	VIETTEL AI RACE	Public 103
	HUẤN LUYỆN VÀ SUY LUẬN LINEAR-CHAIN CRFS	Lần ban hành: 1

1. Giải thích:

Tương tự CRFs, Linear-Chain CRFs phân loại chuỗi dựa trên xác suất $P(Y|X)$. Với chuỗi x cho trước, CRFs sẽ tìm ra chuỗi y sao cho xác suất $P(Y = y|X = x)$ là lớn nhất.

$$\hat{y} = argmax_y P(y|x)$$

Xác suất $P(Y|X)$ được xây dựng thông qua việc định nghĩa các hàm đặc trưng $f_k$và g_k và xác định giá trị λ_k, μ_k. Các trọng số được tối ưu trong quá trình huấn huyện với tập dữ liệu huấn luyện. Nói cách khác, quá trình huấn luyện CRFs là quá trình học phân phối xác suất $P(Y|X)$ của tập dữ liệu huấn luyện.

Việc tối ưu hóa các trọng số $\theta = (\lambda_1, \dots, \lambda_k; \mu_1, \dots, \mu_k)$ tương đương với việc tìm kiếm hàm năng lượng tối ưu cho mô hình. Mô hình CRFs sẽ điều chỉnh các trọng số để hàm đặc trưng phản ánh chính xác mối quan hệ giữa chuỗi quan sát và chuỗi nhãn, từ đó đưa ra dự đoán chính xác nhất. Do đó, hàm đặc trưng đóng vai trò then chốt trong việc xác định mối quan hệ giữa chuỗi quan sát x và chuỗi nhãn y. Việc lựa chọn và thiết kế hàm đặc trưng phù hợp với bài toán cụ thể là rất quan trọng để đảm bảo mô hình có thể học được các mẫu quan trọng từ dữ liệu và đưa ra dự đoán chính xác.

2. Huấn luyện

Việc huấn luyện thường sử dụng phương pháp MLE (Maximum Likelihood Estimation) để tối ưu hóa các trọng số $\theta = (\lambda_1, \dots, \lambda_k; \mu_1, \dots, \mu_k)$ từ tập huấn luyện $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$. Mục tiêu của quá trình huấn luyện là tìm ra bộ trọng số θ để hàm mục tiêu log-likelihood $L(\theta)$ là lớn nhất.

$$L(\theta) = \sum_{i=1}^{N} log\left(P_\theta\left(y^{(i)}|x^{(i)}\right)\right)$$

$$= \sum_{i=1}^{N}\left(\sum_{t=1}^{n}\left(\sum_{k}\lambda_k f_k\left(y_{t-1}^{(i)}, y_t^{(i)}, x^{(i)}, t\right) + \sum_{k}\mu_k g_k\left(y_t^{(i)}, x^{(i)}, t\right)\right) - log\left(Z_\theta\left(x^{(i)}\right)\right)\right)$$

Việc tối ưu hóa hàm mục tiêu có thể sử dụng các phương pháp tối ưu dựa trên việc tính gradient như Gradient Descent, Stochastic Gradient Descent (SGD), L-BFGS (Limited-memory BFGS). Do đó chúng ta cần tính gradient của $L(\theta)$.

$$\frac{\partial L}{\partial \lambda_k} = \sum_{i=1}^{N}\left(\sum_{t=1}^{n} f_k\left(y_{t-1}^{(i)}, y_t^{(i)}, x^{(i)}, t\right) - \frac{1}{Z_\theta(x^{(i)})}\frac{\partial Z_\theta}{\partial \lambda_k}\right)$$

$$\frac{1}{Z_\theta(x^{(i)})}\frac{\partial Z_\theta}{\partial \lambda_k} = \frac{1}{Z_\theta(x^{(i)})}\sum_{y'\in\Omega_y}\left(\sum_{t=1}^{n} f_k\left(y'_{t-1}, y'_t, x^{(i)}, t\right) e^{-E\left(x^{(i)}, y'\right)}\right)$$

