



RÉ-ORDONNANCEMENT VIA PROGRAMMATION DYNAMIQUE POUR L'ADAPTATION CROSS-LINGUE D'UN ANALYSEUR EN DÉPENDANCES

Nicolas Devatine*

Caio Corro**

François Yvon**

* Université Toulouse 3, IRIT

** Université Paris-Saclay, CNRS, LISN



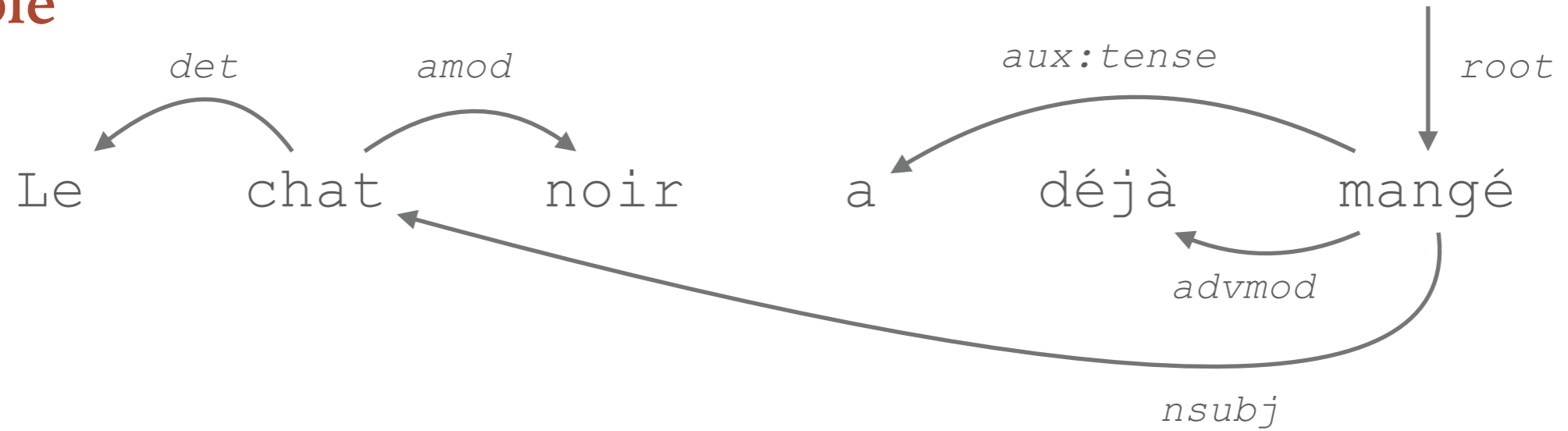
ANALYSE EN DÉPENDANCES SYNTAXIQUES

Exemple

Le chat noir a déjà mangé

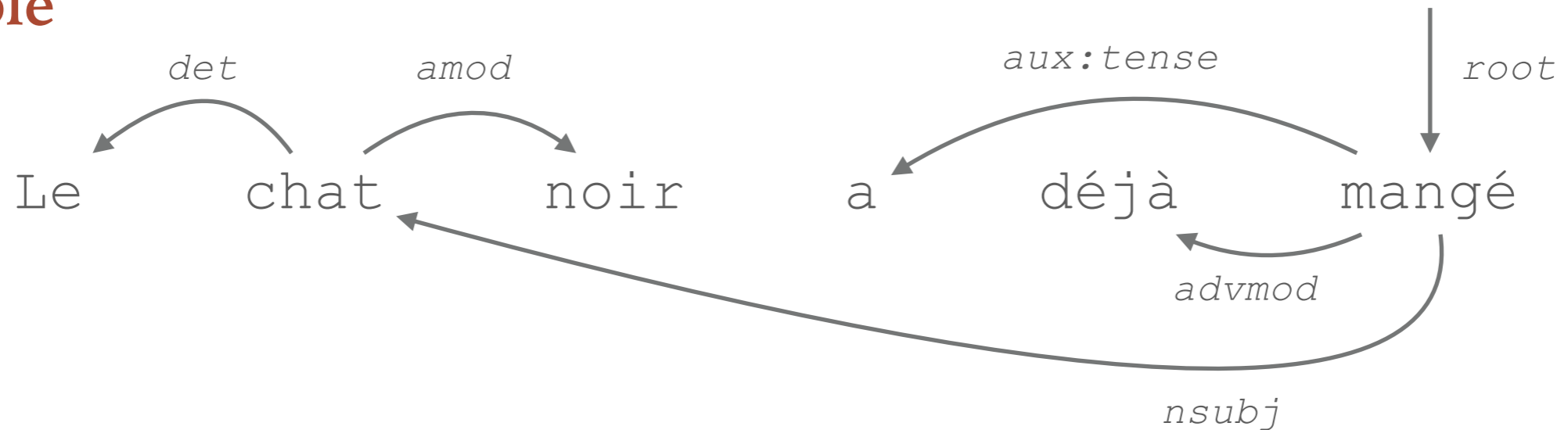
ANALYSE EN DÉPENDANCES SYNTAXIQUES

Exemple



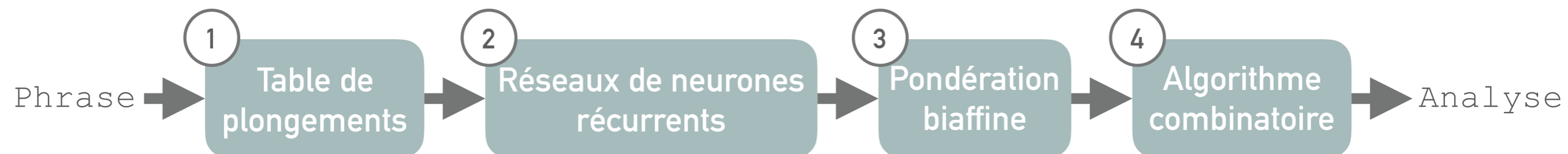
ANALYSE EN DÉPENDANCES SYNTAXIQUES

Exemple



Prédiction fondée sur les graphes

1. Récupération des plongements lexicaux
2. Calcul de représentations sensibles au contexte
3. Calcul de la vraisemblance (intensité de l'association) des dépendances entre chaque couple de mots
4. Calcul de la structure en dépendances de vraisemblance maximale

