Expériences d'analyse syntaxique statistique du français

Benoît Crabbé, Marie Candito

Expériences d'analyse syntaxique statistique du français

Benoît Crabbé et Marie Candito
Université Paris 7 (UFRL) et INRIA (Alpage)
30 rue du Château des Rentiers 75013 Paris
{bcrabbe,candito}@linguist.jussieu.fr

Résumé. Nous montrons qu'il est possible d'obtenir une analyse syntaxique statistique satisfaisante pour le français sur du corpus journalistique, à partir des données issues du French Treebank du laboratoire LLF, à l'aide d'un algorithme d'analyse non lexicalisé.

Abstract. We show that we can acquire satisfactory parsing results for French from data induced from the French Treebank using an unlexicalised parsing algorithm.

Mots-clés : Analyseur syntaxique statistique, Analyse syntaxique non lexicalisée, Analyse du français.

Keywords: Statistical parser, Unlexicalised parsing, French parsing.

1 Introduction

À l'heure actuelle, la communauté francophone dispose de plusieurs environnements de développement de grammaires et d'analyse syntaxique symbolique automatique dite profonde. Cependant, dans la perspective de réaliser une analyse syntaxique profonde à grande échelle sur corpus, l'analyse syntaxique symbolique pose deux problèmes : (1) l'ambiguité inhérente des grammaires[1] et (2) leur faible capacité à rendre compte d'un très grand nombre de phénomènes épars[2].

En vue de traiter ces deux problèmes, nous investiguons ici l'usage de grammaires de treebank, afin de constituer une chaîne de traitement d'analyse syntaxique profonde intégrant une composante statistique. Nous montrons plus particulièrement comment réutiliser un algorithme d'analyse « état de l'art » pour le français de manière à obtenir des résultats comparables à ceux obtenus pour la plupart des langues européennes excepté l'anglais.

La plupart des algorithmes d'analyse syntaxique statistique reposant sur une grammaire de treebank sont mis au point à partir du Penn Treebank (Marcus *et al.*, 1994), corpus arboré de réfé-

[1] Les analyseurs symboliques basés sur des grammaires formelles ne fournissent pas tels quels de solution à la désambiguisation syntaxique. D'autres approches symboliques intègrent directement la tâche de désambiguisation, comme (Bourigault & Fabre, 2000)

[2] Les règles de grammaires extraites d'un corpus arboré suivent une distribution de Zipf. On a comme corollaire que la taille de la grammaire augmente de manière constante avec la taille du corpus, et pour conséquence qu'une grammaire de corpus n'est pas bornée. On a donc une contradiction avec l'hypothèse fondamentale qu'une grammaire formelle est un objet borné, ce qui explique – entre autres – les difficultés méthodologiques à établir un traitement robuste sur corpus avec des grammaires dites de *compétence*. Par contraste, les méthodes probabilistes sont équipées de mécanismes bien étudiés pour le traitement de phénomènes de basse fréquence.

rence pour l'anglais, comprenant en particulier le corpus du Wall Street Journal (ci-après WSJ), auquel nous référons dans toute la suite. Utiliser tels quels ces algorithmes pour d'autres langues donne souvent des résultats décevants.

Nous présentons divers tests d'entraînement de parsers statistiques du français, avec l'objectif de : (i) tester sur le français le comportement d'un algorithme d'apprentissage non lexicalisé (Petrov *et al.*, 2006), (ii) tirer profit des spécificités du corpus d'entraînement utilisé : le French Treebank du laboratoire LLF (Abeillé *et al.*, 2003) (ci-après FTB), et particulièrement son annotation morphologique riche, (iii) contraster ces spécificités avec celles du WSJ.

Nous décrivons d'abord les caractéristiques pertinentes du FTB (Section 2), puis les travaux antérieurs sur le français et l'algorithme qu'ils utilisent (Section 3). Nous présentons ensuite l'algorithme que nous utilisons (Section 4) et discutons les expériences réalisées (Section 5).

