HAL open science

Ré-ordonnancement via programmation dynamique pour l'adaptation cross-lingue d'un analyseur en dépendances

Nicolas Devatine, Caio Corro, François Yvon

To cite this version:

Nicolas Devatine, Caio Corro, François Yvon. Ré-ordonnancement via programmation dynamique pour l'adaptation cross-lingue d'un analyseur en dépendances. 29ème Conférence sur le Traitement Automatique des Langues Naturelles (TALN 2022), Association pour le Traitement Automatique des Langues (ATALA); Laboratoire d'Informatique et Systèmes (LIS); Laboratoire Informatique d'Avignon (LIA), Jun 2022, Avignon, France. pp.183-197. hal-03701508

HAL Id: hal-03701508

https://hal.science/hal-03701508

Submitted on 24 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Ré-ordonnancement via programmation dynamique pour l'adaptation cross-lingue d'un analyseur en dépendances

Nicolas Devatine[1*] Caio Corro[2] François Yvon[2]
(1) Université Toulouse 3, IRIT, 31062, Toulouse, France
(2) Université Paris-Saclay, CNRS, LISN, 91400, Orsay, France
nicolas.devatine@irit.fr, {caio.corro,francois.yvon}@lisn.upsaclay.fr

RÉSUMÉ

Cet article s'intéresse au transfert cross-lingue d'analyseurs en dépendances et étudie des méthodes pour limiter l'effet potentiellement néfaste pour le transfert de divergences entre l'ordre des mots dans les langues source et cible. Nous montrons comment apprendre et implémenter des stratégies de réordonnancement, qui, utilisées en prétraitement, permettent souvent d'améliorer les performances des analyseurs dans un scénario de transfert « zero-shot ».

ABSTRACT

Sentence reordering via dynamic programming for cross-lingual dependency parsing

This paper studies cross-lingual transfer for dependency parsers, with the aim to study ways to mitigate word order differences between the source and the target language in transfer. We show how to learn and implement reordering strategies for target words, that, when used in pre-processing, allow us to improve the parsing accuracy in a zero-shot transfer scenario.

MOTS-CLÉS : Analyse en dépendances, transfert cross-lingue, programmation dynamique.

KEYWORDS: Dependency parsing, cross-lingual transfer, dynamic programming.

1 Introduction

Dans cet article, nous nous intéressons au transfert entre langues d'analyseurs syntaxiques : à partir d'un modèle appris sur une ou plusieurs langues sources, nous souhaitons pouvoir analyser des textes dans une ou plusieurs langues cibles.

Pour rendre ce transfert possible, il est nécessaire de disposer de représentations similaires, ou du moins compatibles, dans les langues sources et cibles, qui s'appuient par exemple sur :

— des parties du discours « universelles » (Petrov *et al.*, 2012) ;
— des plongements de mots multilingues alignés, c-à-d tels que les mots ayant des « rôles similaires » dans leur langue respective ont des plongements proches.

Nous nous intéressons ici aux variations d'ordre des mots dans les langues sources et cibles et à la manière dont elles influent sur le transfert d'analyseurs entre langues. Ce problème est étudié en particulier dans (Søgaard, 2011; Aufrant *et al.*, 2016) qui montrent que les variations dans l'ordre des mots induisent des divergences dans la distribution des représentations utilisées pour prédire les attachements syntaxiques. Nous le revisitons pour des modèles neuronaux en étudiant les questions

*. Ce travail a été réalisé lorsque Nicolas Devatine était en stage de master au LISN.

suivantes :

— les variations d'ordre des mots est-il un problème pour les modèles neuronaux ?
— comment améliorer leur robustesse à cette variation dans un cadre cross-lingue ?

Nous nous focalisons sur les architectures récurrentes de type LSTM (Hochreiter & Schmidhuber, 1997). Dans ces réseaux, contrairement aux modèles attentionnels (Vaswani *et al.*, 2017), l'ordre des mots est encodé « par essence », ce qui peut nuire au transfert entre langues (Ahmad *et al.*, 2019).

Nous montrons qu'un pré-traitement consistant à *permuter les mots dans des phrases en langue cible de manière à reproduire l'ordre des mots en langue source* peut permettre d'améliorer les performances du modèle, même lorsque les langues source et cible sont typologiquement proches. Qui plus est, nous proposons une approche pour apprendre à réaliser cette permutation sans avoir accès à des données parallèles. En résumé, nos principales contributions sont les suivantes :

— une méthode pour entraîner un modèle d'ordonnancement sans texte parallèle ;
— une présentation unifiée de différents algorithmes de ré-ordonnancement ;
— des expériences impliquant une vingtaine de langues qui s'appuient sur un analyseur syntaxique préentraîné dont les paramètres sont disponibles publiquement, ce qui permet de reproduire nos résultats.

Les résultats expérimentaux soulèvent de nombreuses questions sur le fonctionnement des analyseurs syntaxiques cross-lingues et ouvrent diverses pistes de recherche. Ils suggèrent en particulier que les difficultés de l'analyse cross-lingue ne proviennent pas uniquement des limites des plongements multilingues, mais que le développement d'architectures neuronales robustes aux variations typologiques est une piste de recherche importante.

2 Modèles d'ordonnancement

Pour simplifier les notations, nous supposons que toutes les phrases ont une longueur de n mots. Soit $\boldsymbol{w} \in V^n$ un vecteur de n mots tirés dans un vocabulaire V. Dans cette section, le vecteur peut être interprété comme un sac de mots, mais nous gardons la notation $\boldsymbol{w}$ pour des raisons de cohérence avec la suite de l'article. Un ordonnancement de $\boldsymbol{w}$ est représenté par une fonction bijective $\pi : \{1 \dots n\} \to \{1 \dots n\}$ où $\pi(i)$ donne la position du mot w_i dans l'ordonnancement. L'ensemble de toutes les fonctions d'ordonnancement est noté Π.

