Spis rzeczy

4 Ukryte modele Markowa **41**
4.1 Algorytm Viterbiego 43
4.2 Prawdopodobieństwo wyemitowania zadanego słowa 44
4.2.1 Algorytm prefiksowy 44
4.2.2 Algorytm sufiksowy 45
4.3 Estymacje parametrów 45
4.3.1 Algorytm Bauma-Welcha 46

4 Ukryte modele Markowa

Zaczniemy od definicji łańcucha Markowa. Będziemy się wyłącznie zajmować skończonymi łańcuchami Markowa działającymi w dyskretnym czasie. Niech $Q \neq \emptyset$, będzie skończonym zbiorem (*zbiór stanów*). Pewien stan $k_0 \in Q$ jest wyróżniony jako *stan początkowy*. *Łańcuch Markowa* jest zadany przez macierz przejść $M = (p_{k,l})_{k,l \in Q}$, która dla $k, l \in Q$ podaje prawdopodobieństwo $p_{k,l}$ przejścia ze stanu k do stanu l. M musi spełniać następujący warunek: dla każdego $k \in Q$ mamy

$$\sum_{l \in Q} p_{k,l} = 1.$$

Łańcuch Markowa opisuje pewien układ , który w każdym momencie może się znajdować tylko w jednym ze stanów $k \in Q$. Układ ten obserwujemy w dyskretnych chwilach czasowych $t = 0, 1, \ldots$. Przyjmujemy, że na początku układ znajduje się w stanie początkowym k_0. Jeśli w danym momencie t, układ znajduje się w stanie k, to w momencie $t + 1$ przechodzi on do stanu l z prawdopodobieństwem $p_{k,l}$. Podstawową cechą łańcucha Markowa jest to, że następny stan zależy tylko od obecnego stanu i nie zależy od wartości t, ani od historii dojścia do obecnego stanu.

Ukryte modele Markowa (HMM, od "hidden Markov models") stanowią pewne rozszerzenie definicji łańcucha Markowa. Niech Σ będzie alfabetem. Będziemy rozważać łańcuchy Markowa mogące się komunikować z otoczeniem poprzez emitowanie ciągów liter z alfabetu Σ. Tak więc mając dany HMM będący w stanie $k \in Q$, emituje symbol $x \in \Sigma$ z prawdopodobieństwem $e_k(x)$ oraz przechodzi do stanu l z prawdopodobieństwem $p_{k,l}$. Ukryte modele

Markowa znane są w literaturze informatycznej również pod nazwą *probabilistycznych automatów z wyjściem.* te Jeśli chcemy aby w każdym stanie $k \in Q$ emitowany był jakiś symbol, to musimy przyjąć $\sum_{x\in\Sigma} e_k(x) = 1$. W ogólności, jeśli dopuszczamy sytuacje, że w pewnych stanach (z pewnym prawdopodobieństwem, nic nie jest emitowane, to przyjmujemy słabsze założenie $\sum_{x\in\Sigma} e_k(x) \leq 1$. Przyjmujemy, że to co daje się obserwować to symbole emitowane przez układ, a nie stany wewnętrzne układu (stąd nazwa "ukryte").

Przykład 4.0.1 Przykładem ukrytego łańcucha Markowa jest nieuczciwe kasyno gry, które ma dwa rodzaje kości: uczciwą kostkę do gry (z prawdopodobieństwem 1/6 wyrzuca się każda z sześciu możliwych wartości) oraz nieuczciwą kostkę (dla której prawdopodobieństwo wyrzucenia szóstki wynosi 1/2, a dla pozostałych liczb 1/10). Tak więc mamy dwa stany: F (uczciwa kostka) i L (nieuczciwa). Układ może zmieniać swój stan z pewnym prawdopodobieństwem, ale my stanu nie możemy zaobserwować. Jedynie widzimy ciąg liczb będących wynikiem rzutów kostką. □

Przykład 4.0.2 (Wyspy CpG) Wiadomo, że w ludzkim genomie, jeśli pojawi się para nukleotydów CG (tzn G pojawia się tuż za C w jednej nici, w kierunku 5' do 3'), to nukleotyd C zwykle ulega zmianie (pod wpływem metylacji) i z dużym prawdopodobieństwem zostaje zmutowany na T. Tak więc pary CpG[1] zwykle pojawiają się rzadziej w genomie człowieka. Z pewnych biologicznych powodów proces metylacji jest zatrzymywany na krótkich odcinkach genomu w pobliżu miejsc rozpoczynających rejony kodujące białko (lub też w pobliżu tzw. odcinków promotorowych). Takie odcinki są nazywane *wyspami CpG.* Rozpoznawanie wysp CpG może być wykorzystane jako wskazówka przy poszukiwaniu biologicznie interesujących fragmentów genomu.