$$= \sum_{y'\in\Omega_y}\left(\sum_{t=1}^{n} f_k\left(y'_{t-1}, y'_t, x^{(i)}, t\right)\frac{e^{-E\left(x^{(i)}, y'\right)}}{Z_\theta(x^{(i)})}\right)$$

$$= \sum_{y'\in\Omega_y}\left(\sum_{t=1}^{n} f_k\left(y'_{t-1}, y'_t, x^{(i)}, t\right) P\left(y'\middle|x^{(i)}\right)\right)$$

$$= \sum_{t=1}^{n}\sum_{y'\in\Omega_y} f_k\left(y'_{t-1}, y'_t, x^{(i)}, t\right)P\left(y'\middle|x^{(i)}\right)$$

$$\Rightarrow \frac{\partial L}{\partial \lambda_k} = \sum_{i=1}^{N}\sum_{t=1}^{n}\left(f_k\left(y_{t-1}^{(i)}, y_t^{(i)}, x^{(i)}, t\right) - \sum_{y'\in\Omega_y} f_k\left(y'_{t-1}, y'_t, x^{(i)}, t\right)P\left(y'\middle|x^{(i)}\right)\right)$$

Gọi $E_x(f_k)$ là kì vòng hàm đặc trưng f_k theo phân phối xác suất $P(y|x)$:

$$E_x(f_k) = \sum_{t=1}^{n} \sum_{y' \in \Omega_y} f_k(y'_{t-1}, y'_t, x^{(i)}, t) P(y'|x^{(i)})$$

$$\Rightarrow \frac{\partial L}{\partial \lambda_k} = \sum_{i=1}^{N} \left(\sum_{t=1}^{n} f_k\left(y_{t-1}^{(i)}, y_t^{(i)}, x^{(i)}, t\right) + E_{x^{(i)}}(f_k) \right)$$

Tương tự ta cũng có gradient cho μ_k:

$$E_x(g_k) = \sum_{t=1}^{n} \sum_{y' \in \Omega_y} g_k(y'_t, x^{(i)}, t) P(y'|x^{(i)})$$

$$\frac{\partial L}{\partial \mu_k} = \sum_{i=1}^{N} \left(\sum_{t=1}^{n} g_k\left(y_t^{(i)}, x^{(i)}, t\right) + E_{x^{(i)}}(g_k) \right)$$

Nếu tính trực tiếp kì vọng của các hàm đặc trưng từ công thức trên thì độ phức tạp tính toán sẽ là hàm mũ $(O(n \times |\mathcal{Y}|^n))$. Do đó không khả thi khi số lượng nhãn và bộ dữ liệu lớn. Để giảm độ phức tạp tính toán ta biến đổi công thức trên thành dạng sau:

$$E_x(f_k) = \sum_{t=1}^{n} \sum_{y' \in \Omega_y} f_k(y'_{t-1}, y'_t, x^{(i)}, t) P(y'|x^{(i)})$$

$$= \sum_{t=1}^{n} \sum_{y', y'' \in \mathcal{Y}} f_k(y', y'', x^{(i)}, t) P(Y_{t-1} = y', Y_t = y''|x^{(i)})$$