2 Treebank français

Le FTB est le seul corpus arboré français. Disponible depuis 2003, il est le résultat d'un projet d'annotation supervisée d'articles du journal Le Monde, mené à l'université de Paris 7 depuis une dizaine d'années, sous la direction d'Anne Abeillé. Les annotations (cf. exemple en Figure 1) sont morphologiques et syntaxiques.

Ce corpus est toujours en cours de développement. Une version finalisée et complètement corrigée de la sous-partie du corpus annotée en fonctions syntaxiques(Abeillé & Barrier, 2004) est attendue prochainement. Nous avons travaillé avec une version provisoire de cette sous-partie, partiellement corrigée, qui comporte 12-351 phrases (dans toute la suite FTB réfère à cette version).

```
<SENT>
  <NP fct="SUJ">
    <w cat="D" ee="D-def-ms" ei="Dms" lemma="le" mph="ms" subcat="def">le</w>
    <w cat="N" ee="N-C-ms" ei="NCms" lemma="bilan" mph="ms" subcat="C">bilan</w>
  </NP>
  <VN>
    <w cat="ADV" ee="ADV-neg" ei="ADV" lemma="ne" subcat="neg">n'</w>
    <w cat="V" ee="V--P3s" ei="VP3s" lemma="être" mph="P3s" subcat="">est</w>
  </VN>
  <AdP fct="MOD">
    <w compound="yes" cat="ADV" ee="ADV" ei="ADV" lemma="peut-être">
       <w catint="V">peut</w>
       <w catint="PONCT">-</w>
       <w catint="V">être</w>
    </w>
    <w cat="ADV" ee="ADV-neg" ei="ADV" lemma="pas" subcat="neg">pas</w>
  </AdP>
  <AP fct="ATS">
    <w cat="ADV" ee="ADV" ei="ADV" lemma="aussi">aussi</w>
    <w cat="A" ee="A-qual-ms" ei="Ams" lemma="sombre" mph="ms" subcat="qual">sombre</w>
  </AP>
  <w cat="PONCT" ee="PONCT-S" ei="PONCTS" lemma="." subcat="S">.</w>
</SENT>
```

FIG. 1 – Exemple simplifié de la source du FTB

Nous donnons ci-dessous les caractéristiques du FTB par rapport au WSJ, qu'il s'agisse de caractéristiques liées à la langue, au corpus ou au schéma d'annotation choisi :

La taille : Le FTB compte 385 458 occurrences de tokens, soit environ 3 fois moins de mots que le WSJ.

Un grand nombre de composés : Contrairement au WSJ, les composés sont explicitement annotés dans le FTB (cf. le codage de *peut-être*, Figure 1), et sont très nombreux : 14,52% des occurrences de tokens entrent dans un mot composé. Ils incluent des séquences dont les composants n'existent pas isolément (*aujourd'hui*), dont la sémantique est non compositionnelle (*carte bleue*), ou dont la syntaxe n'est pas régulière (*à la va-vite*), des expressions verbales (*mettre en garde*), des entités nommées (*Union hospitalière privée*), des séquences à sémantique compositionnelle mais où l'insertion est peu ou pas possible (*garde d'enfants, commission exécutive*).

Les nombres en chiffres ou en lettres sont également marqués sous forme de composés (plus de 10% des occurrences de composés). Ainsi l'apprentissage à réaliser sur le FTB est double : pour une séquence `N1 prép N2`, il faut décider entre former un composé, ou attacher la préposition au `N1` ou plus haut dans l'arbre.

La longueur des phrases : Le FTB est segmenté en 12351 phrases dont la longueur moyenne est de 31 tokens contre 24 pour le WSJ.

La flexion du français : La flexion riche du français comparativement à l'anglais a potentiellement un fort impact sur l'entraînement d'un analyseur. Le FTB compte 24 098 formes distinctes, soit une moyenne de 16 occurrences par forme, contre 12 pour le WSJ. Cela a potentiellement un impact négatif par dispersion des données : l'utilisation brute (i.e. sans lemmatisation) du FTB impose un nombre de formes distinctes en moyenne 1,3 fois supérieur à celui d'un corpus équivalent anglais. À l'inverse, la flexion peut constituer un atout en fournissant des indices pour les rattachements syntaxiques.

