Modèles probabilistes pour l'accès à l'information à grande échelle

Modèles Structurés

François Yvon

LIMSI — CNRS and Université Paris Sud

2017 / 2018

Limsi

Sommaire

1. Modèles structurés
   - Généralités
2. HMM, une révision rapide
3. Modèles pour l'alignement de mots
4. Modèles pour la syntaxe

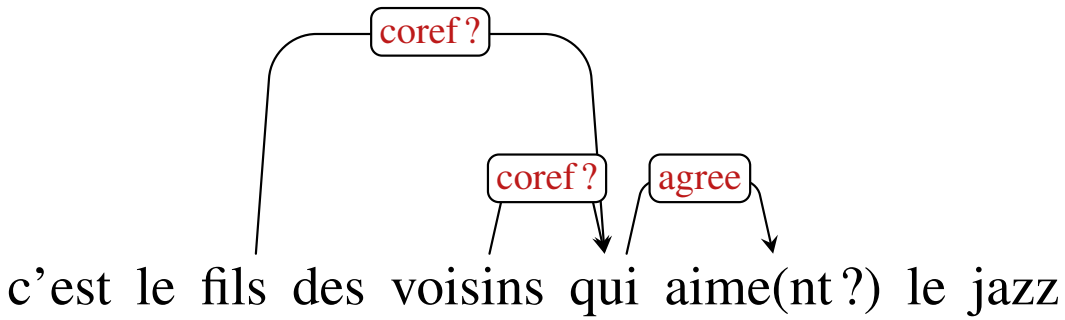
Limsi

Modèles structurés

Tentative de définition

Les modèles structurés représentent des ensembles de VA impliquées dans des structures / dépendances complexes

- des séquences (de mots, de lettres, de phrases) impliquant des dépendances structurales ou contextuelles



- des paires de séquences (symboles, étiquettes)
- des alignements
- des arbres syntagmatiques ou des graphes de dépendance

Modèles structurés

Tentative de définition

Les modèles structurés représentent des ensembles de VA impliquées dans des structures / dépendances complexes

- des séquences (de mots, de lettres, de phrases) impliquant des dépendances structurales ou contextuelles
- des paires de séquences (symboles, étiquettes)

He	*reckons*	*the*	*current*	*account*	*deficit*	*will*	*narrow*	...
B-NP	B-VP	B-NP	I-NP	I-NP	I-NP	B-VP	I-VP	...

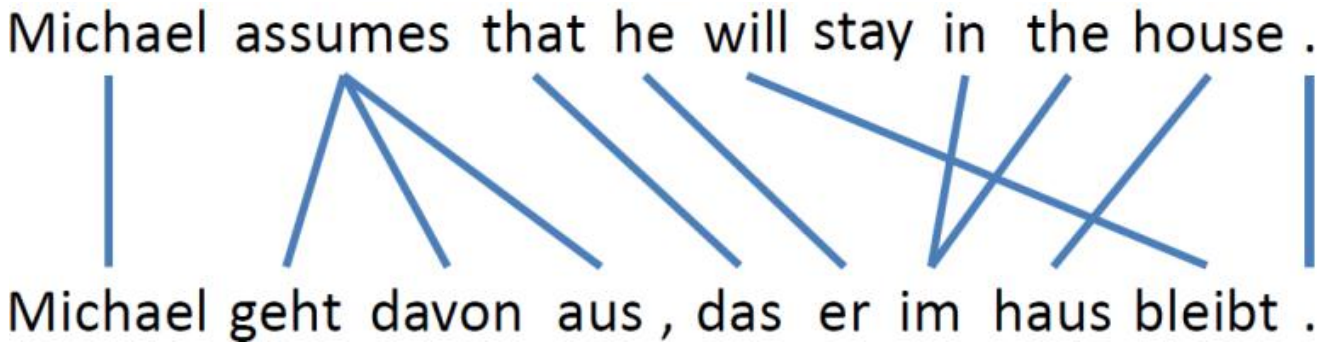
- des alignements
- des arbres syntagmatiques ou des graphes de dépendance

Modèles structurés

Tentative de définition

Les modèles structurés représentent des ensembles de VA impliquées dans des structures / dépendances complexes

- des séquences (de mots, de lettres, de phrases) impliquant des dépendances structurales ou contextuelles
- des paires de séquences (symboles, étiquettes)
- des alignements



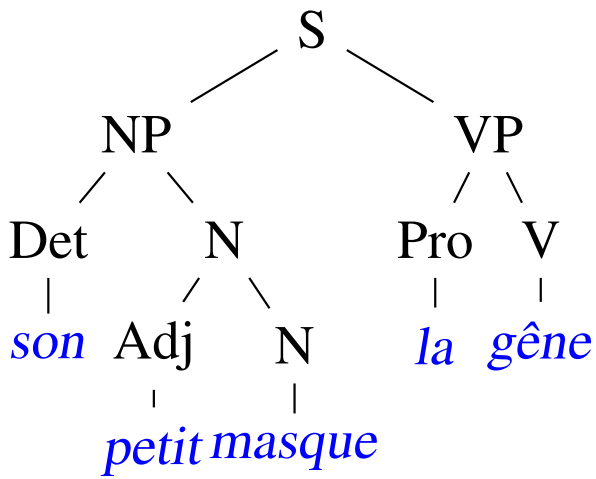
- des arbres syntagmatiques ou des graphes de dépendance

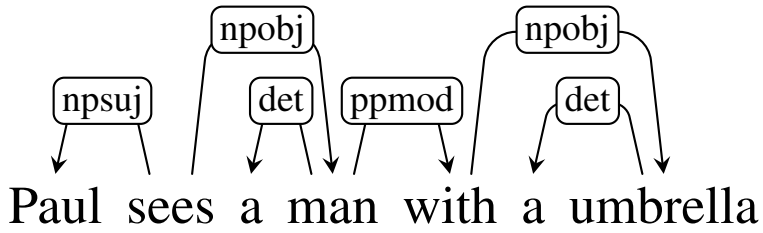
Modèles structurés

Tentative de définition

Les modèles structurés représentent des ensembles de VA impliquées dans des structures / dépendances complexes

- des séquences (de mots, de lettres, de phrases) impliquant des dépendances structurales ou contextuelles
- des paires de séquences (symboles, étiquettes)
- des alignements
- des arbres syntagmatiques ou des graphes de dépendance





Modèles structurés

Tentative de définition

Les modèles structurés représentent des ensembles de VA impliquées dans des structures / dépendances complexes

- des séquences (de mots, de lettres, de phrases) impliquant des dépendances structurales ou contextuelles
- des paires de séquences (symboles, étiquettes)
- des alignements
- des arbres syntagmatiques ou des graphes de dépendance

Difficultés

- Les VA représentant une instance ne sont pas indépendantes entre elles
- Trouver les factorisations qui rendent l'inférence faisable est difficile

Sommaire

1 Modèles structurés

2 HMM, une révision rapide
- Apprentissage supervisé : POS tagging
- Généralités
- Deux problèmes combinatoires
- Apprentissage non supervisé

3 Modèles pour l'alignement de mots

4 Modèles pour la syntaxe

Étiquetage en parties du discours

ou Étiquetage Morphosyntaxique ou "Part of speech Tagging"

Classification d'une séquence de mots

$$W_{[1:T]} = w_{[1:T]} \Rightarrow E_{[1:T]} = e_{[1:T]}$$

avec e_s une valeur possible pour la VA E_s parmi les étiquettes $\mathcal{E}$

Catégorie morphosyntaxique d'un mot ?

Désambiguïse les propriétés d'un mot graphique en contexte

nom, verbe, adjectif, etc + informations morphologiques (genre, nombre, temps, mode, cas, etc)

Construction et utilisation d'un étiqueteur morphosyntaxique

- Modélisation : formuler un modèle paramétrique de $\mathrm{P}(W_{[1:T]}, E_{[1:T]}; \theta)$
- Apprentissage : sachant $\{(W_{[1:T]}, E_{[1:T]})_i, i = 1 \ldots N\}$: estimer θ
- Inférence : sachant θ, prédire la séquence d'étiquettes $E^\star_{[1:T]}$ de $W^\star_{[1:T]}$

Étiquetage en parties du discours

ou Étiquetage Morphosyntaxique ou "Part of speech Tagging"

Classification d'une séquence de mots

$$W_{[1:T]} = w_{[1:T]} \Rightarrow E_{[1:T]} = e_{[1:T]}$$

avec e_s une valeur possible pour la VA E_s parmi les étiquettes $\mathcal{E}$

Catégorie morphosyntaxique d'un mot ?

Désambiguïse les propriétés d'un mot graphique en contexte

nom, verbe, adjectif, etc + informations morphologiques (genre, nombre, temps, mode, cas, etc)

Problème difficile : $\approx$ 50% des occurrences sont ambiguës

- *le* : pronom ou déterminant ? *la* : pronom ou nom déterminant ? *est* : verbe ou adjectif ?
- *bus* : verbe (présent / passé) ou nom ? *couvent* : verbe ou nom ?

Applications : synthèse de parole, traduction, extraction d'information, etc.

Construction et utilisation d'un étiqueteur morphosyntaxique

- Modélisation : formuler un modèle paramétrique de $P(W_{[1:T]}, E_{[1:T]}; \theta)$

Étiquetage en parties du discours

ou Étiquetage Morphosyntaxique ou "Part of speech Tagging"

Classification d'une séquence de mots

$$W_{[1:T]} = w_{[1:T]} \Rightarrow E_{[1:T]} = e_{[1:T]}$$

avec e_s une valeur possible pour la VA E_s parmi les étiquettes $\mathcal{E}$

Catégorie morphosyntaxique d'un mot ?

Désambiguïse les propriétés d'un mot graphique en contexte

nom, verbe, adjectif, etc + informations morphologiques (genre, nombre, temps, mode, cas, etc)

Construction et utilisation d'un étiqueteur morphosyntaxique

- Modélisation : formuler un modèle paramétrique de $\mathrm{P}(W_{[1:T]}, E_{[1:T]}; \theta)$
- Apprentissage : sachant $\{(W_{[1:T]}, E_{[1:T]})_i, i = 1 \ldots N\}$: estimer θ
- Inférence : sachant θ, prédire la séquence d'étiquettes $E^\star_{[1:T]}$ de $W^\star_{[1:T]}$

Étiquetage : de la loi jointe à la loi conditionnelle

Le modèle du canal bruité

$$E^*_{[1:T]} = \underset{E_{[1:T]}}{\operatorname{argmax}}\, \mathrm{P}(E_{[1:T]} \mid W_{[1:T]}; \boldsymbol{\theta}) = \underset{E_{[1:T]}}{\operatorname{argmax}}\, \frac{\mathrm{P}(W_{[1:T]}, E_{[1:T]}; \boldsymbol{\theta})}{\mathrm{P}(W_{[1:T]}; \boldsymbol{\theta})}$$

$$= \underset{E_{[1:T]}}{\operatorname{argmax}}\, \mathrm{P}(W_{[1:T]} \mid E_{[1:T]}; \boldsymbol{\theta})\, \mathrm{P}(E_{[1:T]}; \boldsymbol{\theta}) = \underset{E_{[1:T]}}{\operatorname{argmax}}\, \mathrm{P}(W_{[1:T]}, E_{[1:T]}; \boldsymbol{\theta})$$

Factorisation agnostique de la loi jointe

En appliquant la règle de Bayes et en définissant des termes comme $e_{1,0}$

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1})\, \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

Quelles hypothèses linguistiques pour factoriser P() ?

Étiquetage : de la loi jointe à la loi conditionnelle

Le modèle du canal bruité

$$E^*_{[1:T]} = \underset{E_{[1:T]}}{\operatorname{argmax}}\, \mathrm{P}(E_{[1:T]} \mid W_{[1:T]}; \boldsymbol{\theta}) = \underset{E_{[1:T]}}{\operatorname{argmax}}\, \frac{\mathrm{P}(W_{[1:T]}, E_{[1:T]}; \boldsymbol{\theta})}{\mathrm{P}(W_{[1:T]}; \boldsymbol{\theta})}$$

$$= \underset{E_{[1:T]}}{\operatorname{argmax}}\, \mathrm{P}(W_{[1:T]} \mid E_{[1:T]}; \boldsymbol{\theta})\, \mathrm{P}(E_{[1:T]}; \boldsymbol{\theta}) = \underset{E_{[1:T]}}{\operatorname{argmax}}\, \mathrm{P}(W_{[1:T]}, E_{[1:T]}; \boldsymbol{\theta})$$

Factorisation agnostique de la loi jointe

En appliquant la règle de Bayes et en définissant des termes comme $e_{1,0}$

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t, e_t \mid w_{1,t-1}, e_{1,t-1})$$

$$= \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1})\, \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

Quelles hypothèses linguistiques pour factoriser P() ?

Étiquetage : de la loi jointe à la loi conditionnelle

Le modèle du canal bruité

$$E^*_{[1:T]} = \underset{E_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(E_{[1:T]} \mid W_{[1:T]}; \boldsymbol{\theta}) = \underset{E_{[1:T]}}{\operatorname{argmax}} \frac{\mathrm{P}(W_{[1:T]}, E_{[1:T]}; \boldsymbol{\theta})}{\mathrm{P}(W_{[1:T]}; \boldsymbol{\theta})}$$

$$= \underset{E_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(W_{[1:T]} \mid E_{[1:T]}; \boldsymbol{\theta}) \mathrm{P}(E_{[1:T]}; \boldsymbol{\theta}) = \underset{E_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(W_{[1:T]}, E_{[1:T]}; \boldsymbol{\theta})$$

Factorisation agnostique de la loi jointe

En appliquant la règle de Bayes et en définissant des termes comme $e_{1,0}$

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1}) \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

Quelles hypothèses linguistiques pour factoriser P( ) ?

Hypothèses Markoviennes : localité des dépendances

Factorisation agnostique

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1})\, \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

H1 : (syntaxe locale) E_t est indépendante du passé lointain sachant E_{t-1}

$$\mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(e_t \mid e_{t-1})$$

[pas raisonnable : augmenter l'ordre ou le jeu d'étiquettes aide]

H2 : (les choix lexicaux sont indépendants de la grammaire) W_t ne dépend que de E_t

Conditionnellement à E_t, W_t indépendant de toute autre variable

$$\mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(w_t \mid e_t)$$

[pas raisonnable non plus : lexicalisation, cohérence lexicale...]

Limsi

Hypothèses Markoviennes : localité des dépendances

Factorisation agnostique

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1})\, \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

H1 : (syntaxe locale) E_t est indépendante du passé lointain sachant E_{t-1}

$$\mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(e_t \mid e_{t-1})$$

[pas raisonnable : augmenter l'ordre ou le jeu d'étiquettes aide]

H2 : (les choix lexicaux sont indépendants de la grammaire) W_t ne dépend que de E_t

Conditionnellement à E_t, W_t indépendant de toute autre variable

$$\mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(w_t \mid e_t)$$

[pas raisonnable non plus : lexicalisation, cohérence lexicale...]

Limsi

Hypothèses Markoviennes : localité des dépendances

Factorisation agnostique

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1})\, \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

H1 : (syntaxe locale) E_t est indépendante du passé lointain sachant E_{t-1}

$$\mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(e_t \mid e_{t-1})$$

[pas raisonnable : augmenter l'ordre ou le jeu d'étiquettes aide]

H2 : (les choix lexicaux sont indépendants de la grammaire) W_t ne dépend que de E_t

Conditionnellement à E_t, W_t indépendant de toute autre variable

$$\mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(w_t \mid e_t)$$

[pas raisonnable non plus : lexicalisation, cohérence lexicale...]

Limsi

Hypothèses Markoviennes : localité des dépendances

Factorisation agnostique

$$\mathrm{P}(w_{[1:T]}, e_{[1:T]}) = \prod_{t=1}^{T} \mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1})\, \mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1})$$

H1 : (syntaxe locale) E_t est indépendante du passé lointain sachant E_{t-1}

$$\mathrm{P}(e_t \mid w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(e_t \mid e_{t-1})$$

[pas raisonnable : augmenter l'ordre ou le jeu d'étiquettes aide]

H2 : (les choix lexicaux sont indépendants de la grammaire) W_t ne dépend que de E_t

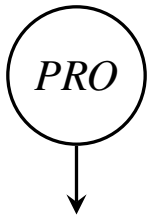
Conditionnellement à E_t, W_t indépendant de toute autre variable

$$\mathrm{P}(w_t \mid e_t, w_{1,t-1}, e_{1,t-1}) = \mathrm{P}(w_t \mid e_t)$$

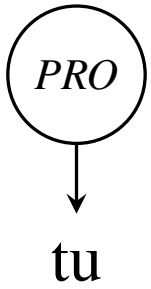
[pas raisonnable non plus : lexicalisation, cohérence lexicale...]

Limsi

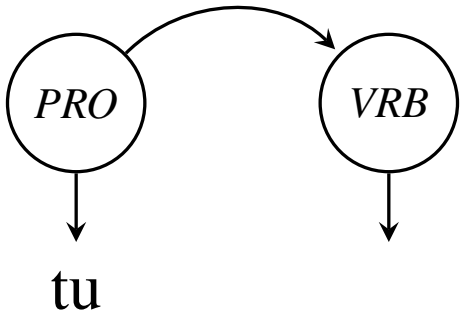
Représentation graphique, génération



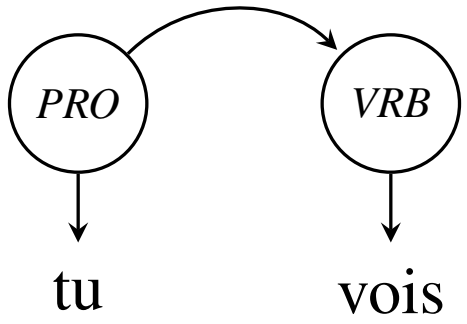
Représentation graphique, génération



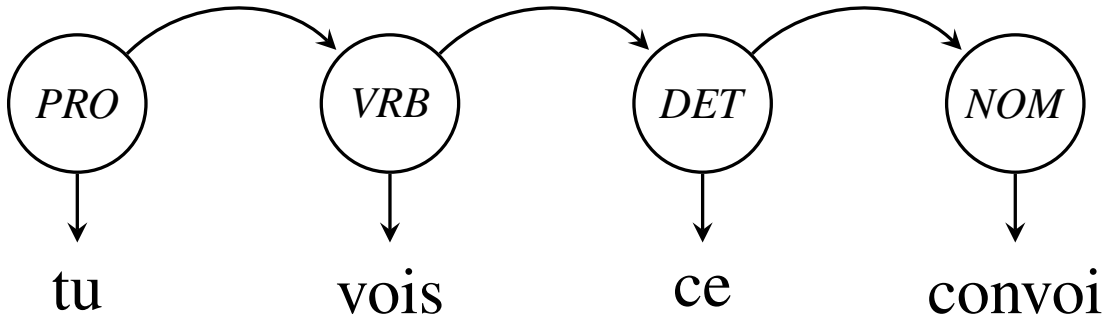
Représentation graphique, génération



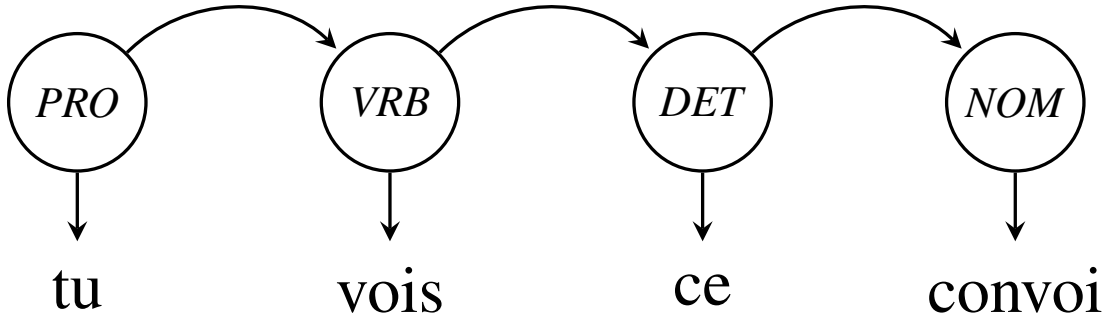
Représentation graphique, génération



Représentation graphique, génération

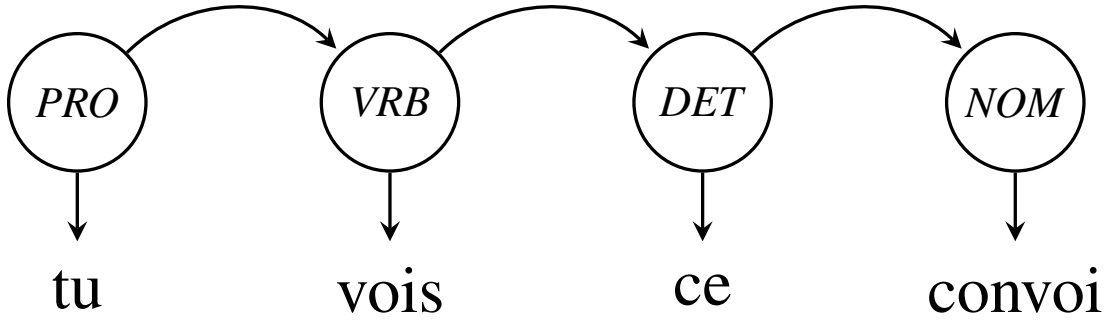


Représentation graphique, génération



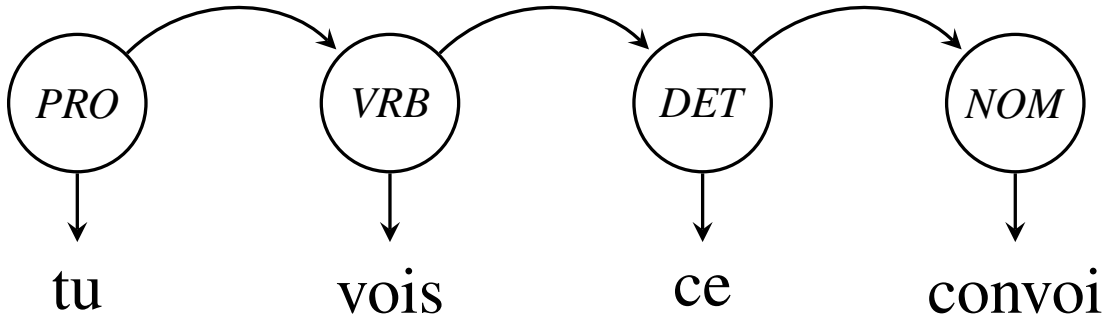
- correction grammaticale de $w_1 \ldots w_k$? $\propto \mathrm{P}(W_{[1:T]} \mid \boldsymbol{\theta})$

Représentation graphique, génération



- correction grammaticale de $w_1 \ldots w_k$? $\propto \mathrm{P}(W_{[1:T]} \mid \boldsymbol{\theta})$
- sachant $w_{[1:T]}$, quelle est la meilleure $e_{[1:T]}$? $\operatorname{argmax} \mathrm{P}(E_{[1:T]} \mid W_{[1:T]}, \boldsymbol{\theta})$

Représentation graphique, génération



- correction grammaticale de $w_1 \ldots w_k$? $\propto \mathrm{P}(W_{[1:T]} \mid \boldsymbol{\theta})$
- sachant $w_{[1:T]}$, quelle est la meilleure $e_{[1:T]}$? $\operatorname{argmax} \mathrm{P}(E_{[1:T]} \mid W_{[1:T]}, \boldsymbol{\theta})$
- probabilité de la séquence *PRO VRB* ? Qu'un *PRO* soit égal à *tu* ? $\Rightarrow$ lexique probabiliste

D'autres problèmes d'étiquetage de séquence

En traitement automatique du langage

Ré-accentuation

je me leve (lève ? levé ?) : Le contexte permet de prendre la décision.

- observation : mot dégradé, bruité par ex. *leve*.
- étiquette : le mot par ex. *lève*.

Ré-capitalisation

- observation : mot dégradé, bruité par ex. *il*.
- étiquette : le mot par ex. *il* et *Il*.