Notre objectif est de construire un modèle capable d'ordonner les mots de $\boldsymbol{w}$ tout en respectant l'ordre dominant des mots d'une langue. Pour cela, nous utilisons un modèle de score bigramme $s : V^n \times \Pi \times \Theta \to \mathbb{R}$ qui assigne un score à chaque ordonnancement possible selon :

$$s(\boldsymbol{w}, \pi, \theta) = a_{w_{\pi(1)}} + b_{w_{\pi(n)}} + \sum_{i=1}^{n-1} D_{w_{\pi(i)}, w_{\pi(i+1)}}, \quad \theta = \langle \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{D} \rangle \in \Theta,$$

où $\boldsymbol{a} \in \mathbb{R}^{|V|}$ (resp. $\boldsymbol{b} \in \mathbb{R}^{|V|}$) donne le score associé au premier (resp. dernier) mot de la phrase et $\boldsymbol{D} \in \mathbb{R}^{|V| \times |V|}$ est une matrice de scores bigrammes, paramétrée par un réseau de neurones qui calcule un score à partir des représentations numériques des mots en entrée. Une distribution sur les ordonnancements de $\boldsymbol{w}$ peut être définie en utilisant la distribution de Boltzmann ou « *softmax* » :

$$\mathbb{P}(\pi | \boldsymbol{w}, \theta) = \exp(s(\boldsymbol{w}, \pi, \theta) - c(\boldsymbol{w}, \theta)),$$

avec $c : V^n \times \Theta \to \mathbb{R}$ la fonction de log-partition qui normalise la distribution (Wainwright *et al.*, 2008, Section 3.2), définie comme :

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

Deux problèmes d'optimisation sur ce modèle sont importants en pratique :

1. calculer l'ordonnancement le plus probable pour $\boldsymbol{w}$, problème équivalent à calculer l'ordonnancement de score maximal car $\mathbb{P}(\pi|\boldsymbol{w}, \theta) \propto \exp(s(\boldsymbol{w}, \pi, \theta))$:

$$\pi^* = \arg\max_{\pi \in \Pi} s(\boldsymbol{w}, \pi, \theta);$$

2. calculer les paramètres θ qui maximisent la log-vraisemblance d'un ensemble de couples $\langle \boldsymbol{w}, \bar{\pi} \rangle \in \mathcal{D}$, avec $\bar{\pi}$ un ordonnancement de référence de $\boldsymbol{w}$:

$$\theta^* = \arg\max_{\theta \in \Theta} \sum_{\langle \boldsymbol{w}, \bar{\pi} \rangle \in \mathcal{D}} \log \mathbb{P}(\bar{\pi}|\boldsymbol{w}, \theta).$$

Une approche standard pour le second problème est d'utiliser l'algorithme de remontée du gradient, qui calcule une séquence de paramètres $\{\theta^{(t)}\}_{t=0}^{T}$ comme suit :

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

avec ϵ la taille du pas. Le gradient de la log-probabilité est défini comme :

$$\nabla_\theta \log \mathbb{P}(\bar{\pi}|\boldsymbol{w}, \theta) = \nabla_\theta s(\boldsymbol{w}, \bar{\pi}, \theta) - \underbrace{\nabla_\theta c(\boldsymbol{w}, \theta)}_{\text{Intractable}}.$$

Dans cette formulation, le calcul du gradient de la log-partition implique une somme sur tous les ordonnancements possibles, qui ne peut être réalisée efficacement. Ce problème est étroitement lié au problème de comptage dans le cas du voyageur de commerce, problème dans $\#P$ (Hemaspaandra & Ogihara, 2001, Appendice A.10).

3 Apprentissage par échantillonnage

Pour approximer le gradient de la fonction de log-partition, nous réécrivons ce gradient afin d'utiliser une approximation par échantillonnage (Koller & Friedman, 2009, Section 20.5.1.3) :

$$\begin{aligned}
\nabla_\theta c(\boldsymbol{w}, \theta) &= \nabla_\theta \log \sum_{\pi} \exp(s(\boldsymbol{w}, \pi, \theta)) \\
&= \frac{1}{\sum_\pi \exp(s(\boldsymbol{w}, \pi, \theta))} \sum_{\pi} \nabla_\theta \exp(s(\boldsymbol{w}, \pi, \theta)) \\
&= \sum_{\pi} \frac{\exp(s(\boldsymbol{w}, \pi, \theta))}{\sum_{\pi'} \exp(s(\boldsymbol{w}, \pi', \theta))} \nabla_\theta s(\boldsymbol{w}, \pi, \theta), \\
&= \sum_{\pi} \mathbb{P}(\pi|\boldsymbol{w}, \theta) \nabla_\theta s(\boldsymbol{w}, \pi, \theta)
\end{aligned}$$

que l'on peut récrire comme une espérance :

$$= \mathbb{E}_{\Pi|\boldsymbol{w},\theta}[\nabla_\theta s(\boldsymbol{w}, \pi, \theta)].$$

Cette espérance peut être approchée par un estimateur de Monte-Carlo, c'est-à-dire comme la moyenne empirique de k échantillons :

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

Même s'il n'est pas possible d'échantillonner directement dans la distribution $\mathbb{P}(\Pi|\boldsymbol{w}, \boldsymbol{\theta})$, nous pouvons utiliser les méthodes de Monte-Carlo par chaînes de Markov, ici l'algorithme de Metropolis-Hastings (Metropolis *et al.*, 1953; Hastings, 1970). 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 $\pi^{(t)}$ 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 continue uniforme entre 0 et 1 ;
3. $\pi^{(t+1)} = \begin{cases} \pi' & \text{if } u \leq \min\left(1, \frac{\mathbb{P}(\pi')}{\mathbb{P}(\pi^{(t)})} \times \frac{\mathbb{Q}(\pi^{(t)}|\pi')}{\mathbb{Q}(\pi'|\pi^{(t)})}\right), \\ \pi^{(t)} & \text{sinon.} \end{cases}$

Si la distribution de transition est symétrique, c-à-d que $\forall \pi', \pi^{(t)} \in \Pi : \mathbb{Q}(\pi'|\pi^{(t)}) = \mathbb{Q}(\pi^{(t)}|\pi')$, le second argument du minimum de l'étape 2 peut être réécrit comme :

$$\begin{aligned} \frac{\mathbb{P}(\pi')}{\mathbb{P}(\pi^{(t)})} \times \frac{\mathbb{Q}(\pi^{(t)}|\pi')}{\mathbb{Q}(\pi'|\pi^{(t)})} &= \frac{\mathbb{P}(\pi')}{\mathbb{P}(\pi^{(t)})} \\ &= \frac{\exp(s(\boldsymbol{w}, \pi', \theta) - c(\boldsymbol{w}, \theta))}{\exp(s(\boldsymbol{w}, \pi^{(t)}, \theta) - c(\boldsymbol{w}, \theta))}. \end{aligned}$$

Notons que les deux termes intractables $c(\boldsymbol{w}, \theta)$ s'annulent, donc :

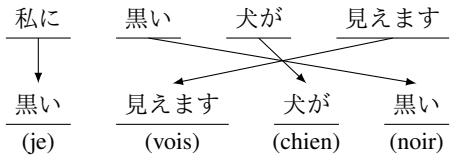
$$= \exp(s(\boldsymbol{w}, \pi', \theta) - s(\boldsymbol{w}, \pi^{(t)}, \theta)).$$

Cette variante est parfois appelée algorithme de Metropolis.

