Annoter et prédire des représentations linguistiques de phrases

Marie Candito

Université de Paris

Annoter et prédire des représentations linguistiques de phrases

Habilitation à diriger des recherches

en Informatique

Marie Candito

Jury
Sylvain Schmitz, président
Frédéric Béchet, rapporteur
Claire Gardent, rapporteure
Paola Merlo, rapporteure
Benoît Crabbé, examinateur
Pierre Zweigenbaum, examinateur

Mémoire soutenu le 19 janvier 2022

Table des matières

Remerciements

La recherche est une activité d'échanges, qu'il s'agisse de lire le travail d'autres, ou de produire ensemble de nouvelles connaissances. C'est avec beaucoup d'émotion que je me suis remémorée toutes les collaborations de ces dernières années, qui me permettent aujourd'hui d'écrire ce document.

Je voudrais sincèrement remercier tous mes co-auteur.e.s, mes collègues, les étudiant.e.s que j'ai pu encadrer, pour ce bout de chemin fait ensemble, professionnel et personnel. En particulier, merci Djamé pour ton enthousiasme, ta disponibilité et bien sûr pour tes introductions visionnaires et personnelles. Merci Benoît pour ton accueil à mon arrivée à Paris Diderot, pour m'avoir mise sur les rails du parsing statitique et pour ton aide pendant toutes ces années. Merci Anne de m'avoir proposé un beau sujet de doctorat. Merci Sylvain, pour ta vision de l'interprétation sémantique des TAG. Merci Laurence pour ton énergie, qui a porté la création de l'équipe Alpage, et pour les poutous émaillant tes mails. Merci Laure et Philippe pour les longues discussions sur la causalité ! Merci Lucie et Richard, pour notre belle classification des pronominaux. Merci Mathieu, pour ton énergie à porter PARSEME-FR, pour nos collaborations et co-encadrement, toujours sous le signe de la rigueur décontractée ! Merci Carlos et Agata pour votre incroyable travail dans PARSEME, et la joie que vous apportez dans tous vos projets. Merci Alexis et Pascal pour votre engagement dans "le cursus LI" à Paris Diderot, et pour m'avoir proposé d'y être intervenante professionnelle : cela m'a redonné l'envie de l'université. Merci Bruno et Guy, de faire vivre le corpus Sequoia. Merci Alexis pour le beau projet Sequoia et nos échanges sur Asfalda. Merci à Enrique, Marianne, Hazem et Vincent, d'avoir mené à bien votre doctorat, entre enthousiasme et doute épistémique : ne pas douter en sciences serait douteux ! Et tous ces petits clins d'oeil sont loin de couvrir tous nos échanges.

Merci Benjamin, Angie, Judith, pour le bonheur de cheminer ensemble !

Table des figures

Liste des tableaux

Chapitre 1

Introduction

Ce mémoire d'habilitation présente mes travaux de recherche depuis ma thèse de doctorat, avec des niveaux de détail très variables. Certains chapitres sont de brefs résumés, ce que je signale dans leur titre. Les autres chapitres, plus détaillés, correspondent à différentes périodes et j'ai essayé de resituer les travaux dans le contexte de l'état de l'art de l'époque, qui apparaît parfois très daté. La plupart des chapitres correspondent à des collaborations, et des co-encadrements de doctorat, que j'explicite en début de chapitre ou de sections. Dans ce document, je n'ai pas distingué les publications dont je suis co-auteure des autres. La liste de mes publications et le résumé des encadrements auxquels j'ai participé se trouve dans le CV à part.

Après ma thèse j'ai travaillé pendant huit ans dans le privé, avant de revenir à une carrière académique comme enseignante-chercheuse. Ceci fait que mon travail de thèse est très différent des travaux de recherche que j'ai menés ensuite à mon retour à l'université, en cela qu'ils n'utilisent pas du tout de corpus ni d'informations de fréquence. En revenant à une carrière académique en 2007, j'ai mis le cap sur l'empirisme et le TAL "guidé par les données".

Le mémoire commence avec un résumé de ma thèse, une représentation de grammaires d'arbres adjoints grâce à une "métagrammaire" (chapitre 2). Mon activité a ensuite concerné essentiellement l'annotation manuelle ou automatique de phrases, explicitant des représentations linguistiques de types divers, et le mémoire suit un ordre traditionnel relatif au type d'informations linguistiques : je commence avec le chapitre 3 sur les corpus arborés. Le chapitre 4 est un court résumé de travaux concernant l'annotation et le traitement automatique des expressions polylexicales, en collaboration avec Mathieu Constant, Carlos Ramisch, Agata Savary, et Hazem Al Saied (doctorant co-encadré par Mathieu et moi-même). Le chapitre 5 décrit des contributions en analyse syntaxique automatique ("parsing" pour aller plus vite), assez anciennes (période 2007-2011), avec Benoît Crabbé, Djamé Seddah et Enrique Henestroza Anguiano, que j'ai encadré en doctorat. Le chapitre 6 concerne le projet de graphes syntaxiques profonds, conçus comme niveau de représentation entre syntaxe et sémantique, avec Bruno Guillaume, Guy Perrier, Corentin Ribeyre et Djamé Seddah, ainsi qu'Éric de la Clergerie et Karen Fört. Dans le chapitre 7, je présente un travail autonome sur le parsing vers des graphes de dépendances bilexicales. Enfin au chapitre 8, je résume le projet de production et utilisation d'un réseau de cadres sémantiques, un FrameNet du français, projet ANR dont j'ai été la porteuse, et dans lequel ont collaboré de nombreuses personnes, en particulier Marianne Djemaa, Philippe Muller, Laure Vieu, Olivier Michalon et Alexis Nasr. Par manque de place, j'ai choisi de ne pas détailler le travail de co-encadrement, avec Benoît Crabbé, de la

thèse de Vincent Segonne portant sur la désambiguïsation lexicale de verbes, ni celui réalisé avec Lucie Barque et Richard Huyghe sur la classification des verbes pronominaux en français.

Ce mémoire couvre une période de temps longue (!), marquée par l'arrivée de méthodes neuronales en TAL. L'apprentissage par transfert permet de fournir des représentations distribuées de mots, d'abord hors contexte puis en contexte, en utilisant des objectifs génériques, en particulier la prédiction d'un mot sachant son contexte. Il est fascinant de constater qu'un objectif aussi simple et brut permet de construire des modèles apportant des gains très importants dans à peu près toutes les tâches de TAL. Le transfert se fait en utilisant des corpus à l'état brut, ne nécessitant pas de modélisation linguistique (outre la définition des unités considérées). C'est ainsi l'objectif même d'analyse automatique de phrases qui est remis en cause. Certaines tâches, comme la traduction automatique, le résumé automatique, l'analyse de sentiments sont actuellement mieux gérées par des techniques de bout-en-bout ("end-to-end"), ne nécessitant pas d'expliciter des représentations linguistiques traditionnelles. On assiste même à une ingénierie inversée, où ce sont les modèles appris sur corpus bruts qui sont sondés pour voir si et où s'y cachent les concepts linguistiques traditionnels.

Même s'il est difficile de prédire l'avenir du concept même d'analyse automatique de phrases, les besoins d'interprétabilité des modèles et de quantification des phénomènes linguistiques font que le concept reste d'actualité. On peut même espérer que les sondes linguistiques des modèles neuronaux permettent d'éclairer d'un jour nouveau certains concepts linguistiques. Pour ma part, je reste fascinée par la complexité des phénomènes langagiers, et tenter d'expliciter automatiquement ces phénomènes me plaît toujours autant !

Chapitre 2

La métagrammaire (résumé)

Je débute ma thèse fin 1995, l'analyse syntaxique probabiliste est encore balbutiante. Les travaux qui s'attaquent au problème du caractère hors-contexte des règles d'une Probabilistic Context-Free Grammar ((Collins, 1999), voir chapitre 5) me sont inconnus.

Je me place dans le cadre de l'analyse syntaxique automatique au moyen d'une grammaire électronique. Après avoir tâtonné sur la représentation des constructions causatives, je cherche un sujet plus formel. Je remercie beaucoup Anne Abeillé, qui me propose de travailler dans le formalisme des grammaires d'arbres adjoints (TAG), en particulier sur l'organisation de la grammaire. Les TAG font la part belle à la description linguistique : l'opération d'adjonction d'arbres (en résumé, où un arbre peut être inséré au sein d'un autre), permet de manipuler comme "règles de grammaire", des structures d'arbres de profondeur quelconque, dits "arbres élémentaires". Des principes linguistiques peuvent être suivis (Abeillé, 1991), qui assurent en général qu'un arbre élémentaire représente un prédicat avec des positions locales pour ses arguments exprimés. On parle de "domaine de localité" étendu. L'opération d'adjonction viendra éventuellement rendre non locales ces positions. L'avantage du domaine de localité étendu est de préparer l'interface avec l'analyse sémantique, avec un contrôle fin des arguments sous-catégorisés par tel ou tel prédicat.

La contre-partie du domaine de localité étendu est la multiplication des règles de grammaire, i.e. dans le cas des TAG, la multiplication des arbres élémentaires, avec une redondance évidente. J'ai défini une architecture de représentation de grammaires électroniques TAG, s'appuyant sur l'unification de description d'arbres, comme proposé par Vijay-Shanker and Schabes (1992). Cette architecture comprend des principes d'organisation des informations morphologiques et syntaxiques, et permet de générer, à partir d'un réseau de descriptions partielles d'arbres, les milliers d'arbres élémentaires d'une grammaire TAG pour une langue donnée. Parce qu'il s'agit de générer une grammaire à partir de représentations plus abstraites, j'ai appelé "métagrammaire" cette architecture générale, et pour une langue donnée, le réseau de descriptions partielles d'arbres forme une métagrammaire "instanciée".

J'ai instancié et comparé deux métagrammaires, pour le français et pour l'italien. Du point de vue de la description linguistique, elles constituent un niveau intermédiaire entre d'un côté une théorie syntaxique dans toute sa complexité et de l'autre une grammaire électronique utilisable par un système d'analyse ou de génération syntaxique. J'ai en particulier étudié et comparé les phénomènes d'alternance syntaxique (ou changements de diathèse, comme par exemple le passif) dans les deux langues. Certains phénomènes de l'italien m'ont amenée à conserver une procéduralité dans la gestion des alternances syntaxiques. En particulier, il existe en italien un passif du causatif et un causatif du passif. Pour obtenir les arbres élémentaires

correspondants, combiner procéduralement passif et causatif est plus économe qu'une combinaison monotone.

Cette proposition de métagrammaire a inspiré différents travaux, notamment à plus grande échelle et pour les formalismes LFG et TAG (Clément and Kinyon, 2003). Dans le formalisme XMG (pour eXtensible MetaGrammar) (Crabbé, 2005 ; Crabbé et al., 2013), le pouvoir expressif est amélioré. Le système FRMG d'Éric de la Clergerie (Villemonte de la Clergerie, 2010) utilise également le principe de compilation d'une métagrammaire pour obtenir une grammaire TAG du français très couvrante.

Mais du point de vue du TAL, l'analyse syntaxique automatique (NEWTERM dorénavant du "parsing") au moyen d'une grammaire TAG est problématique à deux niveaux. D'une part la conception de ces grammaires est orientée vers un objectif de linguistique formelle, à savoir de distinguer les phrases grammaticales des phrases agrammaticales. Les grammaires produites pendant ma thèse sont testées sur des énoncés forgés, construits pour couvrir un certain nombre de phénomènes grammaticaux et agrammaticaux. Il en résulte que ces grammaires manquent de couverture et échouent souvent à analyser des phrases attestées.

D'autre part, le parsing au moyen d'une grammaire TAG, comme avec d'autres types de grammaires électroniques non probabilisées, fournit l'ensemble des arbres syntaxiques formellement possibles pour une phrase, mais sans gérer l'ambiguïté syntaxique artificielle ainsi obtenue. Pour résoudre une telle ambiguïté, il est nécessaire d'utiliser des corpus, et la distribution la plus naturelle possible des phénomènes linguistiques.

À mon retour à l'université en 2007, beaucoup de progrès ont été faits en analyse syntaxique probabiliste. Il n'est plus question de se contenter d'analyseurs proposant parfois des dizaines de milliers d'analyses possibles sans se préoccuper de choisir la meilleure de ces analyses. Je me tourne alors résolument vers le TAL guidé par les données attestées, l'analyse par apprentissage automatique, et l'annotation manuelle de corpus pour nourrir cet apprentissage. Du point de vue du TAL, il s'agit d'obtenir une large couverture des phénomènes et de s'appuyer sur la distribution de ceux-ci en corpus. Du point de vue linguistique, j'y ai trouvé un moyen de se confronter à la réalité des phénomènes linguistiques, sans le biais de la création d'énoncés artificiels.

Chapitre 3

Corpus arborés

Dans ce chapitre, je commence par donner quelques repères historiques de l'utilisation de corpus annotés en TAL et en linguistique, en m'inspirant en particulier de l'introduction, écrite avec Mark Libermann, du numéro spécial de la revue TAL sur les corpus annotés (Candito and Liberman, 2019). Je discute ensuite section 3.2 des deux styles de représentations syntaxiques, et de mon choix de privilégier les arbres de dépendances. La section 3.3 résume la création du corpus SEQUOIA, créé avec Djamé Seddah dans le but initial d'expérimenter l'adaptation de domaine d'analyseurs syntaxiques. Enfin la section 3.4 concerne la conversion d'arbres de constituants vers des arbres de dépendances, et la gestion des dépendances non locales, en collaboration avec Benoît Crabbé, Djamé Seddah, et les étudiants de master à l'époque, Mathieu Falco et François Guérin.

3.1 Repères historiques

3.1.1 Premières annotations de corpus, premiers corpus arborés

Bien avant l'âge numérique, une forme d'annotation de corpus a été utilisée, pour servir l'interprétation de textes culturellement importants, en particulier religieux. Le Talmud constitue un exemple célèbre de texte (le *Mishna*) augmenté de commentaires interprétatifs et discussion (le *Gemara*) (Mielziner, 1903). À la fin du 19ème siècle, Strong (1890) numérote le lexique des "mots racine" hébreux de l'ancien testament, et des grecs du nouveau testament, et annote la version anglaise des deux testaments avec les numéros de mots. Fournies avec un index, ces annotations permettent de retrouver tous les passages utilisant tel ou tel mot hébreu, grec ou anglais.

L'histoire de l'annotation de corpus semble initialement proposée comme moyen de retrouver facilement des exemples de tel ou tel phénomène linguistique, au départ lexical, puis syntaxique ou autre, afin de documenter les usages linguistiques en particulier dans le cadre de dictionnaires. Entre 1949 et 1967, Roberto Busa travailla à IBM à la création de l'*Index Thoministicus*, un concordancier lemmatisé de tous les écrits de Saint Thomas d'Aquin, sous la forme de millions de cartes perforées (Busa, 1974). En Suède dans les années 60, le travail pionnier de Sture Allén à l'Université de Gothenburg a débouché sur le corpus Press-65, un corpus électronique d'un million de mots de textes journalistiques (Allén, 1968). En 1967, Kučera et Francis crérent le Brown corpus (Kučera and Francis, 1967), un corpus d'un million de mots d'anglais américain écrit, destiné à servir "la recherche sur la langue anglaise, aidée par ordinateur", et ce Brown corpus sera la source des citations de la première édition du American

Heritage Dictionary (Morris, 1969). Le premier corpus arboré, bien que difficilement manipulable, est le Talbanken, créé pour le suédois écrit et oral, à l'Université de Lund (Einarsson, 1976a,b). Le projet de Corpus de l'Académie Tchèque [1] a commencé dans les années 70, avec l'annotation manuelle morphologique et syntaxique d'un corpus, dans le but de construire un dictionnaire de fréquences.

En France, le projet de dictionnaire "Trésor de la Langue Française" (TLF) a débuté à l'INALF à Nancy dans les années 60, pour soutenir l'étude du lexique français. La base Frantext [2] de textes littéraires et philosophiques a été créée dans ce cadre, comme source d'exemples du TLF. Les volumes du dictionnaire ont été distribué entre 1971 et 1994, et un moteur de recherche de la base Frantext a été d'abord fourni en interne dès 1985 (Montémont, 2008), puis en ligne en 1998. Le projet MULTEXT financé par la Communauté Européenne au début des années 90 (Ide and Véronis, 1994) avait pour but de fournir des outils de manipulation de corpus et des corpus multilingues avec annotations linguistiques et de structuration.

3.1.2 Des corpus arborés facilement utilisables

Le corpus arboré le plus célèbre et le plus utilisé est sans doute le Penn Treebank (Marcus et al., 1993), contenant des articles anglais du *Wall Street Journal*, et dont la création a débuté à la fin des années 80, et qui a été utilisé pour des milliers de travaux sur la syntaxe anglaise et les systèmes d'analyse syntaxique automatique - Google Scholar recense environ 19 000 travaux qui y font référence. L'une des raisons essentielles de cette popularité (outre l'importance économique de la langue anglaise) est que le Penn Treebank a été mis à la disposition des chercheurs du monde entier, comme l'avait été le corpus Brown, contrairement à d'autres collections anciennes, telles que les treebanks suédois cités supra, qui étaient jalousement conservés par leurs créateurs. Des corpus arborés ont ensuite été construits pour de nombreuses langues typologiquement très variées. Pour le français, le premier et le plus gros corpus arboré est le French Treebank (ci-après FTB) (Abeillé and Barrier, 2004), contenant environ 20000 phrases du journal *Le Monde*.

Un courant parallèle de création de corpus syntaxiquement annotés utilise les dépendances syntaxiques (Tesnière, 1959) plutôt que les constituants, en particulier pour des langues slaves, à ordre beaucoup plus libres qu'en anglais, et ayant une forte tradition de linguistique en dépendances. D'un point de vue formel, ces deux représentations sont équivalentes (pour peu que l'on accepte les constituants discontinus) mais les relations syntaxiques non locales qui sont courantes dans les langues à ordre libre sont plus facilement représentées par des dépendances croisées ("non projectives", voir infra section 3.2.1) que par des constituants discontinus. Le projet emblématique à cet égard est sans conteste le Prague Dependency Bank, avec une première version sortie dès 1998 (Hajič, 1998). On peut également citer comme autres ressources pionnières les treebanks pour le néerlandais (Bouma et al., 2000), le danois (Kromann, 2003), ou le turc (Oflazer et al., 2003).

Aujourd'hui, on assiste au succès retentissant du projet des Universal Dependencies [3] (Nivre et al., 2016), une initiative définissant un schéma d'annotation morphologique et syntaxique en dépendances se voulant multilingue (d'où le terme "universal"). Cet effort collaboratif avec

1. Un historique de ce projet est fourni à http://ufal.mff.cuni.cz/rest/CAC/doc-cac10/cac-guide/eng/html/chapter2.html.
2. https://www.frantext.fr/.
3. https://universaldependencies.org/.

plus de 200 contributeurs aggrège aujourd'hui 157 corpus arborés (pour la version 2.5), couvrant 90 langues typologiquement variées, ces treebanks étant natifs ou bien convertis depuis des schémas d'annotation en constituants ou dépendances spécifiques à une langue. Malgré les inévitables approximations du schéma d'annotation interlingue, cette ressource est abondamment utilisée pour la recherche en typologie linguistique quantitative (comme par exemple (Guzmán Naranjo and Becker, 2018), ou les actes du premier atelier de syntaxe quantitative QUASY 2019 (Chen and Ferrer i Cancho, 2019)) et pour du parsing multilingue (comme par exemple (Kondratyuk and Straka, 2019)).

3.1.3 Enjeux actuels pour les représentations linguistiques en TAL

Concernant l'avenir des corpus annotés, nous pouvons tout d'abord commenter certains aspects "sociologiques", au sein des communautés de recherche aussi bien en linguistique qu'en TAL. Les projets d'annotation sont coûteux et très chronophages, ce qui fait que leur réutilisabilité est cruciale. En outre, les utilisateurs de corpus annotés ne sont pas forcément armés pour réaliser eux-mêmes les annotations, ce qui fait que les chercheurs en TAL ont tendance à réutiliser des annotations déjà existantes. Cela permet d'expliquer l'émergence de standards de schémas d'annotation, souvent issus de projets centrés sur l'anglais, initiés par des acteurs majeurs américains du TAL. Les ressources dans les autres langues utilisent soit les mêmes schémas, en général à plus petite échelle, soit un schéma d'annotation original, au risque de manquer de visibilité internationale. Dans les deux cas, le "retour sur investissement" pour des équipes se lançant dans un projet d'annotation est faible. Dans le premier cas, le fait de réutiliser ou s'inspirer fortement d'un schéma d'annotation prévu pour l'anglais amoindrit la contribution scientifique aux yeux des relecteurs d'articles. Dans le second cas, un schéma d'annotation nouveau, pour une langue autre que l'anglais n'emporte pas l'adhésion et est finalement en général marginalisé. De ce point de vue, le projet Universal Dependencies se démarque de manière notable : bien qu'initialement porté par des poids lourds du TAL américains (Stanford, à travers les dépendances de Stanford, et Google, à travers le jeux de catégories morpho-syntaxiques universel), ce projet a favorisé une réflexion globale sur un schéma d'annotation à vocation multilingue.

D'un point de vue pratique, faciliter la production de ressources linguistiques est un fort enjeu. Une tendance a vu le jour, de lever la barrière de la rareté des experts en recourant à des annotateurs non experts, via des plateformes de "myriadisation" (*crow-sourcing* en anglais). Pour cela, les tâches sont morcelées, et les annotations sont "adjudiquées par le nombre", c'est-à-dire que l'on utilise un vote majoritaire sur les décisions de nombreux annotateurs non experts. La possibilité d'exploiter une force de travail potentiellement mondiale via ces plateformes a radicalement modifié l'économie de la production de ressources, en tous cas pour l'anglais. En effet, les inégalités entre les langues se trouvent renforcées. En 2014, sur 100 langues, 13 seulement étaient considérées comme disposant d'une main-d'oeuvre adéquate parmi les "turkers" (travailleurs via la plateforme Mechanical Turk) (Pavlick et al., 2014). Les préoccupations éthiques font désormais partie des grandes conférences de TAL et font l'objet d'ateliers spécifiques ou de numéros spéciaux (Fort et al., 2016 ; Hovy et al., 2017). Les "jeux constructifs" (game with a purpose) sont une alternative utilisée pour des ressources linguistiques variées, notamment des corpus annotés, par exemple pour l'annotation de coréférences (Chamberlain et al., 2013) ou la syntaxe de dépendance (Guillaume et al., 2016).

Une autre piste de recherche consiste à mieux utiliser les ressources existantes, notamment

avec un apprentissage plus efficace à partir de jeux de données de taille limitée. À noter que l'apprentissage multilingue exploite les données dans plusieurs langues, en faisant bénéficier celles moins dotées des données d'autres langues. De ce point de vue, la réutilisation de mêmes schémas d'annotation pour plusieurs langues s'avère cruciale. Mais produire des ressources avec une visée multilingue est particulièrement difficile, tant sur le plan de l'adéquation linguistique que sur le plan de l'organisation pratique. Comme nous l'avons déjà noté, le projet Universal Dependencies a réussi à faire collaborer des centaines de contributeurs. Le projet PARSEME a produit des directives et des données pour 20 langues, pour les expressions verbales multi-mots. Enfin, l'annotation de données multimodales est un autre défi, avec la nécessité d'annoter les interconnexions entre les différentes modalités (parole, geste, états émotionnels...).

Du côté des représentations linguistiques, l'annotation de corpus au moyen de symboles atomiques (pour n'importe quel type de concept linguistique) est remise en question, notamment en raison de l'utilisation actuelle de représentations continues dans les modèles neuronaux. Les frontières entre les catégories linguistiques sont souvent difficiles à tracer avec précision (des exemples de difficultés bien connues sont la distinction argument/ajout en syntaxe, ou la distinction adjectif/participe dans de nombreuses langues, y compris le français ou l'allemand). Cela a inévitablement un impact sur le processus d'annotation. Par exemple, Plank et al. (2014) montrent que les désaccords sur l'annotation des POS concernent des points linguistiques discutables plutôt que des erreurs aléatoires, et devraient être utilisés dans la phase d'apprentissage. Ces difficultés remettent en cause la théorie, mais aussi la méthodologie d'annotation actuelle, dans laquelle la résolution des conflits d'annotation est la phase la plus chronophage. D'un point de vue théorique, on peut considérer que le succès, au sein des modèles de TAL neuronaux, des représentations continues de mots de tout type de symbole discret (linguistique et méta-linguistique) donne une validation empirique de la non-rigidité des frontières entre catégories.

3.1.4 Quel avenir pour les corpus arborés en TAL ?

En linguistique, le recours à une approche expérimentale se développe, qui passe soit par le contrôle statistique des jugements d'acceptabilité demandés à des locuteurs, soit par le recours à des énoncés attestés, et donc des corpus. De ce point de vue, les corpus arborés constituent une ressource précieuse.

Mais du point de vue du TAL, l'utilité des corpus arborés ne fait plus consensus, étant donné que l'utilité de l'analyse syntaxique automatique est elle-même sérieusement questionnée par le TAL actuel. En effet, le manque de données annotées pour la plupart des langues a amené à des modèles de TAL "auto-supervisés", i.e. où les exemples pour l'apprentissage supervisé sont trivialement créés à partir de corpus bruts, sans annotations linguistiques. Par ailleurs, les modèles dit "de bout en bout" (*end-to-end* en anglais) remettent en question le recours à différents niveaux d'analyse linguistique (de la phonétique à la sémantique). Un exemple typique de remise en cause de l'étape d'analyse syntaxique en TAL, et donc indirectement, de l'utilité des corpus arborés comme source d'apprentissage supervisé, est donné par la traduction automatique. Il y a eu deux étapes majeures de progrès en traduction automatique, qui toutes deux se passent, en tous cas dans un premier temps, d'analyse syntaxique (que ce soit la traduction statistique d'IBM à la fin des années 80, ou la traduction neuronale en particulier avec Bahdanau et al. (2015) ; Vaswani et al. (2017)). Mais on peut citer aussi d'autres tâches pour lesquelles il est connu que des facteurs syntaxiques entrent en jeu, mais qui peuvent être abordées en TAL aujourd'hui sans aucun recours à des représentations syntaxiques symboliques.

C'est le cas par exemple de la tâche de résolution de coférences, avec le système de bout-en-bout de Lee et al. (2017).

Cela dit, on assiste ces dernières années à la naissance d'un véritable domaine de recherche, où des "sondes linguistiques" sont utilisées pour tester les connaissances linguistiques encodées dans les paramètres des modèles de langue pré-entraînés [4] (cf. la série d'ateliers BlackBoxNLP, et le nouveau terme de "bertology", forgé par Rogers et al. (2020)). Ces efforts de recherche sur l'interprétabilité des modèles neuronaux montrent bien que le TAL sans explicitation méta-linguistique n'est pas satisfaisant. Actuellement, il s'agit surtout de rechercher au sein des milliards de paramètres appris, si des concepts linguistiques connus peuvent être retrouvés. À plus long terme, on peut souhaiter que la bertologie permette de questionner les représentations linguistiques elles-mêmes, en faisant émerger les représentations symboliques optimales étant donné un modèle appris sur texte brut.

3.2 Représentations syntaxiques : constituants et dépendances

Les représentations syntaxiques étant centrales dans mes travaux de recherche, et malgré la remise en cause partielle des corpus arborés et du parsing syntaxique, je prends un peu de place pour détailler ces représentations, les comparer et justifier mon choix de travailler principalement avec des dépendances syntaxiques plutôt que des constituants.

La représentation syntaxique de phrases occupe une place de choix en linguistique et en TAL. En linguistique les représentations syntaxiques doivent permettre de formaliser les contraintes d'agencement des mots entre eux pour former des phrases grammaticales. En TAL, comme évoqué en introduction, la prédiction de représentations syntaxiques - le "parsing" - a longtemps été considéré comme une étape préalable cruciale pour un accès automatisé à la sémantique de phrases et de textes.

Les deux types principaux de représentations syntaxiques proposés en linguistique, les arbres de constituants ou arbres syntagmatiques, et les arbres de dépendances, sont issues de traditions linguistiques anciennes et ont été formalisées dans la 2ème moitié du vingtième siècle. Leur utilisation en TAL a une histoire asymétrique : la majorité de la recherche scientifique en parsing, dominée par les occidentaux, a d'abord considéré le parsing en constituants, avec bien sûr une majorité de travaux sur l'anglais. Dans les années 90 débute un basculement vers le parsing en dépendances, et le parsing multilingue, le caractère multilingue n'étant pas étranger à ce mouvement.

La grammaire syntagmatique introduite par Chomsky dans les années cinquante ajoute à la notion directement observable d'ordre linéaire entre les mots, une notion d'organisation hiérarchique, le syntagme, séquence de mots et/ou de syntagmes ayant une certaine cohésion, mise en évidence par des tests de constituance. Les phrases sont vues à la fois comme un enchaînement linéaire et un emboîtement de mots et syntagmes.

La grammaire de dépendances, dont on fait parfois remonter l'histoire dès la grammaire du Sanskrit de Pāṇini (de Marneffe and Nivre, 2019), formalisée dans l'ouvrage pionnier de Tesnière (1959), puis Mel'čuk (1988), privilégie quant à elle des relations directement entre

4. On introduit ces modèles au chapitre sur le parsing vers graphes 7, section 7.1.1.

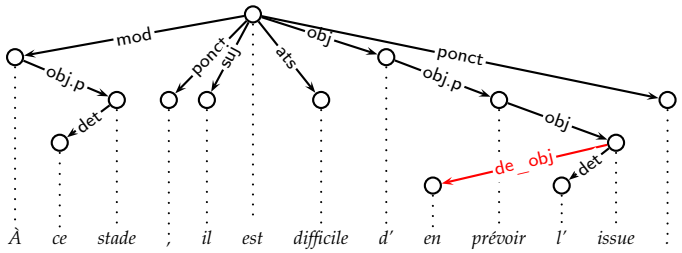

FIGURE 3.1 – Un exemple d'arbre de dépendances non projectif : la dépendance en rouge est non projective, comme le montre l'intersection avec l'axe vertical du mot *prévoir*.

mots[5]. D'un point de vue linguistique, schématiquement on peut dire que la notion de dépendance syntaxique capture le fait que la présence d'un mot dépendant est légitimée par celle du gouverneur. Mel'čuk (2009) propose des critères pour établir une dépendance entre deux mots w_i et w_j, en termes d'ordre relatif (la position linéaire d'un des deux doit être déterminé par rapport à l'autre) ou en termes prosodiques. L'orientation de la dépendance est déterminée en choisissant comme gouverneur le mot qui détermine le plus la distribution de l'ensemble, ou à défaut, celui qui détermine la forme morphologique de l'autre mot.
Les étiquettes des arcs fournissent un typage des dépendances, et correspondent à la notion traditionnelle de fonction grammaticale, entre un gouverneur, et toute la projection lexicale du dépendant.

3.2.1 Projectivité

Projectivité et arbres de dépendances

Si l'ordre linéaire et les relations de dépendance (l'ordre structurel) sont formellement distingués au sein des arbres de dépendances, une propriété fondamentale mêlant les deux types de relations est la notion de projectivité (Lecerf, 1960). Un arbre est projectif ssi tous ses noeuds ont une projection lexicale sans discontinuités. Graphiquement, une visualisation d'un arbre de dépendances comme illustré Figure 3.1 permet de repérer les arcs non projectifs : si l'arbre apparaît au dessus de la phrase dans l'ordre linéaire, avec un axe vertical reliant chaque noeud à son mot, une dépendance non projective coupera au moins un axe vertical.

On peut également donner une définition et une visualisation alternative de la non-projectivité en utilisant la notion de dépendances croisées : un arbre est non-projectif si et seulement si (i) deux dépendances au moins "se croisent" OU (ii) au moins une dépendance $i \rightarrow j$ couvre la racine de l'arbre (i.e. le noeud racine est linéairement entre i et j), les deux cas étant illustrés à la Figure 3.2.1.

Du point de vue computationnel, la contrainte de projectivité réduit en général la complexité de la tâche de parsing. Des analyseurs en dépendances projectives en $O(n)$ ont pu être proposés (Yamada and Matsumoto, 2003 ; Nivre and Scholz, 2004). Ce n'est pas le cas du parsing basé sur les graphes (McDonald et al., 2005b), où la recherche d'un arbre de plus haut poids au sein du graphe connectant tous les mots deux à deux est plus simple computationnellement

5. En réalité avec la notion de nucleus, Tesnières considérait comme unités parfois des groupes de mots.

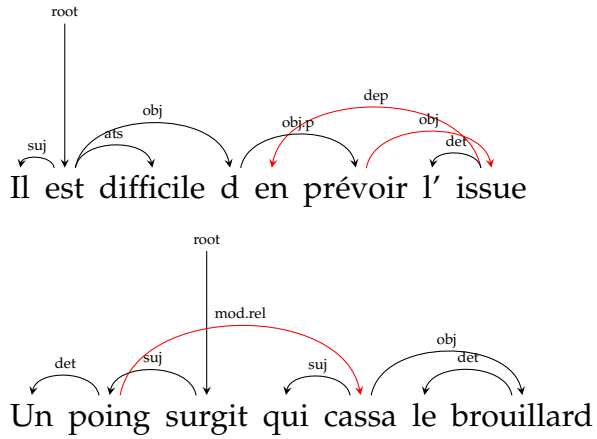

FIGURE 3.2 – Visualisation alternative de la non projectivité : (a) deux arcs se croisant (non-planarité) [FTB flmf7ah1ep-111] et (b) un arc couvrant la racine [Aimé Césaire, Poésie, 2006].

dans le cas non projectif (algorithme de Chu-Liu-Edmonds en $O(n^2)$) que dans le cas projectif (algorithme de Eisner en $O(n^3)$) (cf. chapitre 5, section 5.2.1).

(Non-)Projectivité et arbres syntagmatiques

Les arbres de dépendances découplent relations de dépendances et ordre linéaire, et la projectivité est une contrainte supplémentaire pouvant s'ajouter au cas de base. À l'inverse, la formalisation la plus courante des arbres syntagmatiques correspond à un encodage projectif, où la projection lexicale d'un syntagme est une portion continue de la phrase.

Il existe cependant une formalisation alternative, plus rare, avec constituants discontinus pour représenter les phénomènes non locaux. C'est le cas par exemple du treebank TIGER pour l'allemand (Brants et al., 2002), comme illustré par la capture d'écran 3.3.

Une représentation alternative de la non-localité consiste à utiliser un système d'indices pour relier un noeud en position non locale avec un noeud vide (une trace), placé à la position locale d'origine supposée, et coïndicé avec le constituant effectivement non local. D'un point de vue linguistique, les traces sont apparues dès 1975 (Chomsky, 1975) dans le cadre de la grammaire générative, où les éléments non locaux sont considérés comme déplacés à partir d'une position locale. Ce type de représentation est utilisé dans le Penn Treebank. Par exemple en 3.4, le pronom interrogatif *What* est coindicé avec un noeud trace *T* apparaissant en position locale, i.e. à droite du verbe *eating* (cf. *What* correspond à l'objet direct de *eating*).

D'autres astuces ont été proposées pour représenter la non-localité sans recourir à une discontinuité ou un noeud vide, comme par exemple dans le treebank allemand TüBa/Z (Telljohann et al., 2004), qui explicite la fonction grammaticale d'un constituant plus un codage du constituant gouverneur, si celui-ci n'est pas la tête locale.

La non projectivité est rare

La non projectivité apparaît dans les langues à ordre libre, ou plus généralement en cas de "non localité" d'un élément, que nous détaillons infra section 3.4.2. Malgré une variation importante entre langues, il est communément admis que l'ordre des mots dans les langues

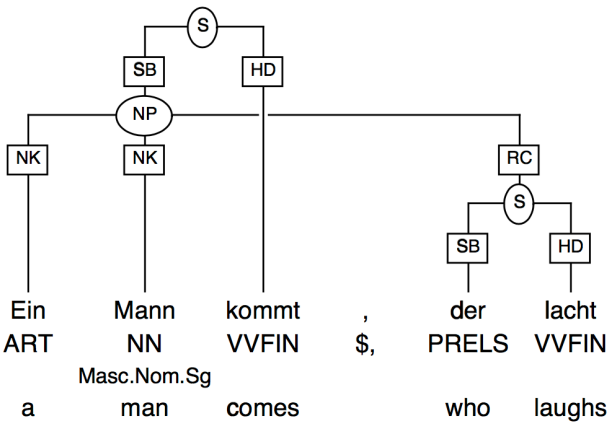

FIGURE 3.3 – Exemple de constituant discontinu dans le corpus allemand TIGER. Les noeuds syntagmatiques sont les cercles, et les arcs de dominance sont étiquetés avec une fonction grammaticale (rectangles). Par exemple le NP *Ein Mann der lacht* est discontinu. Il porte la fonction SB par rapport à la tête (HD) *kommt*. (source : (Brants et al., 2002)).

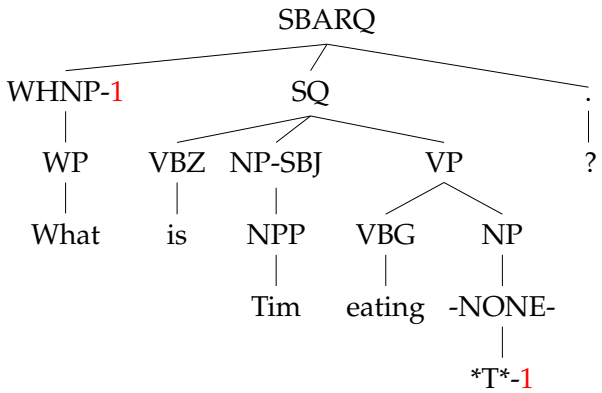

FIGURE 3.4 – Gestion de la non-localité dans le Penn Treebank : l'élément extrait (What) est coindicé avec un noeud trace *T* apparaissant en position locale, i.e. à droite du verbe *eating*, cf. *What* correspond à l'objet direct de *eating* (Source : (Marcus et al., 1994), avec rajout ici des noeuds prélexicaux).

a tendance à être projectif (Mel'čuk, 2009). Des travaux formels permettent de comparer la non-projectivité observée dans des corpus arborés à la non-projectivité moyenne attendue, en considérant des ordres de mots aléatoires, étant donnée une même structure de dépendances. La disponibilité, relativement récente, de corpus en dépendances pour de nombreuses langues, a en effet permis de vérifier empiriquement le faible nombre de croisements de dépendances : Ferrer i Cancho et al. (2018) montrent par exemple que dans les treebanks HamleDT 2.0 (Rosa et al., 2014), une collection de treebanks pour 30 langues, uniformisés dans deux schémas d'annotation (les Universal Stanford Dependencies (de Marneffe et al., 2014) et les Prague Dependencies (Hajic et al., 2006)), le nombre de dépendances se croisant (formant un des cas de non-projectivité, l'autre cas étant celui d'une dépendance couvrant la racine, cf. supra 3.2.1) est nettement inférieur au nombre obtenu en conservant les relations de dépendances, mais en réordonnant les mots selon plusieurs modèles aléatoires.

