Profils d'alignement et HMM

HLIN608 Algorithmique du texte

sylvain.daude@umontpellier.fr
annie.chateau@umontpellier.fr

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement
  - on dispose d'un panel de référence de séquences similaires

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement
  - on dispose d'un panel de référence de séquences similaires
    - ex : protéines codant une même fonction biologique

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement
  - on dispose d'un panel de référence de séquences similaires
    - ex : protéines codant une même fonction biologique
  - on évalue la ressemblance de nouvelles séquences avec ce panel

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement
  - on dispose d'un panel de référence de séquences similaires
    - ex : protéines codant une même fonction biologique
  - on évalue la ressemblance de nouvelles séquences avec ce panel
    - objectif : ces nouvelles séquences codent-elles la même fonction ?

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement
  - on dispose d'un panel de référence de séquences similaires
    - ex : protéines codant une même fonction biologique
  - on évalue la ressemblance de nouvelles séquences avec ce panel
    - objectif : ces nouvelles séquences codent-elles la même fonction ?
  - étape 1 : calcul du "profil d'alignement" du panel de référence

Alignements multiples

- Comment évaluer l'alignement de plusieurs séquences de même longueur n ?
- En général : problème NP-complet $\longrightarrow$ nombreuses approches
- Approche des profils d'alignement
  - on dispose d'un panel de référence de séquences similaires
    - ex : protéines codant une même fonction biologique
  - on évalue la ressemblance de nouvelles séquences avec ce panel
    - objectif : ces nouvelles séquences codent-elles la même fonction ?
  - étape 1 : calcul du "profil d'alignement" du panel de référence
  - étape 2 : calcul du score d'alignement de la séquence candidate sur le profil

Calcul du profil d'alignement

- Exemple : calculer le profil l'alignement du panel

$$
\begin{array}{cccccc}
G & A & T & T & C & A \\
G & - & C & T & - & A \\
G & A & T & T & - & T \\
G & - & - & T & C & -
\end{array}
$$

Calcul du profil d'alignement

- Exemple : calculer le profil l'alignement du panel

  $G \quad A \quad T \quad T \quad C \quad A$
  $G \quad - \quad C \quad T \quad - \quad A$
  $G \quad A \quad T \quad T \quad - \quad T$
  $G \quad - \quad - \quad T \quad C \quad -$

- alphabet de l'alignement : $\Sigma = \{G\ A\ T\ C\ -\}$

Calcul du profil d'alignement

- Exemple : calculer le profil l'alignement du panel

  $G \quad A \quad T \quad T \quad C \quad A$
  $G \quad - \quad C \quad T \quad - \quad A$
  $G \quad A \quad T \quad T \quad - \quad T$
  $G \quad - \quad - \quad T \quad C \quad -$

- alphabet de l'alignement : $\Sigma = \{G\ A\ T\ C\ -\}$
- profil d'alignement = matrice $|\Sigma| \times n$

Calcul du profil d'alignement

- Exemple : calculer le profil l'alignement du panel

  $G \quad A \quad T \quad T \quad C \quad A$
  $G \quad - \quad C \quad T \quad - \quad A$
  $G \quad A \quad T \quad T \quad - \quad T$
  $G \quad - \quad - \quad T \quad C \quad -$

- alphabet de l'alignement : $\Sigma = \{G\ A\ T\ C\ -\}$
- profil d'alignement = matrice $|\Sigma| \times n$
  - les lignes correspondent aux symboles de l'alphabet

Calcul du profil d'alignement

- Exemple : calculer le profil l'alignement du panel

  $G\ A\ T\ T\ C\ A$
  $G\ -\ C\ T\ -\ A$
  $G\ A\ T\ T\ -\ T$
  $G\ -\ -\ T\ C\ -$

- alphabet de l'alignement : $\Sigma = \{G\ A\ T\ C\ -\}$
- profil d'alignement = matrice $|\Sigma| \times n$
  - les lignes correspondent aux symboles de l'alphabet
  - chaque case correspond à un symbole et à une colonne d'alignement