Nous utilisons le noyau de transition 2-OPT, initialement proposée comme heuristique locale pour résoudre le problème du voyageur de commerce (Croes, 1958). Pour l'échantillonnage $\pi'|\pi^{(t)} \sim \mathbb{Q}$, deux indices i et j dans $\{1, ..., n\}$ sont choisis aléatoirement puis la permutation $\pi^{(t)} = [\pi_1, ..., \pi_i, \pi_{i+1}, ..., \pi_j, \pi_{j+1}, ..., \pi_n]$ est remplacée par $[\pi_1, ..., \pi_i, \pi_j, ..., \pi_{i+1}, \pi_{j+1}, ..., \pi_n]$, avec inversion de l'ordre des éléments entre les indices $i + 1$ et j.

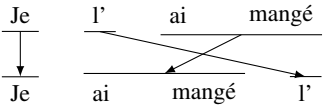
4 Algorithmes de réordonnancement

Le calcul de l'ordonnancement de poids maximum, voir Section 2, peut être réduit au problème du voyageur de commerce (Zaslavskiy *et al.*, 2009; Horvat & Byrne, 2014), un problème NP-difficile (Karp, 1972). De plus, sans aucune contrainte supplémentaire, le problème d'ordonnancement considère $\boldsymbol{w}$ comme un sac de mots, ce qui peut être problématique. Par exemple, supposons la phrase



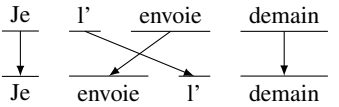
	Items	Règles
1.	$[2, 2, 2, 2]$	Axiome
2.	$[3, 3, 3, 3]$	Axiome
3.	$[2, 3, 3, 2]$	**(b)** avec 1 et 2
4.	$[4, 4, 4, 4]$	Axiome
5.	$[2, 4, 4, 2]$	**(b)** avec 3 et 4
6.	$[1, 1, 1, 1]$	Axiome
7.	$[1, 4, 1, 2]$	**(a)** avec 6 et 5

(a) Réordonnancement avec l'algorithme en $\mathcal{O}(n^6)$



	Items	Règles
1.	$[3, 3, 3, 3]$	Axiome
2.	$[3, 3, 4, 4]$	**(d)** avec 1
4.	$[1, 1, 1, 1]$	Axiome
5.	$[2, 2, 2, 2]$	Axiome
6.	$[1, 4, 1, 2]$	**(e)** avec 4, 5 et 2

(b) Réordonnancement avec l'algorithme en $\mathcal{O}(n^4)$



	Items	Règles
1.	$[1, 1, ?, 1]$	Axiome
2.	$[1, 3, ?, 2]$	**(h)** avec 1
3.	$[1, 4, ?, 2]$	**(g)** avec 2

(c) Réordonnancement avec l'algorithme en $\mathcal{O}(n)$

FIGURE 1 – Exemples de réordonnancements de phrases illustrant les différents algorithmes proposés.

suivante : « Le philosophe grec lit un livre ». Cette exemple est particulièrement intéressant car il est difficile de déterminer quel est l'adjectif et quel est le nom dans le premier groupe nominal, si cette différence est souhaitée (Jespersen, 1924, Chapitre 5). Si nous analysons cette phrase avec un modèle cross-lingue entraîné sur des corpus en anglais, l'ordre des mots dans un groupe nominal est déterminant : les adjectifs précèdent toujours les noms en anglais. Nous voulons donc, éventuellement, réordonner la phrase pour obtenir « Le grec philosophe lit un livre », mais en aucun cas le réordonnancement « Le philosophe lit un grec livre » n'est souhaité. Lors de l'analyse, nous voulons donc réduire l'espace de recherche des ordonnancements à la fois pour des raisons computationnelles et pour des raisons de cohérence.

De nombreux algorithmes de réordonnancement ont été proposés dans la littérature, par exemple dans le cadre du problème du voyageur du commerce (Bompadre & Orlin, 2005) et de la traduction automatique (TA) (Eisner & Tromble, 2006; Tromble & Eisner, 2009). Dans un contexte de TA statistique, la sélection d'un ensemble de réordonnancements possibles contraint l'espace de recherche du décodeur et a un effet direct sur les performances du système, voir par exemple (Zens & Ney, 2003; Pécheux *et al.*, 2016). Notre contribution, dans cette section, est une présentation de trois algorithmes de réordonnancement sous une forme unifiée en utilisant des règles de déduction (Pereira & Warren, 1983), une approche standard de présentation des algorithmes de programmation dynamique en traitement automatique des langues qui permet de facilement vérifier la spécification de l'algorithme et d'implémenter le calcul de différentes valeurs (maximum, k-meilleurs, fonction de partition, etc) (Goodman, 1999). Par « unifié », nous entendons que nous utilisons les mêmes éléments (*items*) pour les trois algorithmes, seules les règles de déduction changent.

Éléments. Sans perte de généralité, nous supposons que l'ordonnancement d'origine est la fonction identité sur {1, . . ., n}. Nos systèmes de déduction utilisent des éléments de forme $[i, j, s, t]$ où :

— i et j t.q. $1 \leq i \leq j \leq n$ identifient la séquence de mots de la phrase originale couverte par l'élément, c-à-d que l'élément couvre les mots $w_i, ..., w_j$;

— s t.q. $i \leq s \leq j$ indique que w_s est le mot le plus à gauche dans la permutation de $w_i, ..., w_j$;

— t t.q. $i \leq t \leq j$ indique que w_t est le mot le plus à droite dans la permutation de $w_i, ..., w_j$.

Les axiomes associés à ces règles sont tous les éléments de type $[i, i, i, i]$. Afin de prendre en compte les scores associés aux premier et dernier mots de la phrase, nous introduisons un élément de but $[1, n]$ qui ne peut être déduit que par :

$$\frac{[1, n, s, t]}{[1, n]} \; a_{w_s} + b_{w_t}$$

où la partie à droite de la règle d'inférence spécifie le score associé à cette règle.

