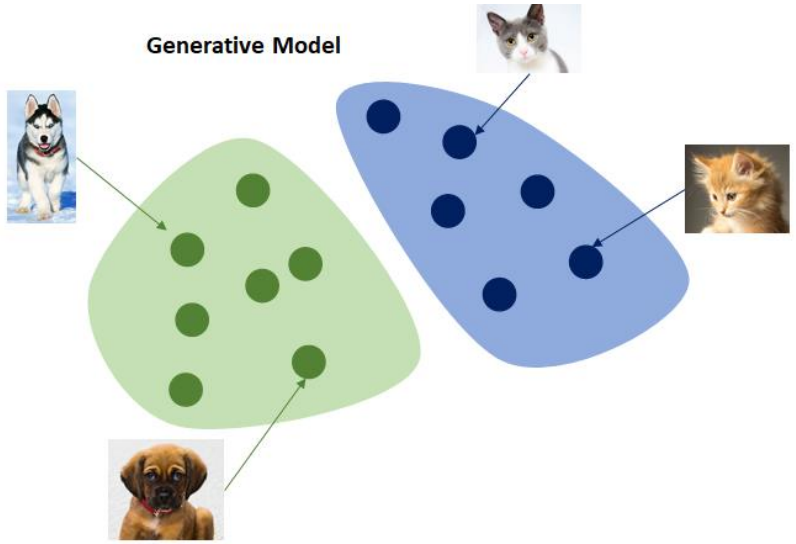
    Công thức: $p(x, y) = p(x|y).p(y)$

    Với $p(x|y), p(y)$ được tính từ tập dữ liệu huấn luyện

    Để phân loại thì cần tính: $p(y|x) = \frac{p(x,y)}{p(x)}$
- **Discriminative Model**:
  - Quá trình học: Mô hình $\boldsymbol{p(y|x)}$ bằng cách:

    Thiết lập tham số ban đầu → Tính y → Tính Loss → Cập nhật lại đến khi hội tụ





Generative vs Discriminative Model

- **Generative Model (Mô hình sinh)**:
  - Mô hình điển hình:
    Classification: Naïve Bayes
    Seq2Seq: **Hidden Markov Model**
- **Discriminative Model**:
  - Mô hình điển hình:
    Classification: Support Vector Machine, Logistic Regression, CNN
    Seq2Seq: **Conditional Random Fields, Maximum Entropy Markov Model**

Generative vs Discriminative Model

- **So sánh 2 mô hình:**

	Generative Model		**Discriminative Model**
Độ chính xác	Do phải giả định về sự độc lập	**<**	Không cần mô hình $p(x\|y)$ nên không cần giả định
Với dữ liệu thiếu (các thuộc tính của X không quan sát đầy đủ)	Có thể tính được: $p(y\|x_{given})$ $= \sum_{x_{miss}} p(y\|x_{given}, x_{miss})$	**>**	Bắt buộc cần phải quan sát được toàn bộ x
Hiệu quả	Thích hợp với tập dữ liệu nhỏ (do giả định về sự độc lập -> tránh overfitting)		Vượt trội hơn nếu có tập dữ liệu lớn

Generative vs Discriminative Model for NER

- **Generative Model:**

Hidden Markov Model (HMM)

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

→ Mô hình cần tính được: $p(x,y)$ cho tất cả các khả năng gán nhãn

$$p(x,y) = \prod_{i=1}^{T} p(y_i|y_{i-1}).p(x_i|y_i)$$

- **Discriminative Model**:

Conditional Random Fields (CRF)

$$p(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{i=1}^{T}\sum_{k=1}^{L} \boldsymbol{\lambda_k} h_k(y_{i-1}, y_i, x, i) + \sum_{i=1}^{T}\sum_{k=1}^{L} \boldsymbol{\mu_k} g_k(y_i, x, i)\right)$$

→ tính trực tiếp dựa vào tham số

Content

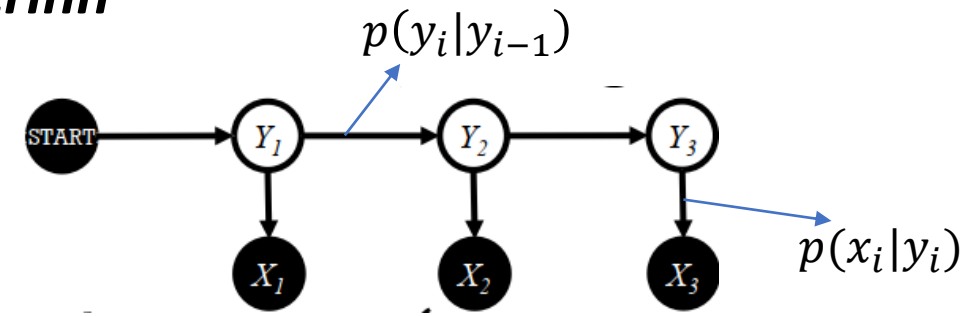
- NER Problem
- Machine Learning Approaches for NER
  - Compare Generative vs Discriminative Models
  - Generative model : HMM
    - Representation
    - Inference
    - Learning: Supervised & Semi-supervised Learning
    - Shortcomings
  - Discriminative model : MaxEnt Model
    - Maximum Entropy Markov Model (MEMM)
    - Conditional Random Fields (CRF)

HMM – Representation

➢ **Hidden Markov Model (HMM)**

- Dùng ***Bayesian Network (BN)*** để biểu diễn mô hình:

  BN: đồ thị ***có hướng, không có chu trình***



Mỗi node: biến ngẫu nhiên

Cạnh: A → B ~ p(B|A)

- Giả định của HMM:
  - Các x_i độc lập với nhau
  - x_i chỉ phụ thuộc vào y_i
  - y_i chỉ phục thuộc vào y_{i-1}
- Từ giả định → xây dựng BN
- Từ BN → cách thức factorize $p(x, y)$:

$$p(x, y) = \prod_{i=1}^{T} p(y_i|y_{i-1}).p(x_i|y_i)$$

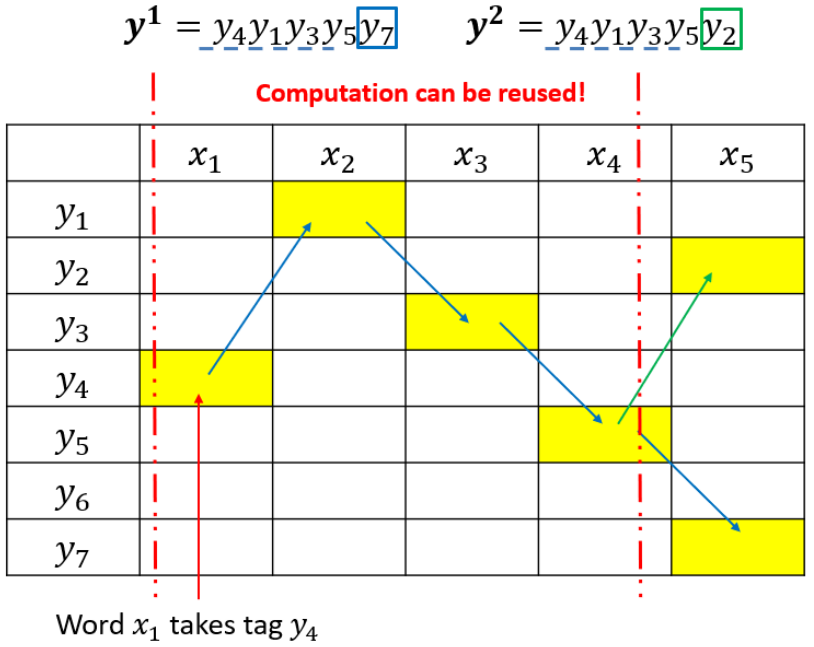
HMM – Inference

- **Mục tiêu:**

$$\boldsymbol{y}^* = argmax_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x})$$

$$= argmax_y \prod_i p(x_i|y_i)p(y_i|y_{i-1})$$

- Độ phức tạp:
  - Mỗi từ có K nhãn, có n từ → có K^n khả năng gán nhãn



Trellis Representation

→ **Viterbi algorithm**

HMM – Inference

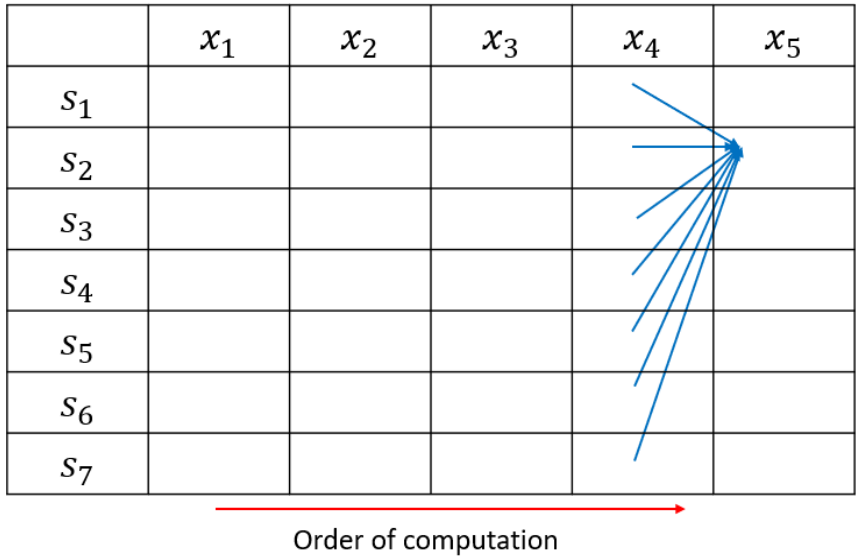
- **Viterbi Algorithm**:

Lưu lại các giá trị trính tại mỗi bước

$$\delta_i(s) = \max_{y \in Y^{i-1}} \prod_{t=1}^{i-1} p(x_t|y_t)p(y_t|y_{t-1}) \; . p(x_i|s)p(s|y_{i-1})$$

$$\delta_i(s) = p(x_i|s) max_{y_{i-1}} \left(\delta_{i-1}(y_{i-1})p(s\,|y_{i-1})\right)$$

Time Complexity: $O(K^2 n)$!



HMM – Learning

- **Tham số mô hình cần học**: K : số nhãn , n : số từ đầu vào, V: kích thước tập từ vựng.