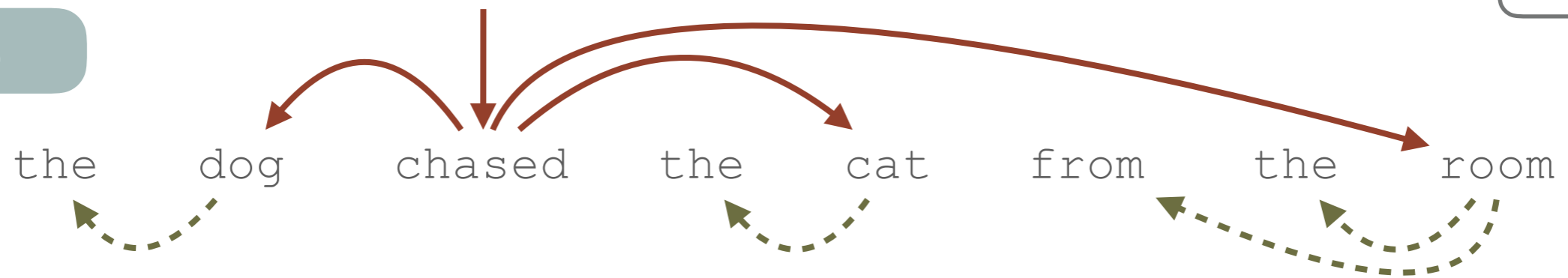


ANALYSE EN DÉPENDANCES SYNTAXIQUES CROSS-LINGUE

Annotations « universal dependencies »

- Simplifie la caractérisation des parties du discours Petrov et al. (2012)
- Favorise les mots pleins comme têtes pour rendre les structures similaires entre langues Nivre et al. (2016)

Anglais



Finnois



- Dépendance entre deux mots pleins
- - - - -→ Dépendance entre un mot plein et un mot outil

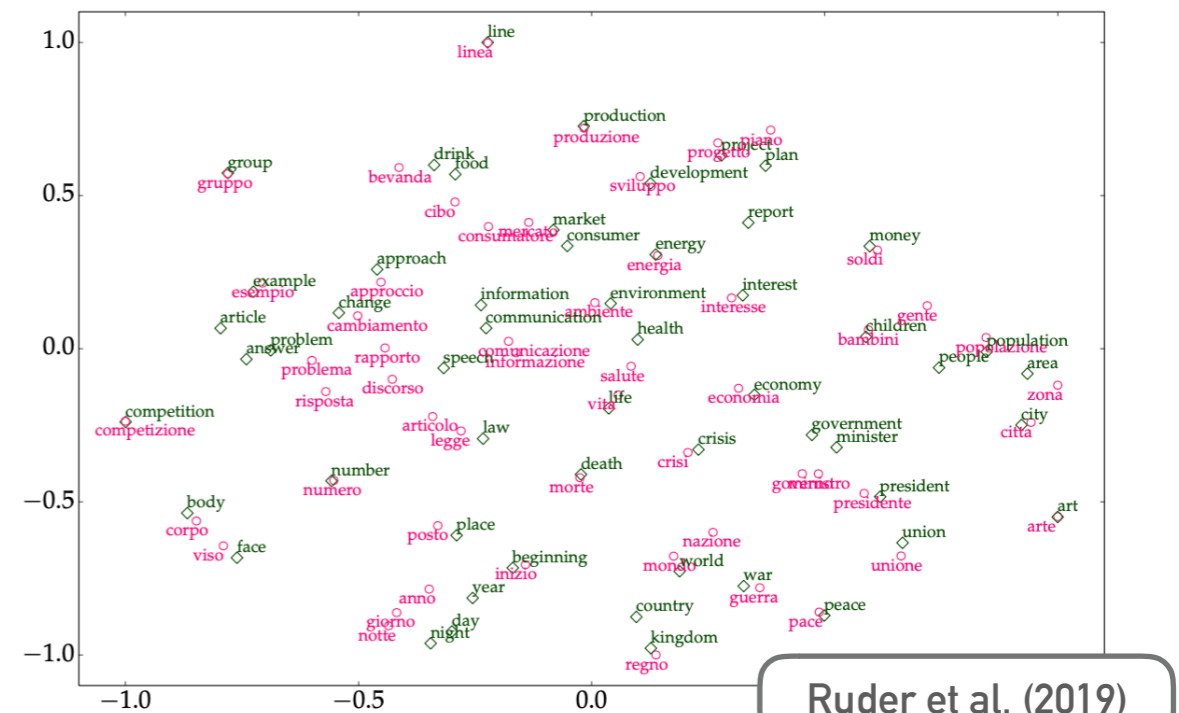
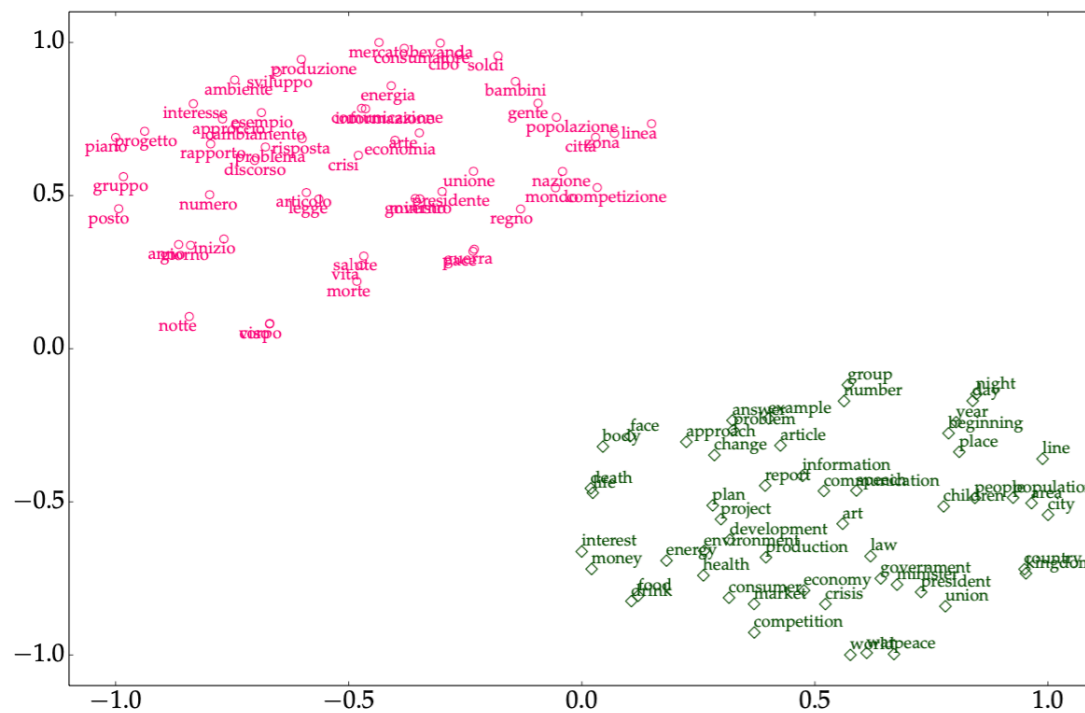
Analyse en dépendances cross-lingue