Algorithme en $\mathcal{O}(n^6)$. Cet algorithme explore l'espace des permutations correspondant au voisinage des ITG (*Inversion Transduction Grammars*) (Wu, 1997). Il contient deux règles de déduction dont les antécédents correspondent à deux empans contigus de la phrase originale :

$$\textbf{(a)} \quad \frac{[i, k, s, u] \quad [k+1, j, v, t]}{[i, j, s, t]} \; D_{w_u, w_v}$$

$$\textbf{(b)} \quad \frac{[i, k, v, t] \quad [k+1, j, s, u]}{[i, j, s, t]} \; D_{w_u, w_v}$$

La règle **(a)** n'intervertit pas l'ordre des mots contenus dans les antécédents, la règle **(b)** les intervertit. Les deux règles sont décrites à l'aide de 7 variables correspondant à des indices positionnels, suggérant que la complexité de l'algorithme est de $\mathcal{O}(n^7)$. Cependant, la complexité peut être diminuée en utilisant une astuce similaire au « *hook trick* » des grammaires lexicalisées (Eisner & Satta, 1999, 2000), résultant en une complexité de $\mathcal{O}(n^6)$ (Eisner & Tromble, 2006). L'idée consiste à récrire chaque déduction en deux étapes. Pour cela, nous introduisons deux nouveaux type d'éléments que nous écrivons $[\rightarrow, i, k, s, v]$, avec $v > k$, et $[\leftarrow, i, k, u, t]$, avec $u > k$. La règle **(a)** peut alors se récrire comme :

$$\textbf{(a')} \quad \frac{[i, k, s, u]}{[\rightarrow, i, k, s, v]} \; D_{w_u, w_v}$$

$$\textbf{(a'')} \quad \frac{[\rightarrow, i, k, s, v] \quad [k+1, j, v, t]}{[i, j, s, t]}$$

Intuitivement, la règle **(a')** produit un item (incomplet) exprimant une possible extension de l'empan courant d'une position à droite, donnant lieu au bigramme w_u, w_v, la règle **(a'')** complète l'opération si l'item attendu à droite est disponible. De façon similaire, la règle **(b)** peut se récrire comme :

$$\textbf{(b')} \quad \frac{[i, k, v, t]}{[\leftarrow, i, k, u, t]} \; D_{w_u, w_v}$$

$$\textbf{(b'')} \quad \frac{[\leftarrow, i, k, u, t] \quad [k+1, j, s, u]}{[i, j, s, t]}$$

Les règles **(a')** et **(b')** possèdent 5 variables libres alors que les règles **(a'')** et **(b'')** ont 6 variables libres. On obtient donc une complexité temporelle en $\mathcal{O}(n^6)$.

Algorithme en $\mathcal{O}(n^4)$. Afin de réduire la complexité de l'algorithme précédent, il est possible d'interdire les réordonnancements « récursifs », c-à-d que si un empan contigu n'a pas les mêmes voisins dans la phrase réordonnée, alors tous les mots dans ce segment doivent être dans le même ordre que dans la phrase originale. Cet espace de recherche correspond au modèle MJ-1 de Kumar & Byrne (2005), à la différence que nous découpons et réordonnons les segments de façon jointe et non comme deux étapes successives. Avec cette contrainte, nous pouvons écrire un algorithme en programmation dynamique qui visite les empans de la phrase de gauche à droite et non des plus petits aux plus grands. Dans un sens, cet algorithme est plus proche des algorithmes d'inférence des modèles semi-Markov (Sarawagi & Cohen, 2004) que des algorithmes de type CYK. Les règles de déduction correspondantes sont :

$$\textbf{(c)} \quad \frac{[1, j, ?, t]}{[1, j+1, ?, j+1]} \; D_{w_t, w_{j+1}}$$

$$\textbf{(d)} \quad \frac{[i, j, i, j]}{[i, j+1, i, j+1]} \; D_{w_j, w_{j+1}}$$

$$\textbf{(e)} \quad \frac{[1, j, ?, t] \quad [j+1, k, j+1, k] \quad [k+1, l, k+1, l]}{[1, l, ?, k]} \; D_{w_t, w_{k+1}} + D_{w_l, w_{j+1}}$$

$$\textbf{(f)} \quad \frac{[1, k, 1, k] \quad [k+1, l, k+1, l]}{[1, l, ?, k]} \; D_{w_l, w_1}$$

Les indices indiqués comme « ? » ne sont pas importants pour l'algorithme et leur valeur n'impacte pas la complexité de l'algorithme car ceux-ci peuvent être récrits avec des éléments intermédiaires. La déduction **(c)** avance d'un mot dans la phrase, en ajoutant celui-ci à la fin de la permutation en cours. La déduction **(d)** est utilisée pour construire un sous-segment qui sera permuté. La déduction **(e)** avance de deux segments dans la phrase en permutant l'ordre de ceux-ci. La déduction **(f)** réalise une permutation de segments dont le premier antécédent commence au début de la phrase

Algorithme en $\mathcal{O}(n)$. Enfin, nous proposons un algorithme en temps linéaire qui n'autorise que des permutations d'un seul mot à la fois :

$$\textbf{(g)} \quad t \in \{j-1, j\} \; \frac{[1, j, ?, t]}{[1, j+1, ?, j+1]} \; D_{w_t, w_{j+1}}$$

$$\textbf{(h)} \quad \frac{[1, i, ?, t]}{[1, i+2, ?, i+1]} \; D_{w_t, w_{i+2}} + D_{w_{i+2}, w_{i+1}}$$

La déduction **(g)** est similaire à la règle **(c)** ci-dessus, avec une contrainte en plus. La déduction **(h)** avance de deux mots en échangeant leurs positions respectives. Afin d'autoriser la permutation des deux premiers mot de la phrase, nous ajoutons l'axiome $[1, 2, ?, 1]$ avec comme poids $D_{2,1}$.

Des exemples d'éxécution de ces trois algorithmes sont reproduits sur la Figure 1.

5 Expériences

Nous réalisons des expériences avec l'analyseur d'Ahmad *et al.* (2019), en utilisant les modèles distribués par les auteurs. Ces modèles sont entraînés sur des données de langue anglaise, sur la base de représentations associant les parties du discours et des plongements lexicaux multilingues. Nous nous focalisons uniquement sur les modèles utilisant des réseaux récurrents, et non sur les modèles reposant sur des architectures moins sensibles à l'ordre des mots comme les architectures de type *Transformer*, car pour ces dernières il ne fait pas sens de permuter l'ordre des mots. De plus, nous comparons les deux décodeurs implémentés par les auteurs, celui fondé sur les graphes et celui utilisant un système par transitions. Nous reportons les lecteurs et lectrices à Ahmad *et al.* (2019) pour les détails.

5.1 Détails d'implémentation

Nous utilisons trois perceptrons à une couche cachée de taille 4096 pour calculer les valeurs dans les vecteurs $\boldsymbol{a}$, $\boldsymbol{b}$ et la matrice $\boldsymbol{D}$. La fonction d'activation est la tangente hyperbolique. Nous utilisons en entré des perceptrons des représentations résultant de la concaténation du plongement FastText (fixé) et d'un plongement de partie du discours de dimension 128 (appris de bout-en-bout).

Pour l'apprentissage, nous utilisons la méthode Adam (Kingma & Ba, 2015) avec une taille de pas de $0,01$. Nous appliquons un *dropout* de $0,5$ sur les plongements lexicaux et de $0,3$ sur la couche cachée. Nous utilisons 1000 échantillons pour approximer la fonction de log-partition.