Ré-ponctuation

- observation : le mot par ex. *il*.
- étiquette : (mot | ponctuation) par ex. *il,* ou *il.* ou il_{rien}

Aussi : frontière de groupes syntaxiques, d'entités nommées, etc.

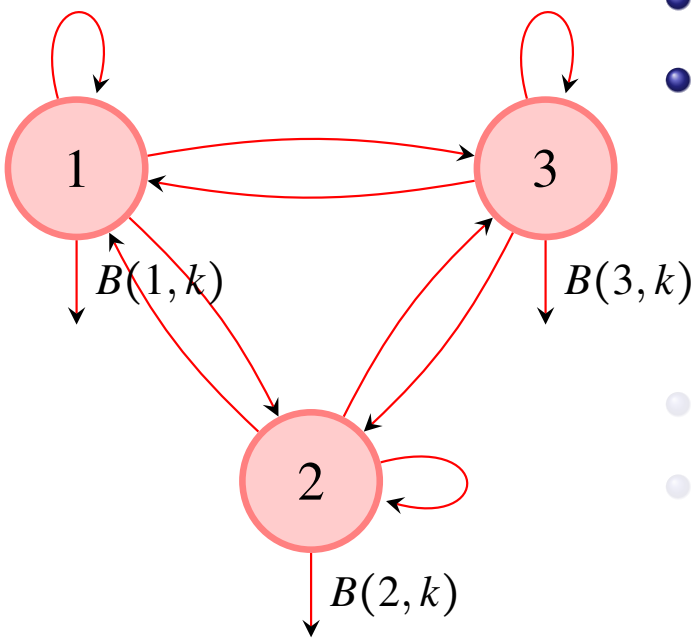
30 ans de POS tagging Markovien

- ☺ Le premier succès des approches empiriques (Chuch, de Rose 1988)
- ☺ Des performances encore (presque) compétitives
- ☺ Combinaison supervisé / non-supervisé (Mérialdo 1994)
- ☺ Intégration raisonnée dans une chaine de traitements
- ☺ Facile à neuronaliser

Limsi

Modèles de Markov Cachés - Hidden Markov Model (HMM)

Définition (cas discret)



Un modèle de Markov « caché » est défini par $(\{1...n_S\}, \{1 \ldots n_K\}, \boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B})$.

- $\{1...n_S\}$ l'ensemble des états,
- $\boldsymbol{\pi}$ les probabilités initiales : $\pi(i) = \mathrm{P}(Q_1 = i)$,
- $\boldsymbol{A}$ la matrice de transition (indépendant de t) :

$$A(i,j) = \mathrm{P}(q_t = j \,|\, q_{t-1} = i) = \mathrm{P}(j \,|\, i)$$

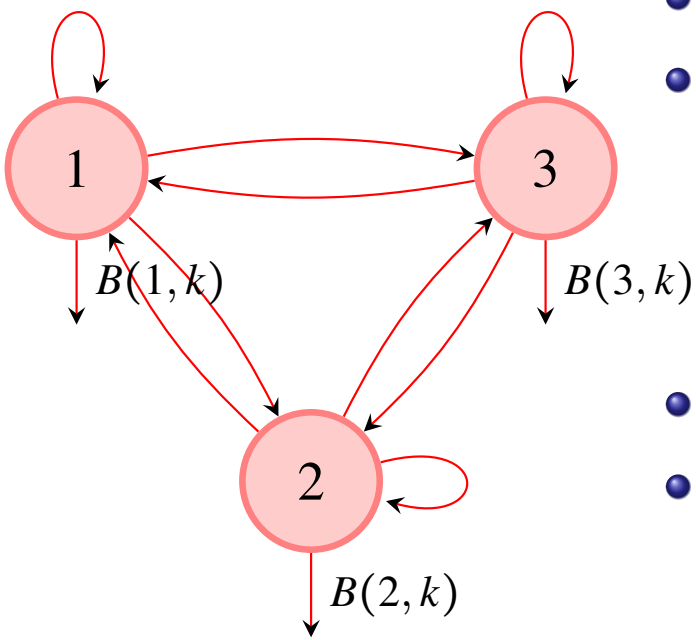
- $\{1 \ldots n_K\}$ l'ensemble des observations,
- $\boldsymbol{B}$ la matrice d'observations :

$$B(j,k) = \mathrm{P}(X_t = k \,|\, Q_t = j) = \mathrm{P}(k \,|\, j)$$

- $\theta = \{\pi, A, B\}$: paramètres du HMM

Modèles de Markov Cachés - Hidden Markov Model (HMM)

Définition (cas discret)



Un modèle de Markov « caché » est défini par $(\{1...n_S\}, \{1...n_K\}, \boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B})$.

- $\{1...n_S\}$ l'ensemble des états,
- $\boldsymbol{\pi}$ les probabilités initiales : $\pi(i) = \mathrm{P}(Q_1 = i)$,
- $\boldsymbol{A}$ la matrice de transition (indépendant de t) :

$$A(i,j) = \mathrm{P}(q_t = j \,|\, q_{t-1} = i) = \mathrm{P}(j \,|\, i)$$

- $\{1...n_K\}$ l'ensemble des observations,
- $\boldsymbol{B}$ la matrice d'observations :

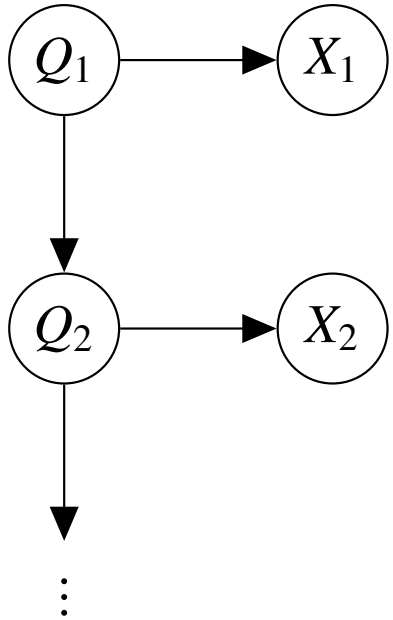
$$B(j,k) = \mathrm{P}(X_t = k \,|\, Q_t = j) = \mathrm{P}(k \,|\, j)$$

- $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}\}$: paramètres du HMM

HMM : probabilité jointe

Le modèle de génération (T est connu)

- Choisir $q_1 \sim \pi$
- Répéter $t \in \{1 \dots T\}$
  - tirer une observation $x_t \sim B(q_t, .)$
  - choisir l'état suivant $q_{t+1} \sim A(q_t, .)$



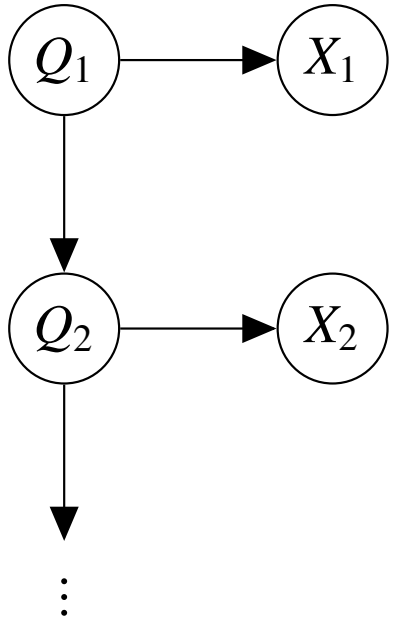
La probabilité jointe : $P(q_1 \dots q_T, x_1 \dots x_T \mid \theta)$

$$P(Q_{[1:T]} = q_1 \dots q_T, X_{[1:T]} = x_1 \dots x_T \mid \theta) = \pi(q_1) B(q_1, x_1) \prod_{t=2}^{T} A(q_{t-1}, q_t) B(q_t, x_t)$$

HMM : probabilité jointe

Le modèle de génération (T est connu)

- Choisir $q_1 \sim \pi$
- Répéter $t \in \{1 \dots T\}$
  - tirer une observation $x_t \sim B(q_t, .)$
  - choisir l'état suivant $q_{t+1} \sim A(q_t, .)$



La probabilité jointe : $\mathrm{P}(q_1 \dots q_T, x_1 \dots x_T \mid \theta)$

$$\mathrm{P}(Q_{[1:T]} = q_1 \dots q_T, X_{[1:T]} = x_1 \dots x_T \mid \theta) = \pi(q_1) B(q_1, x_1) \prod_{t=2}^{T} A(q_{t-1}, q_t) B(q_t, x_t)$$

Les 3 (ou 4) problèmes des HMMs

Prédiction, évaluation

$$\mathrm{P}(X_{[1:T]} \,|\, \boldsymbol{\theta})$$

Classification de séquence, décodage (version 1)

$$q^*_{[1:T]} = \underset{q_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(q_{[1:T]} \,|\, x_{[1:T]}) = \underset{q_{[1:T]}}{\operatorname{argmax}} \prod_{t=1}^{T} \mathrm{P}(q_t \,|\, q_{t-1})\, \mathrm{P}(x_t \,|\, q_t)$$

avec $\mathrm{P}(Q_1 \,|\, Q_0) = \pi(Q)$

Classification de séquence, décodage (version 2)

$$q^*_t = \underset{s \in \mathcal{Q}}{\operatorname{argmax}} \mathrm{P}(Q_t = q \,|\, X_{[1:T]}), \forall t = 1 \ldots T$$

Apprentissage

estimer $\boldsymbol{\theta} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ (ou les distributions a posteriori correspondantes)

avec des données de supervision .. ou sans

Calculer la probabilité d'une séquence d'observation

Formellement : marginaliser les séquence d'états

Soit $X_{[1:T]}$ la séquence observée de longueur T :

$$\mathrm{P}(X_{[1:T]};\boldsymbol{\theta}) = \sum_{Q_{[1:T]}} \mathrm{P}(X_{[1:T]}, Q_{[1:T]} \,|\, \boldsymbol{\theta}) = \sum_{Q_{[1:T]}} \mathrm{P}(X_{[1:T]} \,|\, Q_{[1:T]}, \boldsymbol{\theta})\, \mathrm{P}(Q_{[1:T]};\boldsymbol{\theta})$$

Factorisation de la loi jointe :

- $\mathrm{P}(Q_{[1:T]};\theta) = \pi(q_1)A(q_1,q_2)A(q_2,q_3)\ldots A(q_{T-1},q_T)$
- $\mathrm{P}(X_{[1:T]} \,|\, Q_{[1:T]},\theta) = B(q_1,x_1)B(q_2,x_2)\ldots B(q_T,x_T)$

Calculer la probabilité d'une séquence d'observation

Formellement : marginaliser les séquence d'états

Soit $X_{[1:T]}$ la séquence observée de longueur T :

$$\mathrm{P}(X_{[1:T]};\boldsymbol{\theta}) \quad = \quad \sum_{Q_{[1:T]}} \mathrm{P}(X_{[1:T]}, Q_{[1:T]} \,|\, \boldsymbol{\theta}) = \sum_{Q_{[1:T]}} \mathrm{P}(X_{[1:T]} \,|\, Q_{[1:T]}, \boldsymbol{\theta})\, \mathrm{P}(Q_{[1:T]};\boldsymbol{\theta})$$

Factorisation de la loi jointe :

- $\mathrm{P}(Q_{[1:T]};\boldsymbol{\theta}) = \pi(q_1)A(q_1,q_2)A(q_2,q_3)\ldots A(q_{T-1},q_T)$
- $\mathrm{P}(X_{[1:T]} \,|\, Q_{[1:T]},\boldsymbol{\theta}) = B(q_1,x_1)B(q_2,x_2)\ldots B(q_T,x_T)$

Calculer la probabilité d'une séquence d'observation

Réponse brutale

$$\mathrm{P}(X_{[1:T]};\boldsymbol{\theta}) = \sum_{Q_{[1:T]}} \pi(q_1)B(q_1,x_1)\prod_{t=2}^{T} A(q_{t-1},q_t)B(q_t,x_t)$$

Complexité : $O(2Tn_S{}^T)$ ($2T$ produits, $n_S{}^T$ fois)

- $n = 5$ et $|S| = 3 \rightarrow 2430$
- 5 et $|S| = 100 \rightarrow 10^{11}$
- 10 et $|S| = 100 \rightarrow 2^{21}$

Meilleure solution : par programmation dynamique

Calculer la probabilité d'une séquence d'observation

Réponse brutale

$$\mathrm{P}(X_{[1:T]};\boldsymbol{\theta}) = \sum_{Q_{[1:T]}} \pi(q_1)B(q_1,x_1)\prod_{t=2}^{T} A(q_{t-1},q_t)B(q_t,x_t)$$

Complexité : $O(2Tn_S{}^T)$ ($2T$ produits, $n_S{}^T$ fois)

- $n = 5$ et $|S| = 3 \rightarrow 2430$
- 5 et $|S| = 100 \rightarrow 10^{11}$
- 10 et $|S| = 100 \rightarrow 2^{21}$

Meilleure solution : par programmation dynamique

Calculer la probabilité d'une séquence d'observation

Réponse brutale

$$\mathrm{P}(X_{[1:T]};\boldsymbol{\theta}) = \sum_{Q_{[1:T]}} \pi(q_1)B(q_1,x_1)\prod_{t=2}^{T} A(q_{t-1},q_t)B(q_t,x_t)$$

Complexité : $O(2Tn_S{}^T)$ ($2T$ produits, $n_S{}^T$ fois)

- $n = 5$ et $|S| = 3 \rightarrow 2430$
- 5 et $|S| = 100 \rightarrow 10^{11}$
- 10 et $|S| = 100 \rightarrow 2^{21}$

Meilleure solution : par programmation dynamique

Probabilité d'une séquence

à l'endroit, ...

Considérons la probabilité de se retrouver dans l'état $q_t = s_i$ à l'instant t après avoir observé la séquence d'observation $x_1^t = x_1 \dots x_t$

$$
\begin{aligned}
\alpha_t(i) &= \mathrm{P}(x_{[1:t]}, q_t = i; \theta) \\
&= \mathrm{P}(x_t \,|\, x_{[1:t-1]}, q_t = i, \theta)\, \mathrm{P}(x_{[1:t-1]}, q_t = i; \theta) \\
&= \mathrm{P}(x_t \,|\, q_t = i; \theta)\, \mathrm{P}(x_{[1:t-1]}, q_t = i; \theta) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(x_{[1:t-1]}, q_t = i, q_{t-1} = j; \theta) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(q_t = i \,|\, x_{[1:t-1]}, q_{t-1} = j; \theta)\, \mathrm{P}(x_{[1:t-1]}, q_{t-1} = j; \theta) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(q_t = i \,|\, q_{t-1} = j; \theta)\, \mathrm{P}(x_{[1:t-1]}, q_{t-1} = j; \theta) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} A(j, i)\, \alpha_{t-1}(j)
\end{aligned}
$$

Probabilité d'une séquence

à l'endroit, ...

Considérons la probabilité de se retrouver dans l'état $q_t = s_i$ à l'instant t après avoir observé la séquence d'observation $x_1^t = x_1 \dots x_t$

$$
\begin{aligned}
\alpha_t(i) &= \mathrm{P}(x_{[1:t]}, q_t = i; \boldsymbol{\theta}) \\
&= \mathrm{P}(x_t \,|\, x_{[1:t-1]}, q_t = i, \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_t = i; \boldsymbol{\theta}) \\
&= \mathrm{P}(x_t \,|\, q_t = i; \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_t = i; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(x_{[1:t-1]}, q_t = i, q_{t-1} = j; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(q_t = i \,|\, x_{[1:t-1]}, q_{t-1} = j; \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_{t-1} = j; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(q_t = i \,|\, q_{t-1} = j; \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_{t-1} = j; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} A(j,i)\, \alpha_{t-1}(j)
\end{aligned}
$$

Probabilité d'une séquence

à l'endroit, ...

Considérons la probabilité de se retrouver dans l'état $q_t = s_i$ à l'instant t après avoir observé la séquence d'observation $x_1^t = x_1 \dots x_t$

$$
\begin{aligned}
\alpha_t(i) &= \mathrm{P}(x_{[1:t]}, q_t = i; \boldsymbol{\theta}) \\
&= \mathrm{P}(x_t \,|\, x_{[1:t-1]}, q_t = i, \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_t = i; \boldsymbol{\theta}) \\
&= \mathrm{P}(x_t \,|\, q_t = i; \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_t = i; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(x_{[1:t-1]}, q_t = i, q_{t-1} = j; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(q_t = i \,|\, x_{[1:t-1]}, q_{t-1} = j; \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_{t-1} = j; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} \mathrm{P}(q_t = i \,|\, q_{t-1} = j; \boldsymbol{\theta})\, \mathrm{P}(x_{[1:t-1]}, q_{t-1} = j; \boldsymbol{\theta}) \\
&= B(q_t, x_t) \sum_{1 \le j \le |S|} A(j,i)\alpha_{t-1}(j)
\end{aligned}
$$

Probabilité d'une séquence

Formulation récursive

Algorithme forward

1. **Initialisation :** $\alpha_1(i) = \pi(i)B(i, x_1)$
2. **Récurrence :** $\alpha_t(i) = \sum_{1 \leq j \leq n_S} \alpha_{t-1}(j)A(j, i)B(i, x_t), \forall t = 1 \dots T$
3. **Fin :** $\mathrm{P}(X_{[1:T]}; \boldsymbol{\theta}) = \sum_{1 \leq i \leq n_S} \alpha_T(i)$

Complexité

en temps $O(T{n_S}^2)$ et en espace $O(Tn_S)$

Le calcul de α_t s'écrit matriciellement (avec $B_t(i) = B(i, x_t)$) :

$$\boldsymbol{\alpha}_t = \mathbf{A}\boldsymbol{\alpha}_{t-1} \cdot \boldsymbol{B}_t$$

Probabilité d'une séquence

Formulation récursive

Algorithme forward

1. **Initialisation :** $\alpha_1(i) = \pi(i)B(i, x_1)$
2. **Récurrence :** $\alpha_t(i) = \sum_{1 \leq j \leq n_S} \alpha_{t-1}(j)A(j, i)B(i, x_t), \forall t = 1 \ldots T$
3. **Fin :** $\mathrm{P}(X_{[1:T]}; \boldsymbol{\theta}) = \sum_{1 \leq i \leq n_S} \alpha_T(i)$

Complexité

en temps $O(T{n_S}^2)$ et en espace $O(Tn_S)$

Le calcul de $\boldsymbol{\alpha}_t$ s'écrit matriciellement (avec $B_t(i) = B(i, x_t)$) :

$$\boldsymbol{\alpha_t} = \mathbf{A}\boldsymbol{\alpha}_{t-1} \cdot \boldsymbol{B_t}$$

Probabilité d'une séquence

à l'envers...

$\beta_t(i)$: probabilité d'observer $x_{[t+1:T]}$ sachant $q_t = i$:

$$\beta_t(i) = \mathrm{P}(x_{[t+1:T]} \,|\, q_t = i; \boldsymbol{\theta}) = \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{[t+1:T]}, q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta})$$

$$= \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{[t+1:T]} \,|\, q_t = i, q_{t+1} = j; \boldsymbol{\theta}) \, \mathrm{P}(q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta})$$

$$= \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{t+1}, x_{[t+2:T]} \,|\, q_t = i, q_{t+1} = j; \boldsymbol{\theta}) \, \mathrm{P}(q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta})$$

$$= \sum_{1 \leq j \leq n_S} B(j, x_{t+1}) \beta_{t+1}(j) A(i,j)$$

Récursion pour β

1. **Initialisation :** $\forall i, \beta_T(i) = 1$
2. **Récurrence :** $\beta_t(i) = \sum_{1 \leq j \leq n_S} \beta_{t+1}(j) A(i,j) B(j, x_{t+1})$
3. **Fin :** $\mathrm{P}(x_{[1:T]}; \boldsymbol{\theta}) = \sum_{1 \leq j \leq n_S} \pi(i) B(i, x_1) \beta_1(i)$

Probabilité d'une séquence

à l'envers...

$\beta_t(i)$: probabilité d'observer $x_{[t+1:T]}$ sachant $q_t = i$:

$$\begin{aligned}
\beta_t(i) &= \mathrm{P}(x_{[t+1:T]} \,|\, q_t = i; \boldsymbol{\theta}) = \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{[t+1:T]}, q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta}) \\
&= \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{[t+1:T]} \,|\, q_t = i, q_{t+1} = j; \boldsymbol{\theta}) \, \mathrm{P}(q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta}) \\
&= \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{t+1}, x_{[t+2:T]} \,|\, q_t = i, q_{t+1} = j; \boldsymbol{\theta}) \, \mathrm{P}(q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta}) \\
&= \sum_{1 \leq j \leq n_S} B(j, x_{t+1}) \beta_{t+1}(j) A(i,j)
\end{aligned}$$

Récursion pour β

1. **Initialisation :** $\forall i, \beta_T(i) = 1$
2. **Récurrence :** $\beta_t(i) = \sum_{1 \leq j \leq n_S} \beta_{t+1}(j) A(i,j) B(j, x_{t+1})$
3. **Fin :** $\mathrm{P}(x_{[1:T]}; \theta) = \sum_{1 \leq j \leq n_S} \pi(i) B(i, x_1) \beta_1(i)$

Probabilité d'une séquence

à l'envers...

$\beta_t(i)$: probabilité d'observer $x_{[t+1:T]}$ sachant $q_t = i$:

$$\begin{aligned}
\beta_t(i) &= \mathrm{P}(x_{[t+1:T]} \,|\, q_t = i; \boldsymbol{\theta}) = \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{[t+1:T]}, q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta}) \\
&= \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{[t+1:T]} \,|\, q_t = i, q_{t+1} = j; \boldsymbol{\theta}) \, \mathrm{P}(q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta}) \\
&= \sum_{1 \leq j \leq n_S} \mathrm{P}(x_{t+1}, x_{[t+2:T]} \,|\, q_t = i, q_{t+1} = j; \boldsymbol{\theta}) \, \mathrm{P}(q_{t+1} = j \,|\, q_t = i; \boldsymbol{\theta}) \\
&= \sum_{1 \leq j \leq n_S} B(j, x_{t+1}) \beta_{t+1}(j) A(i,j)
\end{aligned}$$

Récursion pour β

1. **Initialisation :** $\forall i, \beta_T(i) = 1$
2. **Récurrence :** $\beta_t(i) = \sum_{1 \leq j \leq n_S} \beta_{t+1}(j) A(i,j) B(j, x_{t+1})$
3. **Fin :** $\mathrm{P}(x_{[1:T]}; \boldsymbol{\theta}) = \sum_{1 \leq j \leq n_S} \pi(i) B(i, x_1) \beta_1(i)$

Évaluation par l'algorithme forward-backward

Par les deux bouts

Décompositions d'un chemin

1. D'abord les t premières observations, pour aboutir à l'état $i \rightarrow \alpha_t(i)$
2. Puis les $T - t - 1$ dernières, en partant de i : $\beta_t(i)$.

$$\forall t = 1 \ldots T : \mathrm{P}(x_{[1:T]}; \boldsymbol{\theta}) = \sum_{1 \leq i \leq n_S} \mathrm{P}(x_{[1:T]}, q_t = i; \boldsymbol{\theta}) = \sum_{1 \leq i \leq n_S} \alpha_t(i)\beta_t(i)$$

- α pour t croissant à partir du début,
- β pour t décroissant à partir de la fin.
- Complexité : $O(T{n_S}^2)$

Classification de séquence (décodage)

Position du problème d'inférence

Observant $x_{[1:T]}$, quelle est la séquence d'états la plus probable ?

$$q^*_{[1:T]} = \underset{q_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(Q_{[1:T]} \,|\, X_{[1:T]}; \theta) = \underset{Q_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(Q_{[1:T]}, X_{[1:T]} \,|\, \theta)$$

Problème combinatoire : trouver le meilleur parmi ${n_S}^T$ chemins possibles.