  - Xác suất chuyển nhãn: $a_{ij} = p(y_i|y_j)$ số lượng: $(K * K)$
  - Xác suất sinh từ: $b_j(x_t) = p(x_t|y_j)$ số lượng $(V * K)$
  - Xác suất nhãn ở vị trí đầu tiên: $\pi_i = \overline{p}(y_i)$ số lượng (K)

  Gọi tập toàn bộ tham số của mô hình là

$$\boldsymbol{\lambda} = \{a_{ij}, b_j(x_t), \pi_i\}$$

HMM – Learning

➢ **Các cách huấn luyện:**

- **Unsupervised Learning** :

  $\boldsymbol{D_u}$: dữ liệu không gán nhãn (chỉ bao gồm từ)
- **Semi-supervised Learning**:

  $\boldsymbol{D = \{D_u, D_l\}}$: dữ liệu có một phần không được gán nhãn
- **Supervised Learning**:

  $\boldsymbol{D_l}$: dữ liệu được gán nhãn toàn bộ

HMM – Supervised Learning

- Dữ liệu huấn luyện D_l
- Mục tiêu học: Tìm ra **Maximum Likelihood Estimator (MLE)** cho mô hình ~ Cực đại hóa $p(x, y|\boldsymbol{\lambda})$

$$L(\boldsymbol{\lambda}: D) = \log(p(x, y|\boldsymbol{\lambda}))$$

$$= \log\left(\prod_{k=1}^{M} p(y_1^k|\boldsymbol{\lambda}).p(x_1^k|y_1^k, \boldsymbol{\lambda}) \prod_{i=2}^{T} p(y_i^k|y_{i-1}^k, \boldsymbol{\lambda}).p(x_i^k|y_i^k, \boldsymbol{\lambda})\right)$$

$$= \sum_{k=1}^{M} \log\left(p(y_1^k|\boldsymbol{\lambda})\right) + \sum_{k=1}^{M}\sum_{i=1}^{T} \log(p(x_i^k|y_i^k, \boldsymbol{\lambda})) + \sum_{k=1}^{M}\sum_{i=2}^{T} \log\left(p(y_i^k|y_{i-1}^k, \boldsymbol{\lambda})\right)$$

→ Thống kê tần số trên tập dữ liệu để xấp xỉ p

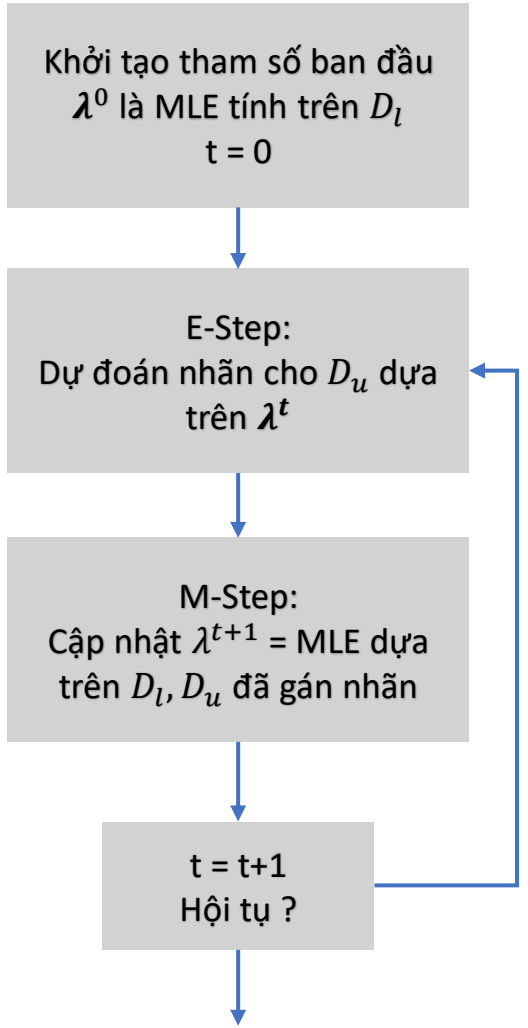
- Xác suất chuyển nhãn: $a_{ij} = p(y_i|y_j)$
- Xác suất sinh từ: $b_{y_j}(x_t) = p(x_t|y_j)$
- Xác suất nhãn ở vị trí đầu tiên: $\pi_i = \overline{p}(y_i)$

Gọi tập toàn bộ tham số của mô hình là $\boldsymbol{\lambda} = \{a_{ij}, b_j(x_t), \pi_i\}$

HMM – Semi-supervised Learning

- Dữ liệu huấn luyện
  $$D = \{D_u, D_l\}.$$

- Hướng thực hiện: sử dụng thuật toán **Expectation - Maximization** (thuật toán lặp để tìm ra ***local MLE*** hoặc ***MAP*** cho tham số của mô hình



MLE: Maximum Likelihood estimator
MAP: Maximum a posteriori estimate

HMM – Semi-supervised Learning

- **Expectation Step**: Tính

$$Q(\lambda|\lambda_{old}) = E[l(\lambda: X, Y)|X, \lambda_{old}]$$

$$= \sum_{y} l(\lambda: X, y).p(y|X, \lambda_{old})$$

- **Maximization Step**:

$$\lambda = argmax_{\lambda} Q(\lambda|\lambda_{old})$$

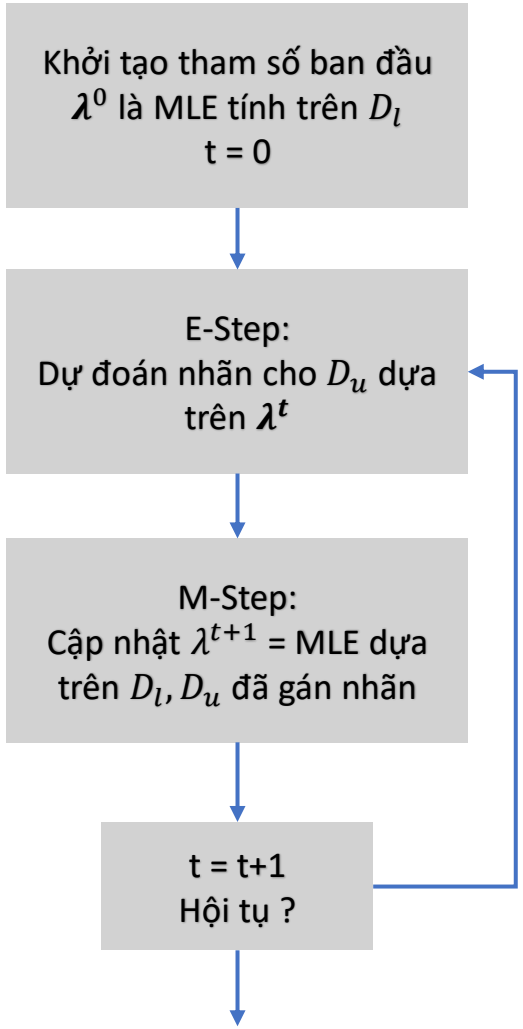
l: log-likelihood function
E: expected value

MLE:

$$\lambda = argmax_{\lambda} \mathcal{L}(\lambda; x, y)$$
$$= argmax_{\lambda} log\ (p(x, y|\lambda))$$

EM:

$$\lambda = argmax_{\lambda}\ Q(\lambda|\lambda_{old})$$
$$= argmax_{\lambda} E[l(\lambda: X, Y)|X, \lambda_{old}]$$



HMM – Semi-supervised Learning

➢ Tính hội tụ của EM:

Tại sao không tối ưu trực tiếp được?

$$l(\lambda; X) = \log p(X|\lambda) = \log \sum_y p(X, y|\lambda)$$

$$= \log \sum_y \frac{p(X, y|\lambda)}{p(y|X, \lambda_{old})} . p(y|X, \lambda_{old})$$

$$= \log E\left[\frac{p(X, y|\lambda)}{p(y|X, \lambda_{old})} |X, \lambda_{old}\right]$$

$$\geq E\left[\log \frac{p(X, y|\lambda)}{p(y|X, \lambda_{old})} |X, \lambda_{old}\right]$$

Dấu = xảy ra khi: $\boldsymbol{\lambda = \lambda_{old}}$ (Jensen inequality)

$$= E[\log p(X, y|\lambda) \,|X, \lambda_{old}] - E[\log p(y|X, \lambda_{old}) \,|X, \lambda_{old}]$$

$$= Q(\lambda|\lambda_{old}) - E[\log p(y|X, \lambda_{old}) \,|X, \lambda_{old}]$$

$$= g(\lambda|\lambda_{old})$$

Có

$$\begin{cases} l(\lambda_{old}; X) = g(\lambda_{old}|\lambda_{old}) \\ l(\lambda; X) > g(\lambda|\lambda_{old}) \ \forall \lambda \neq \lambda_{old} \end{cases}$$

Tìm được λ^* sao cho $g(\lambda^*|\lambda_{old}) > g(\lambda_{old}|\lambda_{old})$
→ $l(\lambda^*; X) > l(\lambda_{old}; X)$

Đưa được bài toán :

$$\lambda = argmax_\lambda l(\lambda; X)$$

Về bài toán:

$$\lambda = argmax_\lambda g(\lambda|\lambda_{old})$$

$$= argmax_\lambda Q(\lambda|\lambda_{old})$$

Tốc độ hội tụ của EM:

- Chậm hơn so với các phương pháp tựa Newton, GD
- Có tính ổn định (ko cần xác định step size như GD)
- Đơn giản

http://home.ustc.edu.cn/~xiaosong/ppt/EM_tutorial.pdf

EM đảm bảo tìm được local MLE. Việc tìm được global MLE hay không phụ thuộc vào giá trị $\boldsymbol{\lambda^0}$ khởi tạo ban đầu

HMM – Semi-supervised Learning

- Xác suất chuyển nhãn: $a_{ij} = p(y_i|y_j)$
- Xác suất sinh từ: $b_{y_j}(x_t) = p(x_t|y_j)$
- Xác suất nhãn ở vị trí đầu tiên: $\pi_i = \overline{p}(y_i)$

Gọi tập toàn bộ tham số của mô hình là $\boldsymbol{\lambda} = \{a_{ij}, b_j(x_t), \pi_i\}$

➢ EM cho HMM:

- E-step: Tính
  - $Q(\lambda|\lambda_{old}) = E[\log p(X, y|\lambda)\,|X, \lambda_{old}]$

$$= \sum_y p(y|X, \lambda_{old}).\log p(X, y|\lambda)$$

$$= \sum_y \left(p(y|X, \lambda_{old}).\left(\sum_{k=1}^{M} \log\left(\pi_{y_1^k}\right) + \sum_{k=1}^{M}\sum_{t=1}^{T} \log\left(b_{y_t^k}(x_t^k)\right) + \sum_{k=1}^{M}\sum_{t=2}^{T} \log\left(a_{y_{t-1}^k y_t^k}\right) \right) \right)$$

- M-step: giải $\lambda = argmax_\lambda Q(\lambda|\lambda_{old})$ sử dụng phương pháp nhân tử Lagrange được nghiệm là:

$$\pi_i = \frac{\sum_{k=1}^{M} P\left(y_1^k = i, X\middle|\lambda_{old}\right)}{\sum_{k=1}^{M} P(X|\lambda_{old})}$$

$$b_i(v) = \frac{\sum_{k=1}^{M}\sum_{t=1\ \&\ x_t^k=v}^{T} P(X, y_t^k = i|\lambda_{old})}{\sum_{k=1}^{M}\sum_{t=1}^{T} P(X, y_t^k = i|\lambda_{old})}$$

$$a_{ij} = \frac{\sum_{k=1}^{M}\sum_{t=1}^{T-1} P(y_t^k = i, y_{t+1}^k = j, X|\lambda_{old})}{\sum_{k=1}^{M}\sum_{t=1}^{T-1} P(y_t^k = i, X|\lambda_{old})}$$