5.2 Résultats

Les résultats pour les langues pour lesquelles des plongements lexicaux FastText alignés (Joulin *et al.*, 2018) sont disponibles figurent dans le tableau 1. Une première observation est que dans notre configuration, la politique de réordonnancement « prudente » en $O(n)$ est systématiquement la meilleure et permet de surpasser (en moyenne) les analyseurs « non permutés », pour les deux parseurs considérés. De surcroît, certaines langues semblent beaucoup bénéficier de la permutation de l'entrée, notamment dans le cas du modèle par transitions. Par exemple, nous obtenons des gains de $+3.9$/$+3.6$ et $+3.1$/$+3.1$ pour le français et le polonais, respectivement. Ces résultats ouvrent de nouvelles questions de recherche : pourquoi n'observons-nous des améliorations que dans certaines langues ? Est-ce dû à des propriétés morpho-syntaxiques de ces langues ou à des problèmes liés à la qualité des plongements lexicaux FastText ? Une autre question porte sur la qualité des réordonnancements appris. De plus, dans tous les cas, l'espace des permutations le plus restreint conduit aux meilleurs résultats : est-ce un problème inhérent aux espaces de recherche ou cela pourrait-il être amélioré avec des meilleures méthodes d'entraînement ?

Langue	Décodeur fondée sur les graphes				Décodeur par transitions			
	Non perm.	Permuté			Non perm.	Permuté		
		$\mathcal{O}(n^6)$	$\mathcal{O}(n^4)$	$\mathcal{O}(n)$		$\mathcal{O}(n^6)$	$\mathcal{O}(n^4)$	$\mathcal{O}(n)$
en	89,3/86,5	-	-	-	89,3/86,5	-	-	-
bg	**77,5/66,2**	74,7/63,9	68,3/58,3	77,2/66,0	**78,2/66,7**	75,0/63,9	68,5/58,2	77,9/66,5
ca	74,2/65,6	73,8/65,3	65,1/57,3	**74,7/66,1**	72,0/62,9	72,5/63,6	64,7/56,6	**73,4/64,4**
cs	**61,9/52,7**	59,4/50,4	53,6/45,6	61,5/52,3	**62,2/52,3**	59,9/50,2	54,0/45,4	61,9/52,0
da	**77,2/68,6**	74,8/66,8	66,1/58,8	76,1/67,8	**77,6/68,2**	75,4/66,6	67,1/59,0	76,9/67,7
de	65,9/56,9	64,9/56,3	61,5/53,6	**66,4/57,5**	65,0/56,2	64,1/55,8	60,7/53,1	**65,5/57,1**
es	76,0/68,2	75,7/68,0	70,3/63,1	**76,3/68,6**	74,6/66,5	75,0/66,9	69,6/62,0	**75,7/67,5**
et	**65,4/45,4**	64,4/44,8	58,9/40,8	65,3/45,3	**64,2/44,2**	63,2/43,8	57,8/40,0	64,0/**44,2**
fi	**68,6/50,6**	65,7/48,3	58,4/42,8	67,9/50,0	**68,4/50,2**	65,6/48,0	58,3/42,4	68,0/49,8
fr	72,4/66,2	69,6/63,2	62,8/57,2	**73,7/67,2**	70,4/64,0	70,0/63,4	63,1/57,1	**74,3/67,6**
he	60,6/53,4	59,3/52,6	53,4/47,7	**61,1/53,8**	57,5/42,8	56,7/43,3	51,5/40,7	**57,9/43,5**
hr	59,7/50,7	59,3/50,5	54,3/46,4	**60,5/51,5**	60,2/50,5	59,2/50,1	54,3/45,8	**60,4/51,0**
it	78,4/72,2	75,7/69,6	66,6/60,9	**79,3/73,0**	76,4/70,3	75,4/69,4	66,0/60,4	**78,8/72,6**
lv	**70,9/48,6**	69,7/47,7	62,0/42,4	70,6/48,3	**70,5/48,0**	69,5/47,3	61,7/41,7	70,2/47,8
nl	**68,4/60,8**	66,8/59,6	61,4/55,0	68,1/60,7	**69,6/61,9**	67,8/60,6	62,2/55,4	69,2/61,7
no	**80,0/71,3**	78,0/69,4	70,2/62,5	79,3/70,7	**80,4/70,9**	78,1/68,9	70,2/61,8	79,6/70,3
pl	64,1/53,1	62,4/51,4	57,1/46,9	**64,6/53,4**	64,3/53,6	63,4/52,9	57,9/48,1	**65,4/54,7**
pt	75,0/67,8	75,2/67,9	67,7/61,1	**76,1/68,8**	73,5/66,2	75,8/68,5	68,3/61,6	**76,6/69,3**
ro	63,2/51,2	61,9/50,1	55,0/44,5	**64,4/52,3**	60,7/48,3	61,1/49,1	54,3/43,5	**63,4/51,0**
ru	**60,6/51,8**	59,7/51,0	54,9/47,0	60,4/51,6	**61,3/53,0**	60,4/52,2	55,6/48,0	61,0/52,7
sk	**64,9/56,5**	63,8/55,3	59,2/51,5	64,4/56,0	**66,3/57,2**	65,3/56,2	60,2/52,1	66,2/57,0
sl	**66,8/55,4**	62,8/51,4	58,6/48,0	66,3/54,7	**67,9/56,0**	64,0/52,6	59,6/48,0	67,5/55,6
sv	**82,3/74,6**	81,1/73,5	73,7/66,4	81,9/74,2	**83,5/75,0**	82,2/73,7	74,2/66,3	83,1/74,6
uk	**58,4/50,8**	57,3/49,7	52,2/45,4	58,2/50,5	**59,5/51,5**	58,4/50,3	53,3/46,3	59,1/51,1
Moyenne	69,2 / **59,1**	67,7 / 57,7	61,4 / 52,3	**69,3 / 59,1**	68,9 / 58,1	67,7 / 57,3	61,4 / 51,9	**69,4 / 58,7**

TABLE 1 – Résultats (UAS/LAS, sans la ponctuation) sur les ensembles de test des données *Universal Dependencies* (Zeman *et al.*, 2021).

	Décodeur fondée sur les graphes					Décodeur par transitions				
	Non perm.	Permuté				Non perm.	Permuté			
		$\mathcal{O}(n^6)$	$\mathcal{O}(n^4)$	$\mathcal{O}(n)$	Gains		$\mathcal{O}(n^6)$	$\mathcal{O}(n^4)$	$\mathcal{O}(n)$	Gains
FTB	66,9/60,7	63,6/57,2	56,5/51,2	**68,0/61,4**	+1,1/+0,7	64,6/58,1	64,4/57,5	56,9/51,0	**68,6/61,4**	+4,0/+3,3
FQB	75,5/66,8	73,6/64,9	69,7/61,2	**77,3/68,5**	+1,8/+1,7	75,9/68,0	73,9/65,9	70,6/62,5	**79,1/71,1**	+3,2/3,1

TABLE 2 – Résultats pour le *French Treebank* (FTB) et le *French Question Bank* (FQB).