Programmation dynamique

$\delta_t(i)$: la probabilité du meilleur chemin menant jusqu'à l'état s_i à l'instant t :

$$\delta_t(i) = max_{q_{[1:t-1]}} \mathrm{P}(q_{[1:t-1]}, q_t = i, x_{[1:t-1]} \,|\, \theta)$$

Classification de séquence (décodage)

Position du problème d'inférence

Observant $x_{[1:T]}$, quelle est la séquence d'états la plus probable ?

$$q^*_{[1:T]} = \underset{q_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(Q_{[1:T]} \,|\, X_{[1:T]}; \theta) = \underset{Q_{[1:T]}}{\operatorname{argmax}} \mathrm{P}(Q_{[1:T]}, X_{[1:T]} \,|\, \theta)$$

Problème combinatoire : trouver le meilleur parmi ${n_S}^T$ chemins possibles.

Programmation dynamique

$\delta_t(i)$: la probabilité du meilleur chemin menant jusqu'à l'état s_i à l'instant t :

$$\delta_t(i) = max_{q_{[1:t-1]}} \mathrm{P}(q_{[1:t-1]}, q_t = i, x_{[1:t-1]} \,|\, \theta)$$

Algorithme de Viterbi

Calcul de $\delta_t(i)$ par récurrence en gardant la trace du meilleur chemin $\psi_t(i)$:

1. **Initialisation :** $\delta_1(i) = \pi(i)B(i, x_1)$ et $\psi_1(i) = 0$, pour $1 \leq i \leq n_S$.
2. **Récurrence :**
   $\delta_t(i) = \max_{1 \leq j \leq n_S} \left(\delta_{t-1}(j)A(j,i)\right)B(i,x_t)$, pour $1 \leq i \leq n_S$
   $\psi_t(i) = \text{argmax}_{1 \leq j \leq n_S} \left(\delta_{t-1}(j)A(j,i)\right)B(i,x_t)$, pour $1 \leq i \leq n_S$
3. **Fin :**
   $q_T^* = argmax_{1 \leq i \leq n_S}(\delta_T(i))$
   $\mathrm{P}(q_{[1:T]}^*; \boldsymbol{\theta}) = max_{1 \leq i \leq n_S}(\delta_T(i))$
4. Le meilleur chemin est obtenu par *backtracking* (parcours arrière) :

$$q_t^* = \psi_{t+1}(q_{t+1}^*)$$

Viterbi ≡ plus court chemin dans un graphe valué (Bellman, Dijkstraa.)

Algorithme de Viterbi

Calcul de $\delta_t(i)$ par récurrence en gardant la trace du meilleur chemin $\psi_t(i)$:

1. **Initialisation :** $\delta_1(i) = \pi(i)B(i, x_1)$ et $\psi_1(i) = 0$, pour $1 \leq i \leq n_S$.
2. **Récurrence :**
   $\delta_t(i) = \max_{1 \leq j \leq n_S} \left(\delta_{t-1}(j)A(j,i)\right)B(i, x_t)$, pour $1 \leq i \leq n_S$
   $\psi_t(i) = \text{argmax}_{1 \leq j \leq n_S} \left(\delta_{t-1}(j)A(j,i)\right)B(i, x_t)$, pour $1 \leq i \leq n_S$
3. **Fin :**
   $q_T^* = argmax_{1 \leq i \leq n_S}(\delta_T(i))$
   $\mathrm{P}(q_{[1:T]}^*; \boldsymbol{\theta}) = max_{1 \leq i \leq n_S}(\delta_T(i))$
4. Le meilleur chemin est obtenu par *backtracking* (parcours arrière) :

$$q_t^* = \psi_{t+1}(q_{t+1}^*)$$

Viterbi ≡ plus court chemin dans un graphe valué (Bellman, Dijkstraa.)

Implémenter Viterbi pour l'étiquetage morphosyntaxique

Principe

Modèle

HMM, un état par étiquette, une multinomiale sur V par état.

Les nœuds du graphe de recherche

Le quadruplet : { mot, étiquette, score, père}

Développement d'un nœud

- Nœuds successeurs : $\{mot, tag\}$, avec $\mathrm{P}(mot \mid tag) \neq 0$
- Score de $n = (w_t, e)$ successeur de $m = (w_{t-1}, e')$:
  $score(n) = score(m)\,\mathrm{P}(e \mid e')\,\mathrm{P}(w_t \mid e)$
- Si un successeur est déjà un nœud connu alors mise à jour du père.
- Étape finale : choisir à $t = T$ le meilleur nœud, puis retour arrière.

Viterbi et Forward

Deux instances du même algorithme

- Viterbi (log)
  - les scores s'aditionnent (+) le long des chemins
  - les scores se maximisent (max) quand des chemins fusionnent
- Forward
  - les scores se multiplient (x) le long des chemins
  - les scores s'ajoutent (+) quand des chemins fusionnent
- Généralisation : semi-anneau $(\mathbb{K}, \oplus, \otimes, 0, 1)$
  - les scores se $\otimes$ le long des chemins
  - les scores se $\oplus$ quand des chemins fusionnent

  Cadre algébrique :
  - $(\mathbb{K}, \oplus, 0)$ monoïde commutatif
  - $(\mathbb{K}, \otimes, 0)$ monoïde
  - $\oplus$ distributif par rapport à $\otimes$

Viterbi et Forward

Deux instances du même algorithme

- Viterbi (log)
  - les scores s'aditionnent (+) le long des chemins
  - les scores se maximisent (max) quand des chemins fusionnent
- Forward
  - les scores se multiplient (x) le long des chemins
  - les scores s'ajoutent (+) quand des chemins fusionnent
- Généralisation : semi-anneau $(\mathbb{K}, \oplus, \otimes, 0, 1)$
  - les scores se $\otimes$ le long des chemins
  - les scores se $\oplus$ quand des chemins fusionnent

  Cadre algébrique :
  - $(\mathbb{K}, \oplus, 0)$ monoïde commutatif
  - $(\mathbb{K}, \otimes, 0)$ monoïde
  - $\oplus$ distributif par rapport à $\otimes$

Viterbi et Forward

Deux instances du même algorithme

- Viterbi (log)
  - les scores s'aditionnent (+) le long des chemins
  - les scores se maximisent (max) quand des chemins fusionnent
- Forward
  - les scores se multiplient (x) le long des chemins
  - les scores s'ajoutent (+) quand des chemins fusionnent
- Généralisation : semi-anneau $(\mathbb{K}, \oplus, \otimes, 0, 1)$
  - les scores se $\otimes$ le long des chemins
  - les scores se $\oplus$ quand des chemins fusionnent

  Cadre algébrique :
  - $(\mathbb{K}, \oplus, 0)$ monoïde commutatif
  - $(\mathbb{K}, \otimes, 0)$ monoïde
  - $\oplus$ distributif par rapport à $\otimes$

Décodage optimal ? Une variante

Minimiser l'erreur d'étiquetage

- $\text{argmax}_{Q_{[1:T]}} \mathrm{P}(q_{[1:T]}, x_{[1:T]})$ minimise l'espérance du nombre d'erreurs par séquence
- Minimiser du nombre d'erreurs d'étiquetage par position :

$$\forall t = 1 \ldots T, q_t^* = \underset{q_t}{\text{argmax}}\, \mathrm{P}(q_t \,|\, x_{[1:T]})$$

- Un calcul familier ?

$$\begin{aligned}
\mathrm{P}(q_t = i \,|\, x_{[1:T]}; \boldsymbol{\theta}) &\propto \mathrm{P}(x_{[1:T]}, q_t = i \,|\, \boldsymbol{\theta}) \\
&\propto \mathrm{P}(x_{[1:t]}, q_t = i, x_{[t+1:T]} \,|\, \boldsymbol{\theta}) \\
&\propto \mathrm{P}(x_{[1:t]}, q_t = i \,|\, \boldsymbol{\theta}) \mathrm{P}(x_{[t+1:T]} \,|\, x_{[1:t]}, q_t = i; \boldsymbol{\theta}) \\
&\propto \alpha_t(i) \beta_t(i)
\end{aligned}$$

Décodage optimal ? Une variante

Minimiser l'erreur d'étiquetage

- $\text{argmax}_{Q_{[1:T]}} \mathrm{P}(q_{[1:T]}, x_{[1:T]})$ minimise l'espérance du nombre d'erreurs par séquence
- Minimiser du nombre d'erreurs d'étiquetage par position :

$$\forall t = 1 \ldots T, q_t^* = \underset{q_t}{\text{argmax}}\, \mathrm{P}(q_t \,|\, x_{[1:T]})$$

- Un calcul familier ?

$$\begin{aligned}
\mathrm{P}(q_t = i \,|\, x_{[1:T]}; \boldsymbol{\theta}) &\propto \mathrm{P}(x_{[1:T]}, q_t = i \,|\, \boldsymbol{\theta}) \\
&\propto \mathrm{P}(x_{[1:t]}, q_t = i, x_{[t+1:T]} \,|\, \boldsymbol{\theta}) \\
&\propto \mathrm{P}(x_{[1:t]}, q_t = i \,|\, \boldsymbol{\theta}) \mathrm{P}(x_{[t+1:T]} \,|\, x_{[1:t]}, q_t = i; \boldsymbol{\theta}) \\
&\propto \alpha_t(i) \beta_t(i)
\end{aligned}$$

Apprendre des HMMs avec des exemples annotés

L'apprentissage supervisé

- Naïves Bayes (multinomial) : $\theta_{i,j} = \mathrm{P}(x = i \,|\, y = j), \alpha_j = P(y = j)$

$$\ell(x, y = j) \propto \log(\alpha_j) + \sum_{i=1}^{d} C_i \log(\theta_{ij})$$

$$\widehat{\theta}_{ij} = \frac{C_i}{\sum_k C_k}$$

- Modèle de Markov $\theta_{i,j} = \mathrm{P}(x_t = i \,|\, x_{t-1} = j), \pi_i$

$$\ell(\boldsymbol{\theta}) \propto \log(\pi_{x_1}) + \sum_{t=2}^{T} \log(\theta_{x_t x_{t-1}})$$

$$\widehat{\theta}_{ij} = \frac{C(i,j)}{\sum_k C(i,k)}$$

- Modèles de Markov cachés $\boldsymbol{\theta} = \{A_{i,j}, B_{j,o}, \pi_i\}$

$$\ell(\boldsymbol{\theta}) \propto \log(\pi(q_1)) + \log(B(x_1, q_1)) + \sum_t \log(A(q_{t-1}, q_t)) + \log(B(q_t, x_t))$$

$$\widehat{\theta}_{ij} = ??$$

idem pour l'estimateur MAP

Apprendre des HMMs avec des exemples annotés

L'apprentissage supervisé

- Naïves Bayes (multinomial) : $\theta_{i,j} = \mathrm{P}(x = i \,|\, y = j), \alpha_j = P(y = j)$

$$\ell(x, y = j) \quad \propto \quad \log(\alpha_j) + \sum_{i=1}^{d} C_i \log(\theta_{ij})$$

$$\widehat{\theta}_{ij} \quad = \quad \frac{C_i}{\sum_k C_k}$$

- Modèle de Markov $\theta_{i,j} = \mathrm{P}(x_t = i \,|\, x_{t-1} = j), \pi_i$

$$\ell(\boldsymbol{\theta}) \quad \propto \quad \log(\pi_{x_1}) + \sum_{t=2}^{T} \log(\theta_{x_t x_{t-1}})$$

$$\widehat{\theta}_{ij} \quad = \quad \frac{C(i,j)}{\sum_k C(i,k)}$$

- Modèles de Markov cachés $\theta = \{A_{i,j}, B_{j,o}, \pi_i\}$

$$\ell(\boldsymbol{\theta}) \quad \propto \quad \log(\pi(q_1)) + \log(B(x_1, q_1)) + \sum_t \log(A(q_{t-1}, q_t)) + \log(B(q_t, x_t))$$

$$\widehat{\theta}_{ij} \quad = \quad \frac{C(q_t = i, q_{t+1} = j)}{\sum_k C(q_t = i, q_{t+1} = k)}, \frac{C(q_t = i, x_t = j)}{\sum_k C(q_t = i, x_t = k)}$$

idem pour l'estimateur MAP

Apprendre des HMMs avec des exemples annotés

L'apprentissage supervisé

- Naïves Bayes (multinomial) : $\theta_{i,j} = \mathrm{P}(x = i \,|\, y = j), \alpha_j = P(y = j)$

$$\ell(x, y = j) \quad \propto \quad \log(\alpha_j) + \sum_{i=1}^{d} C_i \log(\theta_{ij})$$

$$\widehat{\theta}_{ij} \quad = \quad \frac{C_i}{\sum_k C_k}$$

- Modèle de Markov $\theta_{i,j} = \mathrm{P}(x_t = i \,|\, x_{t-1} = j), \pi_i$

$$\ell(\boldsymbol{\theta}) \quad \propto \quad \log(\pi_{x_1}) + \sum_{t=2}^{T} \log(\theta_{x_t x_{t-1}})$$

$$\widehat{\theta}_{ij} \quad = \quad \frac{C(i,j)}{\sum_k C(i,k)}$$

- Modèles de Markov cachés $\boldsymbol{\theta} = \{A_{i,j}, B_{j,o}, \pi_i\}$

$$\ell(\boldsymbol{\theta}) \quad \propto \quad \log(\pi(q_1)) + \log(B(x_1, q_1)) + \sum_t \log(A(q_{t-1}, q_t)) + \log(B(q_t, x_t))$$

$$\widehat{\theta}_{ij} \quad = \quad \frac{C(q_t = i, q_{t+1} = j)}{\sum_k C(q_t = i, q_{t+1} = k)}, \frac{C(q_t = i, x_t = j)}{\sum_k C(q_t = i, x_t = k)}$$

idem pour l'estimateur MAP

Apprentissage sans supervision

Objectif

Estimer $\boldsymbol{\theta} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ maximisant la log-vraisemblance : $\ell(\boldsymbol{\theta}) = \log \prod_i \mathrm{P}(X^i_{[1:T]}; \boldsymbol{\theta})$; les états sont latents.

EM pour les HMM : Baum-Welch

1. Choisir θ^0
2. Calculer θ^1 à partir de θ^0
3. Répéter jusqu'à convergence

$$\theta^n \leftarrow \operatorname{argmax} Q_{\theta^{n-1}}(\mathbb{C}, \theta)$$

Garantie : à chaque itération,

$$\mathrm{P}(X_{[1:T]} \mid \theta^n) \geq \mathrm{P}(X_{[1:T]} \mid \theta^{n-1})$$

Apprentissage sans supervision

Objectif

Estimer $\boldsymbol{\theta} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ maximisant la log-vraisemblance : $\ell(\boldsymbol{\theta}) = \log \prod_i \mathrm{P}(X^i_{[1:T]}; \boldsymbol{\theta})$; les états sont latents.

EM pour les HMM : Baum-Welch

1. Choisir $\boldsymbol{\theta}^0$
2. Calculer $\boldsymbol{\theta}^1$ à partir de $\boldsymbol{\theta}^0$
3. Répéter jusqu'à convergence

$$\boldsymbol{\theta}^n \leftarrow \operatorname{argmax} Q_{\boldsymbol{\theta}^{n-1}}(\mathbb{C}, \boldsymbol{\theta})$$

Garantie : à chaque itération,

$$\mathrm{P}(X_{[1:T]} \mid \boldsymbol{\theta}^n) \geq \mathrm{P}(X_{[1:T]} \mid \boldsymbol{\theta}^{n-1})$$

Apprentissage sans supervision

Retrouver les formules de réestimation

États connus, estimation supervisée

$$\widehat{\pi}(i) = \frac{C(q_1 = i)}{\sum_j C(q_1 = j)}$$

$$\widehat{A}(i,j) = \frac{\sum_{t\geq 1} C(q_t = i, q_{t+1} = j)}{\sum_k \sum_{t\geq 1} C(q_t = i, q_{t+1} = k)}$$

$$\widehat{B}(i,k) = \frac{\sum_{t\geq 1} C(q_t = i, x_t = k)}{\sum_{k'} \sum_{t\geq 1} C(q_t = i, x_t = k')}$$

Apprentissage style "Viterbi"

1. Choisir θ^0
2. Calculer $q^*_{[1:T]} \,|\, \theta^{n-1}$ (Viterbi), utiliser ces annotations pour calculer θ^n
3. Répéter jusqu'à convergence

EM utilise plusieurs étiquetages incertains.

Apprentissage sans supervision

Retrouver les formules de réestimation

États connus, estimation supervisée

$$\widehat{\pi}(i) = \frac{C(q_1 = i)}{\sum_j C(q_1 = j)}$$

$$\widehat{A}(i,j) = \frac{\sum_{t\geq 1} C(q_t = i, q_{t+1} = j)}{\sum_k \sum_{t\geq 1} C(q_t = i, q_{t+1} = k)}$$

$$\widehat{B}(i,k) = \frac{\sum_{t\geq 1} C(q_t = i, x_t = k)}{\sum_{k'} \sum_{t\geq 1} C(q_t = i, x_t = k')}$$

Apprentissage style “Viterbi”

1. Choisir θ^0
2. Calculer $q^*_{[1:T]} \,|\, \theta^{n-1}$ (Viterbi), utiliser ces annotations pour calculer θ^n
3. Répéter jusqu’à convergence

EM utilise plusieurs étiquetages incertains.

Retour de l'algorithme EM

La maximisation de la fonction auxiliaire

Règle de mise à jour de EM

$$\boldsymbol{\theta}^n \leftarrow \operatorname*{argmax}_{\theta} Q_{\boldsymbol{\theta}^{n-1}}(\boldsymbol{\theta}) = \mathbb{E}_{\mathrm{P}(Q \mid X;\boldsymbol{\theta}^{n-1})}(\log \mathrm{P}(X, Q \mid \boldsymbol{\theta}))$$

La fonction auxiliaire

$$Q_{\boldsymbol{\theta}^{n-1}}(\boldsymbol{\theta}) = \sum_{q_1 \cdots q_T} \mathrm{P}(q_{[1:T]} \mid x_{[1:T]}; \boldsymbol{\theta}^{n-1}) \log \left(\pi(q_1) \prod_{t=1}^{T} \mathrm{P}(x_t \mid q_t) \prod_{t=2}^{T} \mathrm{P}(q_t \mid q_{t-1}) \right)$$

Le M de EM

Une histoire de maximisation

Demandez le programme

$$\max_{\theta} Q_{\boldsymbol{\theta}^{n-1}}(\boldsymbol{\theta}) = \mathbb{E}_{\mathrm{P}(Q\,|\,X;\boldsymbol{\theta}^{n-1})}(\log \mathrm{P}(X, Q\,|\,\boldsymbol{\theta}))$$

$$\begin{cases} \forall i, \sum_j A(i,j) = 1, \Rightarrow \ |S| \text{ contraintes, } \lambda_i \\ \forall i, \sum_o B(i,o) = 1, \Rightarrow \ |S| \text{ contraintes, } \mu_i \\ \sum_i \pi(i) = 1, \Rightarrow \ 1 \text{ contrainte, } \nu \end{cases}$$

Conditions d'optimalité

$$\widehat{A}(i,j) = \frac{\text{Espérance de } C(q_t = i, q_{t+1} = j)}{\sum_k \text{Espérance de } C(q_t = i, q_{t+1} = k)}$$

(idem pour $\mathbf{B}$ et π)

Les espérances sont calculées sous $\mathrm{P}(Q_{[1:T]}\,|\,X_{[1:T]}; \boldsymbol{\theta}^{n-1})$

Formules de réestimation

Le calcul de l'espérance

$$
\begin{aligned}
\mathbb{E}(C(i,j)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1}) &= \sum_{q_{[1:T]}} C(i,j)\,\mathrm{P}(q_{[1:T]}\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1}) \\
&= \sum_{q_{[1:T]}} \sum_{t=2}^{T} \mathbb{I}(q_{t-1}=i,q_t=j)\,\mathrm{P}(q_{[1:T]}\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1}) \\
&= \sum_{t=2}^{T} \sum_{q_{[1:T]}} \mathbb{I}(q_{t-1}=i,q_t=j)\,\mathrm{P}(q_{[1:T]}\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1}) \\
&= \sum_{t=2}^{T} \mathrm{P}(q_{t-1}=i,q_t=j\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})
\end{aligned}
$$

Formules de réestimation

Retour des $\alpha()$ et des $\beta()$

$$\mathrm{P}(q_{t-1}=i, q_t=j \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1}) \quad \propto \quad \mathrm{P}(x_{[1:T]}, q_{t-1}=i, q_t=j \,|\, \boldsymbol{\theta}^{n-1})$$

$$\propto \quad \mathrm{P}(x_{[1:t-1]}, q_{t-1}=i \,|\, \theta^i)\, \mathrm{P}(q_t=j \,|\, q_{t-1}=i; \theta^{n-1}) \times$$

$$\mathrm{P}(x_t \,|\, q_t=j)\, \mathrm{P}(x_{[t+1:T]}, q_t=j \,|\, \boldsymbol{\theta}^{n-1})$$

$$\propto \quad \alpha_{t-1}(i) A(i,j) B(j, x_t) \beta_t(j)$$

Le facteur de normalisation ? $\mathrm{P}(x_{[1:T]}; \boldsymbol{\theta}^{n-1})$

Formules de réestimation

Retour des $\alpha()$ et des $\beta()$

$$
\begin{aligned}
\mathrm{P}(q_{t-1}=i, q_t=j \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1}) &\propto \mathrm{P}(x_{[1:T]}, q_{t-1}=i, q_t=j \,|\, \boldsymbol{\theta}^{n-1}) \\
&\propto \mathrm{P}(x_{[1:t-1]}, q_{t-1}=i \,|\, \boldsymbol{\theta}^{i})\, \mathrm{P}(q_t=j \,|\, q_{t-1}=i; \boldsymbol{\theta}^{n-1}) \times \\
&\quad \mathrm{P}(x_t \,|\, q_t=j)\, \mathrm{P}(x_{[t+1:T]}, q_t=j \,|\, \boldsymbol{\theta}^{n-1}) \\
&\propto \alpha_{t-1}(i) A(i,j) B(j, x_t) \beta_t(j)
\end{aligned}
$$

Le facteur de normalisation ? $\mathrm{P}(x_{[1:T]}; \boldsymbol{\theta}^{n-1})$

Formules de réestimation

Retour des $\alpha()$ et des $\beta()$

$$
\begin{aligned}
\mathrm{P}(q_{t-1}=i, q_t=j \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1}) &\propto \mathrm{P}(x_{[1:T]}, q_{t-1}=i, q_t=j \,|\, \boldsymbol{\theta}^{n-1}) \\
&\propto \mathrm{P}(x_{[1:t-1]}, q_{t-1}=i \,|\, \boldsymbol{\theta}^{i})\, \mathrm{P}(q_t=j \,|\, q_{t-1}=i; \boldsymbol{\theta}^{n-1}) \times \\
&\quad \mathrm{P}(x_t \,|\, q_t=j)\, \mathrm{P}(x_{[t+1:T]}, q_t=j \,|\, \boldsymbol{\theta}^{n-1}) \\
&\propto \alpha_{t-1}(i) A(i,j) B(j, x_t) \beta_t(j)
\end{aligned}
$$

Le facteur de normalisation ? $\mathrm{P}(x_{[1:T]}; \boldsymbol{\theta}^{n-1})$

Formules de ré-estimation

On rassemble les morceaux

$$A(i,j)^n = \frac{\mathbb{E}(C(i,j) \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1})}{\sum_k \mathbb{E}(C(i,k) \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1})}$$

$$= \frac{\sum_{t=2}^{T} \mathrm{P}(q_t = j, q_{t-1} = i \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1})}{\sum_k \sum_{t=2}^{T} \mathrm{P}(q_t = k, q_{t-1} = i \,|\, x_{[1:T]}, \boldsymbol{\theta}^{n-1})}$$

$$= \frac{S(i,j)}{\sum_k S(i,k)}$$

Avec :

$$S(i,j) = \sum_{t=2}^{T} \mathrm{P}(q_t = j, q_{t-1} = i \,|\, X_{[1:T]}, \boldsymbol{\theta}^{n-1}) \propto \sum_{t=2}^{T} \alpha_{t-1}(i) A(i,j) B(j, x_t) \beta_t(j)$$

Formules de ré-estimation

On rassemble les morceaux

$$
\begin{aligned}
A(i,j)^n &= \frac{\mathbb{E}(C(i,j)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})}{\sum_k \mathbb{E}(C(i,k)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})} \\
&= \frac{\sum_{t=2}^{T} \mathrm{P}(q_t=j,q_{t-1}=i\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})}{\sum_k \sum_{t=2}^{T} \mathrm{P}(q_t=k,q_{t-1}=i\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})} \\
&= \frac{S(i,j)}{\sum_k S(i,k)}
\end{aligned}
$$