Calcul du profil d'alignement

- Exemple : calculer le profil l'alignement du panel

  $G\ A\ T\ T\ C\ A$
  $G\ -\ C\ T\ -\ A$
  $G\ A\ T\ T\ -\ T$
  $G\ -\ -\ T\ C\ -$

- alphabet de l'alignement : $\Sigma = \{G\ A\ T\ C\ -\}$
- profil d'alignement = matrice $|\Sigma| \times n$
  - les lignes correspondent aux symboles de l'alphabet
  - chaque case correspond à un symbole et à une colonne d'alignement
  - elle contient le taux d'apparition (entre 0 et 1) du symbole dans la colonne

Résultat du calcul

- Profil d'alignement :

	G	A	T	T	C	A
	G	-	C	T	-	A
	G	A	T	T	-	T
	G	-	-	T	C	-
G	1	0	0	0	0	0
A	0	0,5	0	0	0	0,5
T	0	0	0,5	1	0	0,25
C	0	0	0,25	0	0,5	0
-	0	0,5	0,25	0	0,5	0,25

Calcul du score d'alignement sur le profil

- alignement d'une séquence S sur un profil M (de même longueur) ?

Calcul du score d'alignement sur le profil

- alignement d'une séquence S sur un profil M (de même longueur) ?
  - le symbole $S[i]$ obtient le score $M[S[i], i]$

Calcul du score d'alignement sur le profil

- alignement d'une séquence S sur un profil M (de même longueur) ?
  - le symbole $S[i]$ obtient le score $M[S[i], i]$
  - la somme des scores obtenus donne le score de S

Calcul du score d'alignement sur le profil

- alignement d'une séquence S sur un profil M (de même longueur) ?
  - le symbole $S[i]$ obtient le score $M[S[i], i]$
  - la somme des scores obtenus donne le score de S
- ex : CGTTCG, GACCAT

G	1	0	0	0	0	0
A	0	0,5	0	0	0	0,5
T	0	0	0,5	1	0	0,25
C	0	0	0,25	0	0,5	0
-	0	0,5	0,25	0	0,5	0,25
	C	G	T	T	C	G
	0	0	0,5	1	0,5	0
	G	A	C	C	A	A
	1	0,5	0,25	0	0	0,5