Gần giống với hàm cross-entropy (Phụ lục)

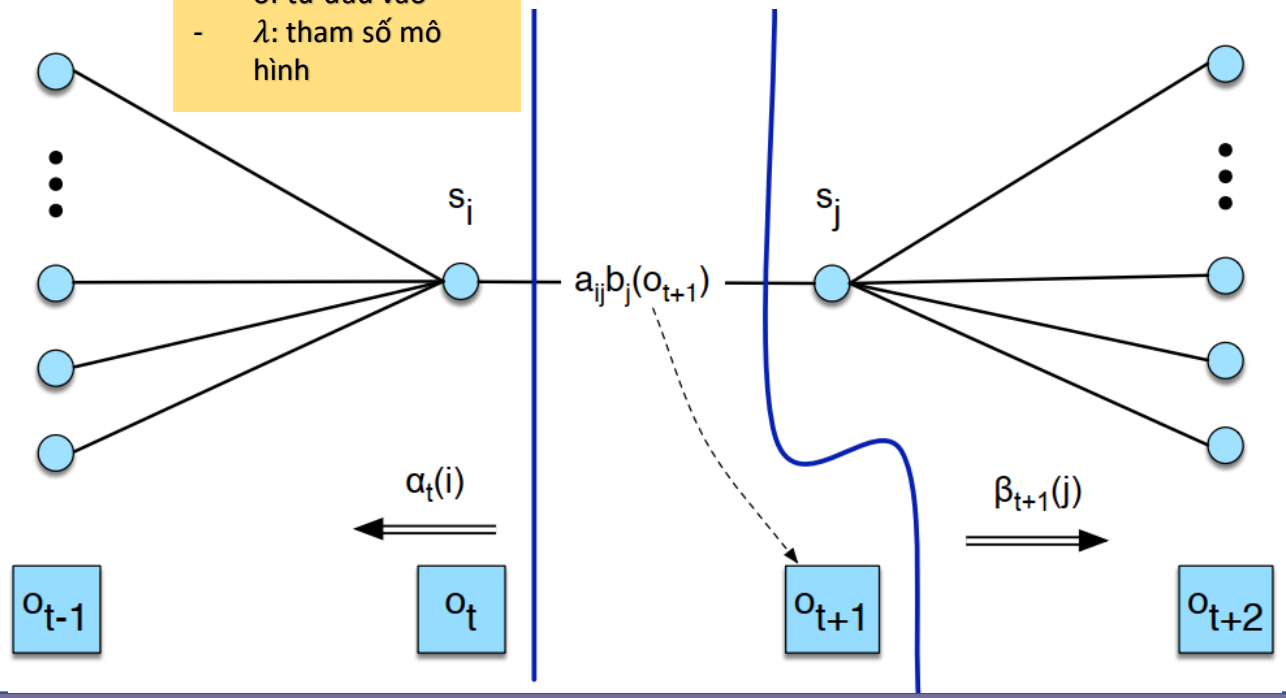
HMM – Semi-supervised Learning

- **Công thức cập nhật a_{ij}:**

$$a_{ij} = \frac{\sum_{k=1}^{M}\sum_{t=1}^{T-1} P(y_t^k = i, y_{t+1}^k = j, X|\lambda_{old})}{\sum_{k=1}^{M}\sum_{t=1}^{T-1} P(y_t^k = i, X|\lambda_{old})}$$

Để tính các xác suất biên $P(y_t^k = i, y_{t+1}^k = j, X|\lambda_{old})$, dùng thuật toán **forward-backward**

Ký hiệu:
- q: nhãn
- o: từ đầu vào
- λ: tham số mô hình



$$P(q_t = i, q_{t+1} = j, O|\lambda) = \alpha_t(i)\, a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

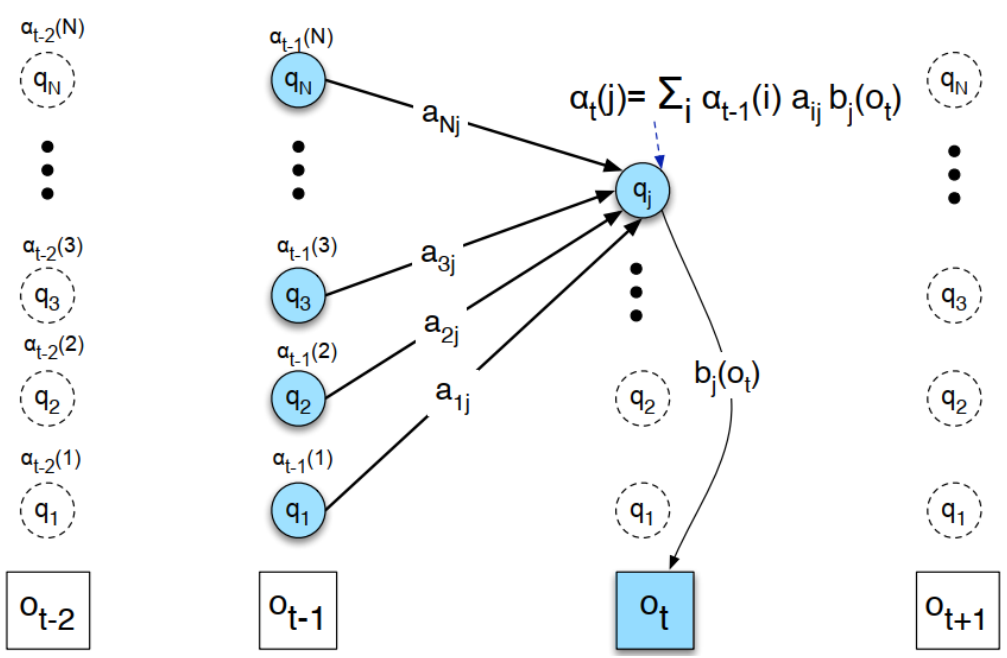
$$\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j|\lambda)$$

$$\beta_t(i) = P(o_{t+1}, o_{t+2} \ldots o_T|q_t = i, \lambda)$$

HMM – Semi-supervised Learning

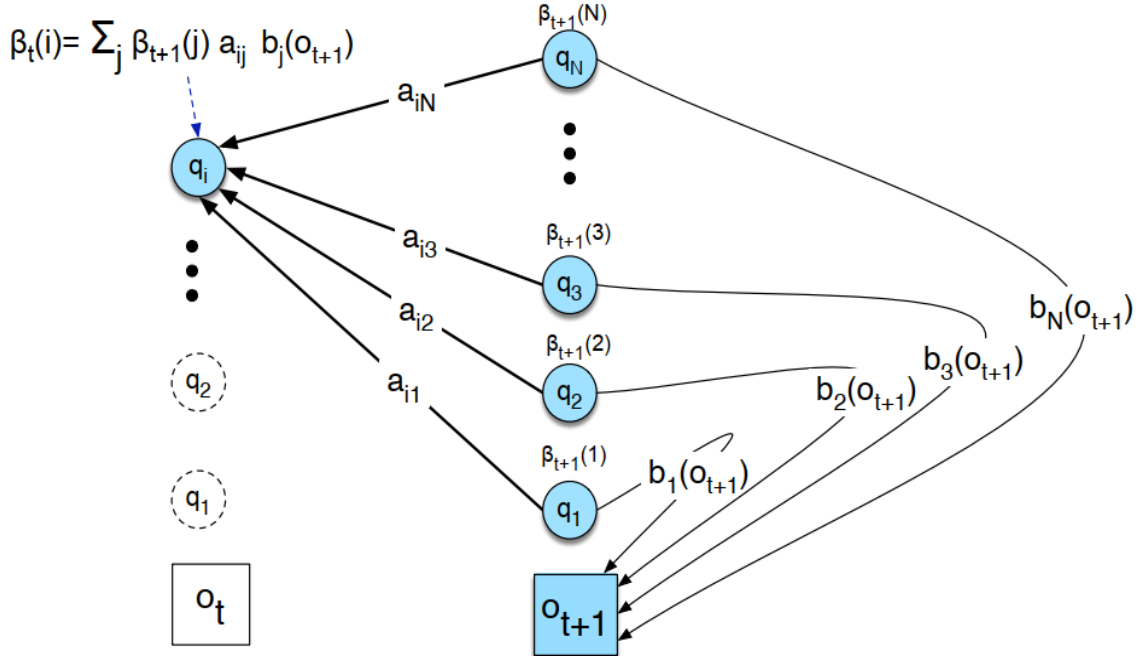
$$a_{ij} = \frac{\sum_{k=1}^{M} \sum_{t=1}^{T-1} P(y_t^k = i, y_{t+1}^k = j, X|\lambda_{old})}{\sum_{k=1}^{M} \sum_{t=1}^{T-1} P(y_t^k = i, X|\lambda_{old})}$$

$$\alpha_t(j) = P(o_1, o_2 \ldots o_t, q_t = j|\lambda)$$



Forward algorithm

$$\beta_t(i) = P(o_{t+1}, o_{t+2} \ldots o_T|q_t = i, \lambda)$$



Backward algorithm

HMM – Semi-supervised Learning

- **Công thức cập nhật $b_i(x_t = v)$, π_i:**

$$\pi_i = \frac{\sum_{k=1}^{M} P\left(y_1^k = i, X \middle| \lambda_{old}\right)}{\sum_{k=1}^{M} P(X|\lambda_{old})}$$

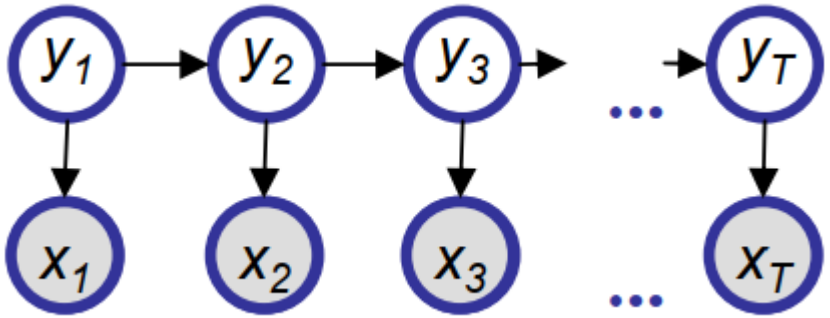
$$b_i(v) = \frac{\sum_{k=1}^{M} \sum_{t=1 \,\&\, x_t^k = v}^{T} P(X, y_t^k = i|\lambda_{old})}{\sum_{k=1}^{M} \sum_{t=1}^{T} P(X, y_t^k = i|\lambda_{old})}$$

→ Vẫn sử dụng thuật toán forward – backward để tính

HMM – Shortcomings

- HMM mô hình hóa được **sự phụ thuộc cục bộ:**

$$p(x, y) = \prod_{i=1}^{T} p(y_i|y_{i-1}).p(x_i|y_i)$$



Với POS tag / NER thì những đặc trưng của từ ở xa vẫn có thể ảnh hưởng đến nhãn của từ hiện tại

