

**ÉCOLE DOCTORALE MATHÉMATIQUES, INFORMATIQUE,
PHYSIQUE THÉORIQUE ET INGÉNIERIE DES SYSTÈMES**

LABORATOIRE : LIFO

THÈSE présentée par :

Marie ARCADIAS

soutenue le : **2 octobre 2015**

pour obtenir le grade de : **Docteur de l'université d'Orléans**

Discipline/ Spécialité : **Informatique**

**Apprentissage non supervisé de dépendances
à partir de textes**

THÈSE dirigée par :

Christel Vrain Professeur des universités, université d'Orléans

et encadrée par :

Guillaume Cleuziou Maître de conférence, université d'Orléans

Edmond Lassalle Ingénieur de recherche, Orange Labs, Lannion

RAPPORTEURS :

Mathieu ROCHE Chercheur, CIRAD, Montpellier

Isabelle TELLIER Professeur des universités, université Paris 3

JURY :

Guillaume CLEUZIOU

Laurence DANLOS

Edmond LASSALLE

Adeline NAZARENKO

Mathieu ROCHE

Isabelle TELLIER

Christel VRAIN

Maître de Conférence, université d'Orléans

Professeur, université Paris Diderot (Paris VII)

Ingénieur de recherche, Orange Labs, Lannion

Professeur, université Paris Nord (Paris XIII)

Chercheur, CIRAD, Montpellier

Professeur, université Sorbonne Nouvelle (Paris III)

Professeur, université d'Orléans

« — I need help on my homework. What's a pronoun ?

— A noun that lost its amateur status.

— ...

— Maybe I can get a point for originality. »

The Essential Calvin and Hobbes copyright © 1988 Bill Watterson

Table des matières

Table des matières	3
1 Introduction	7
1.1 Les grammaires de dépendances	8
1.2 Applications des structures de dépendances	11
1.3 Analyse de dépendances	12
1.4 Nos travaux, la grammaire DGdg	13
2 L'apprentissage non supervisé de dépendances	15
2.1 Travaux préliminaires à DMV	15
2.2 DMV, Dependency Model with Valence	16
2.3 Les descendants de DMV	21
2.4 Autres approches	22
2.5 Notre approche	23
3 Apprentissage non supervisé de dépendances par la méthode DGdg	25
3.1 Rappels sur les grammaires formelles de Chomsky	25
3.1.1 Grammaires hors contexte et forme normale de Chomsky	25
3.1.2 Grammaires hors contexte probabilistes	27
3.2 Algorithmes CYK et Inside Outside	28
3.2.1 CYK et CYK probabiliste	28
3.2.2 Inside Outside	33
3.3 Présentation de la famille de grammaires DGdg, les 5 variantes	38
3.3.1 ID/LP (Immediate Dominance/ Linear Precedence)	38
3.3.2 Principes et définition de la version 4bin de la famille de grammaires DGdg	40
3.3.3 Variantes additionnelles	43
3.4 Intégralité du processus d'apprentissage et d'analyse	48
3.4.1 Apprentissage des poids des règles des grammaires de la famille DGdg	48

3.4.2	Phase d'analyse	52
3.4.3	Traduction des arbres DGdg en arbres de dépendances	53
4	Résultats et analyse	57
4.1	Descriptif des données d'évaluation	57
4.2	Discussion sur la complexité des modèles DGdg et DMV	60
4.3	Expériences sur les catégories à gros grain, les 5 variantes de DGdg et DMV	61
4.4	Expériences partielles sur les catégories à grain fin, les 5 variantes de DGdg et DMV	64
4.5	Synthèse comparative avec l'état de l'art	66
5	Application des arbres de dépendances DGdg - - Extraction d'informations par CRF	71
5.1	Introduction	71
5.2	CRF, Conditional Random Fields	73
5.3	Utilisation des CRF et des arbres de dépendances construits par la méthode DGdg	75
5.3.1	Définition de la tâche	75
5.3.2	Données et choix du modèle de CRF	76
5.3.3	Les différents patrons de CRF testés	79
5.3.4	Résultats et modalités d'évaluation	84
5.3.5	Détail en fonction des étiquettes cibles et perspectives d'études	87
6	Conclusion	89
6.1	Nos travaux	89
6.2	L'application	90
6.3	Perspectives	90
6.3.1	Amélioration de la méthode DGdg	90
6.3.2	Utilisation des arbres de dépendances obtenus par la méthode DGdg	91
7	Annexes	93
7.1	Démonstration de l'équivalence de l'expressivité des formalismes "Arbre de dépendances" et "Arbre formel DGdg (4bin)"	93
7.1.1	Rappels et notations	93
7.1.2	Théorème et démonstration	94
7.2	Détail de la procédure de traduction des arbres DGdg en arbres de dépendances	95
7.3	Analyse par les variantes de la méthode DGdg sur des phrases simples	96
	Bibliographie	101



Remerciements

Merci évidemment à mes 3 encadrants, Christel Vrain, Guillaume Cleuziou et Edmond Lassalle, d'abord pour m'avoir fait découvrir l'informatique en général, l'apprentissage en particulier, et plus spécifiquement le traitement automatique des langues. Je reste fascinée par les classifieurs bayésiens de langues. Il est étonnant que par la seule donnée du décompte des suites de 3 lettres dans des textes dans des langues différentes, on puisse ensuite reconnaître la langue d'une nouvelle phrase.

Merci aux rapporteurs, Mathieu Roche et Isabelle Tellier pour leur lecture très attentive de ce manuscrit, et leurs retours enrichissants pour améliorer cet écrit.

Merci à Johannes Heinecke, pour son aide technique tout au long des 3 ans. Grâce à sa connaissance de la « langue » Latex, cette thèse est joliment présentée, la bibliographie est lisible.

Merci à Frédéric Gavignet et sa pédagogie en scripts bash et awk, pour son amitié et son humour sans failles.

Merci à Géraldine Damnati pour son aide sur les CRE, son attention et sa gentillesse.

Merci à Olivier Collin pour avoir, dès le début, douté de l'intérêt des travaux envisagés, et m'avoir motivé à évaluer concrètement et correctement la qualité de la méthode. Et pour son écoute ainsi que pour son soutien.

Merci à Mariano Belaude pour son aide en Python, sa bienveillance et sa bonne humeur constante.

Merci à Huanhuan Jin et Aleksandra Guerraz pour leur amitié et leur soutien sans faille.

Merci à Jocelyne Besnard pour sa disponibilité et son assistance dans toutes les tâches pratiques.

Merci à tous les collègues de l'équipe FAST d'Orange pour leurs conseils pertinents au long de ces 3 années, et aussi pour la bienveillance et l'intérêt qu'ils ont portés à mes travaux.

Merci à Jean-Yves Chapron pour sa relecture très attentive, et pour m'avoir signalé que Lari et Young n'étaient pas seulement connus pour avoir popularisé Inside-Outside, mais surtout, orthographiés légèrement différemment, pour être un grand organiste de Jazz.

Merci à ma grande Flore qui a bien poussé pendant ces 3 ans, et à ma petite Maëlle, production

personnelle parallèle à la thèse. Merci à toutes les deux pour la fierté que vous m'inspirez. Merci à mon mari, leur papa, pour son soutien quotidien.

Merci à mes parents, frères et sœur, beau-frère, belle-sœur, neveu, nièce, grands-parents et oncles qui ont pris le temps d'une journée à Orléans pour « soutenir ma soutenance ».

Introduction

Le Traitement Automatique des Langues Naturelles (TALN) a connu ses premiers succès dans les années 1950, en abordant la traduction d'une langue source dans une langue cible comme un problème de traitement d'information d'un type particulier. Déjà dans cette application précurseur se mêlaient les trois sciences que sont la linguistique (pour obtenir des phrases correctes dans la langue cible), l'intelligence artificielle (pour modéliser une compréhension des phrases à traduire) et l'informatique (pour mener le processus). Le principe du TALN est d'automatiser, faire faire uniquement par des programmes informatiques, des tâches impliquant l'utilisation de textes ou de paroles humaines, en modélisant une certaine compréhension artificielle du contenu de telles communications. On classe dans ce domaine des applications nombreuses et variées telles que la correction orthographique, la génération de textes écrits « à la manière d'un humain », le résumé automatique de texte, la partie post-traitement des OCR pour reconstituer des phrases à partir de caractères écrits manuellement, possiblement mal reconnus, ou même la reformulation d'expressions pour éviter les redondances.

Avec le développement du Web, et ainsi la mise à disposition d'une quantité phénoménale de textes et de paroles (dans des enregistrements audio, également dans des vidéos), le TALN a trouvé à la fois une source intarissable de données à traiter, mais aussi un très grand potentiel de nouveaux besoins, donc de nouvelles tâches à traiter. Parmi celles-ci, la traduction automatique a connu un nouvel essor, ainsi que la classification et catégorisation de documents. En recherche d'informations, le TALN associé à la fouille de textes est la base des outils de plus en plus performants. Ces tâches sont des exemples d'applications textuelles démultipliées, tant au niveau des besoins que de l'efficacité, par le caractère prolifique des données du net. Pour le discours oral, citons la reconnaissance automatique de la parole et la synthèse de la parole pour avoir connu elles-aussi des progrès fulgurants avec des applications aussi variées qu'utiles. La synthèse de la parole est par exemple un espoir pour certains malades perdant totalement la voix par des opérations programmées, la technologie leur permettant bientôt de s'exprimer complètement grâce à leur voix reconstituée d'après des enregistrements antérieurs à l'opération ¹.

1. <http://www.franceinter.fr/emission-interception-numerique-et-sante-les-voix-de-la-guerison>

L'intelligence artificielle est au centre de ces recherches qui donnent naissance à des technologies d'avenir. Au cœur de toutes ces applications trône la sémantique, la notion de sens d'une expression humaine. Au niveau de la phrase, cette compréhension peut s'interpréter, du point de vue de la machine, comme des liens plus ou moins directs entre les mots de la phrase, liens portés par une structure hiérarchique au sein de laquelle les mots porteurs du sens le plus fort sont au sommet. On peut commencer par reconnaître les catégories grammaticales de tous les mots de la phrase, sachant que souvent les verbes sont porteurs de sens fort, et que l'on peut très bien saisir un message auquel il manquerait les articles et déterminants. Ensuite, deux possibilités, non mutuellement exclusives, sont le regroupement des mots par unités syntaxiques (groupe nominal, groupe verbal ...) suivi d'une hiérarchisation totale ou partielle des groupes (analyse en constituants), ou bien le choix du mot le plus important auquel on attachera ensuite récursivement les mots syntaxiquement les plus proches (comme un adverbe au verbe) jusqu'à obtenir un arbre de mots à racine unique (analyse en dépendances). C'est cette deuxième voie, qui peut transiter par la première, que nous emprunterons pour stocker le sens des phrases dans l'âme d'un ordinateur.

1.1 Les grammaires de dépendances

L'analyse en constituants s'appuie sur une décomposition en syntagmes (nominaux, verbaux, adverbiaux, prépositionnels...) pour constituer une arborescence. Les nœuds de l'arbre sont ces constituants (groupe nominal, groupe verbal...) et les feuilles sont les catégories morphosyntaxiques, que nous désignerons désormais comme catégories grammaticales. La Figure 1.1 montre une analyse en constituants sur une phrase courte et simple. Des fonctions structurelles sont ajoutées sur les arêtes, elles sont un premier pas vers l'analyse en dépendances qui est le cadre des travaux de cette thèse. Nous désignerons indifféremment par le terme anglais « parsing », ou français, « analyse » ce travail de construction de la structure arborescente d'une phrase.

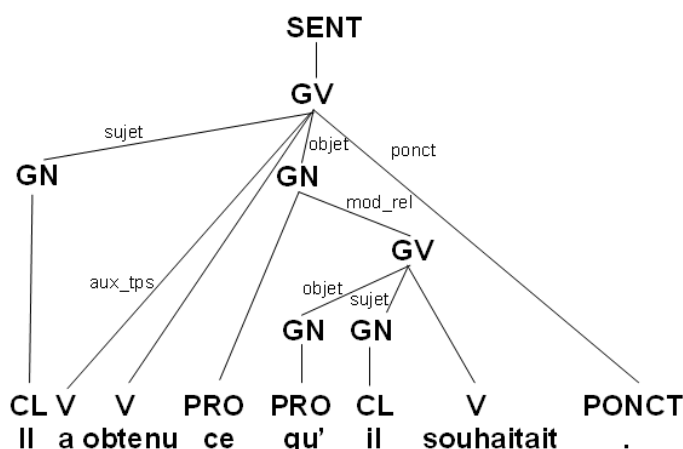


FIGURE 1.1 : Analyse en constituants de la phrase « Il a obtenu ce qu'il souhaitait », proposée dans le French Treebank [Abeillé et Barrier, 2004]. Les catégories grammaticales, ou constituants, CL, V PRO et PONCT signifient respectivement clitique, verbe, pronom et ponctuation. Les syntagmes GN et GV sont les abréviations de groupe nominal et groupe verbal.

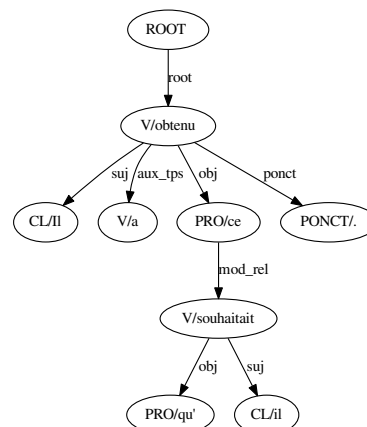


FIGURE 1.2 : Analyse de dépendances de la phrase « Il a obtenu ce qu'il souhaitait », proposée dans le French Treebank de dépendances [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010].

Les grammaires de dépendances trouvent leur origine chez Lucien Tesnière [Tesnière, 1959]. Cette approche vise à structurer les phrases suivant une hiérarchie guidée par des règles. Une phrase étant donnée, il s'agit de construire des relations asymétriques, appelées relations de dépendance, entre un mot considéré comme sémantiquement plus important et un mot moins important. Le deuxième « justifiant » sa place dans la phrase par ce mot plus important. Les grammaires de dépendances ont pour but de construire des arbres de dépendances dont les nœuds sont exactement les mots de la phrase. La Figure 1.2 montre une analyse en dépendances de la même phrase simple et courte que celle de la Figure 1.1.

Dans l'ouvrage de Kübler *et al.* [Kübler *et al.*, 2009] sont exposés des critères usuels d'identification d'une relation de dépendance entre une tête H (Head) et son dépendant D. Nous les reportons à la suite :

- H détermine la catégorie syntaxique de D et peut souvent remplacer D ;
- H détermine la catégorie sémantique de D ; D apporte des précisions sémantiques ;
- H est obligatoire, D peut être optionnel ;
- H sélectionne D et détermine si D est obligatoire ou optionnel ;
- la forme de D dépend de H ;
- la position linéaire de D est précisée en référence à H.

Certaines constructions font consensus dans les grammaires de dépendances. Par exemple, le verbe principal d'une phrase est toujours choisi comme racine de l'arbre de dépendances. Un nom, noyau du groupe nominal sujet, sera directement attaché au verbe principal. Un déterminant au sein d'un groupe nominal sera attaché au nom le plus proche à sa droite. En revanche, d'autres

constructions dépendent du courant théorique qui est respecté. Par exemple, il existe deux façons différentes de construire l'arbre de dépendances d'une phrase comportant une conjonction de coordination, qui sont illustrées par les Figures 1.3 et 1.4. La conjonction peut être considérée comme la tête de tout le groupe nominal (Figure 1.3) ou le dépendant du premier nom et la tête de la deuxième partie du groupe nominal (Figure 1.4). La représentation en ligne utilisée dans les Figures 1.3 et 1.4 est équivalente à la représentation en arbre, utilisée dans la Figure 1.2. Chaque mot de la phrase a exactement un parent, les flèches étant dirigées dans les deux cas du parent vers le dépendant.

Nous signalons que cette thèse n'a pas d'ambition linguistique, et que nous ne discuterons pas ces choix d'attachement. Nous considérons que ces choix sont précisés par l'utilisateur, nous les prenons en considération. Par défaut, nous avons gardé les choix faits par les linguistes nous ayant fourni les corpus d'arbres de dépendances à des fins d'évaluation.

Nous donnons à la suite un ensemble de termes, dont certains sont synonymes, que nous utiliserons indifféremment tout au long de cet écrit.

Dans un arbre de dépendances, les termes du lexique de la famille sont intuitifs. Le parent correspond à ce que nous avons présenté comme la tête dans une relation de dépendance. Le grand-parent est le parent du parent. Un ancêtre est logiquement un ascendant d'un mot à une ou plusieurs générations de distance. Un descendant est un dépendant, pouvant être à plusieurs générations de distance. L'enfant, le fils, sont des termes qualifiant le dépendant d'une relation de dépendance. Le frère d'un mot (vu comme un nœud) est un autre enfant du parent commun. L'ainé d'une tête est le premier de ses enfants que l'on rencontre lors de la lecture d'une phrase. Pour les langues dont on lit la phrase de gauche à droite, l'ainé est l'enfant positionné le plus à gauche. Inversement, pour l'arabe que l'on lit de droite à gauche, l'ainé sera l'enfant le plus à droite, ou pour le japonais qui peut se lire de haut en bas, l'ainé sera l'enfant le plus haut dans la phrase.

En dehors du vocabulaire de la famille, un argument est un autre terme désignant également le dépendant.

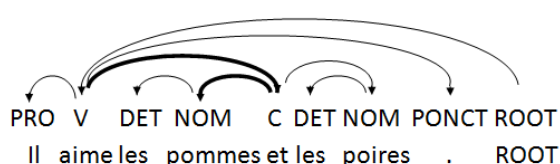


FIGURE 1.3 : Une analyse de dépendances possible. La conjonction de coordination « et » est attachée directement au verbe « aime », et est à la tête du sous-arbre représentant le groupe nominal « les pommes et les poires ».

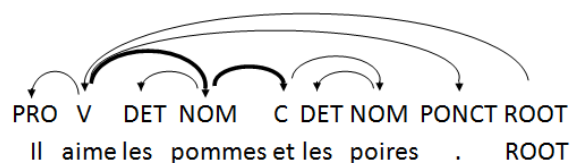


FIGURE 1.4 : Une autre analyse de dépendances possible. Le nom « pommes » est attaché au verbe et ensuite la conjonction de coordination « et » est attachée au nom « pommes », et n'est à la tête que du sous-arbre représentant le groupe nominal « les poires ».

Nous nous plaçons dans le formalisme des arbres de dépendances bare-bones, le terme ayant été introduit dans [Eisner, 1996]. Ce sont des arbres dont les nœuds sont les mots de la phrase considérée, dont on ne précise pas les fonctions syntaxiques liées aux arêtes de l'arbre, i.e. aux relations

de dépendance. On peut les représenter comme l'arbre de la Figure 1.2, en supprimant les indications annotant les arêtes. Nous désignerons dans la suite ces arbres par l'appellation générale arbres de dépendances. Il existe aussi des arbres de dépendances syntaxiques, avec tête. Ce sont des arbres construits par une analyse syntagmatique traditionnelle où chaque constituant possède un sous-constituant tête qui est un mot. Il s'agit de grouper les constituants en les associant à un même sous-constituant tête. Ceci permet de retrouver le parent en dépendance de chaque constituant dans l'arbre de dépendances équivalent à l'arbre syntagmatique avec tête. Le passage d'un formalisme à l'autre est exposé dans [Kahane, 2001].

1.2 Applications des structures de dépendances

L'analyse de dépendances permet de structurer, de hiérarchiser les mots d'une phrase. L'information obtenue est alors bien plus riche, au niveau sémantique, qu'un simple étiquetage grammatical ou même qu'une segmentation en syntagmes (à plat). L'expression *dependency parsing* est utilisée dans le même sens en anglais. L'apport de cette structuration est reconnu dans plusieurs domaines qui utilisent les analyses de dépendances obtenues par des parsers (analyseurs) automatiques performants lorsqu'ils sont correctement paramétrés.

Nous avons présenté les grammaires de dépendances dans le contexte de phrases écrites. Elles sont principalement utilisées en traitement automatique des langues et intelligence artificielle, mais interviennent dans des champs applicatifs plus éloignés, dont le plus prolifique est le domaine de la bio-informatique. La fouille de données, dans des articles médicaux techniques, provoque l'intérêt de nombreux chercheurs. Les articles [Wang *et al.*, 2014 ; Fayruzov *et al.*, 2008 ; Günes Erkan, 2007 ; Fundel *et al.*, 2007] présentent chacun leur outil d'identification automatique d'interaction entre protéines, prenant appui sur une analyse de dépendances des résumés de très nombreux articles médicaux.

Le désormais célèbre système de questions-réponses Watson d'IBM [McCord *et al.*, 2012], vainqueur au jeu Jeopardy, utilise son propre analyseur de dépendances au cours de la procédure d'étude de la question. Leur SG (Slot Grammar) parsing prend en compte les mots, et non les étiquettes grammaticales. On rappelle que l'objectif du jeu Jeopardy est de retrouver les questions dont les réponses sont données aux candidats. D'autres systèmes de questions-réponses avaient déjà envisagé avec succès l'utilisation d'analyses de dépendances, comme Wang *et al.* [Wang *et al.*, 2007]. Plus précisément, ils réussissent à faire correspondre une question à une réponse courte (à partir d'une liste de réponses) en s'appuyant sur les transformations syntaxiques entre question et réponse. En pratique, leur modèle prédit seulement l'étiquetage morphosyntaxique et l'arbre de dépendances de la question à retrouver, à partir de l'analyse de dépendances de la réponse.

L'analyse de dépendances est utilisée aussi pour l'extraction de relations. Dans l'article [Culotta et Sorensen, 2004], les auteurs utilisent des noyaux d'arbres (tree kernels) pour estimer la similarité entre des arbres de dépendances, et ainsi, au moyen d'un SVM (Support Vector Machine [Cortes et Vapnik, 1995]), ils détectent et classent des relations entre entités. Les relations extraites sont classées en types (« AT », « NEAR », « PART », « ROLE » et « SOCIAL ») et en sous-types (par exemple, pour le type « AT », les sous-types sont « Based-in », « Located » et « Residence »).

Les arbres de dépendances sont aussi utilisés pour des traductions automatiques. Quirk *et al.* [Quirk *et al.*, 2005] se servent d'un parser de dépendances dans la langue source, une segmentation

en mots dans la langue cible, et un composant (non supervisé) d'alignement de mots. Ils alignent ainsi les mots, projettent l'analyse de dépendances source sur la phrase cible, extraient des paires de mots liées par une relation de dépendance et apprennent un modèle de construction basé sur les arbres. Un décodeur efficace leur permet d'obtenir de bonnes traductions. Ding et Palmer [Ding et Palmer, 2004] construisent des grammaires d'insertion de dépendances afin d'effectuer un alignement syntaxique entre la langue source et la langue cible.

La construction de réseaux sémantiques, comme WordNet, peut s'appuyer sur des analyses de dépendances pour acquérir un caractère automatique. Snow *et al.* [Snow *et al.*, 2004] proposent un algorithme d'apprentissage d'hyperonymes (relation « is-a », par exemple « auteur » est un hyperonyme de « Shakespeare »). Dans les arbres de dépendances, ils cherchent à apprendre des motifs récurrents qui donnent lieu à des relations d'hyperonymie.

Dans d'autres domaines encore des chercheurs travaillent sur des arbres de dépendances déjà constitués pour profiter de la richesse de cette structure. on peut citer [Chelba *et al.*, 1997] pour la modélisation de langage et [Haghighi *et al.*, 2005] pour l'étude des implications textuelles.

1.3 Analyse de dépendances

Pour construire les arbres de dépendances de phrases données, dans le cadre d'une théorie linguistique, les règles peuvent être appliquées manuellement. Dans le cadre des applications présentées dans la section précédente, il faut automatiser la procédure pour analyser rapidement de très nombreuses phrases. L'option d'une construction manuelle d'arbres de dépendances n'est pas envisageable.

Le champ de recherche de l'analyse de dépendances est présenté en détail dans l'ouvrage [Kübler *et al.*, 2009]. Ils partagent les méthodes en deux types, eux-même scindés en deux types plus précis.

- Le parsing basé sur les grammaires :
 - le parsing de dépendances hors-contexte ;
 - le parsing de dépendances basé sur les contraintes.
- Le parsing de dépendances guidé par les données :
 - le parsing de dépendances basé sur les transitions ;
 - le parsing de dépendances basé sur les graphes.

Le parsing de dépendances basé sur les grammaires trouve son origine chez Gaifman [Gaifman, 1965] et Hays [Hays, 1964], qui ont débuté ces travaux avec des grammaires hors-contexte. Il s'agit, dans ces méthodes basées sur les grammaires, de définir des règles permettant de juger si une phrase donnée en entrée est conforme ou non à la grammaire. Les travaux suivants, qui envisagent les approches à base de contraintes, ont été menés par Maruyama [Maruyama, 1990]. Signalons que la manière de construire les grammaires hors-contexte, ou d'imposer les contraintes est libre. Cela signifie qu'un apprentissage peut être fait statistiquement, à partir de données. Une méthode d'analyse par grammaire hors-contexte peut donc se trouver à la fois dans le type de parsing basé sur les grammaires, et dans celui guidé par les données s'il nécessite un corpus d'apprentissage. Ces deux types de parsing ne sont pas mutuellement exclusifs.

Le parsing de dépendances guidé par les données a été initié par Eisner [Eisner, 1996]. Il a utilisé des méthodes basées sur les graphes. Le parsing basé sur les transitions a ensuite été introduit par

Matsumoto *et al.* [Kudo et Matsumoto, 2000 ; Yamada et Matsumoto, 2003]. Ce type de parsing de dépendances se décompose en deux phases successives. L'apprentissage est l'induction d'un modèle à partir d'arbres de dépendances donnés en exemples. L'analyse vient ensuite, elle consiste à construire les arbres de dépendances de phrases données non étiquetées, la construction devant être en accord avec le modèle induit durant la phase d'apprentissage.

Lorsqu'il existe des données d'apprentissage, i.e. des arbres de dépendances déjà construits, mais qu'elles ne sont pas en nombre suffisant pour induire un modèle de façon supervisée, les modèles semi-supervisés sont alors une alternative. De nombreux travaux ont été proposés : [McClosky *et al.*, 2006], [Koo *et al.*, 2008] ainsi que plus récemment [Li *et al.*, 2014].

Signalons également qu'il existe des travaux cherchant à réinvestir des modèles performants dans une langue pour une autre langue, pour procéder à des analyses de dépendances dans d'autres langues. Ma et Xia [Ma et Xia, 2014] s'appuient naturellement sur l'anglais, riche en modèles de parsing de dépendances performants, pour analyser des langues pauvres en données. Ce domaine avait déjà été désigné par Arun et Keller comme du « cross-linguistic parsing », dans l'article [Arun et Keller, 2005] lorsque pour la première fois des résultats avaient été obtenus pour le parsing en dépendances sur le French Treebank [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010].

1.4 Nos travaux, la grammaire DGdg

Les recherches et méthodes que nous avons évoquées à la section précédente ne nécessitent pas seulement d'importantes connaissances et compétences en informatique, en apprentissage, mais aussi des connaissances linguistiques très pointues, pour construire les modèles. Toutes les méthodes guidées par les données rendent indispensable un nombre conséquent (pour les méthodes semi-supervisées) et très important (pour les méthodes supervisées) d'arbres de dépendances corrects, construits à la main par des experts linguistes.

Nous visons, par nos travaux, à trouver un modèle simple, rapide, adaptable à tout type de texte et à toute langue. Nous excluons de tout le processus d'analyse l'introduction de connaissances linguistiques non basiques. Nous excluons l'apport de données d'apprentissage déjà sous forme d'arbres. Nous nous plaçons dans le cadre de l'Apprentissage Non Supervisé de Dépendances, que nous désignerons dans la suite de ce manuscrit par l'abréviation ANSD. Dans le chapitre 2, nous proposons une vue d'ensemble de ce domaine d'étude qu'est l'ANSD. Le modèle fondateur de l'ANSD est le Dependency Model of Valence (DMV), qui estime le nombre et le type de catégories morphosyntaxiques des catégories des enfants des mots de la phrase pour calculer les probabilités des différents arbres de dépendances possibles. Nous présentons en détail ce modèle dans le chapitre 2, ainsi que les différents travaux améliorant les résultats par des évolutions sur des parties ciblées du processus d'apprentissage. Nous évoquons aussi d'autres approches intéressantes en ANSD.

Le chapitre 3 présente la méthode que nous avons mise en place, basée sur une famille de grammaires hors-contexte que nous avons nommée DGdg, qui permet de construire surjectivement des arbres de dépendances bare-bones. L'apprentissage statistique des poids des règles de nos grammaires hors contexte est effectué après une phase minime d'adaptation au corpus, à la langue traitée. Dans ce chapitre 3, nous rappelons les définitions et propriétés des grammaires hors contexte,

l'algorithme d'apprentissage Inside-Outside et l'algorithme de décodage CYK, dans sa version probabiliste.

Nous avons collecté un volume important de données auxquelles nous comparer. Nous les présentons dans le chapitre 4. Nous disposons de 16 corpus d'arbres de dépendances, dans 12 langues différentes. Nous avons ensuite évalué la qualité des arbres obtenus grâce à notre famille de grammaires DGdg, et donnons pour la première fois des résultats pour le français en ANSD. Nous nous comparons à la version d'origine de DMV [Klein et Manning, 2004], qui nécessite moins de connaissances linguistiques que les versions les plus récentes, et qui est ainsi plus proche, en termes de prétraitements préalables, à notre modèle. Sans parvenir à construire des arbres de dépendances aussi bons que DMV sur tous les corpus, nous réussissons tout de même à obtenir de meilleurs scores sur quelques corpus. La comparaison avec DMV en termes de simplicité, d'adaptabilité et de temps de calcul pour les phases d'apprentissage et d'analyse tourne incontestablement en faveur du nouveau modèle proposé.

Afin de montrer l'intérêt de nos travaux, nous menons au chapitre 5 une expérience à périmètre limité, pour illustrer l'apport des arbres que nous construisons dans un test d'extraction d'information. Convaincus de la richesse des arbres de dépendances obtenus à peu de frais, nous avons mené des tests pour extraire des relations pertinentes dans du texte grammaticalement étiqueté. Nous avons choisi de comparer des expériences similaires tenant ou non compte des structures arborescentes des données d'évaluation, puis des arbres que nous avons obtenus par notre méthode DGdg. Nous présentons le cadre de l'extraction de relations, les modalités, résultats et conclusions de ces expériences dans le chapitre 5. Nous formalisons la tâche d'extraction de relations comme un étiquetage de chaque mot de la phrase, en fonction de son rôle dans un triplet (sujet, prédicat, objet). Nous utilisons les CRF, outil d'apprentissage supervisé particulièrement efficace pour les tâches d'étiquetage de séquences.

L'apprentissage non supervisé de dépendances

L'objectif du parsing de dépendances est de structurer les phrases pour déduire une hiérarchie sémantique entre ses mots, basée sur la syntaxe. Lorsqu'elle n'est pas guidée par les données, l'analyse de dépendances implique la création de modèles non adaptables à différentes langues ou styles de textes sans un travail de refonte intégrale du modèle. Les apprentissages supervisés et semi-supervisés, à partir d'arbres déjà construits à la main par des linguistes, offrent cette possibilité d'adapter un modèle, au prix de certains ajustements. Mais ces types d'apprentissage de dépendances impliquent un travail en amont considérable pour construire à la main suffisamment d'arbres de dépendances. Notre intention a été d'inventer un modèle portable, facilement adaptable, donnant lieu à un outil simple d'utilisation et s'ajustant sans effort à de nouvelles langues. Nous nous sommes donc naturellement placés dans le cadre de l'ANSI. Nous en présentons les travaux des précurseurs, en particulier le modèle fondateur DMV, ainsi que les évolutions et les approches originales.

2.1 Travaux préliminaires à DMV

Dès 1992, Carroll et Charniak [Carroll *et al.*, 1992] proposent un premier modèle de dépendances permettant un apprentissage non supervisé. À partir des catégories grammaticales, sans les mots lexicaux, ils proposent des grammaires hors contexte probabilistes dans le cadre desquelles sont construits des arbres. Chaque catégorie grammaticale possède un analogue qui se positionne comme nœud dans une structure arborescente. Les règles de dérivation indiquent, dans le membre de gauche, la catégorie parent de toutes les catégories situées dans le membre de droite. La procédure d'apprentissage consiste à donner une première estimation de la probabilité de chaque règle par des comptages dans le corpus d'apprentissage, à les réestimer par l'algorithme Inside-Outside (voir sous-section 3.2.2), et supprimer les règles dont les probabilités sont trop faibles. Leur idée de départ était ingénieuse, et a préparé les recherches futures dans ce domaine de l'apprentissage non supervisé de dépendances. Mais ils n'ont pas trouvé de solution aux nombreuses difficultés rencontrées, et leur modèle n'est pas utilisable tel quel. Il leur fallait réduire drastiquement le nombre de règles, une phrase prise au hasard comportant 41 terminaux (catégories grammaticales) était

conforme à 4×10^{13} règles différentes. L'autre difficulté majeure qu'ils n'ont pas pu surmonter est le piège des minima locaux (et non globaux) de l'algorithme Inside-Outside. Pour évaluer la qualité de leur méthode, ils ont artificiellement généré un corpus à partir d'une grammaire qu'ils avaient construite manuellement. L'apprentissage visait à retrouver cette grammaire construite. Après avoir testé 300 grammaires initiales différentes, ils ont obtenu 300 grammaires apprises différentes sur ce corpus généré, sans même réussir à retrouver leur grammaire construite.

La même année, Peirera et Schabes [Pereira et Schabes, 1992] proposent un modèle de dépendance par parenthésage. Leur approche est semi-supervisée, puisqu'elle nécessite un corpus déjà partiellement parenthésé. Ils utilisent Inside-Outside. Ils donnent des premiers résultats sur un corpus réel (Air Travel Information System spoken language corpus), à partir de phrases déjà analysées. Leurs résultats sont bons lorsqu'ils utilisent l'information de parenthésages préalables mais beaucoup moins pertinents sur du texte brut.

Yuret [Yuret, 1998], dans sa thèse, propose un point de vue intéressant : il cherche à lier entre eux les mots à contenu. Il différencie les mots à contenu, porteurs de sens comme les verbes, noms, adjectifs, des mots fonctionnels comme les prépositions ou conjonctions. La plupart des linguistes s'accordent sur les liens de dépendance entre les mots à contenu alors que les positions sur les attachements de mots fonctionnels peuvent être opposées suivant la théorie linguistique appliquée. Cependant, ce point de vue ne s'applique qu'à la partie évaluation de ses travaux, puisque des liens entre tous les mots sont tout de même construits. Contrairement aux autres travaux antérieurs à 2004, il ne tient pas compte, au cours de l'apprentissage et de l'analyse, des catégories grammaticales des mots, mais des mots lexicaux. Il cherche à modéliser l'attraction lexicale entre les mots, par des biais différents mais dans la même logique que des mesures de TF-IDF. Il lui faut alors des quantités considérables de données (100 millions de mots) pour apprendre un modèle moyennement performant (50% de rappel sur les liens entre mots à contenu sur un corpus de test au vocabulaire volontairement limité aux mots les plus fréquents du corpus d'apprentissage).

Le modèle de Paskin [Paskin, 2002] est proche de celui de Yuret, il modélise des « bigrammes » non de mots juxtaposés mais entretenant des liens de dépendance. Le modèle probabiliste qu'il propose implique que chaque tête induit ses dépendants, il s'apprend par une procédure de type EM en $O(n^3)$ [Dempster et al., 1977]. Il se différencie du modèle de Yuret qui utilise un algorithme heuristique, alors que les bigrammes grammaticaux de Paskin s'appuient sur une théorie de la syntaxe. Paskin annonce une précision légèrement inférieure à 40% sur les liens dirigés entre les mots, n'améliorant que peu les liens établis par une procédure aléatoire (31.3%).

En tirant parti des forces de ces différents modèles, et en corrigeant leurs faiblesses, Klein et Manning [Klein et Manning, 2004], en 2004, proposent le modèle considéré désormais comme l'état de l'art.

2.2 DMV, Dependency Model with Valence

La première méthode véritablement efficace en apprentissage non supervisé de dépendances a été élaborée par Klein et Manning et publiée pour la première fois en 2004 dans [Klein et Manning, 2004]. Leur modèle est désigné par DMV (pour Dependency Model with Valence), il s'appuie sur la valence des mots, à savoir le nombre de descendants que les mots peuvent admettre. C'est un modèle statistique de type « split head automaton grammar », qui sont des grammaires qui diffé-

rencient clairement les parties de séquence situées à gauche de celles situées à droite de chaque terminal. À chaque étape de la décomposition de la phrase suivant le canevas d'une split head automaton grammar, une scission claire est établie entre un élément central, la partie à sa gauche et la partie à sa droite.