Avec :

$$
S(i,j) = \sum_{t=2}^{T} \mathrm{P}(q_t=j,q_{t-1}=i\,|\,X_{[1:T]},\theta^{n-1}) \propto \sum_{t=2}^{T} \alpha_{t-1}(i)A(i,j)B(j,x_t)\beta_t(j)
$$

Formules de ré-estimation

On rassemble les morceaux

$$
\begin{aligned}
A(i,j)^n &= \frac{\mathbb{E}(C(i,j)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})}{\sum_k \mathbb{E}(C(i,k)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})} \\
&= \frac{\sum_{t=2}^{T} \mathrm{P}(q_t=j,q_{t-1}=i\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})}{\sum_k \sum_{t=2}^{T} \mathrm{P}(q_t=k,q_{t-1}=i\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})} \\
&= \frac{S(i,j)}{\sum_k S(i,k)}
\end{aligned}
$$

Avec :

$$
S(i,j) = \sum_{t=2}^{T} \mathrm{P}(q_t=j,q_{t-1}=i\,|\,X_{[1:T]},\boldsymbol{\theta}^{n-1}) \propto \sum_{t=2}^{T} \alpha_{t-1}(i)A(i,j)B(j,x_t)\beta_t(j)
$$

Formules de ré-estimation

On rassemble les morceaux

$$
\begin{aligned}
A(i,j)^n &= \frac{\mathbb{E}(C(i,j)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})}{\sum_k \mathbb{E}(C(i,k)\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})} \\
&= \frac{\sum_{t=2}^{T} \mathrm{P}(q_t=j,q_{t-1}=i\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})}{\sum_k \sum_{t=2}^{T} \mathrm{P}(q_t=k,q_{t-1}=i\,|\,x_{[1:T]},\boldsymbol{\theta}^{n-1})} \\
&= \frac{S(i,j)}{\sum_k S(i,k)}
\end{aligned}
$$

Avec :

$$
S(i,j) = \sum_{t=2}^{T} \mathrm{P}(q_t=j,q_{t-1}=i\,|\,X_{[1:T]},\boldsymbol{\theta}^{n-1}) \propto \sum_{t=2}^{T} \alpha_{t-1}(i)A(i,j)B(j,x_t)\beta_t(j)
$$

Suite des formules de réestimation

Les autres paramètres

Notation

$$\gamma_t(i) = \mathrm{P}(q_t = i \,|\, x_{[1:T]}; \boldsymbol{\theta}^{n-1}) \propto \alpha_t(i)\beta_t(i)$$

Probabilités initiales

$$\pi^n(i) = \frac{\mathbb{E}(\mathbb{I}(q_1 = i))}{\sum_k \mathbb{E}(\mathbb{I}(q_1 = k))} = \frac{\gamma_1(i)}{\sum_k \gamma_1(k)}$$

Multinomiales associées aux états

$$B(i,o)^n = \frac{\mathbb{E}(C(q_t = i, x_t = o))}{\sum_p \mathbb{E}(C(q_t = i, x_t = o))} = \frac{\sum_{t,x_t=o} \mathrm{P}(q_t = i \,|\, x_{[1:T]}; \theta^{n-1})}{\sum_p \sum_{t,x_t=p} \mathrm{P}(q_t = i \,|\, x_{[1:T]}; \theta^{n-1})}$$

Suite des formules de réestimation

Les autres paramètres

Notation

$$\gamma_t(i) = \mathrm{P}(q_t = i \,|\, x_{[1:T]}; \boldsymbol{\theta}^{n-1}) \propto \alpha_t(i)\beta_t(i)$$

Probabilités initiales

$$\pi^n(i) = \frac{\mathbb{E}(\mathbb{I}(q_1 = i))}{\sum_k \mathbb{E}(\mathbb{I}(q_1 = k))} = \frac{\gamma_1(i)}{\sum_k \gamma_1(k)}$$

Multinomiales associées aux états

$$B(i,o)^n = \frac{\mathbb{E}(C(q_t = i, x_t = o))}{\sum_p \mathbb{E}(C(q_t = i, x_t = o))} = \frac{\sum_{t,x_t=o} \mathrm{P}(q_t = i \,|\, x_{[1:T]}; \boldsymbol{\theta}^{n-1})}{\sum_p \sum_{t,x_t=p} \mathrm{P}(q_t = i \,|\, x_{[1:T]}; \boldsymbol{\theta}^{n-1})}$$

Convergence de l'EM

Mauvaises nouvelles

- ☹ EM converge vers un optimum local
- ☹ Dépend très fortement des conditions initiales ($\boldsymbol{\theta}^0$)
- ☹ Beaucoup de *minima locaux* !
- ☹ La convergence est très (trop) rapide

Bonnes nouvelles

- ☺ Les minima locaux sont souvent « raisonnables »
- ☹ Peu de données étiquetées suffisent à fournir des bons paramètres initiaux (apprentissage semi-supervisé)
- ☹ nombreuses variantes (cf. cours précédent)
  - en ligne (mise à jour après chaque observation)
  - sommer sur quelques séquences d'états
  - ralentir la convergence (tempered EM)

Convergence de l'EM

Mauvaises nouvelles

- ☹ EM converge vers un optimum local
- ☹ Dépend très fortement des conditions initiales ($\boldsymbol{\theta}^0$)
- ☹ Beaucoup de *minima locaux* !
- ☹ La convergence est très (trop) rapide

Bonnes nouvelles

- ☺ Les minima locaux sont souvent « raisonnables »
- ☹ Peu de données étiquetées suffisent à fournir des bons paramètres initiaux (apprentissage semi-supervisé)
- ☹ nombreuses variantes (cf. cours précédent)
  - en ligne (mise à jour après chaque observation)
  - sommer sur quelques séquences d'états
  - ralentir la convergence (tempered EM)

Sommaire

1 Modèles structurés

2 HMM, une révision rapide

3 Modèles pour l'alignement de mots
- Les alignements mot-à-mot

4 Modèles pour la syntaxe

Traduction automatique

Le triomphe des modèles statistiques

Principe général : du français ($\mathbf{f}$) vers l'anglais ($\mathbf{e}$)

1. construire $P(\mathbf{f}|\mathbf{e};\theta)$ et $P(\mathbf{e}|\theta')$
2. $P(\mathbf{f}|\mathbf{e})$ est appris sur des textes parallèles alignés
3. inférer $\text{argmax}_{\mathbf{e}} P(\mathbf{f}|\mathbf{e})\,P(\mathbf{e})$ pour traduire $\mathbf{e}$ (canal bruité)

Le pétrole : des textes parallèles

In the gayest and happiest spirits she set forward with her father ;	Elle partit avec son père, le visage souriant ;
not always listening, but always agreeing to what he said ;	elle n' écoutait pas toujours, mais elle acquiesçait de confiance.
They arrived .	Ils arrivèrent .
It is Frank and Miss Fairfax, said Mrs. Weston .	– C'est Frank et Mlle Fairfax, dit aussitôt Mme Weston .
I was just going to tell you of our agreeable surprize in seeing him arrive this morning.	– J'allai justement vous faire part de l'agréable surprise que nous avons eue en le voyant arriver.
He stays till tomorrow, and Miss Fairfax has been persuaded to spend the day with us .	Il reste jusqu'à demain et Mlle Fairfax a bien voulu, sur notre demande , venir passer la journée.

Traduction automatique

Le triomphe des modèles statistiques

Principe général : du français ($\mathbf{f}$) vers l'anglais ($\mathbf{e}$)

1. construire $\mathrm{P}(\mathbf{f}\,|\,\mathbf{e};\boldsymbol{\theta})$ et $\mathrm{P}(\mathbf{e}\,|\,\boldsymbol{\theta}')$
2. $\mathrm{P}(\mathbf{f}\,|\,\mathbf{e})$ est appris sur des textes parallèles alignés
3. inférer $\mathrm{argmax}_{\mathbf{e}}\,\mathrm{P}(\mathbf{f}\,|\,\mathbf{e})\,\mathrm{P}(\mathbf{e})$ pour traduire $\mathbf{e}$ (canal bruité)

Le pétrole : des textes parallèles

In the gayest and happiest spirits she set forward with her father ;	Elle partit avec son père, le visage souriant ;
not always listening, but always agreeing to what he said ;	elle n' écoutait pas toujours, mais elle acquiesçait de confiance.
They arrived .	Ils arrivèrent .
It is Frank and Miss Fairfax, said Mrs. Weston .	– C'est Frank et Mlle Fairfax, dit aussitôt Mme Weston .
I was just going to tell you of our agreeable surprize in seeing him arrive this morning.	– J'allai justement vous faire part de l'agréable surprise que nous avons eue en le voyant arriver.
He stays till tomorrow, and Miss Fairfax has been persuaded to spend the day with us .	Il reste jusqu'à demain et Mlle Fairfax a bien voulu, sur notre demande , venir passer la journée.

Introduction des alignments

Décomposer $\mathrm{P}(\mathbf{f}\,|\,\mathbf{e};\theta)$

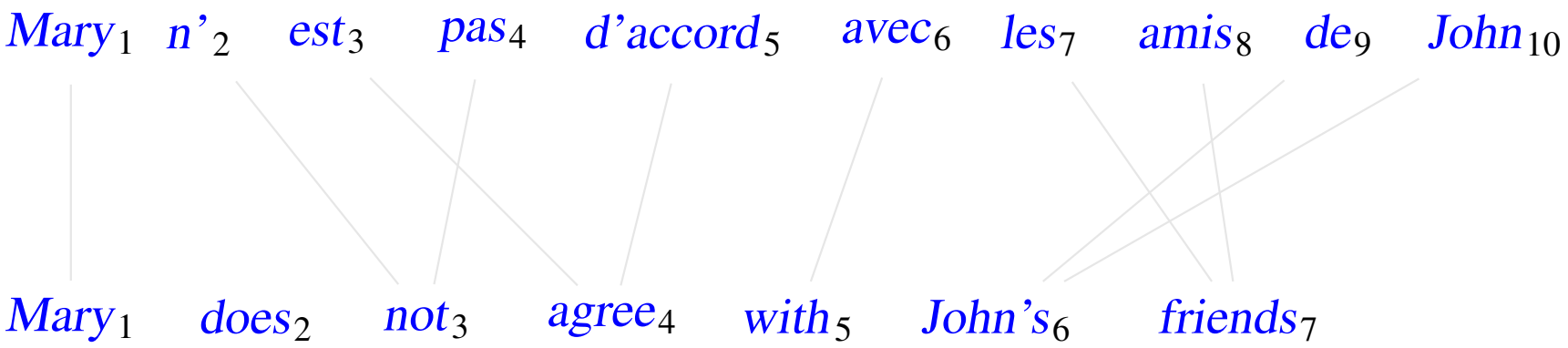
- la décomposition $\mathrm{P}(\mathbf{f}\,|\,\mathbf{e}) = \prod_i \mathrm{P}(f_i\,|\,e_i)$ est trop simpliste
- $\Rightarrow$ décomposition via des alignements latents :

$$\mathrm{P}(\mathbf{f}\,|\,\mathbf{e}) \quad = \quad \sum_{\mathbf{a}\in\mathcal{A}} \mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e})$$

où $\mathcal{A}$ est l'ensemble des alignements de mots entre $\mathbf{e}$ et $\mathbf{f}$

Alignements de mots

Expliciter les relations de traduction



Alignements symétriques et asymétriques

- symétrique : un alignement = relation sur $I \times J$.

$$a = \{(1,1),(2,3),(3,4),(4,2),(5,4)\ldots\}$$

$2^{I \times J}$ relations possibles

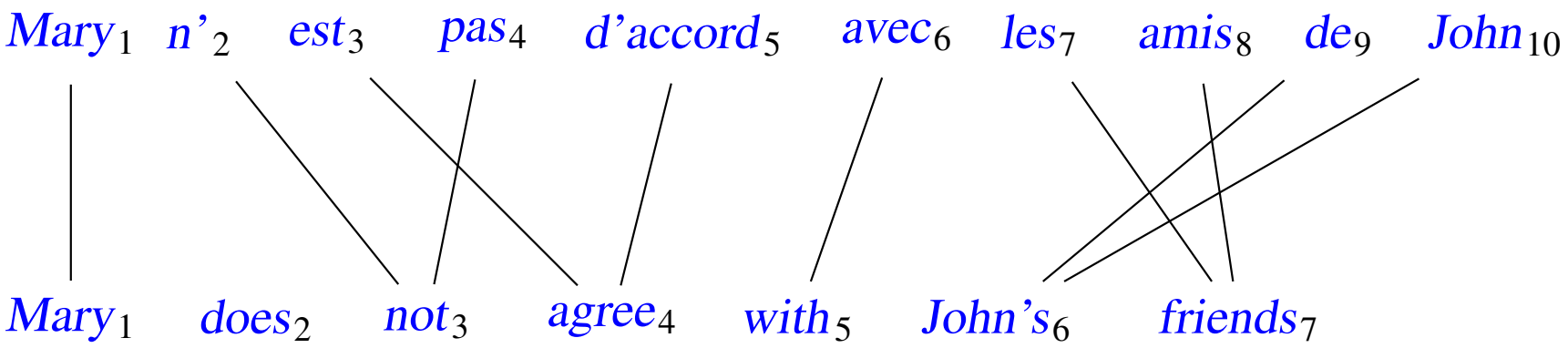
- asymétrique : un alignement = application partielle de J vers I :

$$a = [1,3,4,2,4,5,7,7,6,6]$$

« seulement » I^J applications possibles

Alignements de mots

Expliciter les relations de traduction



Alignements symétriques et asymétriques

- symétrique : un alignement = relation sur $I \times J$.

$$a = \{(1,1),(2,3),(3,4),(4,2),(5,4)\ldots\}$$

$2^{I \times J}$ relations possibles

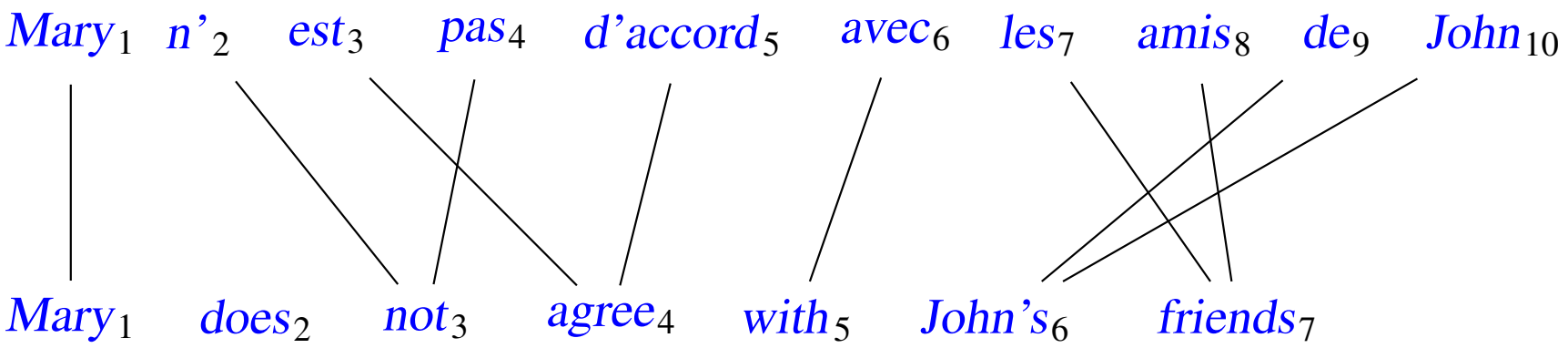
- asymétrique : un alignement = application partielle de J vers I :

$$a = [1,3,4,2,4,5,7,7,6,6]$$

« seulement » I^J applications possibles

Alignements de mots

Expliciter les relations de traduction



Alignements symétriques et asymétriques

- symétrique : un alignement = relation sur $I \times J$.

$$a = \{(1,1),(2,3),(3,4),(4,2),(5,4)\ldots\}$$

$2^{I \times J}$ relations possibles

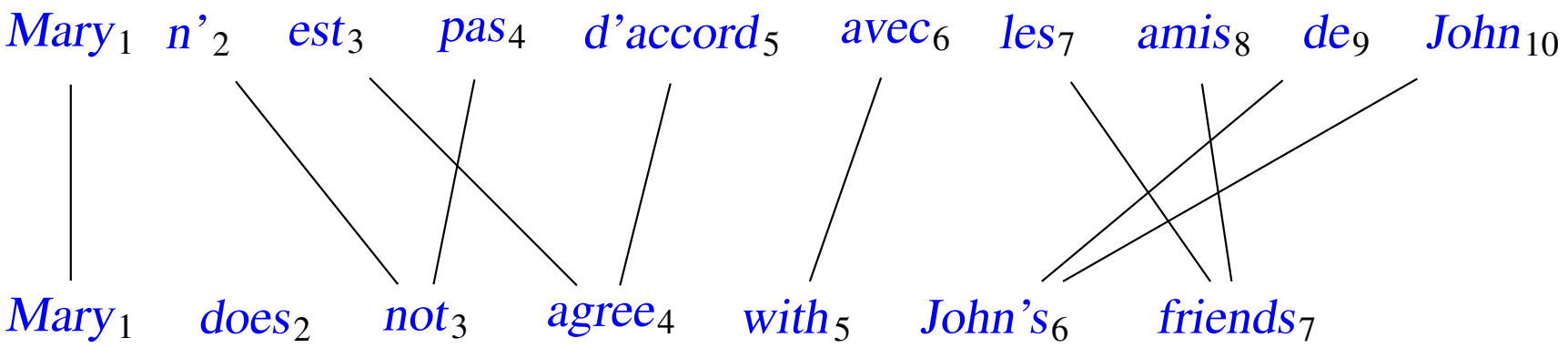
- asymétrique : un alignement = application partielle de J vers I :

$$a = [1,3,4,2,4,5,7,7,6,6]$$

« seulement » I^J applications possibles

Alignements de mots

Expliciter les relations de traduction



Alignements symétriques et asymétriques

- symétrique : un alignement = relation sur $I \times J$.

$$a = \{(1,1),(2,3),(3,4),(4,2),(5,4)\ldots\}$$

$2^{I \times J}$ relations possibles

- asymétrique : un alignement = application partielle de J vers I :

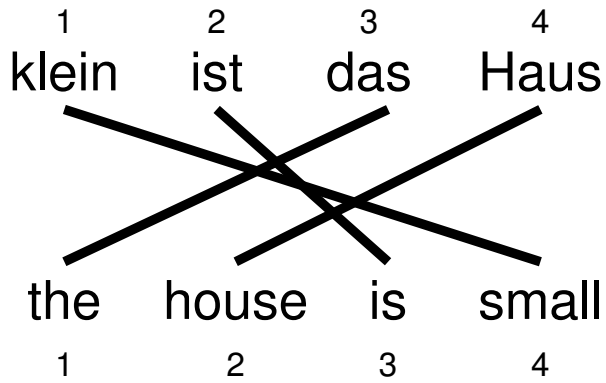
$$a = [1,3,4,2,4,5,7,7,6,6]$$

« seulement » I^J applications possibles

Calculer des alignements de mots

Quelques configurations typiques

Changements de l'ordre



Un mot pour plusieurs

Insertions

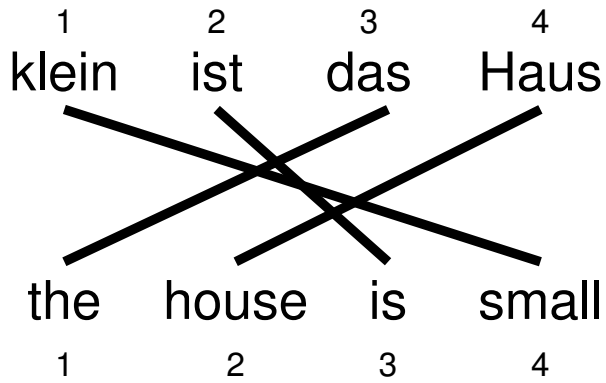
Des mots non-alignés

Limsi

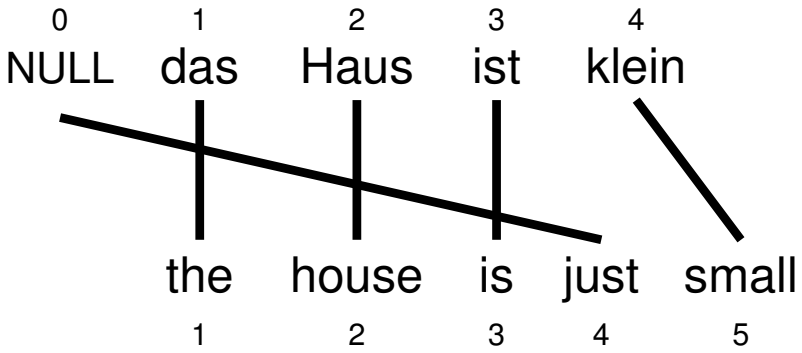
Calculer des alignements de mots

Quelques configurations typiques

Changements de l'ordre



Insertions



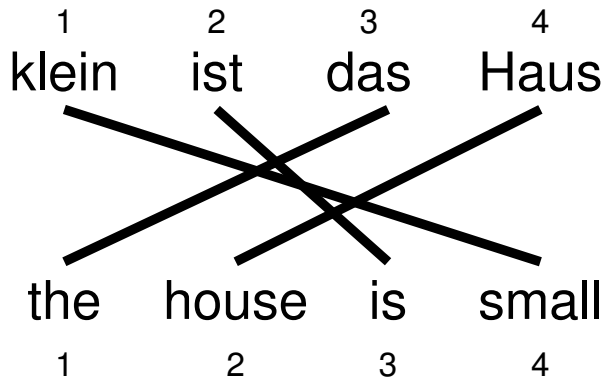
Un mot pour plusieurs

Des mots non-alignés

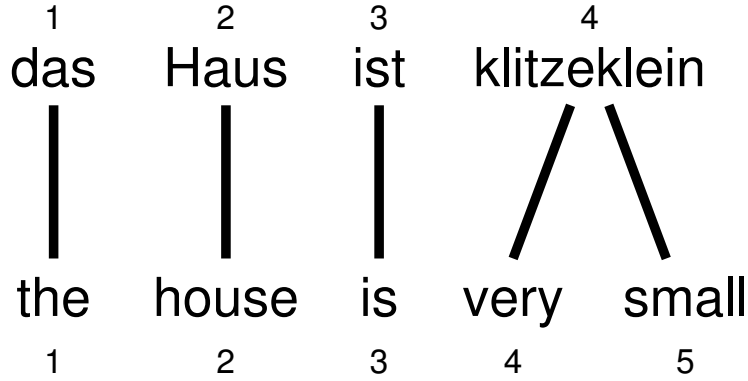
Calculer des alignements de mots

Quelques configurations typiques

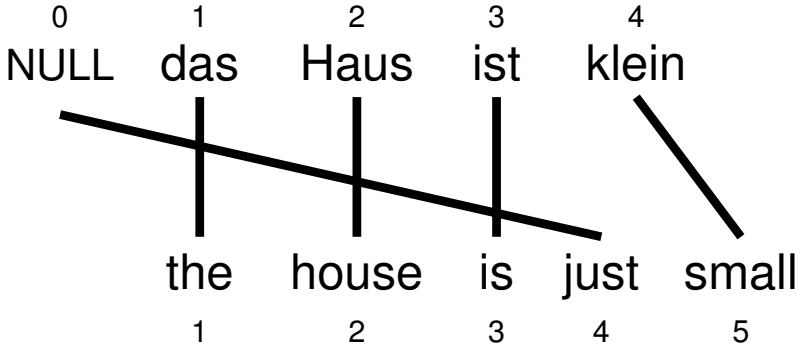
Changements de l'ordre



Un mot pour plusieurs



Insertions

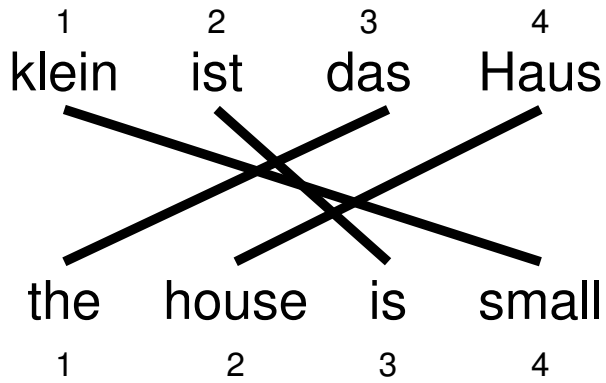


Des mots non-alignés

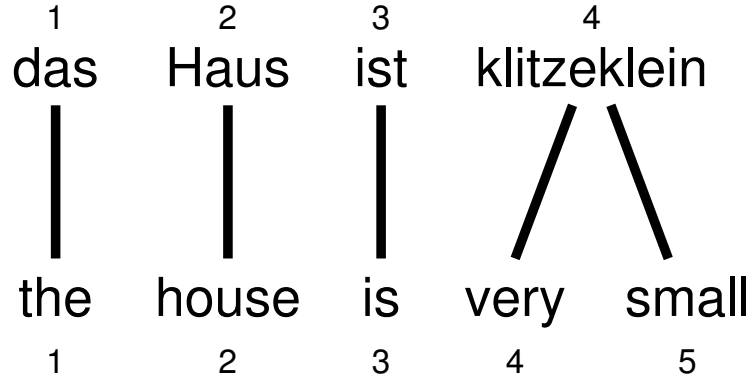
Calculer des alignements de mots

Quelques configurations typiques

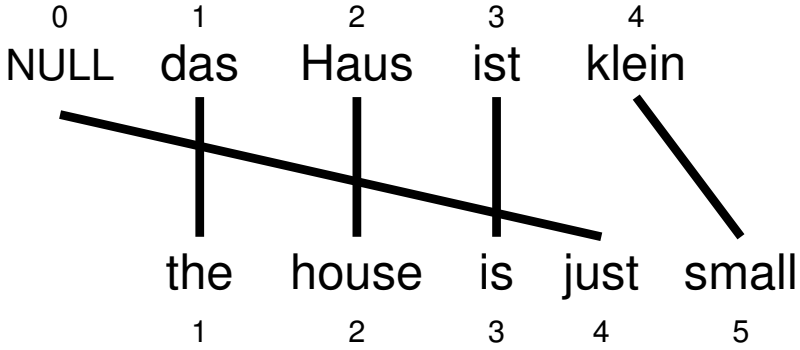
Changements de l'ordre



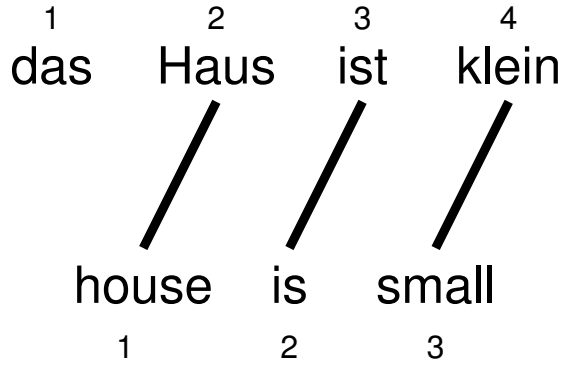
Un mot pour plusieurs



Insertions



Des mots non-alignés



Calculer des alignements de mots

Les hypothèses de l'alignement de mots

☺ certains mots sont faciles à aligner ?