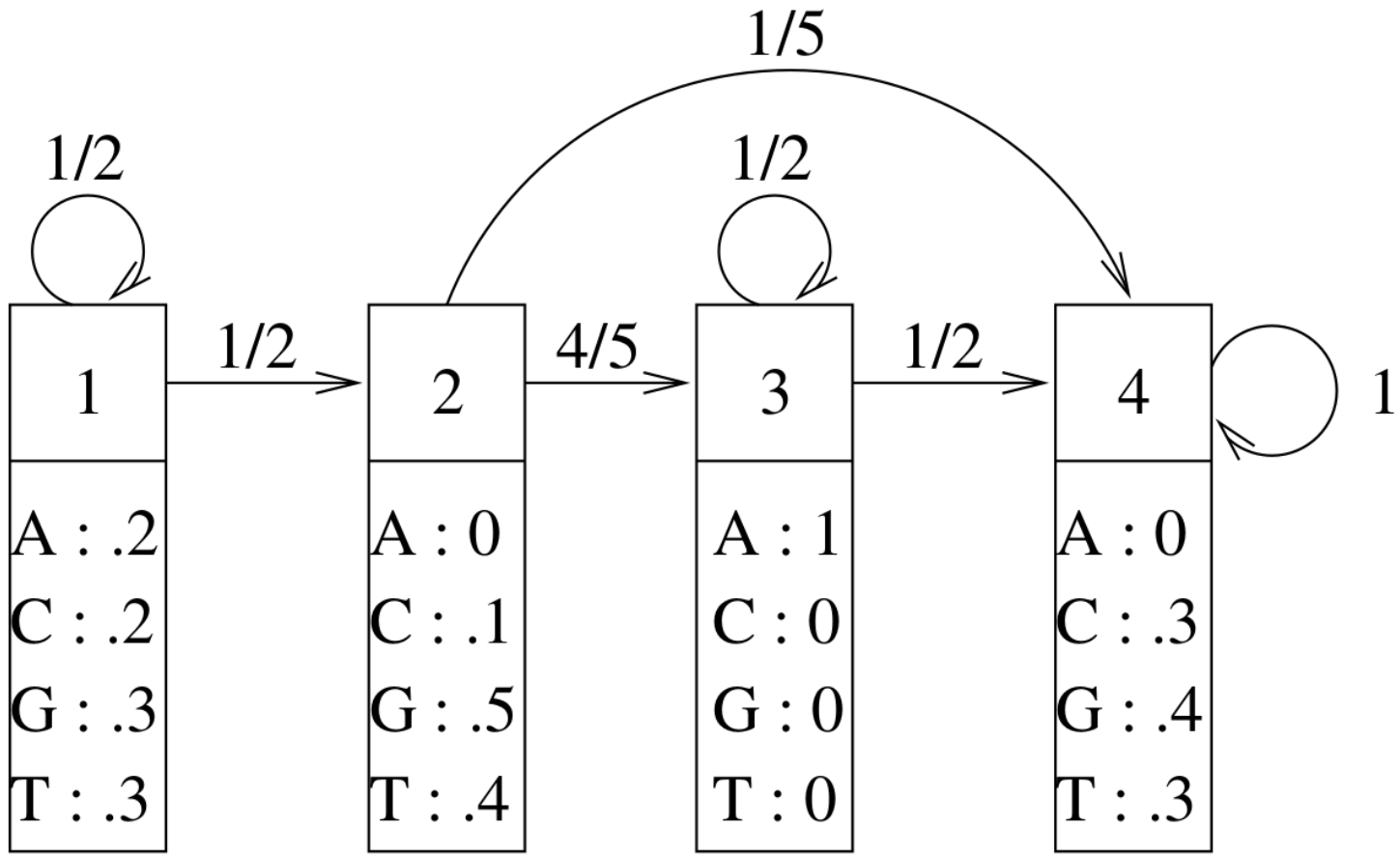
Total : 2

Total : 2,25

Une autre approche, probabiliste : Les chaînes de Markov cachées

Objectif : Représenter et modéliser une famille de séquences.

H =



Structure d'un HMM

HMM : Hidden Markov Model

Une séquence émise par ce modèle probabiliste
W = CGAAAC

Éléments de ce modèle :

- $\mathcal{A}$: alphabet $= \{A, C, G, T\}$; $|\mathcal{A}| = m$
- $\mathcal{S}$: états (sommets) $= \{1, 2, 3, 4\}$; $|\mathcal{S}| = n$
- T : matrice des probabilités de transition
- E : matrice des probabilités d'émission
- Π : vecteur des probabilités de départ

Structure d'un HMM

Matrices de transition, émission, initialisation

	1	2	3	4
1	1/2	1/2	0	0
2	0	0	4/5	1/5
3	0	0	1/2	1/2
4	0	0	0	1

$T = (t_{i,j})_{n \times m}$

	A	C	G	T
1	.2	.2	.3	.3
2	0	.1	.5	.4
3	1	0	0	0
4	0	.3	.4	.3

$E = (e_{i,j})_{n \times m}$

1	1
2	0
3	0
4	0

$\Pi = (\pi_i)_n$

Structure d'un HMM

$$\forall i, \sum_{j=1}^{n} t_{i,j} = 1 : \text{on bouge à coup sûr}$$

$$\forall i, \sum_{s \in \mathcal{A}} e_{i,s} = 1 : \text{on émet à coup sûr}$$

Émission et transition indépendantes du chemin parcouru

Marche : déplacement dans le graphe avec émission d?un symbole à chaque sommet
d'où : une marche engendre un mot sur $\mathcal{A}$

mais il y a plusieurs marches possibles pour un mot :

w		A	C	C	A	C	C
M_1	1	1	1	2	3	4	4
M_2	2	1	1	1	1	2	4

Caché : l'observateur ne voit que la séquence et non la marche

Structure d'un HMM

Vraisemblance d'une marche : $Prob(w, M|H)$

w		A	C	C	A	C	C
M_1	$\boxed{1}$	1	1	2	3	4	4
M_2	$\boxed{2}$	1	1	1	1	2	4

$$\boxed{1} = 1/2 \cdot 1/2 \cdot 4/5 \cdot 1/2 \cdot 1 \cdot (0.2 \cdot 0.2 \cdot 0.2 \cdot 1 \cdot 0.3 \cdot 0.4) = 4/5 \cdot 0.3$$

$$\boxed{2} = 1/2 \cdot 1/2 \cdot 1/2 \cdot 1/2 \cdot 1/5 \cdot (0.2 \cdot 0.2 \cdot 0.2 \cdot 0.2 \cdot 1 \cdot 0.4) = 1/10 \cdot 0.02$$