Or, Alemany-Puig (2019) démontre l'existence d'une forte corrélation entre nombre de dépendances croisées et longueur totale des dépendances. Ceci tend à prouver le lien entre la rareté des dépendances croisées (et donc presque de la non-projectivité) et un principe psycholinguistique de moindre effort cognitif, stipulant que les mots syntaxiquement reliés ont tendance à être linéairement proches [6]. Les arbres de dépendances ont l'avantage de permettre une formalisation facile de ce principe, par le principe dit de minimisation de la longueur des dépendances (DLM) [7], qui stipule qu'étant données les relations de dépendances entre les mots d'une phrase, l'ordre des mots observé dans les langues a tendance à minimiser la longueur totale des dépendances. Une vérification empirique sur 37 langues est par exemple fournie par Futrell et al. (2015), en comparant l'ordre observé et un ordre aléatoire des mots, pour des dépendances données.

3.2.2 Equivalence formelle et différences usuelles

En termes de pouvoir expressif, les constituants et les dépendances sont comparables. Prenons d'abord le cas de constituants continus (i.e. le cas par défaut de la définition des constituants). Pour obtenir des dépendances bilexicales à partir d'un constituant X, il faut isoler une **tête** au sein des éléments dominés par le noeud X, cette notion de tête pouvant être grossièrement définie comme l'élément déterminant le plus la distribution du syntagme [8].

Inversement, la notion de constituant peut être interprétée au sein d'un arbre en dépendances (ce que l'on trouve sous le nom de "noeud" chez Tesnières) : pour chaque mot w_i d'une phrase, la projection lexicale de w_i dans un arbre de dépendances forme l'équivalent d'un constituant, mais sans symbole syntagmatique. Mais les constituants obtenus ne seront

6. Pour ce principe général, de Marneffe and Nivre (2019) remontent à une publication en allemand de O. Behaghel en 1932.

7. Cette longueur est la somme des longueurs des dépendances d'un arbre de dépendances pour une phrase, chaque dépendance $i \xrightarrow{l} j$ ayant pour longueur le nombre de tokens entre i et j plus 1. Par exemple, Figure 3.1, la longueur de la dépendance entre *à* et *est* vaut $4 + 1 = 5$.

8. La notion de tête est présentée comme prototypique par Hudson (de Marneffe and Nivre, 2019), étant clairement définie pour certaines constructions, en particulier la valence verbale, au contraire d'autres où il y a concurrence entre têtes, comme la coordination par exemple. En TAL, il est intéressant de noter que le mécanisme d'auto-attention au sein de réseaux de neurones, proposé initialement en traduction séquence-vers-séquence (Vaswani et al., 2017) implémente une version distribuée de la notion de tête : une séquence n'est pas caractérisée par son unique tête, mais par une combinaison linéaire (des vecteurs) de tous les mots de la séquence, les poids étant appris de façon à optimiser une tâche donnée.

continus que si l'arbre de dépendances de départ est projectif. Pour le cas projectif, Gaifman (1965) a d'ailleurs proposé une formalisation des grammaires de dépendances, et montré une équivalence *forte* entre celles-ci et un certain type de grammaires hors-contexte, où toute partie droite de règle a un et un seul noeud lexical (la tête). Dans le cas non projectif cependant, l'équivalence ne vaut que si l'on admet la notion de constituant discontinu.

Dans le cas général donc, pour des arbres de dépendances potentiellement non projectifs et étiquetés, et des arbres de constituants potentiellement discontinus, on a trois différences en pratique :

— les arbres de constituants comprennent des symboles syntagmatiques, dont le nombre n'est pas borné si l'on admet la possibilité de syntagmes unaires ;
— si l'on peut voir un sous-arbre de dépendances comme définissant un constituant, la tête de celui-ci est forcément définie et unique, alors que dans un arbre de constituant, rien ne contraint à une définition unique de tête.
— les étiquettes de dépendances n'ont pas d'équivalent direct dans les arbres de constituants. Une représentation des fonctions est possible, en surchargeant les symboles syntagmatiques (par exemple NP-SBJ pour un NP sujet) et que l'on a une heuristique ou une annotation explicite de la tête de chaque syntagme (comme fait dans le treebank TIGER, comme illustré supra Figure 3.3).

3.2.3 Choix des arbres de dépendances

Comme vu supra, les différences entre les deux types de représentation apparaissent assez minimes, mais certaines informations sont plus naturellement encodées dans une représentation ou une autre. Pour ma part, j'ai plutôt privilégié l'utilisation des dépendances dans mon travail de recherche, je fais le point ci-dessous sur les motivations et difficultés posées par ce choix.

Absence de symboles syntagmatiques

L'absence de symboles syntagmatiques apparaît comme un manque de généralisation, un symbole syntagmatique permettant de signifier une similarité de distribution (la valence passive) pour des séquences ayant des structures internes différentes. Le moins qu'on puisse dire cela dit, est que cette question est largement débattue en linguistique, avec par exemple Hudson (1980b) montrant d'abord la nécessité du concept de dépendance (tête / dépendant), et arguant de l'absence de preuves de la nécessité du concept de constituant (en tous cas de relations partie-tout), puis admettant dans Hudson (1980a) la nécessité de celui-ci pour représenter les phénomènes de coordination, sous les critiques de Östen Dahl (1980).

En tout état de cause, cette absence de symboles syntagmatiques permet une simplification formelle, en donnant une structure dont le nombre de noeuds est exactement le nombre de mots de la phrase, par opposition au nombre à priori non borné de noeuds syntagmatiques. Cette propriété est largment utilisée par les analyseurs en dépendances.

Généralisation grâce aux fonctions grammaticales

Les fonctions grammaticales neutralisent des différences d'ordre des mots au sein d'une langue. Elles sont également traditionnellement utilisées en typologie, et pour des tentatives d'expressions d'universaux linguistiques.

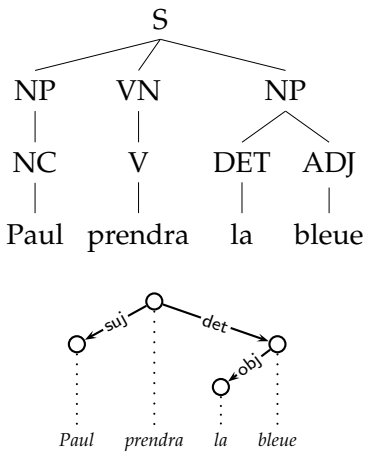

FIGURE 3.5 – Gestion d'une ellipse de nom en constituants et en dépendances : en dépendances, on choisit ici arbitrairement que la tête de *la bleue* est l'adjectif. On ne peut pas représenter que le tout a une distribution de groupe nominal.

L'utilisation explicite des fonctions au sein des arbres de dépendances est donc un puissant moyen d'exprimer des généralisations sur les relations entre un prédicat et ses arguments syntaxiques - au sein d'une langue et entre langues, au-delà des variations d'ordre, et de là de passer plus facilement à un typage sémantique de la relation entre un prédicat et ses arguments sémantiques (i.e. des rôles sémantiques, comme utilisé infra chapitre 8). Pour cette raison, les arbres de dépendances sont souvent présentés en TAL comme donnant accès de manière plus directe aux relations argumentales et aux structures prédicat-argument.

En comparaison, les treebanks en constituants pour langues configurationnelles se passent en général d'expliciter les fonctions grammaticales (cf. en particulier le Penn Treebank pour l'anglais sauf exceptions), et nécessitent une interprétation de la topologie des arbres pour repérer les structures argumentales.

Structures sans tête

L'encodage explicite de la tête dans un arbre de dépendances est problématique pour les constructions où la tête n'est pas clairement définie (Dipper and Kübler, 2017), comme par exemple les ellipses ou les relatives sans tête. Par exemple pour une ellipse du nom au sein d'un SN, comme dans *Anna prendra la bille rouge et Paul prendra la bleue*, au niveau syntagmatique, on peut représenter via un noeud SN que *la bleue* a bien une distribution de SN, malgré l'absence du nom tête, alors qu'en dépendances, il faut choisir arbitrairement une tête parmi *la* et *bleue* (cf. la Figure 3.5).

À noter que certains cas sont problématiques pour les deux types de représentations, comme par exemple les relatives sans tête. Il difficile de capturer à la fois la structure interne d'une relative, avec des dépendants d'un verbe conjugué, et la valence passive de type SN. La représentation qui en est fournie dans le FTB et sa conversion automatique en dépendances (cf. infra section 3.2), illustrée Figure 3.6 privilégie la valence passive de type SN.

Les structures coordonnées constituent un autre défi notoire pour la représentation syntaxique (en constituants ou en dépendances). Le nécessaire choix d'une tête en grammaires de dépendances donne deux grands types de représentation, abondamment comparées dans

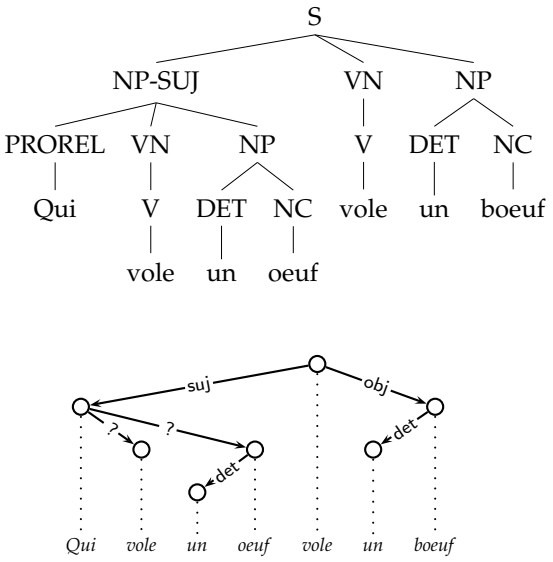

FIGURE 3.6 – Représentation (insatisfaisante) des relatives sans tête en constituants (dans le schéma FTBconst, cf. infra section 3.4.1) et en dépendances obtenues par conversion (cf. infra section 3.2).

la littérature, avec soit la conjonction de coordination comme tête, ou bien un des conjoints comme tête (en général le premier), et les autres conjoints en dépendant à plat ou bien en cascade (3ème conjoint dépendant du second, etc...). Ces deux types de représentations ont été abondamment comparées dans la littérature, et il est clair que les seules dépendances ne permettent pas de représenter toutes les données. Tesnière (1959) définissait d'ailleurs pour les conjoints une relation d'un type spécifique (la jonction), distincte de la dépendance. Dans le même ordre d'idée de séparer coordination et dépendance, on peut citer le formalisme des arbres à bulle (Kahane, 1997) où une sorte de constituant est utilisé pour le syntagme coordonné.

Si l'on s'en tient à des arbres de dépendances cela dit, pour les coordinations simples (deux conjoints liés par un coordonnant), les résultats en termes d'apprenabilité d'un analyseur donnent un clair avantage à choisir un des conjoints comme tête (Schwartz et al., 2012).

Ordre libre et non-projectivité

Les représentations en dépendance séparent l'ordre structurel et l'ordre linéaire, ce qui donne une grande souplesse de représentation. Les phénomènes non locaux sont traitables "en natif", simplement en ne posant pas la contrainte de projectivité.

Le principe linguistique de minimisation de la longueur des dépendances est simplement émis comme une tendance, en compétition avec d'autres phénomènes, en particulier relatifs à la structure informationnelle, pouvant rendre compte de l'existence de linéarisations non projectives. En comparaison, dans les arbres en constituants, les palliatifs cités supra section 3.2.1 pour gérer la non-projectivité complexifient indéniablement les structures.

3.3 Le corpus SEQUOIA : un nouveau corpus pour diversifier les domaines

Une fois ces repères fournis concernant les représentations syntaxiques et les corpus arborés, je décris ici brièvement la création du corpus Sequoia, réalisée au départ avec Djamé Seddah (voir (Candito and Seddah, 2012b) pour une description plus précise), dans le cadre du projet SEQUOIA (ANR-08-EMER-013, " Large coverage probabilistic syntactic parsing of French"), porté par Alexis Nasr.

Ce corpus a été initialement annoté pour la morphologie et la syntaxe en constituants, et a été automatiquement converti en arbres de dépendances. De nombreux contributeurs ont ajouté d'autres types d'annotation : syntaxe profonde (chapitre 6), cadres sémantiques FrameNet (chapitre 8), expressions polylexicales (chapitre 4), classes sémantiques (Barque et al., 2020).

Il s'agit d'un petit corpus de 3099 phrases, issues de quatre sources : l'agence européenne du médicament, Europarl, le journal régional *l'Est Républicain* et Wikipedia Fr, initialement conçu pour nous donner la possibilité, ainsi qu'à la communauté TAL française, d'investiguer l'"adaptation de domaine" d'analyseurs syntaxiques statistiques pour le français. Il s'agissait de pouvoir étudier comment adapter un analyseur syntaxique statistique entraîné sur un corpus donné (données "intra-domaine") à analyser des phrases de genre différent (données dites "hors-domaine"). Accessoirement, notre but était également de fournir un corpus arboré sous licence libre, disponible à toute autre fin.

Pour rester le plus compatible avec le FTB et pouvoir effectivement réaliser des expériences utilisant les 2 corpus, les annotations morphologiques et syntaxiques en constituants ont été faites en suivant, à quelques exceptions près, le schéma d'annotation et les guides d'annotation du FTB (Abeillé and Clément, 1999 ; Abeillé et al., 2004 ; Abeillé, 2004). Les annotations ont été faites par la méthode classique de double annotation en aveugle plus adjudication des conflits. L'accord inter-annotateur obtenu est plutôt bon et valide la qualité des annotations.

Pour obtenir les arbres de dépendances, les annotations en constituants sont ensuite converties par notre procédure automatique de conversion, décrite infra section 3.2. À l'arrivée des Universal Dependencies, Bruno Guillaume a pris en charge la conversion automatique vers le schéma d'annotation des Universal Dependencies (Guillaume et al., 2019). On obtient ainsi un corpus arboré dans le schéma en constituants du FTB, dans le schéma en dépendances FTB-dep, et dans le schéma Universal Dependencies, ce qui permet une description fine adaptée au français, ainsi qu'une description adaptée aux traitements multilingues.

3.3.1 Caractéristiques

On donne table 3.1 quelques caractéristiques des différents sous-corpus annotés, en regard de celles des corpus de développement et d'entraînement du FTB (FTB-dev et FTB-train) utilisés pour les expériences d'adaptation de domaine (cf. section 5.1.3)[9].

9. Nous utilisons ici la version du FTB et son découpage du FTB tel que défini dans (Crabbé and Candito, 2008) : il s'agit de la partie du FTB annotée en fonctions grammaticales, telle que distribuée en 2007, qui contient 12351 phrases, découpées en 1235 phrases de test, 1235 phrases de développement et 9881 phrases d'entraînement, dans cet ordre. Depuis, l'annotation des fonctions grammaticales a été complétée pour environ 21000 phrases, et la version 1.0 a pu être livrée en 2017.

	Corpus Sequoia					FTB	
	Médical		Neutre				
	EMEA dev	**EMEA test**	**Est Rép.**	**Euro Parl**	**Fr Wiki**	**dev**	**train**
Nb de phrases	574	544	529	561	996	1235	9881
Longueur moyenne	16,3	22,0	21,0	26,3	22,2	29,6	28,1
Ecart type sur la longueur	14,7	15,0	12,9	15,0	18,0	16,0	16,5
Données pour tout type de formes fléchies (y compris ponctuation)							
Taille du vocabulaire	1916	1737	3337	3300	4687	7222	24110
% d'inconnus	41.4	35.8	29,2	20,6	34,2	22,5	-
Nb d'occ.	9343	11964	11114	14745	22080	36508	278083
% d'occ. d'inconnus	23.0	19.7	11,2	6,6	12,9	5,2	-
% d'occ. de Noms propres	1,7	2.7	5,1	2,9	9,7	4,1	4,0

TABLE 3.1 – Caractéristiques chiffrées des corpus manuellement annotés. Les *inconnus* sont les formes absentes du FTB-train.

Les différents sous-corpus ont chacun environ 500 phrases, sauf FrWiki (961 phrases). À noter que les phrases sont en moyenne moins longues que dans le FTB. On peut constater que le corpus médical comporte de loin le vocabulaire le plus éloigné de celui du FTB (plus d'une forme sur trois est absente du FTB-train). Pour le corpus FrWiki, la forte proportion d'inconnus (34,2%) peut s'expliquer par une grande fréquence des noms propres (cf. la ligne *% d'occurrences de noms propres* : environ une occurrence sur 10 est un nom propre dans FrWiki).

Les lignes sur les nombres d'occurrences et le pourcentage d'inconnus parmi ces occurrences donnent une vision plus précise de la diversité lexicale des corpus. Dans les corpus médicaux, une occurrence sur 5 (et presque une sur 4 pour EMEA-dev) correspond à un inconnu du FTB-train, ce qui, avec la faible proportion d'occurrences de noms propres (1,7 et 2,7) indique que les mots inconnus sont plutôt des mots fréquemment utilisés dans ces corpus. Au contraire, pour FrWiki on voit que, calculée sur les occurrences, la proportion d'inconnus tombe à 12,9 (la majorité des inconnus du vocabulaire sont des noms propres, apparaissant rarement). Le corpus le plus proche lexicalement du FTB semble être Europarl : seulement 6,6% des occurrences sont des inconnus, formant un cinquième du vocabulaire, ce qui de manière surprenante constitue une moindre proportion d'inconnus que dans le FTB-dev.

3.4 Conversion des constituants vers les dépendances

Cette section décrit un travail en collaboration avec Benoît Crabbé, Mathieu Falco et François Guérin.

C'est dans le contexte d'un engouement pour les corpus arborés en dépendances, évoqué section 3.2 que nous avons travaillé, en 2009, à la conversion semi-automatique du FTB, depuis les arbres en constituants vers des arbres en dépendances. Nous désignons par FTBdep le schéma d'annotation en dépendances résultant [10].

10. Le schéma résultant est décrit en ligne http://alpage.inria.fr/statgram/frdep/Publications/FTB-DescriptionDepSurface.pdf.

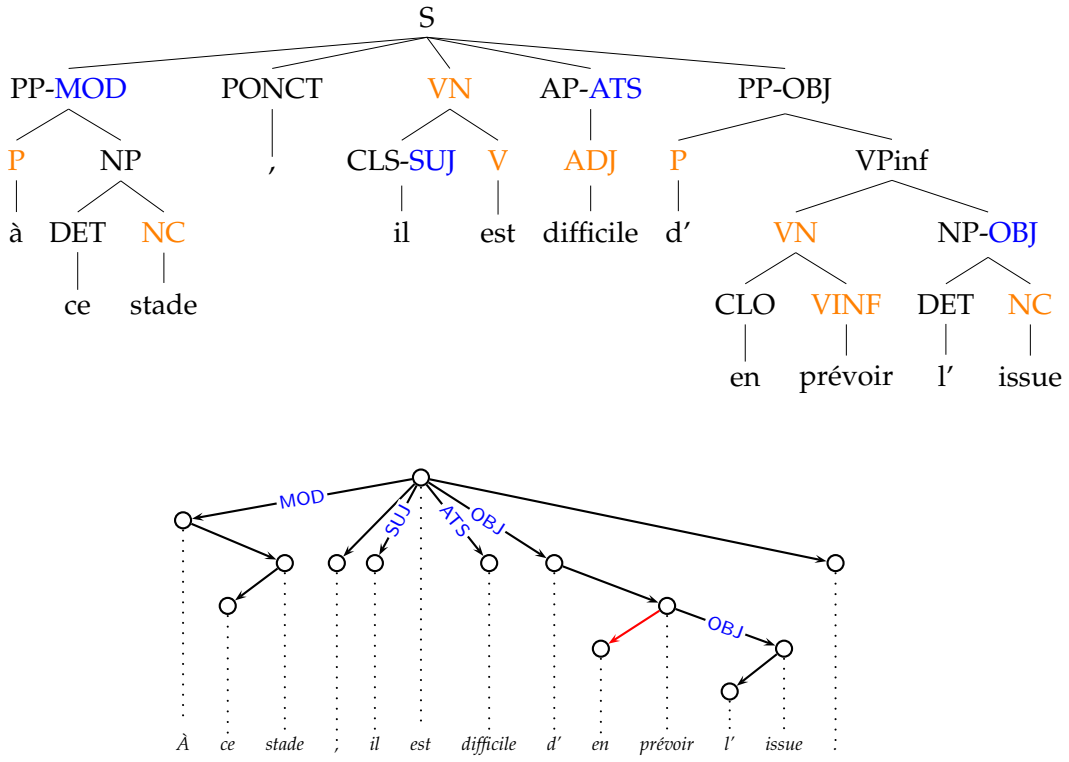

FIGURE 3.7 – **Haut** : Arbre d'entrée de la conversion automatique en dépendances (schéma FTBcont-predep). Les étiquettes fonctionnelles sont en bleu. Au sein de chaque syntagme, le noeud fils tête est indiqué en orange. **Bas** : Arbre en dépendances projectif résultant, avant prédiction des étiquettes manquantes, et hors correction des dépendances non locales décrite section 3.4.2 (la dépendance en rouge étant ici fausse).

Nous décrivons d'abord la partie automatique produisant des arbres de dépendances projectifs, puis la gestion des dépendances non locales, aboutissant à des arbres non projectifs est détaillée section 3.4.2.

3.4.1 Conversion vers arbres projectifs

Pour obtenir un treebank en dépendances pour le français, nous partons du FTB (Abeillé and Barrier, 2004), seul corpus avec annotation complète en constituants à l'époque disponible. Son schéma d'annotation syntagmatique est défini dans le guide d'annotation syntaxique du FTB (Abeillé et al., 2004) [11]. On donne un exemple Figure 3.7 (haut). Les fonctions grammaticales, annotées manuellement dans FTB-XML sous forme de traits pour les dépendants de verbes conjugués sont concaténées à la catégorie morpho-syntaxique ou syntagmatique du noeud (comme par exemple les suffixes -SUJ et -OBJ en bleu dans l'arbre de la Figure 3.7).

11. Nous partons plus précisément d'une version parenthésée, initialement obtenue par Benoît Crabbé (Crabbé and Candito, 2008), où du schéma initial du FTB (ci-après FTB-XML) est dérivé un schéma avec 28 catégories morphosyntaxiques (obtenues par combinaison des 13 catégories morpho-syntaxiques grossières du FTB-XML, et de traits morpho-syntaxiques) (ci-après **schéma FTBconst-dep**)

Conversion récursive vers un arbre de dépendances projectif

La conversion peut se faire de manière récursive, pour peu que pour chaque syntagme, on dispose du moyen d'identifier un unique noeud fils comme étant la tête du syntagme.

On donne l'algorithme en 1, où on suppose une structure d'arbre `ctree` (un noeud ayant accès à ses noeuds fils, children(ctree), s'ils existent, dont son noeud fils tête (`head_child(ctree)`). gram_function(child) récupère la fonction grammaticale du noeud si elle y est annotée, et une étiquette vide dans le cas contraire.

Algorithme 1 Conversion récursive d'un arbre en constituants vers un arbre de dépendances projectives

```
1 : convert(ctree) :
2 : if is_leaf(ctree) then
3 :    return ctree
4 : else
5 :    h_child ← head_child(ctree)
6 :    h_dtree ← convert(h_child)
7 :    for all child in children(ctree) / child ≠ h_child do
8 :       d_dtree ← convert(child)
9 :       add_edge in h_dtree from root of h_dtree to root of d_dtree, with label
          gram_function(child))
10 :   end for
11 :   return h_dtree
12 : end if
```

La Figure 3.7 illustre le résultat obtenu après cette phase. La dépendance non locale entre *en* et *issue* ne peut pas être gérée par cette procédure automatique. Nous quantifierons ces cas section 3.4.2, à partir d'annotations manuelles.

Heuristiques de prédiction des étiquettes de dépendances manquantes

Le schéma du FTB ne prévoit des fonctions grammaticales que pour certains noeuds, correspondant en gros aux dépendants des verbes (cf. le guide d'annotation des fonctions du FTB (Abeillé, 2004)). Après cette étape, il reste donc des dépendances à étiqueter, ce qui est fait via des heuristiques syntaxiques et lexico-syntaxiques simples. À noter que la distinction entre argument et ajouts n'est pas faite dans le FTB initial pour les dépendants du nom, et nous avons décidé de ne pas trancher non plus au sein des arbres de dépendances, car cela aurait forcément nécessité une phase importante d'annotation manuelle, des heuristiques paraissant trop hasardeuses dans ce cas. Ainsi une même étiquette sous-spécifiée *dep* est utilisée pour la préposition *à* dans *l'obéissance à la loi* et dans *l'obéissance à midi*.

Choix des têtes des noeuds syntagmatiques

Pour identifier automatiquement la tête de chaque syntagme, on utilise des *règles de tête* (Magerman, 1995), non lexicalisées, i.e. n'utilisant que la catégorie du noeud C et de ses noeuds fils. Une telle règle s'écrit par exemple "pour un noeud NP, choisir comme tête le premier nom en partant de la gauche, ou à défaut le premier pronom, ou à défaut " etc...

La tête d'un syntagme est en général définie par le type du syntagme, et donc la plupart des choix de tête découlent du schéma d'annotation en constituants du FTB. Nous avons simplement fait quelques modifications, pour uniformiser le traitement des complémenteurs et des prépositions.

Évaluation sur le FTB

Pour évaluer la qualité des dépendances produites automatiquement, en particulier la qualité des heuristiques pour les étiquettes manquantes, et pour quantifier les dépendances non locales, nous avons corrigé manuellement la conversion de 120 phrases consécutives du FTB, représentant environ 3000 mots sans compter la ponctuation (Candito et al., 2009). On obtient, hors ponctuation, 98.78% des tokens, hors ponctuation, qui reçoivent le gouverneur correct (score de type UAS pour *unlabeled attachment score*) et 98% des tokens qui reçoivent le gouverneur correct avec la bonne étiquette de dépendance (score de type LAS pour *labeled attachment score*).

Les erreurs proviennent d'erreurs dans la source ou de manque de précision des règles de tête ou des heuristiques d'étiquetage des dépendances. Cela permet de conclure à une bonne qualité globale du corpus en dépendances résultant.

3.4.2 Dépendances non locales

Cette section résume un travail réalisé avec Djamé Seddah (Candito and Seddah, 2012a), concernant la typologie, l'annotation et la quantification de certaines dépendances non locales, dans les corpus Sequoia et FTB, dépendances qui ne peuvent pas être récupérées via la conversion détaillée supra. Nous souhaitions quantifier plus précisément la quantité de non-projectivité occasionnée. En effet, la majorité des algorithmes d'analyse syntaxique automatique, testés en premier lieu sur l'anglais, sont prévus pour ne produire que des arbres projectifs, une contrainte qui a l'avantage de restreindre drastiquement l'espace de recherche que doit considérer un analyseur. Il est donc important d'évaluer quelle quantité d'arcs non projectifs seront de facto non prédits si l'on utilise par commodité un analyseur projectif.

En l'absence de traces ou constituants discontinus dans les arbres de constituants du FTB et le Sequoia, nous avons vu que la conversion automatique vers arbres de dépendance pouvait donner des dépendances erronées. Celles-ci relèvent du phénomène linguistique d'"extraction", omniprésent dans la littérature linguistique, chaque théorie linguistique devant rendre compte des contraintes, variables d'une langue à l'autre, sur quels éléments peuvent être en position extraite. Mais, en passant à une représentation en dépendances, seules certains cas d'extraction posent problème à la conversion.

Cas d'extraction causant des dépendances non locales

Le terme d'extraction, issu de la grammaire transformationnelle, perdure pour désigner une position non locale, "extraite" depuis une position canonique. En français, un premier type d'extraction concerne l'antéposition d'un des dépendants d'un verbe, avec quatre principales constructions : topicalisation (1), relative (2), question (3), clivée (4), dans lesquels l'élément extrait (antéposé) (en italique) dépend d'un verbe (en gras) :

(1) *À nos arguments*, (nous savons que) Paul **opposera** les siens.

(2) Je connais l'homme *que* (Jules pense que) Lou **épousera**.

(3) Sais-tu *à qui* (Jules pense que) Lou **donnera** un cadeau ?

(4) C'est Luca *que* (Jules crois que) Lou **épousera**.

Bien que relativement rare dans les textes, l'extraction est omniprésente dans la littérature linguistique du fait de son caractère non borné : en termes syntagmatiques, au sein de la clause contenant l'élément extrait e, il n'y a dans certains cas pas de limite à la profondeur du syntagme dont est extrait e (cf. par exemple *Je connais la femme que Marie dit que Paul dit que Jean dit ... que Pierre a vu.*. Mais en ignorant les syntagmes et en se concentrant sur les dépendances bilexicales, on peut cependant distinguer deux cas, selon que :

(i) l'antéposition d'un élément e se fait en conservant la même tête lexicale. Ces cas correspondent aux exemples (1) à (3), sans le matériel entre parenthèses. Par exemple pour (1), l'élément *À nos arguments* dépend toujours de *opposera*, qu'il soit en position locale postverbale ou bien antéposé. Dans ce cas la conversion automatique en dépendances décrite supra donnera un résultat correct.

(ii) ou bien e est extrait hors du domaine de localité de sa tête. C'est le cas dans toutes les versions des exemples (1) à (4) contenant le matériel entre parenthèses. La conversion automatique rattache l'élément extrait à une "tête locale" erronnée. Nous parlerons de **dépendance non locale** pour ce cas seulement.

D'autres cas de dépendances non locales concernent l'extraction hors d'un syntagme adjectival ou nominal. Hors d'un adjectif attribut du sujet, l'élément extrait peut être relativisé (5), questionné (6) ou cliticisé (7). Ces cas donnent systématiquement une dépendance non locale.

(5) la mélancolie *à laquelle* (on sait bien qu') il est **enclin**

(6) On sait *à quels jeux* (tout le monde croit) qu'il est **enclin**.

(7) Quelles raisons peuvent expliquer que l'Inde interdise la cigarette électronique, et que d'autres pays comme les USA *y* sont **enclins** ? (blog médiapart, sept. 2019)

Le complément d'un nom peut également être extrait, sous certaines conditions (Godard and Sag, 1996)[12]. On peut avoir par exemple relativisation ou cliticisation d'un complément hors d'un groupe nominal sujet (8), objet (9), attribut du sujet (10). Ces cas donnent également toujours une dépendance non locale.

(8) (a) une personne *dont* (on sait que) le **nom** n'a jamais été prononcé.
(b) Les **conséquences** *en* sont aujourd'hui connues.

(9) (a) une personne *dont* (Paul espère que) Laure connait le **nom**
(b) Nous *en* examinerons les **causes**.

(10) (a) les victimes *envers lesquelles* (on sait bien que) ce film est une **insulte**
(b) ce film *en* est la **preuve**.

Annotation et analyse quantitative

En pratique, une recherche exhaustive de la non localité étant hors de notre budget, nous avons choisi de cibler les mots indices typiques d'extraction, i.e. les pronoms relatifs et interrogatifs, et le clitique *en*. Parmi les cas de non localité cités supra, ce choix exclut de fait

12. Selon (Godard and Sag, 1996), seul le complément "premier" d'un nom peut être extrait.

les cas de topicalisation (exemple (1)), a priori très rares (1 cas détecté dans les 120 phrases d'évaluation citées supra section 3.4.1).

Ces occurrences d'indices ont été doublement annotées et adjudiquées [13]. L'annotation a été faite sur les arbres en constituants, de manière interprétable par la procédure de conversion en dépendances (utilisation de chemins fonctionnels, non détaillés ici).

Le résultat permet de quantifier les cas de non localité effective dans le contexte d'une extraction, pour du texte journalistique (FTB) et pour des genres un peu plus variés (corpus SEQUOIA). Les comptages sont fournis Table 3.2. La première constatation est que l'extraction donnant lieu à non localité effective est très rare, avec en tout 0.16% des tokens du FTB + SEQUOIA ayant un gouverneur non local.

Concernant la longueur des dépendances non locales, plutôt qu'une distance linéaire, on peut considérer le chemin dans l'arbre de dépendances entre l'élément non local, et la tête locale (i.e. le gouverneur qu'il aurait dans le cas d'une conversion strictement projective). On constate que dans plus de 80% de ces cas, le chemin fonctionnel est de longueur 2, c'est-à-dire qu'il s'agit de dépendances "pas si longues".

	Nombre de tokens					
Corpus	total	non locaux (%)	lcf=2	lcf=3	lcf>3	non projectif
FTB	350931	555 (0.16 %)	466	69	20	317
SEQUOIA	69238	63 (0.09 %)	47	13	3	42
FTB + SEQUOIA	420169	618 (0.15%)	513	82	23	359

TABLE 3.2 – Quantification de la non localité dans le FTB et le SEQUOIA : nb total de tokens, nb de tokens avec dépendance non locale (i.e. avec longueur de chemin fonctionnel (lcf) > 1), avec lcf=2, avec lcf=3 et lcf > 3. Dernière colonne : nb de tokens avec lcf>1 donnant lieu à non-projectivité.

Concernant la projectivité, les dépendances non locales ne donnent pas forcément lieu à une dépendance non projective. Par exemple, en cas d'extraction hors du NP sujet, si l'élément extrait est adjacent au sujet, la dépendance entre les deux sera projective. C'est le cas en (a) de la Figure 3.8, mais pas en (b).la dépendance non locale est non projective dans un peu plus de la moitié des cas seulement (dernière colonne Table 3.2).

Les mots le plus fréquemment non locaux (Table 3.3) sont les relatifs *que* et *dont*, et le clitique *en*, qui totalisent 572 des 618 dépendances non locales annotées [14]. Près de la moitié des *dont*, et environ 15% des *que* relatifs, et des *en* clitiques ont une dépendance non locale.

Concernant les dépendants non locaux de verbe, le relatif *que* est le plus fréquent. À noter que tous les cas trouvés de non-localité de *que*, l'extraction est faite hors d'une infinitive (comme dans *les dangers qu' elle peut* **receler**) et pas hors d'une clause [15].

Pour le relatif *dont*, environ une occurrence sur deux est non locale, et pratiquement tous les cas des *dont* non locaux sont des extractions hors de SN sujet [16].

13. À l'époque nous n'avions malheureusement pas calculé d'accord inter-annotateur.
14. Les autres cas sont essentiellement des syntagmes interrogatifs non locaux.
15. Nous n'avons trouvé aucun cas d'extraction hors d'une clause dans le FTB, et un cas incertain dans le sequoia, avec la relativisation de tout un SP *pour lesquelles*, dont le rattachement non local est douteux : *cette affaire s'inscrit dans la ligne des affaires précédentes pour lesquelles le Parlement a estimé que l'activité politique devait être protégée (sequoia-Europar.550_00551).*
16. Comme constaté également par Abeillé et al. (2016) sur le FTB et un corpus oral, cela contredit la contrainte de l'îlot nominal et la contrainte de l'îlot sujet de la grammaire générative. Ces autrices font

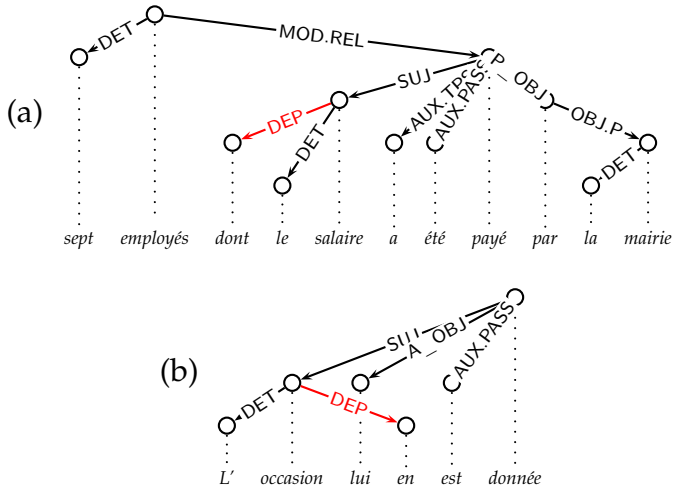

FIGURE 3.8 – Extraction hors d'un NP sujet : **(a)** avec projectivité conservée, **(b)** avec non projectivité.

L'extraction du clitique *en* est globalement moins fréquente, mais on a dans ce cas une nette majorité d'extraction hors du SN objet (50 cas sur 90), en particulier de SN objet d'un verbe infinitif (23 cas sur les 50).

		Nombre d'occurrences dans FTB + SEQUOIA				Exemple
Lemme	POS	total	non locales	3 chemins fonctionnels les plus fréquents		
que	pronom relatif	960	154 (16,0%)	OBJ/OBJ OBJ/OBJ.P/OBJ OBJ/OBJ.P/DE_OBJ	75 22 15	les dangers *qu'* elle peut **receler** le débat *que* vous nous demandez d'**ouvrir** les politiques *qu'* il convient d'**appliquer**
dont	pronom relatif	702	328 (46,7%)	DEP/SUJ DEP/OBJ DEP/ATS	248 30 26	sept employés *dont* le **salaire** a été payé *dont* elle contrôlait les **activités** *dont* il était **familier**
en	clitique	600	90 (15,0%)	DEP/OBJ DEP/ATS DEP/SUJ	50 23 11	afin d'*en* améliorer l'**efficacité** les pays qui *en* sont **victimes** un parfait **exemple** *en* a été fourni

TABLE 3.3 – Analyse des trois mots étant le plus fréquemment un dépendant non local (en absolu) dans le FTB et le SEQUOIA : nombre total d'occurrences, nombre d'occurrences non locales, trois chemins fonctionnels les plus fréquents pour chaque mot, avec un exemple dans chaque cas. Chemin fonctionnel : par exemple "DEP/ATS" signifie que l'élément extrait *dont* est le dépendant (DEP) de l'attribut (ATS) de la tête locale *était*.

une étude contrastive de l'emploi de *dont* entre le FTB (corpus journalistique) et un corpus oral, dont les conclusions sont compatibles avec la théorie de la localité de Gibson (2000). Il ressort que l'extraction du *dont* est d'autant plus facile que la distance entre celui-ci et le syntagme dont il est extrait est "courte", en termes de nombre de nouveaux référents intervenant entre les deux. Ceci prédit un coût nul pour l'extraction hors d'un SN sujet préverbal, et pour l'extraction d'un complément de tout autre chose que le sujet, si le sujet est anaphorique (et n'introduisant donc pas de nouveau référent). Effectivement, sur les 56 cas d'extraction hors d'un SN objet ou attribut, on trouve 31 cas avec sujet clitique ou pronominal.

3.5 Bilan

Nous avons présenté dans ce chapitre des considérations sur les corpus arborés, et présenté la création du corpus SEQUOIA, annoté en constituants en suivant le schéma d'annotation du FTB. Nous avons également présenté la production semi-automatique d'arbres de dépendances pour ces deux corpus, avec un traitement adapté et une étude quantitative de certains cas de dépendances non locales. Seule une partie de celles-ci donnent lieu à non-projectivité. Le FTB en constituants et en dépendances a été utilisé en particulier pour les compétitions internationales SPMRL "Statistical Parsing Morphologically Rich Languages" (Seddah et al., 2013). Le SEQUOIA sort du seul genre journalistique, et permet des expériences d'adaptation de domaine d'analyseurs syntaxiques, résumées dans le chapitre 5.