☹ la plupart des alignements sont 1 to 1 ?

☹☹ monotonicité

Calculer des alignements de mots

Les hypothèses de l'alignement de mots

☺ certains mots sont faciles à aligner ?

☹ la plupart des alignements sont 1 to 1 ?

☹☹ monotonicité

Un modèle générique

Des alignements asymétriques

Notations

- $\mathbf{f} = f_{[1:J]}$ la phrase source (J mots)
- $\mathbf{e} = e_{[1:J]}$ la phrase cible ($I = l_{\mathbf{e}}$ mots)
- problème : décomposer $\mathrm{P}(\mathbf{a}, \mathbf{f} \,|\, \mathbf{e})$

Structure du modèle graphique (e connu)

1. choisir J sachant e_1^I
2. pour chaque position $j \in [1 : J]$
   1. choisir a_j sachant $J, a_{[1:j-1]}, f_{[1:j-1]}, e_{[1:I]}$
   2. choisir f_j sachant $J, a_{[1:j]}, f_{[1:j-1]}, e_{[1:I]}$

$$\mathrm{P}(a_{[1:J]}, f_{[1:J]} \,|\, e_{[1:I]}) = \mathrm{P}(J \,|\, e_{[1:I]}) \prod_j \mathrm{P}(a_j \,|\, a_{[1:j-1]}, f_{[1:j-1]}, e_{[1:I]})\, \mathrm{P}(f_j \,|\, a_{[1:j]}, f_{[1:j-1]}, e_{[1:I]})$$

Un modèle générique

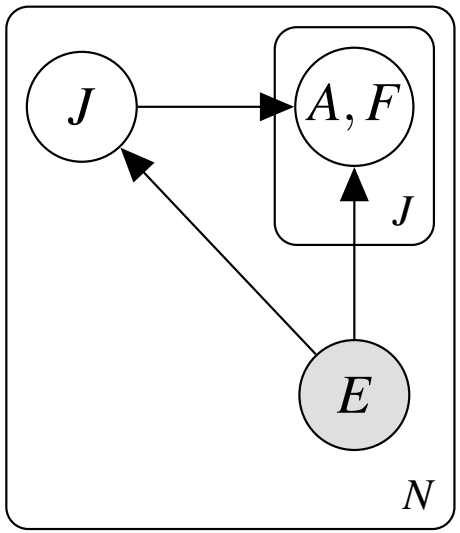
Des alignements asymétriques

Notations

- $\mathbf{f} = f_{[1:J]}$ la phrase source (J mots)
- $\mathbf{e} = e_{[1:J]}$ la phrase cible ($I = l_{\mathbf{e}}$ mots)
- problème : décomposer $\mathrm{P}(\mathbf{a}, \mathbf{f} \,|\, \mathbf{e})$

Structure du modèle graphique ($\mathbf{e}$ connu)

1. choisir J sachant e_1^I
2. pour chaque position $j \in [1 : J]$
   1. choisir a_j sachant $J, a_{[1:j-1]}, f_{[1:j-1]}, e_{[1:I]}$
   2. choisir f_j sachant $J, a_{[1:j]}, f_{[1:j-1]}, e_{[1:I]}$



$$\mathrm{P}(a_{[1:J]}, f_{[1:J]} \,|\, e_{[1:I]}) = \mathrm{P}(J \,|\, e_{[1:I]}) \prod_j \mathrm{P}(a_j \,|\, a_{[1:j-1]}, f_{[1:j-1]}, e_{[1:I]})\, \mathrm{P}(f_j \,|\, a_{[1:j]}, f_{[1:j-1]}, e_{[1:I]})$$

Modéliser les alignements : avec des HMMs

Hypothèses

- J ne dépend que de I
- a_j ne dépend que de a_{j-1} (dépendances markoviennes) et de $\mathbf{e}$
- f_j ne dépend que de e_{a_j} (le mot aligné avec f_j)

Histoire générative de $\mathbf{f}$ sachant $\mathbf{e}$

- choisir J sachant I
- pour chaque position $j \in [1:J]$
  - choisir $a_j|I, a_{j-1}, \mathbf{e} \sim \text{Disc}(u(a|a_{j-1}))$: l'indice associé à j
  - choisir $f_j|J, a_j, e_{[1:I]} \sim \text{Disc}(t(f|e_{a_j}))$: f_j dépend que du mot avec lequel il est aligné

$$\mathrm{P}(\mathbf{a}, \mathbf{f} \,|\, \mathbf{e}) = \mathrm{P}(J \,|\, I) \prod_j u(a_j|a_{j-1}, e_{[1:I]}) t(f_j|e_{a_j})$$

Paramètres : $\theta = \{u() : J^2_{\max} \text{ valeurs}; t() : n_{WE} \times n_{WF} \text{ valeurs}(!)\}$

Modéliser les alignements : avec des HMMs

Hypothèses

- J ne dépend que de I
- a_j ne dépend que de a_{j-1} (dépendances markoviennes) et de $\mathbf{e}$
- f_j ne dépend que de e_{a_j} (le mot aligné avec f_j)

Histoire générative de **f** sachant **e**

- choisir J sachant I
- pour chaque position $j \in [1 : J]$
  - choisir $a_j | I, a_{j-1}, \mathbf{e} \sim \text{Disc}(u(a|a_{j-1}))$: l'indice associé à j
  - choisir $f_j | J, a_j, e_{[1:I]} \sim \text{Disc}(t(f|e_{a_j}))$: f_j dépend que du mot avec lequel il est aligné

$$\mathrm{P}(\mathbf{a}, \mathbf{f} \,|\, \mathbf{e}) = \mathrm{P}(J \,|\, I) \prod_j u(a_j|a_{j-1}, e_{[1:I]}) t(f_j|e_{a_j})$$

Paramètres : $\boldsymbol{\theta} = \{u() : J^2_{\max} \text{ valeurs}; t() : n_{WE} \times n_{WF} \text{ valeurs}(!)\}$

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Engendrer des alignements avec un HMM

Cible : **e**

this$_1$ *seems*$_2$ *to*$_3$ *me*$_4$ *to*$_5$ *be*$_6$ *a*$_7$ *workable*$_8$ *solution*$_9$

$J \sim P(J\|9) \Rightarrow J = 9$		
1	$a_1 \sim u(a_1) : a_1 = 1$	$f_1 \sim t(f\|this) : f_1 = il$
2	$a_2 \sim u(a_2\|1) : a_2 = 2$	$f_2 \sim t(f\|seems) : f_2 = me$
3	$a_3 \sim u(a_3\|2) : a_3 = 2$	$f_3 \sim t(f\|seems) : f_3 = semble$
4	$a_4 \sim u(a_4\|2) : a_4 = 3$	$f_4 \sim t(f\|to) : f_4 = que$
5	$a_5 \sim u(a_5\|3) : a_5 = 5$	$f_5 \sim t(f\|to) : f_5 = c'$
6	$a_6 \sim u(a_6\|5) : a_6 = 6$	$f_6 \sim t(f\|be) : f_6 = est$
7	$a_7 \sim u(a_7\|6) : a_7 = 7$	$f_7 \sim t(f\|a) : f_7 = une$
8	$a_8 \sim u(a_8\|7) : a_8 = 8$	$f_8 \sim t(f\|workable) : f_8 = bonne$
9	$a_9 \sim u(a_9\|8) : a_9 = 9$	$f_9 \sim t(f\|solution) : f_9 = solution$

Source : **f**

il$_1$ *me*$_2$ *semble*$_3$ *que*$_4$ *c'*$_5$ *est*$_6$ *une*$_7$ *bonne*$_8$ *solution*$_9$

Limsi

Deux finesses

Les mots « non-traduits »

Traiter des mots source non alignables : Les et de dans :
Les étudiants sont tenus de s'inscrire / Students must register

- état fictif (ϵ) dans la cible (d'indice 0) atteint avec $P_0 = \mathrm{P}(a_i = 0 \,|\, a_{i-1}, J)$
- une distribution associée à cet état $P = \mathrm{P}(f \,|\, \epsilon)$

Modéliser les sauts

Rendre le modèle d'alignement indépendant des indices absolus :
$\Rightarrow$ remplacer $\mathrm{P}(a_i \,|\, a_{i-1})$ par $\mathrm{P}(a_i - a_{i-1} \,|\, a_{i-1} - a_{i-2})$

Deux finesses

Les mots « non-traduits »

Traiter des mots source non alignables : Les et de dans :
Les étudiants sont tenus de s'inscrire / Students must register

- état fictif (ϵ) dans la cible (d'indice 0) atteint avec $P_0 = \mathrm{P}(a_i = 0 \,|\, a_{i-1}, J)$
- une distribution associée à cet état $P = \mathrm{P}(f \,|\, \epsilon)$

Modéliser les sauts

Rendre le modèle d'alignement indépendant des indices absolus :
$\Rightarrow$ remplacer $\mathrm{P}(a_i \,|\, a_{i-1})$ par $\mathrm{P}(a_i - a_{i-1} \,|\, a_{i-1} - a_{i-2})$

Estimation supervisée du modèle

- à alignements connus...
- ... les paramètres se déduisent par décompte

$$\forall I \in [1 \ldots I_{max}], J \in [1 \ldots J_{max}], \mathrm{P}(J \mid I) = \frac{C(I,J)}{C(I)}$$

$$\forall i,j \in [1 \ldots I_{max}], u(i|j,J,I) = \frac{C(i,j,I,J)}{C(j,I,J)}$$

$$\forall e \in V_e, f \in V_f, t(f|e) = \frac{C(e,f)}{C(e)}$$

Calculer les alignements (à modèle connu)

Problème

- $P(.|I)$ connu ; $u(.|j,I,J)$ connu ; $t(.|e)$ connu
- $\mathbf{e} = e_1^I$ et $\mathbf{f} = f_1^J$ sont toujours observés
- trouver :

$$\begin{aligned} a^* &= \underset{a_1 \ldots a_J}{\operatorname{argmax}} \, \mathrm{P}(f_1^J, a_1^J \,|\, e_j) \\ &= \underset{a_1 \ldots a_J}{\operatorname{argmax}} \, \mathrm{P}(J \,|\, I) \prod_j u(a_j|a_{j-1}) t(f_i|e_{a_j}) \end{aligned}$$

Résolution par programmation dynamique : Viterbi

$\delta(i,j)$ = proba du meilleur alignement de f_1^j avec $\mathbf{e}$ tq. $a_j = i$

$$\begin{cases} \delta(i,1) = u(a_1 = i|a_0) t(f_1|e_i), \forall i \in [0 \ldots I] \\ \delta(i,j) = max_{i' \in [0:I]} \delta(i', j-1) u(a_j = i|a_{j-1} = i') t(f_j|e_i) \forall i, j > 1 \end{cases}$$

Estimation par EM

Étape E(xpectation)

à paramètres connus (étape précédente) :

$$\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f};\boldsymbol{\theta}) = \frac{\mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e};\boldsymbol{\theta})}{\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e};\boldsymbol{\theta})} \qquad \text{(calcul des } \alpha \text{ et } \beta\text{)}$$

Étape M(aximisation de la fonction auxiliaire)

Espérances des comptes (pour une phrase) :

$$\forall I \in [1\ldots I_{max}], J \in [1\ldots J_{max}], \mathrm{P}(J\,|\,I) = \frac{C(I,J)}{C(I)}$$

$$\forall i, i' \in [1\ldots I], u(i'|i,J,I) = \frac{\sum_{\mathbf{e},\mathbf{f}}\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(i,i')}{\sum_{i}\sum_{\mathbf{e},\mathbf{f}}\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(i,i')}$$

$$\forall e,f, t(f|e) = \frac{\sum_{\mathbf{e},\mathbf{f}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(e,f)}{\sum_{f}\sum_{\mathbf{e},\mathbf{f}}\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(e,f)}$$

Estimation par EM

Étape E(xpectation)

à paramètres connus (étape précédente) :

$$\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f};\boldsymbol{\theta}) = \frac{\mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e};\boldsymbol{\theta})}{\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e};\boldsymbol{\theta})} \qquad \text{(calcul des } \alpha \text{ et } \beta\text{)}$$

Étape M(aximisation de la fonction auxiliaire)

Espérances des comptes (pour une phrase) :

$$\forall I \in [1\ldots I_{max}], J \in [1\ldots J_{max}], \mathrm{P}(J\,|\,I) = \frac{C(I,J)}{C(I)}$$

$$\forall i,i' \in [1\ldots I], u(i'|i,J,I) = \frac{\sum_{\mathbf{e},\mathbf{f}}\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(i,i')}{\sum_{i}\sum_{\mathbf{e},\mathbf{f}}\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(i,i')}$$

$$\forall e,f, t(f|e) = \frac{\sum_{\mathbf{e},\mathbf{f}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(e,f)}{\sum_{f}\sum_{\mathbf{e},\mathbf{f}}\sum_{\mathbf{a}}\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f})C(e,f)}$$

Initialiser avec des modèles simples : Modèles IBM 1 et 2

Problème d'optimisation

EM converge vers un optima local $\Rightarrow$ initialiser $t(\,)$ (ou $\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f};\boldsymbol{\theta})$! !) avec des modèles + simples.

IBM Modèle 1

Les probabilités des a_j sont uniformes : $u(a_j|a_{j-1},I,J) = \frac{1}{I+1}$

$$\mathrm{P}(a_1^J, f_1^J \,|\, e_1^I) \quad = \quad \frac{\mathrm{P}(J\,|\,I)}{(I+1)^J} \prod_j t(f_j|e_{a_j})$$

IBM Modèle 2

Les a_j ne dépendent que de j : $\mathrm{P}(a_j\,|\,a_{j-1},I,J) = \mathrm{P}(a_j\,|\,j,I,J)$

$$\mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e}) \quad = \quad \mathrm{P}(J\,|\,I) \prod_j u(a_j|j,I,J)\, t(f_j|e_{a_j})$$

Initialiser avec des modèles simples : Modèles IBM 1 et 2

Problème d'optimisation

EM converge vers un optima local $\Rightarrow$ initialiser $t()$ (ou $\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f};\boldsymbol{\theta})$! !) avec des modèles + simples.

IBM Modèle 1

Les probabilités des a_j sont uniformes : $u(a_j|a_{j-1},I,J) = \frac{1}{I+1}$

$$\mathrm{P}(a_1^J, f_1^J \,|\, e_1^I) \quad = \quad \frac{\mathrm{P}(J\,|\,I)}{(I+1)^J} \prod_j t(f_j|e_{a_j})$$

IBM Modèle 2

Les a_j ne dépendent que de j : $\mathrm{P}(a_j\,|\,a_{j-1},I,J) = \mathrm{P}(a_j\,|\,j,I,J)$

$$\mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e}) \quad = \quad \mathrm{P}(J\,|\,I) \prod_j u(a_j|j,I,J)t(f_j|e_{a_j})$$

IBM Modèle 1

Hypothèses

- J est uniforme sur $[1; N]$, N grand : $P(J) = \epsilon$
- a_j est uniforme sur $[0; I]$
- f_j ne dépend que de e_{a_j} (le mot aligné avec f_j)

Histoire générative de **f** sachant **e**

- choisir J
- pour chaque position $j \in [1 : J]$
  - choisir $a_j \,|\, J \sim \text{Unif}([0 : I])$: l'indice du mot qui engendre f_j
  - choisir $f_j \,|\, J, e_{a_j} \sim \text{Disc}(t(f_j|e_{a_j}))$: sachant le mot cible, choisir le mot source

$$\mathrm{P}(\mathbf{a}, \mathbf{f} \,|\, \mathbf{e}) = \epsilon \prod_{j=1}^{J} \frac{1}{I+1} t(f_j|e_{a_j}) = \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j})$$

IBM Modèle 1

Hypothèses

- J est uniforme sur $[1; N]$, N grand : $P(J) = \epsilon$
- a_j est uniforme sur $[0; I]$
- f_j ne dépend que de e_{a_j} (le mot aligné avec f_j)

Histoire générative de **f** sachant **e**

- choisir J
- pour chaque position $j \in [1 : J]$
  - choisir $a_j \,|\, J \sim \text{Unif}([0 : I])$: l'indice du mot qui engendre f_j
  - choisir $f_j \,|\, J, e_{a_j} \sim \text{Disc}(t(f_j|e_{a_j}))$: sachant le mot cible, choisir le mot source

$$\mathrm{P}(\mathbf{a}, \mathbf{f} \,|\, \mathbf{e}) = \epsilon \prod_{j=1}^{J} \frac{1}{I+1} t(f_j|e_{a_j}) = \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j})$$

Simplicité d'IBM1

Expression analytique de la vraisemblance

Calcul de la vraisemblance

$$\mathrm{P}(\mathbf{f}|\mathbf{e}) = \frac{\epsilon}{(I+1)^J} \sum_{\mathbf{a}\in\mathcal{A}} \mathrm{P}(\mathbf{a},\mathbf{f}|\mathbf{e},J;\boldsymbol{\theta})$$

$$= \frac{\epsilon}{(I+1)^J} \sum_{a_1=0}^{I} \sum_{a_2=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} t(f_j|e_{a_j})$$

$$= \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} \Big(\sum_{i=0}^{I} t(f_j|e_i) \Big)$$

$$\mathrm{P}(\mathbf{a}|\mathbf{e},\mathbf{f}) = \prod_{j=1}^{J} \Big(\frac{t(f_j|e_{a_j})}{\sum_{i=0}^{I} t(f_j|e_i)} \Big)$$

factorise par positions

Raisonnement graphique :

$$\begin{cases} A_j \perp\!\!\!\perp A_{j'} \,|\, E,\boldsymbol{\theta} \Rightarrow \mathrm{P}(\mathbf{a}|\mathbf{e},\mathbf{f},J) \propto \prod_j \mathrm{P}(A_j|E,F) \\ F_j \perp\!\!\!\perp F_{j'} \,|\, E,\boldsymbol{\theta} \Rightarrow \mathrm{P}(\mathbf{f}|\mathbf{e},J) \propto \prod_j \mathrm{P}(F_j|E) \end{cases}$$

Simplicité d'IBM1

Expression analytique de la vraisemblance

Calcul de la vraisemblance

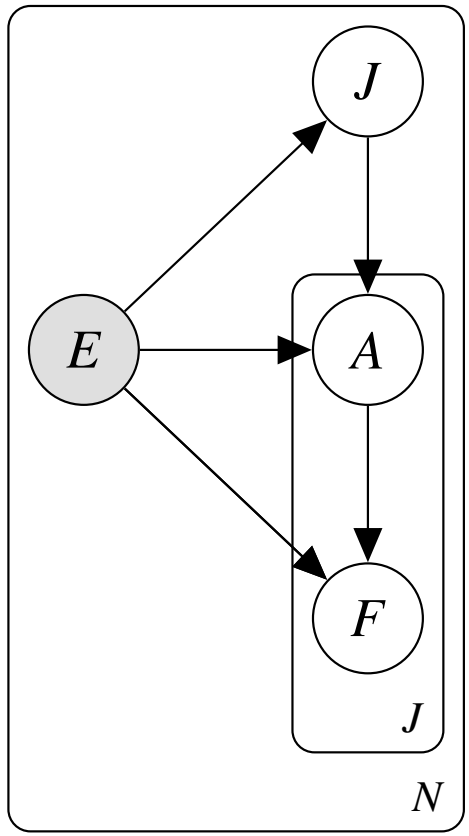
$$\mathrm{P}(\mathbf{f}\,|\,\mathbf{e}) = \frac{\epsilon}{(I+1)^J} \sum_{\mathbf{a}\in\mathcal{A}} \mathrm{P}(\mathbf{a},\mathbf{f}\,|\,\mathbf{e},J;\boldsymbol{\theta})$$

$$= \frac{\epsilon}{(I+1)^J} \sum_{a_1=0}^{I} \sum_{a_2=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} t(f_j|e_{a_j})$$

$$= \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} \Big(\sum_{i=0}^{I} t(f_j|e_i) \Big)$$

$$\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f}) = \prod_{j=1}^{J} \Big(\frac{t(f_j|e_{a_j})}{\sum_{i=0}^{I} t(f_j|e_i)} \Big)$$

factorise par positions



Raisonnement graphique :

$$\begin{cases} A_j \perp\!\!\!\perp A_{j'} \,|\, E,\boldsymbol{\theta} \Rightarrow \mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f},J) \propto \prod_j \mathrm{P}(A_j\,|\,E,F) \\ F_j \perp\!\!\!\perp F_{j'} \,|\, E,\boldsymbol{\theta} \Rightarrow \mathrm{P}(\mathbf{f}\,|\,\mathbf{e},J) \propto \prod_j \mathrm{P}(F_j\,|\,E) \end{cases}$$