Mamy tutaj do czynienia z ukrytym modelem Markowa. Ma on osiem stanów $A_+, C_+, G_+, T_+, A_-, C_-, G_-, T_-$. Stan A_+ oznacza, że znajdujemy się w rejonie wyspy CpG i czytamy nukleotyd A. Podobnie, stan G_- oznacza, że znajdujemy się poza rejonem wyspy CpG i czytamy G. Emitowane symbole: będąc w stanie A_ξ (gdzie $\xi \in \{-, +\}$), emitujemy A z prawdopodobieństwem 1; każdy inny symbol jest emitowany w stanie A_ξ z prawdopodobieństwem 0. Podobnie dla innych stanów. □

[1]Litera 'p' pomiędzy C oraz G oznacza, że chodzi o kolejne nukleotydy w tej samej nici, a nie odpowiadające sobie pary komplementarnych nukleotydów w DNA.

Niech $S \in \Sigma^*$ oraz $\pi \in Q^*$ będą niepustymi ciągami równej długości $n = |S| = |\pi| > 0$. Prawdopodobieństwo tego że S zostanie wyemitowane oraz układ będzie zmieniał stany według kolejności π wynosi

$$P(S, \pi) = \prod_{t=0}^{n-1} e_{\pi(t+1)} \cdot p_{\pi(t),\pi(t+1)},$$

gdzie w powyższym wzorze przyjmujemy, że $\pi(0) = k_0$, jest stanem początkowym.[2]

4.1 Algorytm Viterbiego

Dane słowo $S \in \Sigma^*$ zaobserwowane na wyjściu HMM. Chcemy znaleźć najbardziej prawopodobny ciąg stanów, który doprowadził do wyemitowania S, czyli znaleźć π^* takie, że

$$P(S, \pi^*) = max\{P(S, \pi) \mid \pi \in Q^*, |\pi| = |S|\}.$$

Zastosujemy metodę dynamicznego programowania. Dla $0 < i \leq |S|$ oraz $k \in Q$, niech $v(i, k)$ będzie prawdopodobieństwem najbardziej optymalnej drogi kończącej się w stanie k, która doprowadziła do wyemitowania prefiksu $S[1..i]$ z maksymalnym prawdopodobieństwem. Tzn.

$$v(i, k) = max\{P(S[1..i], \pi) \mid \pi \in Q^i, \quad \pi(i) = k\}.$$

Funkcję v rozszerzamy dla przypadku $i = 0$, definiując

$$v(0, k) = \begin{cases} 1 & \text{gdy } k = k_0, \\ 0 & \text{gdy } k \neq k_0. \end{cases}$$

Twierdzenie 4.1.1 *Dla $0 < i \leq |S|$ oraz $k \in Q$ zachodzi*

$$v(i, k) = e_k(S(i)) \cdot \max_{l \in Q}[v(i-1, l) \cdot p_{l,k}].$$

Wówczas maksymalne prawdopodobieństwo $P(S, \pi^*)$ znajduje się ze wzoru

$$P(S, \pi^*) = \max_{k \in Q}[v(|S|, k)].$$

[2] Czasami przyjmuje się, że HMM ma wyróżniony stan *końcowy*, do którego układ przechodzi po wyemitowaniu słowa S. Wprowadzenie takiego stanu niewiele zmienia rozważania, więc dla prostoty będziemy go opuszczać.

Optymalną drogę π^* (może ich być więcej niż jedna) znajduje się przez zapamiętywanie wskaźników, dla których maksimum jest realizowane w równaniu rekurencyjnym z Twierdzenia 4.1.1. Twierdzenie to w naturalny sposób prowadzi do algorytmu wyznaczania najbardziej prawdopodobnej ścieżki emitującej zadane słowo. Jest to tzw. *algorytm Viterbiego*.

Twierdzenie 4.1.2 *Algorytm Viterbiego wyznacza najbardziej prawdopodobną ścieżkę emitującą przez HMM dane słowo S w czasie $O(|S|\cdot|Q|^2)$ i przestrzeni $O(|S|\cdot|Q|)$, gdzie Q jest zbiorem stanów modelu HMM.*

4.2 Prawdopodobieństwo wyemitowania zadanego słowa

Zajmiemy się algorytmem obliczania prawdopodobieństwa $P(S)$, wyemitowania słowa S przez dany HMM. Mamy

$$P(S) = \sum_{\pi} P(S, \pi),$$

gdzie w powyższej sumie przyjmujemy, że $P(S,\pi) = 0$ dla dróg π, takich że $|S| \neq |\pi|$.