Une annotation morphologique riche : Les formes fléchies (simples ou composées) sont réparties en 13 catégories principales (trait `cat`), contre 44 pour le WSJ. Mais les 13 catégories pour le français sont divisés en sous-catégories (trait `subcat`) : 34 en tout. Les traits flexionnels (trait `mph`) et le lemme sont explicités.

Une structure plus plate : Comme le WSJ, le FTB annote en constituants et pas en dépendances, mais avec une structure moins hiérarchisée. Par phrase, on trouve en moyenne 19,6 occurrences de symboles non terminaux autres que les catégories, contre 18,7 pour le WSJ, et 24 en normalisant sur la longueur des phrases. L'impact du schéma d'annotation est mal connu : deux hypothèses sont envisageables : (1) la structure plate faciliterait la tâche de parsing (moins de frontières à marquer) et (2) elle cause une dispersion des données (plus de parties droites concurrentes pour un même symbole gauche, ce qui compliquerait la tâche dans le cadre PCFG (cf. Section 3).

Fonctions syntaxiques : Les constituants du FTB sont également partiellement marqués par une fonction (strictement) syntaxique (cf. trait `fct`, Figure 1). Le WSJ comporte des symboles fonctionnels partagés entre fonctions syntaxiques et sémantiques.

3 Analyse syntaxique lexicalisée pour le français

Nous présentons ici, en indiquant leurs limites, les travaux antérieurs en analyse syntaxique statistique du français. Ceux-ci reposent principalement sur une application de techniques d'analyse dites lexicalisées, en adaptant au français l'algorithme d'analyse de Collins.

Les grammaires hors-contexte probabilistes (PCFG) sont un formalisme classique pour l'analyse syntaxique. Il s'agit d'un modèle de langage qui assigne en particulier une probabilité $P(t) = \prod_{A \to \alpha \in t} P(A \to \alpha)$ à tout arbre t engendré par la grammaire en posant une hypothèse d'indépendance conditionnelle entre les règles émises. L'analyse syntaxique désambiguisée consiste alors à renvoyer l'arbre t qui a la plus grande probabilité, parmi les analyses concurrentes pour une phrase. Si ce premier problème d'optimisation se résout techniquement en adaptant des algorithmes de programmation dynamique bien connus (analyse tabulaire, Viterbi), il reste que pour l'analyse du langage naturel, deux critiques sont formulées à PCFG : (1) les hypothèses d'indépendance conditionnelles sont trop fortes (2) le modèle accorde trop peu d'importance au lexique (les catégories de mots sont une généralisation trop forte). Un troi-

sième problème, pratique cette fois, est la dispersion des données. Dans le cas de grammaires de treebank, l'estimation de probabilités pour les règles apparaissant rarement est rendue difficile. Pour résoudre ce problème, les algorithmes d'analyse comportent des procédures de lissage qui ont un impact considérable sur leurs performances[3].

L'analyse syntaxique dite lexicalisée, introduite par (Collins, 2003) répond à la critique (2) en annotant les noeuds syntagmatiques par le mot tête, probabilisant ainsi des dépendances lexicalisées. Pour contrer l'effet de dispersion de données, Collins formule son modèle en posant des hypothèses d'indépendance supplémentaires, entre les symboles non-tête des règles de grammaire. C'est ce paradigme d'analyse qui permet d'obtenir les meilleurs parsers statistiques de l'anglais, appris sur le WSJ. Cependant, nous pensons que l'application de ce type de modèle au français pose plusieurs problèmes. Premièrement, Collins intègre des heuristiques dépendantes du schéma d'annotation (distinction argument/ajout, coordination) non applicables telles quelles au FTB. Deuxièmement, la caractéristique majeure du modèle lexicalisé, les dépendances bi-lexicales, est transposable telle quelle à une autre langue. Mais elle est fortement remise en cause comme explication des meilleures performances du modèle lexicalisé : (Gildea, 2001) montre non seulement que supprimer les dépendances lexicalisées a peu d'impact dans le cas où phrases d'entraînement et de test proviennent du même corpus (tests intra WSJ, ou intra Brown), mais en plus que cela n'a aucun impact dans le cas où phrases d'entraînement proviennent du WSJ et phrases de test proviennent du Brown corpus. En bref, les dépendances lexicalisées sont rarement disponibles et c'est le lissage qui s'applique en général, a fortiori lorsque l'on change de corpus. Ce point est crucial pour une chaîne de traitement robuste, et également pour l'apprentissage à partir du FTB, corpus de petite taille.