Problèmes autour des HMM

HMM = modèle probabiliste "capturant" les propriétés d'une famille de séquences ET outil de production (émission) de séquences (toute marche M produit une séquence w)

$\Rightarrow$ Informations importantes pour M et/ou w

$Prob(w, M|H)$: vraisemblance que M engendre w vis-à-vis du modèle H

$Prob(w|H)$: vraisemblance de la séquence w vis-à-vis du modèle H

Problèmes autour des HMM

- Évaluation : étant donnés H et w , calculer $Prob(w|H)$
- Décodage : étant donnés H et w, calculer M tel que $Prob(w, M|H)$ est maximale
- Apprentissage : étant donnés H et une famille $\mathcal{F}_0$ de séquences, ajuster les paramètres E , T , et ? de H pour maximiser la vraisemblance des séquences de $\mathcal{F}_0$

Évaluation

Rappel : L'objectif de l'évaluation est de calculer la vraisemblance d'une séquence w de longueur ℓ par rapport à un modèle H.

$$\begin{aligned} Prob(w|H) &= \sum_{M=q_1,\ldots,q_\ell} Prob(w, M|H) \\ &= \sum_{M=q_1,\ldots,q_\ell} \left\{ \prod_{i=1}^{\ell-1} t_{q_i,q_{i+1}} \times \prod_{i=1}^{\ell} e_{q_i,w_i} \right\} \end{aligned}$$

$\Rightarrow$ nombre exponentiel de marches !

La solution : **la programmation dynamique**

Évaluation

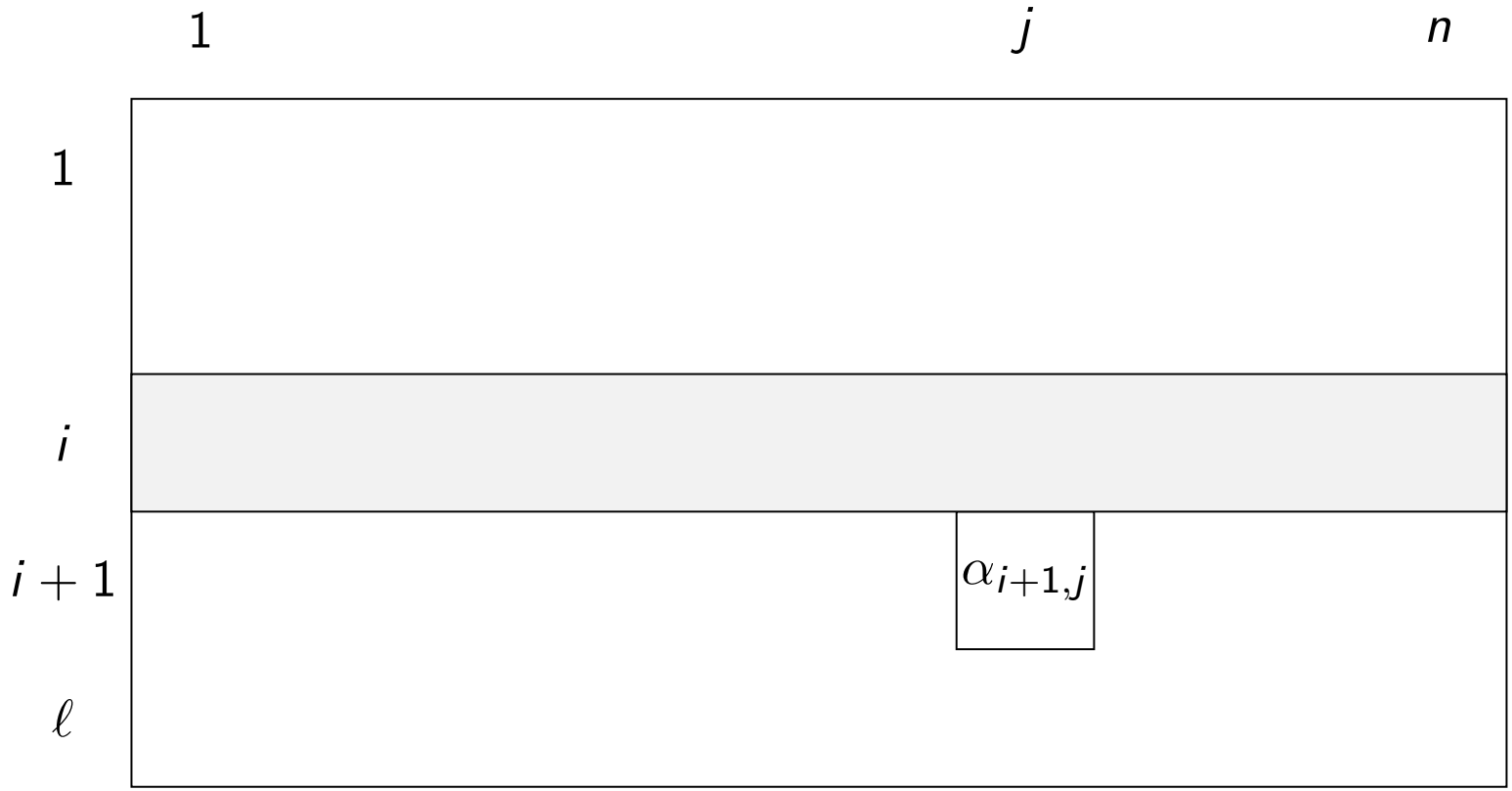
Tableau des $\alpha_{i,j}$: $\alpha_{i,j}$ est la probabilité qu'une marche se terminant en l'état j produise le préfixe $w_1 \dots w_i$