1. Apprentissage sur une langue source (anglais)
2. Prédiction sur d'autres langues cibles (français, etc.)

REPRÉSENTATION NON CONTEXTUELLE MULTI-LINGUE

Plongement lexicaux multilingue

Plongement tels que les mots ayant des « rôles similaires » dans leur langue respective ont des plongements proches



Ruder et al. (2019)



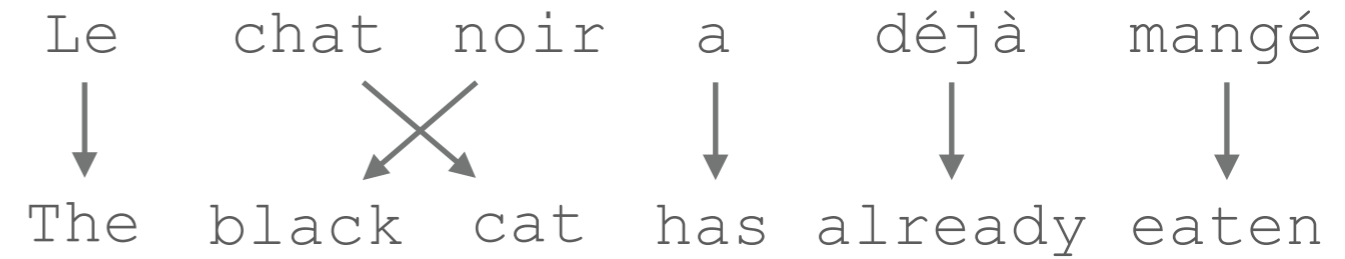
Multilingues ! ✓
Joulin et al. (2018)

REPRÉSENTATION CONTEXTUELLE MULTI-LINGUE

Ordre dominant des mots

Diversité des ordres des mots dans différentes langues :

- SVO, SOV, VSO, ...
- Adjectif avant et/ou après le nom
- ...

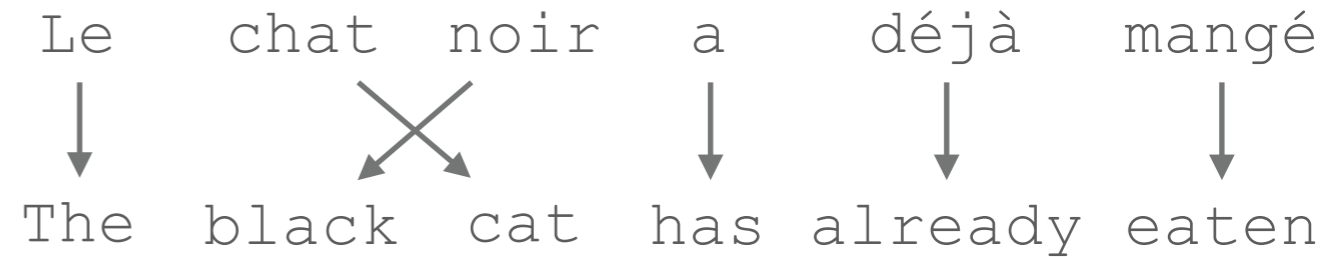


REPRÉSENTATION CONTEXTUELLE MULTI-LINGUE

Ordre dominant des mots

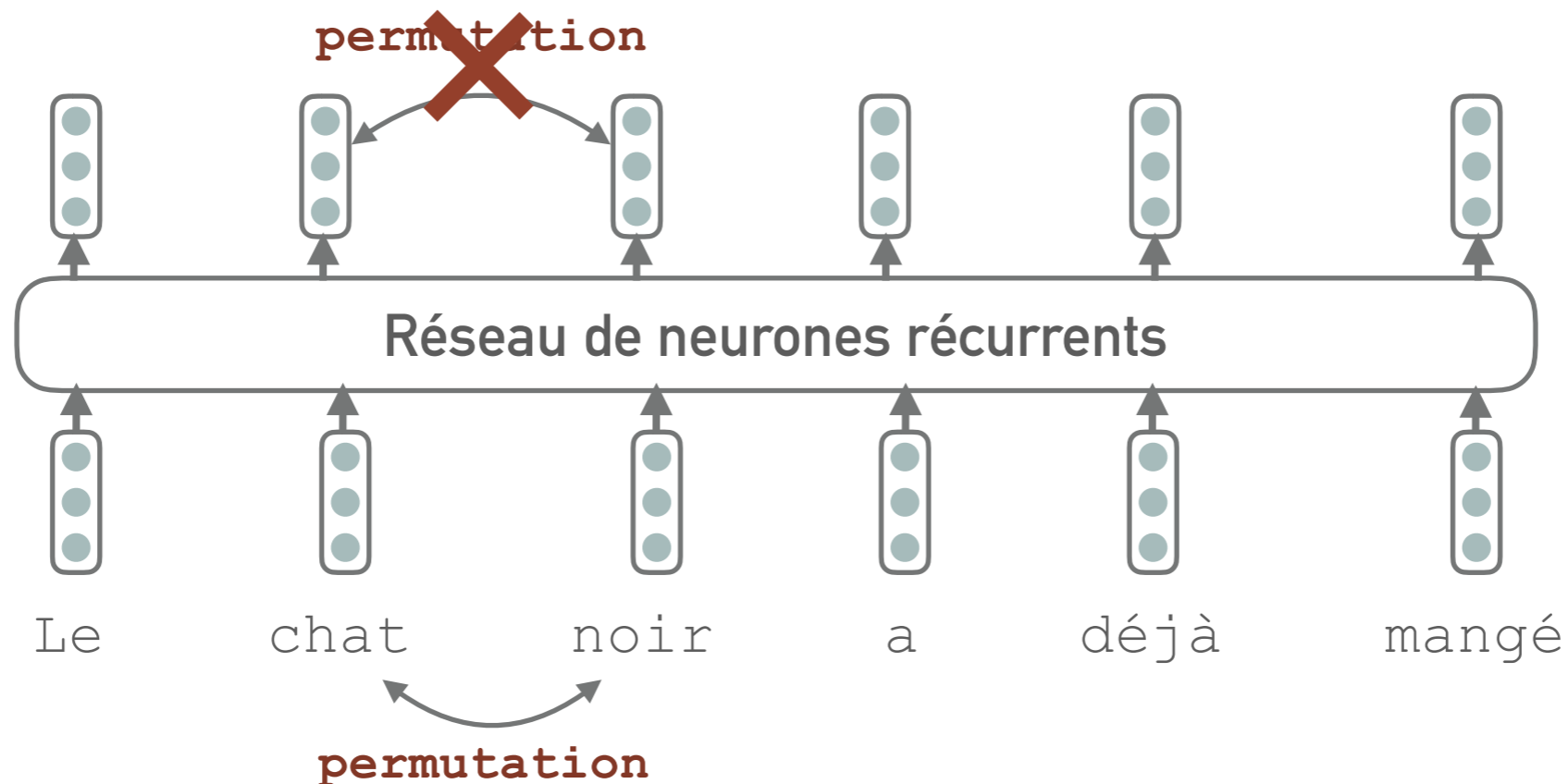
Diversité des ordres des mots dans différentes langues :