Cette observation est renforcée par les résultats mitigés obtenus avec des analyseurs lexicalisés, à partir de versions antérieures et/ou modifiées du FTB : (Arun & Keller, 2005) et (Schluter & van Genabith, 2007). Ils ont été amenés à modifier les structures de données (automatiquement pour les premiers, avec réannotation manuelle pour les seconds), pour se rapprocher des hypothèses sous-jacentes à l'algorithme de Collins. (Arun & Keller, 2005) obtiennent un F-score de 80.45 sur un corpus de 20609 phrases, et (Schluter & van Genabith, 2007) obtiennent 79.95 sur un corpus réannoté d'environ 4700 phrases[4]. (Nasr, 2006) décrit des expériences de parsing probabiliste en dépendances, à partir du FTB, mais nous ne sommes pas actuellement en mesure de comparer ses résultats à ceux obtenus avec une analyse en constituants.

4 Analyse statistique non lexicalisée

Nous proposons ici d'investiguer des analyseurs qui relèvent du paradigme dit non lexicalisé (Johnson, 1998; Klein & Manning, 2003). Ceux-ci répondent à la critique (1) de PCFG (supra) en utilisant des transformations du treebank internes au processus d'entraînement/analyse. Les

[3]Ce problème est particulièrement vrai dans le cas d'un corpus à structure plate : les grammaires induites du FTB comportent 13148 règles contre 8433 pour un corpus anglais équivalent, extrait du WSJ (Section 5).

[4](Schluter & van Genabith, 2007) estiment par régression linéaire un résultat de 82.44 en passant au corpus français total, y compris phrases non fonctionnellement annotées, soit 18548 phrases. Les scores donnés correspondent à des expériences avec algorithme lexicalisé, fusion des composés, tagging interne, et évaluation PARSEVAL. Les différences avec nos résultats fournis Section 5, sont l'algorithme (non lexicalisé), la non prise en compte de la ponctuation dans l'évaluation PARSEVAL, et le corpus d'entraînement : ces auteurs ont travaillé chacun avec une sous-partie différente d'une version antérieure du corpus. Anne Abeillé a depuis fourni une version des phrases fonctionnellement annotées avec des corrections supplémentaires.

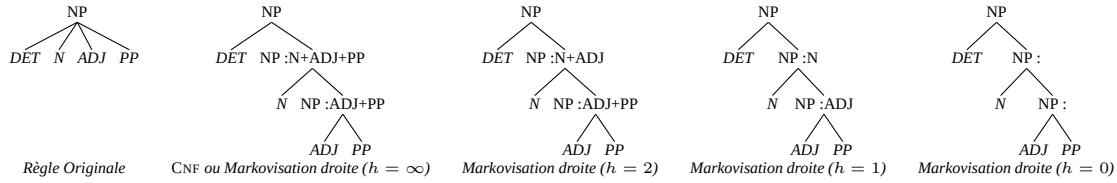

FIG. 2 – Exemples de markovisations horizontales d'ordre h

manipulations effectuées sont de deux types : (a) Modifications de structure : une forme spécifique de binarisation des arbres, la markovisation horizontale, est couramment utilisée pour pallier à la dispersion des données (Klein & Manning, 2003) comme illustré en figure 2[5] (b) Modifications de l'étiquetage des noeuds par spécialisation / généralisation des catégories syntagmatiques ou lexicales pour réduire les hypothèses d'indépendances trop fortes de PCFG.

