Algoritmy pre HMM

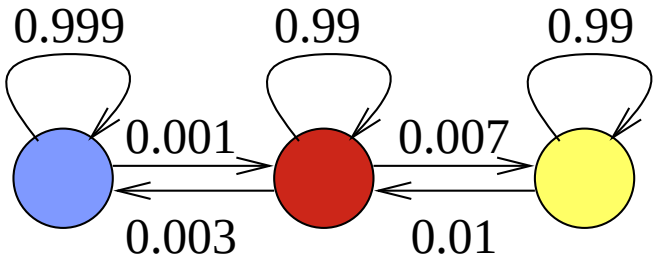
Askar Gafurov

7.11.2019

Opakovanie: HMM (skrytý Markovov model)

→ náhodná DNA sekvencia S, náhodná anotácia A
(podobné na ozajstnú DNA)

$\Pr(S, A)$ – pravdepodobnosť, že model vygeneruje pár (S, A).



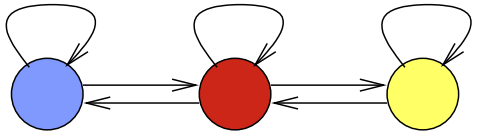
a:0.27	a:0.24	a:0.26
c:0.23	c:0.27	c:0.22
g:0.23	g:0.28	g:0.22
t:0.27	t:0.21	t:0.30

Predpokladajme, že model vždy začína v modrom stave.

$\Pr(\text{aca}) = 0.27 \cdot 0.001 \cdot 0.27 \cdot 0.99 \cdot 0.24 = 0.000017$

$\Pr(\text{aca}) = 0.27 \cdot 0.999 \cdot 0.23 \cdot 0.999 \cdot 0.27 = 0.017$

Parametre HMM (označenie)



Sekvencia $S_1, \ldots, S_n$

Anotácia $A_1, \ldots, A_n$

Parametre modelu:

Prechodová pravdepodobnosť $a(u, v) = \Pr(A_{i+1} = v | A_i = u)$,

Emisná pravdepodobnosť $e(u, x) = \Pr(S_i = x | A_i = u)$,

Počiatočná pravdepodobnosť $\pi(u) = \Pr(A_1 = u)$.

a			
	0.99	0.007	0.003
	0.01	0.99	0
	0.001	0	0.999

e	a	c	g	t
	0.24	0.27	0.28	0.21
	0.26	0.22	0.22	0.30
	0.27	0.23	0.23	0.27

Výsledná pravdepodobnosť: $\Pr(A_1, \ldots, A_n, S_1, \ldots, S_n) = \pi(A_1)e(A_1, S_1) \prod_{i=2}^{n} a(A_{i-1}, A_i)e(A_i, S_i)$

Viterbiho algoritmus

Pre danú sekvenciu S nájde najpravdepodobnejšiu anotáciu
$A = \arg\max_A \Pr(A|S)$

Dynamické programovanie v čase $O(nm^2)$

Podproblém $V[i,u]$: pravdepodobnosť najpravdepodobnejšej cesty končiacej po i krokoch v stave u, pričom vygeneruje $S_1 S_2 \ldots S_i$

Rekurencia:

$V[1,u] = \pi_u \cdot e_{u,S_1}$
$V[i,u] = \max_w V[i-1,w] \cdot a_{w,u} \cdot e_{u,S_i}$

Algoritmus:

Inicializuj $V[1,*]$
for $i = 2 \ldots n$ (n=dĺžka reťazca)
    for $u = 1 \ldots m$ (m =počet stavov)
        vypočítaj $V[i,u]$
Maximálne $V[n,j]$ je pravdepodobnosť najpravdepodobnejšej cesty

Dopredný algoritmus

Počíta celkovú pravdepodobnosť, že vygenerujeme sekvenciu S
$\Pr(S) = \sum_A Pr(A, S)$

Podproblém $F[i, u]$**:** pravdepodobnosť, že po i krokoch vygenerujeme $S_1, S_2, \ldots S_i$ a dostaneme sa do stavu u.