Le tableau 2 résume les résultats pour deux corpus du français : le French Treebank (Abeillé *et al.*, 2003), qui contient des extraits d'article du journal Le Monde et le French Question Bank (Seddah & Candito, 2016), qui contient essentiellement des questions. Sur le FTB, le décodeur par transitions obtient des résultats nettement moins bons lorsque les phrases ne sont pas permutées ($-2.3/-3.2$). Cependant, une fois les phrases permutées avec l'algorithme en $\mathcal{O}(n)$, les résultats sont grandement améliorés, démontrant la plus grande sensibilité à l'ordre des mots de ce modèle. Notons un dernier point intéressant : le décodeur fondé sur les graphes obtient des gains plus importants sur le FQB que sur le FTB, alors que ce n'est pas le cas du décodeur par transition.

5.3 Analyse détaillée

Dans cette section nous analysons plus en détail l'impact du ré-ordonnancement sur les performances des différents modèles. Nous constatons un gain de performances significatif pour plusieurs langues avec la version $\mathcal{O}(n)$ de l'algorithme de programmation dynamique. Bien que l'espace ainsi exploré soit restreint, il permet de traiter des différences d'ordres bien connues comme l'inversion de l'adjectif

et du nom entre l'anglais et le français. En guise d'illustration, considérons la phrase suivante tirée du corpus FQB ainsi que sa version permutée par l'algorithme en $\mathcal{O}(n)$:

— Phrase originale : « Trois des états membres (la France, l'Italie et l'Espagne) ont ainsi exigé de la commission qu'elle discipline l'activité des mandataires (Le Monde du 13 novembre 1993). » ;
— Phrase permutée $\mathcal{O}(n)$: « Trois des membres états (la France, l'Italie et l'Espagne) ont ainsi exigé de la commission qu'elle discipline l'activité des mandataires (Le Monde du novembre 13 1993). ».

Nous constatons que l'ordre de l'adjectif *membres* et du nom *états* ont bien été inversés afin de correspondre à l'ordre dominant en anglais. De même, pour la date : le jour et le mois sont inversés. Le modèle arrive donc à effectuer correctement les ré-ordonnancements attendus.

Si l'on s'intéresse aux phrases permutées par les versions $\mathcal{O}(n^6)$ et $\mathcal{O}(n^4)$, nous constatons que ces mêmes permutations sont également effectuées. Cependant, le fait d'explorer des espaces de permutations plus riches donne la possibilité au modèle d'effectuer des permutations à plus longue distance qui vont souvent perturber la cohérence syntaxique de la phrase. Considérons l'exemple suivant :

— Phrase originale : « Ainsi le scénario le plus probable est une dépendance accrue vis-à-vis des flux d'aide extérieurs. » ;
— Phrase permutée $\mathcal{O}(n^4)$: « Ainsi le scénario est une dépendance accrue vis-à le plus probable vis des flux - d'extérieurs aide. » ;
— Phrase permutée $\mathcal{O}(n^6)$: « Ainsi le scénario le plus probable dépendance est une accrue vis-à-vis des flux d'extérieurs aide. ».

Nous constatons que pour les deux versions de l'algorithme, la permutation de l'adjectif *extérieurs* et du nom *aide* a été effectuée, mais (a) le couple *dépendance accrue* n'est pas permuté, et (b) que des permutations supplémentaires indésirables sont introduites. Notons que bien que ces permutations supplémentaires ne soient pas justes, elles ne sont pas forcément incohérentes syntaxiquement. En effet, pour la version $\mathcal{O}(n^6)$ on peut noter que les couples de mots *probable dépendance* et *une accrue* forment des bigrammes plausibles lorsqu'on les considère indépendamment du contexte. Ainsi, le fait de considérer des poids bigrammes peut avoir pour conséquence la génération d'un ré-ordonnancement globalement incorrect à cause de certains bigrammes syntaxiquement valides et dont le poids est très important.

Nous constatons donc que le modèle effectue des ré-ordonnancements cohérents où l'on retrouve des permutations attendues, mais que l'exploration d'espaces riches et le fait d'utiliser un modèle bigramme favorisent des permutations locales qui peuvent pénaliser la cohérence syntaxique globale de la phrase.

6 Conclusion

Nous avons proposé et évalué une approche pour l'adaptation cross-lingue d'un analyseur en dépendances qui ne nécessite pas de réentraîner le modèle et qui peut être appliquée dans un scénario « zéro-shot ». Cette approche n'utilise que des données monolingues anglaises, et peut se combiner avec de multiples modèles de réordonnancement. Les résultats expérimentaux obtenus sur une vingtaine de langues montrent que cette approche est viable et permet d'améliorer en moyenne les résultats obtenus. Les variations observées entre langues cibles restent à analyser en profondeur en relation

avec les différents composants du modèle (plongements lexicaux, modèles de réordonnancement) et les différences typologiques avec l'anglais utilisé comme langue source.

Remerciements

Nous remercions les 3 relecteurices anonymes pour leurs remarques et suggestions. Nous remercions Shu Okabe pour l'aide avec l'exemple en japonais.

Références

ABEILLÉ A., CLÉMENT L. & TOUSSENEL F. (2003). *Building a Treebank for French*, In A. ABEILLÉ, Éd., *Treebanks : Building and Using Parsed Corpora*, p. 165–187. Springer Netherlands : Dordrecht. DOI : 10.1007/978-94-010-0201-1_10.

AHMAD W., ZHANG Z., MA X., HOVY E., CHANG K.-W. & PENG N. (2019). On difficulties of cross-lingual transfer with order differences : A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, p. 2440–2452, Minneapolis, Minnesota : Association for Computational Linguistics. DOI : 10.18653/v1/N19-1253.

AUFRANT L., WISNIEWSKI G. & YVON F. (2016). Zero-resource dependency parsing : Boosting delexicalized cross-lingual transfer with linguistic knowledge. In *Proceedings of the 26th International Conference on Computational Linguistics : Technical Papers*, COLING 2016, p. 119–130, Osaka, Japan : The COLING 2016 Organizing Committee.

BOMPADRE A. & ORLIN J. B. (2005). Using grammars to generate very large scale neighborhoods for the traveling salesman problem and other sequencing problems. In *Proceedings of the 11th International Conference on Integer Programming and Combinatorial Optimization*, IPCO'05, p. 437–451, Berlin, Heidelberg : Springer-Verlag. DOI : 10.1007/11496915_32.

CROES G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, **6**(6), 791–812.

EISNER J. & SATTA G. (1999). Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, p. 457–464, College Park, Maryland, USA : Association for Computational Linguistics. DOI : 10.3115/1034678.1034748.