- $\boldsymbol{p(y|x)}$ phải tính gián tiếp qua p(x, y). HMM phải mô hình hóa cả $p(x_i|y_i)$

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

Content

- NER Problem
- Machine Learning Approaches for NER
  - Compare Generative vs Discriminative Models
  - Generative model : HMM
  - Discriminative model : MaxEnt Model
    - MaxEnt Model Overview:
      - Intuition
      - Features Function
      - Models
    - Maximum Entropy Markov Model (MEMM)
    - Conditional Random Fields (CRF)
- Reference

Maximum Entropy Model (MaxEnt)

➢ Ý tưởng của MaxEnt

Input → **{ features }** → **Tổ hợp tuyến tính các features** → **Dự đoán**

X $\quad F = \{f_1, \dots f_k\} \quad \sum_i w_i f_i \quad p(c|x) = \frac{1}{Z}\exp(\sum_i w_i f_i)$

Lý do dùng hàm exp: VD với bài toán phân loại nhị phân
$p(c|x) \in [0,1]$ cần được tính dựa trên $\sum_i w_i f_i \in (-\infty, +\infty)$
Để đưa 2 vế của phép gán về cùng miền giá trị, cần tính như sau:

$$\ln\left(\frac{p(y=true|x)}{1-p(y=true|x)}\right) = w \cdot f \rightarrow \begin{cases} p(y=\text{true}|x) = \dfrac{e^{w\cdot f}}{1+e^{w\cdot f}} \\ p(y=false|x) = \dfrac{1}{1+e^{w\cdot f}} \end{cases}$$

Mở rộng với phân loại nhiều nhãn →

$$p(c|x) = \frac{\exp\left(\sum_{i=0}^{N} w_{ci} f_i\right)}{\sum_{c' \in C} \exp\left(\sum_{i=0}^{N} w_{c'i} f_i\right)}$$

MaxEnt – Features

- **Feature Function**
  - Các features sẽ được trích rút bởi **feature function**
  - Giá trị của feature function: $\in \{0, 1\}$

$$f_1(c, \boldsymbol{x}, i) = \begin{cases} 1, & isCapital(\boldsymbol{x}_i)\ and\ c = B - LOC \\ 0, & còn\ lại \end{cases}$$

$$f_2(c, \boldsymbol{x}, \boldsymbol{i}) = \begin{cases} 1, & x_{i-1} = \text{"ông"}\ and\ isCapital(x_i)\ and\ c = B - PER \\ 0, & còn\ lại \end{cases}$$

$$p(c|x) = \frac{\exp\left(\sum_{i=0}^{N} w_{ci} f_i(c,x)\right)}{\sum_{c' \in C} \exp\left(\sum_{i=0}^{N} w_{c'i} f_i(c',x)\right)} \longrightarrow \hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|x)$$

Tổng_Công_ty	B-ORG
Đầu_tư	I-ORG
phát_triển	I-ORG
hạ_tầng	I-ORG
và	I-ORG
đầu_tư	I-ORG
tài_chính	I-ORG
Việt_Nam	?????

Cần phải tự định nghĩa các thuộc tính (isCapital, x_{i-1},…) và các features (tổ hợp thuộc tính)

MaxEnt – Models

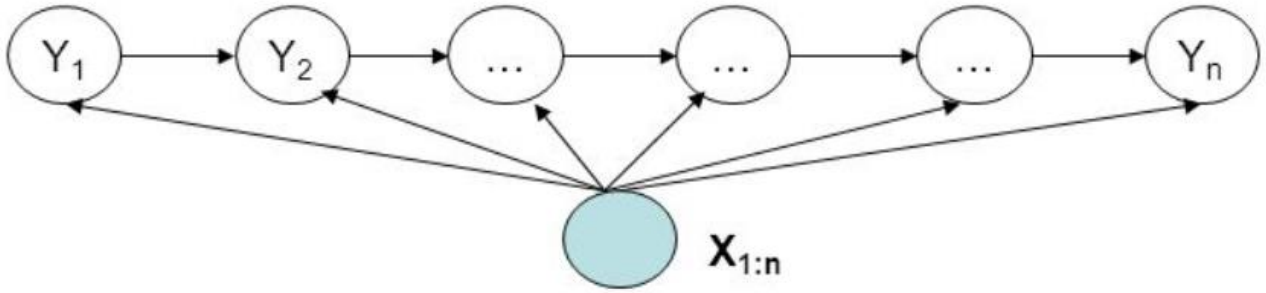
- **Các mô hình thuộc họ MaxEnt Model:**
  - **Classification:**
    - Logistic Regression – binary classification
    - Multinomial Logistic Regression - multiclass classification
  - **Sequence Labeling:**
    - Maximum Entropy Markov Model (MEMM)
    - Conditional Random Fields (CRF)

Content

- NER Problem
- Machine Learning Approaches for NER
  - Compare Generative vs Discriminative Models
  - Generative model : HMM
  - Discriminative model : MaxEnt Model
    - MaxEnt Model Overview
    - Maximum Entropy Markov Model (MEMM):
      - Overview
      - Shortcoming: Label bias problem
    - Conditional Random Fields (CRF)
- Reference

Maximum Entropy Markov Model (MEMM)

- **MEMM :** là mô hình MaxEnt dùng cho bài toán Sequence Labeling

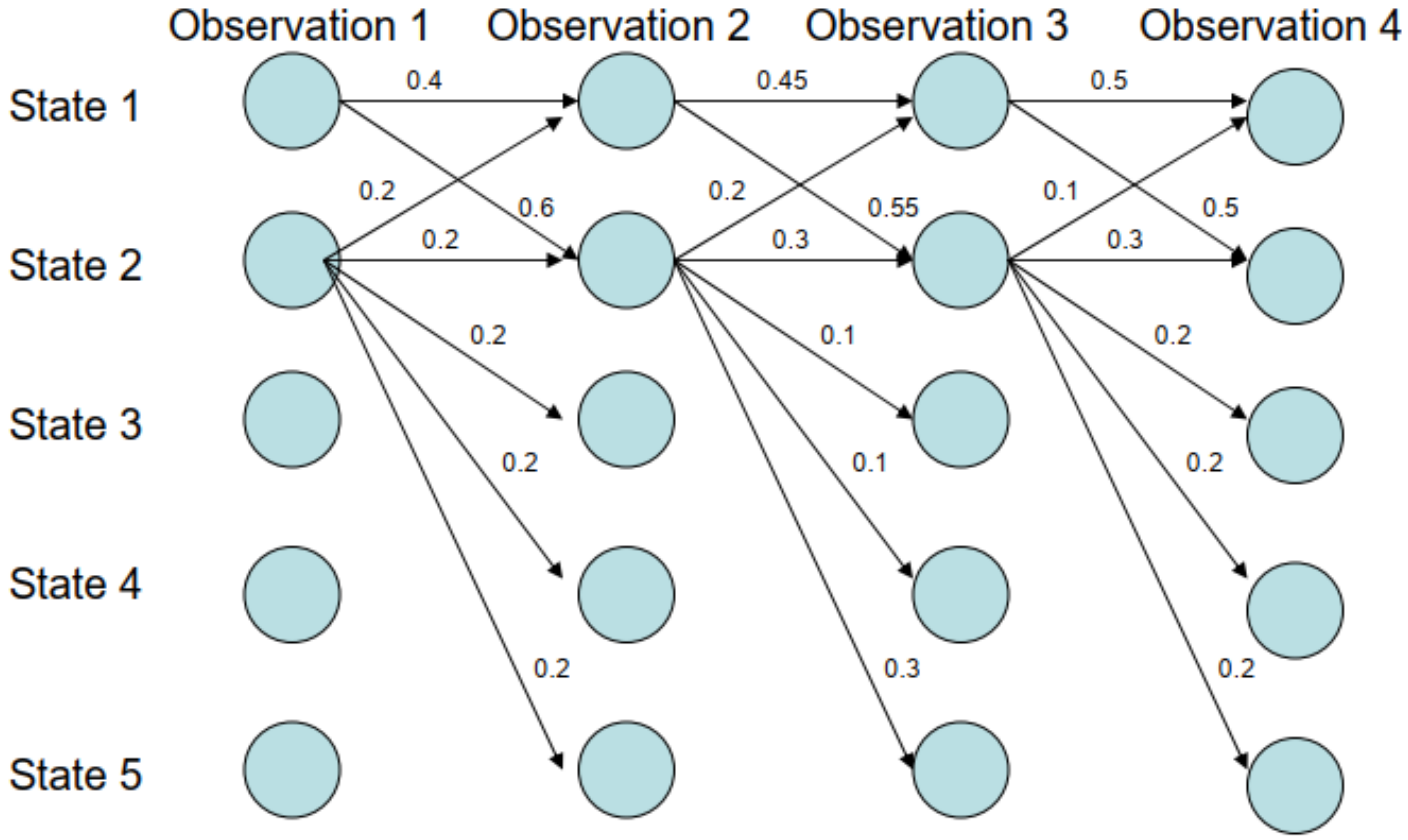


$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \prod_{i=1}^{n} P(y_i|y_{i-1},\mathbf{x}_{1:n}) = \prod_{i=1}^{n} \frac{\exp(\mathbf{w}^T\mathbf{f}(y_i,y_{i-1},\mathbf{x}_{1:n}))}{Z(y_{i-1},\mathbf{x}_{1:n})}$$

→ Giải quyết được cả 2 vấn đề của HMM (phụ thuộc cục bộ & tính toán gián tiếp qua p(x,y))

MEMM – Label bias problem

➢ **Quá trình suy diễn trong MEMM**



$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \prod_{i=1}^{n} P(y_i|y_{i-1}, \mathbf{x}_{1:n})$$

Probability of path 1-> 1-> 1-> 1:

- 0.4 x 0.45 x 0.5 = 0.09

Probability of path 2->2->2->2 :

- 0.2 X 0.3 X 0.3 = 0.018

Probability of path 1->2->1->2:

- 0.6 X 0.2 X 0.5 = 0.06

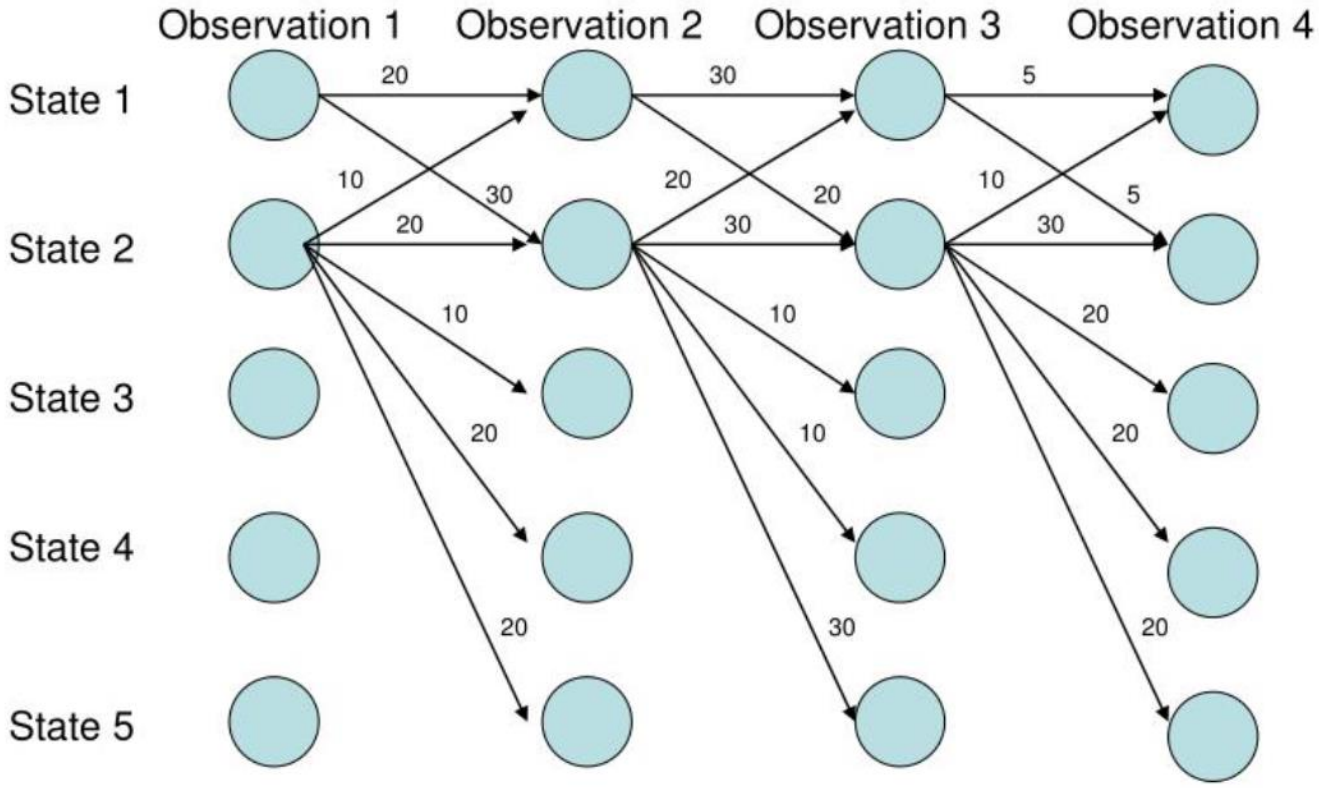
Probability of path 1->1->2->2:

- 0.4 X 0.55 X 0.3 = 0.066

Trong khi đáng ra : path 1-> 2-> 2 -> 2 : mới là tốt nhất

MEMM – Label bias problem

- **Quá trình suy diễn trong MEMM (Viterbi)**



$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \prod_{i=1}^{n} P(y_i|y_{i-1}, \mathbf{x}_{1:n})$$

Mô hình thiên về các **state có ít cạnh** đi ra từ nó hơn

→ Không normalize cục bộ

$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \prod_{i=1}^{n} \frac{\exp(\mathbf{w}^T \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_{1:n}))}{Z(y_{i-1}, \mathbf{x}_{1:n})}$$

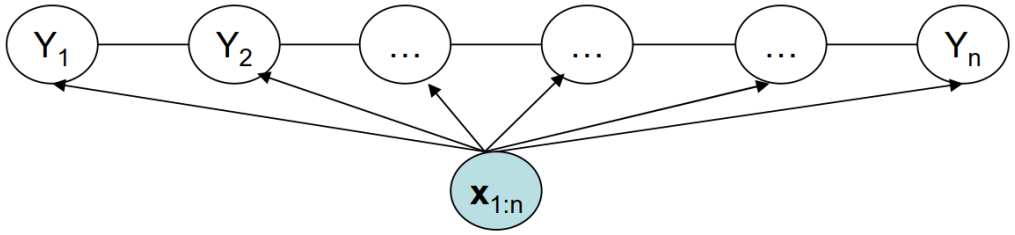
→ Giải pháp : **Conditional Random Fields**

Content

- NER Problem
- Machine Learning Approaches for NER
  - Compare Generative vs Discriminative Models
  - Generative model : HMM
  - Discriminative model : MaxEnt Model
    - MaxEnt Model Overview
    - Maximum Entropy Markov Model (MEMM)
    - Conditional Random Fields (CRF):
      - Overview
      - Inference: y*, Z, $p(y_{m-1}, y_m|x)$
      - Learning
      - Regularization
- Reference

Conditional Random Fields

- **Conditional Random Field (CRF)**



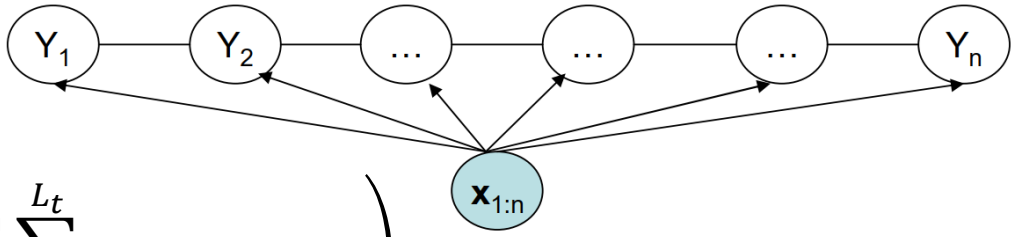
$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n})}\prod_{i=1}^{n}\phi(y_i, y_{i-1}, \mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n})}\prod_{i=1}^{n}\exp(\mathbf{w}^T\mathbf{f}(y_i, y_{i-1}, \mathbf{x}_{1:n}))$$

ϕ : potential factor (thành phần cấu tạo của Markov network)

- Bayesian Network: đồ thị có hướng; cạnh ~ xác suất có điều kiện
- Markov Netwok: đồ thị vô hướng; cạnh ~ potential factor

Conditional Random Fields

- **Công thức chung:**



$$P(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{i=1}^{n} \sum_{k=1}^{L_o} \lambda_k h_k(y_{i-1}, y_i, x, i) + \sum_{i=1}^{n} \sum_{k=1}^{L_t} \mu_k g_k(y_i, x, i) \right)$$

Viết gọn lại:

$$P(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{i=1}^{n} \sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i) \right)$$

Với :

$$Z(x) = \sum_{y \in \boldsymbol{Y}} \exp\left(\sum_{i=1}^{n} \sum_{k=1}^{L_o} \lambda_k h_k(y_{i-1}, y_i, x, i) + \sum_{i=1}^{n} \sum_{k=1}^{L_t} \mu_k g_k(y_i, x, i) \right)$$

n: Số lượng các đặc trưng
T: Số từ trong câu cần gán nhãn
Y: Tập các khả năng gán nhãn

CRF – Inference

- Trong quá trình dự đoán & học của CRF, các **đại lượng cần được tính toán** bao gồm:
  - Chuỗi nhãn dự đoán: $y^* = argmax_{y \in Y} p(y|x, \theta)$
  - Xác suất biên : $p(y_t, y_{t-1}|x)$ (phục vụ cho việc học của mô hình)
  - Đại lượng chuẩn hóa: $Z(x)$

Độ phức tạp để tính các đại lượng trên theo cách brute-force là không khả thi

→ Sử dụng **các thuật toán quy hoạch động** để giảm độ phức tạp

CRF – Inference

- **Chọn chuỗi nhãn tốt nhất cho đầu vào x:**

Tức là cần tìm $\boldsymbol{y^* = argmax_{y \in Y} p(y|x, \theta)}$ với $|Y|^n$ khả năng gán nhãn

$$y^* = argmax_{y \in Y^n} p(y|x, \theta)$$

$$= argmax_{y \in Y^n} \frac{1}{Z(x)} \exp\left(\sum_{i=1}^{n}\sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i)\right)$$

$$= argmax_{y \in Y^n} \left(\sum_{i=1}^{n}\sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i)\right)$$

n: Số lượng các đặc trưng
T: Số từ trong câu cần gán nhãn
Y: Tập nhãn

→ Sử dụng thuật toán Viterbi

CRF – Inference

➢ Thuật toán Viterbi

$$\boldsymbol{y}^* = argmax_{y \in Y^n} \left(\sum_{i=1}^{n} \sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i) \right)$$

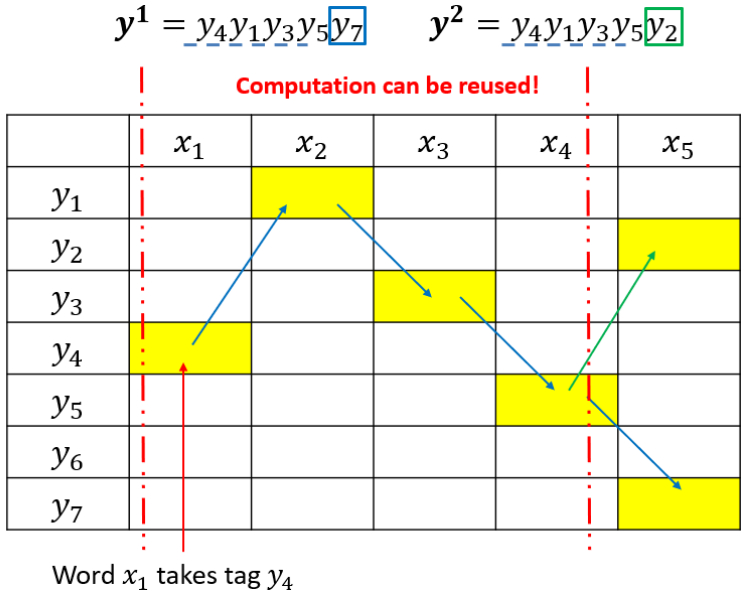
Đặt $g_i(y_{i-1}, y_i) = \sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i)$, ta được:

$$\boldsymbol{y}^* = argmax_{y \in Y^n} \left(\sum_{i=1}^{n} g_i(y_{i-1}, y_i) \right)$$

Điểm của nhãn s ở vị trí thứ m:

$$\delta_m(s) \triangleq \max_{\{y_1, \cdots, y_{m-1}\}} \left[\sum_{i=1}^{m-1} g_i(y_{i-1}, y_i) + g_m(y_{m-1}, s) \right]$$

g_i ~ cạnh
δ : giá trị max của các đường đi tới đỉnh đó



Để tìm chuỗi nhãn tối ưu, Viterbi thực hiện:

- Gán nhãn cho từng vị trí (**trái -> phải**)
- Dùng **quy hoạch động**: Dựa trên các kết quả tính trước đó để tính điểm cho nhãn ở vị trí hiện tại
- Lưu lại các kết quả trung gian để **quay lui lại tìm chuỗi nhãn tối ưu**