$F[i, u] = \Pr(A_i = u \wedge S_1, S_2, \ldots, S_i) =$
$\sum_{A_1, A_2, \ldots, A_i = u} \Pr(A_1, A_2, ..., A_i \wedge S_1, S_2, ..., S_i)$

Rekurencia:

$F[1, u] = \pi_u \cdot e_{u, S_1}$
$F[i, u] = \sum_v F[i-1, v] \cdot a_{v,u} \cdot e_{u, S_i}$

Celková pravdepodobnosť $\Pr(S) = \sum_u F[n, u]$

Spätný algoritmus

Obdoba dopredného algoritmu

Dopredný algoritmus: $F[i,u] = \Pr(A_i = u \wedge S_1, \ldots, S_i)$

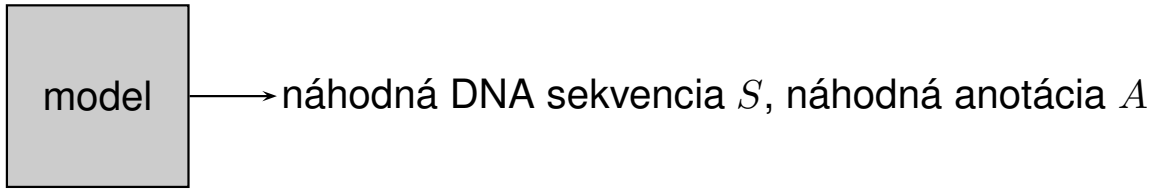
$F[1,u] = \pi_u \cdot e_{u,S_1}$

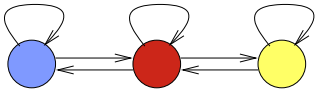
$F[i,u] = \sum_v F[i-1,v] \cdot a_{v,u} \cdot e_{u,S_i}$

$\Pr(S) = \sum_u F[n,u]$

Spätný algoritmus: $B[i,u] = \Pr(S_{i+1} \ldots, S_n | A_i = u)$

Hľadanie génov s HMM



- **Určenie stavov a prechodov v modeli:** ručne, na základe poznatkov o štruktúre génu. 
- **Trénovanie parametrov:** pravdepodobnosti určíme na základe sekvencií so známymi génmi (**trénovacia množina**).
  Model zostavíme tak, aby páry (S, A) s vlastnosťami podobnými skutočným génom mali veľkú pravdepodobnosť $\Pr(S, A)$
- **Použitie:** pre novú sekvenciu S nájdi najpravdepodobnejšiu anotáciu $A = \arg\max_A \Pr(A|S)$ Viterbiho algoritmom

Trénovanie HMM

- Stavový priestor + povolené prechody väčšinou ručne
- Parametre (pravdepodobnosti prechodu, emisie a počiatočné) automaticky z trénovacích sekvencií
- Čím zložitejší model a viac parametrov máme, tým potrebujeme viac trénovacích dát, aby nedošlo k preučeniu, t.j. k situácii, keď model dobre zodpovedá nejakým zvláštnostiam trénovacích dát, nie však ďalším dátam.
- Presnosť modelu testujeme na zvláštnych testovacích dátach, ktoré sme nepoužili na trénovanie.

Trénovanie HMM z anotovaných sekvencií

Vstup: topológia modelu a niekoľko trénovacích párov $S^{(i)}, A^{(i)}$

Cieľ: nastaviť π_u, $e_{u,x}$, $a_{u,v}$ tak, aby $\prod_i \Pr(S^{(i)}, A^{(i)})$ bola čo najväčšia

Dosiahneme jednoduchým počítaním frekvencií

Napr. $a_{u,v}$: nájdeme všetky výskyty stavu u a zistíme, ako často za nimi ide stav v

Trénovanie HMM z neanotovaných sekvencií

Vstup: topológia modelu a niekoľko trénovacích sekvencií $S^{(i)}$ anotácie $A^{(i)}$ nepoznáme

Cieľ: nastaviť π_u, $e_{u,x}$, $a_{u,v}$ tak, aby $\prod_i \Pr(S^{(i)})$ bola čo najväčšia

Používajú sa heuristické iteratívne algoritmy, napr. Baum-Welchov, ktorý je verziou všeobecnejšieho algoritmu EM (expectation maximization).

Tvorba stavového priestoru modelu

Príklad HMM na hľadanie génov