4.2.1 Algorytm prefiksowy

Dla $0 \leq i \leq |S|$ oraz $k \in Q$ niech $f(i,k)$ będzie prawdopodobieństwem wyemitowania prefiksu $S[1..i]$ przy dodatkowym założeniu, że droga prowadząca do tej emisji kończy się w stanie k. Czyli

$$f(i,k) = \sum_{\{\pi \ \mid \ \pi(i)=k\}} P(S[1..i], \pi).$$

Podobnie jak dla algorytmu Viterbiego mamy

$$f(0,k) = \begin{cases} 1 & \text{gdy } k = k_0, \\ 0 & \text{gdy } k \neq k_0. \end{cases}$$

Poniższe twierdzenie sugeruje pewien sposób obliczania wartości $f(i,k)$ oraz prawdopodobieństwa $P(S)$.

Twierdzenie 4.2.1 *Dla $0 < i \leq |S|$ oraz $k \in Q$ mamy*

$$f(i,k) = e_k(i) \cdot \sum_{l \in Q} f(i-1, l) \cdot p_{l,k}.$$

Prawdopodobieństwo wyemitowania słowa S wynosi wówczas

$$P(S) = \sum_{k \in Q} f(|S|, k).$$

4.2.2 Algorytm sufiksowy

Prawdopodobieństwo $P(S)$ możemy również obliczyć używając sufiksów. Niech $0 \leq i \leq |S| = m$ i niech $b(i,k)$ oznacza prawdopodobieństwo wyemitowania sufiksu $S[i{+}1..m]$, zakładając, że stan początkowy modelu jest k. Mamy więc $b(|S|,k) = 1$, dla $k \in Q$ (bo oczywiście prawdopodobieństwo wyemitowania pustego słowa przy dowolnym stanie początkowym jest równe 1). Ponadto oczywiście mamy

$$P(S) = b(0, k_0).$$

Poniższe twierdzenie podaje sposób obliczania wartości $b(i,k)$.

Twierdzenie 4.2.2 *Dla $0 \leq i < |S|$ oraz $k \in Q$ mamy*

$$b(i,k) = \sum_{l \in Q} p_{k,l} \cdot e_l(S(i+1)) \cdot b(i+1, l).$$

Twierdzenie 4.2.3 *Opierając się na równaniach z Twierdzeń 4.2.1 oraz 4.2.2, wartości $f(i,k)$ oraz $b(i,k)$ można obliczyć w czasie $O(|S| \cdot |Q|^2)$ i w pamięci $O(|S| \cdot |Q|)$.*

Używając funkcji f oraz b możemy wyrazić warunkowe prawdopodobieństwo tego, że na i-tym kroku dany HMM był w stanie k, pod warunkiem że wyemitowane zostało słowo S:

$$P(\pi(i) = k \mid S) = \frac{P(\pi(i) = k \ \& \ S)}{P(S)} = \frac{f(i,k) \cdot b(i,k)}{P(S)}. \tag{4.4.1}$$

4.3 Estymacje parametrów

Problem estymacji parametrów polega na jak najlepszym doborze prawdopodobieństw przejść i emisji w oparciu o zaobserwowany zbiór słów $S_1, \ldots, S_n$, wyemitowanych przez pewien HMM. Słowa te są nazywane *uczącymi* bądź *treningowymi* sekwencjami. Zakładamy, że znamy alfabet Σ nad którym tworzone są słowa $S_1, \ldots, S_n$ oraz zbiór Q stanów modelu HMM. Nie znamy natomiast prawdopodobieństw $p_{k,l}$ oraz $e_k(x)$, dla $k, l \in Q$ oraz $x \in \Sigma$. Tak więc, naszym zadaniem jest dobranie tych prawdopodobieństw tak, aby zmaksymalizować prawdopodobieństwo wyemitowania $S_1, \ldots, S_n$. Niech $\mathcal{M}$ oznacza pewien HMM nad alfabetem Σ i zbiorem stanów Q. Przez $P_{\mathcal{M}}(S_1 \& \ldots \& S_n)$ będziemy oznaczać prawdopodobieństwo wyemitowania słów $S_1, \ldots, S_n$ w modelu $\mathcal{M}$. Ponieważ słowa są emitowane niezależnie, to

$$P_{\mathcal{M}}(S_1 \ \& \ \ldots \ \& \ S_n) = \prod_{j=1}^{n} P_{\mathcal{M}}(S_j).$$