Simplicité d'IBM1

La révélation

$$\mathrm{P}(\mathbf{a}\,|\,\mathbf{e},\mathbf{f}) = \prod_{j=1}^{J} \left(\frac{t(f_j|e_{a_j})}{\sum_{i=0}^{I} t(f_j|e_i)} \right)$$

Conséquences

- La probabilité a posteriori d'un alignement est un produit sur les liens
- Chaque lien $a_j : f_j \to e_{a_j}$ a posteriori : $\mathrm{P}(a_j\,|f_j,\mathbf{e}) = t(f_j|e_{a_j})$ (normalisé sur $\mathbf{e}$)
- Tous les calculs (yc l'inférence) se décomposent par position :

$$\mathbf{a}^* = a^*_{[1:J]} \text{ avec } j^* = \operatorname*{argmax}_i t(f_j|e_i)$$

Simplicité perdue dans le modèle HMM : il faut faire forward / backward

Simplicité d'IBM1

Expectation - Maximisation : décomposé par lien

- Fonction auxiliaire de l'EM (concave - mais pas strictement concave : il peut exister plusieurs optimum globaux)

$$Q_{\boldsymbol{\theta}'}(\boldsymbol{\theta}) = \sum_{(\mathbf{f},\mathbf{e})} \sum_{\mathbf{a}} \mathrm{P}(\mathbf{a} \mid \mathbf{e}, \mathbf{f}; \boldsymbol{\theta}') \log\left(\prod_{j=1}^{J} t(f_j|e_{a_j})\right)$$

$$vtheta^* = \operatorname*{argmax}_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}'}(\boldsymbol{\theta}) \text{ avec } \forall e, \sum_f t(f|e) = 1$$

- Maximiser $L(t,t') = Q_{\boldsymbol{\theta}'}(\boldsymbol{\theta}) + \sum_e \lambda_e (1 - \sum_f t(f|e))$

$$\frac{\delta L}{\delta t(f|e)} = \sum_{(\mathbf{f},\mathbf{e})} \sum_{\mathbf{a}} P(\mathbf{a}|\mathbf{e},\mathbf{f};t') \frac{\sum_{j=1}^{J} \mathbb{I}(f_j = f)\mathbb{I}(e_{a_j} = e)}{t(f|e)} - \lambda_e$$

$$= \sum_{(\mathbf{f},\mathbf{e})} \frac{\sum_{j=1}^{J} \mathbb{I}(f_j = f) \sum_{i=1}^{I} \mathbb{I}(e_i = e)}{t(f|e)} \sum_{\mathbf{a},(f,e)\in\mathbf{a}} P(\mathbf{a}|\mathbf{e},\mathbf{f};t') - \lambda_e$$

$$= \sum_{(\mathbf{f},\mathbf{e})} \frac{\sum_{j=1}^{J} \mathbb{I}(f_j = f) \sum_{i=1}^{I} \mathbb{I}(e_i = e)}{t(f|e)} \frac{t'(f|e)}{\sum_i t'(f|e_i)} - \lambda_e$$

- Conditions d'optimalité

$$t(f|e) \propto \sum_{(\mathbf{e},\mathbf{e})} \left(\sum_{j=1}^{J} \mathbb{I}(f_j = f) \sum_{i=1}^{I} \mathbb{I}(e_i = e)\right) \frac{t'(f|e)}{\sum_i t'(f|e_i)}$$

Simplicité d'IBM1

Estimation

- Initialiser $t(f|e), \forall e, \forall f$
- pour chaque couple de phrases $(\mathbf{e}, \mathbf{f}) \in \mathbb{C}$, calculer :

$$\begin{aligned}\mathbb{E}(C(e,f|\mathbf{e},\mathbf{f},t')) &= \sum_{\mathbf{a}} P(\mathbf{a}|\mathbf{e},\mathbf{e}) C_{\mathbf{a}}(f) C_{\mathbf{a}}(e) \\ &= \frac{t'(f|e)}{\sum_{i=0}^{I} t'(f|e_i)} \sum_j \mathbb{I}(f_j,f) \sum_i \mathbb{I}(e_i,e)\end{aligned}$$

- remettre à jour :

$$t(f|e) = \frac{\sum_{(\mathbf{e},\mathbf{f})} C(e,f|\mathbf{e},\mathbf{f};t')}{\sum_f \sum_S C(e,f|\mathbf{e},\mathbf{f};t')}$$

Inférence position par position

$$\mathbf{a}^* = \operatorname{argmax} \prod_j t(f_j|e_{a_j}) = \prod_j \operatorname*{argmax}_i t(f_j|e_i)$$

IBM modèle 2

Hypothèses

- J est uniforme sur $[1;N]$, N grand : $P(J) = \epsilon$
- a_j dépend de j, de I et de J
- f_j ne dépend que de e_{a_j} (le mot aligné avec f)

Histoire générative de **f** sachant **e**

- choisir J
- pour chaque position $j \in [1:J]$
  - choisir $a_j|j,I,J \sim \text{Disc}(u(a_j|j,I,J))$: choix de l'indice
  - choisir $f_j|J,e_{a_j} \sim \text{Disc}(t(f_j|e_{a_j})$: choix du mot source

$$P(\mathbf{a},\mathbf{f}|\mathbf{e}) = \epsilon \prod_{j=1}^{J} u(a_j|j,I,J)t(f_j|e_{a_j})$$

IBM modèle 2

Hypothèses

- J est uniforme sur $[1; N]$, N grand : $P(J) = \epsilon$
- a_j dépend de j, de I et de J
- f_j ne dépend que de e_{a_j} (le mot aligné avec f)

Histoire générative de **f** sachant **e**

- choisir J
- pour chaque position $j \in [1 : J]$
  - choisir $a_j|j, I, J \sim \text{Disc}(u(a_j|j, I, J))$: choix de l'indice
  - choisir $f_j|J, e_{a_j} \sim \text{Disc}(t(f_j|e_{a_j})$: choix du mot source

$$P(\mathbf{a}, \mathbf{f}|\mathbf{e}) = \epsilon \prod_{j=1}^{J} u(a_j|j, I, J) t(f_j|e_{a_j})$$

Simplicité d'IBM2

Mêmes indépendances que pour IBM modèle 1 (alignement position par position)

Calcul analytique de la vraisemblance

$$
\begin{aligned}
P(\mathbf{f}|\mathbf{e}) &= \epsilon \sum_{\mathbf{a}\in\mathcal{A}} P(\mathbf{f},\mathbf{a}|\mathbf{e}) \\
&= \epsilon \sum_{a_1=0}^{I} \sum_{a_2=0}^{I} \dots \sum_{a_J=0}^{I} \prod_{j=1}^{J} t(f_j|e_{a_j}) u(a_j|j,I,J) \\
&= \epsilon \prod_{j=1}^{J} \left(\sum_{i=0}^{I} t(f_j|e_i) u(i|j,I,J) \right)
\end{aligned}
$$

IBM2 : Estimation des paramètres

- Initialiser $t(f|e), \forall e, \forall f$ (par ex. avec IBM1)
- pour chaque couple de phrases $(\mathbf{e}, \mathbf{f}) \in S$, calculer :

$$\forall e,f : \mathbb{E}(C(e,f|\mathbf{e},\mathbf{f},t')) = \sum_{j=0}^{J}\sum_{i=0}^{I} \frac{t'(f|e)u(i|j,I,J)\mathbb{I}(f,f_j)\mathbb{I}(e,e_j)}{\sum_k t'(f|e_k)u(k|j,I,J)}$$

$$\forall i,j : \mathbb{E}(n(i|j,I,J)) = \frac{t(f_j|e_i)u(i|j,I,J)}{\sum_k t(f|e_k)u(k|j,I,J)}$$

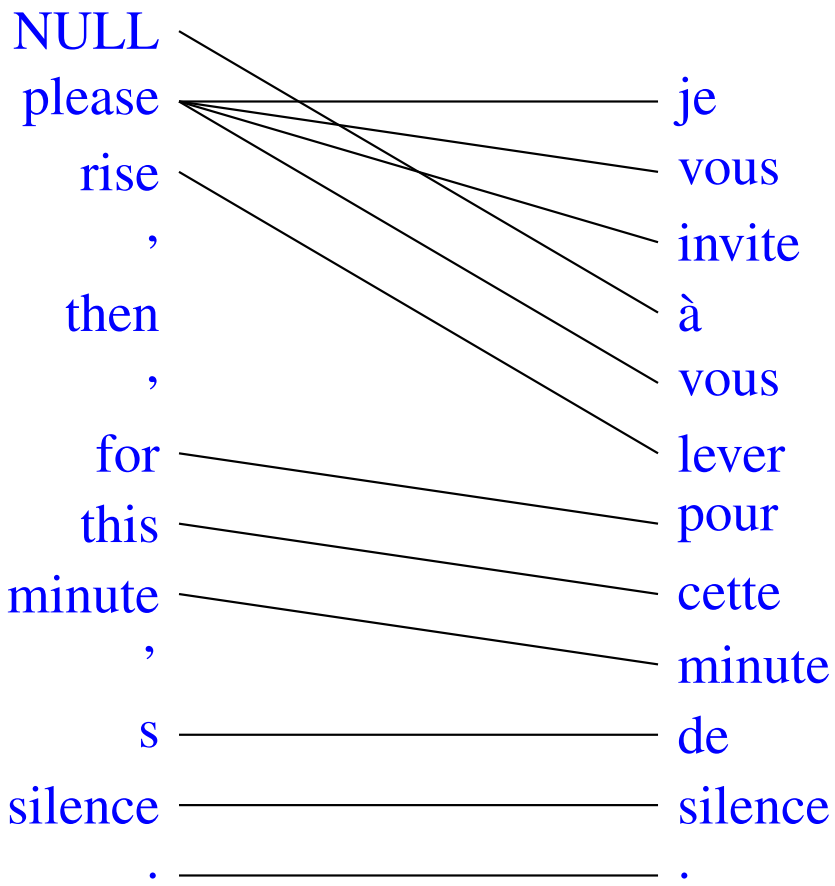
- remettre à jour :

$$t(f|e) = \frac{\sum_{(\mathbf{e},\mathbf{f})} C(e,f|\mathbf{e},\mathbf{f};t')}{\sum_f \sum_{(\mathbf{e},\mathbf{f})} C(e,f|\mathbf{e},\mathbf{f};t')}$$

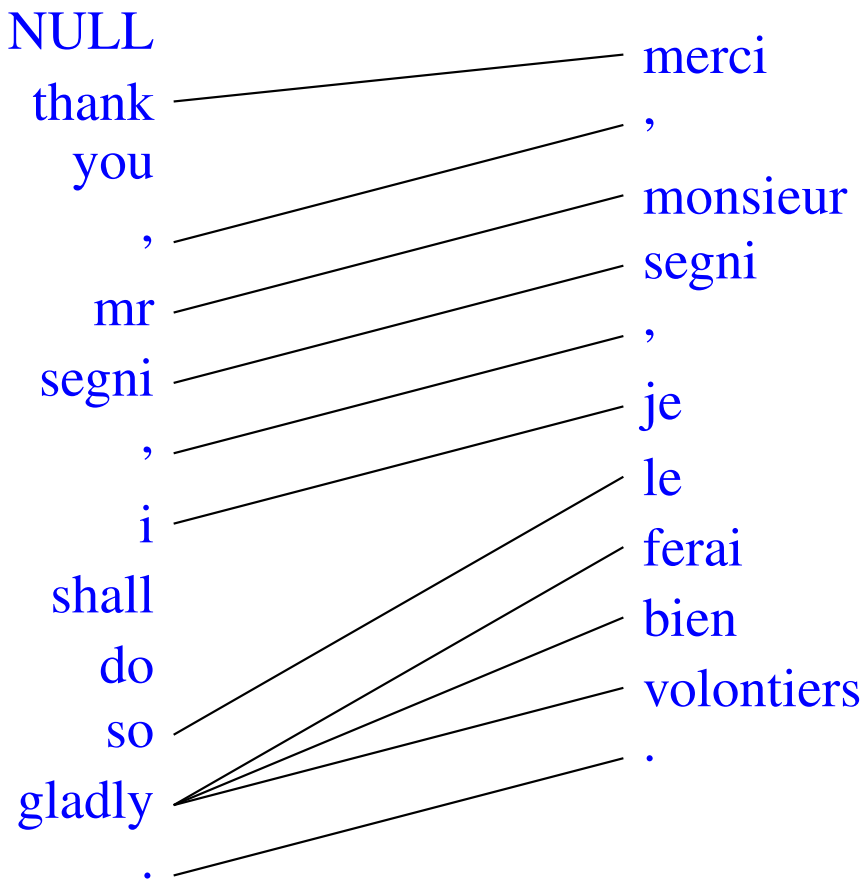
$$u(i|j,I,J) = \frac{\sum_{(\mathbf{e},\mathbf{f})} C(i|j,\mathbf{e},\mathbf{f};t')}{\sum_i \sum_{(\mathbf{e},\mathbf{f})} C(i|j,\mathbf{e},\mathbf{f};t')}$$

Limsi

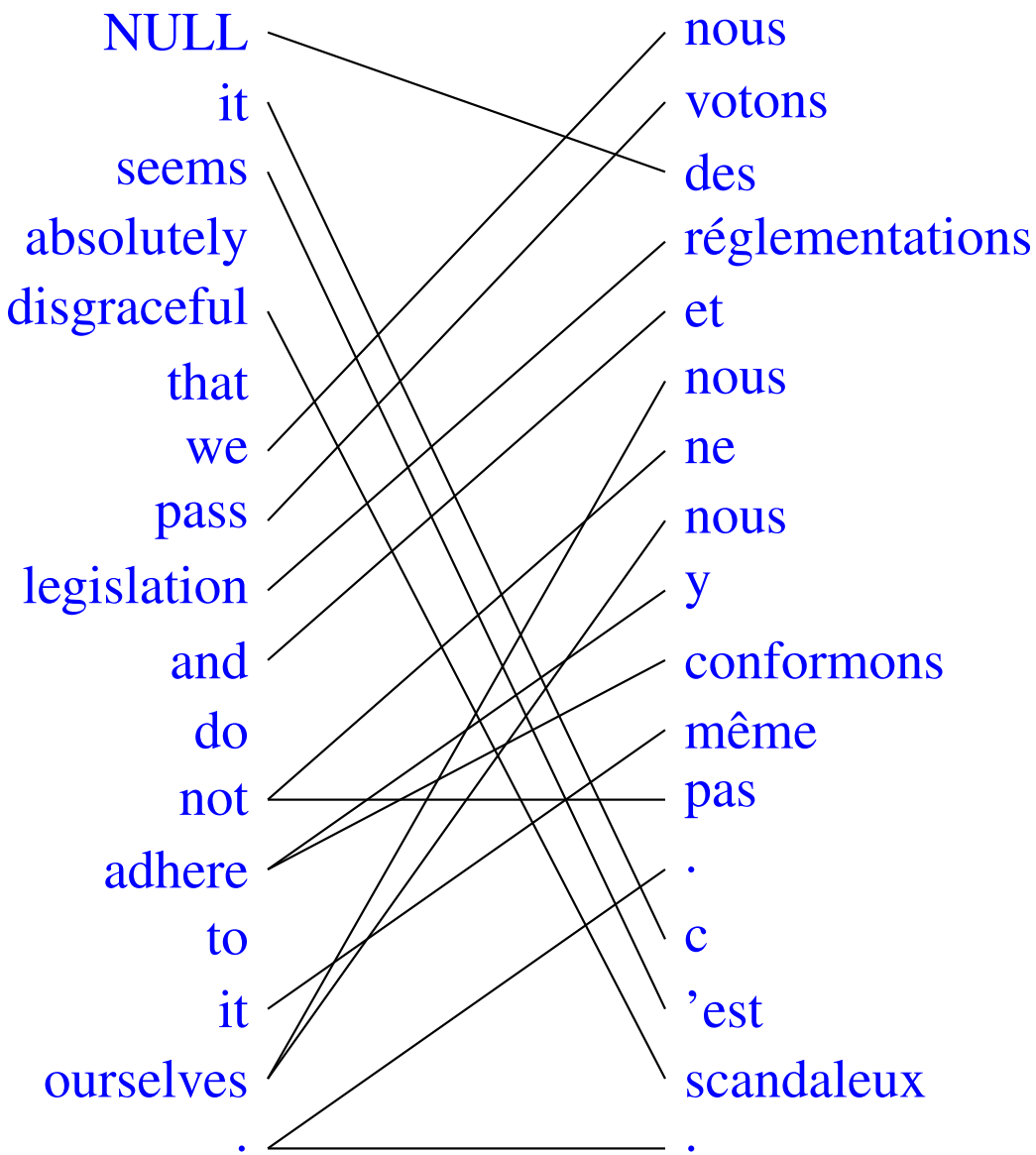
Des alignements... plus ou moins heureux



Des alignements... plus ou moins heureux



Des alignements... plus ou moins heureux



Pour en savoir plus...

- *The mathematics of statistical machine translation* (Brown & al, 1993) : publication de référence sur la traduction mot-à-mot et les modèles d'alignement
- *A Statistical MT tutorial workbook* (Knight, 1999) : le même, en pédagogique
- `Giza++`, `Giza-pp`, `fast_align` : logiciels *open-source* pour la construction d'alignements

Sommaire

1 Modèles structurés

2 HMM, une révision rapide

3 Modèles pour l'alignement de mots

4 Modèles pour la syntaxe
- Engendrer des arbres de dépendances
- Grammaires hors-contexte : les bases
- Grammaires
- Dérivations et arbres de dérivation
- Parsage : résultats élémentaires
- Grammaires probabilistes : Définitions

Limsi

Le domaine de la syntaxe

- Identifier et décrire les énoncés bien formés [corrects] du langage
  - À partir de grammaires : des règles qui supportent une description intensionnelle des énoncés
  - Qui prennent en compte et permettent de calculer des informations sur la structure interne des énoncés
- Articuler les représentations syntaxiques et les autres composants de la grammaire : sémantique, morphologie, morpho-syntaxe
- Déduire des généralisations sur les mécanismes cognitifs impliqués dans la construction et la manipulation de telles représentations : la capacité de langage

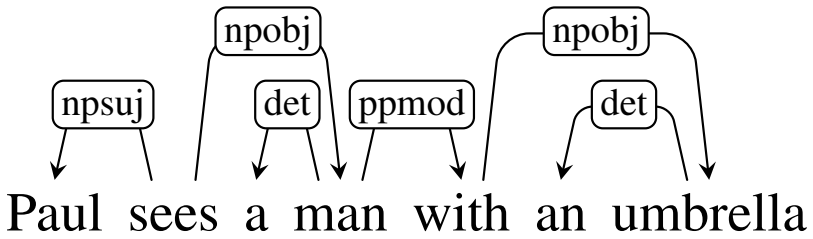
Limsi

L'analyse en dépendances

Un modèle universel de la syntaxe ?

Principe

- représente des relations de dominance / dépendance entre mots/formes
- les relations sont orientées et typées (fonctions syntaxiques)

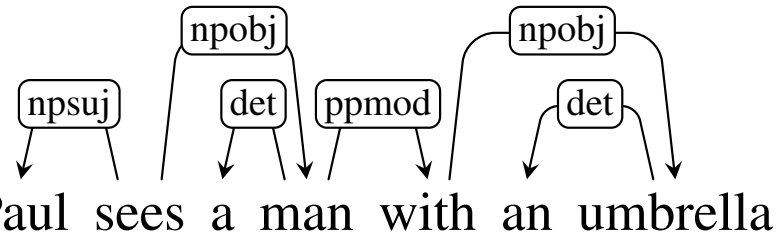


Modèle formel pour les dépendances

Représenter les dépendances

$w_{[1:T]} = w_1 \dots w_T$ une phrase + w_0 (racine)
$G = \{(d, h, t) \in [0:T] \times [0:T] \times \mathcal{R}\}$, tel que :

- $\forall i > 0, !(h, i, t) \in G$ (G est connecté, acyclique)
- $!(0, i, t) \in G$
- (+projectivité) $(h, d, t) \in G$ implique $\forall \min(h, d) < j < \max(h, d)$, j est soit un descendant de h, soit un descendant de d.



$$G = \{(1,2); (2,0); (3,4); (4;2); (5,4); (6,7); (7,5)\}$$

Modèles formels pour les dépendances

Une formulation directe

Les variables du modèle

- $W_{[1:T]}$ pour les mots, $W_i \in [1 : |V|]$
- $H_{[1:T]}$ pour les têtes $H_i \in [0 : T]$
- $R_{[1:T]}$ pour les types $R_i \in \mathcal{R}$
- (+contraintes de projectivité)

Une analogie possible

Traiter la construction du graphe comme un problème d'auto-alignement.

Modèles formels pour les dépendances

Une formulation directe

Les variables du modèle

- $W_{[1:T]}$ pour les mots, $W_i \in [1 : |V|]$
- $H_{[1:T]}$ pour les têtes $H_i \in [0 : T]$
- $R_{[1:T]}$ pour les types $R_i \in \mathcal{R}$
- (+contraintes de projectivité)

Une analogie possible

Traiter la construction du graphe comme un problème d'auto-alignement.

Analyser comme on aligne

Auto-aligner avec IBM modèle 1

$\mathbf{f} = W_{[1:T]}, \mathbf{e} = W_{[1:T]}$ $\mathrm{P}(\,)\mathbf{f}, \mathbf{e} = \prod_{j=1}^{J} \mathrm{P}(W_i \mid W_{a_i})$

Problèmes

- paramétrisation : $\forall u, v \theta_{uv}$?
- interdiction des liens (i, i)
- une seule racine ?
- positionnement des liens ?

On peut faire mieux (IBM2, etc) voir (Brody 2010)

Analyser comme on aligne

Auto-aligner avec IBM modèle 1

$\mathbf{f} = W_{[1:T]}, \mathbf{e} = W_{[1:T]}$ $\mathrm{P}(\,)\mathbf{f}, \mathbf{e} = \prod_{j=1}^{J} \mathrm{P}(W_i \mid W_{a_i})$

Problèmes

- paramétrisation : $\forall u, v \theta_{uv}$?
- interdiction des liens (i, i)
- une seule racine ?
- positionnement des liens ?

On peut faire mieux (IBM2, etc) voir (Brody 2010)

Analyser comme on aligne

Auto-aligner avec IBM modèle 1

$\mathbf{f} = W_{[1:T]}, \mathbf{e} = W_{[1:T]}$ $\mathrm{P}(\,)\mathbf{f}, \mathbf{e} = \prod_{j=1}^{J} \mathrm{P}(W_i \mid W_{a_i})$

Problèmes

- paramétrisation : $\forall u, v \theta_{uv}$?
- interdiction des liens (i, i)
- une seule racine ?
- positionnement des liens ?