$$P(w|H) = \sum_{j \in \mathcal{S}} \alpha_{\ell,j}$$

Initialisation : pour tout $j \in \mathcal{S}, \alpha_{1,j} = \pi_j \times e_{j,w_1}$

Évaluation

Passage de i à $i+1$: $\alpha_{i+1,j} = e_{j,w_{i+1}} \times \sum_{k \in \mathcal{S}} \alpha_{i,k} \times t_{k,j}$



Évaluation

Algorithme forward (calcul du tableau des $\alpha_{i,j}$)

Entrée : un HMM $\mathcal{H} = \{\mathcal{A}, \mathcal{S}, T, E, \Pi\}$, un mot w sur $\mathcal{A}$

Sortie : la matrice des $(\alpha_{i,j})_{i=1\ldots\ell, j=1\ldots n}$

Variables intermédiaires : une matrice α de taille $\ell \times n$, des entiers i, j, k pour le remplissage

Évaluation

Algorithme 1 : Algorithme Forward

Initialisation :

Pour j *allant de 1 à* n
  $\alpha[1][j] \leftarrow \Pi[j] \times E[j][w_1]$
Fin Pour

Remplissage :

Pour i *allant de 2 à* ℓ
  Pour j *allant de 1 à* n
    $\alpha[i][j] \leftarrow 0$
    Pour k *allant de 1 à* n
      $\alpha[i][j] \leftarrow \alpha[i][j] + E[j][w_i] \times \alpha[i-1][k] \times T[k][j]$
    Fin Pour
  Fin Pour
Fin Pour

Renvoyer α

Complexité : $O(\ell n^2)$

Évaluation

Algorithme Backward

Au lieu de conserver les probabilités d'obtenir les **préfixes** du mot w, on peut conserver les probabilités d'obtenir les **suffixes** de w.

Principe dual de l'algorithme Backward

$\beta_{i,j} =$ probabilité d'obtenir le suffixe $w_i \dots w_\ell$ en partant de l'état j.

Initialisation : pour tout $j \in \mathcal{S}, \beta_{\ell,j} = e_{j,w_\ell}$

Le remplissage se fait de bas en haut, et on ajuste la première ligne avec les probabilités de départ.