Przechodząc do logarytmicznej skali i oznaczając $Score_{\mathcal{M}}(S)$ przez $\log(P_{\mathcal{M}}(S))$, mamy za zadanie znaleźć $\mathcal{M}$ tak, aby zmaksymalizować

$$\sum_{j=1}^{n} Score_{\mathcal{M}}(S_j).$$

Zacznijmy od prostszego zadania. Załóżmy dodatkowo, że dla każdego słowa S_j znamy ciąg stanów π_j, przy którym to słowo zostało wyemitowane. Niech $P_{k,l}$ będzie równe liczbie przejść ze stanu k w stan l w ciągach $\pi_1, \ldots, \pi_n$. Podobnie, niech $E_k(x)$ będzie równe liczbie emisji symbolu x, gdy układ był w stanie k. Wówczas przyjmujemy

$$p_{k,l} = \frac{P_{k,l}}{\sum_{q \in Q} P_{k,q}}, \tag{4.4.2}$$

oraz

$$e_k(x) = \frac{E_k(x)}{\sum_{y \in \Sigma} E_k(y)}. \tag{4.4.3}$$

Aby uniknąć zerowych prawdopodobieństw w sytuacjach gdy mamy do czynienia z małym zbiorem uczących sekwencji, często wprowadza się poprawkę, dodając 1 do każdego $P_{k,l}$ oraz każdego $E_k(x)$.

Teraz zajmiemy się trudniejszym przypadkiem, kiedy ciągi π_i nie są znane.

4.3.1 Algorytm Bauma-Welcha

Niech $\mathcal{M}$ będzie pewnym modelem HMM i niech $f^{(j)}_{\mathcal{M}}(i,k)$ oraz $b^{(j)}_{\mathcal{M}}(i,k)$ będą odpowiednio wartościami zdefiniowanymi dla algorytmu prefiksowego (por. Sekcja 4.2.1) oraz dla algorytmu sufiksowego (por. Sekcja 4.2.2) dla słowa S_j w modelu $\mathcal{M}$.

Wejście: n słów uczących $S_1, \ldots, S_n \in \Sigma^*$ oraz zbiór stanów Q pewnego HMM.
Wyjście: HMM $\mathcal{M}$ maksymalizujący[3] $\sum_{j=1}^{n} Score_{\mathcal{M}}(S_j)$.

Krok 1: (*Inicjalizacja*) przyjmijmy pewien model $\mathcal{M}$ (Σ oraz Q są znane i ustalone).

[3]Model $\mathcal{M}$ znajdowany przez algorytm realizuje lokalne maksimum funkcji $Score$.

Krok 2: Obliczmy wartość oczekiwaną liczby przejść w modelu $\mathcal{M}$ ze stanu k do stanu l. Zauważmy, że dla $1 \leq i \leq |S_j|$, prawdopodobieństwo tego że w i-tym kroku emisji przez $\mathcal{M}$ słowa S_j model znajdował się w stanie k, a w $(i+1)$-szym kroku w stanie l, wynosi:

$$\frac{f_{\mathcal{M}}^{(j)}(i,k) \cdot p_{k,l}^{\mathcal{M}} \cdot e_l^{\mathcal{M}}(S_j(i+1)) \cdot b_{\mathcal{M}}^{(j)}(i+1,l)}{P_{\mathcal{M}}(S_j)}.$$

Zatem wartość oczekiwana liczby przejść ze stanu k w l wynosi

$$P_{k,l} = \sum_{j+1}^{n} \sum_{i=1}^{|S_j|} \frac{f_{\mathcal{M}}^{(j)}(i,k) \cdot p_{k,l}^{\mathcal{M}} \cdot e_l^{\mathcal{M}}(S_j(i+1)) \cdot b_{\mathcal{M}}^{(j)}(i+1,l)}{P_{\mathcal{M}}(S_j)}.$$

Podobnie liczymy wartość oczekiwaną liczby emisji symbolu x w stanie k (por. (4.4.1)):

$$E_k(x) = \sum_{j=1}^{n} \sum_{i \in I_j} \frac{f_{\mathcal{M}}^{(j)}(i,k) \cdot b_{\mathcal{M}}^{(j)}(i,k)}{P_{\mathcal{M}}(S_j)},$$

gdzie $I_j = \{i \mid S_j(i) = x\}$.

Krok 3: Mając nowe wartości $P_{k,l}$ oraz $E_k(x)$ wyznaczamy nowe prawdopodobieństwa $p_{k,l}$ oraz $e_k(x)$, stosując wzory (4.4.2) i (4.4.3). W ten sposób otrzymujemy nowy model $\mathcal{M}'$.

Krok 4: Obliczamy $\sum_{j=1}^{n} Score_{\mathcal{M}'}(S_j)$. Jeśli ta wartość nie różni się o wiecej niż z góry zadane $\varepsilon > 0$ od poprzedniej wartości $\sum_{j=1}^{n} Score_{\mathcal{M}}(S_j)$, to przerywamy iterowanie z wynikiem $\mathcal{M}$. W przeciwnym przypadku powtarzamy kroki 2,3,4 algorytmu.

Typowy problem z powyższym algorytmem polega na tym, że może on znaleźć lokalne maksimum zamiast globalnego. Można częściowo obejść ten problem startując algorytm dla różnych wartości początkowych (krok 1). Jeśli w większości przypadków dostaniemy podobny wynik, to jest szansa, że mamy do czynienia z maksimum globalnym.