Chapitre 4

Annotation et identification d'expressions polylexicales (résumé)

Ce chapitre résume des travaux sur les expressions polylexicales, auxquels j'ai participé en collaboration avec Mathieu Constant, Carlos Ramisch, Agata Savary, et Hazem Al Saied (doctorant co-encadré par Mathieu et moi-même). Ces travaux ont été faits pour la plupart dans le cadre du projet COST PARSEME, porté par Agata Savary, et du projet ANR PARSEME-FR, porté par Mathieu Constant.

Les expressions polylexicales (ci-après EP), telles que *pomme de terre*, *tout à coup*, *prendre une décision*, *se comporter* ou *avoir l'air*[1], sont des objets linguistiques constitués d'au moins deux composants (i.e. des éléments pouvant se comporter comme un mot dans d'autres contextes, ou bien des éléments ne pouvant apparaître de manière autonome, comme *fi* dans *faire fi*), mais dont la composition ne suit pas les règles syntaxiques et/ou sémantiques productives d'une langue donnée (Gross, 1986)[2]. Cette irrégularité amène à considérer ces expressions comme des "unités linguistiques", qu'elles soient totalement figées et insécables (comme *tout à coup*), elles peuvent alors être considérées comme des mots, ou bien qu'elles admettent des variations et insertions (comme l'expression verbale *prendre part*, dans *ils ne prirent pas part aux travaux / ils prirent une part active aux travaux*), auquel cas les considérer comme un seul mot ne permet pas de rendre compte de la part de régularité dans leur comportement morphologique et syntaxique.

L'idiosyncrasie définitoire des EPs en font un défi bien connu pour la modélisation linguistique d'une part, et pour le traitement automatique des langues d'autre part. Côté modélisation linguistique, le périmètre exact et la caractérisation des EPs ne fait pas consensus. Côté traitement automatique des langues, l'irrégularité des EPs, peu importe le niveau auquel elle se manifeste, perturbe les traitements automatiques. C'est ce qui a fait émerger la tâche d'identification des EPs : repérer au sein d'un texte l'occurrence d'EPs, au préalable ou conjointement à d'autres tâches est sensé faciliter ces tâches. Par exemple, repérer au sein de *il ne vend pas de phares longue portée*, la séquence *longue portée* fonctionne comme un adjectif permet

1. Pour citer une EP, nous adoptons la convention de souligner les composants de l'EP elle-même, à côté d'éléments ouverts, notés simplement en italiques : *il a l'air pressé, ils n'ont pas l'air mécontents*. On utilisera des identifiants lorsque plusieurs EPs se chevauchent.

2. Certains travaux incluent également des expressions, parfois appelées collocations, dont la combinaison est régulière sur les plans morphologique, syntaxique et sémantique, mais qui ont une fréquence anormalement élevée (on parle alors d'idiosyncrasie statistique). Dans tout ce chapitre, nous n'incluons pas dans le champ des EP les expressions n'étant idiosyncratique que sur le plan statistique.

de se ramener à une séquence régulière de catégories, et une structure syntaxique régulière. En outre, les EPs pouvant avoir un sens entièrement ou partiellement non compositionnel, leur identification est une condition nécessaire pour toute tâche d'ordre sémantique. Ainsi par exemple, repérer qu'au sein de "*ils ne jettent pas la pierre aux grévistes*", on a un prédicat "*jeter la pierre*", à deux arguments sémantiques, est nécessaire à l'analyse sémantique de la phrase.

L'identification des EPs pose des difficultés variées (voir par exemple (Constant et al., 2017) pour un panorama). Une première concerne l'ambiguïté entre lecture littérale et lecture idiomatique : par exemple Ramisch et al. (2016) ont évalué que la séquence *bien que* est idiomatique (comme dans *Elle parle bien qu'elle dorme*) dans 37% des cas seulement, versus 63% d'occurrences littérales (comme par exemple dans *Il semble bien qu'elle dorme*).

La seconde difficulté est que le figement est un phénomène à la fois fréquent mais difficilement prévisible, ce qui rend l'identification tributaire de ressources, soit des corpus annotés en EPs, soit des lexiques d'EPs. Mais ces ressources ne s'avèrent jamais assez couvrantes, ce qui amène Constant et al. (2017) à distinguer la tâche d'identification d'EPs de la tâche de "découverte d'EPs". La première consiste à repérer en corpus des occurrences effectivement idiomatiques d'EPs connues (soit dans un corpus annoté d'entraînement, soit dans un lexique). La seconde consiste à repérer des EPs vues ni à l'apprentissage ni dans un lexique.

Dans ce domaine ma contribution a porté sur la production de guides d'annotation et de ressources annotées, et sur l'identification d'EPs conjointement ou pas à l'analyse syntaxique.

- On obtient l'annotation complète des EPs au sein du corpus SEQUOIA :
  - Cela a commencé pour les EPs verbales, dans le cadre du projet COST PARSEME. La particularité de cette campagne a été de viser un guide d'annotation fonctionnant pour 20 langues, typologiquement variées (Savary et al., 2018).
  - Les membres du projet PARSEME-FR ont étendu, pour le français, la couverture à tout type d'EP, avec comme particularités (Candito et al., 2020) :
    - D'une part un effort pour clarifier la distinction entre entités nommées et les autres types d'EPs, fondée sur le type de convention de nommage permettant de récupérer le référent d'une expression.
    - D'autre part, on acte le degré variable de figement au sein d'EPs, en utilisant exclusivement des critères suffisants et non pas nécessaires. Une EP annotée peut satisfaire un nombre variable de critères suffisants.
    - Enfin, on découple si besoin l'annotation du statut d'EP et celle de la structure syntaxique, qui peut rester régulière pour certaines EPs. Par exemple dans une EP comme *arme blanche* ou *jeter son/le dévolu (sur qqch)*, on peut retrouver une structure interne régulière et une distribution externe régulière, ce qui n'est pas le cas par exemple de *en vain*.
- Concernant l'identification automatique d'EPs, citons :
  - le travail conjoint avec Mathieu Constant, sur la mise à profit pour l'analyse en dépendances, d'une représentation syntaxique interne régulière pour certaines EPs (Candito and Constant, 2014) ;
  - ainsi que le co-encadrement, toujours avec Mathieu, d'Hazem Al Saied, qui a mis au point en particulier un identificateur d'EPs, qui utilise une analyse par transitions, prédites grâce à un classifieur neuronal (Saied, 2019)

Chapitre 5

Analyse syntaxique automatique

La première section de ce chapitre décrit des travaux sur le parsing en constituants, intra-domaine et hors-domaine, menés avec Benoît Crabbé, Djamé Seddah et Enrique Henestroza Anguiano. Je décris ensuite section 5.2 une partie du travail de thèse d'Enrique, dont j'ai été l'encadrante principale, concernant la correction automatique d'arbres de dépendances.

5.1 Analyse en constituants : généralisation de mots et adaptation de domaine

Je suis arrivée à Alpage en 2007, où Benoît Crabbé avait commencé à travailler au parsing statistique en constituants pour le français, en fournissant une version opérationnelle du FTB, et en s'intéressant à l'analyseur phare du moment, celui des PCFG avec annotations latentes (PCFG-LA) de Petrov et al. (2006) (cf. section 5.1.1). Ma contribution porte sur la représentation des composés et l'amélioration de la robustesse des parsers appris face aux mots rares ou inconnus du corpus d'apprentissage, en utilisant des symboles généralisant les mots (des clusters de mots). Avant de détailler cette partie sur les généralisations de mots, je commence par resituer ces travaux en décrivant brièvement l'algorithme PCFG-LA et les idées qui ont permis d'y aboutir. Un aperçu des techniques plus actuelles de parsing sera donné au chapitre 7, concernant le parsing vers graphes de dépendances.

5.1.1 Contexte : le parsing statistique au milieu des années 2000 : markovisation, lexicalisation, annotations latentes

La tâche d'analyse syntaxique a longtemps été abordée séquentiellement, avec d'abord la production d'un ensemble d'analyses possibles, via des analyseurs symboliques, puis la désambiguisation syntaxique, i.e. le choix d'une seule analyse. L'analyse syntaxique probabiliste a pris rapidement le pas sur les analyseurs symboliques à partir du début des années 90, précisément parce qu'elle intègre les deux étapes : en associant un score aux analyses, la désambiguisation devient simplement la sélection de la ou des analyses de score maximal. Utilisant initialement des grammaires hors-contexte probabilisées (PCFG), extraites de corpus arborés, des gains importants de performance ont été obtenus d'une part en spécialisant les symboles syntagmatiques en fonction de leur contexte, affaiblissant ainsi les hypothèses d'indépendance inhérentes aux règles hors-contexte, et d'autre part en généralisant les règles, pour garantir une estimation plus fiable de leur probabilité.

Spécialisation des règles hors-contexte

Johnson (1998) a proposé d'acoler à tout symbole syntagmatique celui de son noeud père, capturant ainsi un pseudo-contexte, et il a obtenu des gains spectaculaires pour l'anglais sur le PTB. Klein and Manning (2003) illustrent l'intuition sous-jacente avec l'exemple des NP sujets versus les NP objets (dont la différence est bien capturée par l'annotation du symbole père, car les objets sont au sein de noeuds VP) : les NP sujets ont 8 fois plus de chances que les NP objets dans le PTB d'être réduits à un pronom. La différence sujet/objet est capturée de manière configurationnelle dans le PTB, les NP sujets et objets n'ont pas le même type de noeud père, cf. l'objet est au sein d'un VP [1].

D'autre part, dans la technique dite des PCFG *lexicalisées* (Magerman, 1995 ; Collins, 1999), les symboles syntagmatiques sont spécialisés, avec l'ajout systématique de leur mot tête. En effet, en définissant au sein de chaque syntagme quel est son noeud fils tête, on peut faire remonter ("percoler") dans l'arbre syntagmatique la tête lexicale de chaque syntagme (c'est le principe des formalismes linguistiques comme HPSG, où la grammaire est guidée par les têtes). On donne le mécanisme précis infra section 3.4.1. Si on choisit en particulier d'utiliser comme mots têtes les têtes lexicales plutôt que fonctionnelles, on peut voir figure 5.1 comment spécialiser les symboles d'après leur tête lexicale peut aider les rattachements syntagmatiques, et en particulier fournir l'information cruciale pour résoudre les ambiguïtés artificielles de rattachement prépositionnel.

Généralisation des règles hors-contexte

À l'inverse, les règles sont trop dispersées en général, et l'estimation de leur probabilité par fréquence relative est alors peu fiables. Par exemple pour un corpus à schéma d'annotation assez plat comme le FTB, on compte environ 13700 règles distinctes, dont 9800 n'apparaissent qu'une seule fois dans le corpus [2]. La spécialisation des symboles décrites supra vient aggraver ce problème. Aussi une autre technique s'est-elle imposée, la markovisation horizontale, qui au contraire enlève du contexte et généralise les règles : un syntagme d'origine avec n noeuds fils est binarisé en un sous-arbre de profondeur n, introduisant à chaque profondeur un seul des n fils. Les symboles artificiels introduits peuvent aussi bien conserver toute l'information initiale, ou bien "oublier" certains noeuds frères droit ou gauche d'un fils (d'où le nom de markovization horizontale), le but étant de généraliser les règles hors-contexte observées dans un corpus, et éviter d'avoir des comptes trop faibles pour des parties droites rares (ou non vues à l'estimation). Ce qui nous intéresse ici en particulier est la markovisation horizontale d'ordre 0, où un fils est généré indépendamment de ses noeuds frères. Par exemple figure 5.2, pour représenter le syntagme NP $\rightarrow$ DET ADJ N ADJ PP, on peut utiliser un symbole NP pour le syntagme complet, et un symbole :NP pour un NP incomplet, et générer en cascade les 5 noeuds fils (arbre (a) figure 5.2). La génération des fils peut privilégier le noeud tête, comme illustré avec l'arbre (b) : un symbole :NP(N) y est utilisé pour représenter un NP de tête N incomplet.

1. Un exemple pour le français pourrait être celui de la distribution différente d'un syntagme adjectival selon que l'adjectif est postmodifié ou pas : seul le type non postmodifié peut apparaître en position prénominale.

2. Il s'agit de comptes sur la version du FTB de 12531 phrases utilisée dans (Crabbé and Candito, 2008), dans le schéma FTBconst.

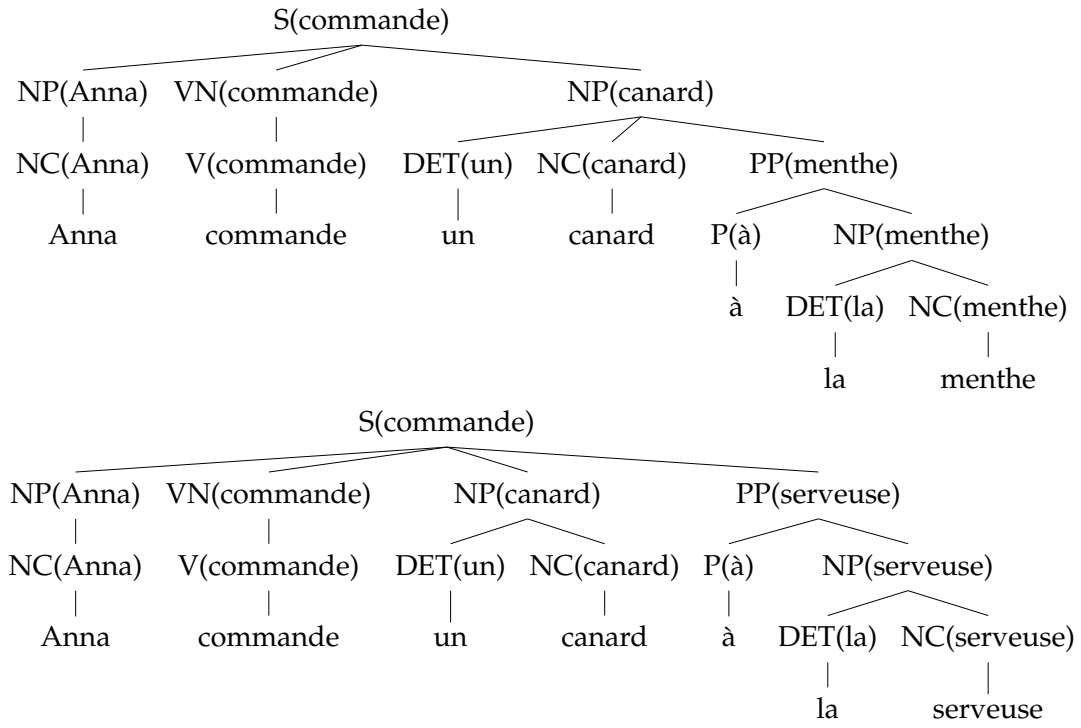

FIGURE 5.1 – Rattachement prépositionnel différent, pour une même séquence de catégories. L'information sémantique fournie par la lexicalisation des symboles syntagmatiques apporte une information déterminante pour le bon rattachement du PP.

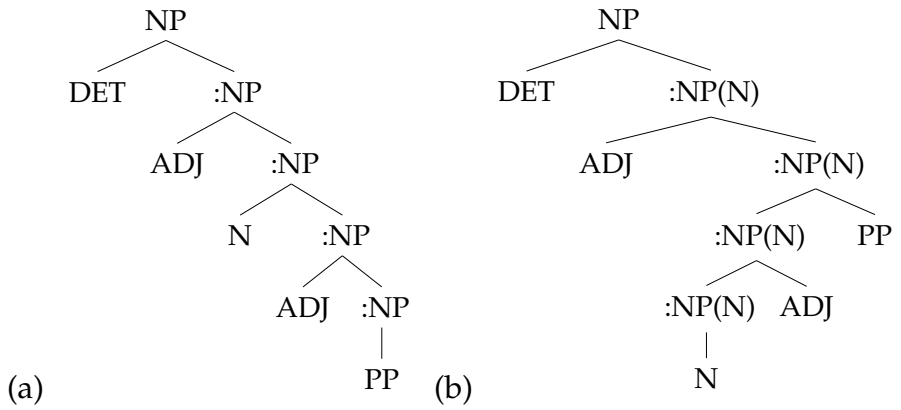

FIGURE 5.2 – Markovisation horizontale d'ordre 0 : (a) génération des fils de droite à gauche (le symbole artificiel :NP est utilisé pour un NP incomplet) et (b) génération des fils dans l'ordre tête, frères droits de la tête, frères gauches de la tête (Klein and Manning, 2003) (le symbole :NP(N) est utilisé pour un NP incomplet de tête N).

On peut voir que la technique de lexicalisation des PCFG, couplée à ce type de markovisation ramène la grammaire à des règles bilexicales, encodant une dépendance entre une tête et un de ses dépendants droits ou gauches, i.e. très proche des dépendances bilexicales manipulées dans les arbres de dépendances.

Automatisation de la spécialisation des symboles : l'algorithme PCFG-LA

Matsuzaki et al. (2005) ; Petrov et al. (2006) ont automatisé le processus de spécialisation des symboles syntagmatiques. À partir de la CFG induite par un corpus arboré, chaque symbole non terminal A observé dans la grammaire est démultiplié en k sous-symboles $A_1, ..., A_k$. Les indices $1, ..., k$ constituent des annotations dites latentes, car on ne dispose à l'apprentissage que d'arbres syntaxiques où les symboles ne sont pas indicés. Si la CFG de départ est mise en forme normale de Chomsky avec r règles binaires de la forme $A \rightarrow B\ C$, et u règles unaires introduisant un symbole terminal $A \rightarrow \alpha$, la grammaire avec annotations comportera rk^3 règles binaires et uk règles unaires.

Les probabilités des règles avec annotations latentes ne peuvent pas être estimées de manière supervisée, car le treebank observé n'a pas d'annotations latentes. Les auteurs utilisent l'algorithme inside-outside défini au départ pour l'apprentissage non supervisé des probabilités des règles d'une PCFG, lorsqu'on ne dispose que de la CFG non probabilisée, et d'un corpus de phrases sans les arbres syntaxiques [3]. L'algorithme garantit de trouver les probabilités des règles à annotations latentes fournissant un maximum local de la vraisemblance du treebank observé (i.e. des arbres sans annotations latentes). La fonction de vraisemblance est non-convexe, le maximum atteint est un maximum local, la solution trouvée dépend donc de l'initialisation aléatoire des probabilités cherchées (nous revenons sur ce problème infra section 5.1.4).

Petrov et al. (2006) introduisent une technique dite de division-fusion (split-merge), qui s'est avérée donner de très bons résultats. Cette technique permet de varier le degré de spécialisation des différents symboles syntagmatiques : au final, le nombre k de sous-symboles variera d'un symbole initial à un autre. L'apprentissage utilise des cycles division-fusion, où (i) d'abord chaque symbole A est divisé en deux symboles A_1, A_2 ; (ii) ensuite, chacune de ces paires est évaluée grâce au calcul efficace approché de la perte de vraisemblance du treebank observé qui serait induite si la paire de symboles était refusionnée ; (iii) enfin les paires occasionnant la moindre perte de vraisemblance (et donc les divisions les moins utiles) sont refusionnées.

L'analyseur de Petrov et al. (2006) a obtenu de très bons résultats à l'époque. J'ai collaboré avec Benoît Crabbé pour son utilisation pour le français (Crabbé and Candito, 2008). Je décris dans la section suivante comment nous l'avons couplé à des techniques de généralisations de mots.

5.1.2 Parser en utilisant des grappes de mots

Un problème majeur en TAL et en parsing supervisé sont les mots "inconnus", i.e. rencontrés en phase de prédiction mais absents du corpus d'apprentissage, qui sont le sujet d'une vaste littérature. Aujourd'hui le problème est résolu en partie par l'utilisation de vecteurs de mots

3. Inside-outside est une instantiation de l'algorithme Expectation-Maximization (EM). Ici, le problème est plus simple que le inside-outside habituel, cf. à l'apprentissage la structure des arbres est connue, seules les annotations latentes ne le sont pas.

calculés en combinant les vecteurs de sous-chaînes composant le mot (Bojanowski et al., 2017 ; Devlin et al., 2019), limitant ou supprimant totalement les sous-chaînes inconnues.

La variation lexicale est en particulier problématique dans le cas d'une langue à la morphologie riche, la flexion augmentant la dispersion des données lexicales. Pour le français, sans être à morphologie "riche", la flexion reste beaucoup plus importante qu'en anglais.

C'est ce qui nous a poussés à l'époque (Candito and Crabbé, 2009) à explorer l'utilisation de classes d'équivalence de mots ("clusters") en lieu et place des mots eux-mêmes, permettant d'une part de réduire la taille du vocabulaire et d'augmenter globalement les fréquences des formes utilisées, et d'autre part de couvrir un vocabulaire plus large que celui du treebank d'apprentissage de l'analyseur : une forme absente du corpus d'apprentissage, mais appartenant au cluster d'une forme connue bénéficiera des paramètres appris pour le cluster, exactement comme une forme effectivement présente dans le corpus d'apprentissage.

À noter que l'approche actuelle est totalement inverse, avec l'apprentissage de vecteurs de mots (non-contextuels ou contextuels), au lieu de grouper des mots en classes, on distribue la représentation des mots sur quelques centaines de dimensions, ce qui a bien un effet de généralisation sur les mots, cf. une entrée non vue à l'apprentissage, contenant entre autres un vecteur de mot v pourra bénéficier de tout exemple d'apprentissage contenant un vecteur de mot proche de v dans l'espace vectoriel de mots.

Clustering morphologique

Une première méthode de généralisation de mots consiste à neutraliser certaines variations flexionnelles, ce que nous avons désigné par "déflexion". On se place cependant dans un scenario où l'étiquetage morpho-syntaxique n'est pas fourni en amont, mais réalisé en même temps que le parsing. La neutralisation flexionnelle doit donc préserver parfaitement l'ambiguïté de catégories morphosyntaxiques de départ. Ainsi on ne peut pas par exemple simplement lemmatiser le texte, ce qui implique souvent de désambiguiser la catégorie morpho-syntaxique.

En outre, l'intuition est que certaines marques flexionnelles, comme le nombre, genre, temps grammatical et personne jouent un rôle assez redondant par rapport à l'information syntaxique codée par l'ordre des mots. En revanche le mode verbal est déterminant pour capturer certaines différences de structure syntaxique. D'où l'idée de neutraliser la flexion, en conservant la distinction de mode, et à ambiguïté de catégories constante.

L'algorithme, qui utilise un lexique morphologique (le Lefff (Sagot et al., 2006)), fonctionne comme suit : pour une forme graphique w, notons $lex(w)$ l'ensemble des entrées lexicales de forme w. Si toutes les entrées dans $lex(w)$ sont au pluriel, et ont une entrée équivalente au singulier de même forme w_{sing}, alors on remplace w par w_{sing}. Par exemple, si w est la forme *entrées*, ambigüe entre le nom *lem=entrée,cat=N,genre=fem,nb=pl* et le verbe *lem=entrer,cat=V,mode=part-passé,genre=fem,nb=pl*, les entrées lexicales au singulier ont toutes la même forme *entrée*. On répète le même processus pour passer éventuellement du féminin au masculin. Ceci permet de changer *écoutée* en *écouté*, mais pour la forme *entrée*, seul le participe a une entrée au masculin. De manière à conserver l'ambiguïté entre le nom et le participe, on conserve donc la forme *entrée*. Enfin, on remplace les verbes finis par leur forme au présent indicatif 2ème personne du pluriel, à condition de ne pas perdre ni rajouter d'ambiguïté. Par exemple *mangeaient, mangerions, mange, mangera* sont remplacés par *mangez*, *était, seras, serions, furent* sont remplacés par *êtes*, mais pas *suis*, qui est ambigu entre *être* et *suivre*. La forme 2ème personne du pluriel a été choisie car elle est rarement ambigue avec d'autres catégories, et permet donc un remplacement effectif. À l'inverse utiliser par exemple

la forme 3ème personne du singulier engendrerait des ambiguités fréquentes avec un nom, et le remplacement serait bloqué (par exemple remplacer *jouèrent* par *joue* serait bloqué).

Clustering non supervisé

Pour aller plus loin dans les regroupements de mots, nous complétons le clustering morphologique avec le clustering de Brown et al. (1992), entièrement non supervisé. Nous nous sommes ici inspirés de Koo et al. (2008), qui montrent qu'utiliser de tels clusters de mots sous la forme de traits dans un analyseur en dépendances discriminant est bénéfique. Nous transposons l'utilisation au parsing en constituants, et avec un remplacement complet d'une forme par son cluster.

Le clustering de Brown est un partitionnement ("hard clustering" en anglais) de type hiérarchique ascendant agglomératif, où la paire de clusters à fusionner à chaque itération est la paire qui lorsque fusionnée occasionne le moins de perte de vraisemblance du corpus, d'après un modèle de langue de type ngrammes intégrant le partitionnement en classes (en l'occurrence avec n=2). Plus précisément, en notant c_k le cluster associé au kième mot w_k, la probabilité de ce mot sachant ses n-1 mots précédents est définie comme $P(w_k|w_{k-(n-1)}^{k-1}) = P(w_k|c_k)P(c_k|c_{k-(n-1)}^{k-1})$. L'algorithme repose sur le calcul efficace de la perte de vraisemblance occasionnée par la fusion de deux clusters, que nous ne détaillons pas ici.

L'algorithme approché agglomératif ascendant est alors le suivant : pour obtenir C clusters sur un vocabulaire de V mots, chacune des C formes de V les plus fréquentes dans le corpus d'estimation forme un cluster singleton. Pour le (C+1)ième mot le plus fréquent, on crée un (C+1)ième cluster. On identifie la paire dont la fusion occasionne la plus petite perte de vraisemblance du corpus total, et on réalise cette fusion. On répète l'opération jusqu'à épuiser le vocabulaire (i.e. V - C fois), ce qui fournit un clustering en C clusters.

Afin d'obtenir différents niveaux de granularité des clusters, le processus agglomératif peut être poursuivi en fusionnant des paires au sein de C clusters, jusqu'à arriver à un seul cluster au bout de C-1 fusions.

Application au parsing intra-domaine

Dans (Candito and Crabbé, 2009), nous avons testé les différentes formes de clustering pour le parsing en constituants via l'implémentation de l'algo PCFG-LA de Petrov et al. (2006), adaptée pour le français pour la gestion des mots inconnus (Crabbé and Candito, 2008). Le scenario est simplement de remplacer à l'apprentissage chaque occurrence de mot par son cluster avant l'apprentissage du parser. Lors de la prédiction, les mots sont d'abord ainsi remplacés, la séquence de clusters est analysée, et les mots d'origine sont réintroduits à la place des clusters. Les clusters de Brown sont appris sur le corpus *L'Est Républicain*, préalablement défléchi comme décrit en 5.1.2. [4].

Nous comparons plusieurs types de clustering combinant le clustering morphologique et le clustering de Brown. En effet l'étude qualitative des clusters Brown obtenus indique que certaines distinctions de catégories sont parfois perdues, ce qui nous a amenés a re-séparer les clusters de Brown d'après la casse et certains suffixes :

— defl : le clustering morphologique défini supra, fournissant des formes "défléchies"

4. Il s'agit d'articles du journal régional du même nom, pour un total d'environ 125 millions de mots, disponibles sur le site du CNRTL (http://www.cnrtl.fr/corpus/estrepublicain). Les détails de l'entraînement du clustering de Brown et du parser sont fournis dans (Candito and Crabbé, 2009).

— defl+brown+cap : le clustering de Brown appliqué aux formes défléchies, où en outre on sépare chaque cluster en deux, selon que la forme commence par une majuscule ou pas
— defl+brown+cap+suff : idem que defl+brown+cap, mais on partitionne en outre les formes dans chaque cluster d'après 9 suffixes (dont un défaut), destinés à capturer grossièrement le mode du verbe au sein des formes défléchies [5]

La table 5.1 fournit les performances du parser PCFG-LA sur le corpus de test du French Treebank [6], sur les phrases de moins de 40 mots [7]. Si l'on étudie la taille du vocabulaire du corpus d'apprentissage, on constate que la déflexion réduit le vocabulaire d'environ un quart (de 27143 à 20268). Le clustering subséquent de Brown, avec marques de capitalisation donne un vocabulaire d'environ 1200 clusters, alors que rajouter quelques suffixes fait remonter à environ 2000 clusters. En termes de performance de parsing, les formes défléchies donnent une amélioration, mais leur appliquer le clustering de Brown (defl+brown+cap) ne donne pas d'amélioration supplémentaire. L'explication tient dans les performances d'étiquetage morpho-syntaxique : les clusters de Brown visiblement groupent des mots de catégories différentes, et le tagging baisse légèrement. C'est finalement la configuration avec ré-introduction de suffixes (defl+brown+cap+suffixes) qui fournit les meilleurs résultats, à la fois en tagging et en parsing.

Type des formes en entrée	Parsing (F-mesure)	Taille du vocabulaire	Tagging (précision)
fléchies (baseline)	86.8	27143	96.9
défléchies	87.4	20268	96.8
défl. + brown + cap	87.8	1201	96.4
défl. + brown + cap + suffixes	**88.3**	1987	**97.0**

TABLE 5.1 – Performance de parsing sur le FTB (corpus de test, phrases de moins de 40 mots), en utilisant différents types de symboles en entrée. 1ère ligne : symboles initiaux (formes fléchies). Lignes suivantes : les formes fléchies sont remplacées à l'entraînement et au test par les formes défléchies, les clusters brown avec marque de capitalisation, les clusters brown avec marques de capitalisation et suffixes. Colonne 2 : F-mesure sur les constituants étiquetés (hors tagging). Colonne 3 : taille du vocabulaire dans le corpus d'entraînement. Colonne 4 : performance de tagging (précision).

5.1.3 Application au parsing hors-domaine : pont lexical

Ce travail a été réalisé avec Djamé Seddah et Enrique Henestroza (Candito et al., 2011 ; Candito and Seddah, 2012b). Un problème majeur du TAL par apprentissage supervisé est celui de la dépendance au corpus d'apprentissage. L'hypothèse de l'apprentissage supervisé est que les données sur lesquelles les paramètres d'un algorithme sont appris et celles sur lesquelles il sera utilisé suivent la même distribution de probabilité, ce qui n'est bien sûr en pratique souvent pas le cas. En parsing, un analyseur appris sur du texte journalistique ne sera

5. Nous avons utilisé en particulier *-ant*, *-r*, *-ez*. Avec le recul, une sélection automatique des suffixes à utiliser aurait été préférable.
6. Dans sa version de 12531 phrases créée par Crabbé and Candito (2008), avec en outre ici les composés syntaxiquement réguliers "défaits", c'est-à-dire qu'ils apparaissent dans les arbres syntagmatiques avec une structure syntaxique régulière.
7. Cette limitation était encore courante à l'époque !

en pratique pas optimal si utilisé sur des textes scientifiques, ou sur des forums de discussion en ligne.

Cette problématique a été abondamment étudiée dans la littérature, sous le terme d'"adaptation au domaine". La configuration expérimentale de cette tâche est que l'on dispose d'un corpus arboré -en général assez petit car les annotations sont coûteuses-, d'un domaine dit source, et on cherche à optimiser les performances d'un parser pour un domaine différent, dit domaine cible. Une tâche légèrement différente est celle de l'augmentation de la robustesse au domaine : on vise alors d'obtenir un analyseur performant sur un ou plusieurs domaines cibles, mais également restant performant sur le domaine source.

Différentes techniques ont été proposées pour adapter des modèles d'analyse existants à de nouveaux genres, toutes conçues pour combler la variation syntaxique et lexicale entre les domaines source et cible, avec en particulier :

— l'adaptation au domaine via de l'auto-entraînement (*self-training*) (McClosky et al., 2006 ; Sagae, 2010) : un analyseur entraîné sur le domaine source est utilisé pour analyser du domaine cible, et on réentraîne un analyseur sur les données validées du domaine source et les données prédites du domaine cible. Le corpus d'entraînement ainsi obtenu, bien que bruité, capture suffisamment de régularités du domaine cible pour améliorer les performances d'analyse sur ce domaine (tout en dégradant les performances sur le domaine source) ;
— co-entraînement avec sélection d'exemples (Steedman et al., 2003) : deux analyseurs sont itérativement re-entraînés sur leurs sorties respectives, les phrases du domaine cible à utiliser étant choisies de manière à minimiser les erreurs d'analyse tout en maximisant l'utilité à l'entraînement ;
— transformation de treebank et adaptation du domaine cible (dans le cas de textes non édités, de type forums de discussion) (Foster, 2010) ;

C'est dans ce contexte que nous avons investigué l'utilisation du clustering de mots pour l'adaptation au domaine. La constatation de départ est que la baisse de performance d'un analyseur sur le domaine cible est en grande partie liée au vocabulaire spécifique du corpus cible. Pour l'illustrer, détaillons les données et domaines utilisés dans (Candito et al., 2011) : le domaine source est le domaine journalistique (le FTB). Le domaine cible est le domaine médical, plus précisément le corpus de l'Agence Européenne du Médicament, EMEA, qui est une des sources pour la création du corpus Sequoia (décrit supra section 3.3). La partie EMEA du SEQUOIA correspond à deux rapports publics d'évaluation de médicament, donnant les corpus EMEA-dev et EMEA-test, contenant environ 9300 tokens et 11600 tokens respectivement. L'écart lexical entre les domaines source et cible est très important, comme déjà évoqué lors de la description du SEQUOIA (section 3.3.1) : environ un token sur 5 des corpus EMEA-dev et EMEA-test correspond à une forme absente du corpus d'entraînement du FTB.

Pour utiliser les clusters de mots pour l'adaptation au domaine, l'idée de base est d'apprendre des clusters sur un corpus brut mixant domaine source et domaine cible. On espère ainsi que certains clusters regrouperont des mots issus des deux différents domaines. L'intuition est qu'un mot w_c présent uniquement dans le domaine cible puisse avoir des contextes proches de mots du domaine source, directement ou indirectement : w_c peut partager certains de ses contextes avec un autre mot du domaine cible, qui lui partage d'autres contextes avec des mots source.

En pratique, nous avons utilisé comme corpus brut du domaine source, le corpus *L'Est Républicain* cité supra (125 millions de tokens environ), ce qui est imparfait en cela que ce

journal régional aborde des thèmes différents du corpus *Le Monde*[8]. Et comme corpus brut du domaine cible, le corpus EMEA (5 millions de tokens environ).

On peut ensuite utiliser le même protocole que décrit supra section 5.1.2, où on apprend un parser en remplaçant directement les mots par leur cluster. Les résultats sont fournis Table 5.2, où l'on fournit également les résultats avec de l'auto-apprentissage (en utilisant 200 000 phrases d'EMEA). On constate sur le FTB-test une augmentation modeste en passant aux formes défléchies (defl), puis aux clusters brown appris sur corpus du domaine source (dfl+brown-ER) puis finalement les clusters réalisant le "pont lexical" entre les deux domaines (dfl+brown-ER+EMEA).

Cependant, en comparant ces résultats à ceux obtenus avec un simple auto-apprentissage, on constate que ce dernier donne un incrément plus net, et le couplage du clustering de mots plus auto-apprentissage ne fournit qu'un incrément très modeste (F-score=84.7 versus 85.2). En outre, avec l'auto-apprentissage, les clusters dfl+brown-ER+EMEA n'apportent pas d'avantage par rapport aux clusters dfl+brown-ER.

Ces quelques expériences donnent donc des résultats mitigés pour le clustering de mots avec "pont lexical". Mais il est difficile de donner des conclusions définitives. En particulier la taille respective du corpus brut source et cible utilisé pour le calcul des clusters aurait dû être un hyper-paramètre à régler.

Type des formes en entrée	Sans self-training	Avec self-training (200k)
fléchies	81.2	84.7
défléchies	81.8	84.7
dfl+brown-ER	82.6	85.1
dfl+brow-ER+EMEA	83.5	85.2

TABLE 5.2 – F-mesure des constituants étiquetés pour les phrases de moins de 40 tokens de EMEA-test, en utilisant différents types de clustering pour remplacer les mots : formes fléchies, défléchies, clusters brown appris sur l'Est Républicain défléchi (dfl+brown-ER), clusters brown appris sur l'Est Républicain défléchi plus EMEA (dfl+brown-ER+EMEA). Colonne 2 : sans auto-apprentissage, colonne 3 : avec auto-apprentissage de 200 000 phrases.

5.1.4 Sensibilité de l'algorithme PCFG-LA à l'initialisation aléatoire

Dans notre enthousiasme devant l'élégance et les performances de l'analyseur PCFG-LA de Petrov et al. (2006), nous avions, comme la communauté scientifique à l'époque, ignoré le problème de la sensibilité de l'algorithme inside-outside aux initialisations aléatoires des probabilités des règles hors-contexte avec symboles latents. En 2010, Petrov (2010) a étudié empiriquement cette sensibilité et montré que faire varier les graines aléatoires à l'initialisation d'EM engendrait des écarts de performance allant jusqu'à 0.6 point de F-mesure sur la section 22 du Penn Treebank (traditionnellement utilisée comme corpus de développement). Cette section a une taille comparable au corpus de développement du FTB (environ 40000 tokens versus 36500 respectivement), on peut donc s'attendre à ce que la variation de performance sur le FTB-dev soit au moins du même ordre de grandeur, et plutôt plus, étant donné que le

8. Comme nous l'avons calculé a posteriori, en constituant le corpus Sequoia complet, finalement, le corpus Europarl aurait été plus proche du FTB, avec le plus faible taux d'inconnus, cf. supra table 3.1.

corpus d'apprentissage du PennTreebank est plus de trois fois plus grand que celui utilisé dans nos expériences avec le FTB (environ 1 million de tokens, versus 280000 tokens).

Ceci vient donc tempérer un peu nos conclusions précédentes sur l'impact positif d'utiliser un parsing de clusters de mots (morphologiques et non supervisés) à la place d'un parsing de mots. On peut tout de même mentionner que nos diverses expériences de l'époque n'ont pas donné de tendances contradictoires d'un run à un autre.

5.2 Un classifieur pour corriger les arbres de dépendances

Je décris ici brièvement une partie du travail de thèse d'Enrique Henestroza Anguiano, dont j'ai été l'encadrante principale. Il s'agit de travaux toujours en analyse syntaxique statistique, mais cette fois en analyse syntaxique en dépendances. Je commence par décrire brièvement le contexte de l'époque concernant le parsing en dépendances, avec l'arrivée de données pour de nombreuses langues, et d'algorithmes performants.

5.2.1 Contexte : essor du parsing en dépendances

Alors que les arbres syntaxiques de type arbres de dépendances ont longtemps joué un rôle relativement marginal en TAL, la tendance a commencé à s'inverser à la fin des années 90 et début des années 2000. Nous décrivons ci-dessous deux facteurs importants permettant d'expliquer cette évolution.