CRF – Inference

➢ **Thuật toán Viterbi**

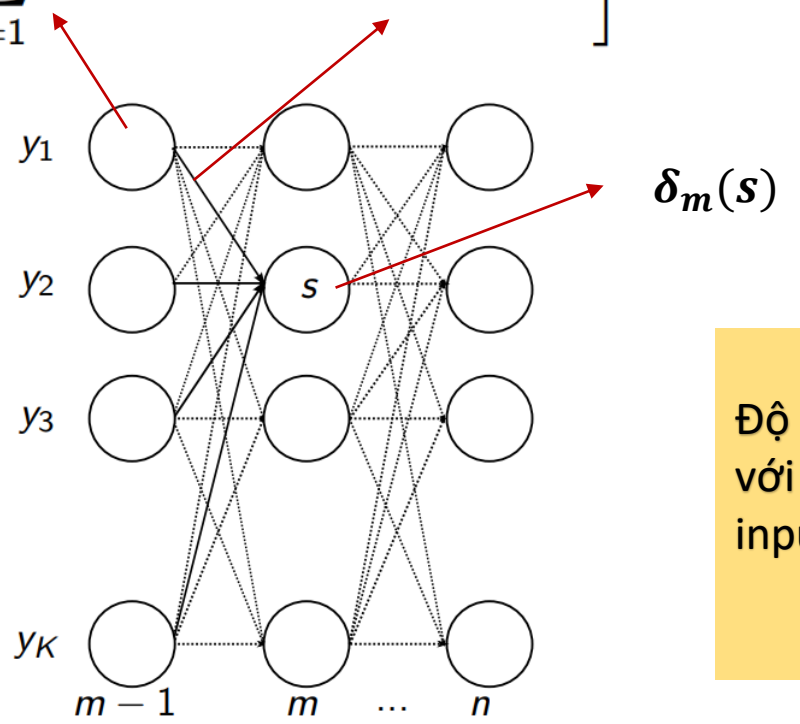
$$\delta_m(s) \triangleq \max_{\{y_1,\cdots,y_{m-1}\}} \left[\sum_{i=1}^{m-1} g_i(y_{i-1}, y_i) + g_m(y_{m-1}, s) \right]$$

Khởi tạo:

$$\delta_1(s) = g_1(y_0, s);\ \forall s \in \mathcal{Y};\ y_0 = \text{start}$$

Đệ quy:

$$\delta_m(s) = \max_{y \in \mathcal{Y}} [\delta_{m-1}(y) + g_m(y, s)]$$



Độ phức tạp: $O(K^2 n)$
với K: số lượng nhãn, n độ dài input

CRF – Inference

- **Tính Z(x):**

$$Z(x) = \sum_{y \in \boldsymbol{Y}} \exp\left(\sum_{i=1}^{n}\sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i)\right)$$

Dùng thuật toán **forward** hoặc **backward** giúp giảm độ phức tạp :

$$O(K^n) \rightarrow O(K^2 n)$$

n: Số lượng các đặc trưng
T: Số từ trong câu cần gán nhãn
Y: Tập các khả năng gán nhãn
$K = |Y|$: số lượng nhãn

CRF – Inference

- **Tính Z(x) dùng thuật toán forward:**

$$Z(x) = \sum_{y \in \boldsymbol{Y}} \exp\left(\sum_{i=1}^{n} \sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i) \right)$$

Đặt: $$M_i(y_{i-1}, y_i, x) = \exp\left(\sum_{k=1}^{L} \theta_k f_k(y_{i-1}, y_i, x, i) \right)$$

$$Z(x) = \sum_{y \in \boldsymbol{Y}} \prod_{\boldsymbol{i=1}}^{\boldsymbol{n}} M_i(y_{i-1}, y_i, x)$$

n: Số lượng các đặc trưng
T: Số từ trong câu cần gán nhãn
Y: Tập các khả năng gán nhãn
$K = |Y|$: số lượng nhãn

CRF – Inference

➢ **Tính Z(x) dùng thuật toán forward:**

$$Z(x) = \sum_{y \in Y} \prod_{i=1}^{n} M_i(y_{i-1}, y_i, x)$$

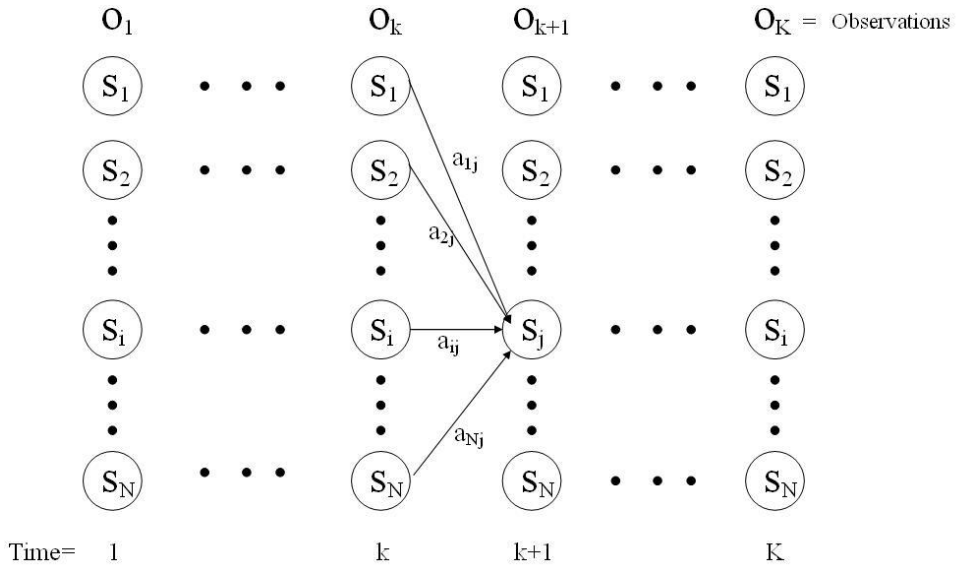
$$\alpha_1(y_1) = M_1(y_0, y_1)$$

$$\alpha_2(y_2) = \sum_{y_1 \in \mathcal{Y}} M_2(y_1, y_2)\alpha_1(y_1)$$

$$\alpha_3(y_3) = \sum_{y_2 \in \mathcal{Y}} M_3(y_2, y_3)\alpha_2(y_2)$$

$$\vdots$$

$$\alpha_m(y_m) = \sum_{y_{m-1}} M_m(y_{m-1}, y_m)\alpha_{m-1}(y_{m-1})\,;\ 2 \le m \le n$$



Khi tính xong với m từ 1 -> n, $Z(x)$ được tính như sau:

$$\sum_{y_n \in \mathcal{Y}} \alpha_n(y_n) = \sum_{\underline{y} \in \mathcal{Y}^n} \prod_{i=1}^{n} M_i(y_{i-1}, y_i, \underline{x}) = Z(\underline{x}, \theta).$$

Forward algorithm : cơ bản giống Viterbi.

Chỉ khác ở chỗ: Viterbi lấy **max** còn Forward lấy **sum**

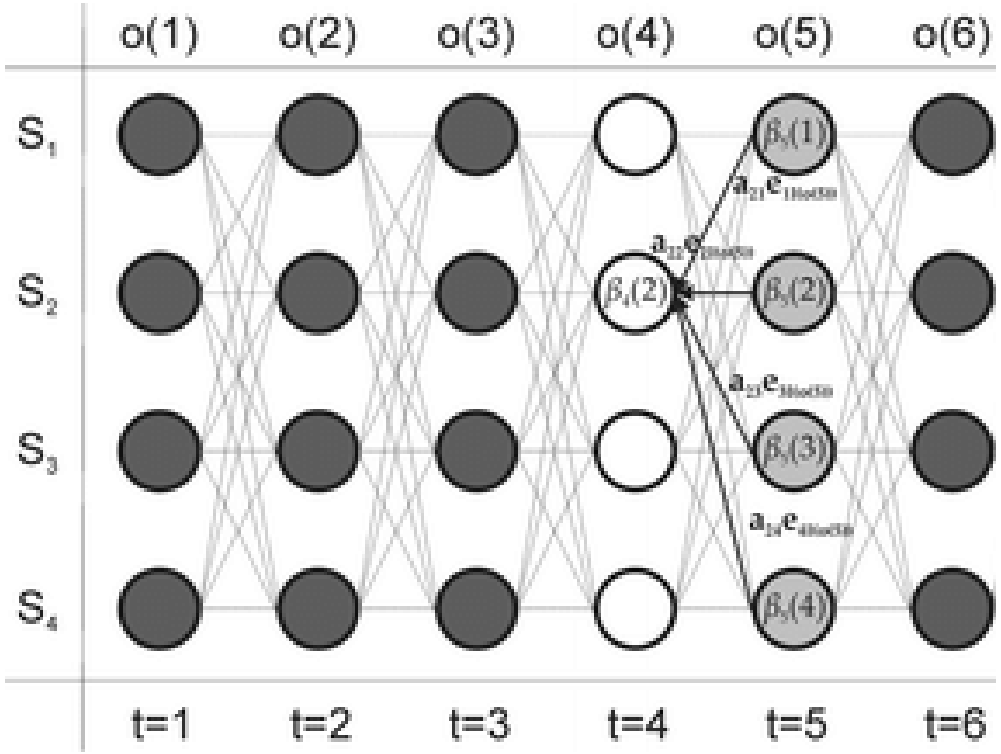
CRF – Inference

- **Tính Z(x) dùng thuật toán backward:**

$$Z(x) = \sum_{y \in Y} \prod_{i=1}^{n} M_i(y_{i-1}, y_i, x)$$

$$\beta_m(y_m) = \sum_{y_{m+1} \in \mathcal{Y}} M_{m+1}(y_m, y_{m+1}) \beta_{m+1}(y_{m+1}) \; ; \; 1 \leq m \leq n-1$$

$$\beta_n(y_n) = 1$$



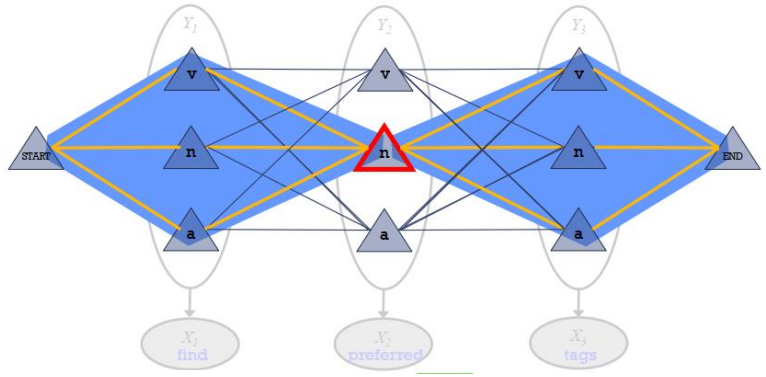
Khi tính xong với m từ n->1, $Z(x)$ được tính như sau:

$$Z(\underline{x}, \theta) = \sum_{y_1 \in \mathcal{Y}} M_1(y_0, y_1) \beta_1(y_1).$$

Giống với forward nhưng theo chiều ngược lại

CRF – Inference

- **Tính xác suất biên : $p(y_{m-1}, y_m|x)$**



Hình minh họa việc tính xác suất biên $p(y_2|x)$

$$p(y_{m-1}, y_m|\underline{x}) = \sum_{\underline{y}\setminus\{y_{m-1},y_m\}} p(\underline{y}|\mathbf{x})$$

$$= \frac{1}{Z(\underline{x})} M_m(y_{m-1}, y_m, \underline{x}) \times \sum_{\{y_1,\cdots,y_{m-2}\}} \prod_{i=1}^{m-1} M_i(y_{i-1}, y_i, \underline{x})$$

$$\times \sum_{\{y_{m+1},\cdots,y_n\}} \prod_{i=m+1}^{n} M_i(y_{i-1}, y_i, \underline{x})$$