Si (Klein & Manning, 2003) montrent que la combinaison de ces techniques de précompilation augmentent considérablement la correction (et l'efficacité) de l'analyse, les modifications de type (a) et (b) pêchent par leur caractère procédural et leur interdépendance. Les définir manuellement reste laborieux. Aussi, (Matsuzaki *et al.*, 2005) améliorent cette première version en la simulant par apprentissage[6]. Ils définissent un modèle appelé PCFG-LA ou PCFG augmentée de symboles latents (cachés). La grammaire latente est engendrée automatiquement en combinant tout non terminal d'une grammaire induite du treebank à tout symbole caché pris dans un ensemble prédéfini, ce qui a pour effet de démultiplier considérablement la taille de la grammaire. Les paramètres de la grammaire latente sont estimés sur les arbres observés à l'aide d'une instanciation spécifique de l'algorithme Espérance-Maximisation (EM).

Afin d'assigner les symboles cachés de manière optimale, (Petrov *et al.*, 2006) proposent la méthode suivante : à partir d'une grammaire de base G_0 induite sur corpus, l'algorithme d'apprentissage crée itérativement n grammaires $G_1 \dots G_n$ (avec $n = 5$ en pratique). Chaque étape de l'itération comporte les étapes suivantes :

– SPLIT : produire une nouvelle grammaire G_i à partir de G_{i-1} en divisant chaque non terminal de G_{i-1} en deux nouveaux non terminaux. Estimer G_i par maximum de vraisemblance sur le treebank observé en utilisant une variante de inside/outside. Cette étape consiste à ajouter des annotations latentes.
– MERGE : Pour chaque symbole divisé à l'étape précédente : le fusionner à nouveau. Si la baisse de vraisemblance (utilisation d'une variante de inside) du treebank observé est faible, alors préserver la fusion, sinon préserver la division. Cette étape vise à éviter les divisions inutiles et à minimiser les risques de surentraînement.
– SMOOTH : Lisser les probabilités des règles qui ont le même symbole père par interpolation.

Pour le français cet algorithme a deux intérêts : (a) Une markovisation horizontale d'ordre 0 permet d'éviter un effet d'éparpillement des données dû au schéma d'annotation plat et à la petite taille du corpus d'entraînement et (b) la spécialisation de la grammaire est indépendante de toute hypothèse a priori sur la structure des arbres du treebank, contrairement aux hypothèses sous-jacentes aux modèles lexicalisés.

[5]La markovisation droite d'ordre h se formule comme suit : $P(A \rightarrow B_1 \dots B_n) =^{\text{def}} P(B_1, \dots, B_n|A) = \prod_{i=1}^{n-1} P(B_i|B_{i+1}, \dots, B_n, A) \approx \prod_{i=1}^{n-1} P(B_i|B_{i+1}, \dots, B_{i+h}, A)$. Ce que (Petrov *et al.*, 2006) implémentent en binarisant les arbres comme décrit en figure 2, c'est-à-dire en approximant cette formule par un fragment de dérivation produit par une PCFG.

[6]Ce qui donne en pratique des résultats équivalents et généralement meilleurs. L'apprentissage ne porte que sur les catégories et pas sur la structure.

5 Expériences

Les travaux en analyse syntaxique statistique pour le français cités précédemment ont exploité un minimum d'information du corpus arboré : les catégories principales (attribut `cat`) associées aux préterminaux ainsi que les catégories standard des non terminaux. Nous étudions ici comment tirer parti de l'information supplémentaire contenue dans le treebank à des fins d'analyse automatique. Il s'agit de tester l'impact de différents paramètres liés au schéma d'annotation du FTB sur l'algorithme de (Petrov *et al.*, 2006). Nous comparons enfin les performances du meilleur analyseur ainsi obtenu avec un apprentissage sur une sous-partie comparable du WSJ, afin d'évaluer la marge d'amélioration restante.

Protocole L'ensemble des observations que nous présentons ci-dessous se fondent systématiquement sur un argument d'évaluation, avec le protocole suivant : pour chaque instance engendrée, les 12351 phrases du treebank sont divisées en trois sections : (1) test (les 1235 premières phrases), (2) développement (les 1235 phrases suivantes), et (3) entraînement (le reste).