EISNER J. & SATTA G. (2000). A faster parsing algorithm for Lexicalized Tree-Adjoining Grammars. In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+5)*, p. 79–84, Université Paris 7.

EISNER J. & TROMBLE R. W. (2006). Local search with very large-scale neighborhoods for optimal permutations in machine translation. In *Proceedings of the HLT-NAACL Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, p. 57–75, New York.

GOODMAN J. (1999). Semiring parsing. *Computational Linguistics*, **25**(4), 573–606.

HASTINGS W. K. (1970). Monte carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**(1), 97–109.

HEMASPAANDRA L. & OGIHARA M. (2001). *The complexity theory companion*. Springer Science & Business Media.

HOCHREITER S. & SCHMIDHUBER J. (1997). Long short-term memory. *Neural computation*, **9**(8), 1735–1780.

HORVAT M. & BYRNE W. (2014). A graph-based approach to string regeneration. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, p. 85–95, Gothenburg, Sweden : Association for Computational Linguistics. DOI : 10.3115/v1/E14-3010.

JESPERSEN O. (1924). *The philosophy of grammar*. George Allen & Unwin.

JOULIN A., BOJANOWSKI P., MIKOLOV T., JÉGOU H. & GRAVE E. (2018). Loss in translation : Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

KARP R. M. (1972). *Reducibility among Combinatorial Problems*, In R. E. MILLER, J. W. THATCHER & J. D. BOHLINGER, Éds., *Complexity of Computer Computations : Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*, p. 85–103. Springer US : Boston, MA. DOI : 10.1007/978-1-4684-2001-2_9.

KINGMA D. P. & BA J. (2015). Adam : A method for stochastic optimization. In Y. BENGIO & Y. LECUN, Éds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

KOLLER D. & FRIEDMAN N. (2009). *Probabilistic graphical models : principles and techniques*. MIT press.

KUMAR S. & BYRNE W. (2005). Local phrase reordering models for statistical machine translation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, p. 161–168, Vancouver, British Columbia, Canada : Association for Computational Linguistics.

METROPOLIS N., ROSENBLUTH A. W., ROSENBLUTH M. N., TELLER A. H. & TELLER E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, **21**(6), 1087–1092. DOI : 10.1063/1.1699114.

PÉCHEUX N., ALLAUZEN A., NIEHUES J. & YVON F. (2016). Reordering Space Design in Statistical Machine Translation. *Language Resources and Evaluation*, **50**, 375–410. DOI : 10.1007/s10579-016-9353-8, HAL : hal-01620902.

PEREIRA F. C. N. & WARREN D. H. D. (1983). Parsing as deduction. In *21st Annual Meeting of the Association for Computational Linguistics*, p. 137–144, Cambridge, Massachusetts, USA : Association for Computational Linguistics. DOI : 10.3115/981311.981338.

PETROV S., DAS D. & MCDONALD R. (2012). A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, p. 2089–2096, Istanbul, Turkey : European Language Resources Association (ELRA).

SARAWAGI S. & COHEN W. W. (2004). Semi-Markov conditional random fields for information extraction. In L. SAUL, Y. WEISS & L. BOTTOU, Éds., *Advances in Neural Information Processing Systems*, volume 17 : MIT Press.

SEDDAH D. & CANDITO M. (2016). Hard time parsing questions : Building a QuestionBank for French. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, p. 2366–2370, Portorož, Slovenia : European Language Resources Association (ELRA).

SØGAARD A. (2011). Data point selection for cross-language adaptation of dependency parsers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies*, p. 682–686, Portland, Oregon, USA : Association for Computational Linguistics.

TROMBLE R. & EISNER J. (2009). Learning linear ordering problems for better translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, p. 1007–1016, Singapore : Association for Computational Linguistics.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. U. & POLOSUKHIN I. (2017). Attention is all you need. In I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems*, volume 30 : Curran Associates, Inc.

WAINWRIGHT M. J., JORDAN M. I. *et al.* (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, **1**(1–2), 1–305.

WU D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, **23**(3), 377–403.

ZASLAVSKIY M., DYMETMAN M. & CANCEDDA N. (2009). Phrase-based statistical machine translation as a traveling salesman problem. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, p. 333–341, Suntec, Singapore : Association for Computational Linguistics.

ZEMAN D., NIVRE J., ABRAMS M., ACKERMANN E., AEPLI N., AGHAEI H., AGIĆ Ž., AHMADI A., AHRENBERG L., AJEDE C. K., ALEKSANDRAVIČIŪTĖ G., ALFINA I., ANTONSEN L., APLONOVA K., AQUINO A., ARAGON C., ARANZABE M. J., ARICAN B. N., ARNARDÓTTIR Ĥ., ARUTIE G., ARWIDARASTI J. N., ASAHARA M., ASLAN D. B., ATEYAH L., ATMACA F., ATTIA M., ATUTXA A., AUGUSTINUS L., BADMAEVA E., BALASUBRAMANI K., BALLESTEROS M., BANERJEE E., BANK S., BARBU MITITELU V., BARKARSON S., BASILE R., BASMOV V., BATCHELOR C., BAUER J., BEDIR S. T., BENGOETXEA K., BERK G., BERZAK Y., BHAT I. A., BHAT R. A., BIAGETTI E., BICK E., BIELINSKIENĖ A., BJARNADÓTTIR K., BLOKLAND R., BOBICEV V., BOIZOU L., BORGES VÖLKER E., BÖRSTELL C., BOSCO C., BOUMA G., BOWMAN S., BOYD A., BRAGGAAR A., BROKAITĖ K., BURCHARDT A., CANDITO M., CARON B., CARON G., CASSIDY L., CAVALCANTI T., CEBIROĞLU ERYIĞIT G., CECCHINI F. M., CELANO G. G. A., ČÉPLÖ S., CESUR N., CETIN S., ÇETINOĞLU Ö., CHALUB F., CHAUHAN S., CHI E., CHIKA T., CHO Y., CHOI J., CHUN J., CHUNG J., CIGNARELLA A. T., CINKOVÁ S., COLLOMB A., ÇÖLTEKIN Ç., CONNOR M., COURTIN M., CRISTESCU M., DANIEL P., DAVIDSON E., DE MARNEFFE M.-C., DE PAIVA V., DERIN M. O., DE SOUZA E., DIAZ DE ILARRAZA A., DICKERSON C., DINAKARAMANI A., DI NUOVO E., DIONE B., DIRIX P., DOBROVOLJC K., DOZAT T., DROGANOVA K., DWIVEDI P., ECKHOFF H., EICHE S., ELI M., ELKAHKY A., EPHREM B., ERINA O., ERJAVEC T., ETIENNE A., EVELYN W., FACUNDES S., FARKAS R., FERDAOUSI J., FERNANDA M., FERNANDEZ ALCALDE H., FOSTER J., FREITAS C., FUJITA K., GAJDOŠOVÁ K., GALBRAITH D., GARCIA M., GÄRDENFORS M., GARZA S., GERARDI F. F., GERDES K., GINTER F., GODOY G., GOENAGA I., GOJENOLA K., GÖKIRMAK M., GOLDBERG Y., GÓMEZ GUINOVART X., GONZÁLEZ SAAVEDRA B., GRICIŪTĖ B., GRIONI M., GROBOL