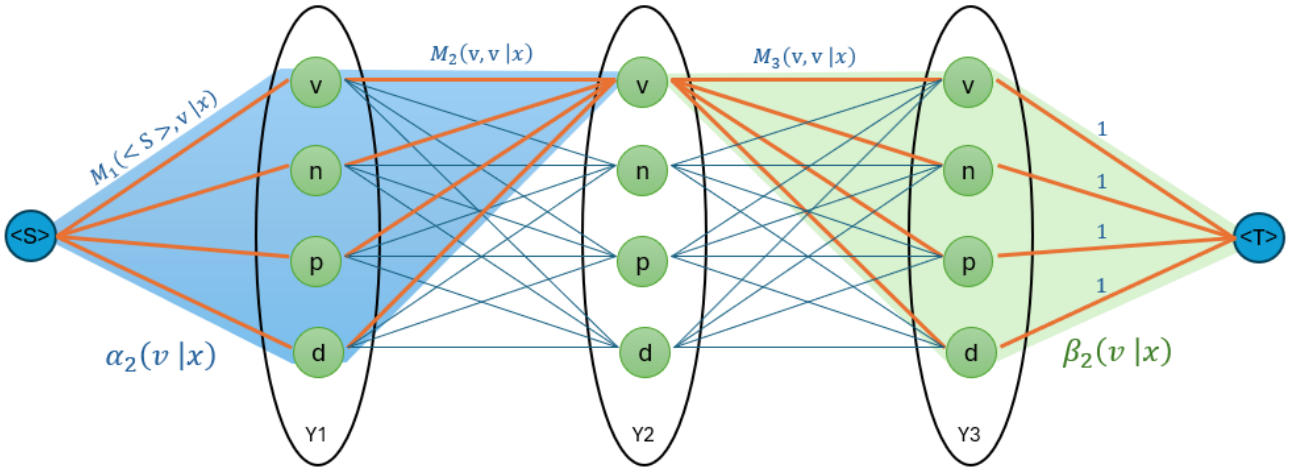
Trong đó $P(Y_{t-1} = y', Y_t = y''|x^{(i)})$ là xác xuất biên của $Y_{t-1} = y', Y_t = y''$ khi biết chuỗi quan sát $x^{(i)}$, tức xác suất để cặp nhãn (y', y'') được gán tại vị trí

t-1 và t khi biết $x^{(i)}$ mà không quan tâm đến các nhãn còn lại. Xác suất biên này có thể được tính trong thời gian đa thức bằng thuật toán Forward-Backward.

Tương tự cho μ_k:

$$E_x(g_k) = \sum_{t=1}^{n} \sum_{y' \in y} g_k(y', x^{(i)}, t) P(Y_t = y' | x^{(i)})$$

3. Thuật toán Forward-Backward áp dụng trong tính gradient



Hình 3.1. Minh họa thuật toán Forward-Backward trong việc xác suất biên tại 1 nút

Ý tưởng của thuật toán Forward-Backward là tính xác suất biên dựa vào việc tính xác suất tiến $\alpha_i(x)$ và xác suất lùi $\beta_i(x)$. Hình 8 mô tả ý tưởng tính xác suất biên $P(Y_2 = v|x)$ và hình 9 mô tả ý tưởng cách tính xác suất biên $P(Y_{t-1} = y', Y_t = y''|x)$ cho bài toán POS. Mỗi một đường đi từ <S> đến <T> là 1 trường hợp của chuỗi Y. Trọng số của của mỗi cạnh được tính theo công thức $M_i(C_j, C_k|x)$ đã trình bày ở phần trước thể hiện khả năng nhãn của từ liền kề khi biết trước nhãn, trọng số của 1 đường đi là tích các trọng số cạnh mà đường đi qua.

$$p_\theta(Y = y|X = x) = p_\theta(path_y|X = x) = \frac{Trọng\ số\ của\ path_y}{Z_\theta(x)}$$

Xác suất biên $P(Y_2 = v|x)$ sẽ là tổng xác suất của tất cả các đường đi đi qua v tại Y_2 hay tổng trọng số các đường đi đó. Ta có thể phân tích tổng này thành tích của 2 tổng $\alpha_2(v|x)$ và $\beta_2(v|x)$.

$$P(Y_2 = v|x) = \alpha_2(v|x) \times \beta_2(v|x)$$

Trong đó $\alpha_2(v|x)$ là tổng trọng số tất cả các đường đi từ <S> đến v tại Y_2, $\beta_2(v|x)$ là tổng trọng số tất cả các đường đi từ v tại Y_2 đến <T>.

Để tính $\alpha_2(v|x)$ ta sẽ tính α_1 của tất cả các giá trị của Y1 rồi nhân với trọng số chuyển đổi thành nhãn v tương ứng với từng giá trị (v => v, n => v, p => v, d => v). Như vậy thì α_i sẽ được tính dựa theo α_{i-1} và quá trình này là quá trình tiến của thuật toán Forward-Backward. Tương tự β_i cũng được tính toán dựa trên quy hoạch động và quá trình này là quá trình lùi.