- SVO, SOV, VSO, ...
- Adjectif avant et/ou après le nom
- ...

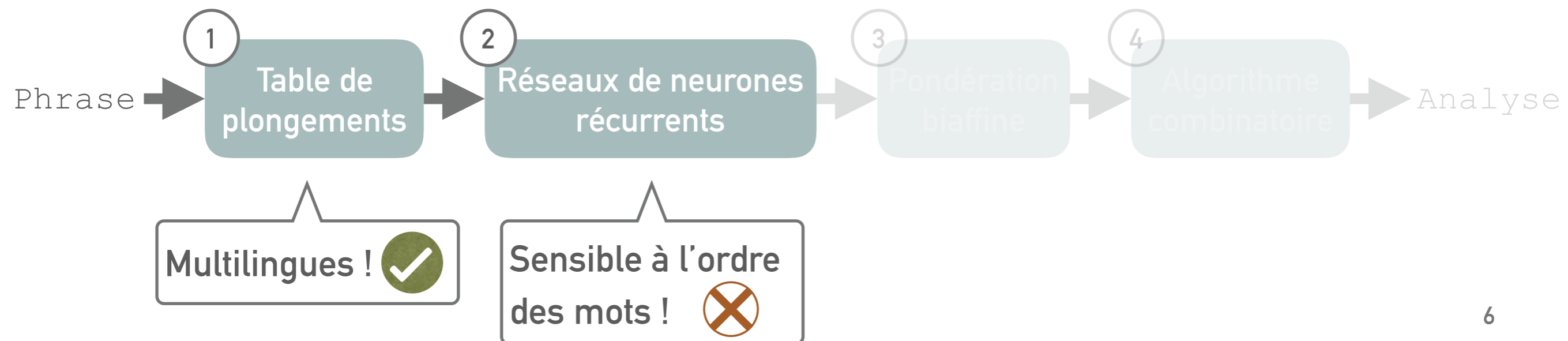


Non équivariance des réseaux de neurones utilisés en TAL

- Réseaux de neurones récurrents (p. ex. LSTM) : par essence
- Architecture attentionnelle : plongements de position



MÉTHODOLOGIE PROPOSÉE



MÉTHODOLOGIE PROPOSÉE

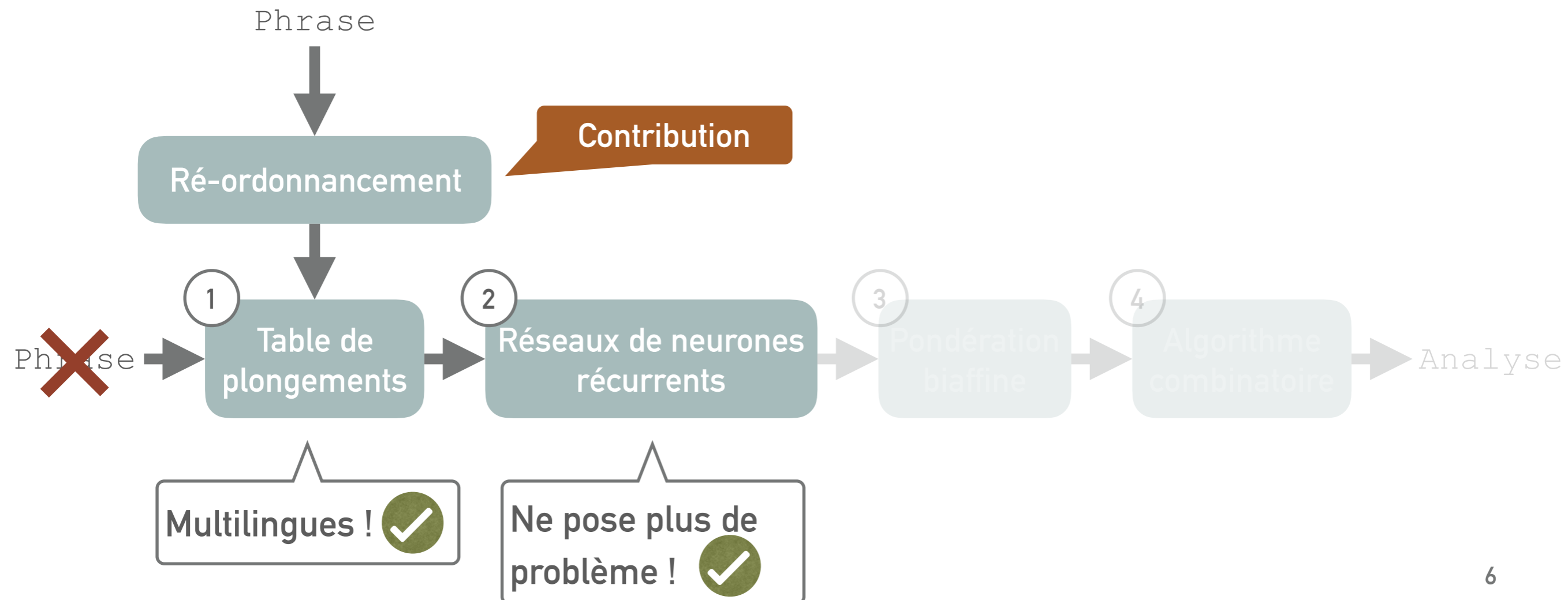
Amélioration d'un analyseur cross-lingue par ré-ordonnancement

- Apprentissage : modèle d'ordonnancement des mots sur la langue source
- Décodage : modèle de ré-ordonnancement des mots sur les langues cibles

Pas besoin de données alignées !