On peut faire mieux (IBM2, etc) voir (Brody 2010)

Un modèle alternatif

Dependances + Valence = DMV (Klein & Manning, 2004)

Construction récursive avec DMV : `Gen`

Partant de la racine :

1. engendrer (récursivement) les fils gauches (de droite à gauche)
2. engendrer (récursivement) les fils gauches (de gaucge à droite)

Variables et paramètres

Pour chaque tête h :

- $\mathrm{P}(\mathrm{stop} \,|\, \mathrm{dir}, h, \mathrm{adj})$ s'arrêter d'engendrer des filles (à gauche, à droite)
- $\mathrm{P}(a \,|\, \mathrm{dir}, h, adj)$ choisir la prochaine fille

$$\mathrm{P}(D(h)) = \prod_{d=l,r} \Big(\prod_{a} \mathrm{P}(s=0 \,|\, h, d, \mathrm{adj})\, \mathrm{P}(a \,|\, h, \mathrm{dir})\, \mathrm{P}(D(a)) \Big) \mathrm{P}(s=1 \,|\, h, d, \mathrm{adj})$$

Estimation par EM (Klein et Manning, 2004)

Limsi

Un modèle alternatif

Dependances + Valence = DMV (Klein & Manning, 2004)

Construction récursive avec DMV : `Gen`

Partant de la racine :

1. engendrer (récursivement) les fils gauches (de droite à gauche)
2. engendrer (récursivement) les fils gauches (de gaucge à droite)

Variables et paramètres

Pour chaque tête h :

- $\mathrm{P}(\mathrm{stop}\,|\,\mathrm{dir}, h, \mathrm{adj})$ s'arrêter d'engendrer des filles (à gauche, à droite)
- $\mathrm{P}(a\,|\,\mathrm{dir}, h, adj)$ choisir la prochaine fille

$$\mathrm{P}(D(h)) = \prod_{d=l,r} \Big(\prod_{a} \mathrm{P}(s=0\,|\,h,d,\mathrm{adj})\,\mathrm{P}(a\,|\,h,\mathrm{dir})\,\mathrm{P}(D(a)) \Big) \mathrm{P}(s=1\,|\,h,d,\mathrm{adj})$$

Estimation par EM (Klein et Manning, 2004)

Limsi

Un modèle alternatif

Dependances + Valence = DMV (Klein & Manning, 2004)

Construction récursive avec DMV : `Gen`

Partant de la racine :

1. engendrer (récursivement) les fils gauches (de droite à gauche)
2. engendrer (récursivement) les fils gauches (de gaucge à droite)

Variables et paramètres

Pour chaque tête h :

- $\mathrm{P}(\mathrm{stop}\,|\,\mathrm{dir}, h, \mathrm{adj})$ s'arrêter d'engendrer des filles (à gauche, à droite)
- $\mathrm{P}(a\,|\,\mathrm{dir}, h, adj)$ choisir la prochaine fille

$$\mathrm{P}(D(h)) = \prod_{d=l,r} \Big(\prod_{a} \mathrm{P}(s=0\,|\,h,d,\mathrm{adj})\,\mathrm{P}(a\,|\,h,\mathrm{dir})\,\mathrm{P}(D(a)) \Big) \mathrm{P}(s=1\,|\,h,d,\mathrm{adj})$$

Estimation par EM (Klein et Manning, 2004)

Mots, langages

Définitions de base

Notions de base

Σ un ensemble fini.

- L'ensemble des séquences finies de symboles de Σ est le langage universel $\Sigma^\star$.
- Une séquence $w = w_1 \dots w_n$ de $\Sigma^\star$ est appelée mot
- ε est le mot de longueur nulle ($|\varepsilon| = 0$).
- Un langage est un sous-ensemble de $\Sigma^\star$.

Exemples

- Exemple 1 : *langage* est un mot sur $\Sigma = \{a, b, \dots, z\}$
- Exemple 2 : *ceci est une phrase* est un mot sur $\Sigma = \{ceci, est, une, phrase\}$
- Exemple 3 : *AACTGCACCAGT* est un mot sur $\Sigma = \{A, C, G, T\}$

Limsi

Mots, langages

Définitions de base

Notions de base

Σ un ensemble fini.

- L'ensemble des séquences finies de symboles de Σ est le langage universel $\Sigma^\star$.
- Une séquence $w = w_1 \dots w_n$ de $\Sigma^\star$ est appelée mot
- ε est le mot de longueur nulle ($|\varepsilon| = 0$).
- Un langage est un sous-ensemble de $\Sigma^\star$.

Exemples

- Exemple 1 : *langage* est un mot sur $\Sigma = \{a, b, \dots, z\}$
- Exemple 2 : *ceci est une phrase* est un mot sur $\Sigma = \{ceci, est, une, phrase\}$
- Exemple 3 : *AACTGCACCAGT* est un mot sur $\Sigma = \{A, C, G, T\}$

Limsi

Règles de récriture

- Une règle de récriture ou production $\alpha \to \beta$ exprime le remplacement du facteur α par β ;
- α est la partie gauche ou tête de la récriture et β est le corps.
- Soit $\Sigma = \{a, b\}$ et $aba \to bab$ une règle ; elle récrit $b\underline{aba}bb$ en $b\underline{bab}bb$.
- Un ensemble de règles P définit une relation de dérivation denotée $\Rightarrow$ sur les mots de $\Sigma^\star$:

$$\alpha\beta\gamma \Rightarrow \alpha\beta'\gamma \text{ si et seulement si } \beta \to \beta' \in P$$

- la fermeture transitive de $\Rightarrow$ est $\overset{\star}{\Rightarrow}$, définie par :

$$\alpha \overset{\star}{\Rightarrow} \gamma \text{ ssi } \begin{cases} \alpha = \gamma \\ \alpha \Rightarrow \beta \text{ et } \beta \overset{\star}{\Rightarrow} \gamma \end{cases}$$

Grammaires hors-contexte

Définitions

Une grammaire hors contexte (*context-free*) est définie par $G = (\Sigma, V, S, P)$ avec :

- Σ l'alphabet fini des symboles terminaux
- V un ensemble fini de symboles non-terminaux, $V \cap \Sigma = \varnothing$
- S un symbole distingué de V, appelé l'axiome de la grammaire
- P est un ensemble de productions telles que si $\alpha \to \beta$ est dans P, alors (i) $\alpha \in V$ et (ii) $\beta \in (V \cup \Sigma)^*$

Le langage $L(G)$ d'une grammaire G est :

$$L(G) = \{w \in \Sigma^\star \text{ st. } S \stackrel{\star}{\Rightarrow} w\}$$

Un mot de $L(G)$ dérive de S et ne contient que des terminaux

Un protomot est un mot α de $(V \cup \Sigma)^*$ tel que $S \stackrel{\star}{\Rightarrow} \alpha$.

Une grammaire CF simple

Le langage $a^n b^n$

Soit $G_1 = (\{a, b\}, \{S\}, S, \{S \to aSb, S \to ab\})$

- $ab \in L(G_1)$, car $S \Rightarrow ab$
- $aabb \in L(G_1)$:
  $S \Rightarrow aSb$ (par la règle 1) et $aSb \Rightarrow aabb$ (par la règle2), donc $S \overset{\star}{\Rightarrow} aabb$.
- $\forall n > 0, a^n b^n \in L(G)$ (par récurrence)
- $L(G_1) = \{a^n b^n, n > 0\}$ (considérer les proto-mots).

Une grammaire jouet de l'anglais

La grammaire

$S \to NP\ VP$	$DET \to the \mid a \mid my\ldots$
$NP \to DET\ N$	$N \to boy \mid girl \mid book$
$NP \to NP\ PP$	$V \to cries \mid sleeps$
$VP \to V$	$V \to likes \mid reads$
$VP \to V\ NP$	$V \to dreams \mid complains$
$VP \to V\ PP$	$V \to shares \mid discusses$
$VP \to V\ NP\ NP$	$V \to gives \mid tells$
$VP \to V\ NP\ PP$	$N \to talks \mid speaks$
$VP \to V\ PP\ PP$	$P \to of \mid about \mid$
$PP \to P\ NP$	$P \to to \mid with$

$X \to \alpha \mid \beta$ remplace deux règles $X \to \alpha$ et $X \to \beta$.

Des « mots »

my girl reads a book

Une grammaire jouet de l'anglais

La grammaire

$S \to NP\ VP$	$DET \to the \mid a \mid my\ldots$
$NP \to DET\ N$	$N \to boy \mid girl \mid book$
$NP \to NP\ PP$	$V \to cries \mid sleeps$
$VP \to V$	$V \to likes \mid reads$
$VP \to V\ NP$	$V \to dreams \mid complains$
$VP \to V\ PP$	$V \to shares \mid discusses$
$VP \to V\ NP\ NP$	$V \to gives \mid tells$
$VP \to V\ NP\ PP$	$N \to talks \mid speaks$
$VP \to V\ PP\ PP$	$P \to of \mid about \mid$
$PP \to P\ NP$	$P \to to \mid with$

$X \to \alpha \mid \beta$ remplace deux règles $X \to \alpha$ et $X \to \beta$.

Des « mots »

my girl reads a book

Les langages hors-contexte

Définition

Un langage L est hors-contexte s'il existe G hors-contexte tq. $G = L(G)$

Propriétés

- les langages hors-contexte contiennent strictement les langages rationnels
- les langages hors-contexte sont rationnellement clos
- l'intersection de deux hors-contexte n'est pas toujours hors-contexte
- plus compliqué : les langages des grammaires contextuelles.

Les langages hors-contexte

Définition

Un langage L est hors-contexte s'il existe G hors-contexte tq. $G = L(G)$

Propriétés

- les langages hors-contexte contiennent strictement les langages rationnels
- les langages hors-contexte sont rationnellement clos
- l'intersection de deux hors-contexte n'est pas toujours hors-contexte
- plus compliqué : les langages des grammaires contextuelles.

Limsi

Dérivations

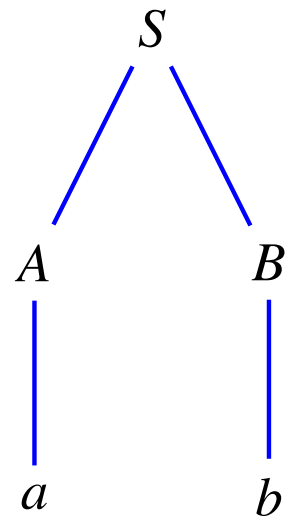
- une dérivation de w dans $L(G)$ est une suite de productions $P_1 \ldots P_k$ telles que :

$$S \Rightarrow_{P_1} \alpha_1 \Rightarrow_{P_2} \alpha_2 \ldots \Rightarrow_{P_k} w$$

- plusieurs manières de dériver w
- $G_2 = (\{a,b\}, \{A,B,S\}, S, \{S \rightarrow AB, A \rightarrow a, B \rightarrow b\})$. ab a deux dérivations :
  - $S \Rightarrow AB \Rightarrow aB \Rightarrow ab$
  - $S \Rightarrow AB \Rightarrow Ab \Rightarrow ab$
- une dérivation gauche récrit toujours le non-terminal le plus à gauche du proto-mot courant. (1) est une dérivation gauche.
- une dérivation droite récrit toujours le non-terminal le plus à droite. (2) est une dérivation droite.

Arbres de dérivation

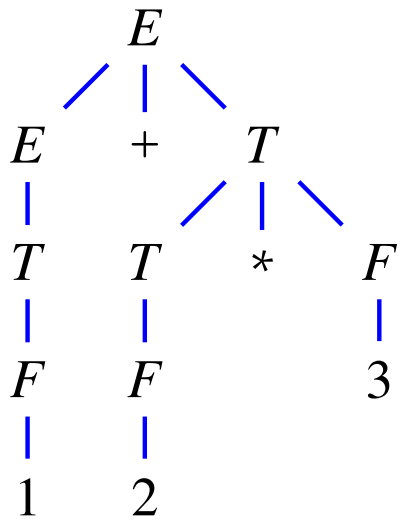
- Un arbre de dérivation de w dans $L(G)$ représente un ensemble de dérivations équivalentes.



- L'arbre de dérivation de $w = aabb$ dans G_2 :
- Un arbre de dérivation est un arbre (acyclique, connexe) étiqueté :
  - chaque nœud interne (resp. feuille) est étiqueté par un non-terminal (resp. terminal)
  - si $n_1, n_2 \ldots n_k$, respectivement étiquetés $X_1 \ldots X_k$ sont des filles de n (étiqueté X), alors $X \to X_1 \ldots X_k$ est une production de G.

Mots linéaires, arbres hiérarchiques

- L'arbre de dérivation représente la structure (hiérarchique) calculée par la grammaire.
- $G_3 = (\{1, 2, \ldots, +, *\}, \{T, F, E\}, E, \{E \to T, E \to E + T, T \to F, T \to T * F, F \to 1 \ldots\})$. L'arbre de dérivation pour $w = 1 + 2 * 3$ dans G_3 :



- l'arbre de dérivation détermine l'interprétation (la sémantique) d'un mot

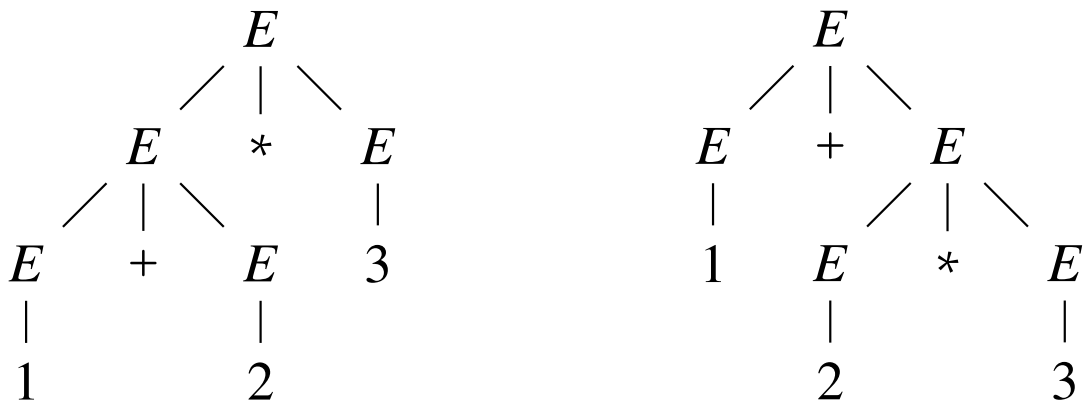
Ambiguïté

- Une grammaire est ambiguë si au moins un mot a plus d'une dérivation gauche ($\Leftrightarrow$ a plus d'un arbre de dérivation)
- $G_4 = (\{1, 2, \ldots, +, *\}, \{E\}, E, \{E \to E + E, E \to E * E, E \to 1 \ldots\})$.
  deux arbres de dérivation pour $w = 1 + 2 * 3$:

- Un langage est ambigu ssi toutes ses grammaires sont ambiguës.

Ambiguïté

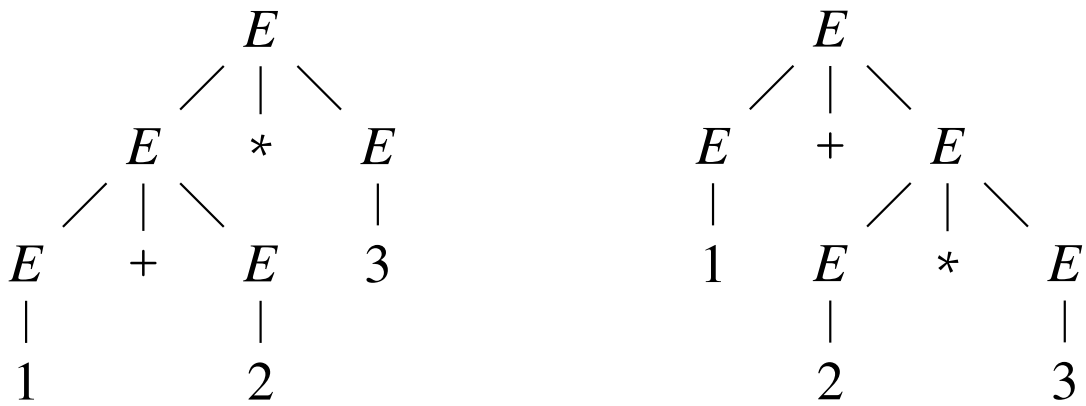
- Une grammaire est ambiguë si au moins un mot a plus d'une dérivation gauche ($\Leftrightarrow$ a plus d'un arbre de dérivation)
- $G_4 = (\{1, 2, \ldots, +, *\}, \{E\}, E, \{E \to E + E, E \to E * E, E \to 1 \ldots\})$.
  deux arbres de dérivation pour $w = 1 + 2 * 3$:



- Un langage est ambigu ssi toutes ses grammaires sont ambiguës.

Ambiguïté

- Une grammaire est ambiguë si au moins un mot a plus d'une dérivation gauche ($\Leftrightarrow$ a plus d'un arbre de dérivation)
- $G_4 = (\{1, 2, \ldots, +, *\}, \{E\}, E, \{E \to E + E, E \to E * E, E \to 1 \ldots\})$.
  deux arbres de dérivation pour $w = 1 + 2 * 3$:



- Un langage est ambigu ssi toutes ses grammaires sont ambiguës.

Ambiguité (suite)

- Les langages formels doivent être sans ambiguïté : les ambiguités doivent être résolues (comment ? par reformulation de la grammaire).
- Les langages naturels sont ambigus pour les humains :
  - *time flies like an arrow* (rare)
  - *I saw a man with a telescope* (fréquent)
- ... et encore plus pour les machines
  - *Dog trains like an arrow*
  - *I saw a man with an umbrella*

Comment faire ? Probabiliser les décisions

Limsi

Equivalence et formes normales

- deux grammaires G_1 et G_2 sont équivalentes ssi $L(G_1) = L(G_2)$
- elles sont fortement (resp. faiblement) équivalentes ssi de plus les arbres de dérivation dans G_1 et G_2 sont identiques (resp. différents).
- pour toute grammaire, $\exists$ une grammaire fortement équivalente qui ne contient que des terminaux, non terminaux et productions utiles (= utilisés pour engendrer au moins un mot)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \rightarrow BC$ ou $A \rightarrow a$ (Forme Normale de Chomsky ou CNF)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \rightarrow a\,\alpha$ (Forme Normale de Greibach)

Equivalence et formes normales

- deux grammaires G_1 et G_2 sont équivalentes ssi $L(G_1) = L(G_2)$
- elles sont fortement (resp. faiblement) équivalentes ssi de plus les arbres de dérivation dans G_1 et G_2 sont identiques (resp. différents).
- pour toute grammaire, $\exists$ une grammaire fortement équivalente qui ne contient que des terminaux, non terminaux et productions utiles (= utilisés pour engendrer au moins un mot)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \to BC$ ou $A \to a$ (Forme Normale de Chomsky ou CNF)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \to a\,\alpha$ (Forme Normale de Greibach)

Equivalence et formes normales

- deux grammaires G_1 et G_2 sont équivalentes ssi $L(G_1) = L(G_2)$
- elles sont fortement (resp. faiblement) équivalentes ssi de plus les arbres de dérivation dans G_1 et G_2 sont identiques (resp. différents).
- pour toute grammaire, $\exists$ une grammaire fortement équivalente qui ne contient que des terminaux, non terminaux et productions utiles (= utilisés pour engendrer au moins un mot)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \rightarrow BC$ ou $A \rightarrow a$ (Forme Normale de Chomsky ou CNF)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \rightarrow a\,\alpha$ (Forme Normale de Greibach)

Equivalence et formes normales

- deux grammaires G_1 et G_2 sont équivalentes ssi $L(G_1) = L(G_2)$
- elles sont fortement (resp. faiblement) équivalentes ssi de plus les arbres de dérivation dans G_1 et G_2 sont identiques (resp. différents).
- pour toute grammaire, $\exists$ une grammaire fortement équivalente qui ne contient que des terminaux, non terminaux et productions utiles (= utilisés pour engendrer au moins un mot)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \to BC$ ou $A \to a$ (Forme Normale de Chomsky ou CNF)
- pour toute grammaire, $\exists$ une grammaire faiblement équivalente dont toutes les productions sont de la forme $A \to a\,\alpha$ (Forme Normale de Greibach)

Parsage = recherche

- $\Rightarrow$ definit un graphe orienté (=relation binaire) sur l'ensemble des proto-mots.
- $w \stackrel{?}{\in} L(G)$ équivaut à trouver un chemin dans ce graphe.
- attention : le graphe est infini (seulement localement fini)
- stratégies de recherche :
  - de S vers w : parsage descendant ou top-down
  - de w vers S : parsage ascendant ou bottom-up
  - profondeur d'abord (exploration séquentielle des chemins)
  - largeur d'abord (exploration en parallèle des chemins)
  - ...

Limsi

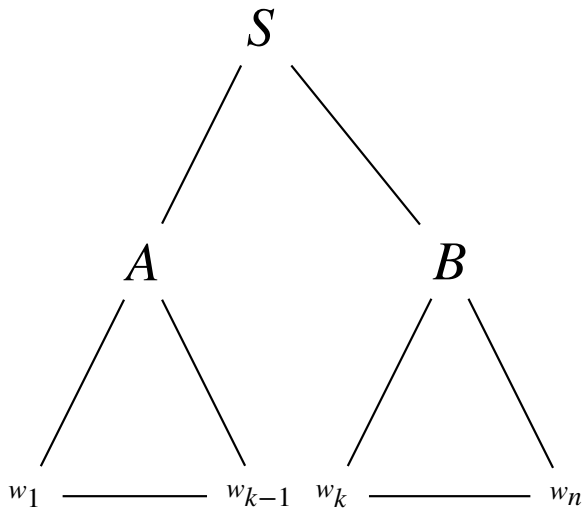
Une grammaire ambiguë

p_1	S	$\rightarrow GN\ GV$		p_{15}	V	$\rightarrow mange \mid sert$
p_2	GN	$\rightarrow DET\ N$		p_{16}	V	$\rightarrow donne$
p_3	GN	$\rightarrow GN\ GNP$		p_{17}	V	$\rightarrow boude \mid s'ennuie$
p_4	GN	$\rightarrow NP$		p_{18}	V	$\rightarrow parle$
p_5	GV	$\rightarrow V$		p_{19}	V	$\rightarrow coupe \mid avale$
p_6	GV	$\rightarrow V\ GN$		p_{20}	V	$\rightarrow discute \mid gronde$
p_7	GV	$\rightarrow V\ GNP$		p_{21}	NP	$\rightarrow Louis \mid Paul$
p_8	GV	$\rightarrow V\ GN\ GNP$		p_{22}	NP	$\rightarrow Marie \mid Sophie$
p_9	GV	$\rightarrow V\ GNP\ GNP$		p_{23}	N	$\rightarrow fille \mid cousine \mid tante$
p_{10}	GNP	$\rightarrow PP\ GN$		p_{24}	N	$\rightarrow paternel \mid fils \mid$
p_{11}	PP	$\rightarrow de \mid à$		p_{25}	N	$\rightarrow viande \mid soupe \mid salade$
p_{12}	DET	$\rightarrow la \mid le$		p_{26}	N	$\rightarrow dessert \mid fromage \mid pain$
p_{13}	DET	$\rightarrow sa \mid son$		p_{27}	ADJ	$\rightarrow petit \mid gentil$
p_{14}	DET	$\rightarrow un \mid une$		p_{28}	ADJ	$\rightarrow petite \mid gentille$

Les deux idées de CYK

Propriété de sous-structure

Si G est CNF, tout arbre couvrant $w[1:n]$ se décompose en exactement deux sous-arbres couvrant $w[1:k-1]$ et $w[k:n]$ (et récursivement).



Tabulation

Pour chaque $A \in V$, $i,j \in 1:n+1$, $A \stackrel{\star}{\Rightarrow} ?w[i:j]$ n'est calculé qu'une fois et stocké dans une table

Les deux idées de CYK

Propriété de sous-structure

Si G est CNF, tout arbre couvrant $w[1:n]$ se décompose en exactement deux sous-arbres couvrant $w[1:k-1]$ et $w[k:n]$ (et récursivement).