Une phrase étant donnée, DMV considère la séquence de ses catégories grammaticales et vise à déterminer la racine de la phrase, puis de façon récursive les descendants directs de cette racine jusqu'à ce que tous les mots (i.e. les catégories grammaticales) aient trouvé leur place dans l'arbre de dépendances. Les choix successifs de la racine puis des descendants s'effectuent en fonction de poids fixés portant sur la propension de chaque catégorie à générer des enfants vers la gauche et vers la droite, afin de maximiser la probabilité de l'arbre entièrement construit dans le cadre de la grammaire DMV.

DMV est un modèle génératif, il peut être appris en fonction d'un corpus d'apprentissage, et peut être utilisé afin d'analyser de nouvelles phrases. L'apprentissage de ce modèle s'effectue généralement, et originellement, par une procédure EM, au cours de laquelle les probabilités des règles de la grammaire sont réestimées en fonction du corpus d'apprentissage. DMV permet une analyse de nouvelles phrases menant à la construction d'arbres dont la probabilité est maximisée en fonction de la grammaire stochastique apprise.

Dans l'article fondateur, [Klein et Manning, 2004], Klein et Manning suggèrent un apprentissage par l'algorithme Inside-Outside (voir [Lari et Young, 1990] et sous-section 3.2.2), en convertissant leur grammaire en grammaire probabiliste hors-contexte, tout en précisant, dans [Klein, 2005], que ce n'est pas la procédure la plus efficace et en indiquant seulement utiliser un algorithme EM. Quant à la phase d'analyse, ou parsing, ils n'évoquent pas l'algorithme utilisé. L'algorithme Viterbi, utilisé par leurs successeurs, semble une solution efficace.

Cohen et Smith, dans leur article [Cohen et Smith, 2010], présentent en détail ce modèle introduit par Klein et Manning, ainsi qu'un exemple de calcul de probabilité d'un arbre donné pour une phrase. Nous nous inspirons ici fortement de leur présentation du modèle.

La probabilité d'un arbre couvrant tous les mots de la phrase est calculée en combinant les probabilités de règles élémentaires dans un modèle récursif sur les sous-arbres. Les probabilités de ces règles élémentaires sont les suivantes :

1. Probabilité pour une catégorie d'être la racine d'une phrase.
2. Probabilité du lien de dépendance orienté d'une catégorie parent vers une catégorie enfant située à sa gauche dans la phrase.
3. Probabilité pour une catégorie de générer un premier enfant vers la gauche.
4. Probabilité pour une catégorie de générer un enfant vers la gauche alors qu'au moins un enfant a déjà été généré vers la gauche.
5. Probabilité pour une catégorie de ne pas générer un enfant vers la gauche.
6. Probabilité pour une catégorie de ne plus générer d'enfant vers la gauche alors qu'au moins un enfant a déjà été généré vers la gauche.

Les règles de 2 à 6, ont un exact équivalent en version droite au lieu de gauche.

L'arbre est donné en entrée, et on cherche sa probabilité.

Le modèle permet de calculer, à partir de ces poids élémentaires, la probabilité de chaque sous-arbre, puis finalement la probabilité de l'arbre de la phrase.

Les notations suivantes sont introduites pour exprimer, par l'équation 2.1, la probabilité du sous-arbre généré par le mot de position i . L'arbre de dépendances est construit, on considère que les enfants successifs, dans chaque direction (gauche ou droite), sont découverts au fur et à mesure du calcul récursif de la probabilité de l'arbre, du plus proche au plus éloigné de la catégorie en cours de considération.

- $\mathbf{y}$ est l'arbre de la phrase.
- $\mathbf{x}$ est la séquence des catégories de la phrase.
- x_i est la catégorie de position i dans la phrase.
- $\mathbf{y}^{(i)}$ est le sous-arbre généré par i . Il contient l'ensemble de ses descendants, directs et indirects.
- dir indique la direction, gauche ou droite, de génération des enfants.
- $\mathbf{y}_{dir}(i)$ est le sous-arbre généré par i dans la direction dir . $[\mathbf{y}_{dir}(i) = \emptyset]$ est un paramètre binaire testant si la catégorie de position i a déjà généré des enfants plus proches dans la direction D .
- θ est la désignation du modèle, de l'ensemble des poids des règles élémentaires. Ce modèle est constitué des distributions θ_S (S pour *STOP*) et θ_C (C pour *CHOOSE*).
- θ_S donne la distribution des probabilités d'arrêt ou de poursuite de la génération d'enfants, étant donnée une direction, la catégorie du parent et d'un booléen.
- θ_C est la distribution des catégories des enfants en fonction de la catégorie du parent et de la direction de génération.
- $firsty(j)$ est un prédicat. Plus précisément il est vérifié si la catégorie j est l'ainée, i.e. si son parent n'a pas déjà généré un enfant plus proche dans la même direction, s'il est le premier enfant.

L'équation 2.1 donne la probabilité du sous-arbre généré par le mot de position i .

$$\begin{aligned}
 P(\mathbf{y}^{(i)} | x_i, \theta) = & \prod_{dir \in \{gauche, droite\}} \theta_S(stop | x_i, dir, [\mathbf{y}_{dir}(i) \neq \emptyset]) \\
 & \times \prod_{j \in \mathbf{y}_{dir}(i)} \theta_S(\neg stop | x_i, dir, \overline{firsty(j)}) \times \theta_C(x_j | x_i, dir) \times p(\mathbf{y}^{(j)} | x_j, \theta)
 \end{aligned} \tag{2.1}$$

Intuitivement, cette formule modélise que certaines catégories sont plus ou moins enclines à générer des enfants vers la gauche (ou la droite), et que les probabilités varient au fur et à mesure du nombre d'enfants que la catégorie a effectivement déjà générés dans la phrase. Pratiquement, la probabilité $\theta_S(stop | x_i, dir, [\mathbf{y}_{dir}(i) \neq \emptyset])$ signifie qu'il y a des cas où la catégorie arrête de générer des enfants dans une direction, en fonction du fait qu'elle en ait déjà généré ou non. Si c'est la décision opposée qui est prise, alors la deuxième ligne de l'équation 2.1 intervient et les probabilités $\theta_C(x_j | x_i, dir)$ et $p(\mathbf{y}^{(j)} | x_j, \theta)$ entrent en jeu. La première pour juger si la catégorie envisagée comme enfant de la catégorie observée est probable, la seconde entretient l'aspect récursif, et prévoit la probabilité des sous-arbres possibles de cet enfant envisagé.

Illustrons sur un exemple cette formule récursive.

Exemple 2.1 : formule complète du calcul de la probabilité d'un arbre par DMV

Nous rappelons que DMV ne travaille que sur les catégories, pas sur les mots lexicaux.

Il est, pour des raisons pratiques, artificiellement ajouté un mot ROOT qui n'admet qu'un seul enfant, la racine de la phrase.

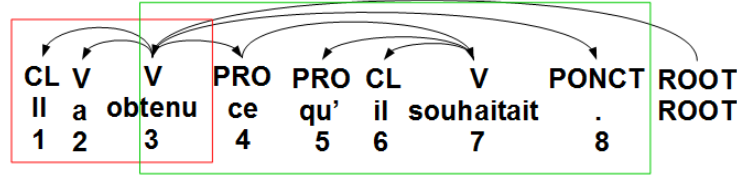


FIGURE 2.1 : Arbre de dépendances de la phrase « Il a obtenu ce qu'il souhaitait. ». Le sous-arbre généré par « obtenu » $\mathbf{y}^{(3)}$, vers la gauche $\mathbf{y}_{gauche}(3)$ est encadré de rouge, vers la droite $\mathbf{y}_{droite}(3)$ est encadré de vert.

$$\begin{aligned}
 P(\mathbf{y}^{(3)}|V, \boldsymbol{\theta}) = & \theta_S(\neg stop|V, gauche, false) \times \theta_C(V|V, gauche) \times p(\mathbf{y}^{(2)}|V, \boldsymbol{\theta}) \\
 & \times \theta_S(\neg stop|V, gauche, true) \times \theta_C(CL|V, gauche) \times p(\mathbf{y}^{(1)}|CL, \boldsymbol{\theta}) \\
 & \times \theta_S(stop|V, gauche, true) \\
 & \times \theta_S(\neg stop|V, droite, false) \times \theta_C(PRO|V, droite) \times p(\mathbf{y}^{(4)}|PRO, \boldsymbol{\theta}) \\
 & \times \theta_S(\neg stop|V, droite, true) \times \theta_C(PONCT|V, droite) \times p(\mathbf{y}^{(8)}|PONCT, \boldsymbol{\theta}) \\
 & \times \theta_S(stop|V, droite, true)
 \end{aligned}
 \tag{2.2}$$

Expliquons les trois premières lignes de cette équation, à partir du verbe « obtenu » vers la gauche :

$\theta_S(\neg stop|V, gauche, false)$: « obtenu » va générer des enfants vers la gauche, c'est un verbe. Le verbe « a » étant son plus proche descendant à gauche, le paramètre $[y_{gauche}(3) \neq \emptyset]$ est faux car aucun enfant n'a été généré vers la gauche avant « a ». Le sous-arbre $\mathbf{y}_{gauche}(3)$ est pour l'instant vide.

$\theta_C(V|V, gauche)$: « obtenu » est un verbe qui a généré le verbe « a » vers la gauche.

$p(\mathbf{y}^{(2)}|V, \boldsymbol{\theta})$: le calcul se continuera récursivement sur le sous-arbre de racine « a ».

$\theta_S(\neg stop|V, gauche, true)$: « obtenu » continue de générer des enfants vers la gauche, c'est un verbe. Le paramètre $[y_{gauche}(3) \neq \emptyset]$ est vrai car « obtenu » a déjà généré « a ». Le sous-arbre $\mathbf{y}_{gauche}(3)$ n'est plus vide, il contient « a ».

$\theta_C(CL|V, gauche)$: « obtenu » est un verbe qui a généré le clitique « Il » vers la gauche.

$p(\mathbf{y}^{(1)}|CL, \boldsymbol{\theta})$: le calcul se continuera récursivement sur le sous-arbre de racine « Il ».

$\theta_S(stop|V, gauche, true)$: « obtenu » ne va plus générer d'enfants vers la gauche. Le paramètre $[y_{gauche}(3) \neq \emptyset]$ est vrai car « obtenu » a déjà généré « a » et « Il ». Le sous-arbre $y_{gauche}(3)$ n'est plus vide, il contient « a » et « Il ».

Pour obtenir la probabilité de l'arbre de toute la phrase, il faut connaître la probabilité que cette troisième catégorie, V de « obtenu » soit la racine de la phrase, et les probabilités respectives des sous-arbres $y^{(1)}, y^{(2)}, y^{(4)}$ et $y^{(8)}$.

$$p(y|x, \theta) = \theta_C(V|ROOT, gauche) \times P(y^{(3)}|V, \theta) \quad (2.3)$$

Nous avons déjà vu que

$$\begin{aligned} p(y^{(3)}|V, \theta) &= \theta_S(\neg stop|V, gauche, false) \times \theta_C(V|V, gauche) \times p(y^{(2)}|V, \theta) \\ &\quad \times \theta_S(\neg stop|V, gauche, true) \times \theta_C(CL|V, gauche) \times p(y^{(1)}|CL, \theta) \\ &\quad \times \theta_S(stop|V, gauche, true) \\ &\quad \times \theta_S(\neg stop|V, droite, false) \times \theta_C(PRO|V, droite) \times p(y^{(4)}|PRO, \theta) \\ &\quad \times \theta_S(\neg stop|V, droite, true) \times \theta_C(PONCT|V, droite) \times p(y^{(8)}|PONCT, \theta) \\ &\quad \times \theta_S(stop|V, droite, true) \end{aligned}$$

On doit ensuite calculer les probabilités des sous-arbres des enfants de « obtenu ».

$$\begin{aligned} p(y^{(2)}|V, \theta) &= \theta_S(stop|V, gauche, false) \\ &\quad \times \theta_S(stop|V, droite, false) \\ p(y^{(1)}|CL, \theta) &= \theta_S(stop|CL, gauche, false) \\ &\quad \times \theta_S(stop|CL, droite, false) \\ p(y^{(4)}|PRO, \theta) &= \theta_S(stop|PRO, gauche, false) \\ &\quad \times \theta_S(\neg stop|PRO, droite, false) \times \theta_C(V|PRO, droite) \times p(y^{(7)}|V, \theta) \\ &\quad \times \theta_S(stop|PRO, droite, true) \\ p(y^{(8)}|PONCT, \theta) &= \theta_S(stop|PONCT, gauche, false) \times \theta_S(stop|PONCT, droite, false) \\ p(y^{(7)}|V, \theta) &= \theta_S(\neg stop|V, gauche, false) \times \theta_C(CL|V, gauche) \times p(y^{(6)}|CL, \theta) \\ &\quad \times \theta_S(\neg stop|V, gauche, true) \times \theta_C(PRO|V, gauche) \times p(y^{(5)}|PRO, \theta) \\ &\quad \times \theta_S(stop|V, gauche, true) \\ &\quad \times \theta_S(stop|V, droite, false) \\ p(y^{(6)}|CL, \theta) &= \theta_S(stop|CL, gauche, false) \\ &\quad \times \theta_S(stop|CL, droite, false) \\ p(y^{(5)}|PRO, \theta) &= \theta_S(stop|PRO, gauche, false) \\ &\quad \times \theta_S(stop|PRO, droite, false) \end{aligned}$$

La probabilité de l'arbre est donc entièrement décomposée sur ses sous-arbres, elle peut maintenant être calculée à partir des poids des règles élémentaires décrites avant l'exemple, les distributions θ_S et θ_C .

2.3 Les descendants de DMV

Les auteurs suivants ont tous gardé le modèle probabiliste DMV, et ont travaillé sur différentes approches de l'apprentissage pour améliorer la qualité des arbres ensuite analysés.

Gillenwater *et al.* [Gillenwater *et al.*, 2010 2011] ont travaillé sur un modèle de régularisation a posteriori de la disparité des catégories grammaticales. Dans DMV, il est supposé que toutes les catégories peuvent être en lien les unes avec les autres, ce qui n'est pas vrai dans les textes. Ils induisent alors une limitation du nombre de paires parent-enfant au cours de l'apprentissage. Ce biais pénalise les modèles avec un nombre important de types de dépendances distincts, et leur permet, grâce à cette introduction de connaissances linguistiques, d'obtenir de très bons résultats sur des corpus variés, en particulier en anglais. Ils utilisent le modèle de régularisation postérieure de [Graça *et al.*, 2007].

Cohen et Smith [Cohen et Smith, 2010 ; Cohen *et al.*, 2011], eux, ont travaillé sur les distributions a priori des probabilités conditionnelles θ_S et θ_C . Il s'agit d'attribuer des poids aux règles de la grammaire initiale. Ils proposent des distributions de Dirichlet, puis présentent des familles de distributions logistiques normales. Ce cadre leur permet de lier faiblement les paramètres du modèle et ainsi d'améliorer les performances de l'apprentissage des dépendances. Ils réutilisent aussi des distributions adaptées à une langue pour une autre langue et, par cette approche transverse, gagnent encore quelques points de performance. Pour ceci, ils doivent regrouper les catégories grammaticales des corpus considérés pour obtenir des catégories communes aux différentes langues afin de lier par des matrices de covariance les distributions des règles mettant en jeu les mêmes catégories.

Spitkovsky *et al.* [Spitkovsky *et al.*, 2011 ; Spitkovsky, 2013] introduisent un modèle nommé : « Lateen EM ». Ils agissent sur la phase EM de l'apprentissage. Ils cherchent à échapper au piège des optima locaux non globaux rencontré par Carroll et Charniak [Carroll *et al.*, 1992]. Pour ceci, ils font varier les objectifs d'espérance pour éviter les optima locaux en suivant deux objectifs, l'un contraint par une procédure hard-EM, l'autre par une procédure soft-EM. Les deux procédures collaborent en inter-validant les propositions de modification de la grammaire de l'autre. Comme il est expliqué dans l'article [Spitkovsky *et al.*, 2011], les voiles triangulaires (lateen, nom du modèle) permettent aux embarcations de naviguer à la fois par vents favorables et par vents contraires. Et de la même façon qu'un capitaine ne peut compter sur des vents favorables, une procédure d'apprentissage non supervisé ne peut s'appuyer sur des gradients coopérants. Leur méthode permet donc d'éviter les optima locaux, elle accélère la procédure par rapport à hard et soft-EM, mais ne permet d'améliorer les scores que par rapport à hard-EM.

Mareček et Straka [Mareček et Straka, 2013] se sont concentrés sur l'estimation a priori des probabilités STOP, ou d'arrêt de génération d'enfants dans une direction. Ils estiment ces probabilités sur un large corpus non structuré, par un principe de « réductibilité », et injectent ces estimations dans la grammaire d'initialisation de DMV. Sur de nombreux corpus issus des défis de CoNLL 06 et 07, ils montrent expérimentalement une amélioration notable des résultats par injection de ces

estimations par rapport à une initialisation standard de DMV, mais tout de même de moins bons résultats que l'injection de probabilités calculées sur les corpus arborés de référence.

Smith et Eisner [Smith et Eisner, 2006] exploitent leur constat que la très grande majorité des liens de dépendance sont locaux, et cherchent alors à favoriser les liens courts, en pénalisant les liens distants. Leur technique, qu'ils appellent « structural annealing », consiste à commencer par pénaliser fortement les liens distants en début d'apprentissage, puis au fur et à mesure du temps, à augmenter la tolérance à de plus grandes portées. En combinant cette idée avec une approche qu'ils avaient mise au point précédemment [Smith et Eisner, 2005], « Contrastive Estimation », qui consiste, dans la même optique que Spitkovsky, à modifier régulièrement la fonction qu'EM doit maximiser, Smith et Eisner améliorent considérablement les scores pour 4 langues sur les 6 testées.

Comme nous le noterons dans le chapitre 4, une des limites de ce modèle DMV est une sensibilité extrême au paramétrage initial, au point que les expériences menées dans les articles précédemment cités ne sont pas reproductibles, malgré la mise à disposition du code utilisé ([Cohen et Smith, 2010 ; Cohen *et al.*, 2011] et <https://code.google.com/p/jdageem/>). L'apprentissage ne convergeant pas, le choix du nombre d'itérations de la procédure EM est laissé à l'utilisateur sans qu'il existe d'heuristique qui guiderait ce choix. Les performances annoncées dans les différentes langues doivent être alors considérées comme un résultat intéressant en soi, mais pas comme un gage de qualité sur de nouvelles langues sans un travail approfondi d'adaptation. Signalons également que les algorithmes utilisés sont gourmands en mémoire, et peuvent prendre beaucoup de temps. Ce sont des raisons pour lesquelles l'apprentissage n'est jamais effectué sur un corpus intégral, mais sur un sous-corpus limité aux phrases de moins de 10 mots.

D'autres approches sont proposées dans la section suivante. Nous avons nous-même envisagé, puis effectué, et présenté dans cette thèse, de nouveaux travaux sur la question parce que nos contraintes et nos objectifs n'étaient pas compatibles avec l'approche de DMV. Nous cherchons, avec des textes nouveaux, grammaticalement étiquetés, dans des langues différentes, à obtenir rapidement et facilement une structuration sémantique des phrases. Le but est d'obtenir des arbres de dépendances de qualité suffisante pour en déduire les mots essentiels de la phrase et leurs relations, afin d'améliorer des procédures d'extraction d'information, pour obtenir le sens global du texte par des représentants chargés de sens. Le manque de souplesse et la lenteur de DMV nous ont poussé à développer une approche nouvelle et personnelle.

2.4 Autres approches

Brody [Brody, 2010] propose un modèle original basé sur l'alignement des mots. Son approche cherche à joindre les deux domaines que sont l'apprentissage non supervisé de dépendances et la traduction automatique. Chaque phrase est alignée avec elle-même, et au lieu de chercher des correspondances entre des groupes de mots dans les langues sources et cibles, on cherche des liens entre un mot et ses dépendants. Il réussit à faire la synthèse entre DMV pour l'ANSD et IBM [Brown *et al.*, 1993] pour la traduction automatique, en utilisant des caractéristiques communes aux deux modèles. Sans obtenir de très bons résultats, l'approche est néanmoins intéressante pour les perspectives qu'elle engendre, à savoir la transversalité de son modèle qui utilise les progrès dans un domaine et les réinvestit dans l'autre.

Søgaard [Søgaard, 2011] considère un arbre de dépendances comme un ordre partiel des mots

de la phrase, par rapport à la centralité ou saillance des mots, c'est-à-dire leur importance sémantique dans la phrase. Il travaille au niveau du lexique et non des catégories grammaticales des mots. Sa méthode nécessite un pré-apprentissage pour définir des clusters de mots lexicaux plus ou moins saillants. En revanche, les phrases à analyser a posteriori n'ont pas besoin d'être étiquetées, et deux algorithmes efficaces successifs permettent l'analyse, en $O(n \log(n))$ puis en $O(n^2)$, ce qui est plus rapide que DMV. Les résultats qu'il obtient sont comparables à ceux de DMV, légèrement inférieurs en moyenne, mais supérieurs pour 3 langues sur les 12 testées.

Headden, Johnson et McClosky [Headden III *et al.*, 2009] proposent un modèle différent, mais inspiré de DMV. Ils tiennent plus compte des liens de parenté entre catégories, tout en continuant de considérer la valence, c'est-à-dire le nombre d'arguments accepté par une catégorie. Dans DMV, la valence est considérée dans la distribution θ_S . Leur modèle EVG (pour Extended Valence Grammar) peut être enrichi (L-EVG pour Lexicalized EVG) par des informations lexicales. L'apport de ces informations complexifie le modèle et induit des problèmes de disparité de données, pris en charge par un lissage. Ils obtiennent de meilleurs résultats que leurs concurrents, mais ont seulement testé leur méthode sur la langue anglaise.

2.5 Notre approche

Dans la lignée principale de l'apprentissage non supervisé de dépendances, nous choisissons d'utiliser l'étiquetage syntaxique pour l'apprentissage puis l'analyse en dépendance, celui-ci étant généralement bien obtenu par des étiqueteurs automatiques. Nous choisissons aussi de ne pas exploiter l'information lexicale : trop disparate, elle alourdirait considérablement tout modèle visant la rapidité et l'efficacité.

Comme nous l'avons évoqué à la fin de la section 2.3, DMV et ses différentes versions ne correspondent pas à nos besoins et nos exigences. Nous construisons alors un nouveau modèle, situé dans le formalisme des grammaires hors contexte, qui utilise des outils existants, reconnus, efficaces, et régulièrement améliorés par une large communauté de chercheurs. La simplicité de notre modèle et la performance des outils impliqués permet de construire, à faible coût en temps et en mémoire, des arbres de dépendances de qualité correcte, avec une prise en main très simple et très rapide pour l'adaptation à une nouvelle langue.

Apprentissage non supervisé de dépendances par la méthode DGdg

Le chapitre précédent décrit le cadre de l'ANSD, les modèles proposés permettent de se déga-
ger de la lourde contrainte de la construction manuelle de corpus d'arbres de dépendances. Nous
avons cependant signalé la sensibilité des paramétrages et l'importance de l'apport de connais-
sances linguistiques pour obtenir des performances correctes.

Ce chapitre présente notre contribution. L'originalité de notre approche par rapport à nos pré-
décesseurs consiste à formaliser une famille de grammaires hors contexte permettant d'apprendre
de façon non supervisée des dépendances, différemment de nos prédécesseurs. Le but est d'obtenir
des arbres de qualité suffisante par une méthode plus rapide et plus adaptable que DMV. Avant de
présenter ces grammaires, les procédures d'apprentissage et d'analyse, nous rappelons différentes
définitions et présentons les algorithmes préexistants que nous utilisons (Inside-Outside et CYK).

3.1 Rappels sur les grammaires formelles de Chomsky

3.1.1 Grammaires hors contexte et forme normale de Chomsky

Définition et notations

Les notations et définitions que nous utilisons sont issues de l'ouvrage de Gonzalez et Thoma-
son [Gonzalez et Thomason, 1978].

Une grammaire G , suivant le formalisme de Chomsky, est un quadruplet $G = (N, \Sigma, P, S)$. N dé-
signe un ensemble de symboles non-terminaux, Σ un ensemble de symboles terminaux, P des
règles de production de G et S le non-terminal d'initialisation.

Le langage généré par la grammaire G est l'ensemble des chaînes σ de terminaux de Σ obtenues
par dérivation du symbole initial S par les règles de production de P . On note $S \Rightarrow^* \sigma$.

Deux grammaires sont dites équivalentes lorsqu'elles génèrent le même langage.

Hierarchie des types de grammaires

La hiérarchie de Chomsky permet de classer les différents types de grammaires, de la moins contraignante (type 0) à la plus restrictive (type 3), en fonction de la forme des règles de production P . Cette hiérarchie est présentée dans la Table 3.1.

Type	Appellation	Règles autorisées
0	Grammaires sans restriction	$\theta \rightarrow \rho$
1	Grammaires contextuelles	$\theta A \delta \rightarrow \theta \rho \delta$
2	Grammaires hors contexte (ou non contextuelles ou encore algébriques)	$A \rightarrow \rho$
3	Grammaires régulières	$A \rightarrow a$ et $A \rightarrow aB$

TABLE 3.1 : **Types de grammaires.** Nous notons par des lettres majuscules (A, B) les symboles non-terminaux, par des lettres minuscules les symboles terminaux (a) et par des lettres grecques les chaînes comprenant des terminaux et/ou des non-terminaux (δ, θ, ρ).

Les grammaires de types 0 et 1 ne sont, en pratique, pas utilisables. Nous ne les envisageons pas dans nos travaux qui visent la simplicité et l'adaptabilité. Les grammaires de type 3 étant un cas particulier des grammaires de type 2, nous ne les considérons pas véritablement mais pouvons les retrouver parmi les grammaires hors contexte. Nous nous concentrons ainsi sur le formalisme des grammaires hors contexte dans la suite.

Exemple 3.1 : grammaire hors contexte

On considère $G = (\{S, A, B\}, \{a, b\}, P, S)$ avec $P = \{S \rightarrow BA, A \rightarrow a, A \rightarrow abABa, B \rightarrow b\}$

Une dérivation possible d'une chaîne générable par G :

$$S \Rightarrow BA \Rightarrow bA \Rightarrow babABa \Rightarrow babaBa \Rightarrow bababa. \quad (3.1)$$

Le langage généré par cette grammaire est l'ensemble des chaînes $(ba)^{2n+1}$, $n \in \mathbb{N}$.

Remarque 3.1 : notation

On note $S \xRightarrow{*} bababa$ une chaîne de dérivation impliquant l'utilisation successive de plusieurs règles de production.

Réécriture sous forme normale de Chomsky

Les grammaires hors contexte peuvent être réécrites sous des formes standards en conservant le langage qu'elles génèrent. La forme normale de Chomsky nous intéresse particulièrement car elle est exigée dans deux algorithmes que nous utilisons : l'algorithme de d'analyse syntaxique CYK [Kasami, 1965] et l'algorithme d'apprentissage Inside-Outside [Lari et Young, 1990].

Une grammaire hors contexte est sous forme normale de Chomsky lorsque ses règles de production sont d'une des deux formes suivantes :

Forme 1 : $A \rightarrow BC$ avec $(A, B, C) \in N^3$

ou

Forme 2 : $A \rightarrow a$ avec $A \in N$ et $a \in \Sigma$.

Toute grammaire hors contexte peut se mettre sous forme normale de Chomsky, moyennant l'ajout de non-terminaux et modification des règles. Ceci est démontré de manière constructive dans [Gonzalez et Thomason, 1978]. L'exemple suivant montre une réécriture se basant sur la procédure proposée dans cette démonstration.

Exemple 3.2 : mise en forme normale de Chomsky

Soit G la grammaire de l'exemple 3.1.

$G = (\{S, A, B\}, \{a, b\}, P, S)$ avec $P = \{S \rightarrow BA, A \rightarrow a, A \rightarrow abABa, B \rightarrow b\}$

Nous conservons dans la grammaire modifiée G' les non-terminaux utilisés $\{S, A, B\}$ et les règles qui ont déjà la forme requise $\{S \rightarrow BA, A \rightarrow a, B \rightarrow b\}$.

L'unique règle à modifier est : $A \rightarrow abABa$.

Selon la procédure utilisée dans la démonstration, nous substituons à cette règle 6 règles en créant des non-terminaux intermédiaires Y .

$A \rightarrow Y_a Y_{bABa} ; Y_{bABa} \rightarrow Y_b Y_{ABa} ; Y_{ABa} \rightarrow A Y_{Ba} ; Y_{Ba} \rightarrow B Y_a ; Y_a \rightarrow a ; Y_b \rightarrow b$.

$G' = (\{S, A, B, Y_a, Y_b, Y_{ABa}\}, \{a, b\}, P', S)$ avec $P' = \{S \rightarrow BA, A \rightarrow a, B \rightarrow b, A \rightarrow Y_a Y_{bABa} ; Y_{bABa} \rightarrow Y_b Y_{ABa} ; Y_{ABa} \rightarrow A Y_{Ba} ; Y_{Ba} \rightarrow B Y_a ; Y_a \rightarrow a ; Y_b \rightarrow b\}$. G' est sous forme normale de Chomsky et équivalente à G .

3.1.2 Grammaires hors contexte probabilistes

Une grammaire stochastique (probabiliste) est un quintuplet, une grammaire non probabiliste et une distribution D . Ce sont les poids des probabilités des règles de production de P .

$$G_S = (N, \Sigma, D, P, S)$$

On ne considèrera que des grammaires hors contexte non restrictives, c'est-à-dire des grammaires pour lesquelles on suppose que toute application de règle de production de P est indépendante de la séquence de règles précédemment appliquées dans la dérivation.

Pour les grammaires hors contexte, la somme des probabilités des règles dérivant un même non-terminal doit être égale à 1.

La probabilité d'une dérivation est le produit des probabilités des règles utilisées dans la dérivation.

La probabilité d'une chaîne du langage générée par une grammaire est la somme des probabilités de toutes les dérivations possibles aboutissant à cette chaîne.

Exemple 3.3 : dérivation probabiliste

On enrichit la grammaire G de l'exemple 3.1 par deux nouveaux non-terminaux, C et D , trois nouvelles règles et une distribution de probabilités sur toutes les règles. Soit G_S cette grammaire stochastique.

$G_S = (\{S, A, B, C, D\}, \{a, b\}, D, P, S)$ avec les règles probabilisées par $D : 0.6 : S \rightarrow BA, 0.4 : S \rightarrow CD, 0.7 : A \rightarrow a, 0.3 : A \rightarrow abABa, 1 : B \rightarrow b, 1 : C \rightarrow bab, 1 : D \rightarrow aba$.

La probabilité de la dérivation de l'exemple 3.1 vaut alors $p_1 = 0.6 \times 1 \times 0.3 \times 0.7 \times 1 = 0.126$ car ont été successivement utilisées les règles suivantes : $0.6 : S \rightarrow BA, 1 : B \rightarrow b, 0.3 : A \rightarrow abABa, 0.7 : A \rightarrow a$ et $1 : B \rightarrow b$.

Cette grammaire G_S est ambiguë, i.e. elle permet d'aboutir à une même chaîne de terminaux par plusieurs ensembles différents de règles. C'est pourquoi la probabilité de la chaîne *bababa* est égale à la somme des probabilités de la dérivation présentée dans l'exemple 3.1 (p_1) et de l'autre dérivation possible : $S \Rightarrow CD \Rightarrow babD \Rightarrow bababa, p_2 = 0.4 \times 1 \times 1$.

$$p(bababa) = 0.126 + 0.4 = 0.526$$

3.2 Algorithmes CYK et Inside Outside

Les deux algorithmes présentés ici sont utilisés dans nos travaux. CYK permet l'analyse, c'est-à-dire étant donnée une chaîne de terminaux, la recherche de l'existence d'une chaîne de dérivation dans le cadre d'une grammaire. Lorsqu'il en existe plusieurs (en pratique, c'est toujours le cas dans notre étude), CYK probabiliste propose la chaîne la plus probable. Inside-Outside apprend par estimations successives la distribution D des probabilités des règles données par l'utilisateur, de manière à maximiser la vraisemblance d'un ensemble de chaînes de terminaux (corpus d'apprentissage) données elles aussi par l'utilisateur. La vraisemblance est ici définie comme la probabilité que le corpus d'apprentissage soit généré par la grammaire probabiliste considérée.

3.2.1 CYK et CYK probabiliste

L'algorithme CYK (**Cocke-Younger-Kasami**) est un analyseur syntaxique. Kasami [Kasami, 1965] a publié les premiers travaux sur ce sujet ; des versions probabilistes ont ensuite été déclinées. Nous présentons le principe et un exemple inspiré de l'ouvrage de Jurafsky [Jurafsky et Martin, 2009].

Soit G une grammaire hors contexte, $G = (N, \Sigma, P, S)$. On note en majuscules les non-terminaux de N et en minuscules les terminaux de Σ . On considère $G_S = (N, \Sigma, P, D, S)$ une grammaire probabiliste associée.

Soit la séquence de n symboles terminaux à analyser. À partir de cette séquence, on construit une demi-table de n colonnes (indiquées par i) et n lignes (indiquées par j).

Cette table d'analyse est construite de manière bottom-up en partant de la séquence d'observation des terminaux. Des non-terminaux sont inscrits successivement dans les cellules, ligne par ligne, colonne par colonne. Une fois atteint le symbole S au sommet de la table, l'algorithme analyse l'ensemble des dérivations générées pour retenir la plus probable.

Nous présentons cet algorithme avant de l'appliquer sur un exemple.

L'algorithme CYK détermine si la chaîne « $a_1 a_2 \dots a_n$ » est générable par la grammaire G . La version probabiliste identifie la séquence de dérivation la plus probable autorisée par la grammaire G .

Les étapes de construction de la table sont détaillées à la suite.

Remarque 3.2 : CYK et forme normale de Chomsky

La grammaire hors contexte doit être sous forme normale de Chomsky pour que l'on puisse construire la table CYK.

Étape 1. Complétion de la première ligne.

Chaque colonne correspond à un symbole terminal. Lorsqu'il existe des règles de production $A_1 \rightarrow a_i, \dots, A_p \rightarrow a_i$ dans P , les non-terminaux $A_1, \dots, A_p$ sont inscrits dans la cellule $t_{i,1}$ (colonne i , ligne $j=1$). Si aucun non-terminal ne se dérive en a_i , la cellule est laissée vide et le processus est interrompu car il n'y aura aucune analyse possible pour toute la chaîne.

Dans la version probabiliste, on inscrit dans la cellule la probabilité de chaque règle utilisable.

Étape 2. Complétion de la deuxième ligne.

Les non-terminaux A_l sont inscrits dans $t_{i,2}$ si et seulement si il existe un entier k , $1 \leq k < j$ et des règles de production $A_l \rightarrow BC$ dans P avec B dans la cellule $t_{i,k}$ et C dans la cellule $t_{i+k,j-k}$ (pour cette étape, $j=2$). S'il n'en existe pas, la cellule est laissée vide, mais le processus continue car des analyses ne nécessitant pas cette cellule peuvent être possibles.

Dans la version probabiliste, le maximum des probabilités des séquences de règles utilisées générant " $a_i a_{i+1}$ " à partir de A_l est inscrit dans la cellule $t_{i,2}$, à côté de chaque non-terminal A_l . En d'autres termes, c'est la probabilité de la « meilleure » séquence de règles générant " $a_i a_{i+1}$ " à partir de A_l .

Étape 3. L'étape 2 est répétée pour chaque j en complétant toutes les cellules, ligne par ligne, colonne par colonne jusqu'à ce que la table soit entièrement remplie ($j = n$). La Figure 3.1 illustre le mécanisme de recherche de non-terminaux à associer dans les lignes inférieures.

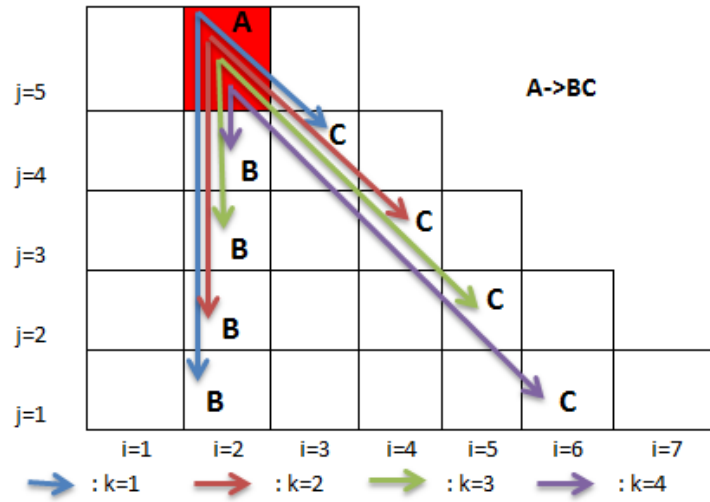


FIGURE 3.1 : Les différentes possibilités de cellules pour chercher une règle existante. Illustration de la recherche de règles de production $A \rightarrow BC$ avec B dans la cellule $t_{i,k}$ et C dans la cellule $t_{i+k,j-k}$, pour inscrire A dans la cellule $t_{i,j}$, $1 \leq k < j$. Un non-terminal A placé dans la cellule (i, j) génère la chaîne $a_i \dots a_{i+j-1}$. Les différentes dérivations à partir de A génèrent les paires de sous-chaînes contigües $a_i a_{i+1}$ et $a_{i+2} \dots a_{i+j-1}$ et/ou $a_i a_{i+1} a_{i+2}$ et $a_{i+3} \dots a_{i+j-1} \dots$ et/ou $a_i \dots a_{i+j-3}$ et $a_{i+j-2} a_{i+j-1}$ par l'intermédiaire des non-terminaux B et C.