Méthodologie

- Utilisation d'un pipeline d'analyse syntaxique existant
- Pas de ré-entraînement des étapes 2 et 3 pour ne pas biaiser les résultats en notre faveur



ORDONNANCEMENT : MODÉLISATION

Notations

- \mathbf{w} : phrase de longueur n
- $\pi : \{1 \dots n\} \rightarrow \{1 \dots n\}$: ordonnancement sur la phrase, fonction bijective

Modèle bi-gramme d'ordonnancement

$$s(\mathbf{w}, \pi, \theta) = a_{w_{\pi(1)}} + b_{w_{\pi(n)}} + \sum_{i=1}^{n-1} D_{w_{\pi(i)}, w_{\pi(i+1)}}$$

Score d'une phrase
dans l'ordre π

Paramètres calculés
par un réseaux de

$$\theta = \langle \mathbf{a}, \mathbf{b}, \mathbf{D} \rangle \in \Theta$$

Le chat noir a déjà mangé
 $a_{mangé}$ $D_{chat, noir}$ $b_{mangé}$

Pondération neuronale

Réseau de neurones pour la régression prenant en entrée :

- Parties du discours « universelles »
- Plongement lexicaux multilingues

ORDONNANCEMENT : PROBLÈME D'APPRENTISSAGE

Objectif d'apprentissage : maximisation de la log-probabilité des données

Soit \mathcal{D} un jeu de données de référence, c-à-d que $\forall \langle \mathbf{w}, \bar{\pi} \rangle \in \mathcal{D}$:

- \mathbf{w} : phrase
- $\bar{\pi}$: ordonnancement de référence sur \mathbf{w}

$$\theta^* = \operatorname{argmax}_{\theta \in \Theta} \sum_{\langle \mathbf{w}, \bar{\pi} \rangle \in \mathcal{D}} \log \mathbb{P}(\bar{\pi} | \mathbf{w}, \theta)$$

avec \mathbb{P} une distribution de Boltzmann ou « softmax » :

$$\mathbb{P}(\pi | \mathbf{w}, \theta) = \exp(\mathbf{s}(\mathbf{w}, \pi, \theta) - \mathbf{c}(\mathbf{w}, \theta)) \quad \text{avec}$$

$$\mathbf{c}(\mathbf{w}, \theta) = \log \sum_{\pi \in \Pi} \exp(\mathbf{s}(\mathbf{w}, \pi, \theta))$$

Fonction de log-partition

ORDONNANCEMENT : PROBLÈME D'APPRENTISSAGE

Objectif d'apprentissage : maximisation de la log-probabilité des données

Soit \mathcal{D} un jeu de données de référence, c-à-d que $\forall \langle \mathbf{w}, \bar{\pi} \rangle \in \mathcal{D}$:

- \mathbf{w} : phrase
- $\bar{\pi}$: ordonnancement de référence sur \mathbf{w}

$$\theta^* = \operatorname{argmax}_{\theta \in \Theta} \sum_{\langle \mathbf{w}, \bar{\pi} \rangle \in \mathcal{D}} \log \mathbb{P}(\bar{\pi} | \mathbf{w}, \theta)$$

avec \mathbb{P} une distribution de Boltzmann ou « softmax » :

$$\mathbb{P}(\pi | \mathbf{w}, \theta) = \exp(\mathbf{s}(\mathbf{w}, \pi, \theta) - \mathbf{c}(\mathbf{w}, \theta)) \quad \text{avec} \quad \mathbf{c}(\mathbf{w}, \theta) = \log \sum_{\pi \in \Pi} \exp(\mathbf{s}(\mathbf{w}, \pi, \theta))$$

Fonction de log-partition

Optimisation par remontée de gradient

$$\theta^{(t+1)} = \theta^{(t)} + \epsilon \sum_{\langle \mathbf{w}, \bar{\pi} \rangle \in \mathcal{D}} \nabla_{\theta^{(t)}} \log \mathbb{P}(\bar{\pi} | \mathbf{w}, \theta^{(t)}),$$

où le gradient de la log-probabilité peut se ré-écrire en :

$$\nabla_{\theta} \log \mathbb{P}(\bar{\pi} | \mathbf{w}, \theta) = \nabla_{\theta} \mathbf{s}(\mathbf{w}, \bar{\pi}, \theta) - \nabla_{\theta} \mathbf{c}(\mathbf{w}, \theta)$$

Intractable



ORDONNANCEMENT : APPRENTISSAGE STOCHASTIQUE

Ré-écriture du gradient de la log-partition

$$\nabla_{\theta} c(\mathbf{w}, \theta) = \nabla_{\theta} \log \sum_{\pi \in \Pi} \exp(\mathbf{s}(\mathbf{w}, \pi; \theta))$$