Des treebanks en dépendances, natifs ou obtenus par conversion deviennent disponibles pour de nombreuses langues

Premièrement, la technique de lexicalisation des PCFG, où les mots-têtes sont remontés au niveau des symboles syntagmatiques, est la première étape d'une conversion automatique d'un arbre de constituants en arbres de dépendances (voir infra section 3.4.1). Aller jusqu'à une conversion en arbres de dépendances était donc ensuite un pas naturel, qui a mis à disposition de la communauté des treebanks en dépendances pour des langues dont le ou les corpus arborés de référence étaient à l'origine annotés en constituants, sans avoir à refournir un nouvel effort d'annotation manuelle. Ces corpus obtenus par conversion se sont ajoutés à ceux nativement annotés en dépendances déjà cités (supra section 3.1).

Le parsing en dépendances (projectives) devient particulièrement visible dans la communauté TAL en 2006, alors qu'il constitue la tâche proposée pour la compétition internationale CoNLL-X (Buchholz and Marsi, 2006), proposant aux participants de travailler sur des treebanks natifs en dépendances ou convertis vers des dépendances pour en tout 13 langues, de familles linguistiques variées. Cette compétition a eu un écho très important, d'une part pour le parsing en dépendances, et d'autre part pour le parsing multilingue, alors qu'à l'époque de nombreux travaux se concentraient sur une seule langue (majoritairement l'anglais).

Un algorithme efficace : le parsing par transitions

Un autre facteur important dans l'essor du parsing en dépendances est l'arrivée du parsing dit "par transitions" (Yamada and Matsumoto, 2003 ; Nivre and Scholz, 2004). S'inspirant de l'algorithme shift-reduce pour parsing en constituants (Aho and Ullman, 1972), le parsing par

transitions manipule une pile d'éléments non encore complets et un tampon de tokens non encore traités. L'algorithme parcourt la phrase une seule fois (en général de gauche à droite), et construit incrémentalement un ensemble de dépendances bilexicales au moyen d'un nombre réduit d'actions autorisées (des "transitions").

Différents jeux de transitions ont été proposés, avec au départ en particulier la contrainte que ces jeux garantissent la complétude et la cohérence sur la classe des arbres de dépendances projectives (c'est-à-dire que tout arbre projectif doit pouvoir être produit, et inversement la structure résultant d'une analyse est forcément un arbre projectif) (Nivre, 2008). Pour les jeux de transitions projectifs, le nombre de noeuds de l'arbre à construire étant la longueur de la phrase, l'analyse par transitions se fait *en temps linéaire* par rapport à sa longueur.

Le choix de la transition à appliquer à chaque étape peut être délégué à un classifieur appris de manière supervisée. Cela permet d'intégrer les dernières avancées en classification automatique supervisée, comme le font par exemple à l'époque Yamada and Matsumoto (2003), qui utilisent un classifieur de type machine à vecteur support, ou plus tard des classifieurs neuronaux (Chen and Manning, 2014).

Parsing basé sur les graphes

À peu près à la même période, un autre type d'algorithme a eu un impact très important : l'analyseur dit "MST" pour "maximum spanning tree" de McDonald et al. (2005b). On parle d'algorithme "basé sur les graphes", car on considère pour une phrase de n tokens le graphe orienté contenant un noeud par token et toutes les dépendances bilexicales possibles (tous les arcs orientés non réflexifs). Le problème du parsing en dépendances est alors ramené à trouver, au sein de ce graphe, l'arbre de score maximal, l'arbre couvrant maximal (en anglais "maximum spanning tree", d'où l'acronyme MST pour ce type d'analyseur). Le score d'un arbre est la somme des scores de portions de l'arbre. On parle d'"ordre" pour la taille de ces portions. Dans le modèle initial dit d'ordre 1, ces portions sont simplement chacun des arcs.

Une fois la fonction de score d'arcs apprise, McDonald et al. (2005b) utilisent l'algorithme de Chu-Liu-Edmonds de recherche de l'arbre couvrant maximal, dont il existe une implémentation efficace en $O(n^2)$. À noter que cet algorithme ne contraint pas l'arbre trouvé à être projectif. De manière inhabituelle, ajouter la contrainte de projectivité augmente la complexité : McDonald et al. (2005a) utilisent l'algorithme de programmation dynamique d'Eisner (1996), en $O(n^3)$, pour prédire l'arbre couvrant maximal projectif.

McDonald et al. (2005b) utilisent une fonction de score des arcs linéaire, et en apprennent les paramètres via une version adaptée aux sorties structurées (ici des arbres de dépendance) de l'algorithme MIRA (Margin Infused Relaxed Algorithm) (Crammer and Singer, 2003).

À noter cependant que scorer un arbre avec la simple somme des scores de ses arcs donne un contexte de décision très limité (même si contrebalancé par l'optimisation globale de trouver le meilleur arbre). Les modèles successifs utilisent des portions d'arbres plus grandes. Par exemple McDonald and Pereira (2006) somment les scores de paires d'arcs adjacents (i.e. de même noeud gouverneur). Cela nécessite cependant de complexifier l'algorithme de recherche de l'arbre de plus haut score : pour le cas projectif, l'algorithme d'Eisner peut être adapté en restant en $O(n^3)$. Pour le cas non projectif, il n'y a pas d'algorithme de recherche exacte, une recherche approchée à partir du meilleur arbre projectif est proposée.

L'analyseur de McDonald, en particulier dans sa version d'ordre 2 a fait date en établissant

le nouvel état de l'art pour l'anglais (évalué sur le Penn Treebank converti en dépendances) et le tchèque (évalué sur le Prague Dependency Bank). Aujourd'hui le parsing de type MST est toujours très utilisé, en utilisant notamment une transformation biaffine pour scorer les arcs (Dozat and Manning, 2017) (présenté au chapitre 7).

5.2.2 Corriger des arbres de dépendances en disposant d'un plus grand contexte

La problématique abordée est la difficulté de donner à un analyseur en dépendances un contexte suffisant lui permettant de prendre les bonnes décisions d'attachement. Par exemple, un parser par transitions (cf. section 5.2.1) a accès à quelques éléments seulement de la pile et du tampon. En théorie, complexifier la représentation du contexte doit permettre des décisions mieux informées et donc meilleures. En pratique, cela disperse les données d'apprentissage et le rend moins opérant [9].

Inspirés de (Hall and Novák, 2005), Henestroza Anguiano and Candito (2011) proposent un algorithme de correction d'analyses syntaxiques : à partir de la sortie d'un analyseur en dépendances quelconque, on reconsidère un par un les attachements de chaque mot de la phrase, en bénéficiant du contexte structuré fourni par l'arbre syntaxique de départ pour définir quels sont les gouverneurs potentiels de chaque mot. L'intuition de départ est qu'un arbre syntaxique en dépendances est globalement de bonne qualité (les résultats par exemple pour le français, pour un parser appris et testé sur le FTB sont autour de 90% de score UAS, i.e. 9 tokens sur 10 reçoivent leur bon gouverneur), et aide à cibler le contexte à considérer pour une décision de ré-attachement [10]. C'est a priori particulièrement adapté pour un parser par transition avec analyse gloutonne (pour chaque configuration, on choisit la meilleure transition locale, sans objectif d'optimiser la séquence complète de transitions), qui est sujette à la propagation d'erreurs. C'est également adapté pour un parser de type MST (cf. section 5.2.1), où le score d'un arbre prédit est une somme de scores de sous-arbres très réduits (un à deux arcs), donnant ainsi peu de contexte.

L'autre intuition, contribution originale de ce travail, est que la procédure de ré-attachement de mots permet de spécialiser les modèles en fonction du type de mot à ré-attacher. Nous nous sommes en particulier concentrés sur l'attachement prépositionnel et la coordination, deux difficultés notoires en analyse syntaxique. En effet, pour le français, nous avons constaté par exemple dans le FTB-dev analysé avec l'analyseur basé sur les graphes de McDonald and Pereira (2006) (cf. section 5.2.1), 30% des erreurs d'attachement concernent des prépositions, et environ 15% des prépositions sont mal rattachées. La coordination est plus rare, mais globalement moins bien traitée : les conjonctions de coordination représentent environ 2% des tokens, mais 10% des tokens mal rattachés, et environ 35% des conjonctions de coordination sont mal rattachées.

L'approche par correction d'analyses peut être comparée à l'approche de ré-ordonnancement des n-meilleures analyses (technique dite de "reranking"). La première a l'avantage de ne pas nécessiter les n meilleurs arbres. Elle peut s'appliquer simplement à un seul arbre, peu importe

9. Actuellement, les modèles neuronaux récurrents ou basés sur l'attention permettent de prendre en compte toute la phrase pour une décision d'attachement.

10. Par exemple, Hall and Novák (2005) rapportent qu'environ 2/3 des mots mal attachés dans des arbres syntaxiques prédits (pour le tchèque) ont leur gouverneur correct atteignable par un chemin dans l'arbre de longueur 1, 2 ou 3.

INPUT : Arbre de dépendances prédit T

LOOP : Pour chaque dépendant d dans T

- Identifier les candidats gouverneurs C_d dans T
- Identifier le modèle de score $S^{m(d)}$ à utiliser étant donné la catégorie de d
- Prédire $\hat{c} = \underset{c \in C_d}{\text{argmax}}\ S^{m(d)}(c, d, T)$
- Mettre à jour T : $T\{gov(d) \leftarrow \hat{c}\}$

OUTPUT : T

FIGURE 5.3 – L'algorithme de correction d'arbre

son origine et son espace de recherche n'est pas restreint aux n-meilleures analyses. Elle permet en outre de spécialiser les modèles de ré-attachement en fonction du type de dépendant. La seconde approche permet cependant d'opérer une optimisation globale à la phrase, au contraire des décisions individuelles de ré-attachement.

Algorithme de ré-attachement

L'algorithme de réattachement prend en entrée un arbre de dépendances T. Pour chaque dépendant d dans T (i.e. pour chaque token sauf la racine), on identifie l'ensemble C_d des candidats gouverneurs de d. On récupère le modèle $m(d)$ (en fonction de la catégorie de d), et la fonction de score correspondante $S^{m(d)}$.

Pour définir l'ensemble C_d des candidats gouverneurs pour le dépendant d, notons h_d le gouverneur de d dans T. Les candidats gouverneurs sont les noeuds atteignables à partir de h_d par un chemin de longueur $\leq l$, avec deux contraintes. D'une part, le chemin ne doit pas passer par d, ce qui est un moyen simple de garantir que le graphe modifié par les ré-attachements reste un arbre. D'autre part, on exclut de l'ensemble des candidats gouverneurs les noeuds qui introduirait de la non-projectivité s'ils étaient utilisés comme gouverneur de d.

Par exemple dans l'arbre de la figure 5.4, pour $d =$ *'et'*, les candidats avec un chemin de longueur maximale $l = 2$ à partir de $h_d =$ *'blocage'*, sans passer par d sont *blocage, contre, protestent, le, des, revendications*.

Avec une longueur maximale des chemins $l = 2$, le score maximal atteignable (si tous les réattachements corrects possibles sont faits) permet d'obtenir respectivement un score d'attachement non étiqueté UAS entre 95% et 96.1% sur le FTB-dev, avec différents parsers état de l'art à l'époque.

Scorage des candidats gouverneurs via traits linguistiquement riches

Les fonctions de score $S^{type(d)}$ utilisées à l'époque suivent un modèle linéaire. Pour un dépendant d de type t et un candidat gouverneur c, étant donnés une fonction Φ^t fournissant la représentation vectorielle du triplet (c, d, T) et un vecteur de poids appris w^t, on définit :

$$S^t(c, d, T) = w^t \cdot \Phi^t(c, d, T)$$

Les dépendants sont classés en trois types : préposition, coordination et autre et on utilise trois modèles de ré-attachement : deux modèles spécialisés (pour le rattachement prépositionnel et pour la coordination) et un modèle "générique" pour tous les autres types de dépendants.

L'intérêt est de pouvoir utiliser des traits extraits de l'arbre de dépendance prédit T, comme la longueur du chemin entre c et d, ainsi que des traits sur les dépendants de c. L'arbre T étant a priori correct à 90%, cela fournit des informations contextuelles riches (car syntaxiques) pour une éventuelle décision de ré-attachement.

En outre, des traits spécifiques sont ajoutés pour l'attachement des coordinations, et des prépositions. Pour la coordination il s'agit de capturer le degré de "parallélisme" entre deux éléments coordonnés, étant avéré que des conjoints ont une tendance à avoir la même catégorie, mais également le même nombre, voire le même lemme. Pour capturer cette information, on repère les conjoints coordonnés dans l'hypothèse où on rattacherait le dépendant d au candidat c, et on utilise par exemple des traits booléens encodant si les deux conjoints sont de même catégorie, de même nombre, de même lemme.

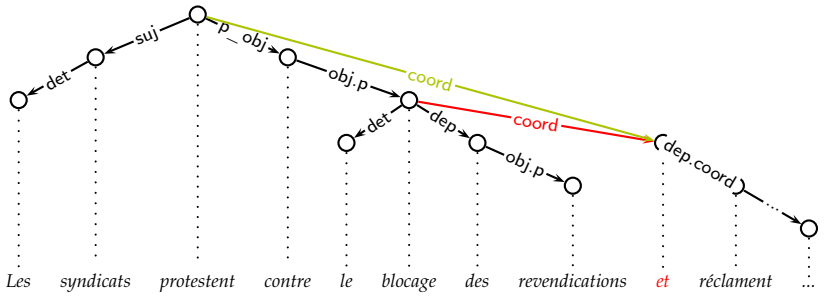

FIGURE 5.4 – Attachement erroné d'une conjonction de coordination (en rouge) et ré-attachement correct (en vert) [simplification de FTB flmf7ab2ep-766].

Par exemple, figure 5.4, le candidat gouverneur *protestent* pour ré-attacher la conjonction *et* donne lieu à des conjoints de même catégorie et nombre, contrairement au gouverneur initialement prédit *blocage*. À noter cependant que la technique n'est pas adaptée dans certains cas de mauvais attachements de coordination, si les deux conjoints sont erronés au départ, ce qui est le cas avec la figure 5.5, qui nécessite de faire trois ré-attachements. Si on commence par tenter de rattacher d'abord *et* à *proteste*, on n'a pas encore le 2ème conjoint correct (*représentent* et pas *%*), et donc les traits de parallélisme ne sont pas actifs.

Apprentissage par ordonnancement

Le modèle appris est un modèle d'ordonnancement ("ranking"), qui est particulièrement adapté dans le cas où on a des candidats mutuellement exclusifs, variant d'une instance à l'autre.

L'apprentissage utilisé est l'algorithme en ligne passif-agressif (Crammer et al., 2006), adapté au cas d'ordonnancement, fourni figure 5.6. On suppose ici avoir un ensemble d'apprentissage X, où les dépendants sont tous du même type. Un exemple d'apprentissage correspond à un dépendant d_i d'un arbre T_i, et un ensemble de candidats C_{d_i}, associé à la réponse attendue g_i, i.e. le vrai gouverneur de d_i, à identifier parmi les candidats. Pour chaque candidat $c \in C_{d_i}$, on a une représentation vectorielle $x_{i,c}$ du triplet (d_i, c, T_i). Comme pour MIRA cité section 5.2.1, l'algorithme opère la mise à jour la plus petite possible, sous la contrainte que le nouveau vecteur de poids classe l'exemple courant avec une marge de 1, la marge étant la différence de score entre le gouverneur gold, et le candidat incorrect de plus haut score. L'algorithme est passif, car la mise à jour n'a lieu que si la marge est inférieure à 1, mais

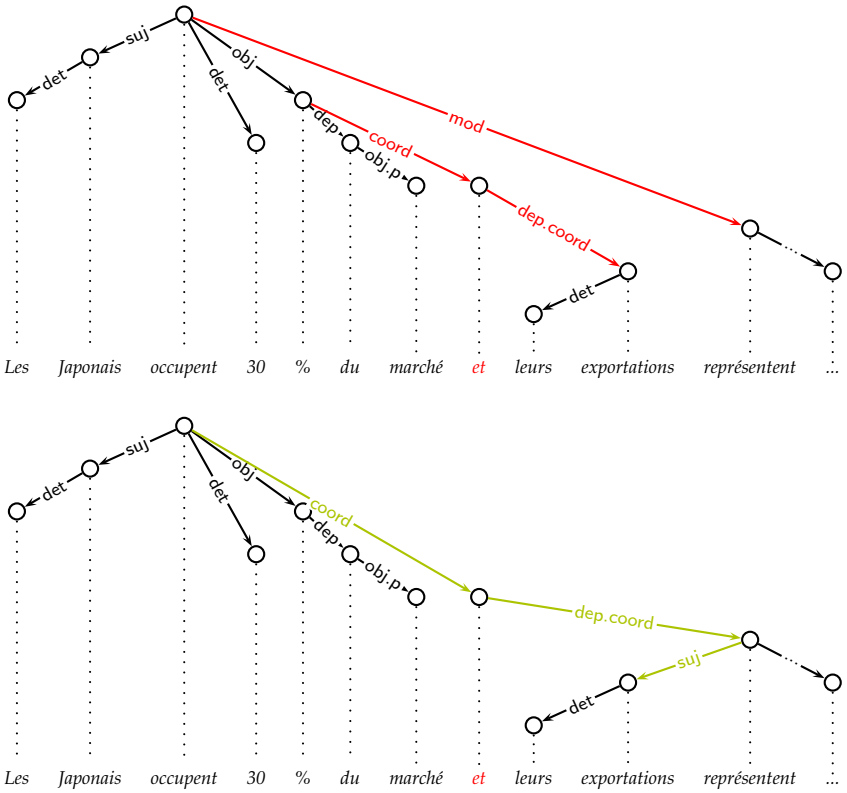

FIGURE 5.5 – Attachement erroné d'une coordination (en rouge) et ré-attachement correct (en vert) [simplification de FTB flmf7ab2ep-866] : cas où la conjonction est mal attachée (à % au lieu de *occupent*, ainsi que le conjoint qu'elle introduit (*exportations* au lieu de *représentent*).

INPUT : Aggressivité C, époques E.
INITIALISATION : $\vec{w}_0 \leftarrow (0, ..., 0), \vec{w}_{avg} \leftarrow (0, ..., 0)$
RÉPÉTER : E fois
  LOOP : For $i = 1, 2, \ldots, |X|$
    · Récupérer les vecteurs $\{\vec{x}_{i,c} : c \in C_{d_i}\}$
    · Récupérer le gouverneur gold $g_i \in C_{d_i}$
    · $h_i = \underset{c \in C_{d_i} - \{g_i\}}{\text{argmax}} \ (\vec{w}_{i-1} \cdot \vec{x}_{i,c})$
    · $m_i = (\vec{w}_{i-1} \cdot \vec{x}_{i,g_i}) - (\vec{w}_{i-1} \cdot \vec{x}_{i,h_i})$
    SI : $m_i < 1$
      · Let $\tau_i = \min\left\{C\,,\ \frac{1-m_i}{\|\vec{x}_{i,g_i} - \vec{x}_{i,h_i}\|^2}\right\}$
      · Set $\vec{w}_i \leftarrow \vec{w}_{i-1} + \tau_i(\vec{x}_{i,g_i} - \vec{x}_{i,h_i})$
    SINON :
      · Set $\vec{w}_i \leftarrow \vec{w}_{i-1}$
    · Set $\vec{w}_{avg} \leftarrow \vec{w}_{avg} + \vec{w}_i$
  · Set $\vec{w}_0 \leftarrow \vec{w}_{|X|}$
OUTPUT : $\vec{w}_{avg} / (E \cdot |X|)$

FIGURE 5.6 – Algorithme d'apprentissage par ordonnancement, de type Passif-Agressif

agressif car la mise à jour garantit une marge de 1, modulo l'hyperparamètre C, qui fixe la borne supérieure de mise à jour.

Résultats et discussion

Les expériences sont réalisés sur le FTB converti en dépendances, avec quatre analyseurs performants à l'époque. Nous nous concentrons sur les deux extrêmes : l'analyseur par transitions MaltParser (Nivre et al., 2007), en temps linéaire $O(n)$, obtenant une précision d'attachement non étiqueté UAS=89.3% sur le FTB-dev et le BohnetParser (Bohnet, 2010), de type MST avec facteurs d'ordre 2, obtenant un meilleur score UAS=91.3%, mais beaucoup plus lent car en $O(n^3)$.

Les résultats montrent qu'appliquer les modèles spécialisés (coordination et prépositions) et le modèle générique pour les autres types de dépendants permet une amélioration modeste mais statistiquement significative pour le MaltParser (de 89.8 à 90.5 sur le FTB-test), et une amélioration non significative pour le BohnetParser (de 91.8 à 91.9). L'amélioration sur l'attachement des conjonctions de coordination est effective (par exemple de 70.5 à 72.7 pour le BohnetParser), mais plus mitigée pour le rattachement prépositionnel.

Pour conclure, à l'époque on pouvait remarquer que notre solution de correction d'analyses appliquée aux prédictions du MaltParser était un bon compromis entre qualité et temps d'analyse, avec un temps global environ 4 fois moindre que le BonhetParser, sur le FTB-dev.

Chapitre 6

Syntaxe "profonde" : expliciter plus d'informations syntaxiques

Je décris dans ce chapitre le fruit d'une collaboration des équipes INRIA Alpage et Sémagramme, avec Bruno Guillaume, Guy Perrier, Corentin Ribeyre et Djamé Seddah. Éric de la Clergerie et Karen Fört y ont également participé [1].

Le projet "deep Sequoia"[2] a porté sur la définition de "graphes syntaxiques profonds", conçus comme niveau de représentation intermédiaire entre syntaxe et sémantique.

Le terme syntaxe profonde est très "chargé" d'un point de vue théorique, puisqu'il rappelle les temps de la grammaire générative transformationnelle, où une représentation profonde était centrale, et la représentation de surface des phrases obtenue par application de transformations. Nous l'utilisons avec un objectif pratique en premier lieu : rendre plus directement utiles les représentations syntaxiques, sans recourir à un lexique sémantique ni à une désambiguïsation lexicale (au-delà de la désambiguïsation obtenue avec les catégories morpho-syntaxiques). En effet, certes, les arbres de dépendances et les analyseurs en dépendances se sont généralisés, et les arbres de dépendances sont présentés comme permettant une lecture plus immédiate de la structure argumentale des prédicats. Mais nous sommes partis du constat qu'en pratique, à partir d'un arbre de dépendances, il reste encore beaucoup à expliciter pour obtenir ces structures argumentales.

— Certains phénomènes entièrement déterminés par la syntaxe ne sont pas directement encodés dans les représentations "surfaciques" couramment utilisées en TAL (arbres de constituants ou arbres de dépendances). C'est le cas par exemple du phénomène basique de contrôle : dans *Anna veut embaucher Laure*, les propriétés lexicales du verbe *vouloir* permettent de manière déterministe d'interpréter que dans l'acte d'embauche évoqué, *Anna* joue le rôle du sujet d'*embaucher*, mais ce n'est en général pas explicité dans les

1. Ma contribution au projet se situe en particulier dans la définition du schéma d'annotation (en particulier avec Guy Perrier), et l'organisation et mise au point de règles de conversion arbre de dépendance vers graphe syntaxique profond (avec Corentin Ribeyre). Dans le cadre de sa thèse, encadrée par Djamé Seddah et Eric de la Clergerie, Corentin a mis au point un analyseur (Ribeyre et al., 2015) produisant directement des graphes syntaxiques profonds, en utilisant comme corpus d'apprentissage le sequoia-deep (validé manuellement), et un corpus pseudo-gold plus important (le FTB annoté automatiquement converti en syntaxe profonde, à partir des arbres de dépendances gold (Ribeyre et al., 2014)). Dans le chapitre 7, j'utilise ces données pour entraîner un analyseur vers graphes de dépendances.

2. https://deep-sequoia.inria.fr/fr/, l'annotation en syntaxe profonde du corpus sequoia est accessible en téléchargement. Le site de Grew, de Bruno Guillaume : http://match.grew.fr/?corpus=sequoia.deep_and_surf@master permet de visualiser les graphes syntaxiques profonds.

corpus arborés [3]. On peut également citer la coordination de VP, par exemple dans *Anna terminera son mémoire le 10 et l'enverra au correcteur dans la foulée*, l'information que c'est *Anna* qui joue le rôle du sujet de *enverra* n'est pas immédiatement accessible dans un arbre de dépendances.

— D'un autre côté, construire automatiquement une représentation sémantique (même superficielle) à partir d'une représentation syntaxique de surface est un problème ouvert. Il s'agit en particulier de désambiguiser les formes en identifiant des unités lexicales. Par exemple dans les deux structures transitives en (1)ab, des informations d'ordre sémantique sont nécessaires pour distinguer les sens de *investissent*. En outre, interpréter si les *12 millions* correspondent à l'investissement total ou bien à celui de chacun des deux groupes nécessite d'interpréter un contexte large et est actuellement hors de portée des traitements automatiques.

(1) a. Les forces de l'ordre investissent la place.
b. Les deux groupes investissent 12 millions.

D'où le terme de syntaxe profonde, comme un objet pratique, explicitant le plus possible les informations non ambiguës dérivables d'arbres de dépendances, en vue d'aider à une analyse sémantique ultérieure. Ce but pratique ne va certes pas sans choix théoriques, mais devrait limiter l'impact du terme "syntaxe profonde".

Nous avons donc fait le choix d'une représentation syntaxique profonde facilitant le plus possible l'identification ultérieure des arguments sémantiques des verbes, sous la forme d'un graphe d'arcs bilexicaux, i.e. un graphe de "dépendances syntaxiques profondes", comprenant :

— l'explicitation du partage d'arguments syntaxiques de verbes, car c'est un phénomène très contraint syntaxiquement (contrôle, coordination...) ;
— la neutralisation des alternances syntaxiques dans le but de pouvoir facilement réaliser l'appariement entre arguments syntaxiques et arguments sémantiques : typiquement faciliter de repérer que *les deux groupes* correspondent au même argument sémantique d'*investir* dans *Anna fait investir 12 millions aux deux groupes* et *12 millions ont été déjà investis par les deux groupes* ;
— et le repérage explicite des marqueurs grammaticaux sémantiquement vides.

Nous avons défini ce schéma d'annotation (section 6.2), écrit des règles de transformations de graphe (section 6.3.1) gérant les phénomènes traitables de manière complètement déterministe ou via heuristiques quasi-déterministes, puis réalisé la validation manuelle des graphes obtenus pour le corpus Sequoia (section 6.3.2).

6.1 Travaux reliés

Les travaux reliés à citer concernent les efforts d'annotation de corpus intégrant des informations "non surfaciques", i.e. soit allant au-delà de simples arbres syntagmatiques (sans traces), soit des arbres de dépendances. Certains de ces travaux sont directement associés à des théories syntaxiques précises, et mettent en jeu plusieurs couches d'annotations, représentant

3. Pour l'anglais, le Penn TreeBank représente effectivement les informations de contrôle sous la forme de traces (noeuds vides coindicés avec un noeud plein), mais pour l'écrasante majorité des travaux utilisant le PTB comme corpus d'entraînement, la première action est de supprimer les traces. Pour le français, le choix théorique du French Treebank a été de n'encoder que des informations surfaciques (Abeillé et al., 2004).

le passage entre une forme de surface (une phrase) et sa représentation sémantique.

C'est le cas du corpus AnCoraUPF (Mille et al., 2013), qui comprend 3513 phrases en espagnol, et qui suit la théorie sens-texte (Melčuk, 1988) (MTT pour meaning-text theory) en fournissant quatre niveaux de représentation des phrases du corpus : morphologique, syntaxe de surface, syntaxe profonde et sémantique. Contrairement à l'approche que nous avons choisie, les structures syntaxiques profondes restent des arbres, et les relations actantielles représentées sont uniquement celles explicites dans la structure de surface [4]. La syntaxe profonde dans AnCoraUPF explicite sous forme de trait des phénomènes comme les auxiliaires, et gère également la coréférence résolvable par la syntaxe, i.e. les antécédents des relatifs dans le cas d'une relative modifiant un nom. La structure actantielle complète n'est obtenue qu'au niveau de la représentation sémantique, qui est cette-fois ci un graphe. Les étiquettes actantielles sont des numéros (arg1, arg2 etc...), qui ne peuvent être interprétées qu'au moyen d'une entrée lexicale associée au prédicat. À côté de ce corpus dont les annotations ont été manuellement validées, des travaux concernent la production automatique de ce type d'annotation, mais qui s'arrêtent au niveau syntaxique profond, la structure sémantique étant beaucoup moins accessible automatiquement. Ballesteros et al. (2014) utilisent un système déterministe de transducteurs d'arbres pour produire le niveau syntaxe profonde. En comparaison, la représentation syntaxique profonde que nous définissons va un peu plus loin (avec la complétion de la structure argumentale des verbes et adjectifs), tout en conservant, à quelques exceptions près, la possibilité d'obtenir ces annotations de manière déterministe à partir d'un arbre syntaxique de surface parfait.

Le Prague Dependency Bank (déjà cité chapitre 3, section 3.1) définit un niveau de représentation "analytique" (un arbre de dépendances de surface) et un niveau "tectogrammatical", où les noeuds sont des unités sémantiques (mots simples pleins ou bien expressions polylexicales), et les relations prédicat-argument sont typées en utilisant des rôles sémantiques grossiers.

Pour l'anglais, suite à la première version du Penn Treebank en 1993, d'autres ont suivi où les structures syntagmatiques hors-contexte ont été manuellement augmentées avec annotations non-surfaciques (Marcus et al., 1994) : des traces (noeuds vides coïndicés avec des éléments explicites) permettent de représenter les dépendances non locales (cf. chapitre 3, section 3.2, Figure 3.4), ainsi que le sujet des infinitifs dans le cas de phénomènes de contrôle ou de montée. Si la plupart des analyseurs appris sur le PTB ignorent ces informations, elles ont été cependant utilisées pour extraire des grammaires probabilistes couvrantes dans des formalismes lexicalistes tels que CCG, LFG ou HPSG (Hockenmaier, 2003 ; Cahill et al., 2004 ; Miyao and Tsujii, 2005), et construire des parsers capables de prédire des structures riches, dépassant la syntaxe de surface. Le "DeepBank" (Flickinger et al., 2012) contient des analyses syntaxiques HPSG et des représentations sémantiques de type Minimal Recursion Semantics (Copestake et al., 2005), corrigées manuellement à partir des sorties d'un analyseur.

Une simplification de ces analyses syntaxico-sémantiques riches ont été dérivées de ces ressources, sous la forme de "graphes de dépendances sémantiques", avec des arcs bilexicaux, qui ont été utilisés pour la compétition internationale "broad-coverage semantic parsing" (Oepen et al., 2014) une tâche d'analyse superficielle, sans mise en lien avec entrées lexicales, ni traitement des phénomènes de portée, pour l'anglais (Oepen et al., 2014), puis pour l'anglais, le tchèque et le chinois (Oepen et al., 2015). Les ressources produites sont proches de

4. Kahane (2009) propose une définition concurrente des représentations syntaxiques profondes dans la MTT, où il semble que le partage d'argument par exemple dans le cas de verbes à montée soit pris en compte dès la syntaxe profonde, donnant lieu à des graphes.

nos graphes syntaxiques profonds, mais ne sont pas directement dérivables à partir d'arbres syntaxiques. On traite de la tâche de parsing vers graphes bilexicaux au chapitre 7.

Toujours pour l'anglais, à côté de ces analyseurs profonds adossés à des formalismes linguistiques complets, les Stanford dependencies (de Marneffe et al., 2006 ; de Marneffe and Manning, 2008) ont été proposées comme une série de schémas de représentations en dépendances, obtenues par transformation déterministe d'abord d'arbres de constituants de type X-barre, puis par transformation des dépendances, pour aller vers une représentation plus sémantique (par exemple, les prépositions régies sont court-circuitées, certains infinitifs contrôlés reçoivent explicitement un sujet). L'objectif annoncé est que les progrès en analyse syntaxique automatique réalisés dans la communauté TAL soient utilisables facilement pour des chercheurs ou industriels d'autres domaines, pour des tâches d'analyse sémantique superficielle. Pour cette raison, les Stanford dependencies se veulent plus simples d'utilisation que des structures LFG ou HPSG.

Notre travail sur la syntaxe profonde reprend l'objectif pratique des Stanford Dependencies de mieux mettre à profit les informations syntaxiques pour obtenir une représentation dépassant la syntaxe de surface, grâce à des traitements en grande partie déterministes. Nous choisissons de partir d'arbres de dépendances (et pas de structures plus riches de type LFG ou HPSG) étant donné que les analyseurs en dépendance se sont généralisés et il est très facile d'obtenir l'arbre en dépendance pour une phrase en français, pour peu que le style soit relativement proche d'un style journalistique, cf. les corpus d'apprentissage disponibles).

Notre proposition est également proche du niveau tectogrammatical du Prague Dependency Bank, mais nous restons à un niveau plus syntaxique, en n'introduisant pas d'étiquettes de rôles sémantiques, car elles ne sont pas prédictibles directement à partir des seuls arbres de dépendances de surface. Un autre traitement plus syntaxique concerne les expressions polylexicales, qui ne sont pas fusionnées en un seul noeud, mais sont représentées au moyen d'arcs spécifiques ou bien de relation syntaxiques régulières, selon que l'expression polylexicale est syntaxiquement régulière ou pas (cf. chapitre 4). En effet, en l'absence d'un traitement sémantique complet des EPs, conserver les relations syntaxiques est nécessaire, notamment pour interpréter les modifications syntaxiques sur les composants d'EPs (par exemple pour interpréter l'intensifieur *très* dans *à très court terme*).

On peut également citer les "enhanced dependencies"[5], postérieures à notre travail, qui ont été proposées dans le cadre des Universal Dependencies, comme un schéma multilingue de représentations dépassant la syntaxe de surface pour aider à des tâches d'analyse sémantique superficielle. Nous avons proposé une version des "enhanced dependencies" intégrant la neutralisation des alternances syntaxiques (Candito et al., 2017).

6.2 Principales caractéristiques du schéma d'annotation

On décrit ici brièvement les deux phénomènes principaux traités, à savoir l'explicitation des partages d'arguments et la neutralisation des alternances syntaxiques[6].

5. https://universaldependencies.org/u/overview/enhanced-syntax.html
6. Le guide complet d'annotation est disponible à https://deep-sequoia.inria.fr/.

Des étiquettes qui restent syntaxiques

Dans les graphes syntaxiques profonds que nous proposons, les dépendances restent étiquetées au moyen de fonctions grammaticales, y compris pour des arcs non surfaciques : par exemple pour *Anna décide d'embaucher Kim*, on considère un arc profond $Anna \xrightarrow{suj} Kim$.

Considérer que *Anna* est le "sujet" de *embaucher* est discutable, étant donné qu'il n'a pas toutes les caractéristiques d'un sujet. Une alternative est d'utiliser des étiquettes de type rôles sémantiques, ou une simple numérotation d'actants sémantiques (ARG0, ARG1 ...). C'est l'option suivie dans les données utilisées pour la compétition "broad-coverage semantic dependency parsing" citée supra (Oepen et al., 2014). Il s'agit de données issues d'analyses riches, mais simplifiées en coupant tout lien avec un lexique sémantique.

Mais, d'une part, d'un point de vue théorique, il nous semble justifié que des phénomènes entièrement déterminés par la syntaxe (comme le contrôle ou le partage du sujet en cas de coordination de VP) relèvent d'une représentation qui reste syntaxique, et donc avec des étiquettes syntaxiques.

D'autre part, du point de vue pratique, pour des phénomènes plus ambigus, tant que le lien avec une entrée d'un lexique n'est pas fait, il est nécessaire de conserver les informations grammaticales. D'abord les fonctions grammaticales (canoniques) seront plus claires pour les arguments obliques qu'une numérotation des actants : par exemple, il n'est pas si évident de décider comment numéroter les arguments obliques de *X parle de Y à Z*. Ensuite, les fonctions grammaticales résument des marques formelles pouvant aider à désambiguiser dans certains cas. Par exemple représenter avec la même paire d'étiquettes (ARG0,ARG1) les arguments de *Anna parle italien* et ceux de *Anna parle de l'Italie* est contreproductif en l'absence de lien avec un lexique.

Des fonctions aux fonctions canoniques

L'objectif reste cependant d'expliciter le plus possible les arbres syntaxiques, dans le but de faciliter l'analyse sémantique, et donc l'identification des arguments sémantiques. Ceci amène naturellement à vouloir gérer certaines alternances syntaxiques, en particulier le passif, où là encore les marques morpho-syntaxiques sont dans la plupart des cas suffisantes pour réaliser l'appariement avec les arguments sémantiques, donc autant expliciter celui-ci.

Capturer les alternances syntaxiques et le lien entre diathèses différentes un objectif de toute théorie linguistique. Une approche procédurale a été proposée dans la grammaire transformationnelle à ses débuts, et également dans les premiers temps de la Lexical Functional Grammar (LFG), avec la notion de "règles lexicales", qui cependant déplacent le traitement des alternances au niveau du lexique. Ce traitement procédural a été abandonné ensuite. De manière plus ambitieuse, certaines théories, en particulier les développements ultérieurs de LFG, cherchent à capturer les régularités de "linking", i.e. les régularités dans l'assignation des fonctions grammaticales aux arguments syntaxiques, en évitant une approche procédurale.

Mais dans notre cas, en l'absence de lexique sémantique, représenter les alternances syntaxiques en n'utilisant que des étiquettes syntaxiques est la seule option. Nous nous inspirons en cela de la Grammaire Relationnelle (Perlmutter, 1983) en distinguant des **fonctions grammaticales finales** et **fonctions grammaticales canoniques**[7], et en considérant les alternances

7. La Grammaire Relationnelle utilise l'adjectif *initial* plutôt que *canonique*.

syntaxiques comme la redistribution des fonctions grammaticales associées aux différents arguments syntaxiques. Par exemple dans *Anna a été embauchée par TooGoodToGo*, on considère que *Anna* est l'objet canonique du verbe, et est réalisé comme son sujet final.

En conclusion, **en l'absence de recours à un lexique sémantique et d'explicitation des rôles sémantiques, utiliser les fonctions canoniques nous permet de désigner uniformément les arguments sémantiques, en neutralisant les alternances syntaxiques**. L'objectif est de faciliter le repérage ultérieur des arguments sémantiques, avec ressources lexico-sémantiques supplémentaires (comme un lexique sémantique, ou une annotation en rôles sémantiques). On quantifie chapitre 8 (section 8.1.1) le degré de normalisation ainsi obtenu dans l'appariement entre fonctions grammaticales et rôles sémantiques à la FrameNet.

À noter cependant que les fonctions grammaticales finales restent nécessaires pour le calcul sémantique complet, en particulier pour ce qui est des phénomènes de portée. Par exemple, en (2), la portée de *tout le monde* diffère selon qu'il est sujet (final) (2)a versus complément d'agent (2)b. Plus généralement, les arcs profonds ajoutés en représentation profonde peuvent masquer des phénomènes de portée.

(2) a. Tout le monde maîtrise au moins deux langues.
b. Deux langues sont maîtrisées par tout le monde.