La chaîne proposée est reconnue comme générée par la grammaire si et seulement si la cellule $t_{1,n}$ du sommet de la table contient le symbole initial S.

Dans la version probabiliste, on considère la plus grande probabilité associée au symbole S. Elle sera la probabilité de la séquence de règles la plus probable générant toute la chaîne $a_1 a_2 \dots a_n$.

Le pseudo-code proposé dans [Jurafsky et Martin, 2009] est repris dans l'algorithme 1.

Algorithme 1 : L'algorithme probabilisé CYK recherche la chaîne de dérivation la plus probable étant donnée une chaîne de mots (de nb_mots), et une grammaire hors contexte (de nb_regles règles) sous forme normale de Chomsky. Le tableau utilisé pour retrouver la meilleure analyse est *retro*.

```

Fonction CYK_PROBABILISTE(mots, grammaire)
    Créer et mettre à zéro  $\pi[nb\_mots, nb\_mots, nb\_non\_terminaux]$ 
    //  $\pi[colonne, ligne, indice \text{ du non-terminal}]$  stocke les probabilités associées à
    // chaque non-terminal dans chaque cellule de la table.
    # Cas  $j = 1$ 
    pour  $i = 1$  à  $nb\_mots$  faire
        pour chaque non-terminal  $A \in N$  faire
            si  $A \rightarrow m_i \in P$  alors
                // non-terminal et probabilité inscrits dans la
                // table lorsqu'il existe une règle de forme 1
                 $\pi[i, 1, A] = P(A \rightarrow m_i)$ 
    # Cas récursif,  $j \geq 2$ 
    pour  $j = 2$  à  $nb\_mots$  faire
        pour  $i = 1$  à  $nb\_mots - j + 1$  faire
            pour  $k = 1$  à  $j - 1$  faire
                pour  $A = 1$  à  $nb\_non\_terminaux$  faire
                    pour  $B = 1$  à  $nb\_non\_terminaux$  faire
                        pour  $C = 1$  à  $nb\_non\_terminaux$  faire
                             $prob = \pi[i, k, B] \times p[i + k, j - k, C] \times P(A \rightarrow BC)$ 
                            si  $prob > \pi[i, j, A]$  alors
                                // Probabilité maximale inscrite dans la cellule  $(i, j)$ 
                                 $\pi[i, j, A] = prob$  // Mémoire de la dérivation
                                 $retro[i, j, A] = \{k, A, B\}$ 
    retourner meilleure_analyse

```

Exemple 3.4 : fonctionnement de CYK.

Nous illustrons CYK sur l'exemple suivant.

Pour cet exemple particulier, les terminaux ne sont plus des lettres minuscules mais des mots. Quant aux non-terminaux, ils ne sont plus ici représentés par des majuscules uniques mais par des abréviations de catégories syntaxiques pour la majorité d'entre eux.

$G_S = (\{S, GN, GV, Det, Nom, X_1, Verb, X_2, GP, Prep\}, \{le, la, touriste, statue, cire, regarde, de\}, P, D, S)$

Avec les règles suivantes pour P , pondérées par leurs probabilités D et numérotées.

1 :	$S \rightarrow GN\ GV$	(1)	1 :	$X_2 \rightarrow GN\ GP$	(9)
0.6 :	$GN \rightarrow Det\ Nom$	(2)	0.8 :	$Det \rightarrow le$	(10)
0.4 :	$GN \rightarrow Det\ X_1$	(3)	0.2 :	$Det \rightarrow la$	(10')
0.2 :	$GV \rightarrow regarde$	(4)	0.3 :	$Nom \rightarrow touriste$	(11)
0.5 :	$GV \rightarrow Verb\ GN$	(5)	0.6 :	$Nom \rightarrow statue$	(11')
0.3 :	$GV \rightarrow Verb\ X_2$	(6)	0.1 :	$Nom \rightarrow cire$	(11'')
1 :	$GP \rightarrow Prep\ Nom$	(7)	1 :	$Verb \rightarrow regarde$	(12)
1 :	$X_1 \rightarrow Nom\ GP$	(8)	1 :	$Prep \rightarrow de$	(13)

La chaîne de mots à analyser est « le touriste regarde la statue de cire », que l'on note $x = a_1 a_2 a_3 a_4 a_5 a_6 a_7$.

La première ligne du tableau est complétée à partir des règles de forme 2, (10) à (13). Puis les lignes suivantes sont complétées à partir des règles de forme 1, (1) à (9). Voir Figure 3.2.

Deux chaînes de dérivation sont alors possibles, l'une apparaît en bleu, l'autre en rouge. Certaines parties sont communes aux deux dérivation.

Le symbole initial S s'affichant dans la cellule au sommet du tableau, la phrase est bien générée par la grammaire G .

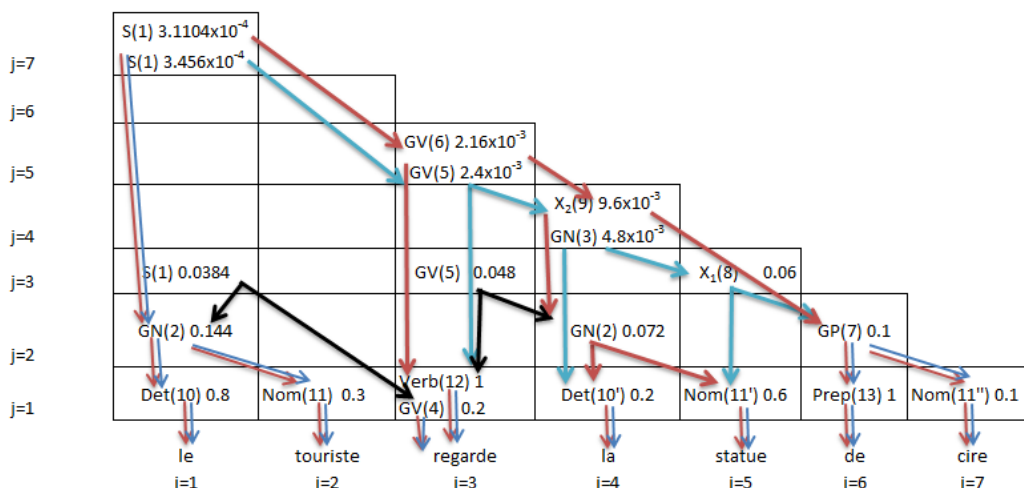


FIGURE 3.2 : Deux dérivation possibles après analyse par CYK.

Dans la figure 3.2, le non-terminal GV de la cellule (3,3) est placé par la règle (5) $GV \rightarrow Verb\ GN$, avec le non-terminal $Verb$ de la cellule (3,1) et le non-terminal GN de la

cellule (4,2) (flèche noire). La probabilité 0.048 est le produit de la probabilité de la règle (5) 0.5, par les probabilités des non-terminaux *Verb* de la cellule (3,1) 1 et *GN* de la cellule (4,2) 0.072.

La dérivation rouge utilise successivement les règles (1),(2), (10), (11), (6), (12), (9), (2), (10'), (11'), (7), (13) et (11'').

La probabilité de cette dérivation est le produit des probabilités des règles utilisées : $1 \times 0.6 \times 0.8 \times 0.3 \times 0.3 \times 1 \times 1 \times 0.6 \times 0.2 \times 0.6 \times 1 \times 1 \times 0.1 = 0.00031104 = 3.1104 \times 10^{-4}$.

La dérivation bleue : (1), (2), (10), (11), (5), (12), (3), (10'), (8), (11'), (7), (13) et (11'').

La probabilité de cette dérivation est le produit des probabilités des règles utilisées : $1 \times 0.6 \times 0.8 \times 0.3 \times 0.5 \times 1 \times 0.4 \times 0.2 \times 1 \times 0.6 \times 1 \times 1 \times 0.1 = 0.0003456 = 3.456 \times 10^{-4}$.

La dérivation bleue, reprise en Figure 3.3 est celle de plus haute probabilité, c'est cette analyse qui sera retenue par l'algorithme CYK probabiliste.

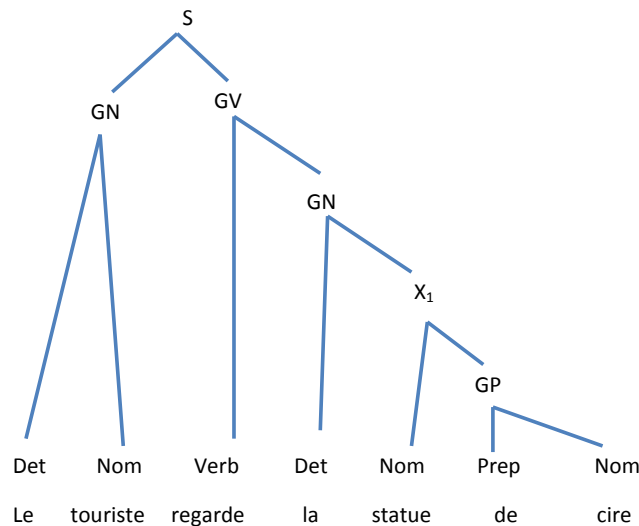


FIGURE 3.3 : La dérivation la plus probable choisie par CYK.

Les expériences réalisées dans cette thèse ont été faites à partir du code de Mark Johnson¹. Il intègre la possibilité d'écrire la grammaire sous un formalisme différent de la forme normale de Chomsky et effectue la conversion en interne.

3.2.2 Inside Outside

L'algorithme Inside-Outside a pour fonction d'apprendre les probabilités d'une grammaire hors-contexte à partir de chaînes de terminaux. Il fut introduit par Baker [Baker, 1979] et popula-

1. <http://web.science.mq.edu.au/~mjohnson/Software.htm>

risé par Lari et Young [Lari et Young, 1990]. Alors que les HMM (Hidden Markov Models) modélisent les grammaires probabilistes régulières, Inside-Outside est plus général et apprend les grammaires stochastiques hors contexte dont les règles de dérivation sont sous forme normale de Chomsky. C'est une procédure non supervisée de réestimation, par itérations successives, des probabilités des règles d'une grammaire hors contexte. Cet algorithme estime des fréquences de dérivations à partir des probabilités intermédiaires des règles et du corpus d'apprentissage afin de maximiser la probabilité que la grammaire ait généré ce corpus. Puis cet algorithme calcule des estimations des nouvelles probabilités des règles comme des rapports appropriés de ces fréquences attendues.

On parle ainsi de probabilités *inside* (ou *inner*) et *outside* (ou *outer*), car les calculs sont effectués sur les estimations des dérivations entre deux positions $s < t$ dans une chaîne de n terminaux $m = m_1 \dots m_n$. La probabilité *inner* correspond à la probabilité intérieure, pour un non-terminal A de dériver la sous chaîne $m_s \dots m_t$. La probabilité *outer* estime la probabilité de la dérivation extérieure $m_1 \dots m_{s-1} A m_{t+1} m_n$.

On suppose en premier lieu que le corpus d'apprentissage est constitué d'une unique séquence d'observations $m_1 \dots m_n$. Le cas réel d'un corpus multiséquentiel est une adaptation logique de ce premier cas simplifié [Lari et Young, 1990].

Soit $G_s = (N, \Sigma, D, P, S)$ la grammaire stochastique hors contexte à estimer en fonction de la chaîne de terminaux $m = m_1 \dots m_n$ (corpus d'apprentissage). On pose $N = \{A_i, i \in I \subset \mathbb{N}\}$, $\Sigma = \{a_l, l \in L \subset \mathbb{N}\}$, les règles de P sont de la forme $A_i \rightarrow A_j A_k$ et $A_i \rightarrow a_l$, avec $(A_i, A_j, A_k) \in N^3$ et $a_l \in \Sigma$.

Les matrices des paramètres qui décrivent cette grammaire stochastique hors contexte G_s sont M_1 pour les règles de forme 1 et M_2 pour les règles de forme 2.

$$M_1[i, j, k] = p(A_i \rightarrow A_j A_k / G_s)$$

où $M_1[i, j, k]$ désigne la probabilité de la règle $A_i \rightarrow A_j A_k$ étant donnée cette grammaire.

$$M_2[i, l] = p(A_i \rightarrow a_l / G_s)$$

La contrainte de cohérence sur la grammaire s'exprime par l'égalité suivante :

$$\sum_{j,k \in I} M_1[i, j, k] + \sum_{l \in L} M_2[i, l] = 1, \forall i \in I$$

Par analogie avec les HMM, le calcul de la vraisemblance du corpus d'apprentissage dans le cadre de la grammaire stochastique G_s est fractionné en plusieurs calculs. Pour cela sont définies les probabilités *inner* (e) et *outer* (f). La quantité $e(s, t, i)$ est la probabilité que la sous-séquence d'observations $m_s \dots m_t$ ait été générée par le non-terminal A_i :

$$e(s, t, i) = p(A_i \Rightarrow^* m_s \dots m_t)$$

Le calcul de e est récursif, et scindé en deux cas :

Pour $s = t$, $e(s, s, i) = p(A_i \Rightarrow m_s / G) = M_2[i, l]$, où l est défini par $m_s = a_l$.

Pour $s < t$, $e(s, t, i) = \sum_{j,k} \sum_{r=s}^{t-1} M_1[i, j, k] \times e(s, r, j) \times e(r+1, t, k)$, $\forall i$. La Figure 3.4 illustre ce schéma de récurrence.

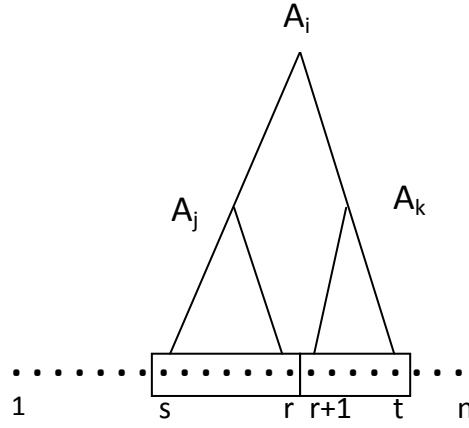


FIGURE 3.4 : Calcul des probabilités *inner*. Le calcul récursif par rapport à un non-terminal A_i , entre les deux positions s et t s'effectue en sommant sur les non-terminaux A_j et A_k , ainsi que sur un indice intermédiaire r , les produits des probabilités *inner* intermédiaires entre les positions s et r (couvertes par un A_j) et entre $r + 1$ et t (couvertes par un A_k).

La quantité $f(s, t, i)$ peut être vue comme la probabilité, au sein de la dérivation, que le non-terminal A_i soit généré, encadré par les sous séquences $m_1 \dots m_{s-1}$ par la gauche et $m_{t+1} \dots m_n$ par la droite (Figure 3.5) :

$$f(s, t, i) = p(S \xRightarrow{*} m_1 \dots m_{s-1} A_i m_{t+1} \dots m_n)$$

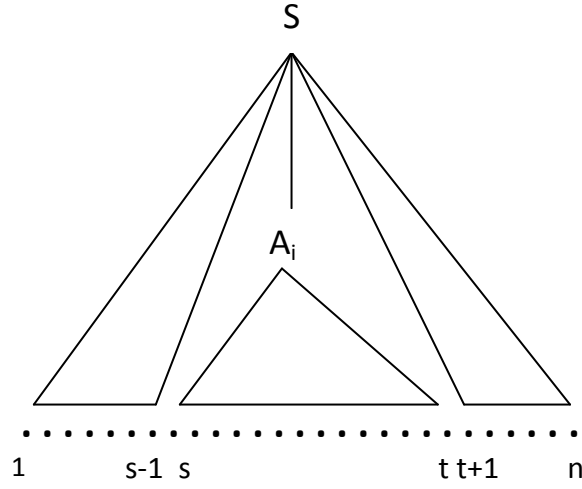


FIGURE 3.5 : Définition des probabilités *outer*. Étant donné le non-terminal A_i et les entiers s et t , la probabilité *outer* des positions $s \leq t$ est la probabilité que S dérive en $m_1 \dots m_{s-1} A_i m_{t+1} m_n$.

Suivant les cas, la règle utilisée est $A_j \rightarrow A_i A_k$ ou $A_j \rightarrow A_k A_i$, comme illustré par la Figure 3.6.

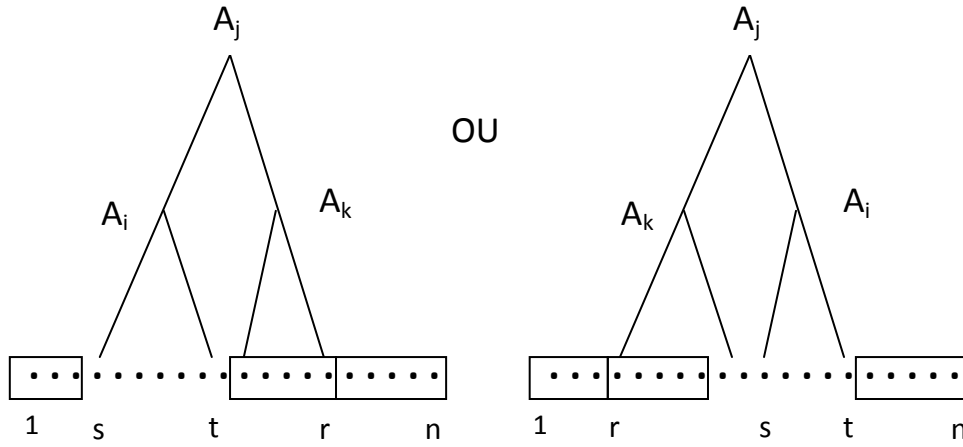


FIGURE 3.6 : Calcul des probabilités *outer*. Le calcul récursif par rapport à un non-terminal A_j en dehors de l'intervalle de positions $[s, r]$ se décompose en sommant sur un indice de position intermédiaire t , en utilisant les probabilités *outer* sur tous les non-terminaux A_i et A_k en dehors de l'intervalle $[s, t]$.

$$f(s, t, i) = \sum_{j,k} \left[\sum_{r=t+1}^n f(s, r, j) \times M_1[j, i, k] \times e(t+1, r, k) + \sum_{r=1}^{s-1} f(r, t, j) \times M_1[j, k, i] \times e(r, s-1, k) \right]$$

Le terme gauche de cette expression, $\sum_{r=t+1}^n f(s, r, j) \times M_1[j, i, k] \times e(t+1, r, k)$, correspond à la partie gauche de la figure 3.6 et le terme de droite $\sum_{r=1}^{s-1} f(r, t, j) \times M_1[j, k, i] \times e(r, s-1, k)$ au cas envisagé dans la partie droite de cette même figure.

$$\text{et } f(1, n, i) = \begin{cases} 1 & \text{si } A_i = S \\ 0 & \text{sinon} \end{cases}$$

Les probabilités *inner* sont donc calculées par un procédé bottom-up alors que les probabilités *outer* sont issues d'un calcul top-down.

La probabilité de la séquence complète d'observations, dans le cadre de la grammaire, est alors calculée en sommant le produit des quantités *inner* et *outer*.

$$p = p(S \Rightarrow^* m/G) = \sum_{i \in I} e(s, t, i) \times f(s, t, i)$$

Cette égalité est vérifiée $\forall s \leq t$. On choisit alors d'effectuer le calcul dans les cas particuliers $s = 1$ et $t = n$ (calcul n'impliquant que les probabilités *inner*) ou $s = t$ (probabilités *outer* et coefficients de M_2). En pratique, c'est le cas $s = 1$ et $t = n$ qui sera implémenté dans la procédure d'apprentissage.

$$p = p(S \Rightarrow^* m/G) = e(1, n, S) \quad (3.2)$$

car $f(1, n, i) = 0$ si $A_i \neq S$ et $= 1$ sinon donc $\sum_{i \in I} e(1, n, i) \times f(1, n, i) = e(1, n, S)$.

L'apprentissage, par itérations successives, réestime les coefficients des matrices M_1 et M_2 par les formules suivantes, obtenues par un raisonnement bayésien :

$$\hat{M}_1[i, j, k] = \frac{\frac{1}{p} \sum_{s=1}^{n-1} \sum_{t=s+1}^n \sum_{r=s}^{t-1} M_1[i, j, k] \times e(s, r, j) \times e(r+1, t, k) \times f(s, t, i)}{\frac{1}{p} \sum_{s=1}^n \sum_{t=s}^n e(s, t, i) \times f(s, t, i)}, \forall i, j, k \quad (3.3)$$

$$\hat{M}_2[i, l] = \frac{\frac{1}{p} \sum_t \text{tel que } m_t = a_l e(t, t, i) \times f(t, t, i)}{\frac{1}{p} \sum_{s=1}^n \sum_{t=s}^n e(s, t, i) \times f(s, t, i)}, \forall i, l \quad (3.4)$$

Nous rappelons que les résultats précédents sont valables dans le cas, non commun, d'une seule séquence d'observations. Pour un apprentissage fiable, plusieurs chaînes d'observations sont nécessaires. Les équations (3.3) et (3.4) sont adaptées en calculant au préalable des vraisemblances partielles de chaque séquence, les nouveaux coefficients $\hat{M}_1[i, j, k]$ et $\hat{M}_2[i, l]$ sont alors obtenus par moyenne de ces quantités intermédiaires.

L'algorithme Inside-Outside apprend les probabilités d'une grammaire stochastique hors contexte G_s en :

- choisissant des valeurs initiales judicieuses pour les coefficients des matrices M_1 et M_2 ;
- puis en calculant les nouveaux coefficients des matrices M_1 et M_2 par les adaptations aux ensembles de séquences des équations (3.3) et (3.4), puis la vraisemblance $p = \prod_{q=1}^Q p_q$ de la nouvelle grammaire par (3.2), p_q étant calculée pour chaque chaîne $m^{(q)}$;

- La différence des valeurs de p entre chaque itération décroît (la convergence de l'algorithme Inside-Outside est empiriquement reconnue, et démontrée par Prescher dans [Prescher, 2004]), la procédure s'interrompt lorsque la différence est inférieure au seuil déterminé par l'utilisateur.

La complexité de l'algorithme Inside-Outside est de l'ordre de $O(N^3)$, où N est le nombre de non-terminaux. Ceci est d'autant plus limitant que Lari et Young, dans leur article [Lari et Young, 1990], exposent leurs difficultés à apprendre une grammaire en adéquation avec le corpus lorsque le nombre de non-terminaux dépasse 8. De plus, cet algorithme est très sensible au choix des paramètres d'initialisation.

Ces limites sont des facteurs que nous avons pris en compte dans notre contribution, et qui ne se révéleront pas pénalisants par la suite.

Nous avons aussi noté, au cours des expérimentations, que le paramétrage initial n'influe que très peu sur nos grammaires finales.

3.3 Présentation de la famille de grammaires DGdg, les 5 variantes

Nous présentons dans cette section notre première contribution, à savoir l'élaboration de grammaires hors contexte permettant la construction d'arbres de dépendances, au moyen de rapports de domination entre les non-terminaux. De même que la plupart de nos prédécesseurs (cf. section 2), nous faisons l'hypothèse que les fonctions grammaticales seules, sans les unités lexicales, sont des informations suffisamment riches pour construire des arbres de dépendances de qualité. Notre méthode suppose donc un étiquetage préalable des textes.

3.3.1 ID/LP (Immediate Dominance/ Linear Precedence)

Les principes de la théorie ID/LP (Immediate Dominance/ Linear Precedence), présentée dans l'ouvrage de Gazdar [Gazdar *et al.*, 1985], nous ont inspiré pour la création d'une famille de grammaires hors contexte probabilistes.

Cette théorie a été mise en place pour compenser le manque de souplesse des grammaires hors contexte, en ce qui concerne l'ordre des mots. Inspirée par les langues makua (langues bantoues parlées au Mozambique, en Tanzanie et au Malawi), dans lesquelles l'ordre des mots est très changeant, moins figé que pour le français ou l'anglais, cette théorie ID/LP respecte la contrainte d'ECPO (Exhaustive Constant Partial Ordering). Ceci impose des ordres respectifs entre les catégories sans avoir à écrire toutes les règles possibles dans le cadre d'une grammaire hors contexte. Ainsi, le format ID/LP capture des généralisations significatives de règles a priori spécifiques d'une langue particulière.

Une grammaire de format ID/LP est constituée de 2 types de relations, qui ne se traduisent pas de façon entièrement compatible en des règles de grammaires hors contexte. Ces relations s'appliquent à des nœuds (équivalents aux non-terminaux des grammaires hors contexte) et imposent des contraintes ou acceptent des formats aux arbres formés de ces nœuds (ceci peut se traduire en règles, mais un ensemble de règles ne peut pas nécessairement se mettre sous cette forme de contraintes et de formats).

Une relation d'Immediate Dominance autorise un type de nœud à être le parent direct d'une « fratrie » de nœuds de types donnés, sans préciser l'ordre des nœuds descendants.

L'exemple donné dans [Gazdar *et al.*, 1985] est la règle ID suivante :

$$A \rightarrow B, C, D$$

Cette règle signifie que sont admis par cette grammaire ID/LP tous les arbres de la figure 3.7.

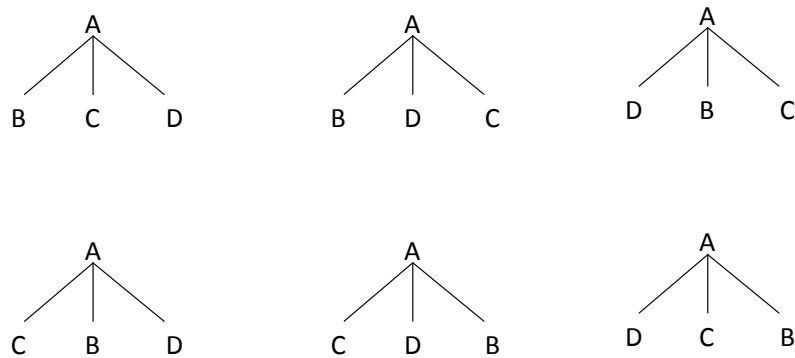


FIGURE 3.7 : Arbres admis par la règle $A \rightarrow B, C, D$ dans le format ID/LP. Figure de [Gazdar *et al.*, 1985].

Une déclaration de Linear Precedence contraint l'ordre de certains types de nœuds, lorsqu'ils se trouvent dans la même fratrie. La déclaration $B < C$ interdit à C d'être un frère gauche de B , même sans être contigu.

En conservant la règle ID $A \rightarrow B, C, D$ et en y ajoutant la déclaration LP $B < C$, le sous-ensemble des arbres de cette grammaire ID/LP, présentés en figure 3.7, se réduit aux trois arbres de la figure 3.8.

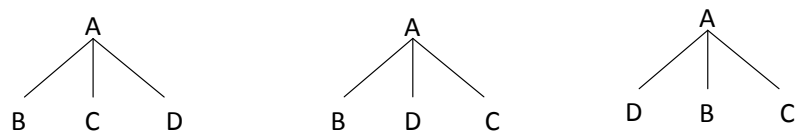


FIGURE 3.8 : Arbres admis par la règle ID $A \rightarrow B, C, D$ et la déclaration LP $B < C$ dans le format ID/LP. Figure de [Gazdar *et al.*, 1985].

Nous avons conservé de cette théorie les idées générales de contrainte de filiation directe et de placement relatif gauche/droite parmi les descendants immédiats d'un non-terminal. Mais souhai-

tant travailler sur les langues usuelles, dont l'ordre des mots reste régulier, nous n'avons pas adopté ce formalisme indépendant, depuis tombé en désuétude. Nous avons introduit ces principes dans le formalisme des grammaires hors contexte reconnues par nombre d'algorithmes efficaces d'apprentissage et de d'analyse.

3.3.2 Principes et définition de la version 4bin de la famille de grammaires DGdg

Nous proposons, en accord avec les principes de la théorie ID/LP, une première grammaire hors contexte. Quatre symboles non-terminaux sont introduits, en supplément du symbole obligatoire $S : D$ (pour DROITE), G (pour GAUCHE), d (pour droite) et g (pour gauche).

Deux principes, inspirés de ID/LP, ont guidé l'élaboration des règles :

- les symboles en majuscule couvrent des parties contigües de phrases dominant une partie couverte par un symbole en minuscule (principe ID) ;
- un symbole gauche (G ou g) est apparié par la gauche à un symbole droit (D ou d) (principe LP).

Nous avons respecté la contrainte de forme normale de Chomsky pour une bonne compatibilité avec les algorithmes CYK et Inside-Outside.

Les règles de forme 2 (définition page 27) sont, a priori, toutes incluses dans la grammaire. De chacun des 5 non-terminaux dérive chacune des catégories grammaticales du corpus étudié.

On désigne par n le nombre de catégories grammaticales distinctes présentes dans le corpus.

G_{4bin}	$(N_{4bin}, \Sigma, P_{4bin}, S)$
N_{4bin}	$\{S, D, G, d, g\}$
Σ_{4bin}	Σ , ensemble de catégories grammaticales adapté au corpus étudié. $ \Sigma = n$.
P_{4bin}	$\{S \rightarrow gD, S \rightarrow Gd, D \rightarrow gD, D \rightarrow Gd, G \rightarrow gD, G \rightarrow Gd, d \rightarrow gD, d \rightarrow Gd, g \rightarrow gD, g \rightarrow Gd, \forall (cat, nt) \in (\Sigma, N_{4bin}) nt \rightarrow cat\}$

La grammaire G_{4bin} contient $5n + 10$ règles.

Exemple 3.5 : arbres de dépendances et DGdg

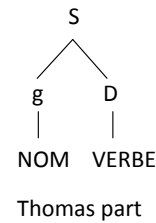
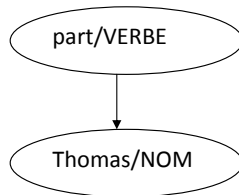


FIGURE 3.9 : Arbre de dépendances bare-bones pour la phrase de 2 mots « Thomas part ».

FIGURE 3.10 : Arbre DGdg pour la phrase de 2 mots « Thomas part ».

La Figure 3.10 illustre la construction d'un arbre DGdg très simple sur la séquence NOM VERBE dont la phrase « Thomas part » est une instantiation. Les règles utilisées sont $S \rightarrow gD$, $g \rightarrow NOM$, $D \rightarrow VERBE$. La Figure 3.9 donne l'arbre de dépendances correspondant.

Les dépendances DGdg entre les parties de phrases se traduisent, comme on peut le constater dans l'exemple 3.5, par des dépendances plus classiques, entre les mots (catégories grammaticales), de la phrase. Dans la sous-section 3.4.3 nous présentons cette procédure de traduction des arbres DGdg en arbres de dépendances, et donnons le détail de cette procédure en annexe. Nous démontrons aussi formellement en annexe l'équivalence entre ces deux modes de représentation.

Tout arbre de dépendances peut correspondre à plusieurs arbres DGdg.

Les différentes analyses DGdg ne mènent pas nécessairement toutes au même arbre de dépendances.

Pour départager ensuite les analyses DGdg, les règles de la grammaire seront probabilisées. Une fois que l'analyse DGdg la plus probable sera déterminée, on pourra alors en déduire l'arbre de dépendances choisi par la méthode DGdg.

Exemple 3.6 : arbres DGdg

Les Figures 3.11 à 3.14 montrent plusieurs analyses DGdg possibles menant au même arbre de dépendances 3.15.

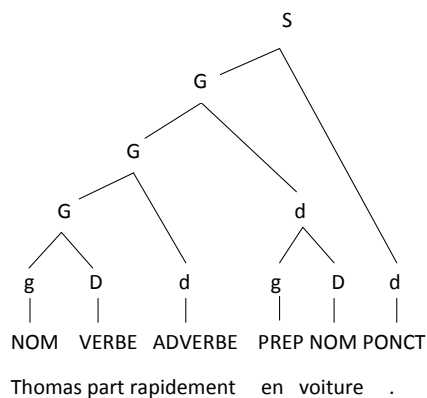


FIGURE 3.11 : Arbre DGdg 1 pour la phrase « Thomas part rapidement en voiture. ».

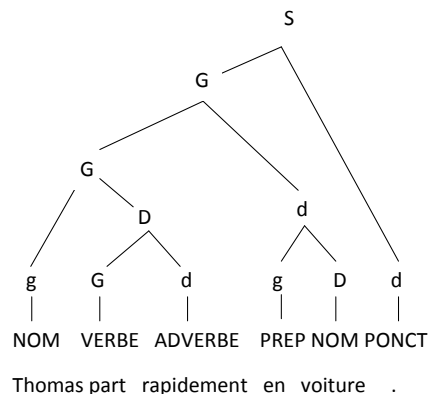


FIGURE 3.12 : Arbre DGdg 2 pour la phrase « Thomas part rapidement en voiture. ».

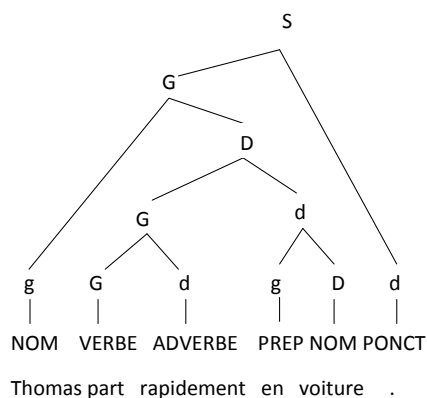


FIGURE 3.13 : Arbre DGdg 3 pour la phrase « Thomas part rapidement en voiture. ».

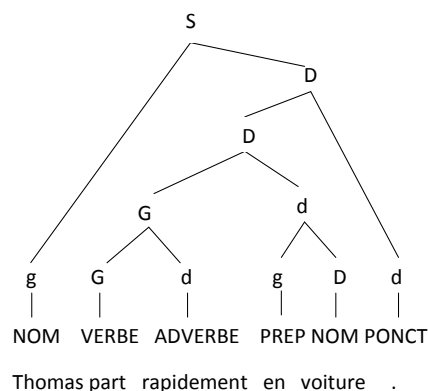


FIGURE 3.14 : Arbre DGdg 4 pour la phrase « Thomas part rapidement en voiture. ».

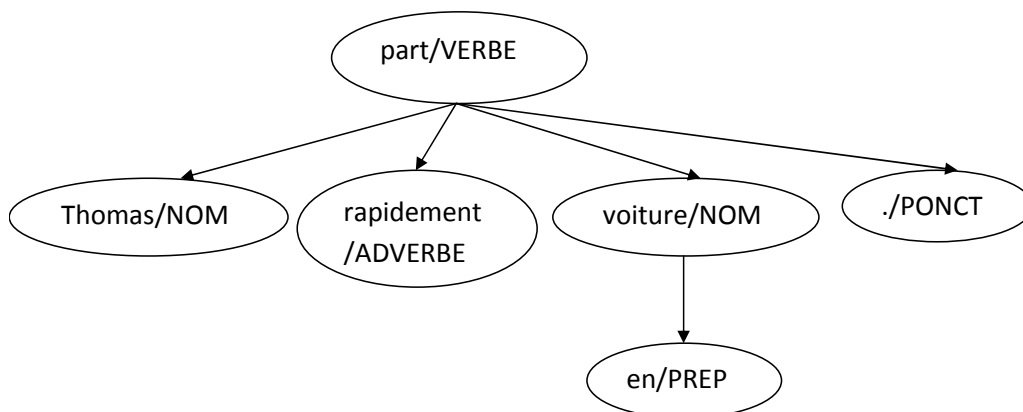


FIGURE 3.15 : Arbre de dépendances bare-bones pour la phrase « Thomas part rapidement en voiture. ».

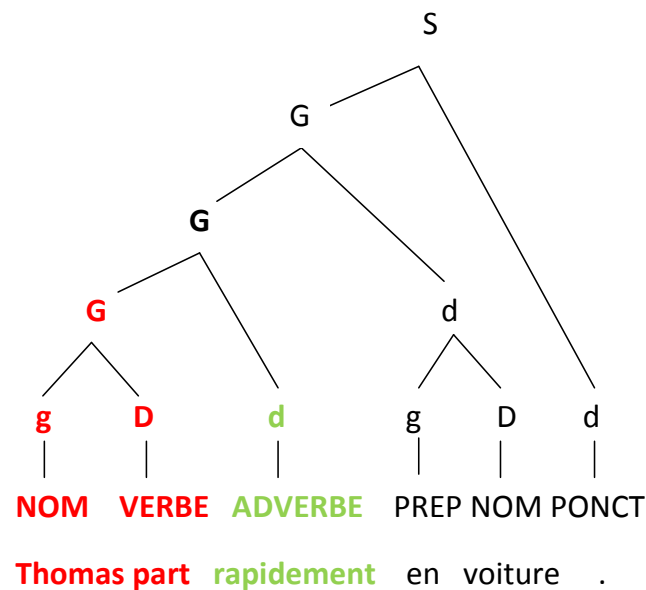


FIGURE 3.16 : Arbre DGdg, la partie « **Thomas part** » domine la partie « **rapidement** » par l'utilisation de la règle $G \rightarrow Gd$.