L., Grūzītis N., Guillaume B., Guillot-Barbance C., Güngör T., Habash N., Hafsteinsson H., Hajič J., Hajič jr. J., Hämäläinen M., Hà Mỹ L., Han N.-R., Hanifmuti M. Y., Hardwick S., Harris K., Haug D., Heinecke J., Hellwig O., Hennig F., Hladká B., Hlaváčová J., Hociung F., Hohle P., Huber E., Hwang J., Ikeda T., Ingason A. K., Ion R., Irimia E., Ishola O., Ito K., Jannat S., Jelínek T., Jha A., Johannsen A., Jónsdóttir H., Jørgensen F., Juutinen M., K S., Kaşikara H., Kaasen A., Kabaeva N., Kahane S., Kanayama H., Kanerva J., Kara N., Katz B., Kayadelen T., Kenney J., Kettnerová V., Kirchner J., Klementieva E., Klyachko E., Köhn A., Köksal A., Kopacewicz K., Korkiakangas T., Köse M., Kotsyba N., Kovalevskaitė J., Krek S., Krishnamurthy P., Kübler S., Kuyrukçu O., Kuzgun A., Kwak S., Laippala V., Lam L., Lambertino L., Lando T., Larasati S. D., Lavrentiev A., Lee J., Lê Hồng P., Lenci A., Lertpradit S., Leung H., Levina M., Li C. Y., Li J., Li K., Li Y., Lim K., Lima Padovani B., Lindén K., Ljubešić N., Loginova O., Lusito S., Luthfi A., Luukko M., Lyashevskaya O., Lynn T., Macketanz V., Mahamdi M., Maillard J., Makazhanov A., Mandl M., Manning C., Manurung R., Marşan B., Mărănduc C., Mareček D., Marheinecke K., Martínez Alonso H., Martín-Rodríguez L., Martins A., Mašek J., Matsuda H., Matsumoto Y., Mazzei A., McDonald R., McGuinness S., Mendonça G., Merzhevich T., Miekka N., Mischenkova K., Misirpashayeva M., Missilä A., Mititelu C., Mitrofan M., Miyao Y., Mojiri Foroushani A., Molnár J., Moloodi A., Montemagni S., More A., Moreno Romero L., Moretti G., Mori K. S., Mori S., Morioka T., Moro S., Mortensen B., Moskalevskyi B., Muischnek K., Munro R., Murawaki Y., Müürisep K., Nainwani P., Nakhlé M., Navarro Horñiacek J. I., Nedoluzhko A., Nešpore-Bērzkalne G., Nevaci M., Nguyễn Thị L., Nguyễn Thị Minh H., Nikaido Y., Nikolaev V., Nitisaroj R., Nourian A., Nurmi H., Ojala S., Ojha A. K., Olúòkun A., Omura M., Onwuegbuzia E., Osenova P., Östling R., Øvrelid L., Özateş Ş. B., Özçelik M., Özgür A., Öztürk Başaran B., Park H. H., Partanen N., Pascual E., Passarotti M., Patejuk A., Paulino-Passos G., Peljak-Łapińska A., Peng S., Perez C.-A., Perkova N., Perrier G., Petrov S., Petrova D., Phelan J., Piitulainen J., Pirinen T. A., Pitler E., Plank B., Poibeau T., Ponomareva L., Popel M., Pretkalniņa L., Prévost S., Prokopidis P., Przepiórkowski A., Puolakainen T., Pyysalo S., Qi P., Rääbis A., Rademaker A., Rahoman M., Rama T., Ramasamy L., Ramisch C., Rashel F., Rasooli M. S., Ravishankar V., Real L., Rebeja P., Reddy S., Regnault M., Rehm G., Riabov I., Riessler M., Rimkutė E., Rinaldi L., Rituma L., Rizqiyah P., Rocha L., Rögnvaldsson E., Romanenko M., Rosa R., Roșca V., Rovati D., Rudina O., Rueter J., Rúnarsson K., Sadde S., Safari P., Sagot B., Sahala A., Saleh S., Salomoni A., Samardžić T., Samson S., Sanguinetti M., Saniyar E., Särg D., Saulīte B., Sawanakunanon Y., Saxena S., Scannell K., Scarlata S., Schneider N., Schuster S., Schwartz L., Seddah D., Seeker W., Seraji M., Shahzadi S., Shen M., Shimada A., Shirasu H., Shishkina Y., Shohibussirri M., Sichinava D., Siewert J., Sigurðsson E. F., Silveira A., Silveira N., Simi M., Simionescu R., Simkó K., Šimková M., Simov K., Skachedubova M., Smith A., Soares-Bastos I., Sourov S., Spadine C., Sprugnoli R., Steingrímsson S., Stella A., Straka M., Strickland E., Strnadová J., Suhr A., Sulestio Y. L., Sulubacak U., Suzuki S., Szántó Z., Taguchi C., Taji D., Takahashi Y., Tamburini F., Tan M. A. C., Tanaka T., Tanaya D., Tella S., Tellier I., Testori M., Thomas G., Torga L., Toska M., Trosterud T., Trukhina A., Tsarfaty R., Türk U., Tyers F., Uematsu S., Untilov R., Urešová Z., Uria L., Uszkoreit H., Utka A., Vajjala S., van der Goot R., Vanhove M., van Niekerk D., van Noord

G., VARGA V., VILLEMONTE DE LA CLERGERIE E., VINCZE V., VLASOVA N., WAKASA A., WALLENBERG J. C., WALLIN L., WALSH A., WANG J. X., WASHINGTON J. N., WENDT M., WIDMER P., WIJONO S. H., WILLIAMS S., WIRÉN M., WITTERN C., WOLDEMARIAM T., WONG T.-S., WRÓBLEWSKA A., YAKO M., YAMASHITA K., YAMAZAKI N., YAN C., YASUOKA K., YAVRUMYAN M. M., YENICE A. B., YILDIZ O. T., YU Z., YULIAWATI A., ŽABOKRTSKÝ Z., ZAHRA S., ZELDES A., ZHOU H., ZHU H., ZHURAVLEVA A. & ZIANE R. (2021). Universal dependencies 2.9. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

ZENS R. & NEY H. (2003). A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, p. 144–151, Sapporo, Japan : Association for Computational Linguistics. DOI : 10.3115/1075096.1075115.