Sử dụng thuật toán **forward-backward** để tính:

$$p(y_{m-1}, y_m|\underline{x}) = \frac{1}{Z(\underline{x})} \alpha_{m-1}(y_{m-1}) M_m(y_{m-1}, y_m, \underline{x}) \beta_m(y_m).$$

CRF – Learning

- Việc ước lượng tham số (học) của CRF được thực hiện dựa trên phương pháp **Maximum Likelihood**

Tập dữ liệu huấn luyện:

$$\mathcal{D} = \{(\underline{\mathbf{x}}^{(1)}, \underline{y}^{(1)}), \cdots, (\underline{\mathbf{x}}^{(N)}, \underline{y}^{(N)})\}.$$

Hàm mục tiêu: **Negative Log-likelihood**

$$L(\mathcal{D};\boldsymbol{\theta}) \triangleq -\sum_{q=1}^{N} \log p(\underline{y}^{(q)}|\underline{\mathbf{x}}^{(q)};\boldsymbol{\theta})$$

$$= \sum_{q=1}^{N}\left\{\log Z(\underline{\mathbf{x}}^{(q)};\boldsymbol{\theta}) - \sum_{i=1}^{n_q}\sum_{j=1}^{D}\theta_j f_j(y_{i-1}^{(q)}, y_i^{(q)}, \underline{\mathbf{x}}^{(q)}, i)\right\}$$

$$= \sum_{q=1}^{N}\left\{\log Z(\underline{\mathbf{x}}^{(q)};\boldsymbol{\theta}) - \sum_{j=1}^{D}\theta_j F_j(\underline{\mathbf{x}}^{(q)}, \underline{y}^{(q)})\right\}.$$

$$= \sum_{q=1}^{N}\sum_{y\in \boldsymbol{Y}\setminus\{\underline{y}^{(q)}\}}\sum_{j=1}^{D}\theta_j F_j(\underline{\mathbf{x}}^{(q)}, \underline{y}) \longrightarrow \textbf{Hàm lồi}$$

D: số lượng các đặc trưng

Phân biệt **likelihood** $\mathcal{L}$ vs **probability** P :

- Likelihood $\boldsymbol{\mathcal{L}(\theta:x)}$: Đo độ tốt của mô hình dựa trên dữ liệu huấn luyện $\boldsymbol{x}$
- Probability $\boldsymbol{P(x|\theta)}$: Xác suất để mô hình tạo ra được $\boldsymbol{x}$

Phương pháp Maximum Likelihood: xấp xỉ tham số θ của mô hình sao cho:

$$\boldsymbol{\mathcal{L}(\theta:x) = P(x|\theta)}$$

Tức là, mô hình cần học tham số để dữ liệu sinh ra từ mô hình khớp với dữ liệu huấn luyện nhất có thể

$$p(y|x) = \frac{1}{Z(x)}\exp\left(\sum_{i=1}^{n}\sum_{k=1}^{L}\theta_k f_k(y_{i-1}, y_i, x, i)\right)$$

$$Z(x) = \sum_{\underline{y}\in \boldsymbol{Y}}\sum_{j=1}^{D}\theta_j F_j(\underline{\mathbf{x}}^{(q)}, \underline{y})$$

CRF – Learning

- **Tính Gradient và cập nhật tham số:**

$$L(\mathcal{D};\boldsymbol{\theta}) = \sum_{q=1}^{N}\left\{\log Z(\underline{\mathbf{x}}^{(q)};\boldsymbol{\theta}) - \sum_{j=1}^{D}\theta_j F_j(\underline{\mathbf{x}}^{(q)},\underline{y}^{(q)})\right\}$$

$$\frac{\partial L(\mathcal{D};\boldsymbol{\theta})}{\partial\theta_k} = \sum_{q=1}^{N}\left\{\frac{\partial}{\partial\theta_k}\log Z(\underline{\mathbf{x}}^{(q)};\boldsymbol{\theta}) - F_k(\underline{\mathbf{x}}^{(q)},\underline{y}^{(q)})\right\}$$

$$\frac{\partial}{\partial\theta_k}\log Z(\underline{\mathbf{x}};\boldsymbol{\theta}) = \frac{1}{Z(\underline{\mathbf{x}};\boldsymbol{\theta})}\sum_{\underline{y}\in\mathcal{Y}^n}\frac{\partial}{\partial\theta_k}\left[\exp\sum_{j=1}^{D}\theta_j F_j(\underline{\mathbf{x}},\underline{y})\right]$$

$$= \frac{1}{Z(\underline{\mathbf{x}};\boldsymbol{\theta})}\sum_{\underline{y}\in\mathcal{Y}^n}F_k(\underline{\mathbf{x}},\underline{y})\exp\sum_{j=1}^{D}\theta_j F_j(\underline{\mathbf{x}},\underline{y})$$

$$= \sum_{\underline{y}\in\mathcal{Y}^n}F_k(\underline{\mathbf{x}},\underline{y})\frac{\exp\sum_j\theta_j F_j(\underline{\mathbf{x}},\underline{y})}{Z(\underline{\mathbf{x}};\boldsymbol{\theta})}$$

$$= \sum_{\underline{y}\in\mathcal{Y}^n}F_k(\underline{\mathbf{x}},\underline{y})p(\underline{y}|\underline{\mathbf{x}};\boldsymbol{\theta})$$

$$\sum_{\underline{y}\in\mathcal{Y}^n}F_k(\underline{\mathbf{x}},\underline{y})p(\underline{y}|\underline{\mathbf{x}};\boldsymbol{\theta}) = \sum_{i=1}^{n}\sum_{\underline{y}\in\mathcal{Y}^n}f_k(y_{i-1},y_i,\underline{\mathbf{x}})p(\underline{y}|\underline{\mathbf{x}};\boldsymbol{\theta})$$

$$= \sum_{i=1}^{n}\sum_{y_{i-1},y_i\in\mathcal{Y}^2}f_k(y_{i-1},y_i,\underline{\mathbf{x}})p(y_{i-1},y_i|\underline{\mathbf{x}};\boldsymbol{\theta})$$

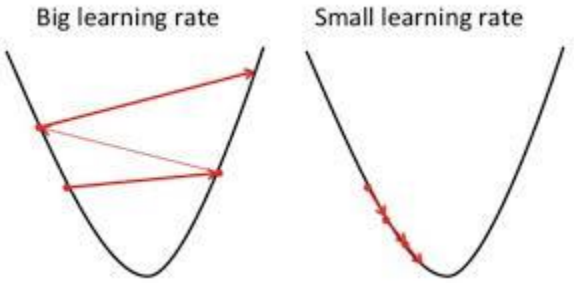
Tính được nhờ thuật toán forward - backward

$$p(y_{i-1},y_i|\underline{\mathbf{x}}) = \frac{1}{Z(\underline{\mathbf{x}})}\alpha_{i-1}(y_{i-1})M_i(y_{i-1},y_i,\underline{\mathbf{x}})\beta_i(y_i).$$

CRF – Learning

➢ **Thuật toán cập nhật tham số theo gradient:**

- Generalized iterative scaling: độ chính xác cao nhưng tối ưu chậm
- Gradient Descent : Cần chọn learning rate phù hợp
- Stochastic Gradient Descent : Hỗ trợ online learning
- Conjugate Gradient : hội tụ nhanh
- **Limited-memory BFGS (L-BFGS)** : hội tụ nhanh, áp dụng được cho dữ liệu lớn



Vấn đề chọn learning rate trong gradient descent

https://dl.acm.org/doi/pdf/10.3115/1118853.1118871 : các phương pháp tối ưu cho hàm log likelihood

CRF – Learning

- **Thuật toán cập nhật tham số theo gradient:**

Dataset	Method	KL Div.	Acc	Iters	Evals	Time
rules	gis	5.124×10^{-2}	47.00	1186	1187	16.68
	iis	5.079×10^{-2}	43.82	917	918	31.36
	steepest ascent	5.065×10^{-2}	44.88	224	350	4.80
	conjugate gradient (fr)	5.007×10^{-2}	44.17	66	181	2.57
	conjugate gradient (prp)	5.013×10^{-2}	46.29	59	142	1.93
	limited memory variable metric	5.007×10^{-2}	44.52	72	81	1.13
lex	gis	1.573×10^{-3}	46.74	363	364	31.69
	iis	1.487×10^{-3}	42.15	235	236	95.09
	steepest ascent	3.341×10^{-3}	42.92	980	1545	114.21
	conjugate gradient (fr)	1.377×10^{-3}	43.30	148	408	30.36
	conjugate gradient (prp)	1.893×10^{-3}	44.06	114	281	21.72
	limited memory variable metric	1.366×10^{-3}	43.30	168	176	20.02
summary	gis	1.857×10^{-3}	96.10	1424	1425	107.05
	iis	1.081×10^{-3}	96.10	593	594	188.54
	steepest ascent	2.489×10^{-3}	96.33	1094	3321	190.22
	conjugate gradient (fr)	9.053×10^{-5}	95.87	157	849	49.48
	conjugate gradient (prp)	3.297×10^{-4}	96.10	112	537	31.66
	limited memory variable metric	5.598×10^{-5}	95.54	63	69	8.52
shallow	gis	3.314×10^{-2}	14.19	3494	3495	21223.86
	iis	3.238×10^{-2}	5.42	3264	3265	66855.92
	steepest ascent	7.303×10^{-2}	26.74	3677	14527	85062.53
	conjugate gradient (fr)	2.585×10^{-2}	24.72	1157	6823	39038.31
	conjugate gradient (prp)	3.534×10^{-2}	24.72	536	2813	16251.12
	limited memory variable metric	3.024×10^{-2}	23.82	403	421	2420.30

Source: A comparison of algorithms for maximum entropy parameter estimation

https://dl.acm.org/doi/pdf/10.3115/1118853.1118871 : các phương pháp tối ưu cho hàm log likelihood