La figure 3.16 illustre l'interprétation des règles en termes de domination d'une partie de la phrase sur une autre partie de la phrase.

3.3.3 Variantes additionnelles

On constate sur le premier exemple de 3.5 que le partage binaire de la phrase « Thomas part » en un côté gauche dominé et un côté droit dominant est parfaitement judicieux. Mais pour une expression nominale du type « petit garçon blond », les deux adjectifs « petit » et « blond » sont dominés par le nom « garçon » de façon équivalente. Le rôle du dominant « garçon » est identique dans les deux cas, seule la position de chaque adjectif diffère.

Nous proposons, pour traiter plus convenablement ce type de structure, deux stratégies différentes.

La première stratégie consiste à supprimer l'orientation des symboles dominants. Les deux non-terminaux dominés orientés *d* et *g* sont conservés, mais on fusionne les deux symboles dominants *D* et *G* en un seul non-terminal neutre *N*. L'application d'une règle indique toujours les positions relatives du dominé par rapport au dominant, mais la « nature » gauche ou droite d'un dominant n'a plus de sens.

Ceci nous amène à définir une deuxième grammaire de la famille DGdg, la grammaire 3bin, ne contenant que 3 non-terminaux en plus du symbole *S*.

G_{3bin}	$(N_{3bin}, \Sigma, P_{3bin}, S)$
N_{3bin}	$\{S, \mathbf{N}, d, g\}$
Σ_{3bin}	Σ , ensemble de catégories grammaticales adapté au corpus étudié. $ \Sigma = n$.
P_{3bin}	$\{S \rightarrow gN, S \rightarrow Nd, N \rightarrow gN, N \rightarrow Nd, d \rightarrow gN, d \rightarrow Nd, g \rightarrow gN, g \rightarrow Nd, \forall (cat, nt) \in (\Sigma, N_{4bin}) nt \rightarrow cat\}$

Soit n le nombre de catégories grammaticales dans le corpus, la grammaire G_{3bin} contient $4n+8$ règles.

La seconde stratégie amène à contourner la contrainte de la forme normale de Chomsky. Puisque toute grammaire hors contexte peut se réécrire sous cette forme, nous avons autorisé des règles ternaires qui seront ensuite interprétées comme plusieurs règles binaires. L'écriture sous forme ternaire permet une interprétation directe et compréhensible des arbres de dérivation, tout en ne s'éloignant que raisonnablement de la contrainte de forme normale de Chomsky.

Ceci nous a mené à construire la grammaire 4ter, qui comprend les règles de structure de la grammaire 4bin, complétées des règles ternaires comportant un non-terminal dominant au centre.

G_{4ter}	$(N_{4ter}, \Sigma, P_{4ter}, S)$
N_{4ter}	$\{S, D, G, d, g\}$
Σ_{4ter}	Σ , ensemble de catégories grammaticales adapté au corpus étudié. $ \Sigma = n$.
P_{4ter}	$\{S \rightarrow gD, S \rightarrow Gd, \mathbf{S} \rightarrow \mathbf{gDd}, \mathbf{S} \rightarrow \mathbf{gGd}, D \rightarrow gD, D \rightarrow Gd, \mathbf{D} \rightarrow \mathbf{gDd}, \mathbf{D} \rightarrow \mathbf{gGd}, G \rightarrow gD, G \rightarrow Gd, \mathbf{G} \rightarrow \mathbf{gDd}, \mathbf{G} \rightarrow \mathbf{gGd}, d \rightarrow gD, d \rightarrow Gd, \mathbf{d} \rightarrow \mathbf{gDd}, \mathbf{d} \rightarrow \mathbf{gGd}, g \rightarrow gD, g \rightarrow Gd, \mathbf{g} \rightarrow \mathbf{gDd}, \mathbf{g} \rightarrow \mathbf{gGd}, \forall (cat, nt) \in (\Sigma, N_{4ter}) nt \rightarrow cat\}$

Pour une structure plus riche et concise, nous proposons de combiner les deux stratégies. On conserve les règles binaires avec des dominants orientés de 4bin. On ajoute des règles ternaires, de dominant central neutre. Ceci compose la grammaire 5ter.

G_{5ter}	$(N_{5ter}, \Sigma, P_{5ter}, S)$
N_{5ter}	$\{S, D, G, \mathbf{N}, d, g\}$
Σ_{5ter}	Σ , ensemble de catégories grammaticales adapté au corpus étudié. $ \Sigma = n$.
P_{5ter}	$\{S \rightarrow gD, S \rightarrow Gd, \mathbf{S} \rightarrow \mathbf{gNd}, D \rightarrow gD, D \rightarrow Gd, \mathbf{D} \rightarrow \mathbf{gNd}, G \rightarrow gD, G \rightarrow Gd, \mathbf{G} \rightarrow \mathbf{gNd}, d \rightarrow gD, d \rightarrow Gd, \mathbf{d} \rightarrow \mathbf{gNd}, g \rightarrow Gd, \mathbf{g} \rightarrow \mathbf{gNd}, \forall (cat, nt) \in (\Sigma, N_{5ter}) nt \rightarrow cat\}$

La dernière version enrichit la grammaire 5ter d'un aspect récursif dans la dérivation ternaire du symbole dominant neutre. Ceci est illustré dans la Figure 3.22, les règles ternaires peuvent successivement s'appliquer, comme dans la version 4ter, mais seulement dans l'optique d'un symbole central neutre. Nous avons appelé cette dernière version 5ter+.

G_{5ter+}	$(N_{5ter+}, \Sigma, P_{5ter+}, S)$
N_{5ter+}	$\{S, D, G, \mathbf{N}, d, g\}$
Σ_{5ter+}	Σ , ensemble de catégories grammaticales adapté au corpus étudié. $ \Sigma = n$.
P_{5ter+}	$\{S \rightarrow gD, S \rightarrow Gd, \mathbf{S} \rightarrow \mathbf{gNd}, D \rightarrow gD, D \rightarrow Gd, \mathbf{D} \rightarrow \mathbf{gNd},$ $G \rightarrow gD, G \rightarrow Gd, \mathbf{G} \rightarrow \mathbf{gNd}, \mathbf{N} \rightarrow \mathbf{gD}, \mathbf{N} \rightarrow \mathbf{Gd}, \mathbf{N} \rightarrow \mathbf{gNd},$ $d \rightarrow gD, d \rightarrow Gd, \mathbf{d} \rightarrow \mathbf{gNd}, g \rightarrow Gd, \mathbf{g} \rightarrow \mathbf{gNd},$ $\forall (cat, nt) \in (\Sigma, N_{5ter}) \ nt \rightarrow cat\}$

La Table 3.2 reprend les caractéristiques des 5 grammaires de la famille DGdg. Les Figures 3.18 à 3.22 montrent des arbres de dérivation dans les 5 grammaires, de la phrase « Thomas est parti rapidement en voiture. ». L'arbre de dépendances bare-bones, construit à la main suivant les règles les plus classiques du French Treebank [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010], de cette phrase (contenant un mot de plus que la phrase hiérarchisée dans la Figure 3.15) est donné par la figure 3.17.

	4bin	3bin	4ter	5ter	5ter+
Les différences principales entre les variantes	Grammaire d'origine	Le dominant n'est pas dirigé à gauche ou à droite	Des règles ternaires sont autorisées	4ter + non-terminal dominant centré N	5ter + règles récursives pour N
Les règles de structure (forme 1 + ternaires)	$nt \rightarrow G d,$ $nt \rightarrow g D,$	$nt \rightarrow N d,$ $nt \rightarrow g N,$	$nt \rightarrow G d,$ $nt \rightarrow g D,$ $nt \rightarrow g G d,$ $nt \rightarrow g D d,$	$nt(\neq N) \rightarrow G d,$ $nt(\neq N) \rightarrow g D,$ $nt(\neq N) \rightarrow g N d$	$nt \rightarrow G d,$ $nt \rightarrow g D,$ $nt \rightarrow g N d,$
non-terminaux	S, D, G, d, g	S, N, d, g	S, D, G, d, g	S, D, N, G, d, g	S, D, N, G, d, g
# règles (pour n cat.)	$4n + 8$	$5n + 10$	$5n + 20$	$6n + 15$	$6n + 18$

TABLE 3.2 : Synthèse des variantes de la famille de grammaires hors contexte DGdg (nt signifie non-terminal).

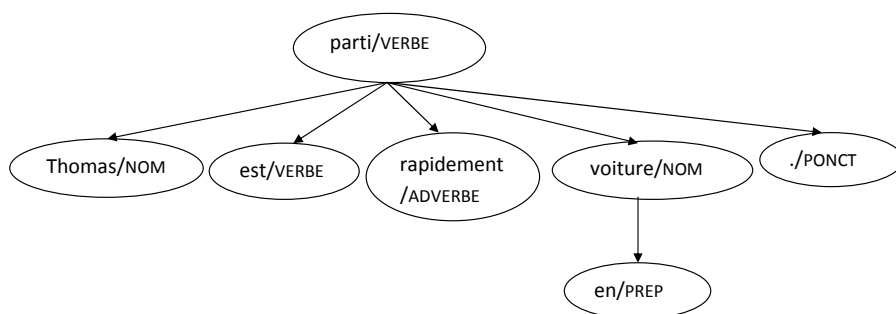


FIGURE 3.17 : Arbre de dépendances bare-bones pour la phrase « Thomas est parti rapidement en voiture. ».

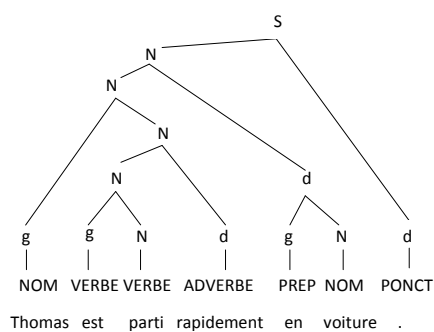


FIGURE 3.18 : Un arbre 3bin pour la phrase « Thomas est parti rapidement en voiture. » se traduisant en l'arbre de dépendances bare-bones de la Figure 3.17.

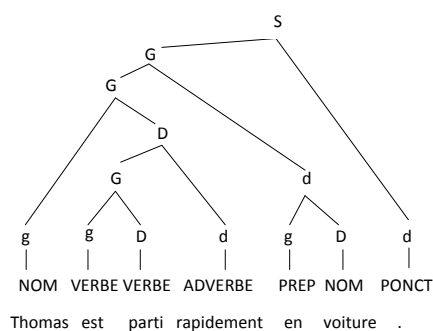


FIGURE 3.19 : Un arbre 4bin pour la phrase « Thomas est parti rapidement en voiture. » se traduisant en l'arbre de dépendances bare-bones de la Figure 3.17.

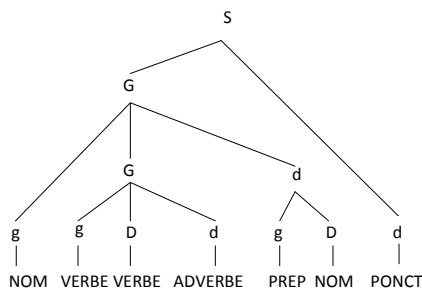


FIGURE 3.20 : Un arbre 4ter pour la phrase « Thomas est parti rapidement en voiture. » se traduisant en l'arbre de dépendances bare-bones de la Figure 3.17.

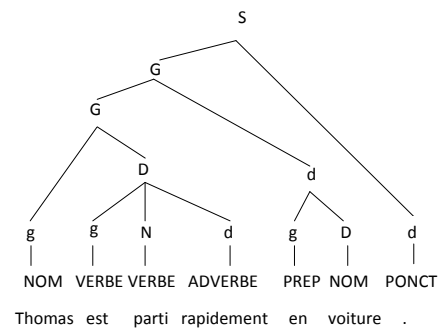


FIGURE 3.21 : Un arbre 5ter pour la phrase « Thomas est parti rapidement en voiture. » se traduisant en l'arbre de dépendances bare-bones de la Figure 3.17.

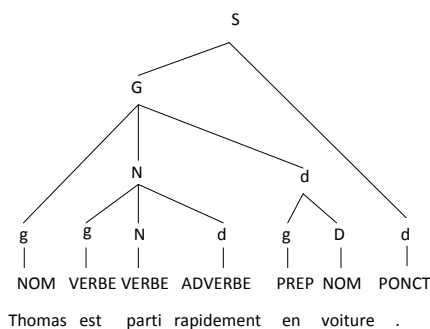


FIGURE 3.22 : Un arbre 5ter+ pour la phrase « Thomas est parti rapidement en voiture. » se traduisant en l'arbre de dépendances bare-bones de la Figure 3.17.

Nous avons introduit une nouvelle famille de grammaires, nous allons dans la section suivante montrer de quelle façon des poids sont attribués aux règles, afin de rendre ces grammaires stochastiques. Nous allons, après avoir présenté ce processus d'apprentissage, indiquer comment les arbres de dépendances sont construits durant la phase d'analyse.

3.4 Intégralité du processus d'apprentissage et d'analyse

3.4.1 Apprentissage des poids des règles des grammaires de la famille DGdg

Principes généraux

L'apprentissage d'une grammaire de la famille DGdg consiste à estimer la pondération de chaque règle, à partir d'un corpus d'apprentissage, de manière à optimiser la vraisemblance du corpus.

Dans notre étude, un corpus d'apprentissage est constitué d'un ensemble de phrases (grammaticalement étiquetées), lues par l'algorithme d'apprentissage comme un ensemble de séquences de catégories grammaticales (ordonnées).

Par exemple les phrases de la Table 3.3 seront données sous la forme de la Table 3.4 à l'algorithme d'apprentissage.

Le	taux	de	pénétration	des	marques	étrangères	a	continué	à	progresser.
Des	conventions	sur_ce_point	pourraient	être	signées	avec	EDF	.		
Une	aide	de	700	millions	de	dollars				

TABLE 3.3 : Des exemples de phrases du corpus d'apprentissage français [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010].

DET	NOM	PRP	NOM	PRP+DET	NOM	ADJ	VER	VER	PRP	VER	PONCT
DET	NOM	ADV	VER	VER	VER	PRP	NOM	PONCT			
DET	NOM	PRP	DET	NOM	PRP	NOM					

TABLE 3.4 : Les séquences de catégories grammaticales correspondant aux phrases de la Table 3.3, données d'entrée de la procédure d'apprentissage (corpus d'apprentissage français, [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010]).

Nous utilisons l'algorithme d'apprentissage Inside-Outside dont les principes ont été rappelés en partie 3.2.2. Étant donné : un corpus d'apprentissage, une grammaire (par exemple 4bin) et une pondération initiale des règles D , l'algorithme Inside-Outside modifiera les poids de D à chaque itération, de façon à maximiser la vraisemblance du corpus par la grammaire stochastique (équation 3.2).

La Figure 3.23 illustre cette partie apprentissage de la procédure globale.

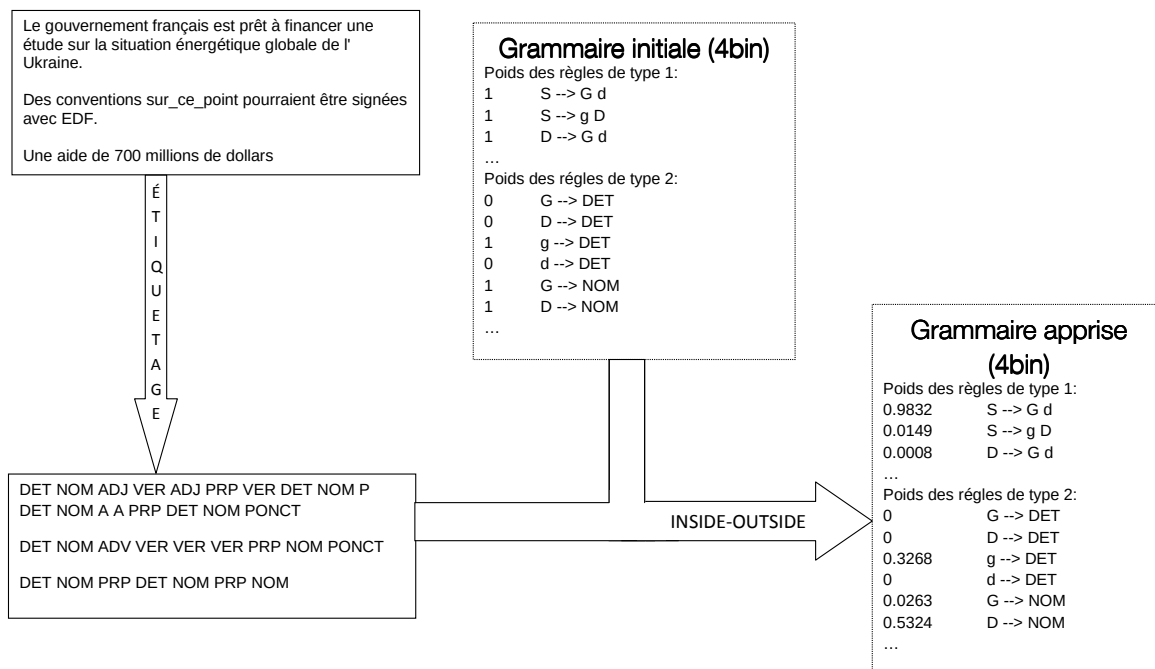


FIGURE 3.23 : Procédure d'apprentissage par Inside-Outside à partir d'un corpus textuel et d'une grammaire initialisée.

L'initialisation de la grammaire

La phase d'initialisation, préliminaire à l'apprentissage par Inside-Outside, consiste seulement à sélectionner, parmi les règles de forme 2 de la grammaire DGdg, les règles pertinentes par rapport à la langue et au corpus d'apprentissage. Nous pourrions aussi donner des probabilités initiales, mais nous avons fait le choix d'une initialisation simple et non experte. Par exemple, en français, les déterminants sont toujours dépendants d'un nom à gauche duquel ils sont situés. Les déterminants ne sont la tête d'aucune catégorie. L'utilisateur précise alors dans la phase d'initialisation que les règles $D \rightarrow DET$, $G \rightarrow DET$ et $d \rightarrow DET$ sont interdites en associant un poids nul. Cet ajustement de la grammaire initiale, concernant les déterminants, est appliqué dans l'illustration de la Figure 3.23 par des pondérations par 0 signifiant l'élimination des règles.

En fonction des langues et des types de corpus, les ajustements peuvent différer. Tout d'abord, les catégories grammaticales n'ont pas nécessairement de correspondance dans toutes les langues ou les types de corpus. Ensuite, une règle pour une catégorie commune à deux langues peut s'appliquer pour l'une et non pour l'autre des langues. Par exemple, les adjectifs en français peuvent assumer chacun des rôles existant dans la grammaire 4bin, et ainsi dériver des 4 symboles non-

terminaux D, G, d et g (voir 3.24), car ils peuvent être dominants situés à gauche, dominants situés à droite, dominés situés à gauche et dominés situés à droite. Au contraire, en japonais, les adjectifs sont toujours dominés, en se situant à gauche de leur tête. Dans ce dernier cas, seule la règle $g \rightarrow ADJ$ doit être conservée dans la grammaire initiale, avant même la procédure d'apprentissage.

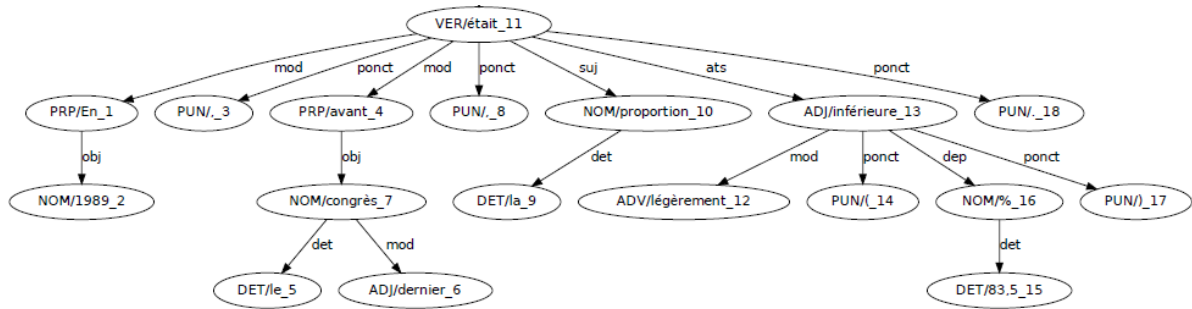


FIGURE 3.24 : Arbre de dépendances (fourni par [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010]) de la phrase « En 1989, avant le dernier congrès, la proportion était légèrement inférieure (83,5%) ». On y voit les différents rôles joués par l'adjectif « inférieure ». Il est dominé et situé à droite du verbe « était ». Il est dominant et situé à droite de l'adverbe « légèrement ». Il est dominant et situé à gauche du nom « % ». Dans cette même phrase, un autre adjectif, « dernier », est dominé et situé à la gauche du nom « congrès ». Ces exemples justifient de conserver, dans la grammaire initiale 4bin, les 4 règles $D \rightarrow ADJ$, $G \rightarrow ADJ$, $d \rightarrow ADJ$, $g \rightarrow ADJ$.

Nous donnons à la suite l'intégralité de la grammaire d'initialisation pour 4bin en français, les poids étant uniformément répartis. Une procédure de normalisation des poids est effectuée en interne de l'algorithme Inside-Outside afin que la grammaire soit effectivement stochastique (somme des poids des règles dérivant un même non-terminal valant 1).

Les catégories à gros grain du French Treebank présentes dans cette grammaire sont : A pour adjectif, ADV pour adverbe, C pour conjonction, CL pour clitique, DET pour déterminant, ET pour mot étranger, I pour interjection, NOM pour nom, P pour préposition, P+D pour préposition + déterminant, PONCT pour ponctuation, P+PRO pour préposition + pronom relatif, PREF pour préfixe, PRO pour pronom, et V pour verbe.

1.0 $S \rightarrow G d$	1.0 $G \rightarrow A$	0 $G \rightarrow C$	0 $G \rightarrow DET$
1.0 $S \rightarrow g D$	1.0 $D \rightarrow A$	1.0 $D \rightarrow C$	0 $D \rightarrow DET$
1.0 $D \rightarrow G d$	1.0 $d \rightarrow A$	1.0 $d \rightarrow C$	0 $d \rightarrow DET$
1.0 $D \rightarrow g D$	1.0 $g \rightarrow A$	1.0 $g \rightarrow C$	1.0 $g \rightarrow DET$
1.0 $G \rightarrow G d$	1.0 $S \rightarrow A$	1.0 $S \rightarrow C$	1.0 $S \rightarrow DET$
1.0 $G \rightarrow g D$	1.0 $G \rightarrow ADV$	0 $G \rightarrow CL$	1.0 $G \rightarrow ET$
1.0 $g \rightarrow G d$	1.0 $D \rightarrow ADV$	0 $D \rightarrow CL$	1.0 $D \rightarrow ET$
1.0 $g \rightarrow g D$	1.0 $d \rightarrow ADV$	1.0 $d \rightarrow CL$	1.0 $d \rightarrow ET$
1.0 $d \rightarrow G d$	1.0 $g \rightarrow ADV$	1.0 $g \rightarrow CL$	1.0 $g \rightarrow ET$
1.0 $d \rightarrow g D$	1.0 $S \rightarrow ADV$	1.0 $S \rightarrow CL$	1.0 $S \rightarrow ET$

1.0 $G \rightarrow I$	1.0 $d \rightarrow P$	1.0 $S \rightarrow PONCT$	1.0 $D \rightarrow PRO$
1.0 $D \rightarrow I$	1.0 $g \rightarrow P$	0 $G \rightarrow P + PRO$	1.0 $d \rightarrow PRO$
1.0 $d \rightarrow I$	1.0 $S \rightarrow P$	0 $D \rightarrow P + PRO$	1.0 $g \rightarrow PRO$
1.0 $g \rightarrow I$	0 $G \rightarrow P + D$	1.0 $d \rightarrow P + PRO$	1.0 $S \rightarrow PRO$
1.0 $S \rightarrow I$	1.0 $D \rightarrow P + D$	1.0 $g \rightarrow P + PRO$	1.0 $G \rightarrow V$
1.0 $G \rightarrow NOM$	1.0 $d \rightarrow P + D$	1.0 $S \rightarrow P + PRO$	1.0 $D \rightarrow V$
1.0 $D \rightarrow NOM$	0 $g \rightarrow P + D$	0 $G \rightarrow PREF$	1.0 $d \rightarrow V$
1.0 $d \rightarrow NOM$	1.0 $S \rightarrow P + D$	0 $d \rightarrow PREF$	1.0 $g \rightarrow V$
1.0 $g \rightarrow NOM$	0 $G \rightarrow PONCT$	1.0 $g \rightarrow PREF$	1.0 $S \rightarrow V$
1.0 $S \rightarrow NOM$	1.0 $D \rightarrow PONCT$	1.0 $S \rightarrow PREF$	
1.0 $G \rightarrow P$	1.0 $d \rightarrow PONCT$	0 $G \rightarrow PRO$	
1.0 $D \rightarrow P$	1.0 $g \rightarrow PONCT$		

Après l'apprentissage par Inside-Outside, sur le corpus `ftb6_1` du French Treebank [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010], cette grammaire apprise se présente sous la forme des règles pondérées suivantes :

0.983203 $S \rightarrow Gd$	0.00590928 $D \rightarrow C$	0.126342 $g \rightarrow P$
0.0149605 $S \rightarrow gD$	0.0310342 $d \rightarrow C$	0.000371177 $D \rightarrow P + D$
0.000815826 $D \rightarrow Gd$	0.028525 $g \rightarrow C$	0.0541536 $d \rightarrow P + D$
0.359552 $D \rightarrow gD$	0.000736151 $d \rightarrow CL$	0.000494461 $D \rightarrow PONCT$
0.108789 $G \rightarrow Gd$	0.0501499 $g \rightarrow CL$	0.229195 $d \rightarrow PONCT$
0.537837 $G \rightarrow gD$	0.326897 $g \rightarrow DET$	0.0274818 $g \rightarrow PONCT$
0.354257 $g \rightarrow Gd$	0.00463891 $G \rightarrow ET$	0.000203998 $S \rightarrow PONCT$
$1.61979e^{-06}$ $g \rightarrow gD$	0.00121383 $D \rightarrow ET$	$5.77072e^{-05}$ $d \rightarrow P + PRO$
0.525488 $d \rightarrow Gd$	$2.83346e^{-05}$ $d \rightarrow ET$	0.000310083 $g \rightarrow P + PRO$
0.000322569 $d \rightarrow gD$	0.000155671 $G \rightarrow I$	0.00148982 $g \rightarrow PREF$
0.089749 $G \rightarrow A$	$9.49462e^{-05}$ $D \rightarrow I$	0.0191713 $D \rightarrow PRO$
0.00075544 $D \rightarrow A$	0.0263235 $G \rightarrow NOM$	0.00756891 $d \rightarrow PRO$
0.00158932 $d \rightarrow A$	0.532454 $D \rightarrow NOM$	0.00942942 $g \rightarrow PRO$
0.0516761 $g \rightarrow A$	0.000222397 $d \rightarrow NOM$	0.151023 $G \rightarrow V$
0.0814846 $G \rightarrow ADV$	0.0234402 $g \rightarrow NOM$	0.0709549 $D \rightarrow V$
0.00821246 $D \rightarrow ADV$	0.00163199 $S \rightarrow NOM$	0.00050253 $d \rightarrow V$
0.000999321 $d \rightarrow ADV$	$3.44179e^{-13}$ $G \rightarrow P$	$8.29518e^{-09}$ $g \rightarrow V$
$2.79189e^{-16}$ $g \rightarrow ADV$	0.148102 $d \rightarrow P$	

On constate que les 10 règles de forme 1 sont gardées. En revanche, 17 règles de forme 2 conservées pour la grammaire d'initialisation, ont été pondérées trop faiblement et ont été éliminées lors de l'apprentissage.

Les règles éliminées sont les suivantes, dans la Table 3.5 :

$S \rightarrow A$	$g \rightarrow ET$	$D \rightarrow P$	$S \rightarrow PRO$
$S \rightarrow ADV$	$S \rightarrow ET$	$S \rightarrow P$	$S \rightarrow V$
$S \rightarrow C$	$d \rightarrow I$	$S \rightarrow P + D$	
$S \rightarrow CL$	$g \rightarrow I$	$S \rightarrow P + PRO$	
$S \rightarrow DET$	$S \rightarrow I$	$S \rightarrow PREF$	

TABLE 3.5 : Les règles éliminées par la procédure d'apprentissage dans nos expériences, pour la grammaire 4bin en français.

Il était attendu que les règles de forme 2 dérivant le symbole S seraient en majorité supprimées. Ces règles ne permettent que l'analyse de phrases d'un seul mot. Dans le corpus, il n'existe des phrases d'un mot que pour les catégories NOM et DET. C'est pourquoi sur 15 catégories, 13 ne sont pas autorisées à dériver directement de S , le cas ne se présentant pas dans le corpus d'apprentissage. En revanche, l'élimination des règles $g \rightarrow ET$, $d \rightarrow I$, $g \rightarrow I$ et $D \rightarrow P$ est beaucoup plus porteuse de sens sur le rôle des catégories « mots étrangers », « interjection » et « préposition ». La grammaire apprise montre que, dans le corpus d'apprentissage, la catégorie ET ne trouve pas de rôle de dominé gauche dans une optimisation de la vraisemblance du langage. De la même façon I ne doit pas être dominé, gauche ou droite, et P ne doit pas être dominant D.

3.4.2 Phase d'analyse

Une fois une grammaire, qui maximise la vraisemblance du corpus d'apprentissage, apprise, l'analyse de nouvelles phrases s'effectue par la procédure suivante.

Chaque phrase est analysée par CYK probabiliste, maximisant la probabilité de la chaîne de dérivation menant à la séquence des catégories de la phrase. Un arbre DGdg est ainsi construit. La phase de traduction depuis l'arbre DGdg en arbre de dépendances, présentée dans la sous-section suivante, est ensuite réalisée. En fin de procédure est donc obtenu un arbre de dépendances, dans lequel à chaque mot est associé un unique parent (la racine étant artificiellement attachée à un mot de position 0). La Figure 3.25 représente cette phase d'analyse, depuis la grammaire apprise jusqu'à l'arbre de dépendances final.

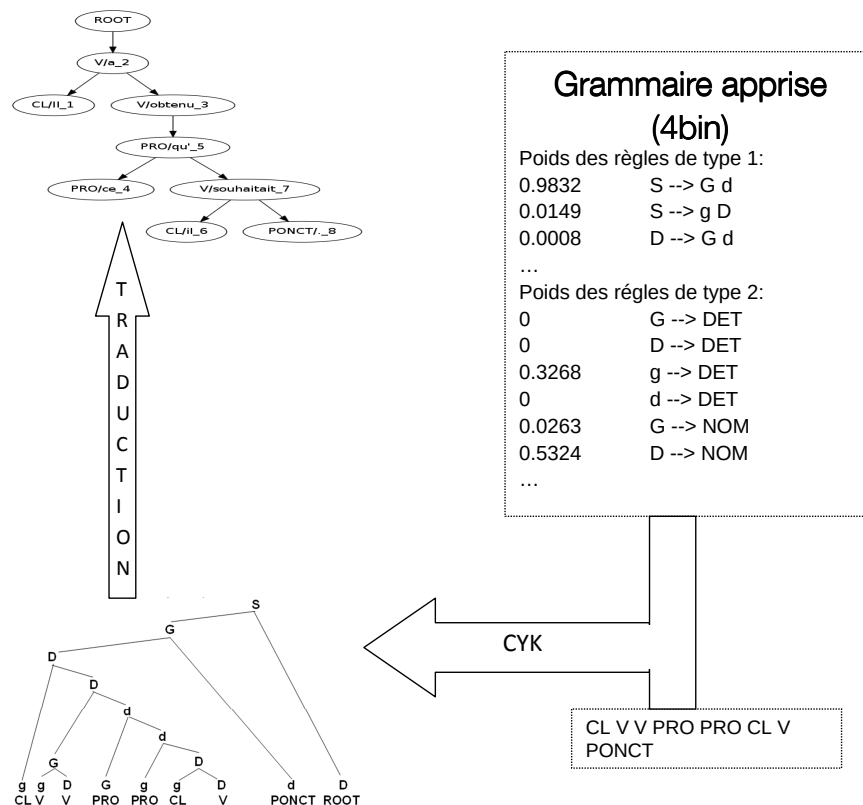


FIGURE 3.25 : Schéma représentant les 2 étapes de la phase d'analyse.

3.4.3 Traduction des arbres DGdg en arbres de dépendances

Nous démontrons en annexe l'équivalence entre notre formalisme DGdg (la version 4bin) et le formalisme de dépendances. Nous démontrons plus précisément que pour tout arbre de dépendances, il existe un arbre DGdg (version 4bin) se traduisant en cet arbre de dépendances. Mais cette fonction n'est pas injective, plusieurs arbres DGdg peuvent se traduire en un même arbre de dépendances, comme l'illustrent les figures 3.11 à 3.14.

Nous indiquons en détail la procédure ainsi qu'un pseudo-code en annexe.

Le processus de traduction d'un arbre DGdg en arbre de dépendances consiste en deux parcours bottom-up de l'arbre DGdg. Chaque nœud, lors du premier parcours, est étiqueté afin de permettre d'identifier la catégorie dominant chaque sous-arbre. Ensuite, le deuxième parcours permet d'associer chaque étiquette de la phrase avec sa tête. Le résultat final est une séquence indiquant la position de la tête (notée *motTête* dans la procédure détaillée en annexe) de chacune des catégories de la phrase (dernière ligne de la figure 3.26). Les nombres entre parenthèses dans la figure 3.26, en

exposant des nœuds internes et sous chaque catégorie sont les indices des motTête dans la phrase. Le motTête d'indice 0 correspond à la racine de l'arbre de dépendances.

Nous introduisons la notion de motTête dans le processus de traduction depuis l'arbre DGdg jusqu'à l'arbre de dépendances déduit. Le motTête désigne l'indice d'un mot considéré dans la phrase de départ. Associé à un non-terminal dans un arbre DGdg, il désigne le mot dominant tout le sous-arbre inférieur. Associé à un mot de la phrase, i.e. un nœud de l'arbre de dépendances, il désigne le parent direct de ce mot. Cela signifie qu'une fois le motTête de chaque feuille de l'arbre DGdg déterminé, l'arbre de dépendances est construit. Les figures 3.26 et 3.27 montrent sur un exemple le résultat des deux parcours successifs puis l'arbre de dépendances obtenu. On y voit par exemple que le motTête du non-terminal *d* (petit-fils de *S* vers la droite) est l'indice (3) désignant le verbe « parti ». Cela signifie que ce verbe « parti » domine, autrement dit est un ancêtre en dépendance, de tous les mots situés sur les feuilles du sous-arbre de racine ce symbole *d*. À la fin du deuxième parcours, (6) est désigné comme le motTête de la préposition « en », le mot « voiture » est le parent immédiat de ce mot « en ».

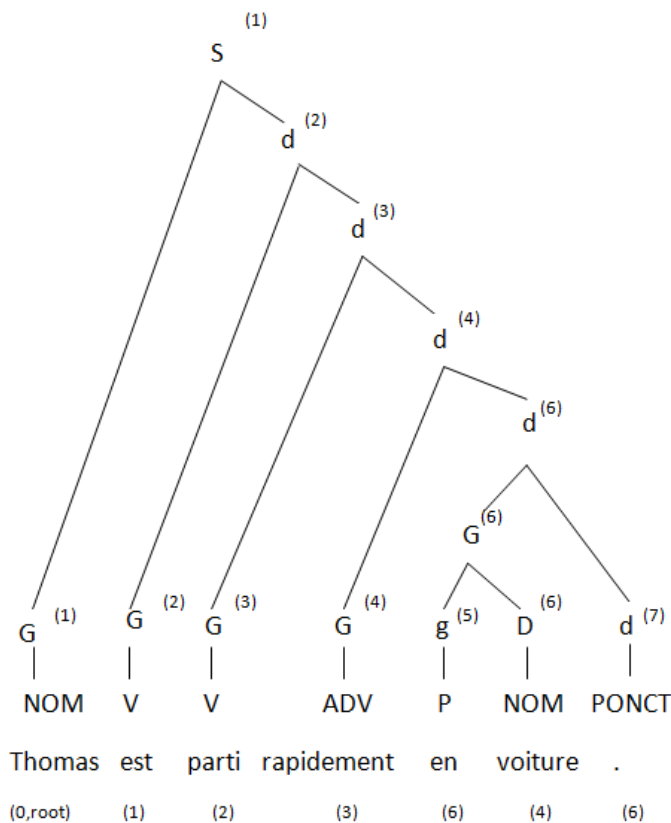


FIGURE 3.26 : Arbre DGdg de la phrase « Thomas est parti rapidement en voiture. » analysée à partir de la grammaire apprise 4bin. Les motsTête se trouvent en exposant de chaque non-terminal, et sous chaque feuille de l'arbre de dépendances.