6.2.1 Alternances syntaxiques

Nous n'avons considéré que les redistributions qui comportent un marquage morpho-syntaxique (typiquement l'auxiliaire pour le passif, ou le clitique sémantiquement vide *se* pour les alternances moyennes et neutres). Dans toute la suite, on utilise des étiquettes doubles, i.e. de la forme xxx:yyy pour représenter un xxx final et un yyy canonique.

Nous retenons comme redistributions [8] :

- le passif :
  - Exemple : *Anna*$_{suj:obj}$ *a été intérrogée par la police*$_{par-obj:suj}$.
  - le sujet final est l'objet canonique et le complément d'agent final est le sujet canonique.
- l'impersonnel :
  - Exemple : *il*$_{suj:\_}$ *est arrivé trois personnes*$_{obj:suj}$
  - le NP postverbal est considéré comme un objet final (il en a les propriétés, comme la possibilité du *en* quantitatif), mais comme le sujet canonique. Le *il* explétif est sujet final, mais n'a pas de fonction canonique.
- le médiopassif et l'anticausatif :
  - Exemples : *Une vitre*$_{suj:obj}$ *comme ça ne se brise pas avec un marteau* versus *La vitre*$_{suj:obj}$ *s'est brisée tout d'un coup*, distingué par une interprétation comprenant forcément l'argument sujet canonique (celui qui brise) ou pas.
  - Dans les deux cas le sujet final est objet canonique.

8. Nous avons repris le recensement des alternances fait dans (Candito, 1999), sauf pour le réflexif (ou "vrai réfléchi"). Bien qu'il présente des propriétés intransitivantes, nous ne le traitons pas via redistribution, préférant une analyse permettant plus facilement de repérer qu'un des actants jouent deux rôles sémantiques, en lui faisant porter deux fonctions grammaticales canoniques (par exemple dans *Anna se défend*, *Anna* porte les fonctions sujet et objet canonique).

— le causatif :
  — Les constructions causatives en français sont bien connues, avec notamment l'étude de Kayne (1975), pour mettre en jeu un type particulier de complémentation infinitive. Abeillé et al. (1997) montrent une compétition entre une construction de type complexe verbal (*faire* + Vinf), versus une construction, plus rare, où l'infinitive est un argument syntaxique du verbe *faire*[9]. Dans le cas d'un complexe causatif *faire* + Vinf), celui-ci a un argument supplémentaire par rapport au Vinf, l'argument causateur, réalisé comme sujet du complexe causatif. Dans notre cadre utilisant les notions de fonction canonique, on considère cet argument comme le sujet final, et on introduit une fonction canonique spécifique (argc, pour "argument causateur"). Selon la transitivité (en contexte) du Vinf et ses propriétés sémantiques, le sujet canonique du Vinf est quant à lui réalisé comme un objet direct (*les élèves* infra), un complément en *par* (*les agents de propreté* infra) ou un complément en *à* (*les Olympiades* infra) :
  — causatif d'un intransitif : *L'enseignant*$_{suj:argc}$ *a beaucoup fait lire les élèves*$_{obj:suj}$
  — causatif en *par* d'un transitif : *La maire*$_{suj:argc}$ *fait installer des nouvelles poubelles*$_{obj:obj}$ *par les agents*$_{par-obj:suj}$ de propreté
  — causatif en *à* d'un transitif : *Les Olympiades*$_{suj:argc}$ *lui*$_{a-obj:suj}$ *ont fait aimer l'informatique*$_{obj:obj}$

À noter que cette fonction canonique argc n'existe pas comme fonction finale, ce qui argumente contre le statut syntaxique donné à la notion de fonction canonique, pris ici dans un but pratique.

6.2.2 Partage d'arguments

L'autre type principal de phénomène traité est celui du partage d'arguments, en particulier du partage d'arguments de verbes, qu'il s'agisse du sujet (final) des verbes non conjugués, ou d'arguments partagés entre verbes coordonnés.

Dans le schéma d'annotation on vise d'expliciter tout partage relevant de ces types, qu'il s'agisse (i) de cas dérivables de manière déterministe à partir des seuls arbres de dépendances de surface ou (ii) de cas nécessitant l'explicitation d'informations sémantiques, dont certains peuvent montrer empiriquement une grande régularité syntaxique. Par exemple la grande majorité des sujets des infinitifs dans une infinitive de but sera le sujet du verbe principal, si celui-ci n'est pas au passif (exemple en (3), et contre-exemple en (4)).

(3) Un groupe de travail [...] s'est réuni pour analyser les propositions [...] [annodis.er_00279]

(4) Quelques mois auront été nécessaires pour mettre en scène le Petit Prince. [annodis.er_00141]

L'annotation manuelle d'un corpus (section 6.3) permet de quantifier les deux types de phénomènes, en mesurant l'écart entre le corpus de référence et le résultat de règles déterministes appliquées à des arbres de dépendance de surface (section 6.3.2, table 6.1).

Pour donner une idée du type de traitements réalisés, on ne détaille ci-dessous que deux phénomènes massifs de partage d'arguments, le contrôle et le partage du sujet de verbes ou

9. On trouvera un résumé de la littérature motivant ces deux analyses dans (Candito, 1999, 157-159). Les données mettent en jeu entre autres la position des clitiques, l'impossibilité de nier l'infinitif, l'interaction avec le réfléchi.

VPs coordonnés.

Contrôle et montée

L'interprétation d'une infinitive fait intervenir un argument, qui serait interprété comme le sujet (final) du verbe si celui-ci n'était pas infinitif : en (5)a, on interprète *Jean* comme l'argument fumeur. Plus rapidement, on parlera de sujet (final) de l'infinitif. Celui-ci peut ne pas être présent dans la phrase (avec une interprétation spécifique dans un contexte plus large, ou une interprétation générique comme en (5)a). S'il est présent dans la phrase, on distingue le contrôle obligatoire et le contrôle arbitraire ((5)b) (Baschung, 1996), selon qu'il y a ou pas une contrainte dure sur quel argument sera interprété comme le sujet de l'infinitive. La contrainte est déclenché par un élément lexical, et est indépendante de la sémantique des actants(6).

(5) a. Jean pense que fumer est dangereux pour la santé.
b. Jean$_i$ pense que fumer$_i$ est dangereux pour sa$_i$ santé. (Baschung, 1996)

(6) La table$_i$ voudrait manger$_i$ le courage.

En syntaxe profonde, on annote systématiquement une dépendance directe entre un infinitif et son sujet s'il est présent dans la phrase, qu'il s'agisse d'un contrôle arbitraire ou obligatoire. On fait de même avec les verbes à montée [10].

À noter que la généralisation pertinente pour les verbes à contrôle est qu'un verbe à contrôle détermine lequel de ses arguments **sémantiques** sera interprété comme le sujet (final) de l'infinitif, ce qui peut se voir en comparant un verbe à contrôle à l'actif et au passif. Par exemple pour un verbe à contrôleur objet comme *encourager*, le sujet de l'infinitif introduit sera le sujet (final) de *encourager* à l'actif (Figure 6.1 en haut à gauche), mais l'objet (final) de *encourager* au passif (Figure 6.1 en haut à droite). Le contrôle se fait par contre toujours sur le sujet final de l'infinitive, peu importe sa voix (Figure 6.1 en bas). Ici en l'absence d'explicitation des rôles sémantiques, on utilise les fonctions canoniques pour désigner uniformément les arguments sémantiques, et on capture la régularité que le sujet **canonique** de *encourager* est le sujet **final** de l'infinitive introduite.

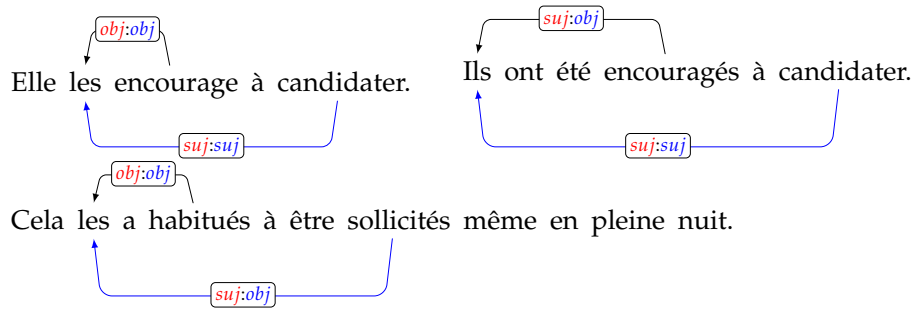

FIGURE 6.1 – Verbe à contrôleur objet : l'**objet canonique** du V à contrôle est le **sujet final** de l'infinitif.[a].

[a] **Dans les figures de ce chapitre**, on note en bas et bleu les arcs profonds non présents dans l'arbre de surface, et au-dessus les arcs appartenant à l'arbre de surface : en rouge les noeuds et les arcs non conservés dans le graphe profond, et en **noir** les arcs à la fois surfaciques et profonds. Les doubles étiquettes *xxx*:*yyy* signifient fonction finale *xxx* et fonction canonique *yyy*.

10. La différence entre montée et contrôle n'est pas traitée au niveau de la syntaxe profonde. Il s'agirait de représenter que le sujet du verbe à montée n'en est pas un argument sémantique.

On a également des noms à contrôle (en (7), l'argument en *de* de *intention* est le sujet de l'infinitive) et des adjectifs à contrôle (en (8) : l'argument sémantique premier de *capables* est le sujet final de l'infinitive). À noter également la possibilité plus rare mais attestée où le nom est interprété comme l'objet de l'infinitif, comme dans *un repas prêt à manger*.

(7) L'intention de Paul$_{suj:suj}$ est de **finir** tôt.

(8) Ils$_{suj:suj}$ sont capables de **réciter** (un poème).

Partage du sujet de verbes ou VPs coordonnés

Un tout autre cas de partage d'arguments déterminés par la syntaxe est le cas de la coordination de verbes (ou de VPs). De manière systématique, on ajoute en syntaxe profonde le sujet (final) du premier conjoint comme sujet final de tous les autres conjoints. Ceci s'applique dans le cas de verbes conjugués (9), mais plus généralement quel que soit le mode des verbes coordonnés, comme par exemple des participes, actifs ou passifs (10), ou infinitifs (11).

(9) Anna$_{suj:suj}$ prend son parapluie et **sort**.[11]

(10) a. Ce travail$_{suj:obj}$ a été reconnu par l'état et largement **enseigné** au collège.
b. Les fonctionnaires$_{suj:obj}$ n'ayant pas à l'époque de statut protecteur et étant **considérés** comme des agents du gouvernement [...] [sequoia-frwiki_50.1000_00305]
c. L'incidence des fractures cliniques [...] a été évaluée chez 2 127 hommes$_{suj:obj}$ [...] ayant une fracture de hanche récente [...] et **suivis** sous traitement pendant environ 2 ans. [sequoia-emea-fr-test_00254]
d. [...] salaires$_{suj:obj}$ ne correspondant à aucun travail effectif et **reversés** à Michel Giraud. [sequoia-frwiki_50.1000_00621]

(11) des pêcheurs$_{suj:suj}$ venus nettoyer les rives et **curer** le ruisseau. [sequoi-annodis.er_00041]

À noter que les verbes conjoints n'ont pas forcément la même voix (voir Figure 6.2), la régularité est que les participes partagent le même sujet final.

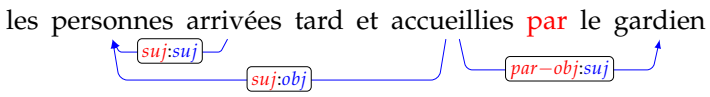

FIGURE 6.2 – Coordination de participiales modifiant un nom : chaque participe est le sujet final du nom modifié (et la fonction canonique dépend de la voix).

À noter également que le premier conjoint de la coordination verbale peut lui-même ne pas avoir de sujet dans l'arbre de surface, et le recevoir dans le graphe profond, comme *arrivées* Figure 6.2, ou *nettoyer* Figure 6.3.

6.2.3 Fonction finale ou canonique d'arcs profonds

Nous pouvons maintenant justifier a posteriori de ne pas avoir utilisé les termes "fonctions profondes" / "fonctions de surface" : **utiliser des fonctions finales sur des arcs profonds**

11. Dans cette série, le V deuxième conjoint est en **gras**, et le sujet final qui lui est ajouté est souligné.

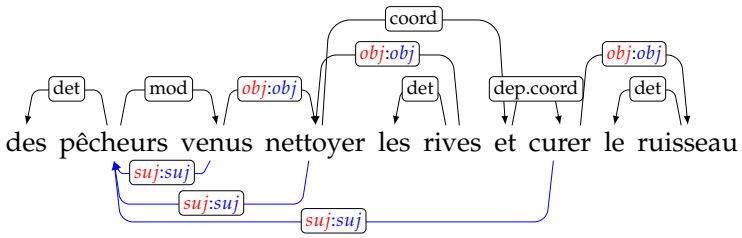

FIGURE 6.3 – Coordination d'infinitives : tous les infinitifs conjoints ont leur sujet final ajouté dans le graphe profond.

(et pas uniquement directement des fonctions canoniques) permet de capturer des généralisations syntaxiques lors du traitement des interactions entre les alternances syntaxiques et le partage d'argument.

Par exemple, un nom modifié par un participe est toujours le sujet *final* du participe (cf. Figure 6.2 supra). Pour un verbe à contrôle, c'est le sujet final de l'infinitif qui est entièrement déterminé quelle que soit sa voix (cf. supra Figure 6.1). Ou encore, deux verbes coordonnés partagent leurs sujets finaux quelles que soient leurs voix. Il est donc important que même des arcs profonds (i.e. ajoutés en syntaxe profonde) aient bien une fonction finale, et pas seulement directement une fonction canonique.

Ainsi peut-on traiter de manière uniforme les alternances syntaxiques pour des arcs profonds aussi bien que pour des arcs déjà présents en surface : par exemple en (13), le sujet final (profond) du verbe au passif *respecté* en est son objet canonique (profond).

De la même manière, les régularités de partage d'argument (contrôle, montée, coordination de verbes...) s'appliquent indépendamment du statut surfacique ou profond des arcs. Ainsi par exemple pour un verbe V à contrôleur sujet, introduisant une infinitive de tête Vinf, la régularité est que si le verbe V a un sujet final, alors il le partage avec le verbe infinitif qu'il introduit, et ceci vaut que l'arc de départ $V \xrightarrow{suj(final)} X$ soit présent dans l'arbre de surface (comme en (12) $veut \xrightarrow{suj(final)} Jean$) ou ajouté en syntaxe profonde (comme en (13) $vouloir \xrightarrow{suj(final)} Jean$). Ces régularités peuvent s'agencer en cascade, comme pour l'exemple réel (14), où *les députés du Bundestag* sont le sujet final de quatre verbes.

(12) Jean veut parler.

(13) Jean semble vouloir parler et être respecté.

(14) Les députés du Bundestag ne peuvent pas être appelés à témoigner ou être arrêtés pour une infraction [sequoia-Europar.550_00544]

En pratique on gère ce type de partages en cascade grâce au système de contraintes du système de réécriture de graphes O.G.R.E, de Corentin Ribeyre (Ribeyre, 2016), non détaillé ici.

6.2.4 Interactions

Comme on peut le voir sur certains des exemples précédents, les différents phénomènes traités en syntaxe profonde interagissent fortement, comme illustré avec quelques exemples supplémentaires Figure 6.4.

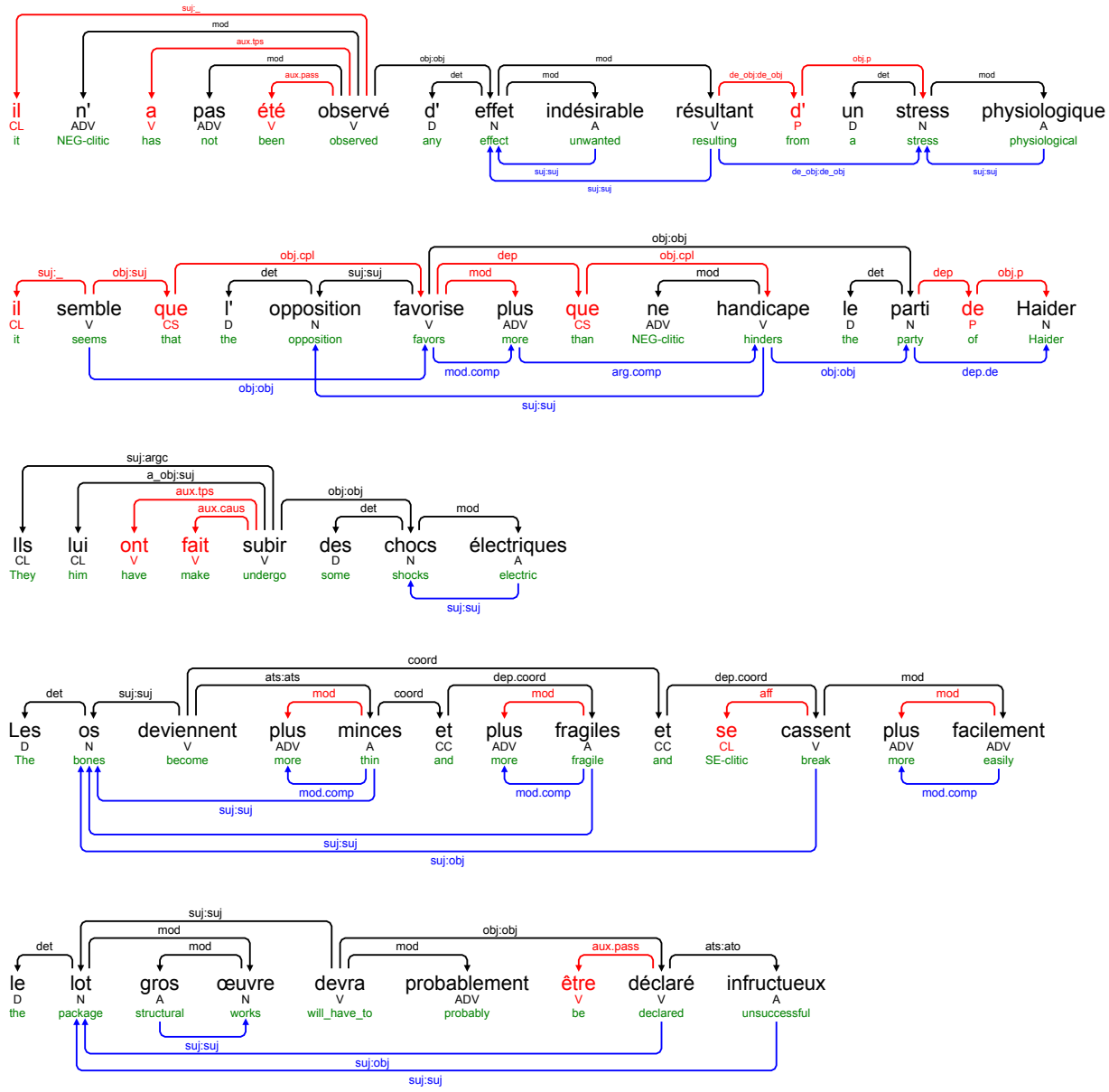

FIGURE 6.4 – Exemples de graphes syntaxiques profonds avec interaction de phénomènes (figure générée avec le système grew-parse-fr de Bruno Guillaume http://parse.grew.fr/).

6.3 Annotation semi-automatique

Dans le cadre du projet Deep-sequoia, le corpus sequoia a été annoté en syntaxe profonde. La méthodologie générale a été de :

— mettre au point deux jeux de règles de transformation d'arbres de dépendances en graphes de dépendances profondes, basés sur les moteurs de transformation de graphe OGRE (Ribeyre et al., 2012) et GREW (Guillaume et al., 2012) ;

— les appliquer au corpus Sequoia, faire corriger manuellement indépendamment les deux sorties automatiques, puis adjudiquer les conflits.

6.3.1 Conversion par règles de transformation de graphes

On résume ici l'organisation du module de conversion des arbres syntaxiques de surface vers des graphes syntaxiques profonds (ci-après surf2deep), qui utilise le système de réécriture de graphes OGRE (Ribeyre, 2016), en particulier le système de contraintes développé pour gérer de manière économe le partage d'arguments "en cascade".

Organisation des règles

Les règles utilisent des motifs structurels, mais également des informations lexicales (par exemple listant quels sont les verbes, noms, adjectifs à contrôle). On n'utilise pas en revanche d'information sur la transitivité des verbes, ce qui limite les traitements [12].

Le module surf2deep utilise cinq jeux de règles appliqués séquentiellement : au sein d'un jeu de règles l'ordre n'importe pas, en revanche les cinq jeux sont effectivement ordonnés (le graphe d'entrée de l'un est le graphe de sortie du précédent).

1. le premier jeu de règles explicite le mode verbal, en convertissant les auxiliaires de temps en traits "profond" portés par le verbe lexical, et en gérant la coordination. Par exemple au base de la Figure 6.4, *déclaré* est participe passé au niveau surfacique, mais porte un mode profond infinitif. Cette normalisation simplifie les règles suivantes.
2. le second jeu de règles ajoute les sujets finaux de verbes [13]. Le partage du sujet de verbes coordonnés est traité par contrainte de partage, ainsi que le contrôle et la montée.
3. le 3ème jeu gère les alternances syntaxiques [14]. Sauf rares exceptions, le passif peut être repéré et traité directement sur la seule base des arbres de surface. En revanche, les règles pour l'impersonnel, le se-moyen et neutre, et le causatif ne sont opérantes que si une annotation préalable spécifique à ces phénomènes a été réalisée (par exemple marquant les *il* impersonnels, cf. infra), soit manuellement soit automatiquement par un autre biais.

12. Par exemple savoir que *rencontrer* est transitif permet d'ignorer l'interprétation impersonnelle dans *Il rencontre trois personnes*, inversement, pour *Il entre trois personnes*, savoir qu'*entrer* est intransitif force l'interprétation impersonnelle.

13. Il traite également des sujets des adjectifs, que ceux-ci soient épithètes ou attributs. Nous omettons cette partie pour simplifier.

14. Une distinction complète entre partage du sujet final et alternances syntaxiques n'est pas possible car la généralisation pour les verbes à contrôle concerne fonction finale (pour le sujet de l'infinitif) et fonction canonique (pour identifier quel argument du verbe à contrôle est sujet de l'infinitif). Cette interaction est résolue en déclinant explicitement dans le 2ème jeu de règles, des règles pour les verbes à contrôle à l'actif et au passif : *encourager* à l'actif est un verbe à contrôleur objet, et *être encouragé* est un verbe à contrôleur sujet.

4. les modules 4 et 5 gèrent divers détails omis ici, dont le court-circuitage de marqueurs grammaticaux.

Les cinq jeux de règles comptent respectivement 19, 40, 21, 39 et 36 règles, pour un total de 155 règles. Cela reste une taille raisonnable, mais constitue un système au final assez complexe à maintenir. Pour tous les phénomènes, l'interaction avec la coordination est un facteur majeur de complexification des règles.

6.3.2 Annotation du corpus deep-Sequoia et caractéristiques quantitatives

L'annotation s'est faite en plusieurs étapes, avec sélection de phrases pilotes pour la mise au point du guide et des règles de conversion, pré-annotation automatique en utilisant les deux systèmes de transformation de graphes distincts (via OGRE à Paris et via GREW à Nancy), puis double validation manuelle et adjudication des conflits. Nous renvoyons à (Candito et al., 2014b) pour la méthodologie précise et l'évaluation de l'accord inter-annotateur, qui est globalement très haut, y compris pour les arcs profonds. On y trouve également une évaluation des règles de conversion, réalisée en comparant, sur des phrases non utilisées pour la mise au point des règles, les annotations profondes validées manuellement et la sortie de la conversion automatique. L'évaluation est rapportée à la Table 6.1. Il s'agit d'une évaluation globale, où les erreurs peuvent provenir d'une insuffisance des règles (par exemple un verbe à contrôle manquant), ou bien des heuristiques non infaillibles introduites pour gérer des cas non déterministes. On constate que les erreurs sont quantitativement marginales, et qu'environ la moitié d'entre elles sont résolues si l'on fournit en amont une annotation des *il* impersonnels, du statut du clitique réfléchi, et du sujet canonique des causatifs.

Type d'entrée	train surf.		train surf. + manuel amont	
Types d'arcs évalués	profonds	profonds non surf.	profonds	profonds non surf.
UF	98,2	95,6	99,2	97,4
LF	96,1	94,7	99,1	97,1

TABLE 6.1 – Evaluation du jeu de règles OGRE deep2surf : comparaison entre graphes profonds de référence (gold) et graphes profonds obtenus par les règles deep2surf, sur la partie "train" du sequoia. Les règles sont soient appliquées directement aux arbres de dépendances gold ("train surf"), ou bien aux arbres gold avec annotations manuelles amont citées supra ("train surf + manuel amont").

On fournit ici quelques comptages sur corpus résultant [15] permettant de mieux comparer les arbres de surface et les graphes syntaxiques profonds.

La Table 6.2 montre qu'environ 13,6% des tokens du Sequoia sont ignorés en syntaxe profonde (auxiliaires, prépositions et complémenteurs régis). On obtient au final des graphes profonds où un token a en moyenne 1,12 gouverneurs syntaxiques.

Parmi les 66760 arcs des graphes profonds, environ 21,4% n'étaient pas dans les arbres de surface. Ce chiffre mélange cependant les arcs ajoutés du fait du court-circuitage des

15. Les comptes sont faits sur la version 7.0, qui contenaient des expressions polylexicales fonctionnelles uniquement, comptées ici comme un seul token.

Arbres de dépendances de surface	
Nb arcs = nb tokens	68802
↳arcs supprimés dans les graphes profonds	16324 (23,7%)
↳tokens supprimés dans les graphes profonds	9348 (13,6%)
Graphes de dépendances profondes	
Nb tokens	59454
Nb arcs	66760
↳dont arcs non surfaciques	14282 (21.4%)
Nb arguments de verbes dans arbres de surface	9589
Nb arguments de verbes dans graphes profonds	11629
↳dont arguments avec fonction finale $\neq$ fonction canonique	2012

TABLE 6.2 – Analyse quantitative du corpus deep-Sequoia

prépositions et complémenteurs régis, et les arcs argumentaux réellement ajoutés [16], qui sont autrement plus importants pour faciliter l'analyse sémantique.

Aussi, pour mieux évaluer l'impact de l'annotation en syntaxe profonde en termes de structure argumentale, on fournit dans les deux avant-dernières lignes de la Table 6.2 le nombre d'arcs de type argument de verbe dans l'annotation de surface et dans l'annotation profonde, et on peut constater que **parmi les 11629 arguments profonds de verbes, 17,5% n'étaient pas présents en surface** (via préposition ou pas). Enfin, environ **17,3% des arguments de verbes ont une fonction canonique différente de la fonction finale** (2012 cas).

On fournit par ailleurs Table 6.3 la comparaison du mode verbal surfacique versus le mode "profond", où l'on a reporté sous forme de traits sur le verbe plein le mode en prenant en compte la présence d'éventuels auxiliaires, et en propageant le mode du premier conjoint aux conjoints suivants en cas de coordination. Sans surprise, on peut voir que la répartition des modes est très différente en syntaxe profonde, avec la proportion de verbes conjugués passant de 40,4 à 60,7%.

Mode	surfacique		profond	
fini	2583	(40,4%)	3878	(60,7%)
↳indicatif	2499		3756	
↳subjonctif	53		91	
↳impératif	31		31	
participe	2601	(40,7%)	1146	(17,9%)
infinitif	1211	(18,9%)	1368	(21,4%)

TABLE 6.3 – Répartition des différents modes des verbes pleins dans le Sequoia, en surface et dans les graphes profonds.

Enfin, on donne Table 6.4 la répartition des diathèses verbales (pour les verbes pleins). Environ 26% des occurrences de verbes ont une alternance annotée [17], le passif étant de loin

16. On a également les sujets des adjectifs, non détaillés dans ce chapitre mais compris dans ces statistiques.

17. On rappelle que dans les annotations, on n'a pas considéré le vrai réfléchi comme une alternance,

l'alternance la plus fréquente, avec environ autant de passifs personnels avec (15) et sans auxiliaire passif (16).

pas d'alternance	4709	
avec alternance	1686	
↳passif personnel	**1495**	
↳avec aux.	733	(15)
↳sans aux.	762	(16)
↳impersonnel	**105**	
↳impers actif	83	(17)
↳passif impers	22	(18)
↳médiopassif/anticausatif	**56**	(20) (19)
↳causatif	**30**	
↳suj canonique = obj final	10	(21)
↳suj canonique = compl. d'agent final	3	(22)
↳suj canonique = a-obj final	1	(23)
↳suj canonique non exprimé	11	(24)
↳se faire Vinf	5	(25)

TABLE 6.4 – Répartition des diathèses verbales dans le corpus deep-Sequoia.

(15) PASSIF AVEC AUX : Les autres troubles des ions minéraux doivent aussi être efficacement **traités** emea-fr-test_00522

(16) PASSIF SANS AUX : [...] grâce à ce trafic **entretenu** par l'avidité de certains. frwiki_50.1000_00963

On a une centaine d'alternance impersonnelle, dont 22 cas de passif impersonnel (18). Parmi ces derniers, la plupart proviennent du sous-corpus médical, mais on trouve quatre cas dans le sous-corpus fait de 500 phrases de *L'Est républicain*.

(17) IMPERSONNEL ACTIF : [...] il **existe** des indices suffisants pour envisager sa mise en examen frwiki_50.1000_00679

(18) PASSIF IMPERSONNEL [...] il est fortement **conseillé** d'administrer des suppléments appropriés de calcium emea-fr-test_00527

L'annotation des clitiques réfléchis a donné lieu à 56 cas de médiopassif (20) ou d'anticausatif (19) (non distingués dans l'annotation, les annotateurs devaient seulement déterminer si le sujet de surface joue le rôle de l'objet canonique dans la version sans *se*).

(19) ANTICAUSATIF : Elle lui donna sept enfants avant que le cercle familial ne s'**agrandisse** de quinze petits-enfants annodis.er_00167

(20) MEDIOPASSIF : Les valeurs minimales s'**observent** après 7 jours pour les marqueurs de résorption emea-fr-test_00203

ni les alternances sans marque formelle (comme l'anticausatif non marqué, par ex. *la corde a finalement cassé*).

Le causatif enfin correspond à environ 0.5% des occurrences de verbes pleins, et correspond à différents sous-types, le plus fréquent étant un complexe causatif avec un infinitif transitif avec objet direct exprimé, et pour lequel le sujet canonique de l'infinitif n'est lui pas exprimé (24). Dans l'annotation, on a utilisé un type à part pour toutes les occurrences de "se faire Vinf" (25), sans distinguer selon que le sujet final est effectivement un argument causateur ou pas (voir (Veecock, 2008) pour une étude de cette construction).

(21) CAUSATIF SUJ CANONIQUE=OBJ FINAL : Faire s'**exprimer** les enfants à travers cette activité, c'est important. annodis.er_00148

(22) CAUSATIF SUJ CANONIQUE=COMPLT D'AGENT FINAL : [...] l'assemblée [...] l'a chargé de faire **établir** par les services de la DDE un dossier de demande de subvention annodis.er_00011

(23) CAUSATIF SUJ CANONIQUE=A-OBJ FINAL : [...] ils lui ont fait **subir** des chocs électriques frwiki_50.1000_00096

(24) CAUSATIF SUJ CANONIQUE NON EXPRIMÉ : Ceux d'Ancerville ont participé à l'événement en faisant **découvrir** leur savoir-faire et leurs recettes. annodis.er_00048

(25) SE FAIRE VINF : la fatigue commençait à se faire **sentir** annodis.er_00226

6.3.3 Evaluation sur le FTB et corpus pseudo-gold

Nous avons également appliqué les règles surf2deep sur le FTB, en ayant au préalable annoté manuellement les *il* impersonnels, le sujet canonique des causatifs et et le statut des clitiques réfléchis *se*, de manière à améliorer la qualité des graphes produits.

Pour évaluer les graphes profonds obtenus, nous avons sélectionné au hasard 200 phrases du FTB (arbres de surface, Surf-Réf), corrigé l'annotation surfacique (Surf-Corr), et appliqué les règles surf2deep, soit à partir de Surf-Réf soit à partir de Surf-Corr, et enfin corrigé manuellement les 200 graphes profonds ainsi obtenus (Ribeyre et al., 2014).

On constate Table 6.5 que même sans la phase de correction manuelle des graphes profonds, on obtient une très bonne qualité (LF=97.3). Une bonne partie des erreurs provient d'erreurs dans l'annotation de surface (cf. on monte à LF=99.4 lorsque l'on part d'arbres de surface corrigés).

	Nb arcs gold	LF	UF
Surf-Réf évaluée par rapport à Surf-Corr	6170	98,4	99,7
Graphes profonds obtenus à partir de Surf-Réf	6012	97,3	98,7
Graphes profonds obtenus à partir de Surf-Corr	6012	99,4	99,5

TABLE 6.5 – Evaluation sur 200 phrases du FTB. LF/UF : F-mesure étiquetée/non-étiquétée. 1ère ligne : évaluation des arbres de surface initiaux par rapport à leur correction manuelle. 2ème ligne (resp. 3ème ligne) : évaluation des graphes profonds obtenus avec règles appliquées sur Surf-Réf (resp. Surf-Corr).

Les graphes profonds obtenus sur le FTB sont donc de qualité suffisante pour servir de pseudo-gold pour entraîner un analyseur produisant directement des graphes profonds, comme l'ont fait Ribeyre et al. (2016), et comme nous le faisons au chapite 7, section 7.2.

6.3.4 Intégration dans les "Universal Dependencies"

Indépendamment de ce projet français de syntaxe profonde, Schuster and Manning (2016) ont proposé une extension des Universal Dependencies, les "enhanced dependencies", reprenant les propositions des Stanford Dependencies de mieux expliciter des dépendances directes entre mots pleins.

L'objectif est assez comparable aux graphes profonds que nous venons de présenter. Une différence importante de notre proposition est celle de neutraliser les alternances syntaxiques en explicitant les fonctions canoniques. Nous avons adapté le système de règles présenté supra au schéma de représentation syntaxique des Universal Dependencies, produisant ainsi une version du Sequoia en "enhanced dependencies" plus neutralisation des alternances (Candito et al., 2017).

6.4 Bilan

J'ai présenté le travail collectif de conception et annotation de graphes syntaxiques profonds, définis dans l'objectif d'expliciter le plus possible d'informations syntaxiques en l'absence de tout recours à un lexique sémantique. J'ai du coup argumenté en faveur de l'utilisation de fonctions canoniques pour identifier de manière plus uniforme les arguments sémantiques des verbes.

Les 3099 phrases du corpus Sequoia ont été manuellement validées pour la syntaxe profonde, avec un fort accord inter-annotateur. Le système de règles de transformation de graphes permettant de produire de manière déterministe des graphes profonds à partir d'arbres de dépendances a été mis au point. La qualité évaluée des graphes obtenus est très bonne, ce qui permet de construire un corpus d'apprentissage pseudo-gold plus gros (le FTB en syntaxe profonde, obtenu en appliquant les règles de conversion à partir des arbres de dépendances gold).

Nous avons adapté les règles et corpus résultants au schéma subséquent des "enhanced dependencies" (Schuster and Manning, 2016) au sein des Universal Dependencies, pour s'inscrire dans une perspective multilingue.

Au vu des chiffres présentés section 6.3.2 sur les 3099 phrases du corpus Sequoia, on peut affirmer que le travail en syntaxe profonde d'explicitation de la structure argumentale des verbes a un poids statistique important, qu'il s'agisse de l'ajout d'arguments que de la neutralisation des alternances syntaxiques, deux phénomènes cruciaux pour aider l'analyse sémantique aval.

Ces annotations en syntaxe profonde continuent d'être maintenues au fur et à mesure que d'autres annotations sont ajoutées au corpus, donnant lieu à un corpus librement disponible [18], de plus en plus richement annoté : une ré-annotation complète des expressions polylexicales (cf. Chapitre 4), et une annotation de classes sémantiques réalisées par Barque et al. (2020).

Le chapitre suivant est consacré à l'analyse automatique vers des graphes bilexicaux, expérimentée sur ces graphes syntaxiques profonds.

18. https://deep-sequoia.inria.fr

Chapitre 7

Analyse automatique en graphes de dépendances

Je présente dans ce chapitre un travail personnel en cours (non publié) pour la production automatique de graphes de dépendances bilexicales. L'objectif est d'intégrer, au sein d'un analyseur biaffine, un peu d'interdépendance entre les décisions de rattachements bilexicaux.

La production de graphes de dépendances bilexicales rentre sous le terme de *semantic dependency parsing* (SDP), dans la mesure où ces graphes sont en général voulus comme des représentations sémantiques superficielles. Mais les algorithmes présentés ici sont valables pour tout type de graphes bilexicaux, qu'il s'agisse des graphes syntaxiques profonds présentés au chapitre 6 ou des ressources des compétitions internationales *broad-coverage semantic parsing* (Oepen et al., 2014, 2015), où les arcs portent des étiquettes sémantiques.

Nous commençons par donner des repères bibliographiques concernant l'analyse en dépendances et l'analyse vers graphes de dépendances, puis nous détaillerons l'apprentissage multi-tâches défini pour capturer de l'interdépendance entre décisions de rattachement.

7.1 Contexte

Dans la littérature, la majorité des algorithmes de production de graphes bilexicaux (ci-après "parsing vers graphes") sont des adaptations d'algorithmes produisant des arbres de dépendances, adaptation levant simplement la contrainte qu'un noeud a exactement un noeud père (éventuellement la racine fictive). C'est pourquoi nous commencerons par résumer les avancées récente en analyse syntaxique en dépendances avant un rapide état de l'art sur l'analyse vers graphes de dépendances.

7.1.1 Avancées récentes en parsing en dépendances

Les avancées récentes en parsing en dépendances concernent surtout l'encodage de la séquence d'entrée plutôt que l'algorithme d'analyse lui-même. Nous passons en revue ces avancées, avec en particulier les encodages récurrents ou auto-attentionnel, et l'utilisation de l'apprentissage par transfert, permettant d'obtenir des vecteurs contextualisés pour chaque position de la séquence.

Encodage via réseau récurrent

Pour le parsing en dépendances, le travail de Chen and Manning (2014) constitue le premier analyseur neuronal utilisant des représentations distribuées de mots, de catégories et d'étiquettes de dépendances. Il s'agit d'un analyseur par transitions, dans lequel la représentation vectorielle d'une configuration comprend des vecteurs denses pour différentes positions de la pile et du buffer (comme par exemple les vecteurs de mots des 3 premiers mots de la pile), et différentes positions dans l'arbre partiellement construit (comme par exemple les vecteurs d'étiquettes de dépendances du dépendant le plus à gauche des deux premiers mots de la pile).

Andor et al. (2016) montrent la légère supériorité d'une modélisation globale de la probabilité d'une séquence de transitions. Au lieu de normaliser le score d'une décision locale en fonction de toutes les décisions locales possibles, la normalisation se fait au niveau de toute la séquence de décisions.