Tabulation

Pour chaque $A \in V$, $i,j \in 1:n+1$, $A \stackrel{\star}{\Rightarrow} ?w[i:j]$ n'est calculé qu'une fois et stocké dans une table

Les sous-chaînes bien formées

6	S				
5					
3	S		GV		
2	GN			GN	
1	DET	N	V, GV	DET	N
	ma	sœur	mange	une	pomme
	1	2	3	4	5

$$X \in T[i:j] \Leftrightarrow X \stackrel{\star}{\Rightarrow} w[i:j-1]$$

Cocke Younger Kasami - CYK

// la phrase à analyser est $w = w_1 \dots w_n$
$T[i,j] \leftarrow \varnothing$;
for $i = 1 \dots n$ **do**
  foreach $A \rightarrow u_i \in P$ **do**
    $T[i, i+1] \leftarrow T[i, i+1] \cup \{A\}$
  end
end
for $l = 2 \dots n$ **do**
  for $i = 1 \dots n - l + 1$ **do**
    for $k = i + 1 \dots i + l - 1$ **do**
      if $B \in T[i,k] \wedge C \in T[k, i+l] \wedge A \rightarrow BC \in P$ **then**
        $T[i, i+l] :=\leftarrow T[i, i+l] \cup \{A\}$
      end
    end
  end
end
// Succès : S couvant $1 \dots n+1$
if $S \in T[1,n]$ **then**
  return (**true**)
end
return (**false**)

CYK - application

p_1	S	$\rightarrow GN\ GV$	p_{11}	V	$\rightarrow joue \mid travaille$
p_2	GN	$\rightarrow DET\ N$	p_{12}	V	$\rightarrow donne \mid corde$
p_3	GN	$\rightarrow GN\ GNP$	p_{13}	V	$\rightarrow gratte \mid compose$
p_4	GV	$\rightarrow V\ GN$	p_{14}	V	$\rightarrow écoute$
p_5	GV	$\rightarrow V\ GNP$	p_{15}	V	$\rightarrow enregistre$
p_6	GNP	$\rightarrow PP\ GN$	p_{16}	N	$\rightarrow air \mid refrain$
p_7	PP	$\rightarrow de \mid à$	p_{17}	N	$\rightarrow guitare \mid gratte \mid corde$
p_8	DET	$\rightarrow la \mid le$	p_{18}	N	$\rightarrow son \mid piano \mid la$
p_9	DET	$\rightarrow sa \mid son$	p_{19}	N	$\rightarrow chanteuse \mid guitariste \mid pièce$
p_{10}	DET	$\rightarrow un \mid une$	p_{20}	N	$\rightarrow composition \mid chanteur \mid concert$

le chanteur compose un refrain ; la guitariste gratte une corde de sa guitare

Probabiliser les langages hors-contexte

Grammaires probabilistes hors-contexte (PCFG)

$G = (\Sigma, V, S, \boldsymbol{\theta})$, avec Σ un alphabet fini, V un ensemble de non-terminaux, S l'axiome et $\boldsymbol{\theta} : V \times (V \cup \Sigma)^* \to [0,1]$ tq. $\forall (A \to \alpha), \sum_\alpha \theta_{A\to\alpha} = 1$

Probabilité d'un arbre de dérivation

Soit τ un arbre de dérivation de G, utilisant $c(A \to \alpha)$ fois la production $A \to \alpha$, alors :

$$\mathrm{P}(\tau) = \prod_{A\to\alpha\in\tau} \theta_{A\to\alpha}^{c(A\to\alpha)}$$

NB. $\mathrm{P}()$ *n'est pas toujours une distribution sur les arbres engendrés par* G *: il faut en plus éviter que les dérivations infinies aient une probabilité* > 0.

Probabilité d'un mot

$$\mathrm{P}(w_1 \dots w_n) = \sum_{yield(\tau)=w_1\dots w_n} \mathrm{P}(\tau)$$

Encore une somme de produits

Probabiliser les langages hors-contexte

Grammaires probabilistes hors-contexte (PCFG)

$G = (\Sigma, V, S, \boldsymbol{\theta})$, avec Σ un alphabet fini, V un ensemble de non-terminaux, S l'axiome et $\boldsymbol{\theta} : V \times (V \cup \Sigma)^* \to [0,1]$ tq. $\forall (A \to \alpha), \sum_\alpha \theta_{A\to\alpha} = 1$

Probabilité d'un arbre de dérivation

Soit τ un arbre de dérivation de G, utilisant $c(A \to \alpha)$ fois la production $A \to \alpha$, alors :

$$\mathrm{P}(\tau) = \prod_{A\to\alpha\in\tau} \theta_{A\to\alpha}^{c(A\to\alpha)}$$

NB. $\mathrm{P}()$ *n'est pas toujours une distribution sur les arbres engendrés par* G *: il faut en plus éviter que les dérivations infinies aient une probabilité* > 0.

Probabilité d'un mot

$$\mathrm{P}(w_1 \dots w_n) = \sum_{yield(\tau)=w_1\dots w_n} \mathrm{P}(\tau)$$

Encore une somme de produits

Probabiliser les langages hors-contexte

Grammaires probabilistes hors-contexte (PCFG)

$G = (\Sigma, V, S, \boldsymbol{\theta})$, avec Σ un alphabet fini, V un ensemble de non-terminaux, S l'axiome et $\boldsymbol{\theta} : V \times (V \cup \Sigma)^* \to [0,1]$ tq. $\forall (A \to \alpha), \sum_\alpha \theta_{A\to\alpha} = 1$

Probabilité d'un arbre de dérivation

Soit τ un arbre de dérivation de G, utilisant $c(A \to \alpha)$ fois la production $A \to \alpha$, alors :

$$\mathrm{P}(\tau) = \prod_{A\to\alpha\in\tau} \theta_{A\to\alpha}^{c(A\to\alpha)}$$

NB. $\mathrm{P}()$ *n'est pas toujours une distribution sur les arbres engendrés par* G *: il faut en plus éviter que les dérivations infinies aient une probabilité* > 0.

Probabilité d'un mot

$$\mathrm{P}(w_1 \dots w_n) = \sum_{yield(\tau)=w_1\dots w_n} \mathrm{P}(\tau)$$

Encore une somme de produits

Tentative de représentation graphique

Décomposition récursive de la loi jointe

Grammaire

$G = (\Sigma, V = V_1 \cup V_2, S, P)$ CNF ; les règles de P

- $A \to BC, A \in V_1, B, C \in V$
- $A \to w, A \in V_2$ (règles pré-terminales)

Représentations

- Un arbre binaire τ couvrant $w_{[1:T]}$ contient $T - 1$ nœuds internes étiquetés par des NT.
- Un arbre de dérivation complet
  - un ensemble de variables $Y_1 \dots Y_{T-1}$ à valeurs dans V_1
  - un ensemble de variables $Y_T \dots Y_{2T-1}$ à valeurs dans V_2 (pré-terminaux)
  - un ensemble de variables $W_1 \dots W_T$ à valeurs dans Σ

Limsi

Tentative de représentation graphique

Décomposition récursive de la loi jointe

$$\tau = \{A, W = w\}$$

$$\mathrm{P}(T = \tau, W = w) = \mathrm{P}(A \rightarrow w)$$

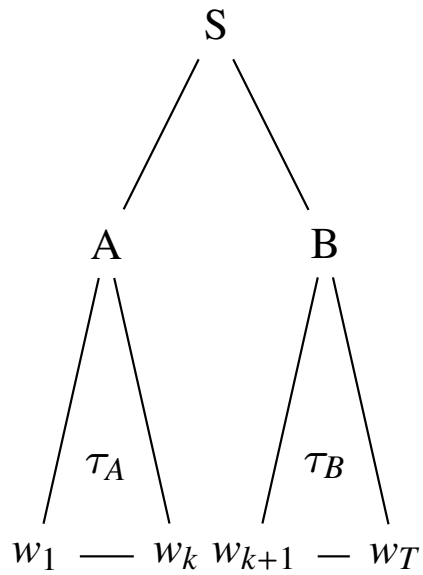
Tentative de représentation graphique

Décomposition récursive de la loi jointe

$$\tau = \{A, W = w\}$$

$$\mathrm{P}(T = \tau, W = w) = \mathrm{P}(A \to w)$$

$$\tau = \{S \to AB, \tau_A, \tau_B\}, W_{[1:T]} = w_{[1:T]}$$



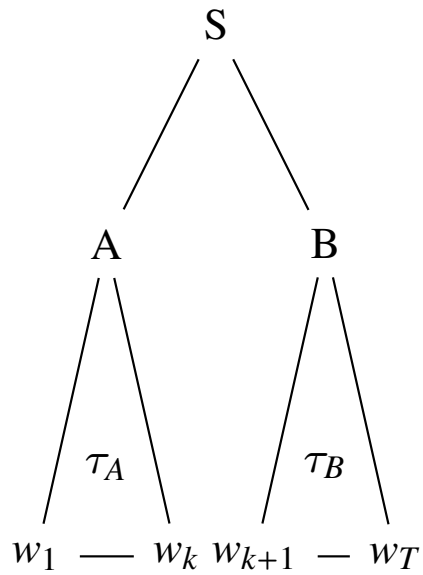
$$\begin{aligned}\mathrm{P}(T = \tau, W = w) = &\mathrm{P}(Y_1 = S)\,\mathrm{P}(Y_2 Y_3 \mid Y_1 = S) \times \\ &\mathrm{P}(T_2 = \tau_A, W_{[1:k]} \mid Y_2 = A) \times \\ &\mathrm{P}(T_3 = \tau_B, W_{[k+1:T]} \mid Y_3 = B)\end{aligned}$$

Tentative de représentation graphique

Décomposition récursive de la loi jointe

$$T_2 \perp\!\!\!\perp T_3 \,|\, Y_2, Y_3$$

$$\tau = \{S \to AB, \tau_A, \tau_B\}, W_{[1:T]} = w_{[1:T]}$$



$$\begin{aligned}\mathrm{P}(T = \tau, W = w) = &\mathrm{P}(Y_1 = S)\,\mathrm{P}(Y_2 Y_3 \,|\, Y_1 = S)\times \\ &\mathrm{P}(T_2 = \tau_A, W_{[1:k]} \,|\, Y_2 = A)\times \\ &\mathrm{P}(T_3 = \tau_B, W_{[k+1:T]} \,|\, Y_3 = B)\end{aligned}$$

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax} \mathrm{P}(A \Rightarrow w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'une fonction des règles si l'on dispose d'une banque d'arbres ;
  - Par l'algorithme EM si l'on ne connait que les productions.
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'un échantillon (si l'on dispose d'une banque d'arbres)
  - Par l'algorithme EM = par inférence si l'on ne connaît que les phrases
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives si on dispose d'une banque d'arbres ;
  - Par l'algorithme EM si on ne dispose que de phrases (cf. apprentissage).
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'utilisation des règles, si l'on dispose d'une banque d'arbres :
  - Par l'algorithme EM = [illegible] si l'on connaît quelques propriétés.
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'utilisation des règles, si l'on dispose d'une banque d'arbres ;
  - Par l'algorithme EM = inside/outside, si l'on ne connait que les productions.
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'utilisation des règles, si l'on dispose d'une banque d'arbres ;
  - Par l'algorithme EM = inside/outside, si l'on ne connait que les productions.
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'utilisation des règles, si l'on dispose d'une banque d'arbres ;
  - Par l'algorithme EM = inside/outside, si l'on ne connait que les productions.
- Apprendre G est beaucoup plus difficile (IG)

Trois problèmes pour les PCFGs

- Calculer $\mathrm{P}(w_1 \dots w_n)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$ comme la somme des probabilités de tous les sous-constituants par l'algorithme inside.
- Calculer $\mathrm{argmax}_{yield(\tau)=w_1 \dots w_n} \mathrm{P}(\tau)$
  - Par programmation dynamique (cf. CYK) en calculant récursivement $T[i,j,A] = \mathrm{argmax}\, \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Estimer les probabilités (G connue) :
  - Au maximum de vraisemblance par les fréquences relatives d'utilisation des règles, si l'on dispose d'une banque d'arbres ;
  - Par l'algorithme EM = inside/outside, si l'on ne connait que les productions.
- Apprendre G est beaucoup plus difficile (IG)

L'algorithme *Inside*

Hypothèses et notations

- G sous forme normale de Chomsky (productions $X \to A\,B$ ou $X \to w$)
- T table d'analyse :

$$\forall A, T[i,j,A] \doteq \sum_{\tau \text{ tq. } \begin{cases} yield(\tau) = w_i \ldots w_{j-1} \\ root(\tau) = A \end{cases}} \mathrm{P}(\tau)$$

Calcul de T par DP *(bottom-up)*

- $T[i,i+1,A] = \mathrm{P}(A \to w_i)$
- $T[i,j,A] = \sum_k \sum_{A \to BC} \theta_{A \to BC} \times T[i,k,B] \times T[k+1,j,C]$

L'algorithme *Inside*

Hypothèses et notations

- G sous forme normale de Chomsky (productions $X \to A\,B$ ou $X \to w$)
- T table d'analyse :

$$\forall A, T[i,j,A] \doteq \sum_{\tau \text{ tq. } \begin{cases} yield(\tau) = w_i \ldots w_{j-1} \\ root(\tau) = A \end{cases}} \mathrm{P}(\tau)$$

Calcul de T par DP *(bottom-up)*

- $T[i,i+1,A] = \mathrm{P}(A \to w_i)$
- $T[i,j,A] = \sum_k \sum_{A \to BC} \theta_{A \to BC} \times T[i,k,B] \times T[k+1,j,C]$

Cocke Younger Kasami - CYK

/* la phrase à analyser est $w = w_1 \dots w_n$ */
$T[i,j] \leftarrow \varnothing$;
for $i = 1 \dots n$ **do**
  foreach $A \rightarrow w_i \in P$ **do**
    $T[i, i+1] \leftarrow T[i, i+1] \cup \{A\}$
  end
end
/* Construit les constituants de longueur croissante */
for $l = 2 \dots n$ **do**
  for $i = 1 \dots n - l + 1$ **do**
    for $k = i + 1 \dots i + l - 1$ **do**
      if $B \in T[i,k] \wedge C \in T[k, i+l] \wedge A \rightarrow BC \in P$ **then**
        $T[i, k+l] :=\leftarrow T[i, k+l] \cup \{A\}$
      end
    end
  end
end
/* Succès: S couvant $1 \dots n+1$ */
if $S \in T[1,n]$ **then**
  return (**true**)
end
return (**false**)

CYK - Inside

/* la phrase à analyser est $w = w_1 \dots w_n$ */
$T[i,j,X] \leftarrow 0$;
for $i = 1 \dots n$ **do**
  foreach $A \rightarrow w_i \in P$ **do**
    $T[i, i+1, A] \leftarrow \mathrm{P}(A \rightarrow u_i)$
  end
end
/* Construit les constituants de longueur croissante */
for $l = 2 \dots n$ **do**
  for $i = 1 \dots n - l + 1$ **do**
    for $k = i + 1 \dots i + l - 1$ **do**
      if $T[i,k,B] > 0 \wedge T[k, i+l, C] > 0 \wedge \mathrm{P}(A \rightarrow BC) > 0$ **then**
        $T[i, k+l, A] :\leftarrow T[i, k+l, A] + \theta_{A \rightarrow BC} \times T[i,k,B] \times T[k, i+l, C]$
      end
    end
  end
end
/* Succès : S couvant $1 \dots n+1$ */
if $T[1, n, S] > 0$ **then**
  return $(T[1, n, S])$
end
return (0)

Inside et Outside

L'analogue arboré de Forward et Backward

Définitions

- Probabilité Inside : $T[i,j,A] = \sum_A \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Probabilité Outside : $S[i,j,A] = \mathrm{P}(S \stackrel{\star}{\Rightarrow} w_1 \dots w_{i-1} A\, w_j \dots w_n)$

Calcul de S par DP *(top-down)*

$$\begin{cases} S[1,m,S] = 1, S[1,m,A] = 0 \text{ pour } A \neq S \\ S[i,j,A] = \sum_{X \to A\,B} \sum_{k>j} S[i,k,X] \times \theta_{X \to A\,B} \times T[j,k,B] + \\ \qquad \sum_{X \to B\,A} \sum_{k} S[k,j,X] \times \theta_{X \to B\,A} \times T[k,i-1,B] \end{cases}$$

Décomposition des probabilités totales

probabilité qu'une dérivation de S utilise X pour couvrir $w_i \dots w_{j-1}$

$$\forall i,j,X, : T[i,j,X] \times S[i,j,X] = \mathrm{P}(S \stackrel{\star}{\Rightarrow} w_1 \dots w_{i-1} X w_j \dots w_n) \times \mathrm{P}(X \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$$

Inside et Outside

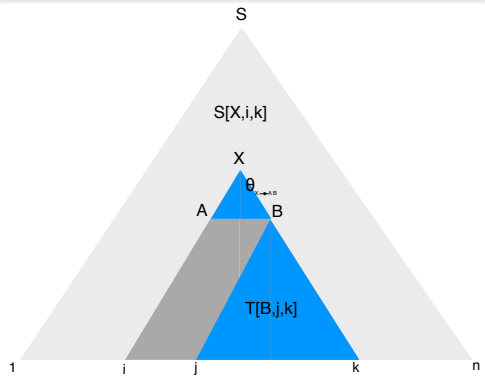
L'analogue arboré de Forward et Backward

Définitions

- Probabilité Inside : $T[i,j,A] = \sum_A \mathrm{P}(A \overset{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Probabilité Outside : $S[i,j,A] = \mathrm{P}(S \overset{\star}{\Rightarrow} w_1 \dots w_{i-1} A\, w_j \dots w_n)$

Calcul de S par DP *(top-down)*

$$\begin{cases} S[1,m,S] = 1, S[1,m,A] = 0 \text{ pour } A \neq S \\ S[i,j,A] = \sum_{X \to A\,B} \sum_{k>j} S[i,k,X] \times \theta_{X \to A\,B} \times T[j,k,B] + \\ \qquad \sum_{X \to B\,A} \sum_{k} S[k,j,X] \times \theta_{X \to B\,A} \times T[k,i-1,B] \end{cases}$$



Limsi

Décomposition des probabilités totales

Inside et Outside

L'analogue arboré de Forward et Backward

Définitions

- Probabilité Inside : $T[i,j,A] = \sum_A \mathrm{P}(A \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$
- Probabilité Outside : $S[i,j,A] = \mathrm{P}(S \stackrel{\star}{\Rightarrow} w_1 \dots w_{i-1} A\, w_j \dots w_n)$

Calcul de S par DP *(top-down)*

$$\begin{cases} S[1,m,S] = 1, S[1,m,A] = 0 \text{ pour } A \neq S \\ S[i,j,A] = \sum_{X \to A\,B} \sum_{k>j} S[i,k,X] \times \theta_{X \to A\,B} \times T[j,k,B] + \\ \qquad \sum_{X \to B\,A} \sum_{k} S[k,j,X] \times \theta_{X \to B\,A} \times T[k,i-1,B] \end{cases}$$

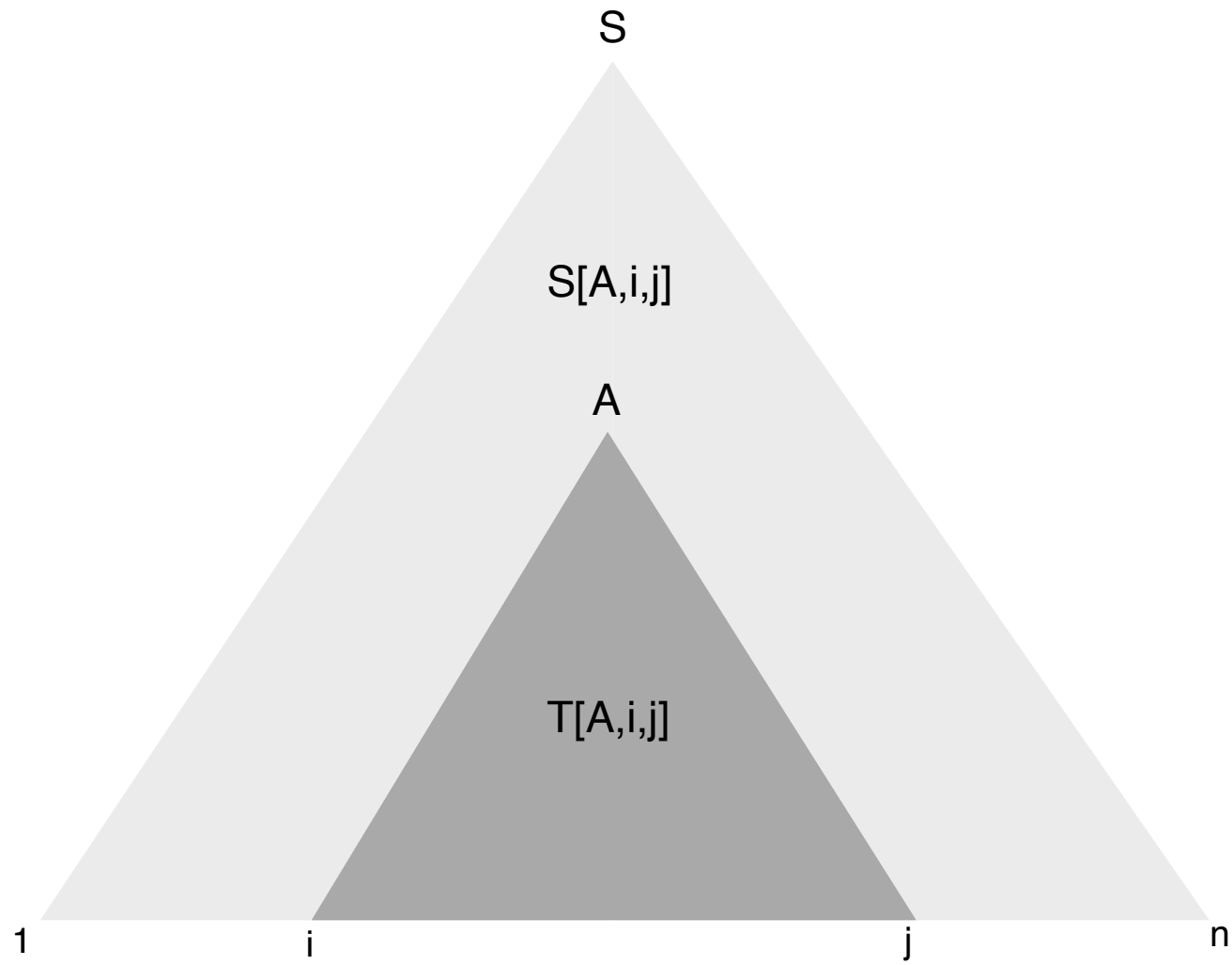
Décomposition des probabilités totales

probabilité qu'une dérivation de S utilise X pour couvrir $w_i \dots w_{j-1}$

$$\forall i,j,X, : T[i,j,X] \times S[i,j,X] = \mathrm{P}(S \stackrel{\star}{\Rightarrow} w_1 \dots w_{i-1} X w_j \dots w_n) \times \mathrm{P}(X \stackrel{\star}{\Rightarrow} w_i \dots w_{j-1})$$

Inside et Outside

L'analogue arboré de Forward et Backward



Retour de EM

En utilisant l'algorithme Inside/Outside

La fonction auxiliaire

$$
\begin{aligned}
Q_{\theta'}(\theta) &= \sum_{\tau} \mathrm{P}(\tau \mid w_1 \ldots w_n; \theta') \log \mathrm{P}(\tau, w_1 \ldots w_n \mid \theta) \\
&= \sum_{\tau} \mathrm{P}(\tau \mid w_1 \ldots w_n; \theta') \sum_{X \to A\,B} c(X \to AB) \log \theta_{X \to A\,B}
\end{aligned}
$$

Maximiser la fonction auxiliaire... fait apparaître l'espérance des comptes

$$
\begin{aligned}
\sum_{\tau} \mathrm{P}(\tau \mid w_1 \ldots w_n; \theta') c(X \to AB) &= \sum_{\tau} \mathrm{P}(\tau \mid w_1 \ldots w_n; \theta') \sum_{i,k,j} c(X \to AB; i,k,j) \\
&= \sum_{i,k,j} \sum_{\tau} \mathrm{P}(\tau \mid w_1 \ldots w_n; \theta') c(X \to AB; i,k,j) \\
&\propto \sum_{i,k,j} S[X,i,j] \times \theta_{X \to AB} \times T[A,i,k] \times T[B,j,k]
\end{aligned}
$$

Les étapes de l'algorithme EM

En utilisant l'algorithme Inside/Outside

- Étape E : pour chaque phrase :
  - remplir T (récursion inside)
  - remplir S (récursion outside)
  - calculer l'espérance des comptes

$$\forall X \to AB, c(X \to AB) + = \sum_{i,j} \sum_{k} \frac{S[X,i,j]T[A,i,k]T[B,k,j]\theta_{X \to AB}}{T[S,1,n]}$$

- Étape M :

$$\theta_{X \to AB} = \frac{c(X \to AB)}{\sum_{\alpha} c(X \to \alpha)}$$

Au delà des PCFGs

- Adaptation pour les grammaires non CNF
- Versions « synchrones » (transducteurs dans les arbres)
- Probabilisation des grammaires contextuelles : TAG, LFG, HPSG
- Probabilisation des grammaires de dépendances
- Probabilisation de tout modèle génératif : automate à pile, automates d'arbres
  Recette (générale) :
  - identifier les actions élémentaires (par ex. récriture)
  - obtenir les dérivations par enchainement (produit) d'actions élémentaires
  - marginaliser sur les dérivations pour les probabilités des structures
  - marginaliser sur les structures pour les probabilités des séquences

Limsi