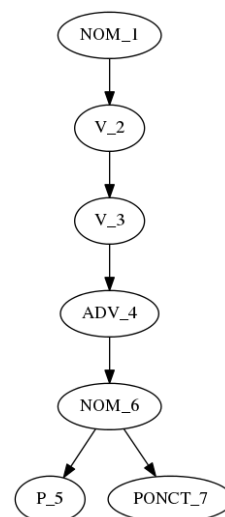
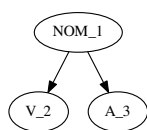
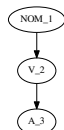
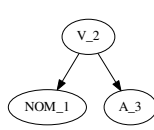
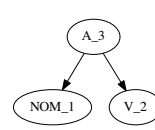
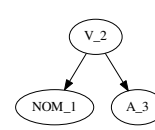
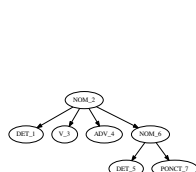
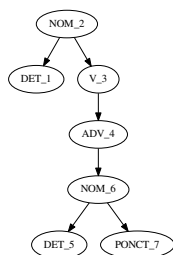
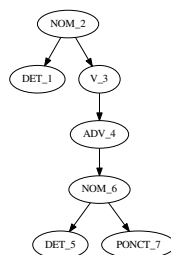
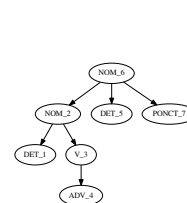
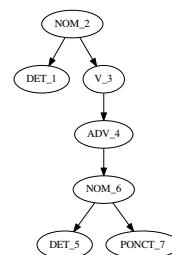


FIGURE 3.27 : Arbre de dépendances obtenu après la phase de traduction de la phrase « Thomas est parti rapidement en voiture. » analysée à partir de la grammaire apprise 4bin.

FIGURE 3.28 : 3bin,
UAS = 0/3.FIGURE 3.29 : 4bin,
UAS = 1/3.FIGURE 3.30 : 4ter,
UAS = 3/3.FIGURE 3.31 : 5ter,
UAS = 0/3.FIGURE 3.32 : 5ter+,
UAS = 3/3.

Arbres de dépendances de la phrase type NOM VA, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

FIGURE 3.33 : 3bin,
UAS = 2/6.FIGURE 3.34 : 4bin,
UAS = 3/6.FIGURE 3.35 : 4ter,
UAS = 3/6.FIGURE 3.36 : 5ter,
UAS = 3/6.FIGURE 3.37 : 5ter+,
UAS = 3/6.

Arbres de dépendances de la phrase type DET NOM V ADV DET NOM PONCT, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

Les figures 7.1 à 7.30, présentées en annexe à partir de la page 97, montrent, pour des phrases types, courtes et simples, l'analyse obtenue par les 5 variantes de la famille DGdg, apprises sur des phrases du journal *Le Monde* (grammaticalement étiquetées dans le French Treebank [Abeillé et Barrier, 2004]). Dans un but d'illustration, quelques figures sont déjà données dans ce chapitre (Figures 3.28 à 3.37), et sont reprises en annexes. Cela nous permet de comparer les arbres de dépendances, et leurs structures. On constate que dans la grande majorité des arbres, c'est le nom, que l'on veut considérer comme sujet de la phrase, qui se trouve à la racine des arbres, au lieu du verbe. Mais le verbe est ensuite attaché au nom, ce qui signifie que seul le sens de cet attachement est faux, tandis que d'autres liens de dépendances correspondent à un arbre classique. Le déterminant est toujours attaché correctement au nom, l'adjectif attribut est souvent attaché correctement au verbe (8/10, 3bin ne trouve pas cet attachement dans ces exemples), l'adverbe est la plupart du temps correctement attaché au verbe dont il dépend (8/10, 3bin n'attache pas correctement) et le nom dont la fonction serait COD est mal attaché au verbe (attaché correctement dans 7 cas sur 20, il est en fait souvent attaché à l'adverbe qui lui-même est attaché au verbe).

On observe que, dans tous nos exemples, la grammaire 3bin n'a su attacher correctement que le déterminant au nom qui le suit. Un nom est choisi comme racine de la phrase, on peut déjà penser que cette variante 3bin ne sera pas bien adaptée pour le français.

La variante 4bin permet de trouver le bon parent, dans nos exemples de phrases courtes, pour les adjectifs, les adverbes et les déterminants. Mais le verbe n'est encore jamais placé à la racine, ce qui est la première règle en analyse de dépendances.

La variante 4ter, vraisemblablement mieux adaptée au français, permet dans 2 cas sur 6 de placer correctement le verbe à la racine. Les erreurs sont alors une inversion de la relation entre le nom et le verbe, et l'attachement incorrect du nom (de fonction COD) à l'adverbe plutôt qu'au verbe.

La variante 5ter conduit quelquefois aux mêmes analyses qu'avec 4ter et 5ter+. Lorsque les analyses sont différentes, c'est en la défaveur de la variante 5ter qui semble donner lieu à des analyses non cohérentes entre elles. Dans nos exemples, l'ajout de la ponctuation perturbe toute la structure des arbres 5ter et provoque de nouvelles erreurs. La variante 5ter ne semble a priori pas bien adaptée à des perturbations mineures dans la phrase.

La variante 5ter+ a ici systématiquement donné lieu aux mêmes arbres que 4ter. Les grammaires sont proches, mais dans nos expérimentations à grande échelle, nous avons constaté que la présence d'un nouveau non-terminal neutre N, ainsi que les règles associées permettaient à 5ter+, pour le français, de se distinguer légèrement favorablement de la variante 4ter. Et pour d'autres langues, les résultats peuvent être sensiblement différents.

Résultats et analyse

Après avoir présenté, dans le chapitre précédent, les processus mis en place, nous montrons et analysons ici les résultats obtenus dans nos expérimentations. Nous avons procédé à une évaluation à grande échelle, sur 16 corpus dans 12 langues différentes, et provenant de sources variées, dans des styles d'écriture très différents. Pour évaluer la qualité des arbres de dépendances construits par notre méthode non supervisée DGdg, avec les 5 variantes, nous les comparons à des arbres construits manuellement, pour partie fournis par des universités ayant contribué à la mise en place des défis CoNLL 2006 [[Buchholz et Marsi, 2006](#)] et 2007 [[Nivre et al., 2007](#)], et d'autre part par des chercheurs linguistes de Google. Nous décrivons ces données dans la première section de ce chapitre. Nous exposons et comparons ensuite nos résultats à ceux obtenus par les arbres construits avec DMV, sur les mêmes corpus.

4.1 Descriptif des données d'évaluation

Les données sur lesquelles nous avons travaillé contiennent plus d'informations que nous n'en utilisons. Nous rappelons que nous considérons les phrases comme des séquences de catégories grammaticales. Ces catégories (ordonnées comme les unités lexicales le sont dans la phrase) constituent les seules entrées, autant pour l'apprentissage des grammaires que pour l'analyse de dépendances des phrases. Afin d'évaluer notre méthode de façon quantitative, nous nous sommes procuré des collections d'arbres de dépendances (treebanks), la provenance et le contenu de ces corpus sont détaillés à la suite. Les treebanks sont disponibles au format CoNLL, à savoir un format tabulé dans lequel chaque ligne décrit un mot, les phrases étant séparées par une ligne vide.

ID	FORM	LEMMA	CPOSTAG	POSTAG	FEATS	HEAD	DEPREL
1	À	À	P	P	–	7	mod
2	cette	ce	D	DET	g=f n=s s=dem	3	det
3	époque	époque	N	NC	g=f n=s s=c	1	obj
4	,	,	PONCT	PONCT	s=w	7	ponct
5	on	on	CL	CLS	g=m n=s p=3 s=suj	7	su
6	avait	avoir	V	V	m=ind n=s p=3 t=impft	7	aux_tps
7	dénombré	dénombrer	V	VPP	g=m m=part n=s t=past	0	root
8	cent_quarante	cent_quarante	D	DET	g=m n=p s=card	9	det
9	candidats	candidat	N	NC	g=m n=p s=c	7	obj
10	.	.	PONCT	PONCT	s=s	7	ponct

TABLE 4.1 : Une phrase du French Treebank de dépendances, fourni sous le format CoNLL. La dernière colonne n'est pas utile dans nos travaux, tout comme la colonne FEATS qui indiquent les traits pour détailler respectivement les étiquettes à gros grain (CPOSTAG) et à grain fin (POSTAG). La colonne HEAD ne sera utile qu'au moment de l'évaluation, puisqu'elle indique l'identifiant de la tête du mot. Nous n'utilisons, pour l'apprentissage et l'analyse, que les informations fournies dans la quatrième ou cinquième colonne.

On donne à la suite les origines des différents corpus, les statistiques de base sont présentées de façon synthétique dans le tableau 4.2 à la page 68. Tous les corpus sont proposés structurés en arbres de dépendances, sous le format imposé par la conférence CoNLL.

– Allemand

- Le corpus allemand peut être téléchargé à l'adresse : <http://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/tiger.en.html>. Il est décrit dans l'article [Brants *et al.*, 2002].
- Le TIGER corpus est constitué d'articles du journal *Frankfurter Rundschau*, pour lesquels chaque phrase a été étiquetée puis structurée en arbres de dépendances par l'université de Stuttgart.

– Anglais

Le Penn Treebank est historiquement le premier corpus à avoir été traité en analyse non supervisée de dépendances. Il est décrit à l'adresse : <http://www.cis.upenn.edu/~treebank/>, et par l'article [Marcus *et al.*, 1993]. Ce sont des articles du *Wall Street Journal*.

– Bulgare

Le corpus BulTreeBank-DP contient des phrases de livres de grammaire bulgare, de journaux, d'ouvrages littéraires et d'autre sources textuelles. La procédure de demande de ces données est indiquée à l'adresse <http://www.bultreebank.org/dpbtb/>. Ce corpus est décrit dans l'article [Simov *et al.*, 2002].

– Danois

Des informations sur le corpus danois se trouvent dans l'article [Kromann, 2003] et <http://ilk.uvt.nl/conll/data/danish/README>. Le Danish Dependency Treebank (DDT), PAROLE-DK contient des textes sélectionnés par Ole Norling-Christensen et la société pour la langue et la littérature danoises. Ces textes sont issus d'une large sélection aléatoire de textes représentatifs écrits entre 1983 et 1992.

- Espagnol

Le corpus espagnol est décrit à l'adresse <http://www.lsi.upc.edu/~nlp/tools.html> et par l'article [Civit et Martí, 2004]. Ce corpus CLOC-TALP est composé pour moitié d'extraits du journal *La Vanguardia*, et d'autre part du corpus LEXESP (lexique informatisé de l'espagnol). Cette partie LEXESP a été recueillie entre 1978 et 1995, elle est issue de différentes sources (littérature, presse, etc.) en espagnol péninsulaire et d'Amérique Latine.

- Français

Le corpus français est soumis à licence, et peut être obtenu à http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html. Il est composé de phrases écrites dans le journal *Le Monde* entre 1989 et 1995. Les articles [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010] décrivent précisément le treebank d'origine et sa conversion en treebank de dépendances.

- Hollandais

Le corpus hollandais (Alpino treebank) est décrit à <http://ilk.uvt.nl/conll/data/dutch/README> ainsi que dans l'article [Van der Beek *et al.*, 2002]. Les phrases de ce corpus sont issues de nombreuses sources : la partie nommée cdbl (journaux) du corpus Eindhoven, le guide d'annotation du corpus Spoken Dutch (CGN), des questions de plusieurs éditions de l'initiative CLEF, des phrases de EANS, des phrases construites spécialement pour ce corpus, des phrases de la documentation CGN Leuven Yellow Pages (i.e. issues des pages jaunes de la ville de Leven, en Flandres, avec étiquetage CGN (Corpus Gesproken Nederlands, corpus oral néerlandais)), de la Battle of the Parsers pour l'école d'hiver LOT 2001 et des questions de quiz.

- Japonais

Le corpus Tübingen Treebank of Spoken Japanese est constitué de transcriptions de dialogues, des conversations téléphoniques. On peut le trouver à l'adresse <http://www.sfs.uni-tuebingen.de/en/ascl/resources/corpora/tueba-js.html>, il est décrit par l'article [Hinrichs *et al.*, 2000].

- Portugais

Les textes portugais viennent du journal portugais quotidien *Público*. Des descriptions précises peuvent être trouvées dans l'article [Afonso *et al.*, 2002] ainsi que sur la page <http://ilk.uvt.nl/conll/data/portuguese/README>.

- Slovène

Le treebank de dépendances slovène est la traduction du premier tiers du roman de George Orwell *1984*. Les détails de l'étiquetage et de la construction des arbres sont exposés dans [Dzeroski *et al.*, 2006], et l'on peut trouver ce corpus à l'adresse <http://nl.ijs.si/sdt/>.

- Suédois

Le corpus suédois assemble deux parties, Talbanken et SUC. Talbanken reporte des phrases écrites (essais lycéens et professionnels) ainsi que du langage oral (entretiens, débats et conversations informelles) et a été produit dans les années 1970. Le corpus SUC contient du texte écrit issu de sources variées. On trouve des informations sur ces données dans l'article [Nilsson *et al.*, 2005] ainsi qu'à la page http://stp.lingfil.uu.se/~nivre/swedish_treebank/.

Ces corpus nous permettent de mener les expériences sur 11 langues, sur des types de discours hétérogènes entre les corpus mais aussi parfois au sein d'une même langue. Avant de mener les

expériences, on peut déjà anticiper sur une influence de cette « variabilité » dans l'apprentissage d'un modèle de dépendance.

En complément de ces données fréquemment utilisées en évaluation par les chercheurs en ANSD, nous avons aussi collecté des corpus librement distribués par Google depuis août 2013. Nous n'avons pas répertorié encore d'études en ANSD testant un modèle sur ces collections. Quatre langues sont communes avec les données CoNLL, avec des styles d'écriture différents. Une nouvelle langue est proposée, le coréen.

UAllemand, UCoréen, UEspagnol, UFrançais, USuédois :

L'article [McDonald *et al.*, 2013] explique l'étiquetage et la construction des arbres de dépendance de l'ensemble des corpus universels de Google. Le terme universel qualifie l'étiquetage harmonisé pour l'ensemble des corpus, dans les différentes langues. Les origines des textes sont résumées dans l'article, et viennent majoritairement d'Internet, de sources non nécessairement écrites par des professionnels. Le corpus allemand est constitué d'extraits de journaux et d'avis de consommateurs. Le slovène contient le Talbanken, qui est une partie du corpus slovène précédemment présenté. L'espagnol, comme le français, proviennent de sources variées telles que des journaux, des blogs et des avis de consommateurs, tandis que le corpus coréen vient de journaux et de blogs.

4.2 Discussion sur la complexité des modèles DGdg et DMV

Nous rappelons ici que l'une des problématiques pour l'apprentissage des grammaires, déjà énoncée dans le chapitre précédent, est liée au nombre de règles.

Le nombre de règles varie peu en fonction de la version de grammaire DGdg, mais est lié au nombre de catégories syntaxiques du corpus. Soit n le nombre de catégories dans un corpus donné, le nombre de règles possibles est : $4n + 8$ pour 3bin, $5n + 10$ pour 4bin, $5n + 20$ pour 4ter, $6n + 15$ pour 5 ter et $6n + 18$ pour 5ter+. La Table 4.3 indique :

- le nombre théorique maximal de règles pour chaque grammaire DGdg et pour DMV, une moyenne (non pondérée) sur les 16 corpus de ce nombre théorique avant toute procédure d'initialisation, une moyenne du nombre réel après l'initialisation, et une moyenne après l'apprentissage ;
- les temps moyens d'apprentissage et d'analyse des variantes DGdg par rapport à DMV, les temps étant calculés pour chaque corpus, la moyenne indiquée est calculée de façon non pondérée sur les 16 corpus.

Ces observations nous permettent d'observer que notre approche est au moins 2 fois plus rapide pour l'analyse et 4 fois plus rapide pour l'apprentissage comparativement à DMV. La complexité de chaque procédure est au moins de l'ordre de $O(m^3)$ (complexité d'Inside-Outside, présentée de cette façon dans les annexes de [Klein, 2005], et de CYK) en fonction du nombre m de règles dans la grammaire. La taille des phrases en entrée intervient également dans la complexité. Il est donc logique que la grammaire de DMV, contenant au moins 5 fois plus de règles que nos grammaires, voie son efficacité en temps tout à fait entravée.

# de règles	DMV	3bin	4bin	4ter	5ter	5ter+
Pour n étiquettes syntaxiques	$2n^2 + 9n$	$8 + 4n$	$10 + 5n$	$20 + 5n$	$15 + 6n$	$18 + 6n$
moyen avant l'initialisation	735.5	84	105	115	129	132
moyen après l'initialisation	583.4	82	98	108	119	122
moyen après l'apprentissage	565.6	55	63	70	84	86
Ratio de temps moyen d'apprentissage normalisé par rapport à DMV	1	0.1	0.09	0.26	0.11	0.17
Nb tokens moyen du corpus d'apprentissage	20 400	141 100 pour toutes les méthodes DGdg				
Ratio de temps moyen d'analyse normalisé par rapport à DMV	1	0.4	0.39	0.4	0.39	0.45
Nb tokens moyen du corpus de test	8 400 pour toutes les méthodes, dont DMV					

TABLE 4.3 : Éléments de comparaison entre DMV et la famille des grammaires DGdg sur les 16 corpus.

Dans la Table 4.3, le nombre de règles et les temps de calcul sont donnés en moyenne. Les temps pour DMV sont considérés comme la référence, les temps d'apprentissage avec les grammaires DGdg sont donnés sous forme de ratio par rapport aux temps de DMV. Le temps moyen d'apprentissage de DMV, calculé sur les 16 corpus, est de 365 minutes sur les phrases de moins de 10 mots (en moyenne 20 400 mots par corpus). Les apprentissages des grammaires DGdg s'effectuent sur toutes les phrases (en moyenne 141 100 mots par corpus). Le temps moyen d'analyse pour DMV est de 10.3 secondes. Toutes ces expériences ont été menées sur le même ordinateur, avec un processeur Intel®Xeon®CPU E5-1620 0 @ 3.6GHz $\times$ 8, et 16 Go RAM. Nous avons utilisé le code DMV fourni par Cohen et Smith [Cohen et Smith, 2010] (<http://code.google.com/p/jdageem/>), avec l'initialisation harmonique, mais nous n'avons pas réussi à retrouver leur paramétrisation, et ainsi pas à retrouver des scores aussi bons que les leurs sur les mêmes corpus. Cela montre l'influence de la paramétrisation pour la méthode DMV.

4.3 Expériences sur les catégories à gros grain, les 5 variantes de DGdg et DMV

Dans cette section, nous présentons les expériences menées sur les phrases considérées comme des séquences de catégories à gros grain, qui sont communes aux corpus Google, mais différentes dans tous les autres corpus (CoNLL et French Treebank). Nous avons utilisé les catégories à gros grain et considéré la ponctuation pour la construction des arbres, mais pas dans l'évaluation. Nous considérons que les signes de ponctuation jouent un rôle dans la construction des arbres DGdg, et qu'ils sont un élément de la phrase qui n'est pas à négliger. En revanche, en ANSD l'évaluation doit toujours être menée par le calcul des UAS (Unlabelled Attachment Score), qui exclut le décompte des liens d'attachement des signes de ponctuation. Les résultats de DMV sont ceux que nous avons obtenus par le code de Cohen et Smith utilisé dans l'article [Cohen et Smith, 2010] avec l'initialisation harmonique, en utilisant aussi les catégories à gros grain, et en ignorant la ponctuation car c'est ainsi que ces auteurs ont choisi de mener leurs expérimentations. Nous avons appris les

modèles sur une partie importante de chaque corpus (en général, 90% des données, les partages entre corpus d'apprentissage et de test sont donnés par les créateurs des corpus, cf. Table 4.2 de la page 68). Nous avons ensuite, sur la partie test, conduit l'analyse avec les modèle appris. Les arbres ainsi construits ont alors été comparés aux arbres de référence fournis pour le corpus de test, et nous avons donc effectué le calcul de l'UAS.

Les résultats quantitatifs, pour chaque corpus, pour chacune des variantes DGdg, sont indiqués dans la Table 4.4. On y indique également, à titre de comparaison, les résultats que nous avons obtenus avec la méthode DMV de Klein et Manning, par le code fourni par Cohen et Smith ([Cohen et Smith, 2010], <http://code.google.com/p/jdageem/>). Nous avons utilisé l'initialisation automatique que Klein et Manning appellent « harmonique ». N'ayant pas réussi à retrouver d'aussi bons résultats que ceux annoncés dans l'article [Cohen et Smith, 2010], nous les avons reportés parmi ces résultats intermédiaires, mais pas dans la table récapitulative comparant nos meilleurs résultats à ceux des baselines.

CoNLL 2006 et 2007 + FTB	3bin	4bin	4ter	5ter	5ter+	DMV
Allemand	20.9	33.1	20.9	31.5	31.7	30.3
Anglais	15.3	29.0	26.4	38.1	39.0	17.6
Bulgare	17.7	23.9	22.9	22.8	23.6	30.7
Danois	26.7	20.5	14.0	13.9	13.6	21.6
Espagnol	20.8	39.2	39.2	30.0	40.2	19.4
Français	29.8	32.9	42.1	37.4	42.2	17.1
Hollandais	29.7	34.6	30.3	27.0	27.0	25.2
Japonais	31.2	32.4	32.2	33.4	64.7	58.9
Portugais	30.0	54.0	42.0	37.8	34.1	23.9
Slovène	12.2	21.5	23.2	21.6	21.8	23.7
Suédois	21.2	18.9	21.6	21.8	21.8	28.7
TBDUG*						
UAllemand	34.9	29.8	31.1	28.5	28.2	27.4
UCoréen	38.8	41.3	38.1	37.2	36.9	44.2
UEspagnol	26.3	34.1	33.1	41.6	36.4	33.8
UFrançais	24.5	32.2	32.5	39.6	36.4	22.4
USuédois	17.4	23.2	23.9	21.5	21.6	27.3
MOYENNE	24.8	31.3	29.6	30.1	32.7	28.3
MOYENNE DGdg			29.7			

TABLE 4.4 : Scores UAS pour toutes les variantes de la famille de grammaires DGdg. Tests effectués sur des treebanks de dépendances de CoNLL 2006 et 2007 shared task et French Treebank et le Treebank de Dépendances Universel de Google (TBDUG*). Ces tests ont pris en compte les catégories à gros grain.

La Table 4.4 donne les résultats obtenus sur les 16 corpus avec notre méthode DGdg (5 va-

riantes). On constate que les différentes variantes ne rencontrent pas un succès équivalent, pour chaque corpus. Il apparaît clairement qu’une ou plusieurs variantes sont à favoriser en fonction de la langue et du type de discours. La variante 5ter+ est la plus performante sur ces corpus, dans le sens où elle obtient les meilleurs UAS pour 4 langues (anglais, espagnol, français et japonais), que ses scores sont bons (entre 39 et 64.7% sur ces 4 corpus) dans des langues répandues et qu’elle donne lieu à la meilleure moyenne sur tous les corpus. Ensuite, la variante 4bin se distingue pour 4 langues aussi, mais en considérant le bulgare qui donne lieu à de mauvais scores pour toutes les variantes (23.9% par 4bin). 5ter et 4ter donnent lieu à de bons scores, mais qui sont tout de même surpassés par ceux d’autres variantes. Ainsi, 5ter est la meilleure variante pour 2 corpus, et 4ter pour un seul. La variante 3bin n’est la plus performante que pour deux corpus, le danois et UAllemand. Pour les autres corpus, elle obtient en général des scores nettement inférieurs au meilleur score par une autre variante.

Nous avons regardé comparativement les scores entre les variantes. Remarquons que, dans nos expériences, avec le code de Cohen et Smith ([Cohen et Smith, 2010], <http://code.google.com/p/jdageem/>), les résultats sont souvent mauvais, meilleurs que toutes les variantes de DGdg dans seulement 3 cas.

Nous observons ensuite ces résultats par le prisme du regroupement en clusters proposé dans le dendrogramme de la Figure 4.1.

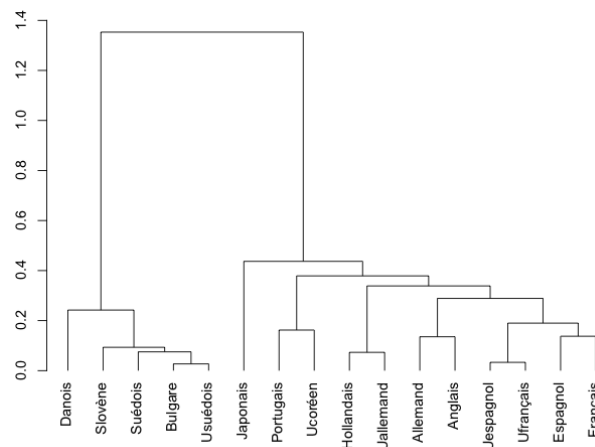


FIGURE 4.1 : Dendrogramme de clusters établi avec la méthode de Ward en utilisant les distances euclidiennes entre les vecteurs de scores UAS par langue (3bin, 4bin, 4ter, 5ter, 5ter+) de la Table 4.4.

Nous notons, dans la Figure 4.1, un net partage entre les deux groupes principaux, celui de gauche (regroupant le danois, le slovène, le suédois, le bulgare et le suédois de Google) et celui de droite (japonais, portugais, coréen de Google, hollandais, allemand de Google, allemand, anglais, espagnol de Google, français de Google, espagnol et français). Le premier groupe (à gauche) contient les langues pour lesquelles aucune des variantes n’obtient de bons résultats, aucun de nos scores sur ces langues ne dépasse 27%. Nous remarquons que ce cluster est formé des langues slaves et scandinaves. Des tests complémentaires sur des corpus dans des langues appartenant à

ces deux familles sont nécessaires pour accréditer le fait que la méthode DGdg n'est pas appropriée pour les traiter.

En revanche, le groupe de droite contient des langues d'origines très diverses : des langues altaïques (japonais et coréen de Google), latines (portugais, espagnol et espagnol de Google, français et français de Google) et germaniques occidentales (hollandais, allemand et allemand de Google, anglais). Pour ces corpus, au moins une grammaire de la famille DGdg obtient des résultats supérieurs à 33%, jusqu'à 64%. Il est tout à fait pertinent de noter que nos 5 variantes, par leurs légères différences, permettent de couvrir honorablement une grande variété de langues.

Dans une approche plus détaillée, nous nous focalisons sur les corpus pour lesquels au moins une méthode obtient de bons résultats, c'est-à-dire le cluster de droite. Le regroupement par paires des 4 corpus les plus à droite interpelle. Le français et l'espagnol sont des langues proches, il est donc cohérent de les voir regroupées. Cependant la répartition dans le niveau le plus bas est intrigante. Les deux corpus espagnols ne sont pas regroupés comme plus proches, il en est ainsi également pour le français. Précisément, les corpus de français et d'espagnol, constitués de phrases issues de journaux, écrites par des journalistes dans un style plutôt formel, obtiennent tous deux leur meilleur résultats par l'analyse avec la grammaire 5ter+. Cette grammaire autorise la récursivité d'un symbole dominant central (voir sous-section 3.3.3), ce qui doit permettre une analyse plus complexe que par 5ter, qui ne l'autorise pas. Il est cohérent que des phrases longues et élaborées soient mieux modélisées par la grammaire autorisant les structures les plus complexes. Ceci est confirmé par le bon score obtenu par cette grammaire 5ter+ sur le corpus anglais, formé d'extraits du *Wall Street Journal*. En revanche, l'extrême réussite de cette grammaire sur le corpus japonais, formé de transcriptions très courtes de discussions téléphoniques questionne.

Inversement, les corpus d'espagnol et de français fournis par Google, formés de flux de nouvelles, d'avis de consommateurs et de textes issus de blogs, écrits par des internautes dans un style informel, sont mieux analysés par la grammaire 5ter, qui produit des analyses moins complexes.

4.4 Expériences partielles sur les catégories à grain fin, les 5 variantes de DGdg et DMV

Les résultats des expériences menées sur les séquences de catégories à grain fin (différentes suivant chaque corpus) sont présentés dans la Table 4.5. Nous avons utilisé les catégories à grain fin et considéré la ponctuation pour la construction des arbres, mais pas dans l'évaluation. Les résultats de DMV sont ceux que nous avons obtenus par le code de Cohen utilisé dans l'article [Cohen et Smith, 2010] avec l'initialisation harmonique, en utilisant aussi les catégories à grain fin, et en ignorant la ponctuation. Pour des problèmes techniques, deux résultats n'ont pas pu être obtenus. Signalons que des problèmes de corpus ont quelques fois provoqué des difficultés pour la phase d'analyse. Par exemple, en allant observer de près les quelques phrases qui ne pouvaient pas être analysées dans le corpus hollandais, nous nous sommes rendus compte qu'il existait des catégories fines absentes dans le corpus d'apprentissage et présentes dans le corpus de test. Les règles de dérivation en ces non-terminaux, de forme 1, disparaissaient donc logiquement de la grammaire apprise, et empêchaient l'analyse. Nous avons donc artificiellement ajouté de nouveau ces règles dans la grammaire apprise pour qu'une analyse des phrases du corpus de test soit possible.

Nous constatons une perte de performance de presque 3 points en moyenne sur toutes les grammaires DGdg et tous les corpus. Ceci peut s'expliquer par le fait que la phase d'initialisation (i.e. la suppression de règles de type 2 suivant les catégories, cf. paragraphe 3.4.1) est plus délicate du fait du nombre important de catégories et de leur signification plus technique (experte), et sans doute moins bien ajustée par nous. Nous pouvons aussi en déduire que l'affinement des catégories, induisant un gain de score pour DMV, n'est pas intéressant pour nos méthodes.

CoNLL 2006 et 2007 + FTB	3bin	4bin	4ter	5ter	5ter+	DMV
Allemand	20.3	-	20.3	30.6	30.6	33.9
Anglais	19.4	3.6	31.1	22.7	22.3	26.2
Bulgare	17.1	18.3	21.6	23.2	21.5	33.6
Danois	4.4	7.0	6.8	11.5	11.5	25.1
Espagnol	26.4	10.0	26.1	30.9	30.9	21.5
Français	27.4	8.8	26.6	-	31.9	24.8
Hollandais	31.2	30.7	31.0	41.3	41.3	29.1
Japonais	19.4	40.9	41.0	46.6	47.1	49.1
Portugais	28.1	54.0	34.4	29.6	29.7	25.7
Slovène	29.1	34.0	28.7	30.2	33.4	32.6
Suédois	20.9	25.2	20.0	21.7	21.7	28.7
TBDUG*						
UAllemand	25.5	30.1	30.3	27.8	28.1	29.3
UCoréen	38.1	38.1	38.1	33.4	33.2	49.7
UEspagnol	29.1	43.5	43.5	26.7	26.6	40.9
UFrançais	37.7	16.8	39.6	29.9	29.8	28.1
USuédois	16.3	7.1	19.3	17.9	17.9	31.2
MOYENNE	24.5	24.5	28.6	28.28	28.6	31.8
MOYENNE DGdg			26.9			

TABLE 4.5 : Scores UAS pour toutes les variantes de la famille de grammaires DGdg. Tests effectués sur des treebanks de dépendances de CoNLL 2006 et 2007 shared task et French Treebank et le Treebank de Dépendances Universel de Google Google (TBDUG*). Ces tests ont pris en compte les catégories à grain fin. Deux résultats sont manquants, à cause d'un problème technique.

D'autres expériences ont été réalisées, que nous ne présentons pas ici, mais dont nous pouvons synthétiser les principaux enseignements ainsi :

- les corpus d'apprentissage et de test sont déjà partagés dans les données fournies. Notre méthode n'étant pas supervisée, nous nous sommes interrogés sur l'altération des résultats pour une analyse conduite sur un corpus par une grammaire apprise sur un corpus différent. Nous avons donc analysé le corpus sur lequel les grammaires avaient été apprises, et constaté la négligeabilité de l'amélioration. La moyenne sur toutes les méthodes, sur tous les corpus, qui

est de 29.7 ne devient que 29.8. Ceci est encourageant pour la portabilité de notre méthode DGdg;

- il est aussi rapporté dans l'article [Spitkovsky *et al.*, 2010] que la trop grande taille des phrases d'apprentissage pouvait agir négativement sur les performances de DMV, incitant ainsi à limiter l'apprentissage aux phrases de moins de 10 mots. Nous avons alors appris seulement sur les phrases de moins de 10 mots du corpus d'apprentissage, puis analysé toutes les phrases du corpus de test. Là encore, nous ne pouvons reprendre pour notre méthode cette particularité de DMV, la perte engendrée étant significative (2.8 points en moyenne).

4.5 Synthèse comparative avec l'état de l'art

Nous comparons par les résultats présentés dans la Table 4.6 de la page 69, l'approche DGdg, et les résultats publiés dans les articles [Cohen et Smith, 2010 ; Cohen *et al.*, 2011], [Gillenwater *et al.*, 2011] et [Spitkovsky *et al.*, 2012]. Nous obtenons de meilleurs résultats que la méthode naïve d'attachement gauche (ou droit suivant les langues) pour 9 corpus sur 16, et en moyenne nous améliorons ce score de 4.4 points. Comme signalé dans la section 4.3, notre méthode n'est pas bonne pour les langues slaves et scandinaves. Si nous écartons ces 5 corpus, nos scores dépassent ceux de l'attachement gauche/droite pour 9 des 10 langues.

Dans les langues pour lesquelles il existe une référence donnée dans [Cohen et Smith, 2010 ; Cohen *et al.*, 2011], [Gillenwater *et al.*, 2011] ou [Spitkovsky *et al.*, 2012], nous nous comparons favorablement pour 4 des 9 langues seulement à Cohen (-3 pts en moyenne), 2 sur 10 à Gillenwater (-4.1 pts en moyenne) et 6 sur 10 à Spitkovsky (+5.4 pts en moyenne). Rappelons cependant l'importance de l'initialisation pour cette méthode DMV. Ces trois références annoncent utiliser DMV avec l'initialisation harmonique, sur les mêmes corpus, avec les catégories à grain fin, c'est-à-dire mener les mêmes expériences dans les mêmes conditions, et ils obtiennent pourtant des résultats significativement différents (jusqu'à 10.6 pts de différence pour le japonais). La Table 4.3 nous rappelle les délais d'apprentissage 4 fois plus longs pour DMV et d'analyse au moins 2 fois plus longs.

Nous cherchons à trouver un critère qui permettrait à un utilisateur présentant un nouveau corpus (grammaticalement étiqueté, mais sans arbres de dépendances) de choisir entre les 5 variantes de la famille de grammaire DGdg la plus adaptée à sa langue, à son corpus. Nous travaillons en apprentissage non supervisé, nous ne pouvons pas, comme nous l'avons fait dans ces expérimentations, nous appuyer sur une analyse a posteriori pour déterminer la variante la plus adaptée à un nouveau corpus, dans une nouvelle langue. En effet, nous considérons que nous ne disposons d'aucun arbre de dépendances, nous ne pouvons donc pas évaluer chacune des variantes même sur une petite partie du corpus. Nous envisageons de travailler sur un critère de choix d'une variante, prenant en compte la valeur de la vraisemblance du corpus par rapport à la variante à la fin de l'apprentissage sur tout ou partie du corpus.

Après avoir analysé les vraisemblances des corpus en fonction des variantes, en regard des UAS, nous sommes en mesure de nier l'affirmation selon laquelle plus la vraisemblance est petite, meilleur sera l'UAS. En effet, plusieurs corpus contredisent cette hypothèse. Par exemple, pour le français du French Treebank, le meilleur UAS est obtenu par la variante 5ter+ (42.3%), à la fin de cet apprentissage, la vraisemblance du corpus d'apprentissage par rapport à la grammaire apprise est de 2.94. Un score intermédiaire est obtenu par 5ter (37.4%) et la vraisemblance du même corpus

par rapport à cette grammaire apprise 5ter est inférieure, elle vaut 2.91. La variante 4bin obtenant le plus mauvais score, 29.7%, a pourtant une vraisemblance surpassant les deux autres, 3.08. Nous voyons clairement par cet exemple (qui se retrouve plusieurs fois parmi les autres relevés de vraisemblance), que ce critère, naïf, ne peut pas être utilisé, car contredit par nos expériences. Nous cherchons à l'affiner.