Mais, plus que de la modélisation probabiliste, les avancées en terme de performance proviennent surtout de la représentation des tokens d'entrée, de manière en capturer le contexte dès cette représentation. Kiperwasser and Goldberg (2016) ont proposé de rendre beaucoup plus générique la représentation d'entrée, en représentant les mots et leur contexte au moyen d'un encodage récurrent (biLSTM), sans traits supplémentaires.

Pour leur analyseur par transitions, la représentation vectorielle d'entrée d'une configuration devient simplement la concaténation de la représentation biLSTM de 4 positions clés (deux mots en haut de la pile, et deux mots du buffer). Pour leur analyseur MST (à la McDonald, cf. section 5.2.1), c'est la méthode pour scorer un arc (h, d) entre un gouverneur h et un dépendant d qui est modifiée : le score est ici obtenu en appliquant un MLP à la concaténation de la représentation biLSTM de h et de d. L'avancée majeure de ce travail est de proposer des traits génériques (réduits à l'encodage biLSTM de quelques positions de la phrase) en lieu et place des jeux de traits complexes mis au point habituellement manuellement.

Analyseur biaffine

Dozat and Manning (2017) s'appuient sur (Kiperwasser and Goldberg, 2016), avec les améliorations suivantes :

1. Ils spécialisent l'encodage biLSTM de chaque mot via des MLP spécifiques, pour distinguer d'une part le mot en tant que dépendant ou en tant que tête, et pour distinguer la prédiction de l'existence d'un arc entre deux mots i, j versus la prédiction de l'étiquette d'un arc donné. Cela donne quatre représentations distinctes pour le mot à la position i, en notant $\mathbf{r}_i$ sa représentation récurrente :

$$\begin{aligned}
\mathbf{h}_i^{(\text{edge-head})} &= \text{MLP}^{(\text{edge-head})}(\mathbf{r}_i) \\
\mathbf{h}_i^{(\text{label-head})} &= \text{MLP}^{(\text{label-head})}(\mathbf{r}_i) \\
\mathbf{h}_i^{(\text{edge-dep})} &= \text{MLP}^{(\text{edge-dep})}(\mathbf{r}_i) \\
\mathbf{h}_i^{(\text{label-dep})} &= \text{MLP}^{(\text{label-dep})}(\mathbf{r}_i)
\end{aligned}$$

2. Deuxièmement, ce qui a donné le nom à l'analyseur est que le score d'un arc $i \rightarrow j$ est obtenu par une transformation biaffine au lieu d'un MLP, appliqué aux représentations $\mathbf{h}_i^{(\text{edge-head})} \in \mathbb{R}^d$ et $\mathbf{h}_j^{(\text{edge-dep})} \in \mathbb{R}^d$, avec une matrice de paramètres $U \in \mathbb{R}^{d \times d}$:

$$\text{Biaff}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \mathbf{U} \mathbf{x}_2^\top + \mathbf{W}(\mathbf{x}_1 \oplus \mathbf{x}_2) + \mathbf{b}$$
$$s_{i \to j}^{(\text{edge})} = \text{Biaff}^{(\text{edge})}\left(\mathbf{h}_j^{(\text{edge-dep})}, \mathbf{h}_i^{(\text{edge-head})}\right)$$

Les scores des étiquettes sont également obtenus par transformation biaffine, avec une matrice $\mathbf{U}^{(l)}$ par étiquette l. À l'apprentissage, on utilise deux pertes de type entropie croisée, une pour maximiser la probabilité de la tête gold de chaque dépendant j et une pour maximiser la probabilité de l'étiquette gold, pour chaque arc gold.

À l'inférence, l'algorithme MST peut être utilisé pour récupérer l'arbre de score maximal, et pour chaque arc prédit, son étiquette de score maximal.

À noter que dans cette architecture, l'encodage de la position relative des mots est uniquement présent dans l'encodage récurrent biLSTM. Les arcs sont scorés séparément les uns des autres, et l'optimisation globale à la phrase est obtenue grâce à la contrainte que la sortie forme un arbre.

La popularité de l'algorithme tient à ses performances à la compétition internations CoNLL 2017, et à la simplicité de l'architecture, le calcul biaffine des scores des arcs et étiquettes étant facilement parallélisable pour un batch de phrases.

Encodage via auto-attention

L'encodage récurrent pose des difficultés pour les dépendances à longue distance : les informations provenant d'une position de la phrase peuvent s'estomper lors du processus de transmission récurrente de l'information. Par ailleurs du point de vue computationnel, l'encodage récurrent est difficile à paralléliser.

C'est ainsi que Vaswani et al. (2017) ont proposé un autre type d'encodage de séquences de mots et de mots en contexte. S'inspirant du mécanisme d'attention (Bahdanau et al., 2015) introduit dans l'architecture encodeur/décodeur en traduction automatique, ils ont proposé l'architecture à la base des avancées actuelles en TAL, le "Transformer". Il s'agit de remplacer totalement l'encodage récurrent du contexte par un empilement de couches d'auto-attention. À gros grain, dans une couche auto-attentive, un token de la séquence d'entrée est représenté par une somme pondérée, sur toutes les positions de la séquence, de représentations obtenues à la couche précédente. Les positions dans la séquence sont encodées sous forme de vecteurs, et non de manière algorithmique comme dans un LSTM. L'auto-attention est un mécanisme facilement parallélisable, qui permet d'apprendre les régularités liées aux positions au lieu de les encoder algorithmiquement.

Ce remplacement d'encodeur a été testé en analyse syntaxique également. Pour le parsing en constituants, Kitaev and Klein (2018) obtiennent même des résultats légèrement supérieurs en remplaçant l'encodage biLSTM par une auto-attention [1].

Du côté du parsing en dépendances, Li et al. (2019) montrent que les deux types d'encodage donnent des performances très similaires, dans le cadre d'un analyseur biaffine testé pour l'anglais et le chinois. En outre, une analyse des performances en fonction de la longueur des dépendances ne révèle pas de comportement très différents pour les deux types d'encodage : on ne retrouve pas spécialement de meilleures performances pour les dépendances longues avec

1. Kitaev and Klein (2018) montrent par ailleurs qu'il est bénéfique, lors du calcul des poids d'auto-attention, de séparer les embeddings positionnels et les embeddings de contenu lexical, au lieu de les sommer ou concaténer.

l'auto-attention plutôt que le biLSTM, ce qui contredit les intuitions souvent citées pour l'auto-attention. Enfin, un système ensemble, qui prédit les arcs en utilisant les sorties d'analyseur avec les 2 types d'encodage, ne donne que très peu d'améliorations, ce qui tend à confirmer que les deux types d'encodage se valent en termes d'informations capturées nécessaires à la tâche de parsing.

Modèles de vecteurs contextuels pré-entraînés

De la même manière que les vecteurs de mots (non contextuels) ont été massivement utilisés en mode pré-entraîné, on a l'équivalent pour les vecteurs contextuels : des améliorations importantes et régulières sur un grand nombre de tâches de TAL ont été obtenues en utilisant des modèles de représentations contextuelles de mots, dont les paramètres sont pré-entraînés sur (très) gros corpus, en utilisant principalement une tâche de type modèle de langue, i.e. la prédiction d'un mot en fonction de son contexte gauche, ou droit ou les deux. Comme pour cette tâche, des exemples peuvent être créés trivialement à partir de corpus brut, on parle d'apprentissage auto-supervisé.

Les modèles phares sont d'abord de type récurrent, avec le modèle ElMO (Peters et al., 2018), puis de type auto-attentif, avec le modèle BERT (Devlin et al., 2019).

Le modèle ElMO comprend plusieurs couches de LSTM dans le sens de gauche à droite et idem dans l'autre sens. La probabilité d'un mot sachant son contexte gauche (resp. droit) est simplement obtenue en appliquant une couche linéaire et un softmax vers tous les mots du vocabulaire. Le pré-entraînement cherche à maximiser la log-vraisemblance sur tous les mots du corpus d'apprentissage, en sommant la log-probabilité d'un mot sachant le contexte gauche et la log-probabilité du mot sachant le contexte droit.

Le modèle BERT abandonne l'encodage récurrent, en capitalisant sur des résultats obtenus en traduction automatique : la partie encodeur de Vaswani et al. (2017) est utilisée, en ignorant le décodeur et la tâche de traduction automatique. Le pré-entraînement est fait en utilisant tout d'abord la tâche de type "modèle de langue masqué" : les exemples d'apprentissage ne concernent que certaines positions aléatoirement masquées, qu'il s'agit de re-prédire. Le masquage permet d'obtenir un modèle véritablement bi-directionnel, où un mot masqué à prédire est représenté en utilisant toutes les positions, gauches et droites. L'autre tâche de pré-entraînement est la classification binaire de paires de séquences, selon que la deuxième séquence est la continuation de la première ou pas.

Le modèle fournit ainsi des représentations contextuelles de (sous-)mots[2], dont les paramètres peuvent être réutilisés, en mode figé ou pas, comme couches basses de réseaux pour tout autre tâche de TAL. On parle d'apprentissage par transfert, avec un apprentissage auto-supervisé très lourd réalisé sur une ou plusieurs tâches très génériques, spécialisable rapidement pour des tâches de TAL plus spécifiques et comportant relativement peu de données d'apprentissage.

Pour ce qui est du parsing, comme pour la plupart des tâches de TAL, ces modèles contextuels ont apporté des nettes améliorations, dans diverses langues, que ce soit pour l'analyse en constituants avec par exemple l'utilisation d'un modèle pré-entraîné pour 10 langues (Kitaev et al., 2019) ou l'analyse en dépendances (Li et al., 2019) (testé pour l'anglais et le chinois).

2. Pour plus de robustesse face aux mots inconnus, les mots sont segmentés en sous-mots d'après un vocabulaire de sous-mots défini sur critère probabiliste ou fréquentiel.

Pour le français, les modèles FlauBERT (Le et al., 2020) et CamemBERT (Martin et al., 2020) donnent des gains de performance en parsing en dépendances, pour des parsers biaffines à la (Dozat and Manning, 2017). Avec FlauBERT dans un parser de ce type, Grobol and Crabbé (2021) rapportent que l'encodage lexical peut se réduire aux vecteurs contextuels plus plongements lexicaux appris. Ni une représentation récurrente au niveau des caractères, ni les plongements de lemmes et de catégories morpho-syntaxiques ne sont finalement nécessaires.

Analyseurs par transitions basés sur les "pointer networks"

Vinyals et al. (2015) ont proposé les "pointer networks", une architecture de type séquence vers séquence pour le cas où le vocabulaire de sortie est directement celui de la séquence d'entrée : dans la partie décodage, ils utilisent directement le vecteur d'attention sur les n positions d'entrée pour modéliser la probabilité de générer une position de sortie. Le nom "pointer network" vient de ce que l'élément de plus fort score est celui "pointé" dans la séquence d'entrée, mais on parle plutôt maintenant d'auto-attention.

Dans des analyseurs par transitions, ce principe a été utilisé pour pointer, à partir d'un mot x dans la phrase, vers un des dépendants de x (Ma et al., 2018), ou bien vers le gouverneur de x (Fernández-González and Gómez-Rodríguez, 2019). Dans les deux cas on a une sorte d'analyseur par transitions avec un parcours unidirectionnel de la phrase, et avec un jeu d'actions simplifié, du type "attacher le mot pointé comme dépendant du mot courant" (ou vice-versa). En comparaison au parser biaffine où tous les scores des arcs sont calculés en même temps, avec le même état des paramètres, on conserve un parcours unidirectionnel, avec le désavantage classique de la propagation d'erreurs. L'avantage est cependant de pouvoir injecter, dans la représentation de la configuration courante de l'analyseur, des informations sur les dépendances précédemment prédites. Et effectivement, malgré sa simplicité, et l'absence d'optimisation globale, l'analyseur de Fernández-González and Gómez-Rodríguez (2019) a obtenu le nouvel état de l'art de l'époque sur le Penn Tree bank converti en Stanford dependencies (en utilisant cependant un faisceau de taille 5, contrebalançant la propagation d'erreurs).

7.1.2 État de l'art du parsing vers graphes bilexicaux

Les analyseurs produisant directement des graphes bilexicaux et non plus des arbres sont en général des adaptations d'algorithme de parsing en dépendances, où l'on lève la contrainte sur le nombre de gouverneurs d'un dépendant. Le terme émergent dans la communauté est celui de "parsing en dépendances sémantiques" ("semantic dependency parsing"), qui insiste sur le fait qu'en passant aux graphes bilexicaux, on cherche en général à capturer des dépendances d'ordre sémantique.

Les données pertinentes peuvent être aussi bien des graphes avec étiquettes sémantiques (de type ARG0, ARG1 ...) que des graphes syntaxiques profonds, tels que ceux présentés chapitre 6. Les données majoritairement utilisées dans la littérature pour cette tâche sont les graphes bilexicaux de la compétition internationale "broad-coverage semantic parsing" (Oepen et al., 2014, 2015) (ci-après BC-SDP-SemEval-2014), dérivés d'analyses syntaxico-sémantiques riches, pour l'anglais, le tchèque et le chinois.

La technique du parsing par transitions peut facilement être utilisée pour traiter des graphes orientés, ou en tous cas des graphes orientés acycliques (DAG), en adaptant les jeux de transitions pour qu'un mot puisse avoir plusieurs gouverneurs. Selon les cas, le jeu de transitions

ne permet que de proposer des graphes planaires[3] (Sagae and Tsujii, 2008) ou bien traite certains cas de non planarité, grâce à une opération inversant la position des deux premiers mots de la pile (Titov et al., 2009).

Martins and Almeida (2014) adaptent au cas des graphes le Turbo parser (Martins et al., 2013), un parser d'ordre 3 (i.e. définissant le score d'un arbre comme somme de scores de sous-parties comprenant jusqu'à 3 arcs), et utilisant l'algorithme approché AD^3 à l'inférence. Ils obtiennent ainsi les meilleurs résultats pour la compétition BC-SDP-SemEval-2014.

Dozat and Manning (2018) montrent que des modifications mineures de leur analyseur (Dozat and Manning, 2017) donnent de très bons résultats pour produire les graphes bilexicaux des données BC-SDP-SemEval-2014. À l'apprentissage, tous les arcs possibles sont utilisés comme exemples d'apprentissage, positifs ou négatifs selon qu'ils sont présents ou pas dans le graphe gold, et une perte de type entropie croisée binaire est utilisée. L'algorithme d'inférence est trivial, il n'y a pas de parcours de la séquence, pas de transitions : tous les arcs de probabilité > 0.5 sont considérés comme prédits, et il n'y a pas de limitation formelle sur les graphes de sortie. Toute la charge repose sur le processus d'apprentissage pour capturer les contraintes présentes dans l'ensemble d'apprentissage.

Autre exemple d'adaptation du parsing vers arbres au parsing vers graphe, Fernández-González and Gómez-Rodríguez (2020) adaptent leur parser en dépendances (Fernández-González and Gómez-Rodríguez, 2019) (cf. section 7.1.1), de telle sorte que le nombre de gouverneurs du mot focus ne soit pas limité à un. L'analyse procède de gauche à droite, en déplaçant le focus sur les différents mots de la phrase. Pour le mot focus d, le pointer network est utilisé pour générer un rang de mot h, qui sera interprété comme son gouverneur. Contrairement à la version générant des arbres, tant que le rang h généré ne vaut pas d, le parser continue d'ajouter des gouverneurs à d. Les arcs préalablement ajoutés sont utilisés partiellement dans la décision d'ajouter un gouverneur : on additionne à la représentation du mot courant celle du dernier gouverneur déjà ajouté pour ce mot.

Wang et al. (2019) obtiennent les meilleurs résultats actuels sur les données anglaises et chinoises de la compétition 2015 (Oepen et al., 2015), avec un analyseur d'ordre deux. Les scores des arcs (respectivement des paires d'arcs) sont obtenus par transformation biaffine (respectivement triaffine). L'inférence se fait par un algorithme itératif approché (inférence variationnelle à champ moyen ou propagation de croyance par boucle), en $O(n^3)$.

Apport d'informations syntaxiques

Enfin, divers travaux ont montré l'apport d'informations syntaxiques pour la prédiction de graphes de dépendances. Par exemple, Ribeyre et al. (2015) montrent qu'intégrer au Turbo Semantic Parser (Martins and Almeida, 2014) des traits issus d'arbres de dépendances prédits améliore les résultats. Cette tendance est confirmée également pour les graphes syntaxiques profonds du français (chapitre 6) (Ribeyre et al., 2016).

Plus récemment, Bernard (2021) propose le système MTI (pour *multi task integration*), qui réalise les tâches de tagging, analyse vers arbres de dépendances et analyse vers graphes de dépendances sémantiques, de manière plus intégrée que dans une approche de type apprentissage multi-tâche. Les trois tâches sont abordées via un unique ensemble d'actions réalisables sur un token (assigner une catégorie morpho-syntaxique, assigner un gouverneur syntaxique ou un gouverneur sémantique). L'analyse est itérative mais ne procède pas dans un ordre

3. I.e. informellement, un graphe où, quand les arcs sont dessinés dans le demi-plan au-dessus de la phrase, aucun arc n'en croise un autre.

linéaire. À chaque itération, pour chaque token l'action de meilleur score est choisie, en écrasant éventuellement des actions passées. De manière cruciale, la représentation vectorielle de la configuration courante intègre les tags et arcs prédits à ce stade. La propagation d'erreurs qui peut en résulter est limitée par la possibilité pour l'analyseur de corriger des actions passées.

7.2 Tâches auxiliaires pour réduire la localité des décisions

Je décris ici un travail en cours sur le parsing vers graphes de dépendances, où j'investigue l'utilisation de tâches auxiliaires pour aider le parsing vers graphes de type biaffine.

Dans la section précédente, nous avons vu différentes propositions pour capturer une interdépendance des décisions de création des arcs du graphe.

Les analyseurs les plus performants sont ceux qui scorent les graphes en utilisant des facteurs d'ordre supérieur à un (des paires ou triplets d'arcs (Martins and Almeida, 2014 ; Wang et al., 2019)). Cette utilisation d'un contexte moins local se fait au prix d'une complexité accrue ($O(n^3)$).

Une autre façon d'intégrer une interdépendance des arcs est de procéder à une analyse séquentielle, et d'intégrer à chaque étape une représentation du graphe partiel prédit jusque là (cf. les analyseurs par transitions cités (Fernández-González and Gómez-Rodríguez, 2020), ou le système itératif MTI (Bernard, 2021)).

Au contraire, le parser biaffine a pour caractéristique que les scores de tous les arcs possibles sont calculés d'un coup, en $O(n^2)$, de manière facilement parallélisable. L'analyseur considère chaque arc isolément, et l'analyse n'étant ni séquentielle (dans l'ordre linéaire de la phrase), ni itérative, la représentation vectorielle d'un candidat arc n'intègre aucune information sur les arcs déjà prédits.

Dans le cas du parsing vers arbres de dépendances (Dozat and Manning, 2017), cette localité est contrebalancée par la contrainte d'avoir un arbre en sortie : l'algorithme MST fournit l'arbre optimal. Mais la prédiction de graphes ne peut pas s'appuyer sur cette contrainte. Dans (Dozat and Manning, 2018), un arc est simplement prédit pour peu que sa probabilité dépasse 0.5, et les arcs sont ainsi prédits totalement indépendamment les uns des autres.

Dans cette section, nous investiguons l'utilisation de tâches auxiliaires pour ramener un peu d'interdépendance dans la prédiction du score de chaque arc.

7.2.1 Le parser biaffine vers graphes de dépendances

On redéfinit ici précisément le calcul biaffine des scores des arcs et étiquettes du modèle défini dans (Dozat and Manning, 2018)[4], modernisé en utilisant des vecteurs contextuels : pour le parsing d'une séquence de mots $w_{1:n}$, on applique à la séquence un modèle de langue pré-entraîné, noté ici BERT. Ainsi $\text{BERT}(w_{1:n})$ désigne les n vecteurs contextuels[5]. On encode le mot à la position i avec un plongement lexical non-contextuel concaténé au vecteur

4. Dozat and Manning (2018) testent également un modèle avec un unique score pour un arc étiqueté, en intégrant une étiquette spéciale pour signifier l'absence d'arcs, ce qui ne donne pas de différences significatives.

5. Dans toutes nos expériences, on prend le vecteur contextuel, à la dernière couche, du premier sous-mot du mot.

contextuel[6]. Le tout est ensuite encodé par un certain nombre de couches biLSTM :

$$\begin{aligned}\mathbf{plm}_i &= (\text{BERT}(w_{1:n}))_i \\ \mathbf{v}_i &= \mathbf{e}_i^{(\text{word})} \oplus \mathbf{plm}_i \\ \mathbf{r}_{1:n} &= \text{biLSTM}(\mathbf{v}_{1:n})\end{aligned}$$

La représentation récurrente $\mathbf{r}_i$ du mot w_i est spécialisée selon deux caractéristiques binaires : mot en tant que gouverneur versus dépendant, et score des arcs versus score des étiquettes des arcs :

$$\begin{aligned}\mathbf{h}_i^{(\text{edge-head})} &= \text{MLP}^{(\text{edge-head})}(\mathbf{r}_i) \\ \mathbf{h}_i^{(\text{label-head})} &= \text{MLP}^{(\text{label-head})}(\mathbf{r}_i) \\ \mathbf{h}_i^{(\text{edge-dep})} &= \text{MLP}^{(\text{edge-dep})}(\mathbf{r}_i) \\ \mathbf{h}_i^{(\text{label-dep})} &= \text{MLP}^{(\text{label-dep})}(\mathbf{r}_i)\end{aligned}$$

Le score d'un arc est défini par une transformation biaffine simplifiée :

$$\begin{aligned}&\text{Biaff}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{U}, \mathbf{b}) = \mathbf{x}_1 \mathbf{U} \mathbf{x}_2^\top + \mathbf{b} \\ &s_{i\to j}^{(\text{edge})} = \text{Biaff}\left(\mathbf{h}_j^{(\text{edge-dep})}, \mathbf{h}_i^{(\text{edge-head})}, \mathbf{U}^{(\text{edge})}, \mathbf{b}^{(\text{edge})}\right)\end{aligned}$$

Pour les scores des étiquettes, on utilise une transformation biaffine par étiquette :

$$s_{i\to j}^{(\text{l})} = \text{Biaff}\left(\mathbf{h}_j^{(\text{label-dep})}, \mathbf{h}_i^{(\text{label-head})}, \mathbf{U}^{(\text{l})}, \mathbf{b}^{(\text{l})}\right)$$

Apprentissage

Soit une phrase d'apprentissage $w_{1:n}$ et son graphe gold Y. En notant collectivement θ les paramètres du réseau, on considère une v.a. binaire EDGE pour l'existence d'un arc. La probabilité d'existence d'un arc peut s'obtenir en appliquant sigmoïde à son score, et la probabilité d'une étiquette en appliquant softmax sur les scores de toutes les étiquettes possibles :

$$\begin{aligned}&P(\text{EDGE} = 1 | i, j, w_{1:n}, \theta) = \sigma(s_{i\to j}^{(\text{edge})}) \\ &P(\text{LABEL} = l | i, j, w_{1:n}, \theta) = (\text{softmax}(s_{i\to j}^{(\text{l}_1)}, s_{i\to j}^{(\text{l}_2)}, ..., s_{i\to j}^{(\text{l}_{|L|})}))_l\end{aligned}$$

On utilise une perte de type entropie croisée binaire pour l'existence ou non-existence d'un arc entre toute paire de positions i, j, et une perte de type entropie croisée pour l'étiquette des arcs effectivement existants dans le graphe gold. Ainsi, si on note δ_{ij} un booléen pour l'existence d'un arc $i \to j$ dans l'arbre gold Y, les deux types de perte s'écrivent :

$$\begin{aligned}&\mathcal{L}^{(\text{edge})}(w_{1:n}, Y, \theta) = \textstyle\sum_{i=1}^{n} \sum_{j=1}^{n} \delta_{ij}(-\ln(\sigma(s_{i\to j}^{(\text{edge})}))) + (1-\delta_{ij})(-\ln(\sigma(-s_{i\to j}^{(\text{edge})}))) \\ &\mathcal{L}^{(\text{label})}(w_{1:n}, Y, \theta) = \textstyle\sum_{i \xrightarrow{l} j \in Y} -\ln(P(\text{LABEL} = l | i, j, w_{1:n}, \theta))\end{aligned}$$

6. Dozat and Manning (2018) utilisent des plongements de lemmes et de catégories morpho-syntaxiques, mais à l'époque, pas de modèle de langue pré-entraîné. Quelques expérimentations nous ont montré que ces plongements devenaient superflus dès lors qu'on utilise les vecteurs contextuels pré-entraînés.

Inférence

Le graphe prédit contient simplement tous les arcs de probabilité sigmoïde > 0.5, i.e. de score positif. Et pour chaque arc prédit, l'étiquette prédite est celle de score maximal pour cet arc :

$$\hat{Y} = \{i \xrightarrow{\hat{l}} j \mid s^{(\text{edge})}_{i \to j} > 0 \text{ et } \hat{l} = \arg\max_{l \in L} s^{(\text{l})}_{i \to j}\}$$

7.2.2 Ajout de tâches auxiliaires

Des expérimentations préliminaires sur les graphes syntaxiques profonds présentés au chapitre 6 nous ont montré que l'analyse biaffine vers graphes donne de bons résultats, mais avec des incohérences clairement liées à la prise de décision locale à chaque arc. En particulier, une rapide analyse d'erreurs faisait apparaître des combinaisons d'arcs incompatibles, plus précisément des ensembles d'étiquettes impossibles. C'est le cas par exemple d'une ponctuation rattachée à deux gouverneurs avec l'étiquette `punct`, ou d'un token rattaché à la fois en tant que composant de mot composé (étiquette `dep_cpd`) et rattaché à un autre gouverneur comme dépendant autonome.

D'où l'idée d'utiliser des tâches auxiliaires prenant en compte l'ensemble des gouverneurs d'un dépendant donné.

On propose un apprentissage multi-tâche sur les tâches suivantes :
- **tâches A et L** : les deux tâches cibles, i.e. prédiction des arcs, et pour un arc, prédiction de son étiquette ;
- **tâche H** : nombre de gouverneurs de chaque token ;
- **tâche D** : nombre de dépendants de chaque token ;
- prédiction de la combinaison d'étiquettes des arcs entrants de chaque token, sous deux formes :
  - **tâche S** : sous forme de symboles atomiques : on considère comme une catégorie atomique la concaténation des étiquettes des arcs entrants, ordonnées par ordre alphabétique (par exemple `obj+suj+suj`)
  - **tâche B** : sous forme d'un "bag of labels" : on considère un vecteur creux dont la taille est le nombre d'étiquettes $|L|$, pour un mot w_j la valeur à la composante l est le nombre d'arcs entrants vers j ayant l'étiquette l.

Techniquement, on utilise des MLP spécifiques pour les tâches de régression (nombre de gouverneurs, nombre de dépendants) et pour les tâches de classification. Ceux-ci sont appliqués sur la représentation récurrente $\mathbf{r}_i$ de chaque token i.

Prédiction du nombre de gouverneurs et nombre de dépendants

On fait une tâche de régression pour prédire le nombre de gouverneurs (respectivement du nombre de dépendants), avec dans les deux cas un MLP à un seul neurone en sortie :

$$nbh_j = \text{MLP}^{(\text{H})}(\mathbf{r}_j)$$
$$nbd_i = \text{MLP}^{(\text{D})}(\mathbf{r}_i)$$

Plus précisément nbh_j est interprété comme le log de $1+$ le nombre de gouverneurs de j (idem pour nbd_j). Les logs sont utilisés pour lisser les nombres entiers, et pénaliser moins les erreurs sur les nombres plus importants. La perte est définie comme la somme, sur tous les tokens des phrases du batch, des différences au carré entre les logs des nombres de gouverneurs gold et prédit (resp. nombres de dépendants).

Tâches sur l'ensemble des étiquettes des arcs entrants

Nous avons investigué deux modélisations pour la prédiction de l'ensemble des étiquettes : sous forme de symbole atomique (tâche S) ou sous forme de vecteur creux (tâche B).

Pour la tâche S, pour chaque token d'un exemple d'apprentissage, on trie par ordre alphabétique les étiquettes de ses arcs entrants (ci-après une "séquence d'étiquettes"). On fait une classification vers le vocabulaire des séquences d'étiquettes ainsi obtenu[7]. Le vecteur de scores des séquences d'étiquettes est obtenu par :

$$\mathbf{s}_j = \mathrm{MLP}^{(\mathsf{S})}(\mathbf{r}_j)$$

Pour la tâche B, on cherche à prédire le vecteur "bag of labels" (BOL) de chaque dépendant, dont la taille est le nombre d'étiquettes atomiques distinctes $|L|$. On utilise un MLP avec une couche de sortie de taille $|L|$:

$$\mathbf{BOL}_j = \mathrm{MLP}^{(\mathsf{B})}(\mathbf{r}_j)$$

Lors du calcul de la perte, pour le dépendant j, on cherche à ce que la l-ième composante du vecteur $\mathbf{BOL}_j$ vaille le (log de $1+$ le) nombre gold d'arcs entrants vers j portant l'étiquette simple numéro l. Pour cela on utilise comme perte, pour chaque dépendant j, la distance L2 entre le vecteur BOL gold et celui prédit.

Apprentissage multi-tâche et propagation en pile

Pour l'intégration multi-tâches, on teste deux configurations : premièrement le simple partage de paramètres jusqu'aux couches de biLSTM incluses (et donc les paramètres du modèle de langue pré-entraîné) : comme vu supra, tous les MLPs de spécialisation s'appliquent sur les représentations récurrentes des tokens.

Deuxièmement, on teste la technique de "*stack propagation*" définie par Zhang and Weiss (2016), et qu'ils expérimentent pour les tâches de tagging et parsing en dépendances. Dans notre cas, il s'agit d'utiliser les couches denses des tâches auxiliaires au sein des vecteurs d'entrée des transformations biaffines utilisées pour scorer les arcs et leurs étiquettes.

Ainsi par exemple pour utiliser la tâche H en mode "stack propagation", le score d'un arc est calculé en concaténant à la représentation du dépendant $\mathbf{h}_j^{(\text{edge-dep})}$, la couche cachée de $\mathrm{MLP}^{(\mathsf{H})}$ pour le dépendant j : $\text{hidden}_j^{(\mathsf{H})}$. On utilise en outre un hyperparamètre réel $c^{(\mathsf{H})}$ comme coefficient multiplicateur, ce qui donne pour le score d'un arc :

$$s_{i\to j}^{(\text{edge})} = \mathrm{Biaff}\left(\mathbf{h}_j^{(\text{edge-dep})} \oplus (c^{(\mathsf{H})}\text{hidden}_j^{(\mathsf{H})}), \mathbf{h}_i^{(\text{edge-head})}, \mathbf{U}^{(\text{edge})}, \mathbf{b}^{(\text{edge})}\right)$$

À noter que la matrice $\mathbf{U}^{(\text{edge})}$ n'est plus carrée dans ce cas.

7. On utilise l'ordre alphabétique et pas l'ordre linéaire des gouverneurs pour limiter le nombre de séquences. Cela neutralise les variations d'ordre.

De même on utilise la tâche B pour modifier le calcul du score de chaque étiquette l :

$$s_{i\to j}^{(\mathsf{l})} = \text{Biaff}\left(\mathbf{h}_j^{(\text{label-dep})} \oplus (c^{(\mathsf{B})}\text{hidden}_j^{(\mathsf{B})}), \mathbf{h}_i^{(\text{label-head})}, \mathbf{U}^{(\mathsf{l})}, \mathbf{b}^{(\mathsf{l})}\right)$$

À noter que l'utilisation de ces représentations auxiliaires pour les tâches principales oblige à réaliser les tâches auxiliaires lors de la phase de prédiction, et pas uniquement à l'apprentissage.

Paramètres appris pour pondération des sous-pertes

Dans les deux cas, pour chaque batch, on cherche à minimiser une somme pondérée des pertes de chaque tâche, pour les tâches principales A et L, et selon l'hyperparamétrage, pour certaines des tâches H, D, S et B. Plutôt que d'utiliser des hyperparamètres fixés manuellement pour les poids des pertes de chaque tâche, on utilise la notion d'incertitude des tâches, et l'approximation proposée par Kendall et al. (2018), qui introduit un paramètre σ_t pour chaque sous-tâche t, à interpréter comme le "bruit" ou l'"incertitude" de chaque tâche. En notant T l'ensemble des tâches, la perte globale pour un batch s'écrit :

$$L = \sum_{t\in T} \frac{1}{\sigma_t^2} L_t + \ln(\sigma_t)$$

Les paramètres σ_t sont initialisés à 1 et modifiés au cours de l'apprentissage. Le premier terme de la somme fait que plus la tâche est incertaine, moins sa perte comptera, alors que le deuxième terme empêche d'augmenter arbitrairement l'incertitude.

7.2.3 Expériences et résultats

Protocole expérimental

On expérimente sur les graphes syntaxiques profonds des phrases du FTB (Ribeyre et al., 2014), obtenus par conversion par règles OGRE appliquées aux arbres gold (avec préannotation manuelle de certains phénomènes pour améliorer la qualité des graphes produits, cf. chapitre 6, section 6.3.3)[8].

Pour les vecteurs contextuels, on utilise le modèle `flaubert-base-cased` (Le et al., 2020), et la bibliothèque transformer de HuggingFace (Wolf et al., 2020).

Après quelques essais, on fixe la configuration suivante pour investiguer l'impact des tâches auxiliaires :

— Le modèle FlauBERT est modifié à l'apprentissage et non pas figé.
— Les plongements lexicaux sont de taille 100, et sont initialisés au hasard et non pas pré-entraînés.
— On utilise un "lexical dropout" de 0.4 : à l'apprentissage, le plongement lexical d'un token est remplacé avec une probabilité 0.4 par un token spécial *DROP*, dont l'embedding est appris.
— On utilise 3 couches de biLSTM, chaque couche de taille 2*600, avec un dropout de 0.33.

8. Dans ces graphes profonds, certains mots de la phrase n'ont aucun arc entrant (comme les auxiliaires, les complémenteurs, certaines prépositions...). Un mot est considéré comme la racine, et a en pratique comme gouverneur un faux token racine. Ainsi pour ces données, dans toutes les modélisations supra, la séquence $w_{1:n}$ correspond à une phrase de $n-1$ tokens, avec un faux token racine w_1. Pour ce dernier, on utilise comme vecteur contextuel celui du token de début de séquence de FlauBERT.

- Tous les MLP de spécialisation cités supra ont une seule couche cachée, avec activation ReLU.
  - Parmi ceux-ci, les MLP pour les tâches auxiliaires ($\text{MLP}^{(\text{H})}$, $\text{MLP}^{(\text{D})}$, $\text{MLP}^{(\text{S})}$, $\text{MLP}^{(\text{B})}$) ont une couche cachée de taille 300 et un dropout de 0.25.
  - Pour $\text{MLP}^{(\text{edge})}$ et les $\text{MLP}^{(\text{l})}$ pour toutes les étiquettes l, la couche cachée et la couche de sortie ont une taille de 600, et un dropout de 0.33.
- L'optimiseur est Adam, avec $\beta_1 = \beta_2 = 0.9$, et un pas d'apprentissage initial de 0.00002, et des batchs de 8.
- L'apprentissage est fait pendant au plus 30 époques, avec un arrêt précoce dès lors que la performance sur l'ensemble de développement baisse.
  - La métrique d'évaluation est le Fscore sur les arcs étiquetés (FL).

Différents essais ont été abandonnés car infructueux dans nos tests préliminaires :

- L'utilisation de plongements lexicaux pré-entraînés, de plongements de lemmes et de catégories morpho-syntaxiques n'a pas d'impact sur les performances en moyenne.
- Geler les paramètres de FlauBERT fait baisser nettement les performances (de l'ordre de 2 points de FL).
- Augmenter le niveau de partage de paramètres entre tâches n'a pas été fructueux : au lieu d'appliquer les MLP des tâches auxiliaires sur les représentations récurrentes $\mathbf{r}_i$, nous avons testé de les appliquer sur les sorties des MLP de spécialisation (i.e. sur $\mathbf{h}_i^{(\text{edge-head})}$ pour tâche D, sur $\mathbf{h}_i^{(\text{edge-dep})}$ pour tâche H, sur $\mathbf{h}_i^{(\text{label-dep})}$ pour tâches B et S). Mais ceci n'améliore pas les performances, et cela a tendance à augmenter le nombre d'époques nécessaires à l'apprentissage.

Avec la configuration de base définie supra, on fait varier la combinaison de tâches et l'utilisation ou pas de la propagation en pile. Pour chaque configuration complète, on lance 9 apprentissages.

Résultats sans propagation en pile

On donne Table 7.1 les résultats pour le système de base, i.e. sans tâche auxiliaire, et pour les modèles entraînés avec diverses combinaisons de tâches auxiliaires, sans propagation en pile. Tout d'abord on constate l'amélioration obtenue par rapport à des architectures plus anciennes : pour cette architecture biaffine avec vecteurs contextuels, sans tâches auxiliaires on obtient un FL moyen de 86,79. Avec un système non neuronal, Ribeyre et al. (2016) obtenaient FL=80,86, et montaient à FL=84,91 grâce à des traits issus de parses en constituants, obtenus par l'analyseur riche FrMG (Villemonte De La Clergerie, 2010).

Ensuite, on constate qu'il y a des gains modestes en ajoutant les tâches auxiliaires. À noter cependant une certaine instabilité, et l'écart-type sur les 9 lancers est parfois du même ordre de grandeur que les gains.

Globalement, on obtient le meilleur lancer individuel avec toutes les tâches BDHS (FLmax = 87,7), mais on peut remarquer une variance assez importante, et donc il convient d'étudier les résultats moyens.

Parmi les configurations avec une seule tâche auxiliaire, c'est la tâche B qui a le meilleur résultat moyenné (87,05), mais aucune des 4 tâches ne donne de différence significative par

Tâches auxiliaires	Sur 9 lancers		
	FLmax	FLmoyen	écart-type
aucune	86,95	86,79	0,19
H	87,64	86,82	0,54
D	87,14	86,83	0,40
S	87,21	86,98	0,30
B	87,60	87,05	0,49
B H	87,48	87,32***	0,18
D H	87,27	86,61	0,71
H S	87,24	87,04	0,17
D H S	87,32	86,86	0,39
B H S	87,55	87,14**	0,39
B D H S	**87,70**	**87,35*****	0,26

TABLE 7.1 – Résultats sur l'ensemble dev, sans propagation en pile, pour diverses combinaisons de tâches auxiliaires, chacune répétée 9 fois (H : nb de gouverneurs, D : nb de dépendants, B : "bag of labels", S : séquence d'étiquettes). Col2-4 : Fscore étiqueté maximum, moyenne des Fscores étiquetés, écart-type. *** : moyenne significativement plus haute que la version sans tâche auxiliaire (α=0.001). ** : idem, avec α=0.01.

rapport à la version sans tâche auxiliaire [9].