La tâche d'évaluation donne en entrée à chacun des analyseurs une chaîne parfaitement segmentée de manière déterministe. L'analyseur testé est chargé de produire un arbre d'analyse unique à comparer avec la référence. Les résultats d'évaluation sont reportés en utilisant le protocole PARSEVAL tel qu'implémenté par l'outil d'évaluation `evalb` avec paramètres standard de Collins. Autrement dit, les scores de précision, rappel et f-mesure tiennent compte du parenthésage mais également des catégories des noeuds[7]. Les résultats sont reportés pour les phrases de la section (1) dont le nombre de mots est ≤ 40.

Implantations utilisées Pour chaque expérience menée, nous utilisons systématiquement deux algorithmes d'analyse. Le premier, qui sert de témoin, est un analyseur PCFG « standard » dont l'étiquetage est réalisé par un étiqueteur trigramme (TNT/LNCKY)[8]. Pour chaque test, cet analyseur a été entrainé sur les sections (2) et (3).

De plus, nous utilisons l'analyseur de Berkeley (Petrov *et al.*, 2006), noté BKY, avec une markovisation horizontale h=0[9]. Nous avons adapté au français (nous inspirant de (Arun & Keller, 2005)) le modèle de lissage lexical de l'analyseur : il fonctionne par clustering de mots inconnus et utilise des indices de capitalisation, marques typographiques et suffixes discriminants. Cet analyseur est entraîné sur (3) et utilise la section (2) pour ajuster les paramètres de EM.

Expériences Les expériences que nous avons réalisées testent principalement 4 facteurs : (1) l'impact des mots composés sur la tâche d'analyse, (2) l'impact de l'annotation morphologique, (3) l'impact de la flexion riche du français, et (4) l'usage de fonctions syntaxiques. Finalement, nous mettons en perspective les résultats obtenus pour le français avec ceux obtenus sur un corpus anglais approximant les propriétés formelles du corpus français[10].

[7] Suivant les conventions classiques utilisées pour l'évaluation sur le Penn TreeBank, nous avons ajouté un noeud racine artificiel à chacun des arbres ; et les noeuds de ponctuation sont ignorés dans l'évaluation, ce qui n'est pas le cas pour (Schluter & van Genabith, 2007)

[8] Formellement, il ne s'agit donc pas strictement d'un algorithme PCFG complètement standard. Il s'agit plutôt d'un compromis « pratique » où l'étiqueteur a pour fonction annexe de fournir un algorithme de lissage. L'étiqueteur utilisé est TNT (Brants, 2000), l'analyseur est l'implémentation LNCKY distribuée par Mark Johnson (`http ://www.cog.brown.edu/~mj/Software.htm`).

[9] Pour le corpus TREEBANK+ (infra), nous obtenons F=86.41 pour h=0, 84.84 pour h=1, et 82.85 pour h=2.

[10] Les tests sont réalisés suite à deux normalisations : (1) les nombres en chiffres arabes parfois encodés en plusieurs tokens, sont systématiquement fusionnés en un seul token. (2) Les amalgames (ex. *à+le = au, à+lequel*

Mots composés Les composés explicites du FTB complexifient la tâche d'apprentissage. Une solution pour simplifier le problème est de **fusionner** les composés, suivant en cela (Arun & Keller, 2005) et (Schluter & van Genabith, 2007). Par exemple un composé initialement marqué `(Adv (P de) (Adv même))` est remplacé par `(Adv de_même)`. Cela facilite la tâche, et suppose une utilisation du parser obtenu avec en entrée une fusion parfaite des composés.

Voici quelques expériences ne supposant pas ce pré-traitement (Table 1). En n'utilisant que la catégorie principale (TREEBANKMIN), on obtient F-score=83.09 sans fusionner, à comparer à F-score=84.85 avec fusion. Pour capturer qu'un composant n'a pas la même distribution qu'un mot simple, nous avons testé de distinguer par un suffixe les symboles de composants de composés `(Adv (P* de) (Adv* même))`, ce qui finalement n'a pas d'impact (F-score = 83.09).