En considérant pour chacune des grammaires la moyenne et l'écart type des vraisemblances obtenues en fin de chaque apprentissage et en faisant une hypothèse de distribution gaussienne avec ces paramètres, on peut mesurer l'adéquation d'un résultat de vraisemblance v avec cette distribution estimée empiriquement en calculant : $\min\{P(X \leq v), 1 - P(X \leq v)\}$. Dans près de la moitié des cas (7 sur 16 corpus exactement), la meilleure adéquation correspond également au meilleur UAS. C'est encore peu, mais c'est bien mieux que l'aléatoire (1/5) et que la comparaison des valeurs (2/16) ou des écarts à la moyenne (6/16). Nous pourrions affiner encore cela en ajustant mieux les paramètres des distributions.

Corpus d'apprentissage							
Langue	Origine	# catégories à gros grain	# catégories à grain fin	# mots ponct comprise	# phrases	Moyenne mots par phrase	# mots dans les phrases de moins de 10 mots
Allemand	J	48	52	699 331	39 214	17.8	77 705
Anglais	J	23	35	143 990	5 987	24.1	6 233
Bulgare	J,L	12	46	190 217	12 823	14.8	38 220
Danois	M	10	24	94 386	5 190	18.2	11 479
Espagnol	J,L	15	41	89 334	3 306	27.0	4 230
Français	J	15	27	278 023	9 881	28.1	10 197
Hollandais	M	13	302	195 069	13 349	14.6	43 405
Japonais	D	20	70	151 461	17 044	8.9	43 501
Portugais	J	16	22	206 490	9 070	22.8	15 015
Slovène	L	12	29	28 750	1 534	18.7	3 857
Suédois	L,D	15	41	191 467	11 042	17.3	11 726
UAllemand	J,A	13	16	30 460	2 200	13.8	7 330
UCoréen	J,B	11	20	62 378	5 308	11.8	19 219
UEspagnol	J,B,A	13	24	94 232	3 345	28.2	3 074
UFrançais	J,B,A	12	14	74 979	3 312	22.6	5 409
USuédois	L,D	12	23	66 631	4 447	15.0	11 726

Corpus de test							
Langue				# mots ponct comprise	# phrases	Moyenne mots par phrase	
Allemand				5 694	357	15.4	
Anglais				12 483	501	24.9	
Bulgare				5 929	396	15.0	
Danois				5 852	322	18.2	
Espagnol				5 694	206	27.6	
Français				35 887	1220	29.4	
Hollandais				5 585	386	14.5	
Japonais				5 711	709	8.0	
Portugais				5 867	288	20.4	
Slovène				6 390	402	15.9	
Suédois				5 656	389	14.5	
UAllemand				16 339	1000	16.3	
UCoréen				2 917	298	9.8	
UEspagnol				8 295	300	27.6	
UFrançais				6 950	300	23.2	
USuédois				20 376	1219	16.7	

TABLE 4.2 : Description chiffrée des corpus d'apprentissage. Les origines sont notées : J pour des extraits de journaux, M pour un mélange de textes de genres différents, L pour littérature, D pour discours oral, B pour blogs et A pour avis de consommateurs sur des sites Internet.

CoNLL 2006 et 2007 +FTB												TBDUG					MOY
	All	Ang	Bul	Dan	Esp	Fra	Hol	Jap	Por	Slo	Sué	UAl	UCo	UFr	UEs	USu	
Left or Right	22.4	29.2	36.0	46.0	29.5	32.4	27.6	56.8	28.8	23.9	25.8	24.9	52.5	27.4	26.2	24.6	32.1
	(right)	(right)	(left)	(left)	(left)	(left)	(right)	(right)	(left)	(left)	(right)	(right)	(right)	(left)	(left)	(right)	
DMV*	-	35.9	36.3	41.4	38.1	-	38.6	43.0	42.5	37.0	42.3	-	-	-	-	-	39.5
DMV**	35.7	45.8	37.8	47.2	40.3	-	37.1	52.8	35.7	42.3	39.4	-	-	-	-	-	41.4
DMV***	27.7	32.3	24.7	22.4	22.0	-	44.9	42.2	37.1	33.4	30.7	-	-	-	-	-	31.1
RLrl	33.1	39.0	23.9	26.7	40.2	42.2	34.6	64.7	54.0	23.2	21.8	34.9	41.3	41.6	39.6	23.9	36.5
	(4bin)	(5ter+)	(4bin)	(3bin)	(5ter+)	(5ter+)	(4bin)	(5ter+)	(4bin)	(4ter)	(5ter)	(3bin)	(4bin)	(5ter)	(5ter)	(4ter)	

TABLE 4.6 : UAS pour 3 méthodes, simple attachement gauche ou droit, DMV meilleure version DGdg sur les 16 corpus. DMV*, DMV** et DMV*** sont respectivement repris des articles de Cohen et Smith [Cohen et Smith, 2010 ; Cohen *et al.*, 2011], Gillenwater *et al.* [Gillenwater *et al.*, 2011] et Spitkovsky *et al.* [Spitkovsky *et al.*, 2012]. Nous rappelons que nous utilisons des catégories à gros grain alors que ces deux références utilisent les catégories à grain fin.

Application des arbres de dépendances DGdg - - Extraction d'informations par CRF

Dans le chapitre précédent, nous avons évalué la qualité des arbres de dépendances construits grâce aux variantes de la méthode DGdg. Ils sont, globalement, de bien meilleure qualité que ceux construits par la méthode de base d'attachement systématique à droite (ou à gauche), et un peu moins bons que les arbres construits par DMV dans des versions dont les paramètres ont été optimisés en fonction du corpus. Nous nous intéressons, dans ce nouveau chapitre, à une utilisation des arbres générés automatiquement par ANSD pour extraire à peu de frais de l'information essentielle dans des textes. Des démonstrations de l'apport des arbres de dépendances dans le domaine de l'extraction de relations ont déjà été faites [Wu et Weld, 2010]. Nous cherchons ici à voir si nos arbres construits automatiquement, rapidement et de façon non supervisée, permettent eux aussi d'améliorer la qualité de l'extraction de relations.

5.1 Introduction

L'extraction d'informations est un champ de recherche très vaste. Nous montrons ici un intérêt particulier de nos travaux, une application rapide et valorisante.

L'objectif d'une tâche d'extraction d'informations est l'identification d'instances particulières d'entités de types prédéfinis, à partir de textes écrits en langage naturel. Les propriétés, ou arguments, des entités doivent être identifiées aussi. Le chapitre consacré à la présentation de la tâche d'extraction d'informations dans [Piskorski et Yangarber, 2013] en propose un panorama complet et récent. On y explique que la tâche d'extraction d'informations est l'identification d'un ensemble prédéfini de concepts dans un domaine spécifique, effectuée tout en ignorant les autres informations non pertinentes. Les relations sont fixées à l'avance, il s'agit d'en retrouver les entités dans un corpus de textes donné. L'objectif de cette tâche est de compléter des bases de données et les liens entre les entités de ces bases. Nous nous intéressons à la tâche plus précise d'extraction de relations qui cherche à détecter et classifier des relations prédéfinies dans un texte. Les entités recherchées sont souvent du type « personne », « organisation » ou « lieux ». Plusieurs notations sont utilisées : relation(premier argument, second argument) ou sous forme de triplet : (premier argument, relation, second argument). On peut, par exemple, détecter des relations telles que situé_à(siège_UNU,

New_York), ou père_de(John_Coltrane, Ravi_Coltrane). En 2008, Etzioni *et al.* [Etzioni *et al.*, 2008] introduisent un nouveau domaine d'extraction d'informations : l'Open Information Extraction. Dans ce cadre, le type des relations n'est plus spécifié, on cherche à extraire le maximum de relations, sans nécessairement s'intéresser précisément au type.

Les structures de dépendance apportent des renseignements très riches, exploités dans les outils d'extraction d'informations. En bio-informatique, l'approche RelEx [Fundel *et al.*, 2007] obtient des scores de 80%, tant en rappel qu'en précision, sur l'extraction de 150 000 relations seulement à partir de résumés d'articles médicaux (dans le domaine de la génétique). Les phrases sont grammaticalement étiquetées par MedPost¹ [Smith *et al.*, 2004], segmentées pour identifier les groupes nominaux par fnTBL² [Ngai et Florian, 2001]. Ces phrases étiquetées sont ensuite soumises au Stanford Lexicalized Parser³ [Klein et Manning, 2003ba] qui construit des arbres de dépendances, Figure 5.1. L'extraction de relations entre protéines (ou gènes) se fait en deux étapes. Des relations candidates sont d'abord créées par la sélection de chemins liant des paires de protéines (repérées par dictionnaire) dans l'arbre de dépendances. Cette étape est effectuée par des règles reflétant les constructions les plus courantes en anglais pour décrire ces relations entre protéines. Ensuite, des filtres sont appliqués, on supprime, parmi d'autres, les relations pour lesquelles une négation a été repérée et celles formulées de façon passive, ce qui inverserait les rôles des protéines.

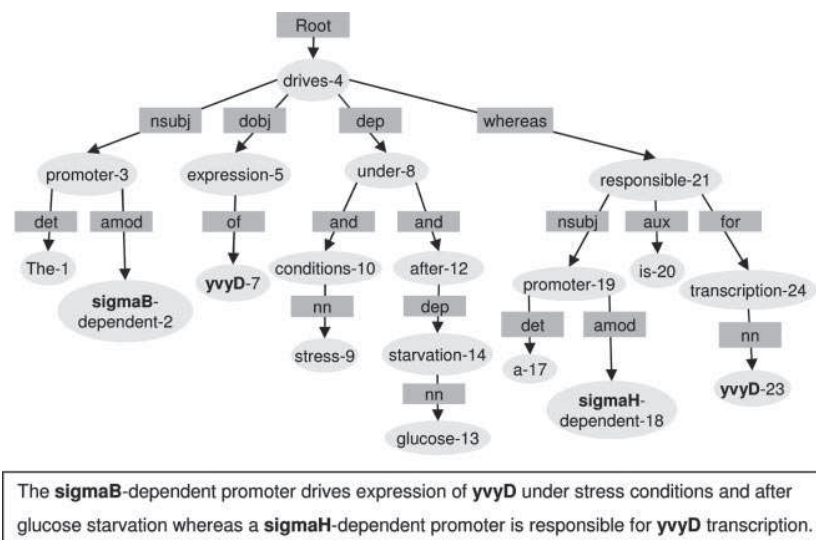


FIGURE 5.1 : Arbre de dépendances fourni par le Stanford Lexicalized Parser pour la phrase encadrée. Les mots en gras indiquent les noms des gènes ou protéines. Cette figure est extraite de l'article [Fundel *et al.*, 2007].

Wu et Weld [Wu et Weld, 2010] montrent que l'analyse en dépendances permet d'augmenter très fortement les performances de leur extracteur d'informations WOE, en particulier le rappel.

1. <ftp://ftp.ncbi.nlm.nih.gov/pub/lsmith/MedPost/medpost.tar.gz>

2. <http://nlp.cs.jhu.edu/~rflorian/fntbl/>

3. <http://nlp.stanford.edu/nlp/javadoc/javanlp-3.5.0/edu/stanford/nlp/parser/lexparser/LexicalizedParser.html>

Ici encore, l'analyse en dépendances, effectuée par le parser Stanford dependencies [Marneffe et Manning, 2008], est obtenue à partir des mots lexicaux, et ne nécessite pas d'étiquetage syntaxique préalable. Ils se placent dans le domaine de l'Open Information Extraction, qui vise à extraire tout type de relations. Ils utilisent Wikipedia pour l'apprentissage auto-supervisé d'extracteurs non lexicalisés, ils utilisent les informations structurées proposées par Wikipedia en ne s'intéressant qu'aux catégories morphosyntaxiques. Les dépendances impliquant les conjonctions ou les prépositions sont condensées pour obtenir des relations directes entre les mots porteurs de sens (tels que noms et verbes). Wu et Weld apprennent alors une sélection de relations pertinentes avec des patterns de chemins courts dans les arbres de dépendances. Ils généralisent cet ensemble de patterns en remplaçant les mots par les étiquettes morphosyntaxiques. Ainsi, ils peuvent extraire de nouvelles relations (de tout type) à partir de leurs trois corpus de test en anglais que sont le *Wall Street Journal* (du Penn Treebank), Wikipedia et des textes généralistes issus du Web.

L'apport de l'analyse en dépendances est clairement établi dans la littérature. En revanche, son fort coût en temps (et en espace) lui est reproché. Ceci est vrai pour les analyses supervisées, et l'adaptation à une nouvelle langue de ces parsers nécessite un travail important de redéfinition manuelle des paramètres de l'outil. Par notre méthode DGdg d'analyse non supervisée, nous proposons de passer outre ces difficultés de coût et d'adaptation, en obtenant des arbres certes de moins bonne qualité. Nous cherchons dans ce chapitre à tester si la structure de nos arbres, différente de celle des arbres « académiques », est une plus-value pour la tâche d'extraction d'informations telle que nous l'avons choisie, par rapport aux séquences de catégories seules. En effet, nous continuons de ne travailler que sur les catégories grammaticales, et non sur les mots lexicaux, afin de garder la logique de légèreté et d'adaptabilité des processus que nous avons adoptée jusqu'à présent.

Nous cherchons dans cette application de nos travaux à identifier, parmi les mots des phrases grammaticalement étiquetées, les éléments des triplets d'information « intéressante » (SUJET, PRÉDICAT, OBJET). Nous annotons à la main quelques exemples, afin de procéder à un apprentissage supervisé. Le but de notre étude est de détecter la différence quantitative, en termes de rappel, précision et taux d'annotation correct de ces étiquettes cibles SUJET, PRÉDICAT et OBJET, entre un apprentissage qui tient compte de la structure d'arbre de dépendances de la phrase et un apprentissage qui l'ignore. Nous cherchons ici de plus à différencier les modèles appris sur des arbres de dépendances reconnus comme corrects, considérés par nous comme des arbres de référence, et les modèles appris sur les arbres construits par notre méthode DGdg, dont les attachements sont différents de ceux des arbres de référence. Nous reviendrons plus en détails sur les modalités de nos expérimentations et le contexte du domaine de recherche dans lequel nous nous plaçons, dans la section 5.3.

5.2 CRF, Conditional Random Fields

Pour la tâche spécifique de prédiction de variables cibles (comme par exemple l'étiquetage en entités nommées (personnes, lieux, organisations...) à partir de mots de phrases), les modèles bayésiens naïfs font des hypothèses très fortes d'indépendance des variables d'observation. Or ces hypothèses ne sont pas vérifiées, dans un grand nombre de cas. Par exemple, dans la tâche de segmentation d'images (une image partagée en pixels dont on connaît la couleur, on veut attribuer à chacun la bonne classe : pixel du ciel, pixel de la plage, pixel de la mer...), les couleurs de pixels

voisins sont très corrélées. Pour ces tâches, des modèles ne cherchant pas à modéliser la distribution des variables d'observation sont mieux adaptés, les hypothèses d'indépendance n'ayant plus lieu d'être formulées. Les CRF (Conditional Random Field) introduits dans l'article [Lafferty *et al.*, 2001] modélisent la probabilité conditionnelle des variables cibles (Y_j), sachant les variables d'observation (X_i). Nous nous inspirons ici des notations de l'article [Constant *et al.*, 2011]. Les CRF modélisent ainsi la distribution conditionnelle $p(Y|X)$ avec pour objectif de prédire une séquence $y^* = \underset{y}{\operatorname{argmax}} P(y|x)$, en ignorant les corrélations entre les variables d'observation et en ne se souciant pas de savoir si elles sont indépendantes ou non.

Les CRF appartiennent à la famille des modèles graphiques, à savoir des distributions complexes sur de nombreuses variables, modélisées comme le produit de facteurs locaux sur de plus petits ensembles de variables. Les CRF sont aussi une généralisation des modèles de régression logistique, et un cas spécifique de modèle log-linéaire.

Pour un CRF, étant donné le paramétrage $\lambda = (\lambda_1, \dots, \lambda_m)$, la probabilité conditionnelle $p(y|x, \lambda)$ s'exprime de la façon suivante :

$$p(y|x, \lambda) = \frac{\prod_{c \in C} \exp(\sum_k \lambda_k f_k(y_c, X, c))}{Z(x, \lambda)} \quad (5.1)$$

avec :

- C l'ensemble des cliques du graphe G choisi pour le modèle de CRF. Les nœuds du graphe G sont les variables d'observation X_i et les variables cible Y_j . Les arêtes caractérisent le modèle CRF choisi. Il existe des arêtes entre chaque variable cible et toutes les variables d'observation. Dans le cas des CRF linéaires, les variables cibles sont reliées entre elles par paires, i.e. les cliques c sont formées au maximum de deux variables successives.
- y_c sont les valeurs des variables du champ Y sur la clique c pour un étiquetage y . Dans le cas d'un modèle CRF linéaire, les y_c ont pour valeurs y_{i-1} et y_i ou y_i et y_{i+1} .
- $Z(x, \lambda)$ est un facteur de normalisation globale.
- les fonctions f_k sont les fonctions de features. Les fonctions de features expriment des caractéristiques de la distribution empirique que l'utilisateur souhaite prendre en compte dans la distribution du modèle. Elles peuvent être à valeurs réelles, mais sont souvent binaires (à valeurs dans $\{0;1\}$). Les modèles de CRF permettent l'utilisation de fonctions de features très expressives, sans se préoccuper de la dépendance ou l'indépendance des observations entre elles.
- Les poids λ_k sont les paramètres du modèle λ et permettent d'attribuer plus ou moins d'importance aux fonctions de features f_k .

Les CRF permettent un apprentissage supervisé du paramétrage λ en fonction de données d'apprentissage (observation et étiquetage cible) afin d'étiqueter ensuite en cibles de nouvelles observations.

- On procède d'abord à l'apprentissage à partir de données :
 - l'utilisateur choisit des fonctions de features expressives, adaptés à sa tâche d'étiquetage ;
 - l'utilisateur fournit des données étiquetées pour l'apprentissage ;
 - les paramètres λ_k du modèle sont déterminés pour s'adapter le mieux possible aux données, les fonctions de features étant fixées. L'apprentissage se fait généralement par maximisation de la log vraisemblance sur le corpus.

- Ensuite on propose une analyse à partir du modèle appris (de paramétrage λ), sur de nouvelles données d'observation non étiquetées. Étant données des observation x , la détermination des étiquettes cibles $y^* = \underset{y}{\operatorname{argmax}} P(y|x)$ maximisant la probabilité conditionnelle $P(y|x)$, calculée dans le modèle appris, donne la solution au problème d'étiquetage.

Les CRF ont à l'origine, dans [Lafferty *et al.*, 2001], été comparés favorablement par rapport aux HMM (Hidden Markov Models) et aux MEMM (Maximum Entropy Markov Models) sur la tâche d'étiquetage syntaxique. Plus tard, [Sha et Pereira, 2003] présente une méthode d'analyse syntaxique de surface par les CRF, ce qui permet un peu plus qu'un étiquetage syntaxique, à savoir une partie segmentation de la phrase en chunks. La tâche de reconnaissance d'entités nommées est elle aussi une application classique des CRF. Un étiquetage par des tags « début d'entité », « milieu d'entité » et éventuellement « fin d'entité » peut être proposé, et une sélection de fonctions de features permet de choisir les critères de reconnaissance de l'entité nommée. McCallum et Li ont proposé une utilisation en ce sens des CRF dans [McCallum et Li, 2003], pour une participation à la shared task de reconnaissance d'entités nommées CoNLL 2003 [Tjong Kim Sang et De Meulder, 2003].

Dans [Etzioni *et al.*, 2011] sont utilisés les CRF sur les séquences de mots lexicaux afin d'extraire des relations non typées. Durant la phase d'apprentissage, un extracteur de relations est appris par un modèle graphique d'étiquetage de séquences (ici, CRF). La contribution principale de ces auteurs est la modélisation de la façon d'exprimer une relation en anglais. Wu et Weld, dans [Wu et Weld, 2010], proposent un modèle sans analyse de dépendances, moins efficace, agissant par extracteur CRF avec des fonctions de features de surface sur les catégories grammaticales.

Le modèle généralement implémenté est un CRF linéaire (sur les chaînes). Les liens entre les étiquettes sont dans ce cas linéaires.

Il existe aussi quelques implémentations de CRF pour les arbres ([Gilleron *et al.*, 2008] et [Cohn et Blunsom, 2005]). Nous expliquerons dans la section suivante pourquoi nous n'avons pas choisi ce modèle à appliquer sur nos arbres de dépendances.

5.3 Utilisation des CRF et des arbres de dépendances construits par la méthode DGdg

5.3.1 Définition de la tâche

Ne souhaitant pas travailler sur le vocabulaire, et ne pouvant pas annoter de très grandes quantités de texte, nous nous sommes naturellement placé dans ce cadre de l'Open Information Extraction [Etzioni *et al.*, 2011].

Wu et Weld [Wu et Weld, 2010] cherchent des triplets de type (arg_1, rel, arg_2) où arg_1 et arg_2 sont des groupes nominaux et rel est un ensemble de mots indiquant une relation sémantique entre les 2 groupes nominaux. Ils ont à leur disposition 3 corpus distincts, WSJ du Penn Treebank, Wikipedia et un corpus généraliste issu du web. Cet ensemble de données est formé d'environ 300 000 phrases. Par une heuristique, en utilisant les infobox de Wikipedia, un premier outil (nommé matcher) signale les relations à extraire dans les corpus. Le moteur d'apprentissage apprend ensuite des patterns d'étiquettes grammaticales sur les corpus annotés. Dans chaque corpus, les auteurs ont sélectionné aléatoirement 300 phrases sur lesquelles ils ont pratiqué l'évaluation. Ces phrases

ont été examinées par deux personnes qui ont étiqueté tous les triplets pertinents. Ces candidats triplets ont été mêlés avec des triplets pseudo-négatifs, et soumis à validation à Amazon Mechanical Turk⁴. Chaque ensemble de 300 phrases a ensuite été divisé en 5. Les résultats annoncés sont la meilleure F-mesure calculée parmi les 5 sous-ensembles de phrases.

Dans l'objectif d'une évaluation moins onéreuse, nous avons manuellement annoté 100 phrases issues du French Treebank [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010], chaque mot étant étiqueté « PRED » pour prédicat, l'équivalent de *rel* dans les travaux de Wu et Weld, « SUJ » pour sujet, restreignant le rôle de arg_1 , « OBJ » pour objet, arg_2 plus spécifique, et « RIEN » pour les mots ne participant à aucun triplet. Pour évaluer de façon quantitative l'étiquetage attribué par les CRF, nous excluons du décompte les étiquettes RIEN des ponctuations, faciles à retrouver par tout modèle et n'apportant pas d'informations.

5.3.2 Données et choix du modèle de CRF

Les 100 phrases étiquetées, comprenant 3085 mots (incluant les ponctuations), ont été partagées en 10 sous-corpus de 10 phrases, afin de procéder à une évaluation par validation croisée du type « leave-one-out cross-validation ».

La tâche d'étiquetage est complexe. La sélection de triplets du type (groupe nominal sujet, verbe, groupe nominal objet) s'avère trop restrictive du fait de la richesse de l'écriture des phrases et du manque de clarté de ce type de triplets. Nous avons finalement étiqueté chaque mot en fonction du rôle qui nous semblait le plus adapté, sans pour le moment nous soucier de le situer au sein d'un triplet. Nous donnons à la suite l'exemple d'une phrase que nous avons annotée. Les informations qui nous sont disponibles grâce aux arbres de dépendances ont été ajoutées à la catégorie grammaticale du mot considéré dans l'apprentissage et le test par les CRF. Ces informations susceptibles d'être intéressantes pour l'apprentissage par les CRF sont des attributs, nous disposons donc d'une description attribut-valeur des variables d'observation. Nous donnons ici la liste des attributs des observations, ainsi que la variable cible

- la position du mot dans la phrase (attribut) ;
- le mot présent dans le texte (attribut) ;
- son lemme (attribut) ;
- son étiquette grammaticale à gros grain (attribut) ;
- son étiquette grammaticale à grain fin (attribut) ;
- l'indice du parent en dépendances (attribut) ;
- la catégorie à gros grain de ce parent (s'il existe) (attribut) ;
- la catégorie à gros grain du grand-parent (s'il existe) (attribut) ;
- la profondeur du mot dans l'arbre de dépendances (attribut) ;
- l'étiquetage que nous avons attribué dans l'objectif d'extraire de l'information (cible).

La Table 5.1 montre une phrase du corpus et un exemple de notre étiquetage en cibles (PRED, SUJ, OBJ et RIEN). L'arbre de dépendances est représenté de façon graphique par la Figure 5.2. La phrase est « À ces cinq **départs** du bureau confédéral, **trois autres sont susceptibles de s'ajouter** : ceux de **Mme Jacqueline Léonard** et de **MM. Johannès Galland** et **Jean-Claude Laroze**. ». Nous avons choisi l'étiquetage en cibles en fonction des informations essentielles délivrées par cette

4. <https://www.mturk.com/mturk/welcome>

phrase. Le sens de la phrase est, dans notre interprétation, porté par les mots en gras. Ces informations ont pour sens :

- le fait qu’il y ait des départs (mot 4 « départs » étiqueté par SUJ) ;
- le fait que soient envisagés trois autres départs (mots 9, 10, 11, 12 puis 15, « trois autres sont susceptibles » « ajouter » étiquetés respectivement par SUJ, SUJ, PRED, OBJ et PRED) ;
- les noms de ces partants (mots 19, 20, 21 puis 24, 25, 26 puis 28 et 29, « Mme Jacqueline Léonard », « MM. Johannès Galland » et « Jean-Claude Laroze »).

Les triplets (non nécessairement complets) que nous extrayons de cette phrase sont :

- (SUJ : départs, PRED : X, OBJ : X) ;
- (SUJ : trois autres, PRED : sont, OBJ : susceptibles) ;
- (SUJ : X, PRED : ajouter, OBJ : (Mme Jacqueline Léonard, MM. (Johannès Galland, Jean-Claude Laroze))).

Nous remarquons ici la complexité des tâches d’étiquetage et de reconstitution des triplets ensuite. Les triplets n’apparaissent pas nécessairement complets, plusieurs mots peuvent être regroupés sous la même étiquette, y compris des mots pouvant être séparés par d’autres mots non porteurs de sens essentiel.

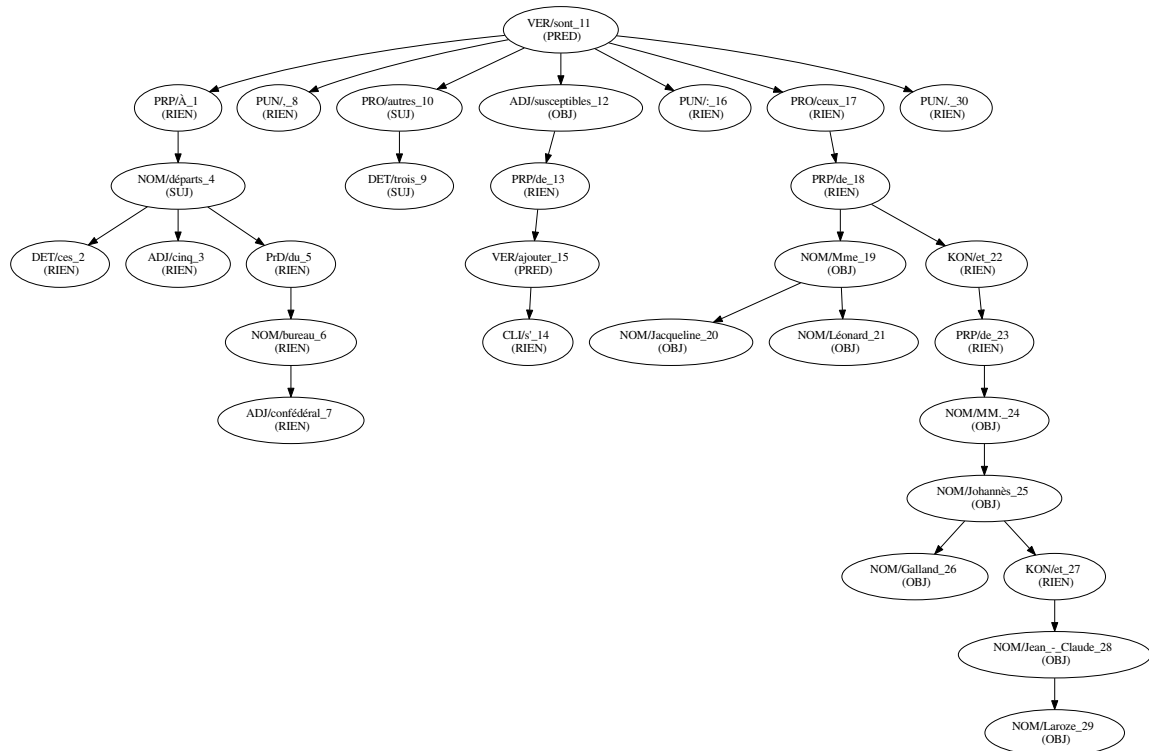


FIGURE 5.2 : Arbre de dépendances (REF) de la phrase de la Table 5.1.

Indice	Mots	Lemmes	catGROS	catFIN	IndParent	catParent	catGdParent	Prof	Cible (étiquette)
1	À	à	P	P	11	V	–	1	RIEN
2	ces	ce	DET	DET	4	NOM	P	3	RIEN
3	cinq	cinq	A	ADJ	4	NOM	P	3	RIEN
4	départs	départ	NOM	NOM	1	P	V	2	SUJ
5	du	de	P+D	P+D	4	NOM	P	3	RIEN
6	bureau	bureau	NOM	NOM	5	PpD	NOM	4	RIEN
7	confédéral	confédéral	A	ADJ	6	NOM	PpD	5	RIEN
8	,	,	PONCT	PONCT	11	V	–	1	RIEN
9	trois	trois	DET	DET	10	PRO	V	2	SUJ
10	autres	autre	PRO	PRO	11	V	–	1	SUJ
11	sont	être	V	V	0	–	–	0	PRED
12	susceptibles	susceptible	A	ADJ	11	V	–	1	OBJ
13	de	de	P	P	12	A	V	2	RIEN
14	s'	s'	CL	CLO	15	V	P	4	RIEN
15	ajouter	ajouter	V	VINF	13	P	A	3	PRED
16	:	:	PONCT	PONCT	11	V	–	1	RIEN
17	ceux	celui	PRO	PRO	11	V	–	1	RIEN
18	de	de	P	P	17	PRO	V	2	RIEN
19	Mme	madame	NOM	NOM	18	P	PRO	3	OBJ
20	Jacqueline	Jacqueline	NOM	NOM	19	NOM	P	4	OBJ
21	Léonard	Léonard	NOM	NOM	19	NOM	P	4	OBJ
22	et	et	C	CC	18	P	PRO	3	RIEN
23	de	de	P	P	22	C	P	4	RIEN
24	MM.	MM.	NOM	NOM	23	P	C	5	OBJ
25	Johannès	Johannès	NOM	NOM	24	NOM	P	6	OBJ
26	Galland	Galland	NOM	NOM	25	NOM	NOM	7	OBJ
27	et	et	C	CC	25	NOM	NOM	7	RIEN
28	Jean-Claude	Jean-Claude	NOM	NOM	27	C	NOM	8	OBJ
29	Laroze	Laroze	NOM	NOM	28	NOM	C	9	OBJ
30	.	.	PONCT	PONCT	11	V	–	1	RIEN

TABLE 5.1 : Format d'une phrase donnée en entrée de l'apprentissage par CRF++.

Notre objectif principal est de montrer l'apport des arbres de dépendances, en particulier de nos arbres DGdg, par rapport au texte non structuré, pour extraire de l'information. Nous voulons plus précisément exploiter les informations issues des arbres de dépendances et non du seul texte grammaticalement étiqueté.

Nous nous sommes naturellement dirigés vers les outils CRF spécifiques aux arbres, présentés dans les travaux de [Gilleron *et al.*, 2008] et [Cohn et Blunsom, 2005]. Mais la structure d'arbre implique une complexité des calculs qui ne permet pas d'exploiter des informations au-delà du parent ou des frères immédiats du mot, vu comme un nœud. Finalement, considérer chaque observation comme un vecteur comportant toutes les informations que nous voulions utiliser nous a fait choisir le modèle plus simple des CRF linéaires sur les chaînes. Nous considérons ces chaînes augmentées par des informations émanant des arbres de dépendances.

5.3.3 Les différents patrons de CRF testés

Formellement, nous considérons ces données (illustrées par un exemple dans la Table 5.1) comme des séquences de vecteurs d'observations, de dimension 9 (les 9 premières colonnes de la Table 5.2), les coefficients des vecteurs sont les contenus des colonnes, on les considère comme des attributs de la variable d'observation. On considère la dernière colonne comme l'étiquetage cible du vecteur d'observations formé par les 9 premières colonnes.

Nous avons choisi d'utiliser l'implémentation CRF++⁵. Le corpus de 100 phrases manuellement étiquetées est partagé en 90 phrases pour la partie apprentissage, et 10 phrases pour la partie test. On effectue 10 partages de ce type afin de mieux estimer la justesse du modèle et multiplier artificiellement nos données peu nombreuses, de sorte à effectuer une validation croisée. Nous donnons la moyenne des résultats sur les 10 corpus de test de tailles similaires (environ 300 mots). Nous avons cherché à retrouver les étiquettes « PRED », « SUJ », « OBJ » et « RIEN » lorsque l'information ne nous paraissait pas intéressante à retenir. Notre étiquetage compte, sur ces 100 phrases, 202 mots étiquetés « PRED », 221 étiquetés « SUJ », 449 « OBJ » et 2113 « RIEN ».

Les patrons sont la donnée initiale des fonctions de features à considérer pour un modèle de CRF à apprendre, sans pondération. Pour chaque patron existe une version « unigramme », et une version « bigramme ». Nous attribuons un sens particulier et personnel à ces qualificatifs, que nous expliquons dans la remarque 5.2 (page 82). Nous avons testé systématiquement les versions unigramme et bigramme de chaque patron.

Nos patrons se divisent en deux types : ceux qui prennent en compte les informations acquises par la structuration en arbres de la phrase (les patrons TREE) et ceux qui se cantonnent aux informations disponibles dans la séquence de mots non hiérarchisée (les patrons SEQ).

Nous n'avons testé que deux paires (unigramme et bigramme) de patrons SEQ car nous avons choisi de n'utiliser que les catégories grammaticales à gros grain pour l'instant dans ces expériences limitées aux phrases vues comme des séquences. Nous avons en revanche utilisé 11 (paires de) patrons TREE, pour étudier l'influence de changements de fonctions de features, légers ou importants.

Indice 1	Mots 2	Lemmes 3	catGROS 4	catFIN 5	IndParent 6	catParent 7	catGdParent 8	Prof 9	Cible(étiquette) 10
-------------	-----------	-------------	--------------	-------------	----------------	----------------	------------------	-----------	------------------------

TABLE 5.2 : 1, Indice : position du mot dans la phrase, 2, Mots : mot présent dans le texte, 3, Lemmes : lemme, 4, catGROS : étiquette grammaticale à gros grain, 5, catFIN : étiquette grammaticale à grain fin, 6, IndParent : indice du parent en dépendance, 7, catParent : catégorie à gros grain du parent (s'il existe), 8, catGdParent : catégorie à gros grain du grand-parent (s'il existe), 9, Prof : profondeur du mot dans l'arbre de dépendances et 10, Cible : étiquetage que nous avons choisi.

Nous définissons ici les fonctions de features unitaires que nous avons utilisées dans nos patrons de CRF (linéaires). Les notations X et Y indiquent respectivement les variables d'observation (vecteur) et cible (étiquette). La notation X_m^n correspond à l'attribut n (souvent colonne 3 qui donne la catégorie à gros grain) du m -ième vecteur d'observation et Y_m est l'étiquette cible du m -ième

5. <http://taku910.github.io/crfpp/>

vecteur d'observation. Les indices x et y désignent des instances particulières respectivement des variables X et Y .

$$f_{x,y}^{m,n}(X, Y) = \begin{cases} 1 & \text{si } X_m^n = x \text{ et } Y_m = y \\ 0 & \text{sinon} \end{cases} \quad (5.2)$$

Remarque 5.1 : valeurs de l'indice m

L'indice m est une position relative à l'observation courante pendant le parcours des données au moment de l'apprentissage comme de l'analyse. Il vaut 0 si l'on fait référence à l'observation courante, il est négatif si l'on fait référence à une observation antérieure (-1 pour la précédente, -2 pour la précédente de la précédente...) et positif si l'on fait référence à une observation postérieure (1 pour la suivante, 2 pour la suivante de la suivante...).