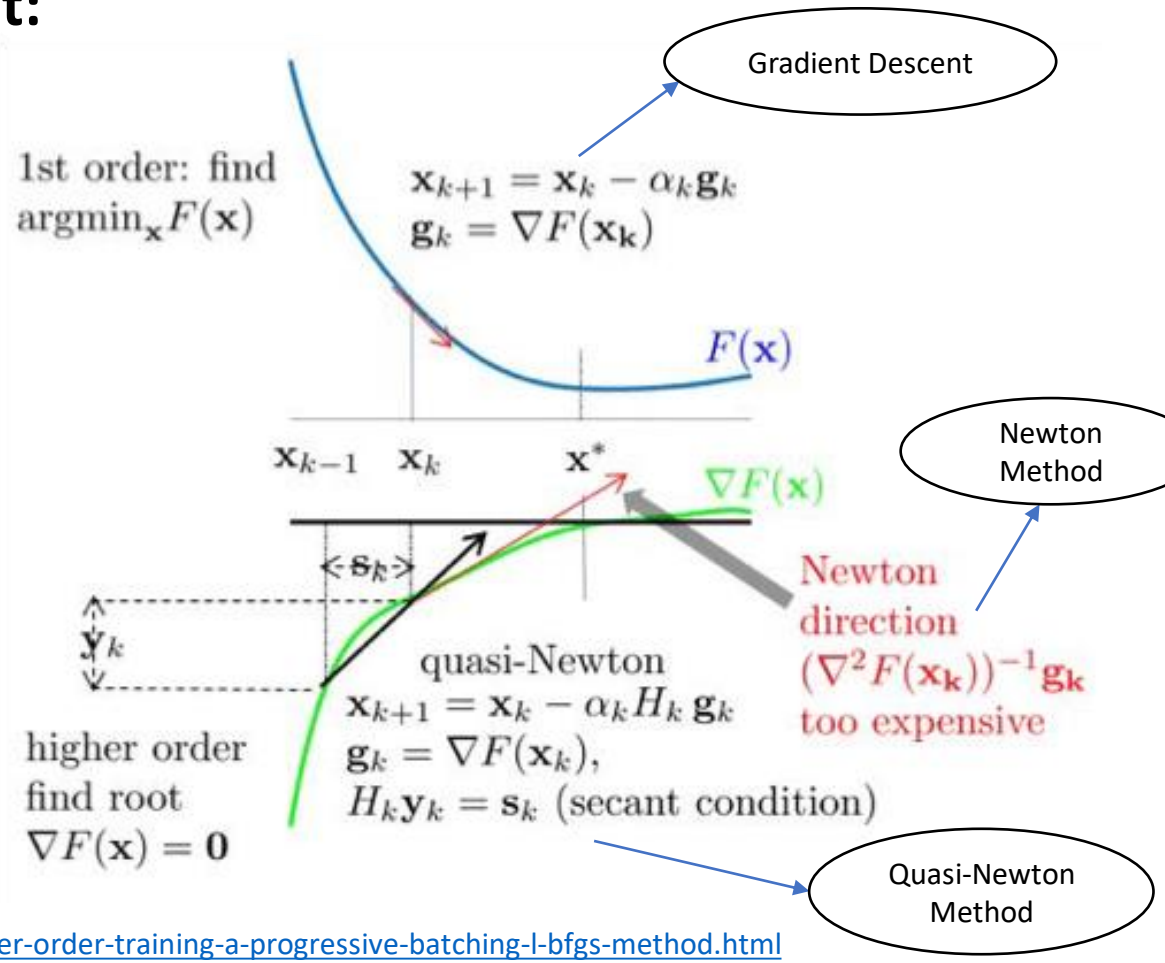
CRF – Learning

➢ Thuật toán cập nhật tham số theo gradient:

- Thuật toán Limited-memory BFGS (L-BFGS):
  - Thuộc họ Quasi-Newton (giúp đơn giản việc tính toán $(\nabla^2 F(x_k))^{-1}$ so với phương pháp Newton thuần túy)
  - Các phương pháp Quasi-Newton có các cách khác nhau để tính ma trận $H_k = g(H_{k-1}, \nabla f(x_{k-1}), \nabla f(x_k)$ $\sim (\nabla^2 F(x_k))^{-1}$
  - BFGS: Tính H_k tính dựa trên $H_0, \dots, H_{k-1}$
  - L-BFGS: Tính H_k tính dựa trên $H_{k-m}, \dots, H_{k-1}$

→ **Được sử dụng nhiều** để tối ưu các bài toán Maximum Likelihood do:

- Hội tụ nhanh (đặc điểm của phương pháp Newton cho hàm tuyến tính)
- Áp dụng được với dữ liệu nhiều chiều (Limited-memory)



https://www.intel.com/content/www/us/en/artificial-intelligence/posts/toward-higher-order-training-a-progressive-batching-l-bfgs-method.html

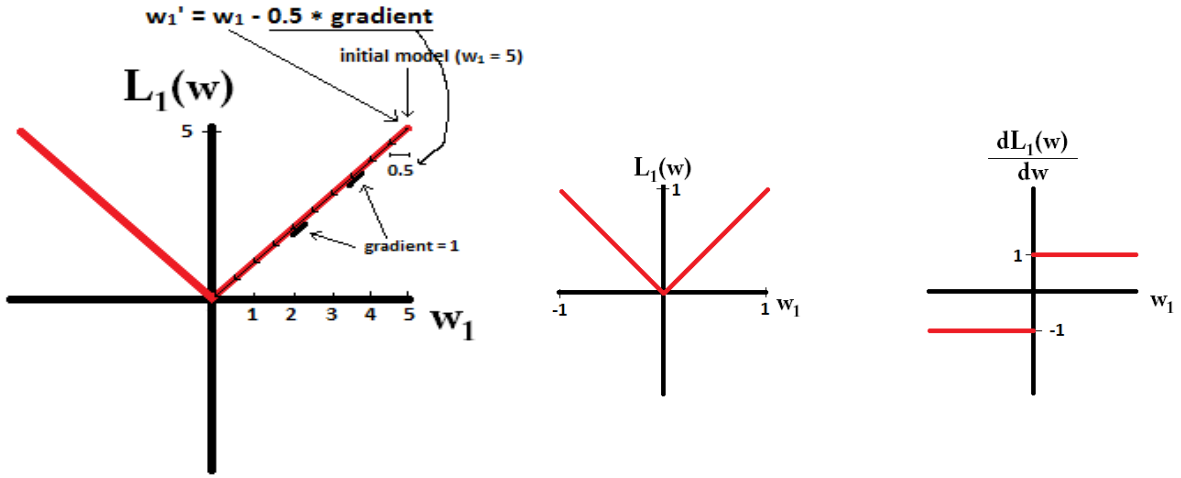
CRF – Regularization

➢ L1- Regularization

K: số lượng các đặc trưng

$$L(D;\theta) = L(D;\theta) + \lambda \sum_{j=1}^{K} |\theta_j|$$

→ Chọn lọc đặc trưng

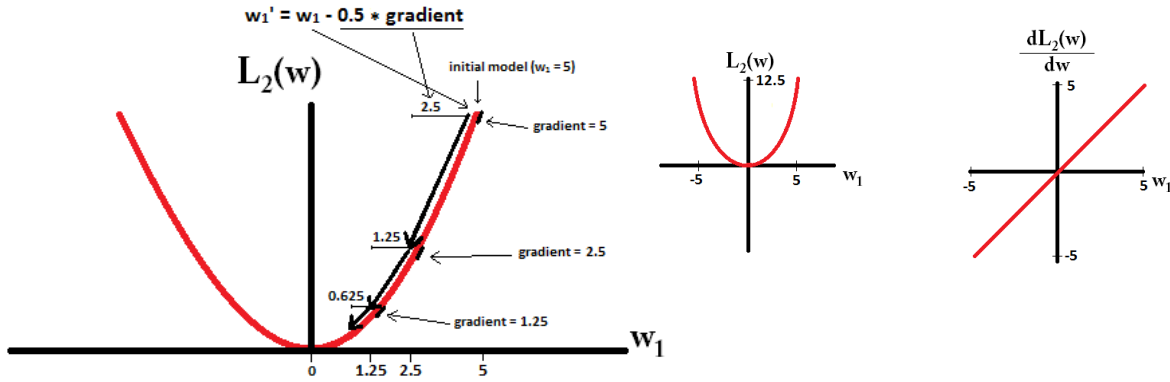


Cập nhật θ với các bước như nhau do $\frac{dL_1(w)}{dw} = \pm 1$ → Có nhiều khả năng $\exists\, j \mid \theta_j = 0$

➢ L2- Regularization

$$L(D;\theta) = L(D;\theta) + \lambda . \left|\left|\theta\right|\right|_2^2$$

→ Tránh overfitting



θ càng bé -> bước cập nhật càng nhỏ do $\frac{dL_2(w)}{dw} \sim \theta$ → Khó để θ_j nhận giá trị 0

https://stats.stackexchange.com/questions/45643/why-l1-norm-for-sparse-models

References

- Generative model vs Discriminative model:
  - Intuition: https://medium.com/@mlengineer/generative-and-discriminative-models
  - Compare models: *An Introduction to Conditional Random Fields* https://arxiv.org/abs/1011.4088
- Hidden Markov Model:
  - HMM for POS tag: *Speech and Language Processing. Daniel Jurafsky & James H. Martin, Chapter A: Hidden Markov Models & Chapter 8: Sequence Labeling for Parts of Speech and Named Entity*
  - HMM full overview: Hidden Markov Models Part 2: Posterior Decoding, Learning
  - Training strategies for HMM:
- EM algorithm:
  - EM idea: https://www.cs.cmu.edu/~epxing/Class/10701-08s/recitation/em-hmm.pdf &
  - Why Does the EM Algorithm Work: http://www.columbia.edu/~mh2078/MachineLearningORFE/EM_Algorithm.pdf
  - Apply EM for HMM: https://f.hubspotusercontent40.net/hubfs/8111846/Unicon_October2020/pdf/bilmes-em-algorithm.pdf
- MaxEnt Model:
  - Idea: *Hidden Markov And Maximum Entropy Models*: https://www.cs.jhu.edu/~jason/papers/jurafsky+martin.bookdraft07.ch6.pdf
  - How MaxEnt Models improve HMM's shortcomings: http://www.cs.cmu.edu/~epxing/Class/10708-07/Slides/lecture12-CRF-HMM-annotation.pdf

References

- CRF:
  - Detail*: An Introduction to Conditional Random Fields* https://arxiv.org/abs/1011.4088
  - CRF detail easier to understand: *A Tutorial On Conditional Random Fields With Applications To Music Analysis* https://perso.telecom-paristech.fr/essid/teach/CRF_tutorial_ISMIR-2013.pdf
  - Training algorithms for CRF: *Shallow Parsing with Conditional Random Fields* *https://www.aclweb.org/anthology/N03-1028.pdf* *& Efficient Training of Conditional Random Fields* *http://dirichlet.net/pdf/wallach02efficient.pdf*
- Regularization in CRF:
  - *Maximum entropy sequence models, Smoothing* https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1162/handouts/MaxentTutorial-16x9-Smoothing-6up.pdf
  - *Why L1 norm for sparse models*: https://stats.stackexchange.com/questions/45643/why-l1-norm-for-sparse-models

Appendix

- Generative vs Discriminative Models
- Probabilistic Graphical Models:
  - Graphic Model
  - Bayesian Network (Directed Graphic Model)
  - Markov Network (Undirected Graphic Model)
  - Conditional Independence
- Maximum Likelihood Estimator (MLE) vs Maximum a Posteriori estimation (MAP)
- Maxent Model, Log-linear model, feature function, smoothing
- Prior probability vs posterior probability
- Optimization:
  - L-BFGS
  - Conjugate Gradient
  - Newton Method, Quasi Newton Method
  - EM
- Regularization: L1, L2


Thanks for Watching