ORDONNANCEMENT : APPRENTISSAGE STOCHASTIQUE

Ré-écriture du gradient de la log-partition

$$\begin{aligned}\nabla_{\theta} c(\mathbf{w}, \theta) &= \nabla_{\theta} \log \sum_{\pi \in \Pi} \exp(s(\mathbf{w}, \pi; \theta)) \\ &= \frac{1}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \sum_{\pi} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta))\end{aligned}$$

gradient du logarithme

ORDONNANCEMENT : APPRENTISSAGE STOCHASTIQUE

Ré-écriture du gradient de la log-partition

$$\begin{aligned}\nabla_{\theta} c(\mathbf{w}, \theta) &= \nabla_{\theta} \log \sum_{\pi \in \Pi} \exp(s(\mathbf{w}, \pi; \theta)) \\ &= \frac{1}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \sum_{\pi} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta)) \\ &= \sum_{\pi} \frac{\exp(s(\mathbf{w}, \pi, \theta))}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta))\end{aligned}$$


gradient du logarithme

gradient de l'exponentielle

Distribution de Boltzmann \mathbb{P}

ORDONNANCEMENT : APPRENTISSAGE STOCHASTIQUE

Ré-écriture du gradient de la log-partition

$$\begin{aligned}\nabla_{\theta} c(\mathbf{w}, \theta) &= \nabla_{\theta} \log \sum_{\pi \in \Pi} \exp(s(\mathbf{w}, \pi; \theta)) \\ &= \frac{1}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \sum_{\pi} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta)) \\ &= \sum_{\pi} \frac{\exp(s(\mathbf{w}, \pi, \theta))}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta)) \\ &= \mathbb{E}_{\Pi | \mathbf{w}, \theta} \left[\nabla_{\theta} s(\mathbf{w}, \pi, \theta) \right]\end{aligned}$$


gradient du logarithme

gradient de l'exponentielle

Ré-écriture sous forme d'une espérance

ORDONNANCEMENT : APPRENTISSAGE STOCHASTIQUE

Ré-écriture du gradient de la log-partition

$$\begin{aligned}\nabla_{\theta} c(\mathbf{w}, \theta) &= \nabla_{\theta} \log \sum_{\pi \in \Pi} \exp(s(\mathbf{w}, \pi; \theta)) \\ &= \frac{1}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \sum_{\pi} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta)) \\ &= \sum_{\pi} \frac{\exp(s(\mathbf{w}, \pi, \theta))}{\sum_{\pi' \in \Pi} \exp(s(\mathbf{w}, \pi', \theta))} \nabla_{\theta} \exp(s(\mathbf{w}, \pi, \theta)) \\ &= \mathbb{E}_{\Pi | \mathbf{w}, \theta} \left[\nabla_{\theta} s(\mathbf{w}, \pi, \theta) \right]\end{aligned}$$

gradient du logarithme

gradient de l'exponentielle

Ré-écriture sous forme d'une espérance

Méthode de Monte-Carlo

Approximation de l'espérance grâce à k échantillons de \mathbb{P} :

$$\mathbb{E}_{\Pi | \mathbf{w}, \theta} \left[\nabla_{\theta} s(\mathbf{w}, \pi, \theta) \right] \simeq \frac{1}{k} \sum_{i=1}^k \nabla_{\theta} s(\mathbf{w}, \pi^{(i)}, \theta) \quad \text{avec} \quad \pi^{(i)} | \mathbf{w}, \theta \sim \mathbb{P}, \quad \forall i \in \{1, \dots, k\}$$

Algorithme d'échantillonnage de Metropolis-Hastings

Soit $\mathbb{Q}(\Pi | \Pi)$ un noyau de transition exprimant la probabilité de transformer une permutation π en une permutation π' , une séquence de permutations $\pi^{(1)} \dots \pi^{(t)}$ est construite comme suit :

1. $\pi' | \pi^{(t)} \sim \mathbb{Q}$
2. $u \sim \mathcal{U}(0,1)$ avec $\mathcal{U}(0,1)$ la distribution uniforme entre 0 et 1

$$3. \pi^{(t+1)} = \begin{cases} \pi' & \text{si } u \leq \min \left(1, \frac{\mathbb{P}(\pi' | \mathbf{w})}{\mathbb{P}(\pi^{(t)} | \mathbf{w})} \times \frac{\mathbb{Q}(\pi^{(t)} | \pi')}{\mathbb{Q}(\pi' | \pi^{(t)})} \right), \\ \pi^{(t)} & \text{sinon.} \end{cases}$$

Ne nécessite pas de calculer la log-partition !

Transition 2-OPT

Implémentation efficace

- Noyau de transition symétrique, c-à-d que $\mathbb{Q}(\pi^{(t)} | \pi') = \mathbb{Q}(\pi' | \pi^{(t)})$
- Ré-écriture du rapport entre deux distributions de Boltzmann :

$$\frac{\mathbb{P}(\pi')}{\mathbb{P}(\pi^{(t)})} = \frac{\exp(s(\mathbf{w}, \pi', \theta) - c(\mathbf{w}, \theta))}{\exp(s(\mathbf{w}, \pi^{(t)}, \theta) - c(\mathbf{w}, \theta))} = \frac{\exp(s(\mathbf{w}, \pi', \theta))}{\exp(s(\mathbf{w}, \pi^{(t)}, \theta))}$$

PROBLÈME DE RÉ-ORDONNANCEMENT

Calcul de l'ordonnement le plus probable

Équivalent au problème du voyageur du commerce => NP-complet

$$\pi^* = \underset{\pi \in \Pi}{\operatorname{argmax}} \mathbb{P}(\pi \mid \mathbf{w}, \theta) = \underset{\pi \in \Pi}{\operatorname{argmax}} \mathbf{s}(\mathbf{w}, \pi, \theta)$$

PROBLÈME DE RÉ-ORDONNANCEMENT

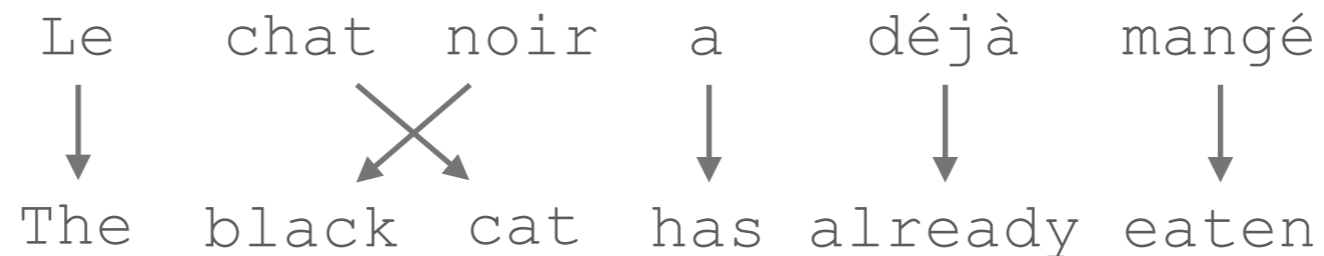
Calcul de l'ordonnement le plus probable