Par exemple, la fonction de feature $f_{ADV,PRED}^{0,catGros}$ renvoie 1 si l'attribut *catGros* de l'observation en cours a pour valeur *ADV* et que son étiquette cible est *PRED*. Sinon, si l'attribut *catGros* de l'observation en cours a une autre valeur ou que son étiquette cible est différente de *PRED*, alors la fonction de feature renvoie 0.

Le produit de fonctions de features unitaires de ce format $f_{x,y}^{m,n} \times f_{x',y'}^{m',n'}$ est une nouvelle fonction de feature, binaire elle aussi, que l'on notera $f_{[x,y][x',y']}^{[m,n][m',n']}$:

$$f_{[x,y][x',y']}^{[m,n][m',n]}(X, Y) = f_{x,y}^{m,n}(X, Y) \times f_{x',y'}^{m',n'}(X, Y) = \begin{cases} 1 & \text{si } X_m^n = x \text{ et } Y_m = y \text{ et } X_{m'}^{n'} = x' \text{ et } Y_{m'} = y' \\ 0 & \text{sinon} \end{cases}$$

Pour une meilleure appréhension des patrons, nous remplacerons dans les notations l'indice n par la nature de l'indication qu'il fournit. Notons également l'ensemble $f^{m,n}$ des fonctions de features tel que $f^{m,n} = \{f_{x,y}^{m,n}, \forall x, \forall y\}$. Cet ensemble de fonctions de features contient toutes les fonctions de features définies à partir de toutes les valeurs x existant pour le n -ième attribut des observations, et toutes les étiquettes cibles y . Dans le cadre de notre expérimentation, nous rappelons que les valeurs y sont « PRED », « SUJ », « OBJ » et « RIEN », et précisons que les valeurs de x , lorsque $n = 3$, sont toutes les catégories grammaticales à gros grain présentes dans le corpus (pour le French Treebank, A, ADV, C,... voir page 50 pour la liste complète des 15 catégories à gros grain).

Exemple 5.1 : ensembles de fonctions de features

Comme précisé précédemment, nous remplacerons la valeur chiffrée de l'indice n (indice de l'attribut de l'observation, i.e. numéro de la colonne) par le titre de la colonne qu'il représente.

L'ensemble $f^{0,catGros}$ désigne alors toutes les fonctions de features entre accolades dans la formule suivante :

$$f^{0,catGros} = \left\{ \begin{array}{l} f_{A,PRED}^{0,catGros}, f_{A,SUJ}^{0,catGros}, f_{A,OBJ}^{0,catGros}, f_{A,RIEN}^{0,catGros}, \\ f_{ADV,PRED}^{0,catGros}, f_{ADV,SUJ}^{0,catGros}, f_{ADV,OBJ}^{0,catGros}, f_{ADV,RIEN}^{0,catGros}, \\ \dots, \\ f_{V,PRED}^{0,catGros}, f_{V,SUJ}^{0,catGros}, f_{V,OBJ}^{0,catGros}, f_{V,RIEN}^{0,catGros} \end{array} \right\}$$

Cet ensemble est composé de $4 \times 15 = 60$ fonctions de features différentes.

L'ensemble noté $f^{[0,catGros][0,catParent]}$ désigne alors tous les produits possibles entre les fonctions de features des ensembles $f^{0,catGros}$ et $f^{0,catParent}$, soit un ensemble de presque $60^2 = 3\,600$ fonctions de features (puisque les $catParent$ sont les mêmes 15 catégories à gros grain que $catGros$, mais ce sont les catégories du parent en dépendances de l'observation courante).

Une simple notation $f^{[0,catGros][0,catParent]}$ désignant un si grand nombre de fonctions de features, il est essentiel que nous signalions qu'au cours de la procédure d'apprentissage, les fonctions de features non pertinentes par rapport aux données d'apprentissage seront affectés d'un coefficient $\lambda_{[0,catGros][0,catParent]} = 0$ et éliminés de l'ensemble des fonctions de features conservées pondérées.

Nous avons choisi 3 combinaisons d'ensembles de fonctions de features, pour un même attribut n , au sein des patrons.

- Combinaison $C_1(n)$:
 - + $f^{-2,n}, f^{-1,n}, f^{0,n}, f^{1,n}$ et $f^{2,n}$, (fonctions de features unitaires) ;
 - + $f^{[-2,n][-1,n]}, f^{[-1,n][0,n]}, f^{[0,n][1,n]}$ et $f^{[1,n][2,n]}$, (fonctions de features produits de deux fonctions de features unitaires) ;
 - + $f^{[-2,n][-1,n][0,n]}$ et $f^{[0,n][1,n][2,n]}$, (fonctions de features produits de trois fonctions de features unitaires).

Cette combinaison $C_1(n)$ des ensembles de fonctions de features est proposée comme exemple pour la tâche de segmentation en groupes syntaxiques dans le challenge CoNLL 2000. Elle tient compte, indépendamment des observations en cours, des deux observations précédentes et des deux observations suivantes (pour l'attribut n , par exemple l'attribut 4 qui est la catégorie à gros grain de l'observation en cours). Elle lie aussi par paires et par triplets les observations successives. C'est la proposition de base que nous avons choisie. Nous proposons aussi les deux combinaisons suivantes, plus légères.

- Combinaison $C_2(n)$:
 - + $f^{-2,n}, f^{-1,n}, f^{0,n}, f^{1,n}$ et $f^{2,n}$, (fonctions de features unitaires) ;
 - + $f^{[-2,n][-1,n]}, f^{[-1,n][0,n]}, f^{[0,n][1,n]}$ et $f^{[1,n][2,n]}$, (fonctions de features produits de deux fonctions de features unitaires).

Nous gardons les fonctions de features unitaires et produits de deux fonctions de features unitaires de $C_1(n)$ et enlevons les produits de trois fonctions de features unitaires. Ceci permet d'avoir un patron moins complexe qui présente moins de risques de sur-ajustement.

- Combinaison $C_3(n)$:
 - + $f^{-2,n}, f^{-1,n}, f^{0,n}, f^{1,n}$ et $f^{2,n}$, (fonctions de features unitaires) ;
 - + $f^{[-1,n][0,n]}$ et $f^{[0,n][1,n]}$, (fonctions de features produits de deux fonctions de features unitaires).

Par rapport à la combinaison $C_2(n)$, $C_3(n)$ conserve toutes les fonctions de features unitaires, mais élimine les liens établis par les produits de deux fonctions de features unitaires entre les observations précédant la précédente et celles succédant à la suivante.

Nous avons utilisé ces combinaisons de fonctions de features sur l'attribut des catégories grammaticales à gros grain. Pour un seul patron de notre expérimentation, nous avons testé l'utilisation

de ces combinaisons aussi sur l'attribut des catégories grammaticales à gros grain du parent de l'observation en cours.

Ces combinaisons sont liées à une colonne n , c'est-à-dire à un descripteur du mot. Ces combinaisons nous ont servi de base pour les patrons. Nous avons voulu tenir compte des interactions entre les différents descripteurs (des différents attributs) pour enrichir nos patrons. La Table 5.3 liste les patrons testés, l'attribut « Combinaisons » indique laquelle des trois combinaisons nous a servi de base. L'attribut « Liens entre colonnes » indique les ensembles de produits de fonctions de features que nous avons éventuellement ajoutés pour lier dans les patrons CRF les différentes informations à disposition, en particulier les descripteurs connus grâce aux arbres de dépendances.

Remarque 5.2 : patrons unigrammes et bigrammes

Les fonctions de features unitaires définies par l'équation 5.2 sont unigrammes. C'est-à-dire qu'ils ne tiennent compte que de l'étiquette cible de l'observation en cours. Pour les CRF linéaires, la formule 5.1 permet que les cliques soient formées de 2 étiquettes cibles successives. C'est ce que font les patrons bigrammes. Dans nos expériences, nos patrons bigrammes sont constitués des templates des patrons unigrammes de mêmes noms, auxquels on ajoute des fonctions de features qui tiennent compte de la succession des étiquettes cibles sans tenir compte des observations.

Pratiquement, on ajoute au patron unigramme les fonctions de features suivantes, pour toutes les observations possibles y_1 et y_2 :

$$f_{y_1, y_2}^m(Y) = \begin{cases} 1 & \text{si } Y_m = y_1 \text{ et } Y_{m-1} = y_2 \\ 0 & \text{sinon} \end{cases} \quad (5.3)$$

La Table 5.3 détaille les patrons testés. Les variations sur les combinaisons (et sur le caractère bigramme ou unigramme des patrons) teste la valeur ajoutée en complexifiant les patrons. Est-il intéressant de considérer plus d'observations (pour le même descripteur) ou de complexifier le patron en considérant les bigrammes sur les cibles ?

La troisième colonne constitue le centre de nos interrogations. Quelles informations (catégorie de l'observation, de son parent, de son grand-parent, profondeur dans l'arbre) sont à considérer pour optimiser le modèle et obtenir de meilleurs résultats ? Et comment les lier entre elles ? La troisième colonne de la Table 5.3 résume les expériences avec différentes informations. Nous avons considéré des patrons simples (SEQ) sans introduire d'informations issues de la structure d'arbre (uniquement la catégorie à gros grain de l'observation), nous avons gardé cette base dans tous les patrons. Nous avons aussi essayé des patrons liant la catégorie de l'observation avec la catégorie de son parent (TREE(a,b,c,m)parent). Nous avons ajouté encore à ce patron une fonction de feature liant en plus la catégorie de l'observation à celle de son grand-parent (TREEeparentGP). Ce patron a encore été enrichi du lien entre la catégorie du parent et du grand-parent (TREE(f,k)parentGP). Nous avons également testé un patron liant la catégorie de l'observation à sa profondeur dans l'arbre de dépendances (TREE(g,j)prof), auquel ont été ajoutés des liens avec la catégorie du parent (TREE(h,l)parentPROF), puis aussi avec la catégorie du grand-parent (TREEiparentGPPROF).

Patrons	Combinaisons	Liens entre colonnes
SEQa	$C_3(3)$	-
SEQb	$C_1(3)$	-
TREEaparent	$C_2(3)$	$f^{[0,catGros][0,catParent]}$
TREEbparent	$C_3(3)$ et $C_3(6)$	-
TREEcparent	$C_3(3)$	$f^{[0,catGros][0,catParent]}$
TREEeparentGP	$C_3(3)$	$f^{[0,catGros][0,catParent]}$ et $f^{[0,catGros][0,catGdParent]}$
TREEfparentGP	$C_3(3)$	$f^{[0,catGros][0,catParent]}$ et $f^{[0,catGros][0,catGdParent]}$ et $f^{[0,catParent][0,catGdParent]}$
TREEgprof	$C_3(3)$	$f^{[0,catGros][0,Prof]}$
TREEhparentPROF	$C_3(3)$	$f^{[0,catGros][0,catParent]}$ et $f^{[0,catGros][0,Prof]}$
TREEiparentGPPROF	$C_3(3)$	$f^{[0,catGros][0,catParent]}$ et $f^{[0,catGros][0,catGdParent]}$ et $f^{[0,catGros][0,Prof]}$
TREEjprof	$C_1(3)$	$f^{[0,catGros][0,Prof]}$
TREEkparentGP	$C_2(3)$	$f^{[0,catGros][0,catParent]}$ et $f^{[0,catGros][0,catGdParent]}$ et $f^{[0,catParent][0,catGdParent]}$
TREElparentPROF	$C_2(3)$	$f^{[0,catGros][0,catParent]}$ et $f^{[0,catGros][0,Prof]}$ et $f^{[0,catParent][0,Prof]}$
TREEUmparent	$C_2(3)$	$f^{[-1,catGros][-1,catParent]}$ et $f^{[0,catGros][0,catParent]}$ et $f^{[1,catGros][1,catParent]}$

TABLE 5.3 : Les patrons de CRF testés dans nos essais.

Les combinaisons possibles de fonctions de features sont vastes, nous avons fait varier peu de paramètres dans cette première série d'expériences. Nous souhaitons constater l'influence des ajouts d'information, en plus de la catégorie grammaticale du mot, à savoir la catégorie grammaticale du parent, celle du grand-parent, et l'indication de la profondeur dans l'arbre de dépendances.

5.3.4 Résultats et modalités d'évaluation

Comme précisé dans la sous-section 5.3.2, nous avons procédé à une évaluation par cross validation du type « leave-one-out ». La phase d'analyse avec chaque modèle CRF appris a conduit à une nouvelle annotation en étiquettes cibles. Pour obtenir une vue d'ensemble de la qualité de l'annotation, nous avons calculé, pour chaque expérience, un taux de bonne classification sur toutes les étiquettes, en excluant des comptages les étiquettes « RIEN » des catégories de ponctuation. Les scores reportés dans la Table 5.4 sont une moyenne des résultats obtenus dans les dix expériences (90% apprentissage et 10% test tournants) pour chaque modèle. L'indication U ou B dans le nom du patron indique le caractère unigramme ou bigramme.

Pour une évaluation plus ciblée sur la qualité de l'annotation par étiquettes, nous avons aussi calculé le rappel et la précision pour chaque étiquette cible (toujours en excluant les ponctuations) pour les 10 expériences pour chaque patron. La moyenne sur les dix expériences a ensuite été calculée. La Table 5.5 présente, en différenciant la manière de générer les arbres de dépendances, les moyennes de ces moyennes calculées sur les 26 modèles. Une dernière ligne (SEQ) a été ajoutée, pour comparaison, donnant les moyennes sur tous les patrons SEQ, i.e. ne tenant pas compte des informations issues des arbres de dépendances.

Nous désignerons abusivement les corpus dont les arbres ont été construits par la méthode DGdg par le nom de la variante utilisée (4bin, 4ter, 5ter et 5ter+), ainsi que par REF le corpus dont les arbres ont été donnés comme référence dans le French Treebank [[Abeillé et Barrier, 2004](#) ; [Candito et al., 2010](#)].

Patrons	arbres obtenus par DGdg					arbres de dépendances	
	3bin	4bin	4ter	5ter	5ter+	référence (REF)	sans arbres
SEQBa	-	-	-	-	-	-	80.75
SEQBb	-	-	-	-	-	-	80.67
SEQUa	-	-	-	-	-	-	81.13
SEQUb	-	-	-	-	-	-	80.95
TREEUa	80.89	80.94	80.58	81.07	80.92	82.50	
TREEBa	80.98	80.47	80.45	80.65	80.73	82.05	
TREEUb	80.89	80.16	79.52	81.25	79.21	82.26	
TREEBb	80.41	79.71	79.60	80.44	78.84	81.74	
TREEUc	80.86	80.33	79.72	80.78	79.07	82.22	
TREEBc	80.25	79.70	79.70	80.11	78.56	81.71	
TREEUeparentGP	81.18	81.10	80.30	80.38	80.83	82.33	
TREEBeparentGP	80.67	81.01	80.01	80.42	81.28	81.80	
TREEUfparentGP	81.00	81.18	80.22	80.50	80.68	82.24	
TREEBfparentGP	80.30	81.15	79.89	80.06	81.29	81.61	
TREEUgprof	81.94	82.07	80.78	80.54	80.58	81.93	
TREEBgprof	81.77	81.87	79.86	80.34	80.14	81.70	
TREEUhparentPROF	81.61	82.28	80.54	80.91	80.70	82.01	
TREEBhparentPROF	80.97	81.59	79.93	80.39	80.88	82.53	
TREEUiparentGPPROF	81.17	82.38	80.23	80.41	80.39	81.86	
TREEBiparentGPPROF	80.59	81.66	79.92	80.11	80.21	81.58	
TREEUjprof	81.78	81.81	80.57	80.68	80.70	82.11	
TREEBjprof	81.55	81.39	81.77	80.49	79.91	81.53	
TREEUlparentPROF	81.08	81.79	80.41	81.05	80.71	82.54	
TREEBlparentPROF	80.44	81.46	80.04	80.48	80.85	82.24	
TREEUmparent	80.47	80.93	80.18	80.49	80.49	82.57	
TREEBmparent	80.38	80.61	80.28	80.68	80.44	82.25	

TABLE 5.4 : Taux de bonne classification, en pourcentages, pour tous les patrons testés, sans tenir compte des étiquettes cibles des ponctuations. « référence » correspond à la version en dépendance du French Treebank [Abeillé et Barrier, 2004 ; Candito *et al.*, 2010], ces arbres de dépendances sont considérés comme exacts.

Les scores, présentés Table 5.4, sont tous très proches, autour de 80% de bonne classification sur toutes les étiquettes cibles (PRED, OBJ, SUJ et RIEN). Ces scores étant déjà assez élevés pour les patrons séquentiels (qui ne tiennent compte que de la catégorie grammaticale du mot, et non des informations issues des arbres), les progrès apportés par les informations liées aux arbres ne pouvaient qu’être mesurés. Dans les faits, le meilleur score (moyen sur l’évaluation par « Leave-One Out Cross Validation ») obtenu par un patron (SEQUa) sur les séquences de catégories est 81.13% de bonne classification.

- Valeur ajoutée des informations issues des arbres de dépendances **de référence** par rapport aux séquences de catégories seules :

Comme il est signalé en gras, dans chaque colonne de la Table 5.4, pour chaque corpus (dont les arbres sont donnés en référence et les arbres construits avec les variantes de la famille de grammaires DGdg), il existe au moins un patron utilisant les informations connues par les arbres (catégorie du parent, du grand-parent, profondeur dans l'arbre) qui majore ces 81.13%. En particulier, tous les résultats obtenus sur le corpus REF sont meilleurs que ceux des patrons ne tenant compte que des séquences de catégories morphosyntaxiques (colonne référence de la Table 5.4 entièrement en gras). Le patron TREEBhparentPROF (qui tient compte de la catégorie du parent et de la profondeur dans l'arbre) permet de retrouver 82.53% des étiquettes cibles sur le corpus REF.

- Valeur ajoutée des informations issues des arbres de dépendances **construits par les méthodes DGdg** par rapport aux séquences de catégories seules :

Nous rappelons, en consultant la Table 4.4 (page 62), que la grammaire 3bin obtient les « moins bons » arbres parmi les 5 grammaires DGdg, avec seulement 29.7% d'attachement corrects. La grammaire 4bin, légèrement meilleure avec 32.9% est encore assez loin des bons scores de 5ter (37.4), 4ter (42.1) et 5ter+ (42.2). Pourtant, les structures de ces arbres 3bin et 4bin, même si elles correspondent mal à la structure de référence attendue par des linguistes experts, semblent plutôt bien adaptées pour l'extraction d'informations telle que nous l'avons modélisée ici.

Ce sont, après les arbres de référence, les arbres 4bin qui obtiennent le deuxième meilleur score avec 82.38% de taux d'étiquetage cible correct, ceci par le patron TREEUiparentGPPROF qui tient compte de toutes les informations que nous avons choisies de fournir grâce aux arbres (catégorie du parent, du grand-parent et profondeur). Sur les 22 patrons TREE, 13 donnent de meilleurs résultats que le meilleur patron SEQ. La Table 5.5 indique même que la moyenne des scores, sur tous les patrons (SEQ et TREE), est de 81.16%, c'est-à-dire légèrement supérieure à 81.13%, le score du meilleur patron SEQ.

Les arbres 3bin ensuite obtiennent le troisième meilleur score avec 81.94% par le patron TREEUgPROF qui ne tient compte que de la profondeur du mot dans l'arbre, en plus de sa catégorie grammaticale. Et sur les 22 patrons TREE, 7 permettent de surpasser les scores du meilleur patron SEQ.

Puis les arbres 4ter, par le patron TREEBjprof (profondeur seulement) obtiennent 81.77%, les arbres 5ter+ et 5ter obtiennent des scores presque égaux avec 81.29% et 81.25% respectivement par les patrons TREEBfparentGP (catégorie du parent et du grand-parent) et TREEUb (qui tient compte de la catégorie du parent, mais ne fait pas le lien entre les catégories du parent et du grand-parent). On note que le corpus 5ter+ permet à 2 patrons (sur 22) de surpasser le meilleur patron SEQ. Un seul patron TREE obtient un meilleur score que les patrons SEQ sur le corpus 5ter. Et seul le patron TREEBjprof permet de surpasser les scores des patrons SEQ pour le corpus 4ter.

- Comparaison entre les résultats obtenus sur les corpus dont les arbres de dépendances sont ceux **de référence** et ceux **construits par les méthodes DGdg** :

Dans cette même Table 5.4 sont encadrés les résultats qui, pour un patron donné, sont meilleurs sur un corpus DGdg que sur le corpus REF. Ce sont 9 expériences sur 110 qui accomplissent cette performance d’obtenir un meilleur étiquetage grâce aux arbres DGdg que par les arbres de référence.

Sur les 22 patrons TREE, 5 entraînent des résultats, pour le corpus 4bin, qui surpassent ceux du corpus REF. Pour les arbres 3bin, 3 patrons TREE améliorent les scores par rapport au corpus REF. Seul le patron TREEBjprof permet aux scores obtenus sur le corpus 4ter de surpasser les scores obtenus sur le corpus REF. Et ni la variante 5ter de la méthode DGdg, ni 5ter+ ne permettent à un patron d’améliorer le score par rapport aux tests sur le corpus REF.

– Valeur ajoutée :

Au vu de ces éléments, il apparaît clairement que les informations liées aux arbres de référence enrichissent et améliorent les modèles CRF, nous rejoignons ainsi, par une méthode différente, la conclusion de Wu et Weld [Wu et Weld, 2010] présentée en section 5.1. En revanche, il n’est pour le moment pas envisageable d’établir de corrélation entre la « qualité UAS » des arbres de dépendances obtenus par les variantes de la méthode DGdg (relativement aux arbres donnés comme référence), et leur capacité à fournir des éléments appropriés pour une extraction d’informations riches et pertinentes.

5.3.5 Détail en fonction des étiquettes cibles et perspectives d’études

Une étude plus poussée, avec des patrons plus affinés, sur un corpus plus étendu, serait nécessaire pour améliorer plus notablement les résultats par des modèles adaptés, par rapport aux résultats obtenus seulement sur les séquences de catégories. Mais ces premiers résultats sont encourageants dans le sens où les informations apportées par les arbres (catégorie du parent, du grand-parent et profondeur), judicieusement combinées dans des patrons, améliorent les résultats par rapport aux séquences seules. Ceci valorise particulièrement nos travaux puisque même avec nos arbres de dépendances obtenus par les grammaires DGdg, de qualité dégradée, les résultats peuvent être meilleurs qu’avec seulement les séquences.

	score	rappel	précision	rappel	précision	rappel	précision	rappel	précision
		RIEN	RIEN	SUJ	SUJ	PRED	PRED	OBJ	OBJ
3bin	80.94	87.96	85.53	65.81	76.08	88.17	79.86	56.23	59.91
4bin	81.16	88.26	85.77	64.23	74.75	88.75	80.61	56.79	60.55
4ter	80.12	87.27	85.37	64.01	73.95	87.03	80.32	55.31	57.26
5ter	80.53	87.59	85.79	64.40	74.70	88.50	81.15	56.29	57.87
5ter+	80.39	87.77	85.42	64.90	73.72	88.82	80.52	54.06	57.95
REF	81.92	89.10	86.48	70.03	77.31	87.97	85.14	55.14	59.20
SEQ	80.87	88.00	85.39	65.10	77.58	89.67	80.72	55.20	58.94

(4 patrons)

TABLE 5.5 : Ces résultats sont des moyennes calculées sur les 26 patrons, par corpus d'arbres, y compris les patrons SEQ.

Pour préciser ces résultats moyens (sur l'évaluation par « Leave-One Out Cross Validation »), y compris avec les patrons SEQ en fonction des étiquettes cibles, nous avons détaillé dans la Table 5.5 les rappels et précisions relatifs à ces cibles. Les scores transcrits ici sont des moyennes, calculées sur les 26 patrons testés. Les écarts-types sont faibles (entre 0.003 et 0.026). Les résultats, filtrés par cible, restent du même ordre dans les différents modèles.

Logiquement, et en cohérence avec la constatation que les arbres de dépendances améliorent l'extraction d'informations [Wu et Weld, 2010], les arbres de référence obtiennent presque systématiquement les meilleurs résultats (moyens) pour tous les rappels et précisions par tous les filtres d'étiquettes cibles. Nous remarquons aussi que sur tous les corpus d'arbres (de référence et DGdg), les ordres de grandeurs des rappels et précisions filtrés par cibles sont similaires. Nous soulignons que les catégories RIEN et PRED sont les plus aisées à retrouver par nos modèles de CRF, alors que SUJ, et plus encore OBJ sont plus délicates à identifier. Cela s'explique probablement par le fait que les catégories grammaticales sont assez stables pour PRED et RIEN, et que c'est moins le cas pour SUJ et OBJ. Ceci se confirme par les résultats du même ordre calculés exclusivement sur les patrons SEQ, i.e. les modèles qui ne tiennent pas compte de la structure d'arbre. Les mots étiquetés comme prédicats sont en grande majorité des verbes, et beaucoup de verbes sont étiquetés comme prédicats. De même, certaines catégories sont pratiquement exclusivement étiquetées RIEN (clitiques, conjonctions... et ponctuations, mais elles ne sont pas considérées dans les calculs de scores). Et lorsque l'intention d'origine est d'identifier des groupes nominaux comme sujets et objets, il est plus difficile de retrouver précisément ces étiquettes seulement avec l'information de la catégorie grammaticale, car les éléments intéressants dans des groupes nominaux peuvent-être variés (déterminant, adjectif, nom, qui sont étiquetés SUJ, OBJ ou RIEN en fonction de la richesse de l'information qu'ils portent), et l'attachement des groupes nominaux ou non à des prédicats est très dépendant de la structure particulière de la phrase.

Rappelons, avant de conclure ce chapitre, que cet étiquetage cible en SUJ, PRED et OBJ ne permet pas directement l'extraction de triplets, comme nous l'avons signalé en présentant nos expérimentations. De plus amples travaux sur ce sujet, ainsi qu'un étiquetage en cibles en volume important et régi par des règles écrites seront envisagés pour une validation plus académique des premières expériences pratiquées ici.

Conclusion

Nous avons un objectif à long terme, l'extraction de relations dans des textes pour repérer de façon structurée les informations importantes. La structure en question, arbres de dépendances, a été notre objectif à moins long terme (thèse). Nous avons déterminé que les arbres de dépendances des phrases nous permettraient une bonne hiérarchisation des mots et de leurs importances relatives. Nous avons dans le même temps admis que la souplesse recherchée et l'absence d'exemples d'arbres de dépendances étaient des contraintes fortes auxquelles nous devons nécessairement nous soumettre. Nous nous sommes alors dirigé vers les recherches en apprentissage non supervisé de dépendances, puis éloigné du modèle DMV trop spécifique pour obtenir des résultats convaincants. Nous avons réussi à les appliquer de façon prometteuse à une première tentative d'extraction de relations.

6.1 Nos travaux

Cherchant à obtenir une nouvelle méthode efficace, adaptable et légère, nous avons choisi de nous placer dans le cadre des grammaires hors contexte. Ainsi, nous avons construit une famille de grammaires avec peu symboles non-terminaux (moins de 6), peu de règles de combinaisons de ces non-terminaux (une quinzaine), pour un total de quelques dizaines de règles par grammaire. Nous avons bâti ces grammaires de la famille DGdg en faisant l'hypothèse, sur chaque segment, d'un partage entre une partie gauche et une partie droite ; l'une est plus essentielle et l'autre moins significative. Ce partage se retrouvant jusqu'au niveau de chaque mot (catégories grammaticales), une hiérarchie totale existe alors sur tous les mots de la phrase, et on en déduit directement l'arbre de dépendances correspondant.

Une fois l'intuition formée de cette séparation gauche/droite entre une partie dominante et une partie dominée, il nous a fallu l'écrire précisément en termes de non-terminaux et de règles de grammaires hors contexte. Ceci est présenté en détail dans la section 3.3. Ensuite, il nous a fallu mettre en place deux protocoles. Le premier porte sur l'apprentissage et l'analyse de textes fournis étiquetés, le second mène à une évaluation d'une grande quantité d'arbres de dépendances issus de corpus en douze langues différentes. La collecte de données pour le premier protocole est simple,

il suffit de choisir des textes issus du web, et de les faire annoter par un étiqueteur syntaxique. En revanche, l'obtention d'arbres de dépendances corrects dans plusieurs langues, pour le protocole d'évaluation à grande échelle, est plus compliquée. Cela renforce notre conviction que l'apprentissage supervisé de dépendances demande un travail préliminaire conséquent, délicat et fastidieux de collecte, ou même de construction manuelle d'arbres de dépendances.

Une fois le premier protocole mis en place, l'apprentissage était effectué sur des volumes importants de données. Mais l'analyse n'étant que maladroitement estimée arbre par arbre par des non linguistes (ce que nous sommes), nous avons dû collecter les données et instaurer le deuxième protocole pour comparer nos différentes variantes et estimer rigoureusement la qualité des arbres que nous construisions. Les résultats, présentés dans le chapitre 4, sont encourageants quant à la qualité relative au domaine de l'ANSI. Sans viser l'état de l'art, nous nous comparons favorablement sur quelques langues à la référence DMV, pour des phases d'apprentissage et d'analyse bien plus rapides, une adaptation aux différentes langues vraiment plus simple, et une sensibilité à la paramétrisation assez faible.

6.2 L'application

Songeant toujours à des perspectives d'amélioration de notre méthode DGdg, nous avons commencé à nous intéresser à l'objectif à plus long terme de nos travaux : l'extraction de relations. Estimant que la structure des arbres construits par la méthode DGdg n'était pas complètement académique, mais qu'elle structurait les données de façon intéressante, nous avons conjointement mis à profit certains aspects des arbres dans une première application présentée dans le chapitre 5. Il apparaît, de façon encourageante, que nos arbres, construits facilement maintenant que le protocole est mis en place, améliorent l'extraction de relations qui synthétisent l'esprit d'un texte tout en gardant la sémantique.

6.3 Perspectives

6.3.1 Amélioration de la méthode DGdg

Avant de présenter nos idées pour améliorer la méthode DGdg, rappelons que lorsque l'on doit traiter une nouvelle langue, le choix d'une des 5 variantes de la grammaire DGdg n'est pas encore clairement guidé. Nous avons discuté dans la section 4.5 des pistes que nous envisageons pour automatiser ce choix de variante. La perspective la plus prometteuse prévoit une étude statistique de la log vraisemblance sur les corpus déjà évalués.

En persévérant dans notre logique de souplesse de la méthode, nous cherchons dans le même temps à améliorer les résultats pour obtenir des arbres de dépendances plus en adéquation avec les règles proposées par les linguistes. Une phase de segmentation en chunks peut être incluse dans l'étiquetage des textes. Il ne serait pas plus coûteux de considérer ces chunks comme terminaux au lieu des catégories grammaticales isolées. Les chunks formant naturellement des unités de sens cohérentes, ils seraient extrêmement intéressants du point de vue de l'extraction de relations. Il faudra adapter les données d'évaluation qui ne sont pas segmentées de cette façon et rectifier légèrement les arbres non compatibles avec certaines segmentations (mots de même chunk dispersés

dans des sous-arbres distincts) si l'on veut de nouveau mener une phase d'évaluation avant d'extraire véritablement les relations.

Il est certain que l'intégration de la donnée des mots lexicaux alourdirait les processus d'apprentissage et d'analyse, étant donnée la variété incomparable des mots par rapport au petit nombre de catégories morphosyntaxiques. Cependant, nous pensons que la sémantique, bien plus forte et précise des mots que des catégories grammaticales, pourrait entraîner un fort gain en qualité des arbres de dépendance construits. Les travaux menés dans [Kawahara et Uchimoto, 2007] montrent qu'une forte amélioration se produit lorsque l'on introduit les mots les plus fréquemment présents dans le texte, mais que l'introduction de beaucoup de mots pénalise fortement le processus d'apprentissage sans pourtant permettre d'amélioration notable de la qualité des arbres de dépendances. Nous envisageons pour le moment la lexicalisation dans la seule phase d'analyse. Nous pensons ajouter un biais dans l'algorithme CYK probabiliste, une pénalisation du terme calculé comme moins dominant que la tête du groupe de mots auquel une règle d'une grammaire DGdg est en train de le comparer (règle $\dots \rightarrow gD$ ou $\dots \rightarrow Gd$). Nous pensons à un premier calcul d'un facteur de « différence de domination » dP qui se calcule entre deux mots m_1 et m_2 de la façon suivante : $dP(m_1, m_2) = P(m_1|m_2) - P(m_2|m_1)$. Ce facteur de domination serait à intégrer à chaque étape dans le calcul des probabilités des arbres par CYK, pour les mots les plus fréquents dans le texte. Notre hypothèse, à vérifier, est que ce facteur est positif lorsque m_1 est globalement, sur tout le texte, plus dominant que m_2 , et négatif lorsque m_2 est plus dominant que m_1 . Il nous semble que ces modifications du processus d'analyse, basées sur de rapides calculs de fréquences, n'alourdiraient pas beaucoup le déroulement total, et permettraient sans doute, toujours de façon non supervisée, d'obtenir de meilleurs résultats.

Nous avons mené les évaluations sur des corpus parfaitement et manuellement étiquetés. Or dans des applications réelles, notre méthode sera appliquée à des textes annotés par des étiqueteurs automatiques, performants mais imparfaits. Il nous importe donc maintenant de tester la robustesse de notre méthode sur des catégories automatiquement attribuées, avec erreurs, sans utiliser les gold POS tags. Ceci s'envisage sans trop de difficultés, sur les mêmes corpus d'évaluation, par le même protocole, en observant les éventuelles dégradations de résultats lorsque l'étiquetage est remplacé par un étiquetage automatique.

Nous avons également remarqué, lors de nos travaux, la faible influence des différentes attributions de poids dans les grammaires initiales. Cela est tout à fait notable. Généralement l'algorithme Inside-Outside est très sensible à cette phase d'initialisation de la grammaire avant apprentissage. C'est aussi une des faiblesses de DMV. Nous préparons une comparaison des grammaires apprises à partir de grammaires initiales constituées des mêmes règles, mais pondérées différemment. Il s'agira d'estimer les différences directes entre les grammaires apprises, puis de les estimer indirectement en construisant les arbres de dépendances induits et d'en comparer la qualité par les UAS.

6.3.2 Utilisation des arbres de dépendances obtenus par la méthode DGdg

Nous avons présenté et évoqué dans le chapitre introductif les nombreuses applications de l'analyse de dépendances. Nous nous focalisons, quant à l'utilisation des arbres de dépendances par la méthode DGdg, sur l'extraction de relations, dans une optique à très long terme de construction ou consolidation d'ontologies et de réseaux sémantiques.

Comme nous l'avons précisé dans le chapitre 5, nous souhaiterions mener à plus grande échelle, avec plus de données annotées en éléments de triplets (SUJ,PRED,OBJ) une évaluation du type de celle que nous avons conduite et qui donne déjà des résultats encourageants. Nous devons aussi appréhender en amont la façon d'attribuer ces étiquettes cibles, établir une série de règles proprement formalisées, et en aval mettre en place une heuristique de reconstitution des triplets (SUJ,PRED,OBJ), à partir de l'étiquetage dispersé. Nous évoquions déjà ces perspectives dans la sous-section 5.3.2.

Nous nous sommes placé pour cette expérimentation dans le domaine de l'Open Information Extraction. Nous réfléchissons également à un changement de cadre pour chercher à extraire des relations typées, lorsqu'une demande précise sera formulée (identifier le dirigeant d'un parti politique, la capitale d'un pays...). Il faudrait alors disposer de données assez nombreuses dans le domaine précis en question. Les 3 étiquettes cibles SUJ, PRED et OBJ pourraient alors être déclinées en étiquettes typées (SUJ_dirigeant, SUJ_secrétaire, SUJ_opposant...).

7.1 Démonstration de l'équivalence de l'expressivité des formalismes "Arbre de dépendances" et "Arbre formel DGdg (4bin)"

Remarque 7.1 : notations

Nous introduisons lors de cette section des notations pour les arbres formels DGdg qui nous sont personnelles, qui sont sans ambiguïté dans le cadre des exemples abordés, mais qui ne se généralisent pas pour de grands arbres.

7.1.1 Rappels et notations

On se place dans le contexte de phrases considérées comme des séquences de N mots (identifiés par $m_i, i \in \llbracket 1; N \rrbracket$), ces mots étant en réalité les catégories grammaticales. On note alors la phrase $p = m_1 \dots m_n$.