Évaluation

$$Prob(w|H) = \sum_{j \in \mathcal{S}} \beta_{1,j}$$

Algorithme backward (calcul du tableau des $\beta_{i,j}$)

Entrée : un HMM $\mathcal{H} = \{\mathcal{A}, \mathcal{S}, T, E, \Pi\}$, un mot w sur $\mathcal{A}$

Sortie : la matrice des $(\beta_{i,j})_{i=1\ldots\ell, j=1\ldots n}$

Variables intermédiaires : une matrice β de taille $\ell \times n$, des entiers i, j, k pour le remplissage

Évaluation

Algorithme 2 : Algorithme Backward

Initialisation :

Pour j *allant de* 1 *à* n
  $\beta[\ell][j] \leftarrow E[j][w_\ell]$
Fin Pour

Remplissage :

Pour i *allant de* $\ell - 1$ *à* 1
  Pour j *allant de* 1 *à* n
    $\beta[i][j] \leftarrow 0$
    Pour k *allant de* 1 *à* n
      $\beta[i][j] \leftarrow \alpha[i][j] + E[j][w_i] \times \beta[i+1][k] \times T[j][k]$
    Fin Pour
  Fin Pour
Fin Pour
Pour j *allant de* 1 *à* n
  $\beta[1][j] \leftarrow \Pi[j] \times \beta[1][j]$
Fin Pour

Renvoyer β

Complexité : $O(\ell n^2)$

Décodage

Rappel : L'objectif du décodage est de calculer la marche la plus vraisemblable pour produire une séquence w de longueur ℓ par rapport à un modèle H.

On cherche M qui atteint :

$$\max_{M=q_1,\ldots,q_\ell} Prob(w, M|H) \quad = \quad \max_{M=q_1,\ldots,q_\ell} \left\{ \prod_{i=1}^{\ell-1} t_{q_i,q_{i+1}} \times \prod_{i=1}^{\ell} e_{q_i,w_i} \right\}$$

$\Rightarrow$ nombre exponentiel de marches !

La solution : **la programmation dynamique**

Décodage

Calcul des vraisemblances maximales pour produire le préfixe $w_1 \dots w_i$ en arrivant en l'état j : similaire à l'algorithme Forward, en remplaçant $\sum$ par max.

Tableau des $\delta_{i,j}$ pour stocker ces vraisemblances maximales

Initialisation : pour tout $j \in \mathcal{S}, \delta_{1,j} = \pi_j \times e_{j,w_1}$

Passage de i à $i+1$: $\delta_{i+1,j} = e_{j,w_{i+1}} \times \max_{k \in \mathcal{S}} \delta_{i,k} \times t_{k,j}$

Décodage

Algorithme de Viterbi (calcul du tableau des $\delta_{i,j}$)

Entrée : un HMM $\mathcal{H} = \{\mathcal{A}, \mathcal{S}, T, E, \Pi\}$, un mot w sur $\mathcal{A}$

Sortie : la matrice des $(\delta_{i,j})_{i=1\ldots\ell, j=1\ldots n}$

Variables intermédiaires : une matrice δ de taille $\ell \times n$, des entiers i, j, k pour le remplissage

Décodage

Algorithme 3 : Algorithme de Viterbi

Initialisation :

Pour j *allant de 1 à* n
  $\delta[1][j] \leftarrow \Pi[j] \times E[j][w_1]$
Fin Pour

Remplissage :

Pour i *allant de 2 à* ℓ
  Pour j *allant de 1 à* n
    $\delta[i][j] \leftarrow 0$
    Pour k *allant de 1 à* n
      $\delta[i][j] \leftarrow$
      $\max(\delta[i][j], E[j][w_i] \times \delta[i-1][k] \times T[k][j])$
    Fin Pour
  Fin Pour
Fin Pour

Renvoyer δ

Complexité : $O(\ell n^2)$

Décodage

Mais la vraisemblance maximale ne donne pas la marche !

Il faut un algorithme de backtracking (comme pour l'alignement optimal)

Principe : on récupère l'état maximal dans la dernière ligne, puis on retrace comment son score a été calculé par l'algorithme de Viterbi, et ainsi de suite jusqu'à la première ligne.