En regardant les combinaisons d'une ou plusieurs tâches, les combinaisons avec une moyenne significativement plus haute que la version de base sont BH et BDHS, et dans une moindre mesure BHS. On retient pour la suite comme meilleure configuration sans propagation en pile, la combinaison la plus simple, BH, avec en moyenne un gain de +0,53 point de Fscore.

Résultats avec propagation en pile

On teste maintenant l'utilisation de la propagation en pile (où les couches denses des MLP des tâches auxiliaires sont utilisées en entrée des transformations biaffines pour les scores des arcs et des étiquettes). On donne les résultats Table 7.2, pour la combinaison de tâches BH. Il semble y avoir un petit incrément statistiquement significatif avec les poids $c^{(B)}$=1 et $c^{(H)}$ =10, qui donnent un FL moyen de 87,66 (à comparer avec FLmoyen=87,32 pour la combinaison BH sans propagation en pile). Cela dit le gain est faible, et est à mettre en balance avec le fait que la propagation en pile complexifie l'inférence, cf. elle oblige à l'inférence à prédire les tâches auxiliaires.

9. On évalue la significativité des différences de performance entre deux configurations, en utilisant un test exact de permutation de Fisher-Pitman (comme le fait par exemple Bernard (2021)). Plus précisément, supposons que l'on considère deux échantillons de Fscores, pour nA lancers correspondant à la configuration A, et nB lancers pour la configuration B, avec en moyenne la configuration B meilleure que A. L'hypothèse nulle est que les deux échantillons suivent la même distribution. La p-value estime la probabilité que séparer l'ensemble des Fscores en deux échantillons A' et B' de taille nA et nB donne une différence de moyenne au moins aussi grande que celle observée. Avec le test exact, la p-value est calculée de manière exacte, sur toutes les divisions en échantillons A' et B' possibles.

Poids pour propagation en pile	Sur 9 lancers		
	LF max	LF moyen	écart-type
$c^{(B)}$=1 $c^{(H)}$=1	87,54	87,49	0,06
$c^{(B)}$=5 $c^{(H)}$=5	87,60	87,37	0,33
$c^{(B)}$=1 $c^{(H)}$=10	**87,82**	**87,66*****	0,18
$c^{(B)}$=1 $c^{(H)}$=20	87,66	87,45	0,34
B H sans propag. en pile	87,48	87,32	0,18

TABLE 7.2 – Résultats sur l'ensemble dev, avec tâches B et H, avec propagation en pile et divers poids pour les couches denses (cf. hyperparamètres $c^{(B)}$, $c^{(H)}$ section 7.2.2). Col2-4 : Fscore étiqueté maximum, moyenne des Fscores étiquetés, écart-type. *** Moyenne significativement plus haute (α=0.001) par rapport à la moyenne sans propagation en pile (donnée à la dernière ligne).

Résultats sur test des meilleures configurations

Pour évaluer sur l'ensemble de test l'ajout de tâches auxiliaires, on sélectionne la configuration de base, la meilleure configuration sans propagation en pile (BH), et la meilleure avec propagation en pile (BH $c^{(B)}$=1 $c^{(H)}$=10). On constate que les configurations avec tâches auxiliaires (avec et sans propagation en pile) ont une moyenne significativement plus haute que la version sans tâche auxiliaire (α=0.001). Par contre l'écart avec et sans propagation en pile est réduit, et non significatif.

Configuration	Sur 9 lancers		
	FLmax	FLmoyen	écart-type
aucune tâche auxiliaire	87,16	86,76	0,34
meilleure sans propagation (BH)	87,75	87,42***	0,38
meilleure avec propagation (BH $c^{(B)}$=1 $c^{(H)}$)	87,92	87,66***	0,19

TABLE 7.3 – Résultats sur l'ensemble de test, pour le meilleur modèle (d'après résultats sur dev) dans trois configurations : sans tâche auxiliaire (ligne 1), avec tâches auxiliaires sans propagation (ligne 2), avec tâches auxiliaires et propagation en pile (ligne 3). Col2-4 : Fscore étiqueté maximum, moyenne des Fscores étiquetés, écart-type.

Utilisation des sous-tâches à l'inférence

Le partage de paramètres avec les tâches auxiliaires est sensé favoriser une meilleure prédiction des arcs et des étiquettes d'arcs. On rappelle que l'inférence de base est de considérer comme prédits les arcs de score positif (ce qui va de pair avec l'utilisation de la perte entropie croisée binaire à l'apprentissage). Donc par exemple, on peut espérer qu'avec la tâche H (nombre de gouverneurs), pour un dépendant ayant de nombreux gouverneurs gold, la représentation partagée du dépendant favorisera des scores positifs. De la même manière, pour la tâche B (bag of labels), on peut imaginer que la représentation du dépendant va favoriser les étiquettes d'arcs ayant les bons labels.

L'influence des tâches auxiliaires est donc a priori très indirecte.

Pour évaluer si les tâches auxiliaires pourraient être mieux mises à profit, nous pouvons comparer la performance directe obtenue sur ces tâches versus leur performance "indirecte", i.e. telle que calculée a posteriori dans les graphes de dépendance prédits. Par exemple évaluer la prédiction directe du nombre de gouverneurs versus évaluer les nombres de gouverneurs comptés dans les graphes prédits. En outre, on constate que les performances "directes" pour ces tâches auxiliaires : on peut les comparer aux précisions obtenues pour ces tâches, telles que calculées dans les graphes prédits. Les tendances générales que nous avons observées sont que pour la tâche B, les performances directes sont du même niveau que celles calculées dans les graphes prédits (autour de 87-88). Dans les configurations où on utilise la tâche S, équivalent logique de B, la performance directe pour S peut monter autour de 90. Pour la tâche D, les performances directes sont plutôt moins bonnes (autour de 86) que dans les graphes prédits (autour de 90) : il est difficile de prédire directement le nombre de dépendants d'un token. Il est plus facile de prédire le nombre de gouverneurs, qui varie moins. En outre, on constate que la prédiction directe des nombres de gouverneurs reste meilleure que les nombres de gouverneurs calculés a posteriori.

Plus précisément, on donne Table 7.4 la précision sur les tâches auxiliaires, pour les trois configurations sélectionnées sur dev, les précisions étant obtenues soit directement soit indirectement. Pour la meilleure configuration (dernière ligne), on voit que la tâche H permet de prédire le nombre de gouverneurs avec une précision de 96,2. Si l'on compare cela au nombre de gouverneurs tels que calculé dans les graphes prédits, il est de 92,2 dans la configuration de base, et de 93,9 dans la meilleure configuration.

Donc d'une part, la prédiction indirecte du nombre de gouverneurs n'est pas très bonne avec le système de base (tâche H indirecte : 92,2, alors qu'on obtient +4 points en prédiction directe, 96,2). D'autre part, pour la meilleure configuration avec tâches auxiliaires, on voit que la prédiction indirecte du nombre de gouverneurs s'améliore, mais reste en deça de la prédiction directe (93,9 versus 96,2), ce qui montre qu'**il reste une marge de progression dans la manière d'utiliser cette tâche auxiliaire**.

Enfin, malheureusement, l'incrément obtenu sur la performance globale (FL) est beaucoup moins fort que pour la tâche H.

	Précision indirectes (dans graphes prédits)			Précision directe	
Configuration	B=S	H	D	B	H
aucune tâche auxiliaire	86,7	92.2	89,5	NA	NA
meilleure sans propagation (BH)	87,9	93,5	90,1	87,6	96,5
meilleure avec propagation (BH $c^{(B)}$=1 $c^{(H)}$=10)	88,4	93,9	90,4	87,6	96,2

TABLE 7.4 – Précisions moyennes sur ensemble dev pour la meilleure combinaison (d'après résultats sur dev) dans trois configurations : sans tâche auxiliaire (ligne1), avec tâche auxiliaire sans propagation (ligne2), avec tâche auxiliaire et propagation en pile (ligne3). Col1-3 : précision pour les tâches auxiliaires, telle qu'observée dans les prédictions des graphes (prédiction du nombre de têtes, nb de dépendants, et de l'ensemble des étiquettes), Col4-5 : précision obtenue directement pour les tâches auxiliaires.

Cela dit, pour mieux mettre à profit les bonnes performances obtenues directement sur les tâches auxiliaires, nous avons testé une légère modification de la prédiction des arcs :

au lieu de prédire tous les arcs de score positif, on prédit les n meilleurs arcs entrants pour chaque token, avec n prédit soit directement par la tâche H (nombre de gouverneurs), soit indirectement au sein du BOL (tâche B) ou de la séquence d'étiquettes (tâche S). Mais cela donne des résultats équivalents voire un peu moins bons que l'inférence de base, tout en complexifiant la prédiction : prédire le bon nombre de gouverneurs est encore loin de prédire les bons gouverneurs.

7.2.4 Conclusion et perspectives

Pour améliorer la cohérence de graphes de dépendances prédits, nous avons proposé d'utiliser des tâches auxiliaires concernant l'ensemble des arcs entrants ou sortants d'un noeud. Testé sur les graphes syntaxiques profonds du FTB (chapitre 6), notre réimplémentation du parser biaffine vers graphes voit ses performances s'améliorer légèrement avec certaines combinaisons de tâches, alors même que l'architecture de base donne des performances assez hautes, grâce en particulier à l'utilisation de vecteurs contextuels [10].

Il reste à valider l'approche sur d'autres langues et données (en particulier les données anglaises et chinoises de (Oepen et al., 2015)). En outre, sur les données françaises, utiliser les arbres syntaxiques, comme tâche auxiliaire par exemple, devrait aider, car les graphes profonds sont assez proches des arbres de départ.

Enfin, je compte investiguer un compromis entre cette approche où les arcs sont tous scorés en même temps et indépendamment, et une architecture prenant les décisions séquentielles, avec un unique parcours de la phrase. En particulier, on peut envisager de commencer par scorer au plus un arc entrant avant de scorer des arcs supplémentaires représentant le partage d'arguments.

10. L'implémentation est disponible à https://github.com/mcandito/biaffine-graph-parser

Chapitre 8

Le projet ASFALDA : un FrameNet pour le français (résumé)

Je résumé ici le travail réalisé dans le cadre du projet ANR ASFALDA [1] (2012-2016) centré sur la production de ressources et d'analyseurs de type FrameNet, projet dont j'ai été la porteuse. De nombreuses personnes ont contribué au projet, et tout particulièrement Marianne Djemaa (dont la thèse portait sur la construction de la ressource FrameNet du français). Laure Vieu et Philippe Muller ont géré la structuration des cadres sémantiques et leur annotation en corpus (Djemaa et al., 2016 ; Vieu et al., 2016b ; Djemaa, 2017), et participé à la construction préalable du lexique (en collaboration avec Pascal Amsili, Lucie Barque, Farah Benamara, Gaël de Chalendar, Pauline Haas, Richard Huyghe, Yannick Mathieu et Benoît Sagot) (Candito et al., 2014a). Un analyseur de type FrameNet a été construit par Olivier Michalon dans le cadre de sa thèse (Michalon, 2017), encadrée par Alexis Nasr.

L'origine du projet tient à l'absence, à l'époque, d'outils et données pour l'analyse sémantique du français, que ce soit la désambiguïsation lexicale et/ou l'explication des rôles sémantiques des arguments des prédicats [2]

Nous avons choisi de repartir du projet Berkeley FrameNet, déjà réutilisé pour différentes langues, tout en nous permettant de corriger certains aspects. Il s'agit de grouper les sens d'unités lexicales en cadres sémantiques, au sein desquels on type les rôles sémantiques des participants. L'annotation en corpus (manuelle ou automatique) consiste à repérer les mots évoquant des cadres (des déclencheurs), et repérer quelles expressions linguistiques remplissent quels rôles sémantiques. À noter que l'analyse de type FrameNet reste superficielle en comparaison à une représentation sémantique complète des phrases. D'une part, les phénomènes de portée et de factivité ne sont pas du tout traités, d'autre part, les différentes annotations en cadres FrameNet au sein d'une même phrase ne sont pas actuellement organisées en une seule structure connectée.

Les particularités de FrameNet sont d'abord de permettre de grouper des unités lexicales en classes sémantiques plus grossières (les cadres). C'est également le cas de VerbNet

1. https://sites.google.com/site/anrasfalda/home

2. Pour les données annotées en sens, des annotations arriveront plus tard, en 2013, dans le cadre de la compétition internationale SemEval tâche 12 (Navigli et al., 2013), et ensuite avec les annotations de sens de verbes coordonnées par Vincent Segonne (Segonne et al., 2019). Pour les données annotées en classes sémantiques et/ou en rôles sémantiques, il existait des annotations issues de portage depuis l'anglais (Padó, 2007 ; van der Plas et al., 2011), de moindre qualité. Pour des données annotées en rôles sémantiques, le corpus CALOR (Marzinotto et al., 2018) a été créé concommitamment au projet ASFALDA. Il s'agit de l'annotation en corpus français de 50 cadres anglais, dans des textes spécialisés.

(Kipper-Schuler, 2005), qui organise le lexique en une hiérarchie de classes de verbes. Mais le regroupement en classes dans VerbNet se fonde sur un critère syntaxique, selon le principe des classes de Levin : il s'agit de regrouper des verbes d'après les alternances syntaxiques qu'ils admettent, l'idée fondamentale étant que le partage de propriétés syntaxiques fines donne au final des classes sémantiques grossières. Au contraire, la définition des cadres de FrameNet est beaucoup moins contrôlée par la syntaxe. Si cela peut apparaître moins bien motivé d'un point de vue linguistique, en pratique cela donne d'une part des cadres a priori plus portables à d'autres langues (comme prouvé en pratique par plusieurs projets de FrameNet dans d'autres langues (Boas, 2009)). D'autre part, cela permet dès le départ d'intégrer au sein d'un même cadre des déclencheurs de différentes catégories. Un exemple typique de cadre trans-catégorie est le cadre CAUSATION, qui représente une relation de cause à effet (les deux rôles sont donc Cause et Effect . Pour le français, nous avons pu annoter pour ce cadre des déclencheurs de type verbe (1), mais également des prépositions (2) ou conjonctions (3).

(1) L'appauvrissement du monde agricole a **CAUSÉ** de surcroît une forte décapitalisation [FTB-flmf7ah2ep-875]

(2) les nappes aquifères souterraines sont en voie de tarissement rapide **DU FAIT DU** pompage anarchique et intensif [FTB-flmf7ao2ep-810]

(3) il est en prison **PARCE QU'**il a osé, il y a quelque temps, proposer la recherche d'une solution politique au problème kurde. [Sequoia-Europar.550_00271]

En outre le même cadre peut grouper des déclencheurs dont les arguments sémantiques se réalisent dans des positions syntaxiques différentes (i.e. des déclencheurs ayant un "linking" différent) comme par exemple la paire de verbes converses *causer* (1) / *résulter* (4).

(4) [...] dégradations écologiques **RÉSULTANT** des abus commis au nom de la sacro-sainte productivité [FTB-flmf7ao2ep-856]

Les cadres peuvent également avoir des déclencheurs nominaux. Pour le cadre CAUSATION, ces déclencheurs ont la particularité qu'en emploi référentiel, un des rôles est rempli par le SN dont le déclencheur est la tête, comme en (5) : le déclencheur est *effet*, et le SN dont il est la tête réfère bien à l'effet de la causalité évoquée (i.e. en FrameNet, remplit le rôle Effect du cadre CAUSATION évoqué).

(5) la baisse de 0,8 % des commandes [...] a complètement annihilé l'**EFFET** positif attendu de la baisse des taux d'intérêt [...] [FTB-flmf7ag2ep-789]

La caractéristique de grouper des déclencheurs de différentes catégories au sein d'un même cadre traduit une orientation résolument plus sémantique de FrameNet, qui permet de grouper au sein de cadres des déclencheurs au comportement syntaxique potentiellement très différent.

Par ailleurs, la particularité des projets comme FrameNet, PropBank (Palmer et al., 2005), VerbNet est d'utiliser des rôles sémantiques, au contraire par exemple du célèbre réseau lexical Wordnet. Les rôles sémantiques sont à la fois un concept linguistique et un outil permettant des généralisations utiles en TAL. Côté linguistique, les rôles sémantiques sont un concept controversé, posé dès les années 60, dans le cadre de théories du "linking", i.e. la relation entre les arguments sémantiques d'un prédicat et leurs réalisations syntaxiques. Une théorie du linking doit rendre compte des généralisations observables dans la manière dont les arguments

sémantiques des prédicats se réalisent en syntaxe, en fonction du contenu sémantique du prédicat. Par exemple, si un verbe transitif a un argument agentif, celui-ci se réalisera comme sujet à l'actif (sauf dans les langues ergatives (Wechsler et al., 2020)).

Cependant, l'établissement d'une liste limitée de rôles sémantiques trans-lexicaux (i.e. non spécifiques à un prédicat donné, comme le sont les rôles "mangeur" et "mangé" pour le verbe *manger*), avec des critères définitoires linguistiquement motivés, s'est vite avéré difficile lorsqu'il s'est agi de passer à l'échelle dans la couverture du lexique des verbes. Dowty (1989) résume les deux problèmes majeurs rencontrés : le manque de consensus sur la liste des rôles sémantiques nécessaires et le manque de critères opérationnels pour assigner les rôles aux arguments apparaissant dans les phrases[3]. Cela amène certains formalismes à se passer totalement de la notion de rôles sémantiques pour établir des généralisations de linking, comme par exemple HPSG, qui leur préfère une décomposition du sens des prédicats, incluant un inventaire réduit de relations entre arguments sémantiques (Wechsler et al., 2020). À noter cependant que la notion de rôle sémantique est utilisée également en psycholinguistique. Par exemple, même si les rôles d'Agent et Patient ne peuvent être définis en termes de conditions nécessaires et suffisantes, Rissman and Majid (2019) passent en revue les arguments en faveur de l'existence d'un biais universel chez les humains d'encoder ces deux catégories distinctes, quoique de manière variée entre langues.

Quoi qu'il en soit, du côté du TAL, désigner par des rôles sémantiques les arguments des prédicats permet une généralisation dans la représentation du sens, une description plus économe ne serait-ce que pour la description du contenu propositionnel des phrases, mais également par exemple pour l'expression plus économe de règles d'inférence. Ces rôles permettent une généralisation neutralisant en partie la variation lexicale (si par exemple des verbes distincts ont des arguments de même rôle), et la variation syntaxique, en particulier les alternances syntaxiques. Les problèmes soulevés en linguistique comme par exemple la difficulté à définir précisément la notion d'Agent (Cruse, 1973) peuvent être contournés en faisant au besoin des choix d'annotation moins fins. La ressource VerbNet est un exemple de lexique anglais à relativement grande échelle, utilisant une liste de 24 rôles sémantiques généraux. Ces rôles ont une définition succincte, qui n'est pas vraiment opératoire[4], mais ils ont permis de couvrir 5200 sens verbaux[5], représentant environ 3800 lemmes distincts. Au contraire, comme on l'a vu supra, le choix de FrameNet a été d'abandonner l'idée d'une liste restreinte de rôles génériques, pour définir des rôles spécifiques à chaque cadre, tout en permettant d'établir des liens entre rôles.

Ce choix nous a paru éviter des incohérences dans l'assignation de rôles génériques, et être plus conforme aux résultats linguistiques cités supra. Un exemple typique est celui des verbes converses (et leurs nominalisations), comme *acheter/vendre*, qui ont été bien identifiés comme problématiques dans la littérature des rôles sémantiques (voir par exemple (Dowty, 1991)), étant donné que l'acheteur et le vendeur sont tout aussi agentifs. Ne pouvant faire autrement

3. Newmeyer (2010) insiste sur le manque de consensus dans la définition des rôles, malgré une littérature très abondante sur le sujet. Je renvoie au résumé de Djemaa (2017) pour l'historique des différentes propositions ultérieures concernant les rôles sémantiques, en particulier la proposition de Dowty (1991) d'utiliser seulement des proto-rôles, assignés à une partie seulement des arguments, et sur la base d'une compétition entre critères.

4. C'est-à-dire que la définition elle-même ne permet pas toujours de trancher. Par exemple, le rôle Theme est utilisé pour l'objet direct du verbe *to cause*, sans satisfaire la définition générale, connue pour être problématique, du rôle Theme "used for participants in a location or undergoing a change of location" (Kipper-Schuler, 2005, p. 34).

5. https://verbs.colorado.edu/~mpalmer/projects/verbnet.html

avec une liste réduite de rôles, VerbNet (respectivement PropBank) utilise l'étiquette Agent (resp. A0) pour l'acheteur du verbe *acheter*, mais pour le vendeur du verbe *vendre*. On donne en (6) et (7) le traitement de deux exemples tirés de FrameNet, pour lesquels nous inférons les étiquettes qui seraient utilisées dans PropBank/Verbnet. FrameNet permet d'unifier la représentation de la situation de transaction commerciale.

(6) I Buyer(Arg0/Agent) **bought** a Sovereign guitar Goods(Arg1/Theme)
for 20 pounds Money(Arg3/Asset) from an absolute prat Seller(Arg2/Recipient)

(7) China canceled its Seller(Arg0/Agent) **sale**
of a uranium conversion facility Goods(Arg1/Theme) to Iran Buyer(Arg2/Recipient)

Il est clair que l'assignation du rôle Agent pour VerbNet (ou Arg0 pour PropBank) suit ici la syntaxe, en favorisant le sujet à l'actif. Des rôles spécifiques à un cadre, comme dans FrameNet, permettent d'éviter ce biais et d'obtenir une uniformisation de la représentation de situations comparables. Les généralisations entre cadres sont obtenues en établissant des relations entre cadres et entre rôles. Du point de vue TAL, cette organisation permet de choisir un niveau de granularité des rôles.

Mais la réticence à utiliser des critères syntaxiques dans la définition des concepts de FrameNet a le désavantage de moins pouvoir s'appuyer sur des critères formels. Par exemple on ne trouve pas dans la littérature FrameNet de critères clairs justifiant la définition de tel ou tel rôle au sein d'un cadre. L'indication que le type sémantique basique d'un rôle doit être constant (Ruppenhofer et al., 2006, p. 10) n'est pas assez précise. La définition de la notion de rôle "core" (que l'on peut traduire par "noyau")[6] n'est pas utilisable en pratique. Un autre retour d'expérience majeur suite au projet ASFALDA est que la littérature FrameNet n'est pas assez précise pour définir de manière opérationnelle la granularité des cadres. Un cadre est sensé rassembler des déclencheurs dont les arguments partagent les mêmes rôles noyaux, remplis par des remplisseurs de même type, et "le sens basique des déclencheurs doit être similaire" (Ruppenhofer et al., 2006, p. 11). De nombreux exemples d'incohérences dans les données anglaises découlent de ces imprécisions, en particulier concernant les rôles non essentiels. C'est en pratique trop vague pour décider quand grouper des déclencheurs dans un même cadre, versus les séparer, et cela mine la justification théorique de la notion de cadre.

La ressource annotée obtenue

Je renvoie à la thèse de Marianne Djemaa (Djemaa, 2017) pour la description de la ressource et des choix qui la sous-tendent.

Les contraintes de temps du projet nécessitaient de faire des choix pour la couverture visée. Nous avons opté pour une couverture exhaustive d'un nombre restreint de quatre domaines notionnels : la causalité, la communication verbale, les transactions commerciales et les positions cognitives (i.e. les cadres portant sur le degré de certitude d'un être doué de conscience sur une proposition). L'objectif a donc été de décrire de manière exhaustive ces domaines, en termes de cadres nécessaires, et pour chaque cadre, en termes de lexique, en partant d'un portage automatique simple à partir de l'anglais, avec révision manuelle.

6. "A core frame element is one that instantiates a conceptually necessary component of a frame, while making the frame unique and different from other frames" (Ruppenhofer et al., 2006, p. 19).

Réaliser le travail domaine par domaine a fait apparaître des paires de cadres dans FrameNet anglais qui nous semblaient proches du point de vue sémantique. Certaines précisions ont été apportées pour le FrameNet français concernant les critères de définition des rôles d'un cadre, et des conditions nécessaires pour grouper des unités lexicales dans un même cadre (Djemaa, 2017, pp. 70-85). Des critères syntaxiques explicites ont été introduits pour assoir les décisions de distinctions de cadres. Seuls les rôles essentiels ont été modélisés, i.e. les éléments sémantiquement obligatoires, pouvant être réalisés comme argument sous-catégorisés syntaxiquement. L'enjeu était une meilleure justification théorique et aussi d'éviter de créer de la polysémie artificielle.

Une fois le lexique et le réseau de cadres stabilisés pour les quatre domaines notionnels retenus, une phase pilote d'annotation a permis d'écrire le guide général d'annotation [7]. L'écriture du guide n'a pu que marginalement s'appuyer sur le guide FrameNet original car celui-ci a été écrit pour des annotations lexicographiques (i.e. sur exemples choisis). Une annotation en corpus tout venant requiert des solutions pour un plus large spectre de difficultés linguistiques classiques. Par ailleurs, les difficultés spécifiques pour l'annotation de la causalité ont fait l'objet d'un guide à part (Vieu et al., 2016a) [8].

L'annotation a été faite pour les occurrences des lemmes du lexique apparaissant dans les corpus Sequoia et FTB [9].

Environ 75% des 24922 annotations d'occurrences de cadres ont été faites en double annotation, par des annotatrices ayant une formation en linguistique computationnelle [10].

L'accord inter-annotateur sur les cadres est bon (85,9%). Pour les empans des remplisseurs de rôle, les annotateurs sont parfaitement d'accord les trois quarts du temps, avec de fortes disparités d'un domaine à un autre, qui s'avèrent être en fait liées à la catégorie des déclencheurs : les rôles pour les déclencheurs verbaux sont les plus faciles à annoter, ceux pour les déclencheurs nominaux sont les plus difficiles (Djemaa et al., 2016).

Analyse quantitative

La ressource est visualisable en ligne [11]. Les annotations sémantiques en elles-mêmes sont sous licence libre, mais pour obtenir le corpus avec toutes les couches d'annotations (morphologie, syntaxe, sémantique), il faut au préalable obtenir la licence pour le FTB. Les données distribuées comprennent les cadres et leurs relations, les annotations en corpus, et le lexique tel que ré-extrait des données annotées.

On donne Table 8.1 une caractérisation quantitative de la ressource obtenue, qui comprend 105 cadres. Parmi ceux-ci, 38 correspondent à des fusions ou modifications à partir des cadres anglais, et 16 sont complètement nouveaux.

On compte 873 lemmes ayant au moins une annotation, dont 490 sont "complètement couverts", i.e. dont toutes les occurrences rencontrées sont couvertes parmi les 105 cadres.

7. (Candito and Djemaa, 2016), http://asfalda.linguist.univ-paris-diderot.fr/documentation/asfalda_guide_annotation.pdf

8. http://asfalda.linguist.univ-paris-diderot.fr/documentation/asfalda_guide_desamb_Causation_Evidence.pdf

9. Avec un maximum de 100 premières occurrences pour les lemmes fréquents. Avant cela, une pré-annotation manuelle du statut du clitique réfléchi *se* a été faite, pour décider si l'unité lexicale à annoter doit comprendre le clitique (verbe intrinsèquement réfléchi, comme *s'apercevoir*) ou pas.

10. Il s'agit de Vanessa Combet, Noémie Faivre, Virginie Mouilleron et Emilia Verzeni, que je remercie chaleureusement pour leur implication dans le projet.

11. http://asfalda.linguist.univ-paris-diderot.fr/frameIndex.xml

Environ la moitié des lemmes sont des verbes, et un gros tiers sont des noms, les autres catégories étant plus marginales.

Par ailleurs, les différents rôles spécifiques à chaque cadre ont été regroupés en 41 "macro-rôles", trans-cadre [12]. Ce regroupement reste parfois spécifique à un domaine. On a par exemple un macro-rôle "money_giver" pour tous les rôles des cadres du domaine des transaction commerciale pour désigner l'entité qui achète. On conserve donc le principe d'une définition sémantique des rôles.

	Nb cadres distincts	Nb lemmes distincts	Nb sens	Nb cadres annotés (≠ Other_sense)
TOUS	105	873	1109	**16167**
lemmes complètement couverts		490		**7213**
Trans. commerciales	19	90	99	2930
Causalité	11	243	285	3895
Pos. cognitives	44	372	442	5426
Communication	47	347	411	5233
N	-	296	346	5282
V	-	446	594	9165
PREP	-	35	43	674
ADV	-	26	42	407
CONJ	-	22	28	301
ADJ	-	43	48	234

TABLE 8.1 – Statistiques du FrameNet du français (certains cadres appartiennent à plusieurs domaines).

8.1 Etude de l'interface syntaxe-sémantique

L'annotation sémantique décrite supra a été réalisée sur des phrases du Sequoia et du FTB, disposant donc déjà des analyses syntaxiques, ce qui permet d'étudier l'interface syntaxe-sémantique. On peut en particulier extraire la réalisation syntaxique des remplisseurs de rôles d'un cadre donné. Plus précisément, on peut considérer le chemin syntaxique entre le déclencheur et la tête du remplisseur de rôle.

Par exemple pour le déclencheur *apprendre*, sur les 18 occurrences annotées [13], le "linking" le plus fréquent est `[Cognizer: +suj, Content: +obj]`, i.e. simplement un rôle Cognizer réalisé comme sujet et un rôle Content réalisé comme objet, comme en (8).

(8) j' Cognizer ai **APPRIS** hier soir que M. Akim Birdal [...] a été remis en prison Content.
[Sequoia-Europar.550_00269]
LINKING :`[Cognizer: +suj, Content: +obj]`

12. http://asfalda.linguist.univ-paris-diderot.fr/macroroles.xml
13. Avec le cadre FR_COGNIZER_AFFECTING.VERACITY, cf. http://asfalda.linguist.univ-paris-diderot.fr/lu/lu1078.xml?mode=annotation.

On peut identifier des linkings différents pour ce même déclencheur *apprendre*, qui permet différentes organisations syntaxiques pour les rôles sémantiques Content (le contenu appris), Cognizer (la personne qui apprend le nouveau contenu), et la source de l'apprentissage, qui peut être un autre humain ou groupe d'humains (le Persuader) ou un autre type de source (rôle Evidence), comme illustré [14] dans les exemples (9) et (10), où la source est sujet, ou (11), où on retrouve le Cognizer en sujet, et la source est en complément oblique de_obj. L'intérêt ici est d'avoir le lien avec le corpus annoté, et une quantification des différents linkings.

(9) Nous Persuader le Content leur Cognizer **APPRENDRONS**.[FTB-flmf7ai1exp-63]
LINKING :[Cognizer: +a_obj, Content: +obj, Persuader: +suj]

(10) Cette exposition Evidence nous Cognizer **APPREND** que dès le XIIe siècle, [...], une industrie métallurgique existait Content. [Sequoia-annodis.er_00002]
LINKING :[Cognizer: +a_obj, Content: +obj, Evidence: +suj]

(11) les Saoudiens Cognizer ont **APPRIS** des chocs pétroliers de 1973 et de 1979 Evidence que maximiser les prix était une stratégie hasardeuse Content [...] [FTB-flmf7am2ep-648]
LINKING :[Cognizer: +suj, Content: +obj, Evidence: +de_obj]

8.1.1 Evaluation du degré de normalisation obtenu par la syntaxe profonde

De manière évidente, on peut faire ici le lien avec la syntaxe profonde présentée au chapitre 6, et distinguer les linkings utilisant les chemins syntaxiques de surface versus profonds, où une partie de la variation syntaxique est neutralisée. Comparer les régularités de linking obtenues avec les deux types de syntaxe permet d'évaluer le degré de normalisation fourni par la syntaxe profonde.

En effet, tout a été fait pour que l'éventail des réalisations syntaxiques pour les remplisseurs d'un même rôle soit moins varié en syntaxe profonde qu'en syntaxe de surface.

Dans (Michalon et al., 2016), nous avons quantifié cet effet de normalisation. On type les réalisations syntaxiques de rôles en considérant le chemin syntaxique (dans l'arbre de surface versus dans le graphe syntaxique profond) entre le déclencheur du cadre et le mot tête du remplisseur de rôle. On calcule ensuite l'entropie moyenne des distributions $P(chemin\_synt|cadre, role)$, qui sera d'autant plus grande que la variation syntaxique est grande. Et on obtient bien que cette entropie, moyennée sur tous les couples (cadre, rôle), est plus faible avec les chemins syntaxiques profonds (entropie=0.762) qu'avec les chemins syntaxiques de surface (entropie=1.149) [15].

Une autre façon d'observer la neutralisation syntaxique est d'observer simplement les chemins syntaxiques des remplisseurs de rôles, tous cadres et rôles confondus [16].

On observe Figure 8.1 les 15 chemins les plus fréquents, pour les déclencheurs verbaux. Sans surprise, les remplisseurs de rôles sont majoritairement réalisés comme sujet et objet du

14. L'extraction complète des linkings profond est disponible à l'adresse http://www.linguist.univ-paris-diderot.fr/~mcandito/divers/sequoiaftb.linkings_deep.summary.

15. Dans (Michalon et al., 2016) nous calculions l'entropie des distributions $P(chemin\_synt|role)$ au lieu de $P(chemin\_synt|cadre, role)$, mais la tendance est bien la même.

16. Dans toute cette section, on ignore les remplisseurs de rôles annotés comme antécédent de remplisseurs anaphoriques également annotés, cf. on étudie ici la régularité syntaxique, or les antécédents introduisent une variation syntaxique ad hoc.

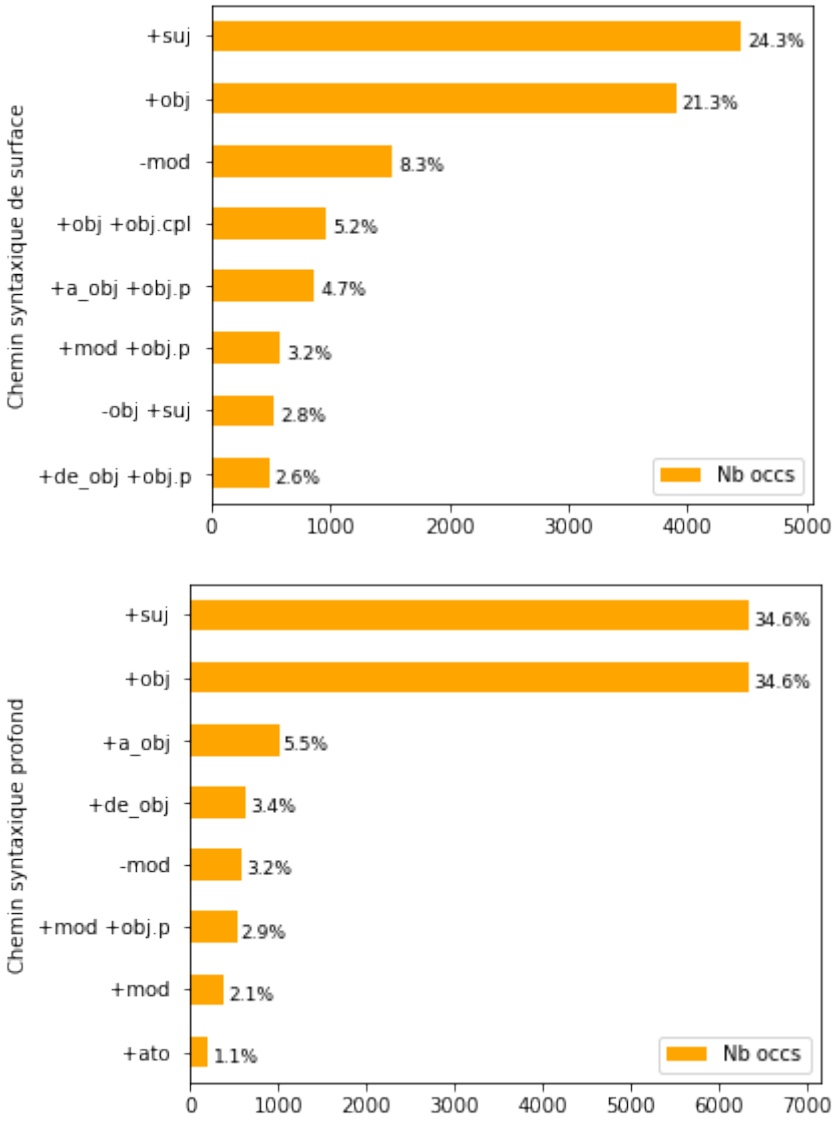

FIGURE 8.1 – Chemins syntaxiques des remplisseurs de rôles, tous cadres et rôles confondus, pour les déclencheurs verbaux. Comparaison entre chemins syntaxiques de surface (Haut) et profonds (Bas).

verbe. Alors qu'en syntaxe de surface, ces deux chemins couvrent 45% des rôles, on passe à presque 70% en utilisant la syntaxe profonde.

Mise à profit pour le parsing FrameNet

Cette normalisation syntaxique a pu être mise à profit dans un analyseur en cadres sémantiques (Michalon et al., 2016). Il s'agit d'une architecture simpliste en séquence, avec un classifieur par lemme pour prédire le cadre d'une occurrence de déclencheur, puis d'un classifieur par cadre pour prédire la tête syntaxique des remplisseurs de rôles. Dans les classifieurs vers rôles, le chemin syntaxique entre le déclencheur et le candidat remplisseur de rôle est utilisé comme trait. Or on vient de voir que ces chemins sont nettement plus réguliers dans les graphes syntaxiques profonds que dans les arbres de dépendances. Et effectivement, on constate une nette amélioration de l'identification des remplisseurs de rôles, en particulier dans le cas de déclencheurs verbaux (la Fmesure pour les rôles étiquetés passant de 63.1 à 68, en utilisant des graphes syntaxiques profonds prédits, en utilisant un analyseur en dépendances, puis les règles de conversion).

Il reste à démontrer que cet effet bénéfique persiste avec une architecture plus récente, en particulier un apprentissage multi-tâches pour l'analyse en graphes de dépendances et l'analyse en cadres sémantiques.

8.2 Bilan

Nous avons résumé la création d'un FrameNet du français, contenant une centaine de cadres et environ 16000 annotations de cadres, avec leurs remplisseurs de rôles. On obtient une ressource richement annotée, couvrant complètement quatre domaines notionnels (communication verbale, causalité, transactions commerciales, positions cognitives). La couverture est malheureusement beaucoup plus faible qu'initialement espéré. En particulier, nous avions sous-estimé la difficulté de définir les cadres sémantiques de manière cohérente entre eux. Pour obtenir une ressource couvrante, nous allons nous tourner à l'avenir vers l'induction automatique de cadres comme évoqué dans le prochain et dernier chapitre de conclusions et perspectives.

Chapitre 9

Conclusion et perpectives de recherche

J'ai présenté dans ce mémoire différentes recherches visant à expliciter des représentations linguistiques de phrases (comme des arbres syntaxiques, des graphes de dépendances bilexicales, ou des cadres et rôles sémantiques). J'ai travaillé à la production de ressources, et à la conception de systèmes appris sur ce type de ressources. Dans cette lignée, une perspective à court terme est le travail sur l'analyse en graphes de dépendances (cf. fin du chapitre 7), où diverses pistes sont possibles. On peut investiguer de mieux mettre à profit les tâches auxiliaires au moment de l'apprentissage, ou encore, de séparer la prédiction d'au plus un arc entrant principal, de la prédiction d'arcs supplémentaires.