- un arbre de dépendances projectif noté t est constitué d'une unique racine m à laquelle sont attachés k sous-arbres projectifs t_j , $0 \leq k \leq N - 1$. On note un arbre de dépendances par l'ensemble des liens entre ses mots et/ou par des arbres dont les liens entre les mots sont communs avec l'arbre principal. Exemple : $t = \{m \rightarrow m_{i_1}, m \rightarrow m_{i_2}, m_{i_1} \rightarrow m_{i_3}, \dots\}$ ou $t = \{m \rightarrow t_j, j \in \llbracket 1; k \rrbracket\}$;
- un sous-arbre projectif est un arbre projectif contenant une partie convexe de la phrase, i.e. tous les mots entre les deux mots aux extrémités ;
- on notera sp_j la partie convexe de la phrase (ou sous-phrase) comprenant les mots de t_j . D'autre part, on peut écrire $sp_j = m_{j_1} \dots m_{j_{n_j}}$. On a $\sum_j |sp_j| = \sum_j n_j = N - 1$;
- un arbre formel DGdg noté T est constitué d'une unique racine S dont dérivent deux sous-arbres T_{j_1} et T_{j_2} OU un unique terminal m_i . On note un arbre formel DGdg par l'ensemble de ses liens/dérivations ou par des arbres contenant l'ensemble de ses dérivations. Exemple : $T = \{S(m_1) \rightarrow G(m_1), S(m_1) \rightarrow d(m_2), G(m_1) \rightarrow m_1, d(m_2) \rightarrow m_2\}$ ou $T = \{S(m) \rightarrow T_a, S(m) \rightarrow T_b\}$;

- le mot noté entre parenthèses à droite du non-terminal indique son mot Tête associé (voir Algorithme 2) ;
- un sous-arbre formel DGdg T_j est constitué d'une unique racine D, G, d ou g dont dérivent deux sous-arbres T_{j_1} et T_{j_2} OU un unique terminal m_i ;
- on peut considérer des phrases d'un seul mot si l'on ajoute aux règles de départ les règles $S \rightarrow m$. en théorie, une phrase est toujours constituée d'au moins deux mots, un "vrai" mot et une ponctuation, mais en pratique, dans les corpus, il arrive que des phrases d'un seul mot apparaissent. On ajoute alors ces règles de dérivation de S en terminal pour pouvoir les traiter ;
- un sous-arbre DGdg couvre de la même façon qu'un arbre de dépendances une partie convexe de la phrase.

7.1.2 Théorème et démonstration

Théorème Pour tout arbre de dépendances t d'une phrase p , il existe au moins un arbre formel DGdg se traduisant par l'algorithme 2 en t .

Démonstration. Nous démontrons ce théorème par récurrence sur la taille (i.e. le nombre de mots, de nœuds) des arbres de dépendances t .

- $N=1$. Lorsque t est constitué d'un seul mot (cas non pertinent), t est un arbre constitué d'une unique racine m_1 . L'arbre formel $T = \{S(m_1) \rightarrow m\}$ se traduit immédiatement en t .
- $N=2$. Lorsque t est constitué de deux mots, sa structure est $\{m_1 \rightarrow m_2\}$ ou $\{m_2 \rightarrow m_1\}$. Dans le premier cas, l'arbre formel $T = \{S(m_1) \rightarrow G(m_1), S(m_1) \rightarrow d(m_2), G(m_1) \rightarrow m_1, d(m_2) \rightarrow m_2\}$ se traduit en t , dans le second cas, $T = \{S(m_2) \rightarrow g(m_1), S(m_2) \rightarrow D(m_2), g(m_1) \rightarrow m_1, D(m_2) \rightarrow m_2\}$ se traduit en t .
- On suppose que pour tout arbre de dépendances t_j de taille strictement inférieure à N , il existe au moins un arbre formel DGdg se traduisant en t_j par l'algorithme 2.
- On démontre que pour un arbre de dépendances t de taille N d'une phrase p , il existe au moins un arbre formel DGdg se traduisant en t par l'algorithme 2.

Par définition, t est constitué d'une racine m et de k sous-arbres t_j .

$$t = \{m \rightarrow t_j, j \in [1; k]\}, k < N.$$

On considère alors séparément les arbres de dépendances t_1 de racine m_{i_1} et $t' = \{m \rightarrow t_j, j \in [2; k]\}$. (Si $[2; k] = \emptyset$, t' ne contient que le mot m , c'est un arbre à un mot).

L'arbre t' ne contient pas le mot m_{i_1} , donc sa taille est strictement inférieure à N , de même t_1 ne contient pas le mot m , donc sa taille est aussi strictement inférieure à N . Par hypothèse de récurrence, il existe alors au moins un arbre formel DGdg T' se traduisant en t' et un arbre formel DGdg T_1 se traduisant en t_1 par l'algorithme 2.

On peut réécrire $t = \{m \rightarrow_1, t'\}$.

On rappelle que t' et t_1 sont des arbres de dépendances projectifs, donc ils couvrent des parties convexes qui ne s'intersectent pas (respectivement sp' et sp_1) de la phrase p .

La phrase s'écrit $p = sp_1 sp'$.

On utilise la notation $T = \{S(m) \rightarrow T_a, S(m) \rightarrow T_b\}$ où T_a et T_b sont des sous-arbres formels de T . Le lien/dérivation $S(m) \rightarrow T_x$ se réécrit en toutes les dérivations de T_x auxquelles s'ajoutent $S(m) \rightarrow \text{Tête Formelle}(T_x)$.

On pose $T_a = \{g(m_{i_1}) \rightarrow T_{1,a_1}, g(m_{i_1}) \rightarrow T_{1,a_2}\}$ où T_{1,a_1} et T_{1,a_2} sont les sous-arbres dérivant de la racine S de T_1 , $T_b = \{D(m) \rightarrow T'_{b_1}, D(m) \rightarrow T'_{b_2}\}$ où T'_{b_1} et T'_{b_2} sont les sous-arbres dérivant de la racine S de T' .

Le cas où T' ou T_1 ne contiennent qu'un seul mot se traite exactement de la même manière, au détail près que $T_b = \{D(m) \rightarrow m\}$ ou $T_a = \{g(m_{i_1}) \rightarrow m_{i_1}\}$.

□

7.2 Détail de la procédure de traduction des arbres DGdg en arbres de dépendances

Dans la première étape, chaque mot de la phrase (i.e. sa catégorie grammaticale), est indicé par sa position dans la phrase. Pour chaque feuille de l'arbre DGdg, on annote les non-terminaux des nœuds ancêtres par cet indice de position jusqu'à rencontrer un nœud dont le non-terminal est un dominé, en pratique une minuscule, ou le symbole S du sommet de l'arbre. À la fin de cette étape, on obtient un arbre DGdg annoté d'indices entre parenthèses que nous avons appelé les motTête des non-terminaux. La Figure 3.26 montre l'arbre de l'exemple annoté de ces motTête.

Dans la deuxième étape, on décode cette annotation pour déterminer le parent en dépendances de chaque mot (catégorie) de la phrase. Pour chaque feuille indicée k de l'arbre DGdg, on remonte le long des nœuds parents jusqu'à rencontrer une annotation de motTête $j \neq k$ ou le non-terminal S . Cet indice $j \neq k$ donne la position du parent en dépendances du mot de position k dans la phrase considérée. Si l'ascension ne s'interrompt pas avant le symbole S au sommet de l'arbre, cela signifie que la catégorie considérée est la racine de l'arbre de dépendances. On note artificiellement 0 son parent. Les motTête, ou indices des parents en dépendances de chaque mot, sont indiqués dans l'exemple de la Figure 3.26 sous chaque catégorie. L'arbre de dépendances est donc construit, il suffit de la mettre en forme comme dans la Figure 3.27.

Ces deux étapes de la procédure de traduction sont présentées dans le pseudo-code de l'Algorithme 2.

Algorithme 2 : Traduction d'un arbre DGdg en arbre de dépendances bare-bones

Étape 1 : Recherche des motsTête des nœuds de l'arbre DGdg. On remonte les nœuds de l'arbre DGdg jusqu'à rencontrer un symbole non-terminal minuscule. Chaque nœud rencontré au cours de la montée est annoté de ce motTête.

pour chaque feuille de l'arbre DGdg **faire**

 DGdgNœud = DGdgparent(feuille)

tant que DGdgNœud=D **ou** DGdgNœud=G **ou** DGdgNœud=S **faire**

 motTêteAssocié(parent(DGdgNœud))=feuille

si DGdgparent(DGdgNœud) ≠ S **alors**

 nœud=DGdgparent(DGdgNœud)

Étape 2 : Construction de l'arbre de dépendances. Ses nœuds sont les feuilles de l'arbre DGdg. On détermine la feuille tête (parent) de chaque nœud de l'arbre de dépendances. On parcourt récursivement l'arbre DGdg. Pour trouver le parent en dépendance de chaque feuille de l'arbre DGdg, on remonte à nouveau l'arbre DGdg. Le premier motTête différent rencontré est la position de la feuille du parent en dépendance.

pour chaque feuille de l'arbre DGdg **faire**

 DGdgNœud = DGdgparent(feuille)

tant que motTêteAssocié(DGdgNœud)=feuille **et** DGdgNœud ≠ S **faire**

 DGdgNœud=DGdgparent(DGdgNœud)

si DGdgNœud=S **et** motTêteAssocié(DGdgNœud)=leaf **alors**

 parentDependance(feuille)=0

sinon

 parentDependance(feuille)=motTêteAssocié(DGdgNœud)

7.3 Analyse par les variantes de la méthode DGdg sur des phrases simples

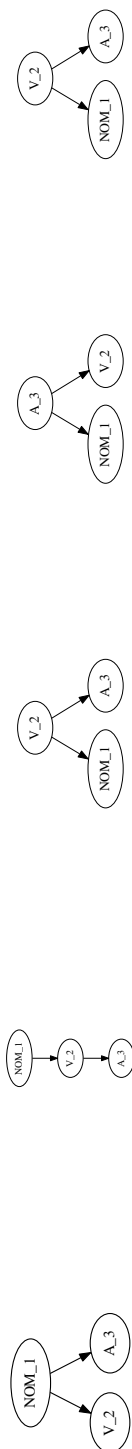


FIGURE 7.1 : 3bin, UAS = 0/3. FIGURE 7.2 : 4bin, UAS = 1/3. FIGURE 7.3 : 4ter, UAS = 3/3. FIGURE 7.4 : 5ter, UAS = 0/3. FIGURE 7.5 : 5ter+, UAS = 3/3.

Arbres de dépendances de la phrase type NOM V A, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

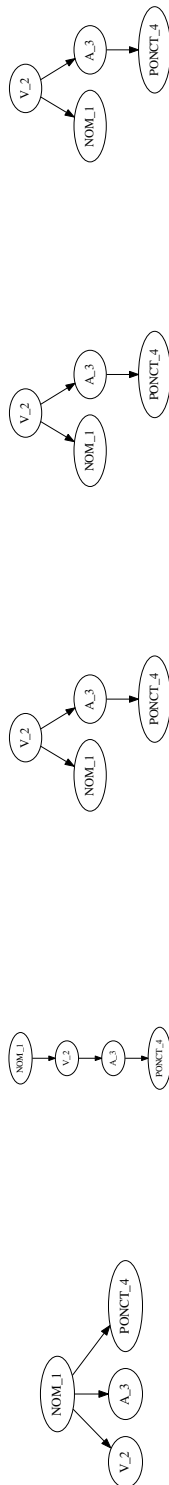


FIGURE 7.6 : 3bin, UAS = 0/3. FIGURE 7.7 : 4bin, UAS = 1/3. FIGURE 7.8 : 4ter, UAS = 3/3. FIGURE 7.9 : 5ter, UAS = 3/3. FIGURE 7.10 : 5ter+, UAS = 3/3.

Arbres de dépendances de la phrase type NOM V A PUNCT, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

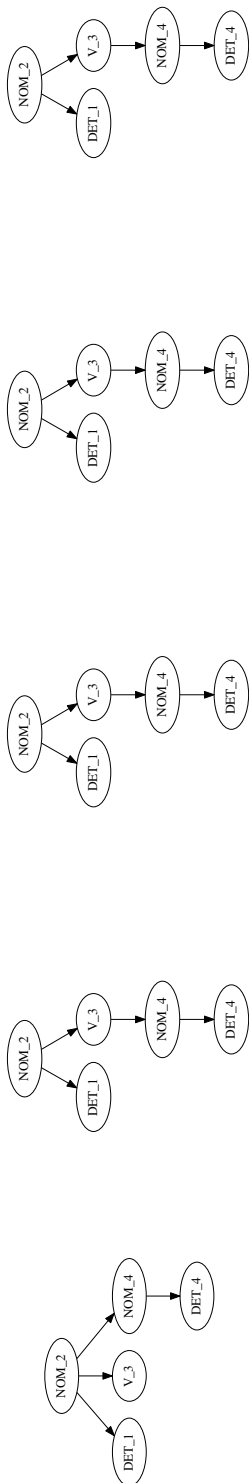


FIGURE 7.11 : 3bin,
UAS = 2/5.

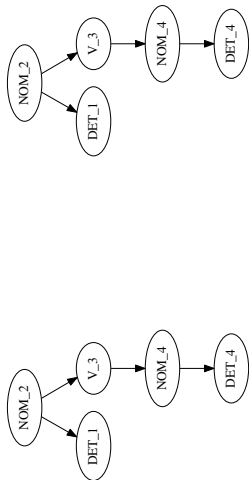


FIGURE 7.12 : 4bin,
UAS = 3/5.

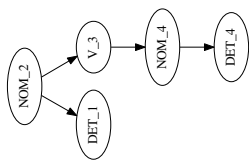


FIGURE 7.13 : 4ter,
UAS = 3/5.

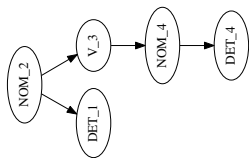
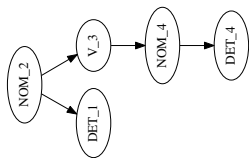


FIGURE 7.14 : 5ter,
UAS = 3/5.



Arbres de dépendances de la phrase type DET NOM V DET NOM, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

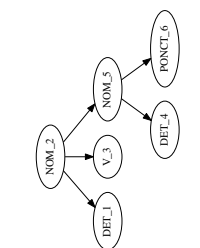


FIGURE 7.16 : 3bin,
UAS = 2/5.

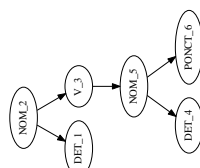


FIGURE 7.17 : 4bin,
UAS = 3/5.

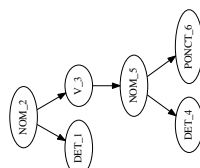


FIGURE 7.18 : 4ter,
UAS = 3/5.

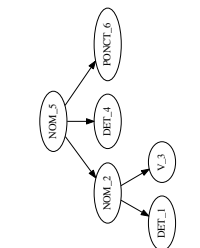


FIGURE 7.19 : 5ter,
UAS = 2/5.

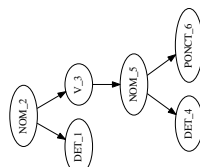


FIGURE 7.20 : 5ter+,
UAS = 3/5.

Arbres de dépendances de la phrase type DET NOM V DET NOM PUNCT, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

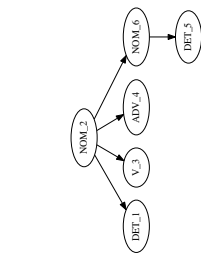


FIGURE 7.21 : 3bin,
UAS = 2/6.

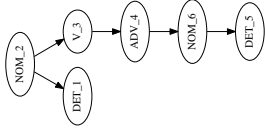


FIGURE 7.22 : 4bin,
UAS = 3/6.

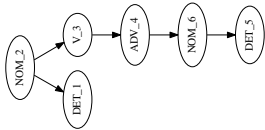


FIGURE 7.23 : 4ter,
UAS = 3/6.

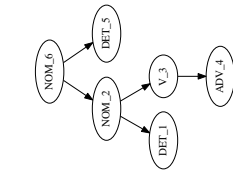
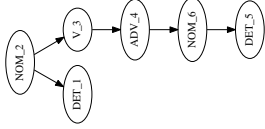


FIGURE 7.24 : 5ter,
UAS = 3/6.



Arbres de dépendances de la phrase type DET NOM V ADV DET NOM, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.

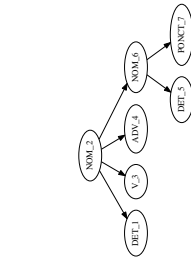


FIGURE 7.26 : 3bin,
UAS = 2/6.

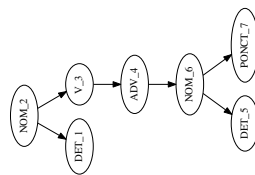


FIGURE 7.27 : 4bin,
UAS = 3/6.

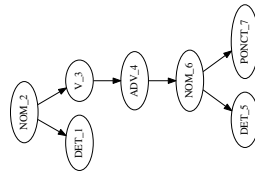


FIGURE 7.28 : 4ter,
UAS = 3/6.

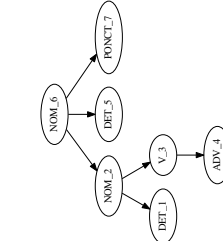
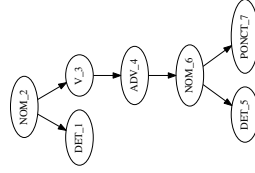


FIGURE 7.29 : 5ter,
UAS = 3/6.



Arbres de dépendances de la phrase type DET NOM V ADV DET NOM PONCT, construits après analyse à partir des grammaires DGdg. Les grammaires ont été apprises sur le French Treebank de dépendances.



Bibliographie

- Abeillé, A. et Barrier, N. (2004). Enriching a french treebank. *In : Proceedings of LREC 2004, Lisbon*. Cité pages [9](#), [13](#), [45](#), [48](#), [50](#), [51](#), [55](#), [59](#), [76](#), [84](#) et [85](#).
- Afonso, S., Bick, E., Haber, R. et Santos, D. (2002). Floresta sintà (c) tica : A treebank for portuguese. *In : LREC*. Cité page [59](#).
- Arun, A. et Keller, F. (2005). Lexicalization in crosslinguistic probabilistic parsing : the case of French. *In : Proceedings of ACL*, pages 306–313. Cité page [13](#).
- Baker, J. K. (1979). Trainable grammars for speech recognition. *In : Klatt, D. H. et Wolf, J. J., éditeurs : Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550. Cité page [33](#).
- Brants, S., Dipper, S., Hansen, S., Lezius, W. et Smith, G. (2002). The TIGER treebank. *In : Proceedings of the workshop on treebanks and linguistic theories*, page 24 à 41. Cité page [58](#).
- Brody, S. (2010). It depends on the translation : unsupervised dependency parsing via word alignment. *In : Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1214–1222. Cité page [22](#).
- Brown, P. F., Pietra, V. J. D., Pietra, S. A. D. et Mercer, R. L. (1993). The mathematics of statistical machine translation : Parameter estimation. *Comput. Linguist.*, 19(2):263–311. Cité page [22](#).
- Buchholz, S. et Marsi, E. (2006). Conll-x shared task on multilingual dependency parsing. *In : Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X '06*, pages 149–164, Stroudsburg, PA, USA. Cité page [57](#).
- Candito, M., Crabbé, B. et Denis, P. (2010). Statistical french dependency parsing : treebank conversion and first results. *In : LREC 2010*, page 1840 à 1847. Cité pages [9](#), [13](#), [45](#), [48](#), [50](#), [51](#), [59](#), [76](#), [84](#) et [85](#).

- Carroll, G., Carroll, G., Charniak, E. et Charniak, E. (1992). Two experiments on learning probabilistic dependency grammars from corpora. *In : Working Notes of the Workshop Statistically-Based NLP Techniques*, pages 1–13. AAAI. Cité pages 15 et 21.
- Chelba, C., Engle, D., Jelinek, F., Jimenez, V., Khudanpur, S., Mangu, L., Printz, H., Ristad, E., Rosenfeld, R., Stolcke, A. et others (1997). Structure and performance of a dependency language model. *In : EUROSPEECH*. Cité page 12.
- Civit, M. et Martì, M. (2004). Building Cast3LB : a spanish treebank. *Research on Language and Computation*, 2(4):549–574. Cité page 59.
- Cohen, S. B., Das, D. et Smith, N. A. (2011). Unsupervised structure prediction with non-parallel multilingual guidance. *In : Proceedings of EMNLP*. Cité pages 21, 22, 66 et 69.
- Cohen, S. B. et Smith, N. A. (2010). Covariance in unsupervised learning of probabilistic grammars. *Journal of Machine Learning Research*, 11:3017–3051. Cité pages 17, 21, 22, 61, 62, 63, 64, 66 et 69.
- Cohn, T. et Blunsom, P. (2005). Semantic role labelling with tree conditional random fields. *In : Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 169–172. Cité pages 75 et 78.
- Constant, M., Tellier, I., Duchier, D., Dupont, Y., Sigogne, A. et Billot, S. (2011). Intégrer des connaissances linguistiques dans un crf : application à l'apprentissage d'un segmenteur-étiqueteur du français. *TALN 2011 (Traitement automatique des langues naturelles)*. Cité page 74.
- Cortes, C. et Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273–297. Cité page 11.
- Culotta, A. et Sorensen, J. (2004). Dependency tree kernels for relation extraction. *In : Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Cité page 11.
- Dempster, A. P., Laird, N. M. et Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society : Series B*, 39:1–38. Cité page 16.
- Ding, Y. et Palmer, M. (2004). Synchronous dependency insertion grammars : A grammar formalism for syntax based statistical MT. *In : Workshop on Recent Advances in Dependency Grammars (COLING)*, pages 90–97. Cité page 12.
- Dzeroski, S., Erjavec, T., Ledinek, N., Pajas, P., Zabokrtsky, Z. et Zele, A. (2006). Towards a slovene dependency treebank. *In : Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*. Cité page 59.
- Eisner, J. M. (1996). Three new probabilistic models for dependency parsing : An exploration. *In : Proceedings of the 16th Conference on Computational Linguistics - Volume 1, COLING '96*, pages 340–345, Stroudsburg, PA, USA. Cité pages 10 et 12.
- Etzioni, O., Banko, M., Soderland, S. et Weld, D. S. (2008). Open information extraction from the web. *Commun. ACM*, 51(12):68–74. Cité page 72.

- Etzioni, O., Fader, A., Christensen, J., Soderland, S. et Mausam, M. (2011). Open Information Extraction : The Second Generation. *In : IJCAI*, volume 11, pages 3–10. Cité page 75.
- Fayruzov, T., Cock, M. D., Cornelis, C. et Hoste, V. (2008). Deeper : A full parsing based approach to protein relation extraction. *In : Marchiori, E. et Moore, J. H., éditeurs : EvoBIO*, volume 4973 de *Lecture Notes in Computer Science*, pages 36–47. Springer. Cité page 11.
- Fundel, K., Küffner, R. et Zimmer, R. (2007). Relex—relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371. Cité pages 11 et 72.
- Gaifman, H. (1965). Dependency systems and phrase-structure systems. *Information and Control*, 8(3):304 – 337. Cité page 12.
- Gazdar, G., Klein, E., Pullum, G. K. et Sag, I. A. (1985). *Generalized Phrase Structure Grammar*. Harvard University Press. Cité pages 38 et 39.
- Gillenwater, J., Ganchev, K., Graça, J., Pereira, F. et Taskar, B. (2010). Sparsity in Dependency Grammar Induction. *In : Proceedings of Association for Computational Linguistics (ACL)*. Cité page 21.
- Gillenwater, J., Ganchev, K., Graça, J., Pereira, F. et Taskar, B. (2011). Posterior Sparsity in Unsupervised Dependency Parsing. *Journal of Machine Learning Research (JMLR)*. Cité pages 21, 66 et 69.
- Gilleron, R., Jousse, F., Tommasi, M. et Tellier, I. (2008). Conditional Random Fields for XML Applications. Research Report RR-6738. Cité pages 75 et 78.
- Günes Erkan, Arzucan Özgür, D. R. R. (2007). Semi-supervised classification for extracting protein interaction sentences using dependency parsing. *In : Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 228–237. Cité page 11.
- Gonzalez, R. C. et Thomason, M. G. (1978). *Syntactic pattern recognition : an introduction*. Addison-Wesley Pub. Co. Cité pages 25 et 27.
- Graça, J. V., Ganchev, K. et Taskar, B. (2007). Expectation maximization and posterior constraints. *In : Advances in NIPS*, pages 569–576. Cité page 21.
- Haghighi, A. D., Ng, A. Y. et Manning, C. D. (2005). Robust textual inference via graph matching. *In : Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 387–394. Cité page 12.
- Hays, D. G. (1964). Dependency Theory : A Formalism and Some Observations. *Language*, 40(4): 511–525. Cité page 12.
- Headden III, W. P., Johnson, M. et McClosky, D. (2009). Improving unsupervised dependency parsing with richer contexts and smoothing. *In : Proceedings of Human Language Technologies : The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 101–109. Cité page 23.

- Hinrichs, E. W., Bartels, J., Kawata, Y., Kordoni, V. et Telljohann, H. (2000). The VERBMOBIL treebanks. *In : KONVENS*, page 107 à 112. Cité page 59.
- Jurafsky, D. et Martin, J. H. (2009). *Speech and Language Processing*. Prentice Hall. Cité pages 28 et 31.
- Kahane, S. (2001). Grammaires de dépendance formelles et théorie Sens-Texte. *In : Traitement Automatique des Langues Naturelles*. Cité page 11.
- Kasami, T. (1965). An efficient recognition and syntax analysis algorithm for context-free languages. Rapport technique AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA†. Cité pages 26 et 28.
- Kawahara, D. et Uchimoto, K. (2007). Minimally lexicalized dependency parsing. *In : Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07*, pages 205–208, Stroudsburg, PA, USA. Cité page 91.
- Klein, D. (2005). The unsupervised learning of natural language structure. *In : Ph.D. Thesis*. Stanford University. Cité pages 17 et 60.
- Klein, D. et Manning, C. D. (2003a). Accurate unlexicalized parsing. *In : Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03*, pages 423–430, Stroudsburg, PA, USA. Cité page 72.
- Klein, D. et Manning, C. D. (2003b). Fast exact inference with a factored model for natural language parsing. *In : Advances in Neural Information Processing Systems 15 (NIPS)*, pages 3–10. MIT Press. Cité page 72.
- Klein, D. et Manning, C. D. (2004). Corpus-based induction of syntactic structure : Models of dependency and constituency. *In : Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 478. Cité pages 14, 16 et 17.
- Koo, T., Carreras, X. et Collins, M. (2008). Simple semi-supervised dependency parsing. *In : Proceedings of ACL/HLT*. Cité page 13.
- Kromann, M. T. (2003). The danish dependency treebank and the DTAG treebank tool. *In : Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT)*, page 217. Cité page 58.
- Kübler, S., McDonald, R. et Nivre, J. (2009). *Dependency Parsing*. Synthesis lectures on human language technologies. Morgan & Claypool. Cité pages 9 et 12.
- Kudo, T. et Matsumoto, Y. (2000). Japanese dependency structure analysis based on support vector machines. *In : Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora : Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13, EMNLP '00*, pages 18–25, Stroudsburg, PA, USA. Cité page 13.

- Lafferty, J. D., McCallum, A. et Pereira, F. C. N. (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data. *In : Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Cité pages 74 et 75.
- Lari, K. et Young, S. J. (1990). The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56. Cité pages 17, 26, 34 et 38.
- Li, Z., Zhang, M. et Chen, W. (2014). Ambiguity-aware ensemble training for semi-supervised dependency parsing. *In : Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 457–467, Baltimore, Maryland. Cité page 13.
- Ma, X. et Xia, F. (2014). Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization. *In : Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1 : Long Papers*, pages 1337–1348. Cité page 13.
- Marcus, M. P., Marcinkiewicz, M. A. et Santorini, B. (1993). Building a large annotated corpus of english : The penn treebank. *Computational linguistics*, 19(2):313 à 330. Cité page 58.
- Mareček, D. et Straka, M. (2013). Stop-probability estimates computed on a large corpus improve unsupervised dependency parsing. *In : ACL 13*, pages 281–290. Cité page 21.
- Marneffe, M.-C. d. et Manning, C. D. (2008). The stanford typed dependencies representation. *In : Coling 2008 : Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation, CrossParser '08*, pages 1–8, Stroudsburg, PA, USA. Cité page 73.
- Maruyama, H. (1990). Structural disambiguation with constraint propagation. *In : Berwick, R. C., éditeur : ACL*, pages 31–38. ACL. Cité page 12.
- McCallum, A. et Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. *In : Daelemans, W. et Osborne, M., éditeurs : Proceedings of CoNLL-2003*, pages 188–191. Edmonton, Canada. Cité page 75.
- McClosky, D., Charniak, E. et Johnson, M. (2006). Reranking and self-training for parser adaptation. *In : Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 337–344. Cité page 13.
- McCord, M. C., Murdock, J. W. et Boguraev, B. (2012). Deep parsing in watson. *IBM Journal of Research and Development*, 56(3):3. Cité page 11.
- McDonald, R., Nivre, J., Quirmbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K., Petrov, S., Zhang, H. et Täckström, O. (2013). Universal dependency annotation for multilingual parsing. *In : Proceedings of ACL*. Cité page 60.
- Ngai, G. et Florian, R. (2001). Transformation-based learning in the fast lane. *In : Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies, NAACL '01*, pages 1–8, Stroudsburg, PA, USA. Cité page 72.

- Nilsson, J., Hall, J. et Nivre, J. (2005). MAMBA meets TIGER : reconstructing a treebank from antiquity. Cité page [59](#).
- Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J. et Yuret, D. (2007). The conll 2007 shared task on dependency parsing. In : *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, Prague, Czech Republic. Cité page [57](#).
- Paskin, M. A. (2002). Grammatical digrams. *Advances in Neural Information Processing Systems*, 14(1):91–97. Cité page [16](#).
- Pereira, F. et Schabes, Y. (1992). Inside-outside reestimation from partially bracketed corpora. In : *Proceedings of the 30th Annual Meeting of the ACL*, pages 128–135. Cité page [16](#).
- Piskorski, J. et Yangarber, R. (2013). Information extraction : Past, present and future. In : Poibeau, T., Saggion, H., Piskorski, J. et Yangarber, R., éditeurs : *Multi-source, Multilingual Information Extraction and Summarization*, Theory and Applications of Natural Language Processing, pages 23–49. Springer Berlin Heidelberg. Cité page [71](#).
- Prescher, D. (2004). Inside-outside estimation meets dynamic EM. *CoRR*, abs/cs/0412016. Cité page [38](#).
- Quirk, C., Menezes, A. et Cherry, C. (2005). Dependency treelet translation : Syntactically informed phrasal SMT. In : *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 271–279. Cité page [11](#).
- Søgaard, A. (2011). From ranked words to dependency trees : two-stage unsupervised non-projective dependency parsing. In : *Proceedings of TextGraphs-6 : Graph-based Methods for Natural Language Processing*, pages 60–68. Cité page [22](#).
- Sha, F. et Pereira, F. (2003). Shallow parsing with conditional random fields. In : *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 134–141, Stroudsburg, PA, USA. Cité page [75](#).
- Simov, K., Osenova, P., Slavcheva, M., Kolkovska, S., Balabanova, E., Doikoff, D., Ivanova, K., Simov, A. et Kouylekov, M. (2002). Building a linguistically interpreted corpus of bulgarian : the BulTree-Bank. In : *LREC*. Cité page [58](#).
- Smith, L., Rindfleisch, T. et Wilbur, W. J. (2004). MedPost : a part-of-speech tagger for bioMedical text. *Bioinformatics (Oxford, England)*, 20(14):2320–2321. Cité page [72](#).
- Smith, N. A. et Eisner, J. (2005). Contrastive estimation : Training log-linear models on unlabeled data. In : *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 354–362, Stroudsburg, PA, USA. Cité page [22](#).
- Smith, N. A. et Eisner, J. (2006). Annealing structural bias in multilingual weighted grammar induction. In : *Proceedings of ACL 06*, pages 569–576. Cité page [22](#).

- Snow, R., Jurafsky, D. et Ng, A. Y. (2004). Learning syntactic patterns for automatic hypernym discovery. In : *Advances in Neural Information Processing Systems (NIPS 2004)*. This is a draft version from the NIPS preproceedings ; the final version will be published by April 2005. Cité page [12](#).
- Spitkovsky, V. I. (2013). *Grammar Induction and Parsing with Dependency-and-Boundary Models*. Thèse de doctorat, STANFORD UNIVERSITY. Cité page [21](#).
- Spitkovsky, V. I., Alshawi, H. et Jurafsky, D. (2010). From baby steps to leapfrog : How "less is more" in unsupervised dependency parsing. In : *Human Language Technologies : The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 751–759, Stroudsburg, PA, USA. Cité page [66](#).
- Spitkovsky, V. I., Alshawi, H. et Jurafsky, D. (2011). Lateen EM : Unsupervised training with multiple objectives, applied to dependency grammar induction. In : *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*. Cité page [21](#).
- Spitkovsky, V. I., Alshawi, H. et Jurafsky, D. (2012). Three dependency-and-boundary models for grammar induction. In : *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*. Cité pages [66](#) et [69](#).
- Tesnière, L. (1959). *Éléments de syntaxe structurale*. Éditions Klincksieck. Cité page [9](#).
- Tjong Kim Sang, E. F. et De Meulder, F. (2003). Introduction to the conll-2003 shared task : Language-independent named entity recognition. In : Daelemans, W. et Osborne, M., éditeurs : *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada. Cité page [75](#).
- Van der Beek, L., Bouma, G., Malouf, R. et Van Noord, G. (2002). The alpine dependency treebank. *Language and Computers*, 45(1):8 à 22. Cité page [59](#).
- Wang, M., Smith, N. A. et Mitamura, T. (2007). What is the Jeopardy model ? a quasi-synchronous grammar for QA. In : *Proceedings of the EMNLP-CoNLL*, pages 22–32, Prague, Czech Republic. Cité page [11](#).
- Wang, M., Xia, H., Sun, D., Chen, Z., Wang, M. et Li, A. (2014). Literature mining of protein phosphorylation using dependency parse trees. *Methods*, 67(3):386 – 393. Systems Biology with Omics Data. Cité page [11](#).
- Wu, F. et Weld, D. S. (2010). Open information extraction using wikipedia. In : *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 118–127, Stroudsburg, PA, USA. Cité pages [71](#), [72](#), [75](#), [87](#) et [88](#).
- Yamada, H. et Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In : *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*. Cité page [13](#).
- Yuret, D. (1998). *Discovery of linguistic relations using lexical attraction*. Thèse de doctorat. Cité page [16](#).

Marie ARCADIAS

Apprentissage non supervisé de dépendances à partir de textes

Résumé :

Les grammaires de dépendance permettent de construire une organisation hiérarchique syntaxique des mots d'une phrase. La construction manuelle des arbres de dépendances étant une tâche exigeant temps et expertise, de nombreux travaux cherchent à l'automatiser. Visant à établir un processus léger et facilement adaptable nous nous sommes intéressés à l'apprentissage non supervisé de dépendances, évitant ainsi d'avoir recours à une expertise coûteuse. L'état de l'art en apprentissage non supervisé de dépendances (DMV) se compose de méthodes très complexes et extrêmement sensibles au paramétrage initial. Nous présentons dans cette thèse un nouveau modèle pour résoudre ce problème d'analyse de dépendances, mais de façon plus simple, plus rapide et plus adaptable. Nous apprenons une famille de grammaires (PCFG) réduites à moins de 6 non terminaux et de 15 règles de combinaisons des non terminaux à partir des étiquettes grammaticales. Les PCFG de cette famille que nous nommons DGdg (pour DROITE GAUCHE droite gauche) se paramètrent très légèrement, ainsi elles s'adaptent sans effort aux 12 langues testées. L'apprentissage et l'analyse sont effectués au moins deux fois plus rapidement que DMV sur les mêmes données. Et la qualité des analyses DGdg est pour certaines langues proche des analyses par DMV. Nous proposons une première application de notre méthode d'analyse de dépendances à l'extraction d'informations. Nous apprenons par des CRF un étiquetage en fonctions « sujet », « objet » et « prédicat », en nous fondant sur des caractéristiques extraites des arbres construits.

Mots clés : apprentissage non supervisé, grammaire de dépendances, grammaire hors contexte, CYK, Inside-Outside, CRF, extraction de relations.

Unsupervised dependency parsing from texts

Abstract :

Dependency grammars allow the construction of a hierarchical organization of the words of sentences. The one-by-one building of dependency trees can be very long and it requires expert knowledge. In this regard, we are interested in unsupervised dependency learning. Currently, DMV give the state-of-art results in unsupervised dependency parsing. However, DMV has been known to be highly sensitive to initial parameters. The training of DMV model is also heavy and long. We present in this thesis a new model to solve this problem in a simpler, faster and more adaptable way. We learn a family of PCFG using less than 6 nonterminal symbols and less than 15 combination rules from the part-of-speech tags. The tuning of these PCFG is light, and so easily adaptable to the 12 languages we tested. Our proposed method for unsupervised dependency parsing can show the near state-of-the-art results, being twice faster. Moreover, we describe our interests in dependency trees to other applications such as relation extraction. Therefore, we show how such information from dependency structures can be integrated into condition random fields and how to improve a relation extraction task.

Keywords : unsupervised machine learning, dependency grammar, context-free grammar, CKY, Inside-Outside, CRF, relation extraction.



**Laboratoire d'Informatique
Fondamentale d'Orléans**
Bâtiment 3IA, rue Léonard de Vinci, B.P. 6759
45067 ORLEANS cedex 2, FRANCE