Tổng quát, ta có chuỗi $Y = (Y_0, \dots, Y_n)$, gọi $Y_{i:j} = (Y_i, \dots, Y_j)$ $với\ 0 \le i < j \le n$.

Ta có:

$$\alpha_t(Y_t = y'|x) = \begin{cases} \sum_{y'_{0:t-1}} \prod_{i=1}^{t} M_i(y'_{i-1}, y'_i|x), & với\ 1 < t \le n \\ M_1(< Start >, y'_1|x), & với\ t = 1 \end{cases}$$

$$\beta_t(Y_t = y'|x) = \begin{cases} \sum_{y'_{t+1:n}} \prod_{i=t+1}^{n} M_i(y'_{i-1}, y'_i|x), & với\ 1 \le t < n-1 \\ \sum_{y'' \in y} M_t(y', y''|x), & với\ t = n-1 \\ 1, & với\ t = n \end{cases}$$

Ta chứng minh $\alpha_t(Y_t = y'|x) = \sum_{y'_{t-1} \in y} \alpha_{t-1}(Y_{t-1} = y'_{t-1}|x) \times M_t(y'_{t-1}, y'|x)$ với $1 < t \le n$, thật vậy:

$$\alpha_t(Y_t = y'|x) = \sum_{y'_{0:t-2}} \sum_{y'_{t-1} \in \mathcal{Y}} \prod_{i=1}^{t-1} M_i(y'_{i-1}, y'_i|x) \times M_t(y'_{t-1}, y'_t|x)$$

$$= \sum_{y'_{t-1} \in \mathcal{Y}} \left(\sum_{y'_{0:t-2}} \prod_{i=1}^{t-1} M_i(y'_{i-1}, y'_i|x) \right) \times M_t(y'_{t-1}, y'_t|x)$$

$$= \sum_{y'_{t-1} \in \mathcal{Y}} \alpha_{t-1}(Y_{t-1} = y'_{t-1}|x) \times M_t(y'_{t-1}, y'|x)$$

Tương tự, với $1 \leq t < n - 1$ta cũng có:

$$\beta_t(Y_t = y'|x) = \sum_{y'_{t+1} \in \mathcal{Y}} M_{t+1}(y', y'_{t+1}|x) \beta_{t+1}(Y_{t+1} = y'_{t+1}|x)$$

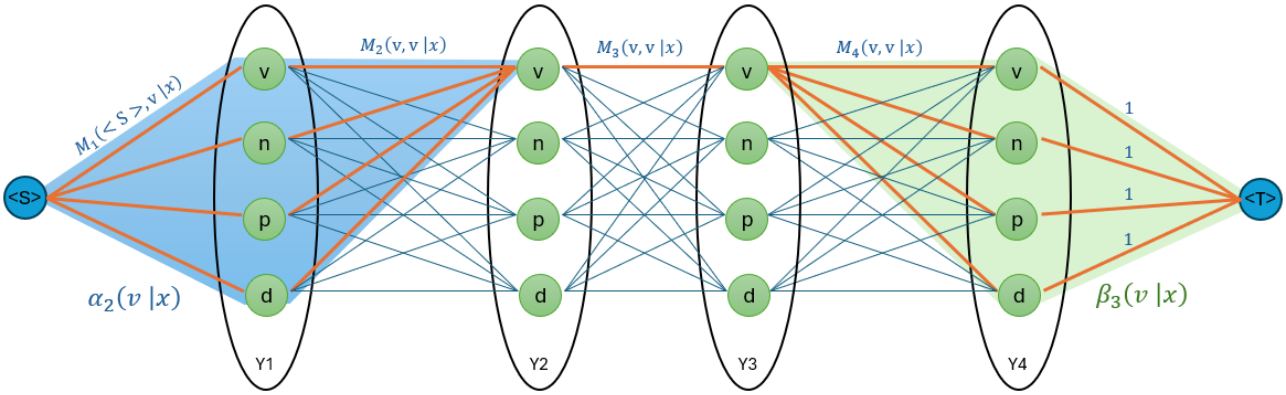
Với cách biểu diễn dưới dạng ma trận công thức $\alpha_t(Y_t = y'|x)$ và $\beta_t(Y_t = y'|x)$ có thể biểu diễn dưới dạng tích ma trận và vector với $M_i(x)_{<Start>}$ là vetor hàng ứng với nhãn $< Start >$, $1_{|y_\prime|\times 1}$ là ma trận các giá trị 1 kích thước $|\mathcal{Y}'| \times 1$:

$$\alpha_t(Y_t = y'|x) = \begin{cases} \left(M_i(x)_{<Start>} \times \prod_{i=2}^{t} M_i(x) \right)_{0,y'} & , với\ 1 < t \leq n \\ M_i(x)_{<Start>,y'} , & với\ t = 1 \end{cases}$$

$$\beta_t(Y_t = y'|x) = \begin{cases} \left(\left(\prod_{i=1}^{t+1} M_i(x) \right) \times 1_{|y_\prime|\times 1} \right)_{y',0} & , với\ 1 \leq t < n \\ 1 , & với\ t = n \end{cases}$$

Công thức xác xuất biên biểu diễn bằng xác suất tiến và lùi có dạng:

$$P(Y_t = y'|x) = \frac{\alpha_t(Y_t = y'|x) \times \beta_t(Y_t = y'|x)}{Z_\theta(x)}$$



Hình 3.2. Minh họa thuật toán Forward-Backward trong việc xác suất biên tại 1 cạnh

Tương tự với xác suất biên $P(Y_{t-1} = y', Y_t = y''|x)$, ta có:

$$P(Y_{t-1} = y', Y_t = y''|x) = \frac{\alpha_{t-1}(Y_{t-1} = y'|x) \times M_t(y', y''|x) \times \beta_t(Y_t = y''|x)}{Z_\theta(x)}$$

Bằng phương pháp quy hoạch động, ta có thể tính các xác suất biên với độ phức tạp $O(n \times |\mathcal{Y}|^2)$ và chính là độ phức tạp khi tính kì vọng của các hàm đặc trưng.

4. Thuật toán Viterbi áp dụng trong suy luận Linear-Chain CRFs

Xác định chuỗi $\hat{y}$ có xác suất xảy ra cao nhất khi biết x:

$$\hat{y} = argmax_y P(y|x) = argmax_y \frac{\prod_{i=1}^{n} M_i(y_{i-1}, y_i|x)}{Z_\theta(x)}$$

Vì $Z_\theta(x)$ là hằng số khi biết nên việc xác định chuỗi $\hat{y}$ có xác suất xảy ra cao nhất khi biết x tương đương với xác định chuỗi $\hat{y}$ để $\prod_{i=1}^{n} M_i(y_{i-1}, y_i|x)$ lớn nhất:

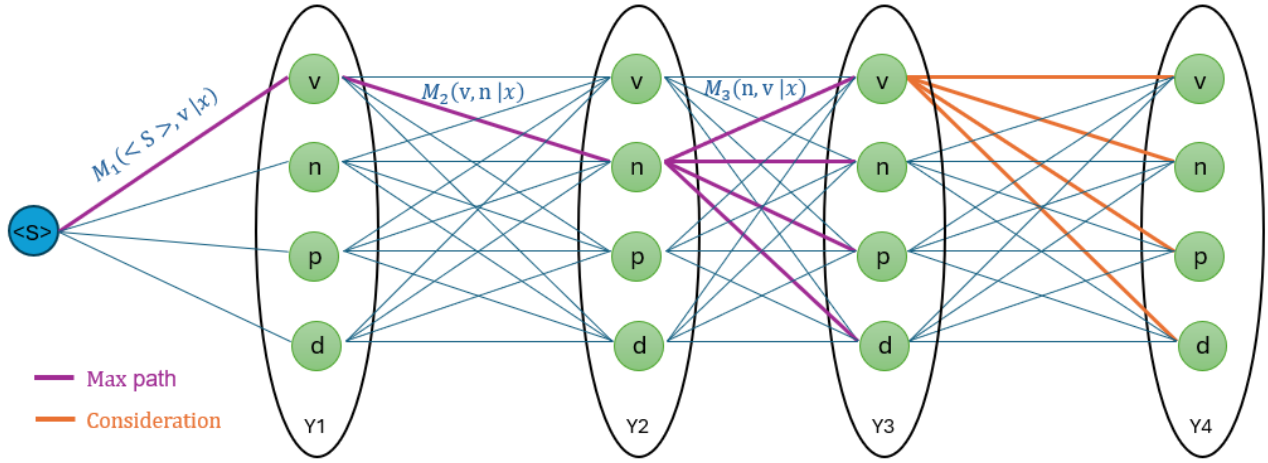
$$\hat{y} = argmax_y \prod_{i=1}^{n} M_i(y_{i-1}, y_i|x)$$

Việc tìm $\hat{y}$ có thể tính trong thời gian $O(n \times |\mathcal{Y}|^2)$ với thuật toán quy hoạch động Viterbi. Thuật toán Viterbi được mô tả bằng mã giả trong hình 10.

```
function Viterbi(trans_list[]={M_1(x), ... M_n(x)}, n_label)
    seq_len = lenght(trans_list)
    P[seq_len][n_label +1] = {0}
    Prev[seq_len][ n_label +1]
    P[0][0] = 1, P_max = 0
    path[seq_len]
    for t = 1 to seq_len - 1 do
        for i = 1 to n_label do
            for j = 0 to n_label do
                new_p = P[t - 1][j] * M_t(x)_{j,i}
                if new_p > prob[t][s] then
                    P[t][i] = new_p
                    prev[t][i] = j

            if t = seq_len – 1 and P[t][i] > P_max then
                P_max = P[t][i]
                path[seq_len - 1] = i

    for t = seq_len - 2 to 0 do
        path[t] ← prev[t + 1][path[t + 1]]

    return path
end
```

Hình 4.1: Thuật toán Viterbi cho suy luận Linear-chain CRFs

$M_i(x)$ là ma trận đã được trình bày trong phần 3 với hàng 0 và cột 0 tương ứng với nhãn <Start>.



Hình 4.2: Hình minh họa thuật toán Viterbi cho POS

Hình 4.2 là minh họa quá trình suy luận Viterbi cho POS. Giả sử sau khi huấn luyện ta đã có được trọng số của các đường đi M_i. Với đầu câu đầu vào có 4 từ, và cần gán nhãn cho 4 từ này một nhãn từ loại là 1 trong 4 giá trị: v, n, p, d. Ở đây, mỗi một miền tương đương với 1 từ cần được gán nhãn và số đỉnh trong miền là nhãn có thể có của từ, ví dụ, miền Y1 có 4 đỉnh là v, n, p, d tương đương với 4 giá trị có thể gán cho từ đầu tiên của câu. Một đường đi hợp lệ là đường đi đi qua duy nhất một đỉnh trong mỗi miền. Thuật toán Viterbi sẽ tìm đường sao cho trọng số là lớn nhất (tương đương với xác suất chuỗi nhãn là lớn nhất.

Ý tưởng của Viterbi là đường đi lớn nhất đến một đỉnh sẽ bao gồm đường đi lớn nhất đến đỉnh trước nó. Xuất phát từ ý tưởng này, để tìm đường đi lớn nhất đến miền Y4, ta sẽ tính đường đi lớn nhất đến các đỉnh của miền Y3, sau đó từ các đỉnh của Y3 ta tính trọng số đến các đỉnh của Y4 và chọn ra đường đi có trọng số lớn nhất. Tương tự đường đi có trọng số lớn nhất đến các đỉnh trong Y3 có thể tính qua đường đi có trọng số lớn nhất đến các đỉnh trong Y2, ….