Il reste que le TAL actuel questionne l'approche même de fournir des représentations symboliques de phrases. Côté représentations syntaxiques, la tâche est certes loin d'être résolue pour des productions langagières "non standard" (comme celles "générées par les utilisateurs"). Mais, même dans le cas de genres bien couverts, l'analyse syntaxique ne semble pas donner d'avantages cruciaux aux systèmes neuronaux, qui peuvent s'en passer (cf. ne serait-ce que les gains rapportés pour toutes les tâches du jeu GLUE, obtenus avec BERT (Devlin et al., 2019), avec une approche sans analyses traditionnelles intermédiaires).

Côté représentations sémantiques, on peut supposer que l'accès à ces représentations serait plus utile que la syntaxe. Mais, d'une part, décider du type adapté de représentation sémantique, pour la linguistique et/ou le TAL, est un problème ouvert. D'autre part, contrairement à la syntaxe, les données richement sémantiquement annotées manquent cruellement.

Pourtant, les représentations symboliques ont l'avantage d'être directement interprétables, contrairement aux paramètres et prédictions neuronales.

Aussi mes perspectives de recherche actuelles portent-elles sur l'émergence de symbolique au sein de réseaux neuronaux. J'explore actuellement avec Pascal Amsili et David Kletz l'étude de la composition au sein des représentations contextuelles, en particulier le traitement de la négation. Le postulat de base est que pour obtenir des modèles s'approchant d'une compréhension du sens, on ne peut faire l'impasse sur le principe de compositionnalité du sens. Il s'agit d'une part d'étudier si une forme de compositionnalité se dégage des modèles existants, et d'autre part d'investiguer quelles tâches génériques favoriseraient plus l'apprentissage de la composition.

Une autre façon d'aborder l'interprétabilité des prédictions neuronales est de réintroduire la notion d'unité lexicale. Aussi mon principal sujet de recherche pour la suite porte sur l'induction semi-supervisée de sens et cadres sémantiques. Ce projet s'inscrit dans le cadre du projet ANR SELEXINI "SEmantic LEXicon INduction for Interpretability and diversity in text processing", porté par Carlos Ramisch, qui vient d'être accepté (appel ANR AAPG 2021). Je serai en charge

en particulier du WP sur l'induction de sens et cadres sémantiques. Le projet comprend en outre toute une partie sur la génération de libellés lisibles par un humain pour les éléments induits automatiquement, ainsi que l'utilisation du lexique sémantique induit au sein d'un systéme neuronal pour la tâche de QA de type extraction de réponse au sein d'un paragraphe.

Le postulat de départ est double. D'une part, malgré l'émergence de systèmes de bout-en-bout pour diverses tâches de TAL, nous faisons l'hypothèse que la notion d'unité lexicale et de lexique reste un moyen privilégié pour un TAL interprétable. Mais d'autre part, la production d'analyseur automatique réalisant la désambiguisation lexicale (WSD) et l'identification de structures argumentales est soumise à l'existence de ressources annotées qui sont de couverture toujours insuffisante, même pour l'anglais, langue la plus dotée.

Pour répondre à cette situation, le projet propose un compromis : un lexique automatiquement induit, qui sera bruité mais couvrant, et assorti de libellés interprétables. L'objectif est d'induire des sens de lemmes ainsi que des cadres sémantiques, à partir de corpus syntaxiquement analysés. Ces unités sont vues comme un compromis entre des représentations non désambiguïsées (de type un plongement non contextuel par lemme) et à l'autre extrême une représentation par occurrence (de type vecteur contextuel).

Pour tester l'utilité du lexique induit, celui-ci sera intégré au sein d'une tâche de question-réponse de type extraction d'extrait de paragraphe, sous forme de tâche auxiliaire. Il s'agira d'évaluer si une pré-structuration en unités lexicales peut aider un système neuronal utilisant un modèle de langue pré-entraîné.

Concernant l'induction proprement dite, la méthode qui sera explorée est le clustering semi-supervisé d'occurrences. L'aspect semi-supervisé tiendra au fait d'utiliser comme graines les sens de Wiktionary, et leurs exemples. Wiktionary est un bon candidat pour ne pas repartir de zéro, car il offre un cadre homogène pour des lexiques couvrants pour un nombre important de langues, et de bonne qualité malgré son mode de création participatif. Nous avons pu le vérifier avec Vincent Segonne et Benoît Crabbé (Segonne et al., 2019) concernant les sens des verbes français. Mais ce même travail a montré que les exemples Wiktionary sont en nombre insuffisant pour de la WSD supervisée. De ce point de vue, la technique d'induction automatique par clustering d'occurrences produit de facto un corpus pseudo-annoté en "sens'' : les occurrences clusterisées peuvent être vues comme annotées avec comme étiquette le cluster.

Les sens induits seront des clusters d'occurrences du même lemme, et les cadres seront des clusters de sens ou bien des clusters d'occurrences en levant la contrainte de correspondre à un seul lemme. La spécificité est l'aspect structuré des cadres, qui doivent comporter une représentation d'arguments sémantiques. Pour aborder ce point, nous souhaitons investiguer le clustering conjoint d'occurrences de prédicats en cadres sémantiques, et d'occurrences des arguments syntaxiques profonds en rôles sémantiques.

Bibliographie

Abeillé, A. (2004). Corpus le monde, annotation fonctionnelle, guide des annotateurs. http://www.llf.cnrs.fr/Gens/Abeille/guide-fonctions.new.pdf.

Abeillé, A. and Barrier, N. (2004). Enriching a French treebank. In LREC'04, Lisbon, Portugal.

Abeillé, A. and Clément, L. (1999). Corpus le monde, annotation morpho-syntaxique : Les mots simples-les mots composés. http://ftb.linguist.univ-paris-diderot.fr/fichiers/public/guide-morphosynt.pdf.

Abeillé, A., Godard, D., and Miller, P. (1997). Les causatives en français : un cas de compétition syntaxique. Langue francaise, 115 :62–74.

Abeillé, A., Hemforth, B., and Winckel, E. (2016). Les relatives en dont du français : études empiriques. In 5e Congrès Mondial de Linguistique Française, volume 27.

Abeillé, A., Toussenel, F., and Chéradame, M. (2004). Corpus le monde, annotation en constituants, guide pour les correcteurs, version du 31 mars 2004. http://ftb.linguist.univ-paris-diderot.fr/fichiers/public/guide-constit.pdf.

Abeillé, A. (1991). Une grammaire lexicalisée d'arbres adjoints pour le français : application à l'analyse automatique. PhD thesis, Université Paris 7.

Aho, A. V. and Ullman, J. D. (1972). The Theory of Parsing, Translation, and Compiling. Prentice-Hall, Inc., USA.

Alemany-Puig, L. (2019). Edge crossings in linear arrangements : from theory to algorithms and applications. Master's thesis, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain.

Allén, S. (1968). Report on work in computational linguistics at the University of Göteborg. In Mater, E. and tindlová, J., editors, Les Machines dans la linguistique, Prague. Éditions de l'Académie Tchécoslovaque des Sciences.

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally Normalized Transition-Based Neural Networks. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers), pages 2442–2452, Berlin, Germany.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In Bengio, Y. and LeCun, Y., editors, 3rd International Conference on Learning Representations, ICLR 2015, pages 7–9, San Diego, CA, USA.

Ballesteros, M., Bohnet, B., Mille, S., and Wanner, L. (2014). Deep-Syntactic Parsing. In Proceedings of COLING 2014, pages 1402–1413, Dublin, Ireland.

Barque, L., Haas, P., Huyghe, R., Tribout, D., Candito, M., Crabbé, B., and Segonne, V. (2020). FrSemCor : Annotating a French corpus with supersenses. In Proceedings of LREC 2020, pages 5912–5918, Marseille, France.

Baschung, K. (1996). Une approche lexicalisée des phénomènes de contrôle. Langages, 30(122) :96–123.

Bernard, T. (2021). Multiple Tasks Integration : Tagging, Syntactic and Semantic Parsing as a Single Task. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics : Main Volume, pages 783–794, Online.

Boas, H. C., editor (2009). Multilingual FrameNets in computational lexicography : methods and applications. Trends in linguistics. Mouton de Gruyter.

Bohnet, B. (2010). Very high accuracy and fast dependency parsing is not a contradiction. In Proceedings of COLING 2010, pages 89–97, Beijing, China.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics, 5 :135–146.

Bouma, G., van Noord, G., and Malouf, R. (2000). Alpino : Wide Coverage Computational Analysis of Dutch. In Computational Linguistics in the Netherlands (CLIN), Tilburg, Nederlands.

Brants, S., Dipper, S., Hansen, S., Lezius, W., and Smith, G. (2002). The TIGER treebank. In The First Workshop on treebanks and linguistic theories (TLT), pages 24–41, Sozopol, Bulgaria.

Brown, P. F., Della, V. J., Desouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. Computational linguistics, 18(4) :467–479.

Buchholz, S. and Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In The Tenth Conference on Computational Natural Language Learning, pages 149–164.

Busa, R. (1974). Index Thomisticus Sancti Thomae Aquinatis Operum Omnium Indices Et Concordantiae in Quibus Verborum Omnium Et Singulorum Formae Et Lemmata Cum Suis Frequentiis Et Contextibus Variis Modis Referuntur. Frommann-Holzboog.

Cahill, A., Burke, M., O'Donovan, R., van Genabith, J., and Way, A. (2004). Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In Proceedings of ACL 2004, pages 320–327, Barcelona, Spain.

Candito, M. (1999). Organisation modulaire et paramétrable de grammaires électroniques lexicalisées application du français et a l'italien. PhD thesis, Université Paris Diderot.

Candito, M., Amsili, P., Barque, L., Benamara, F., de Chalendar, G., Djemaa, M., Haas, P., Huyghe, R., Mathieu, Y. Y., Muller, P., Sagot, B., and Vieu, L. (2014a). Developing a French FrameNet : Methodology and First results. In Proceedings of LREC 2014, pages 1372–1379, Reykjavik, Iceland.

Candito, M., Anguiano, E. H., and Seddah, D. (2011). A Word Clustering Approach to Domain Adaptation : Effective Parsing of Biomedical Texts. In IWPT, pages 37–42, Dublin, Ireland.

Candito, M. and Constant, M. (2014). Strategies for Contiguous Multiword Expression Analysis and Dependency Parsing. In Proceedings of ACL 2014), pages 743–753, Baltimore, Maryland.

Candito, M., Constant, M., Ramisch, C., Savary, A., Guillaume, B., Parmentier, Y., and Cordeiro, S. R. (2020). A French corpus annotated for multiword expressions and named entities. Journal of Language Modelling, 8(2) :415–479.

Candito, M. and Crabbé, B. (2009). Improving generative statistical parsing with semi-supervised word clustering. In IWPT, pages 138–141, Paris, France.

Candito, M., Crabbé, B., Denis, P., and Guérin, F. (2009). Analyse syntaxique du français : des constituants aux dépendances. In TALN 2009, pages –, Senlis, France.

Candito, M. and Djemaa, M. (2016). ASFALDA French FrameNet : Guide d'annotation [in French]. http://asfalda.linguist.univ-paris-diderot.fr/documentation/asfalda_guide_annotation.pdf.

Candito, M., Guillaume, B., Perrier, G., and Seddah, D. (2017). Enhanced UD Dependencies with Neutralized Diathesis Alternation. In Proceedings of the Fourth International Conference on Dependency Linguistics (Depling 2017), pages 42–53, Pisa, Italy. Linköping University Electronic Press.

Candito, M. and Liberman, M. Y. (2019). Introduction to the special issue on annotated corpora. Traitement Automatique des Langues, 60(2) :7–17.

Candito, M., Perrier, G., Guillaume, B., Ribeyre, C., Fort, K., Seddah, D., and de la Clergerie, É. (2014b). Deep Syntax Annotation of the Sequoia French Treebank. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 2298–2305, Reykjavik, Iceland.

Candito, M. and Seddah, D. (2012a). Effectively long-distance dependencies in French : annotation and parsing evaluation. In The 11th International Workshop on Treebanks and Linguistic Theories (TLT11), Lisbon, Portugal.

Candito, M. and Seddah, D. (2012b). Le corpus Sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In TALN, Grenoble, France.

Chamberlain, J., Fort, K., Kruschwitz, U., Lafourcade, M., and Poesio, M. (2013). Using Games to Create Language Resources : Successes and Limitations of the Approach. In Gurevych, Iryna, Kim, and Jungi, editors, Theory and Applications of Natural Language Processing, page 42. Springer.

Chen, D. and Manning, C. (2014). A Fast and Accurate Dependency Parser using Neural Networks. In EMNLP, pages 740–750, Doha, Qatar.

Chen, X. and Ferrer i Cancho, R., editors (2019). Proceedings of the First Workshop on Quantitative Syntax (Quasy, SyntaxFest 2019), Paris, France. Association for Computational Linguistics.

Chomsky, N. (1975). Reflections on Language. Pantheon Books, New York.

Clément, L. and Kinyon, A. (2003). Generating Parallel Multilingual LFG-TAG Grammars from a MetaGrammar. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, pages 184–191, Sapporo, Japan.

Collins, M. (1999). Head driven statistical models for natural language parsing. PhD thesis, University of Pennsylvania, Philadelphia.

Constant, M., Eryiğit, G., Monti, J., van der Plas, L., Ramisch, C., Rosner, M., and Todirascu, A. (2017). Survey : Multiword Expression Processing : A Survey. Computational Linguistics, 43(4) :837–892.

Copestake, A., Flickinger, D., Pollard, J. C., and Sag, I. A. (2005). Minimal Recursion Semantics : an Introduction. Research on Language and Computation, 4(3) :281–332.

Crabbé, B. (2005). Représentation informatique de grammaires fortement lexicalisées. PhD thesis, Université Nancy 2.

Crabbé, B. and Candito, M. (2008). Expériences d'Analyse syntaxique statistique du français. In Proceedings of TALN 2008, pages 45–54, Avignon, France.

Crabbé, B., Duchier, D., Gardent, C., Roux, J. L., and Parmentier, Y. (2013). XMG : eXtensible MetaGrammar. Computational Linguistics, 39(3) :591–629.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. The Journal of Machine Learning Research, 7 :551–585.

Crammer, K. and Singer, Y. (2003). Ultraconservative Online Algorithms for Multiclass Problems. Journal of Machine Learning Research, 3 :951–991.

Cruse, D. A. (1973). Some thoughts on agentivity. Journal of Linguistics, 9(1) :pp. 11–23.

de Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., and Manning, C. D. (2014). Universal Stanford dependencies : A cross-linguistic typology. In The Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA).

de Marneffe, M.-C., MacCartney, B., Manning, C. D., et al. (2006). Generating typed dependency parses from phrase structure parses. In Proceedings of LREC 2006, volume 6, pages 449–454.

de Marneffe, M.-C. and Manning, C. D. (2008). Stanford typed dependencies manual. http://nlp.stanford.edu/software/dependenciesmanual.pdf.

de Marneffe, M.-C. and Nivre, J. (2019). Dependency Grammar. Annual Review of Linguistics, 5(1) :197–218.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of NAACL 2019, pages 4171–4186, Minneapolis, Minnesota.

Dipper, S. and Kübler, S. (2017). German treebanks : Tiger and tüba-d/z. In Ide, N. and Pustejovsky, J., editors, Handbook of Linguistic Annotation, pages 595–639. Springer, Berlin.

Djemaa, M. (2017). Stratégie domaine par domaine pour la création d'un FrameNet du français annotation PhD thesis, Université Paris Diderot.

Djemaa, M., Candito, M., Muller, P., and Vieu, L. (2016). Corpus Annotation within the French FrameNet : a Domain-by-domain Methodology. In Proceedings of LREC 2016, pages 3794–3801, Portorož, Slovenia.

Dowty, D. (1991). Thematic proto-roles and argument selection. Language, 67(3) :547–619.

Dowty, D. R. (1989). On the semantic content of the notion of 'thematic role'. In Properties, types and meaning, pages 69–129. Springer.

Dozat, T. and Manning, C. D. (2017). Deep Biaffine Attention for Neural Dependency Parsing. In 5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings, Toulon, France. OpenReview.net.

Dozat, T. and Manning, C. D. (2018). Simpler but More Accurate Semantic Dependency Parsing. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers), pages 484–490, Melbourne, Australia.

Einarsson, J. (1976a). Talbankens skriftspråkskonkordans. Technical report, Lund University : Department of Scandinavian Languages.

Einarsson, J. (1976b). Talbankens talspråkskonkordans. Technical report, Lund University : Department of Scandinavian Languages.

Eisner, J. (1996). Three New Probabilistic Models for Dependency Parsing : An Exploration. In COLING, pages 340–345, Copenhagen.

Fernández-González, D. and Gómez-Rodríguez, C. (2019). Left-to-Right Dependency Parsing with Pointer Networks. 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), pages 710–716, Minneapolis, Minnesota.

Fernández-González, D. and Gómez-Rodríguez, C. (2020). Transition-based Semantic Dependency Parsing with Pointer Networks. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7035–7046, Online.

Ferrer i Cancho, R., Gómez-Rodríguez, C., and Esteban, J. (2018). Are crossing dependencies really scarce ? Physica A : Statistical Mechanics and its Applications, 493 :311329.

Flickinger, D., Zhang, Y., and Kordoni, V. (2012). Deepbank : A dynamically annotated treebank of the wall street journal. In Proceedings of the Eleventh International Workshop on Treebanks and Linguistic Theories (TLT11), pages 85–96, Lisbon, Portugal.

Fort, K., Adda, G., and Bretonnel Cohen, K. (2016). Éthique et traitement automatique des langues et de la parole : entre truismes et tabous. Revue TAL, 57(2) :7–19.

Foster, J. (2010). "cba to check the spelling" : Investigating Parser Performance on Discussion Forum Posts. In Proceedings of HLT-NAACL 2010, pages 381–384, Los Angeles, California.

Futrell, R., Mahowald, K., and Gibson, E. (2015). Large-scale evidence of dependency length minimization in 37 languages. In Proceedings of the National Academy of Sciences of the USA.

Gaifman, H. (1965). Dependency systems and phrase-structure systems. Information and Control, 8 :304–337.

Gibson, E. (2000). the dependency locality theory : A distance-based theory of linguistic complexity. In Miyashita, Y., Marantz, A., and O'Neil, W., editors, Image, language, brain, pages 95–126. MIT Press, Cambridge, MA.

Godard, D. and Sag, I. (1996). Quels compléments de nom peut-on extraire en français ? Langue française, 109 :60–79.

Grobol, L. and Crabbé, B. (2021). Analyse en dépendances du français avec des plongements contextualisés. In Denis, P., Grabar, N., Fraisse, A., Cardon, R., Jacquemin, B., Kergosien, E., and Balvet, A., editors, Traitement Automatique des Langues Naturelles, pages 106–114, Lille, France. ATALA.

Gross, M. (1986). Lexicon-Grammar : The Representation of Compound Words. In Proceedings of COLING 1986, pages 1–6, Bonn, Germany.

Guillaume, B., Bonfante, G., Masson, P., Morey, M., and Perrier, G. (2012). Grew : un outil de réécriture de graphes pour le TAL (Grew : a Graph Rewriting Tool for NLP) [in French]. In Proceedings of JEP-TALN-RECITAL 2012, volume 5 : Software Demonstrations, pages 1–2, Grenoble, France.

Guillaume, B., de Marneffe, M.-C., and Perrier, G. (2019). Conversion et améliorations de corpus du français annotés en Universal Dependencies. Revue TAL, 60(2) :71–95.

Guillaume, B., Fort, K., and Lefebvre, N. (2016). Crowdsourcing Complex Language Resources : Playing to Annotate Dependency Syntax. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics : Technical Papers, pages 3041–3052, Osaka, Japan.

Guzmán Naranjo, M. and Becker, L. (2018). Word order correlations from a quantitative perspective. In Grammar and Corpora 2018, Paris, France.

Hajic, J., Panevová, J., Hajicová, E., Sgall, P., Pajas, P., Štepánek, J., Havelka, J., Mikulová, M., Zabokrtskỳ, Z., and Razımová, M. Š. (2006). Prague dependency treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No. : LDC2006T01, Philadelphia, 98.

Hajič, J. (1998). Building a Syntactically Annotated Corpus : The Prague Dependency Treebank. In Hajičová, E., editor, Issues of Valency and Meaning. Studies in Honor of Jarmila Panevová, pages 12–19. Prague Karolinum, Charles University Press.

Hall, K. and Novák, V. (2005). Corrective modeling for non-projective dependency parsing. In IWPT, pages 42–52, Vancouver, Canada.

Henestroza Anguiano, E. and Candito, M. (2011). Parse Correction with Specialized Models for Difficult Attachment Types. In EMNLP, pages 1222–1233, Edinburgh, Scotland, UK.

Hockenmaier, J. (2003). Data and models for statistical parsing with Combinatory Categorial Grammar. PhD thesis, University of Edinburgh, College of Science and Engineering, School of Informatics.

Hovy, D., Spruit, S., Mitchell, M., Bender, E. M., Strube, M., and Wallach, H., editors (2017). Proceedings of the First ACL Workshop on Ethics in Natural Language Processing, Valencia, Spain.

Hudson, R. (1980a). A second attack on constituency : a reply to Dahl. Linguistics, 18(5-6) :489–504.

Hudson, R. A. (1980b). Constituency and dependency. Linguistics, 18(3-4) :179–198.

Ide, N. and Véronis, J. (1994). Multext : Multilingual text tools and corpora. In COLING 1994, pages 588–592.

Johnson, M. (1998). PCFG models of linguistic tree representations. Computational Linguistics, 24(4) :613–632.

Kahane, S. (1997). Bubble trees and syntactic representations. In the 5th Meeting of the Mathematics of the Language, Saarbrücken, Germany.

Kahane, S. (2009). Defining the Deep Syntactic Structure : How the signifying units combine. In Proceedings of MTT 2009, Montréal, Canada.

Kayne, R. (1975). French Syntax : the transformational cycle. MIT Press, Cambridge.

Kendall, A., Gal, Y., and Cipolla, R. (2018). Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7482–7491.

Kiperwasser, E. and Goldberg, Y. (2016). Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. Transactions of the Association for Computational Linguistics, 4 :313–327.

Kipper-Schuler, K. (2005). Verbnet : a broad-coverage, comprehensive verb lexicon. PhD thesis, University of Pennsylvania.

Kitaev, N., Cao, S., and Klein, D. (2019). Multilingual Constituency Parsing with Self-Attention and Pre-Training. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3499–3505, Florence, Italy.

Kitaev, N. and Klein, D. (2018). Constituency Parsing with a Self-Attentive Encoder. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers), pages 2676–2686, Melbourne, Australia.

Klein, D. and Manning, C. D. (2003). Accurate Unlexicalized Parsing. In ACL, pages 423–430, Sapporo, Japan.

Kondratyuk, D. and Straka, M. (2019). 75 Languages, 1 Model : Parsing Universal Dependencies Universally. In The 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Koo, T., Carreras, X., and Collins, M. (2008). Simple Semi-Supervised Dependency Parsing. In ACL-08, pages 595–603, Columbus, USA.

Kromann, M. (2003). The Danish dependency treebank and the DTAG treebank tool. In The 2nd International Workshop on Treebanks and Linguistic Theories (TLT2), Växjö, Sweden.

Kučera, H. and Francis, W. N. (1967). Computational analysis of present-day American English. Brown University Press.

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L., and Schwab, D. (2020). FlauBERT : des modèles de langue contextualisés pré-entraînés pour le français (FlauBERT : Unsupervised Language Model Pre-training for French). In Actes de JEP-TALN 2020, pages 268–278, Nancy, France.

Lecerf, Y. (1960). Programme des conflits, modèle des conflits. Bulletin bimestriel de l'ATALA, 1(4) :1118.

Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). End-to-end Neural Coreference Resolution. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017), pages 188–197, Copenhagen, Denmark.

Li, Y., Li, Z., Zhang, M., Wang, R., Li, S., and Si, L. (2019). Self-attentive Biaffine Dependency Parsing. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, pages 5067–5073. International Joint Conferences on Artificial Intelligence Organization.

Ma, X., Hu, Z., Liu, J., Peng, N., Neubig, G., and Hovy, E. (2018). Stack-Pointer Networks for Dependency Parsing. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers), pages 1403–1414, Melbourne, Australia.

Magerman, D. M. (1995). Statistical decision-tree models for parsing. In ACL, pages 276–283, Cambridge, Massachusetts, USA.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank : annotating predicate argument structure. In Proceedings of the workshop on Human Language Technology, pages 114–119, Stroudsburg, USA.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English : The Penn Treebank. Computational Linguistics, 19(2) :313–330.

Martin, L., Muller, B., Ortiz Suárez, P. J., Dupont, Y., Romary, L., de la Clergerie, É., Seddah, D., and Sagot, B. (2020). CamemBERT : a Tasty French Language Model. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7203–7219, Online.

Martins, A., Almeida, M., and Smith, N. A. (2013). Turning on the Turbo : Fast Third-Order Non-Projective Turbo Parsers. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2 : Short Papers), pages 617–622, Sofia, Bulgaria.

Martins, A. F. T. and Almeida, M. S. C. (2014). Priberam : A Turbo Semantic Parser with Second Order Features. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 471–476, Dublin, Ireland.

Marzinotto, G., Auguste, J., Bechet, F., Damnati, G., and Nasr, A. (2018). Semantic Frame Parsing for Information Extraction : the CALOR corpus. In LREC 2018, Miyazaki, Japan.

Matsuzaki, T., Miyao, Y., and Tsujii, J. (2005). Probabilistic CFG with Latent Annotations. In ACL, pages 75–82, Ann Arbor, USA.

McClosky, D., Charniak, E., and Johnson, M. (2006). Effective Self-Training for Parsing. In HLT-NAACL, pages 152–159, New York City, USA.

McDonald, R., Crammer, K., and Pereira, F. C. N. (2005a). Online Large-Margin Training of Dependency Parsers. In ACL, Ann Arbor, USA.

McDonald, R. and Pereira, F. (2006). Online Learning of Approximate Dependency Parsing Algorithms. In EACL, Trento, Italy.

McDonald, R., Pereira, F., Ribarov, K., and Hajič, J. (2005b). Non-Projective Dependency Parsing using Spanning Tree Algorithms. In HLT-EMNLP, pages 523–530, Vancouver, British Columbia, Canada.

Mel'čuk, I. (1988). Dependency Syntax : Theory and Practice. State University of New York Press, New York, USA.

Mel'čuk, I. (2009). Dependency in natural language. In Melčuk, I. and Polguère, A., editors, Dependency in linguistic description, Studies in Language Companion Series, pages 1–110. John Benjamins Publishing Company.

Melčuk, I. (1988). Dependency syntax : theory and practice. State University Press of New York.

Michalon, O. (2017). Modèles statistiques pour la prédiction de cadres sémantiques. PhD thesis, Aix-Marseille Universi'e.

Michalon, O., Ribeyre, C., Candito, M., and Nasr, A. (2016). Deeper syntax for better semantic parsing. In Proceedings of COLING 2016, pages 409–420, Osaka, Japan.

Mielziner, M. (1903). Introduction to the Talmud. Funk & Wagnalls.

Mille, S., Burga, A., and Wanner, L. (2013). AnCoraUPF : A Multi-Level Annotation of Spanish. In Proceedings of DepLing 2013, Prague, Czech Republic.

Miyao, Y. and Tsujii, J. (2005). Probabilistic disambiguation models for wide-coverage HPSG parsing. In Proceedings of ACL 2005, pages 83–90.

Montémont, V. (2008). Discovering frantext. In Auracher, J. and van Peer, W., editors, New Beginnings in Literary Studies, pages 89–107. Cambridge Scholars Publishing, Newcastle, UK.

Morris, W., editor (1969). The American Heritage Dictionary of the English Language. Houghton-Mifflin.

Navigli, R., Jurgens, D., and Vannella, D. (2013). SemEval-2013 Task 12 : Multilingual Word Sense Disambiguation. In Proceedings of SemEval 2013, pages 222–231, Atlanta, Georgia, USA.

Newmeyer, F. J. (2010). On comparative concepts and descriptive categories : A reply to Haspelmath. Language, 86(3) :688–695.

Nivre, J. (2008). Algorithms for Deterministic Incremental Dependency Parsing. Computational Linguistics, 34(4) :513–553.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1 : A Multilingual Treebank Collection. In Proceedings of LREC 2016, Portorož, Slovenia.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). Maltparser : A language-independent system for data-driven dependency parsing. Natural Language Engineering, 13(02) :95–135.

Nivre, J. and Scholz, M. (2004). Deterministic Dependency Parsing of English Text. In COLING 2004, pages 64–70, Geneva, Switzerland. COLING.

Oepen, S., Kuhlmann, M., Miyao, Y., Zeman, D., Cinková, S., Flickinger, D., Hajič, J., and Urešová, Z. (2015). SemEval 2015 Task 18 : Broad-Coverage Semantic Dependency Parsing. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 915–926, Denver, USA.

Oepen, S., Kuhlmann, M., Miyao, Y., Zeman, D., Flickinger, D., Hajič, J., Ivanova, A., and Zhang, Y. (2014). SemEval 2014 Task 8 : Broad-Coverage Semantic Dependency Parsing. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 63–72, Dublin, Ireland.

Oflazer, K., Say, B., Hakkani-Tür, D. Z., and Tür, G. (2003). Building a turkish treebank. In Abeillé, A., editor, Treebanks : Building and using parsed corpora. Kluwer, Dordrecht.

Padó, S. (2007). Cross-Lingual Annotation Projection Models for Role-Semantic Information. PhD thesis, Saarland University.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank : An Annotated Corpus of Semantic Roles. Computational Linguistics, 31(1) :71–106.

Pavlick, E., Post, M., Irvine, A., Kachaev, D., and Callison-Burch, C. (2014). The Language Demographics of Amazon Mechanical Turk. Transactions of the Association for Computational Linguistics, 2 :79–92.

Perlmutter, D. (1983). Studies in Relational Grammar 1. Studies in Relational Grammar. University of Chicago Press.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237, New Orleans, Louisiana.

Petrov, S. (2010). Products of Random Latent Variable Grammars. In NAACL, pages 19–27, Los Angeles, USA.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In ACL-COLING, pages 433–440, Sydney, Australia.

Ramisch, C., Nasr, A., Valli, A., and Deulofeu, J. (2016). DeQue : A Lexicon of Complex Prepositions and Conjunctions in French. In Language Resources and Evaluation Conference (LREC 2016), pages 2293–2298, Portorož, Slovenia.

Ribeyre, C. (2016). Data-driven methods for syntax-semantic interface. PhD thesis, Universit'e Paris Diderot.

Ribeyre, C., Candito, M., and Seddah, D. (2014). Semi-Automatic Deep Syntactic Annotations of the French Treebank. In Proceedings of the 13th International Workshop on Treebanks and Linguistic Theories (TLT13), pages 184–197, T"ubingen, Germany.

Ribeyre, C., de la Clergerie, E. V., and Seddah, D. (2016). Accurate Deep Syntactic Parsing of Graphs : The Case of French. In Proceedings of LREC 2016, pages 3563–3568, Portorož, Slovenia.

Ribeyre, C., Seddah, D., and Villemonte De La Clergerie, É. (2012). A Linguistically-motivated 2-stage Tree to Graph Transformation. In Proceedings of TAG+11, Paris, France.

Ribeyre, C., Villemonte De La Clergerie, E., and Seddah, D. (2015). Because Syntax does Matter : Improving Predicate-Argument Structures Parsing Using Syntactic Features. In Proceedings of NAACL 2015, pages 64–74, Denver, USA.

Rissman, L. and Majid, A. (2019). Thematic roles : Core knowledge or linguistic construct ? Psychonomic Bulletin & Review, 26(6) :1850–1869.

Rogers, A., Kovaleva, O., and Rumshisky, A. (2020). A Primer in BERTology : What We Know About How BERT Works. Transactions of the Association for Computational Linguistics, 8 :842–866.

Rosa, R., Mašek, J., Mareček, D., Popel, M., Zeman, D., and Žabokrtský, Z. (2014). HamleDT 2.0 : Thirty Dependency Treebanks Stanfordized. In Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 2334–2341, Reykjavik, Iceland.

Ruppenhofer, J., Ellsworth, M., Petruck, M. R., Johnson, C. R., and Scheffczyk, J. (2006). Framenet ii : Extended theory and practice. distributed with the data.

Sagae, K. (2010). Self-Training without Reranking for Parser Domain Adaptation and Its Impact on Semantic Role Labeling. In Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing, pages 37–44, Uppsala, Sweden.

Sagae, K. and Tsujii, J. (2008). Shift-Reduce Dependency DAG Parsing. In Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 753–760, Manchester, UK. Coling 2008 Organizing Committee.

Sagot, B., Clément, L., de La Clergerie, E. V., and Boullier, P. (2006). The lefff 2 syntactic lexicon for French : Architecture, acquisition, use. In LREC 06, Genoa, Italy.

Saied, H. A. (2019). Analyse automatique par transitions pour lidentification des expressions polylexicales. PhD thesis, Université Paris Diderot, France.

Savary, A., Candito, M., Mititelu, V. B., Bejček, E., Cap, F., Čéplö, S., Cordeiro, S., Eryiğit, G., Giouli, V., Maarten, G., HaCohen-Kerner, Y., Kovalevskaitė, J., Krek, S., Liebeskind, C., Monti, J., Escartín, C. P., van der Plas, L., QasemiZadeh, B., Ramisch, C., Sangati, F., Stoyanova, I., and Vincze, V. (2018). PARSEME multilingual corpus of verbal multiword expressions. In Markantonatou, S., Ramisch, C., Savary, A., and Vincze, V., editors, Multiword expressions at length and in depth : Extended papers from the MWE 2017 workshop. Language Science Press, Berlin, Germany.

Schuster, S. and Manning, C. D. (2016). Enhanced English Universal Dependencies : An Improved Representation for Natural Language Understanding Tasks. In Proceedings of LREC 2016), pages 2371–2378, Portorož, Slovenia. European Language Resources Association (ELRA).

Schwartz, R., Abend, O., and Rappoport, A. (2012). Learnability-based syntactic annotation design. In COLING, pages 2405–2422, Mumbai, India.

Seddah, D., Tsarfaty, R., Kübler, S., Candito, M., Choi, J. D., Farkas, R., Foster, J., Goenaga, I., Gojenola Galletebeitia, K., Goldberg, Y., Green, S., Habash, N., Kuhlmann, M., Maier, W., Nivre, J., Przepiórkowski, A., Roth, R., Seeker, W., Versley, Y., Vincze, V., Woliński, M., Wróblewska, A., and Villemonte de la Clergerie, E. (2013). Overview of the SPMRL 2013 Shared Task : A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages, pages 146–182, Seattle, Washington, USA.

Segonne, V., Candito, M., and Crabbé, B. (2019). Using Wiktionary as a resource for WSD : the case of French verbs. In Proceedings of the 13th International Conference on Computational Semantics - Long Papers, pages 259–270, Gothenburg, Sweden.

Steedman, M., Hwa, R., Clark, S., Osborne, M., Sarkar, A., Hockenmaier, J., Ruhlen, P., Baker, S., and Crim, J. (2003). Example selection for bootstrapping statistical parsers. In Proceedings of NAACL 2003, pages 157–164.

Strong, J. (1890). The Exhaustive Concordance of the Bible. Hodder and Stoughton.

Telljohann, H., Hinrichs, E., and Kübler, S. (2004). The tüba-d/z treebank : Annotating German with a context-free backbone. In Fourth International Conference on Language Resources and Evaluation (LREC'04), Lisbon, Portugal.

Tesnière, L. (1959). Éléments de syntaxe structurale. Klincksieck, Paris, France.

Titov, I., Henderson, J., Merlo, P., and Musillo, G. (2009). Online Graph Planarisation for Synchronous Parsing of Semantic and Syntactic Dependencies. In Boutilier, C., editor, IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, pages 1562–1567, Pasadena, CA, USA.

van der Plas, L., Merlo, P., and Henderson, J. (2011). Scaling up Automatic Cross-Lingual Semantic Role Annotation. In Proceedings of ACL 2011, pages 299–304, Portland, Oregon, USA.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is All you Need. In The 31st Annual Conference on Neural Information Processing Systems (NIPS), pages 5998–6008, Long Beach, USA.

Veecock, C. (2008). Se faire + infinitif : valeurs pragmatico-enonciatives dune construction "agentive" [in French]. In Proceedings of CMFL 2008, pages 2201–2217, Paris, France.

Vieu, L., Candito, M., and Muller, P. (2016a). Guide de désambiguisation entre les frames Evidence et Causation projet ASFALDA [in French]. http://asfalda.linguist.univ-paris-diderot.fr/documentation/asfalda_guide_desamb_Causation_Evidence.pdf.

Vieu, L., Muller, P., Candito, M., and Djemaa, M. (2016b). A General Framework for the Annotation of Causality Based on FrameNet. In Proceedings of LREC 2016, pages 3807–3813, Portorož, Slovenia.

Vijay-Shanker, K. and Schabes, Y. (1992). Structure Sharing in Lexicalized Tree-Adjoining Grammars. In Proceedings of COLING 1992.

Villemonte De La Clergerie, É. (2010). Convertir des dérivations TAG en dépendances. In Actes de la 17e conférence sur le Traitement Automatique des Langues Naturelles. Articles longs, pages 91–100, Montréal, Canada.

Villemonte de la Clergerie, É. (2010). Building factorized TAGs with meta-grammars. In Proceedings of the 10th International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+10), pages 111–118, New Haven, USA.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer Networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, Advances in Neural Information Processing Systems 28, pages 2692–2700. Curran Associates, Inc.

Wang, X., Huang, J., and Tu, K. (2019). Second-Order Semantic Dependency Parsing with End-to-End Neural Networks. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4609–4618, Florence, Italy.

Wechsler, S., Koenig, J.-P., and Davis, A. (2020). Argument structure and linking (prepublished draft). In Muller, S., Abeillé, A., Borsley, R. D., and Koenig, J.-P., editors, Head-Driven Phrase Structure Grammar : The handbook (prepublished version), Berlin, Germany. Language Science Press.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. (2020). Transformers : State-of-the-Art Natural Language Processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing : System Demonstrations, pages 38–45, Online.

Yamada, H. and Matsumoto, Y. (2003). Statistical Dependency Analysis with Support Vector Machines. In Proceedings of the Eighth International Conference on Parsing Technologies, pages 195–206, Nancy, France.

Zhang, Y. and Weiss, D. (2016). Stack-propagation : Improved Representation Learning for Syntax. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers), pages 1557–1566, Berlin, Germany.

Östen Dahl (1980). Some arguments for higher nodes in syntax : a reply to Hudson's 'constituency and dependency'. Linguistics, 18(5-6) :485–488.