Équivalent au problème du voyageur du commerce => NP-complet

$$\pi^* = \operatorname{argmax}_{\pi \in \Pi} \mathbb{P}(\pi | \mathbf{w}, \theta) = \operatorname{argmax}_{\pi \in \Pi} \mathbf{s}(\mathbf{w}, \pi, \theta)$$

Problème de ré-ordonnement

- Intuition : étant donné l'ordre dans la langue source, la plupart des permutations à réaliser sont « locales »



- Solution : explorer un espace de recherche dépendant de l'ordre de la langue source grâce à des algorithmes de ré-ordonnement tractables

Espace de recherche des « Inversion Transduction Grammars » (ITG)

- L'ordre de deux segments contigus peut être inversé
- Cette transformation peut être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » CYK
- Complexité temporelle : $\mathcal{O}(n^6)$

私に 黒い 犬が 見えます ⇒
(je) (noir) (chien) (vois)

Espace de recherche des « Inversion Transduction Grammars » (ITG)

- L'ordre de deux segments contigus peut être inversé
- Cette transformation peut être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » CYK
- Complexité temporelle : $\mathcal{O}(n^6)$

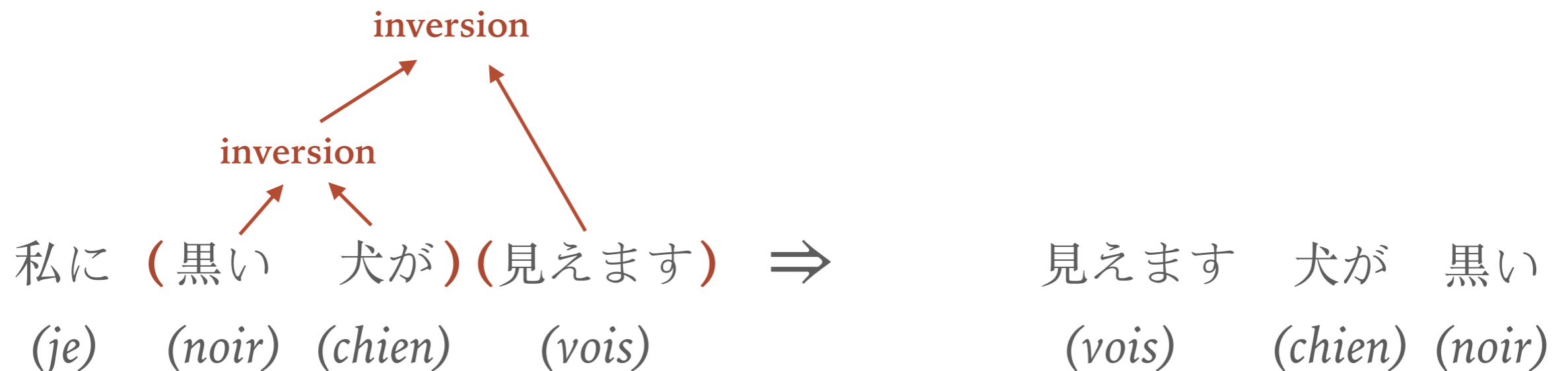


Espace de recherche des « Inversion Transduction Grammars » (ITG)

- L'ordre de deux segments contigus peut être inversé
- Cette transformation peut être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » CYK
- Complexité temporelle : $\mathcal{O}(n^6)$

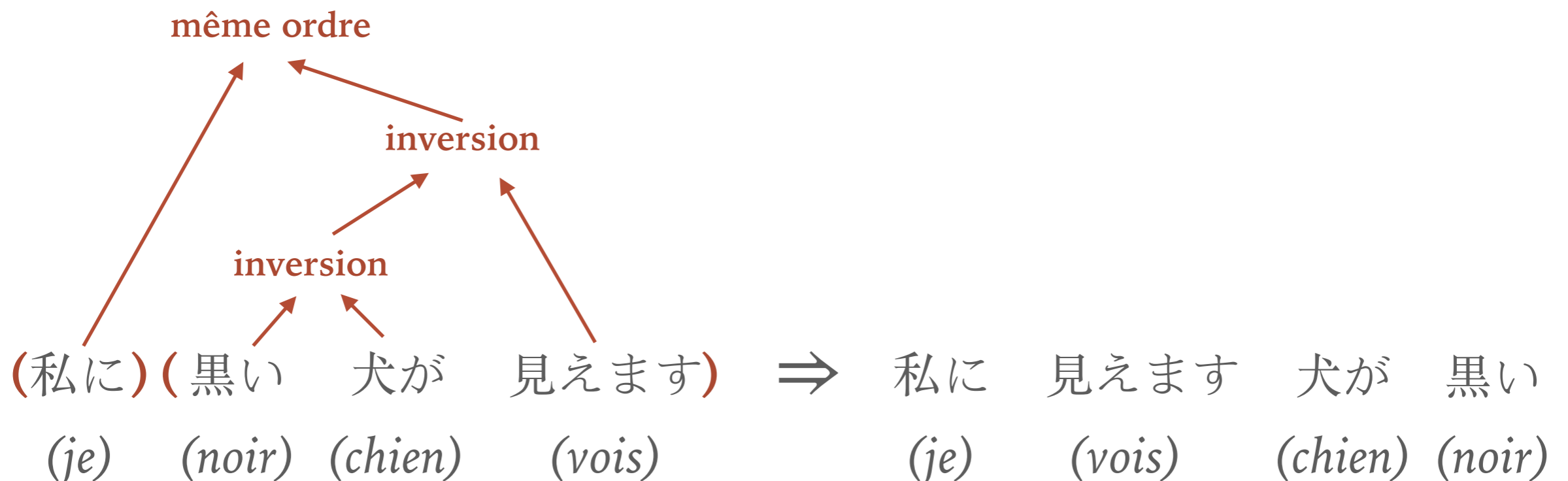


Espace de recherche des « Inversion Transduction Grammars » (ITG)

- L'ordre de deux segments contigus peut être inversé
- Cette transformation peut être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » CYK
- Complexité temporelle : $\mathcal{O}(n^6)$



Espace de recherche des ITG sans récursion

- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

Je l' ai mangé avec dégoût ⇒

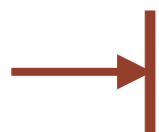
Espace de recherche des ITG sans récursion

- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

Je l' ai mangé avec dégoût \Rightarrow



Espace de recherche des ITG sans récursion

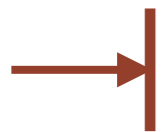
- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

avancer d'un segment
sans changement d'ordre

(Je) l' ai mangé avec dégoût ⇒ Je



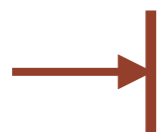
Espace de recherche des ITG sans récursion

- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

Je l' ai mangé avec dégoût \Rightarrow *Je*



Espace de recherche des ITG sans récursion

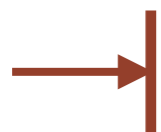
- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

avancer de deux segments
avec changement d'ordre

Je (l') (ai mangé) avec dégoût \Rightarrow Je ai mangé l'



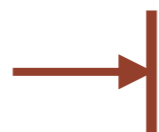
Espace de recherche des ITG sans récursion

- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

Je l' ai mangé avec dégoût \Rightarrow Je ai mangé l'



Espace de recherche des ITG sans récursion

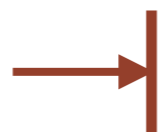
- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

avancer d'un segment
sans changement d'ordre

Je l' ai mangé (avec) dégoût \Rightarrow *Je ai mangé l' avec*



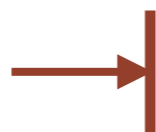
Espace de recherche des ITG sans récursion

- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

Je l' ai mangé avec dégoût \Rightarrow Je ai mangé l' avec



Espace de recherche des ITG sans récursion

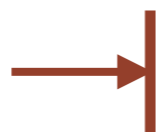
- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

avancer d'un segment
sans changement d'ordre

Je l' ai mangé avec (dégoût) \Rightarrow Je ai mangé l' avec dégoût



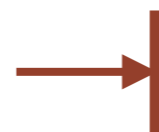
Espace de recherche des ITG sans récursion

- L'ordre de deux segments contigus peut être inversé
- Cette transformation ne peut pas être appliquée de façon récursive

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-Markov
- Complexité temporelle : $\mathcal{O}(n^4)$

Je l' ai mangé avec dégoût \Rightarrow Je ai mangé l' avec dégoût



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

Espace de recherche des ITG avec inversion de « singletons » uniquement

- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

Je l' envoie demain \Rightarrow

RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

Espace de recherche des ITG avec inversion de « singletons » uniquement

- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

Je l' envoie demain \Rightarrow



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

Espace de recherche des ITG avec inversion de « singletons » uniquement

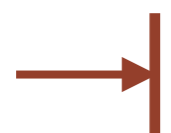
- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

avancer d'un segment
sans changement d'ordre

(Je) l' envoie demain \Rightarrow Je



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

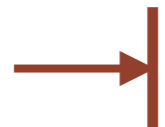
Espace de recherche des ITG avec inversion de « singletons » uniquement

- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

Je l' envoie demain \Rightarrow *Je*



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

Espace de recherche des ITG avec inversion de « singletons » uniquement

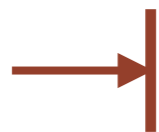
- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

avancer de deux segments
avec changement d'ordre

Je (l') (envoie) demain \Rightarrow *Je envoie l'*



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

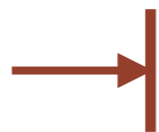
Espace de recherche des ITG avec inversion de « singletons » uniquement

- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

Je l' envoie demain \Rightarrow *Je envoie l'*



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

Espace de recherche des ITG avec inversion de « singletons » uniquement

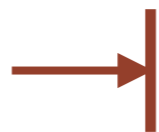
- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

avancer d'un segment
sans changement d'ordre

Je l' envoie (demain) ⇒ Je envoie l' demain



RÉ-ORDONNANCEMENT : ALGORITHME EN $O(N)$

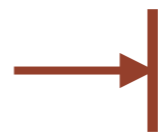
Espace de recherche des ITG avec inversion de « singletons » uniquement

- L'ordre de deux segments contigus peuvent être inversé
- Seuls des segments de longueur 1 peuvent être inversés

Algorithme

- Type d'algorithme : programmation dynamique « façon » semi-markov
- Complexité temporelle : $\mathcal{O}(n)$

Je l' envoie demain \Rightarrow *Je envoie l' demain*



RÉSULTATS EXPÉRIMENTAUX

Configuration expérimentale

- Analyseur de Ahmad et al. (2019)
- Langue source : anglais
- Langue cible : français (FTB : 2541 phrases, FQB : 2289 phrases)
- Métrique : UAS/LAS, sans la ponctuation

Résultats sur plus de 20 langues dans l'article !

Décodeur fondé sur les graphes

Données	Non perm.
FTB	66,9 / 60,7
FQB	75,5 / 66,8

Décodeur par transition

Données	Non perm.
FTB	64,6 / 58,1
FQB	75,9 / 68,0

Baselines

RÉSULTATS EXPÉRIMENTAUX

Configuration expérimentale

- Analyseur de Ahmad et al. (2019)
- Langue source : anglais
- Langue cible : français (FTB : 2541 phrases, FQB : 2289 phrases)
- Métrique : UAS/LAS, sans la ponctuation

Résultats sur plus de 20 langues dans l'article !

Décodeur fondé sur les graphes

Avec ré-ordonnement

Données	Non perm.	$O(n^6)$	$O(n^4)$	$O(n)$
FTB	66,9 / 60,7	56,5 / 51,2	56,5 / 51,2	68,0 / 61,4
FQB	75,5 / 66,8	69,7 / 61,2	69,7 / 61,2	77,3 / 68,5

Décodeur par transition

Données	Non perm.	$O(n^6)$	$O(n^4)$	$O(n)$
FTB	64,6 / 58,1	64,4 / 57,5	56,9 / 51,0	68,6 / 61,4
FQB	75,9 / 68,0	73,9 / 65,9	70,6 / 62,5	79,1 / 71,1

CONCLUSION

Contributions

- Méthode pour l'analyse en dépendances cross-lingue sans données parallèles
- Présentation unifiée de trois algorithmes précédemment présentés dans la littérature
- Comparaison de leurs performances sur une tâche cible

Question ouverte

Pourquoi le ré-ordonnanceur le plus simple donne-t-il les meilleurs résultats ?