Décodage

Algorithme de backtracking

Entrée : un HMM $\mathcal{H} = \{\mathcal{A}, \mathcal{S}, T, E, \Pi\}$, un mot w sur $\mathcal{A}$ et une matrice δ de taille $\ell \times n$ contenant les vraisemblances maximales, produite par l'algorithme de Viterbi sur l'entrée $\mathcal{H}, w$

Sortie : La marche la plus probable ayant produit w dans $\mathcal{H}$ (= leurs indices dans $\mathcal{S}$

Variables intermédiaires : une liste L d'états, des entiers i, j, k, un flottant *max*

Complexité : $O(\ell n)$

Décodage

Algorithme 4 : Algorithme de backtracking

Initialisation :

$L \leftarrow \emptyset$

$max \leftarrow -1$

Pour k *allant de 1 à* n

  Si $\delta[\ell][k] > max$ Alors

    $max \leftarrow \delta[\ell][k]$

    $j \leftarrow k$

  Fin Si

Fin Pour

Ajouter j à L

Remontée :

Pour i *allant de* ℓ *à* 2

  $k \leftarrow 0$

  Tant que $max \neq \delta[i-1][k] \times T[k][j] \times E[j][w_i]$

    $k \leftarrow k + 1$

  Fin Tant que

  Ajouter k à L; $max \leftarrow \delta[i-1][k]$

  $j \leftarrow k$;

Fin Pour

Renvoyer L

Apprentissage

Données : H et w

Objectif : optimiser $\sum_M Prob(M|w, H)$ en modifiant les paramètres de H

On définit les probabilités suivantes :

$\gamma_{i,k}$ = probabilité que l'état i émette w_k parmi toutes les marches engendrant w

$\gamma_{i,j,k}$ = probabilité que i émette w_k et j émette w_{k+1} parmi toutes les marches engendrant w

Apprentissage

Les $\gamma_{i,j,k}$ se calculent à partir des matrices α et β :

$$\gamma_{i,j,k} = \frac{\alpha_{k,i} \times t_{i,j} \times \beta_{k+1,j} \times e_{j,w_{k+1}}}{Prob(w|H) \sum_{s_1,s_2 \in \mathcal{S}} \alpha_{k,s_1} \times t_{s_1,s_2} \times \beta_{k+1,s_2} \times e_{s_2,w_{k+1}}}$$

$$\gamma_{i,k} = \sum_{j \in \mathcal{S}} \gamma_{i,j,k}$$

Apprentissage

Algorithme de Baum-Welch

Principe : Expectation-Maximization (EM)

Entrée : un HMM $\mathcal{H} = \{\mathcal{A}, \mathcal{S}, T, E, \Pi\}$, un mot w sur $\mathcal{A}$, un seuil de précision ϵ

Sortie : un HMM optimisé pour reconnaître w

Variables intermédiaires : des entiers i, j, k, un HMM H' de même taille que H, des matrices $(\gamma_{i,k})$ et $(\gamma_{i,j,k})$ de taille $n \times n$ et $n \times n \times n$ respectivement.

Apprentissage

Algorithme 5 : Algorithme de Baum-Welch

$H' \leftarrow H$

Répéter

$H \leftarrow H'$

Calculer les $\gamma_{i,k}$ et les $\gamma_{i,j,k}$

Calculer Π', T', E' :

$$\Pi'_i = \gamma_{i,1} \qquad t'_{i,j} = \frac{\sum_{k=1}^{\ell-1} \gamma_{i,j,k}}{\sum_{k=1}^{\ell-1} \gamma_{i,k}} \qquad e'_{i,c} = \frac{\sum_{k=1, w_k=c}^{\ell} \gamma_{i,k}}{\sum_{k=1}^{\ell} \gamma_{i,k}}$$

Tant que $|Prob(w|H') - Prob(w|H)| \geq \epsilon$

Renvoyer H

Résumé

Algorithme de Baum-Welch

Avec plusieurs séquences à la fois : On utilise des vecteurs de probabilité pour les $\gamma_{i,k}$ et les $\gamma_{i,j,k}$.

$\mathcal{F}$ famille de séquences $\rightarrow$ un HMM $H_{\mathcal{F}}$ capturant les propriétés de $\mathcal{F}$

Algorithme Forward

Vraisemblance que le modèle $H_{\mathcal{F}}$ ait engendré w : "w ressemble-t-elle aux séquences de $\mathcal{F}$?"

Algorithme de Viterbi

Marche la plus vraisemblable engendrant w : structure de w par rapport à ce que l'on sait de $\mathcal{F}$