Enfin, pour obtenir un parser moins dépendant de la définition assez large de composé du FTB, nous avons également cherché sommairement à ne conserver que les composés syntaxiquement non réguliers (par exemple `(ADV (P en) (A particulier))`, où la séquence `P A` se comporte comme un adverbe). Pour cela nous testons de 'défaire' les composés à syntaxe régulière ayant les patrons les plus productifs : nous défaisons les composés de patrons `N=(N A? P D? N)`, `N=(A N)` et `N=(N A A?)`, soit 5854 occurrences de composés sur 20413. Par exemple le sous-arbre pour le composé `(N (N loyer) (P de) (D l') (N argent))` est remplaçé par une suite de deux noeuds `(N loyer)` et `(PP (P de) (NP (D l') (N argent)))`, qui s'insèrent comme fils du NP père. A noter que cela modifie le nombre de constituants pris en compte par PARSEVAL, donc le F-score obtenu (84.97) est meilleur mais pas comparable. En revanche la diminution du nombre moyen de constituants qui croisent un constituant correct est un signe que les dépendances syntaxiques régulières internes aux composés défaits sont globalement recapturées.

Dans toute la suite nous donnons des résultats en mode fusionné, ce qui facilite les comparaisons avec (Arun & Keller, 2005) et (Schluter & van Genabith, 2007).

TRAITEMENT DES COMPOSÉS SUR TREEBANKMIN	PRÉC.	RAPPEL	F-SCORE	NB MOYEN CROISEMENTS	NB DE PHRASES TEST ≤ 40 MOTS
Composés fusionnés	85.25	84.46	84.85	0.94	992
Composés tels quels	83.44	82.73	83.09	0.92	932
Composants marqués	83.35	82.82	83.09	0.93	932
Composés 'défaits' + Composants restants marqués	85.21	84.74	84.97	0.88	932

TAB. 1 – Différents traitements des composés sur TREEBANKMIN

Annotation morphosyntaxique Le FTB comporte trois champs morphosyntaxiques : la catégorie principale (champ `cat`), une sous-catégorie (champ `subcat` raffinant `cat`), comme par exemple le défini, l'interrogatif, le démonstratif ou le possessif pour un déterminant, ainsi qu'un champ `mph` comportant des traits flexionnels (par exemple genre, nombre, personne). De manière à tester l'impact des informations contenues dans ces champs, nous avons instancié un corpus TREEBANKSUBCAT, dont les préterminaux sont la concaténation des champs `cat` + `subcat`, ainsi qu'un corpus TREEBANKMAX dont les préterminaux sont la concaténation des trois champs `cat` + `subcat` + `mph`.

En comparant TREEBANKSUBCAT et TREEBANKMIN (table 3), on constate que l'annotation

= *auquel* encodés en deux tokens dont un vide sont recodés en un seul, de catégorie `P+D` ou `P+PRO` selon le cas.

TAG	CAT	SOUS-CAT	MODE	TAG	CAT	SOUS-CAT	MODE	TAG	CAT	SOUS-CAT	MODE
V	V	-	indicatif	CLS	CL	suj	-	ADJWH	A	int	-
VIMP	V	-	impératif	CLO	CL	obj	-	ADJ	A	¬int	-
VINF	V	-	infinitif	CLR	CL	refl	-	ADVWH	ADV	int	-
VS	V	-	subjonctif	P	P	-	-	ADV	ADV	¬int	-
VPP	V	-	participe passé	P+D		*voir texte*		PROWH	PRO	int	-
VPR	V	-	participe présent	P+PRO		*voir texte*		PROREL	PRO	rel	-
NPP	N	P	-	I	I	-	-	PRO	PRO	¬(int | rel)	-
NC	N	C	-	PONCT	PONCT	-	-	DETWH	D	int	-
CS	C	S	-	ET	ET	-	-	DET	D	¬int	-
CC	C	C	-								

TAB. 2 – Symboles préterminaux (tags) de TREEBANK+

en sous-catégories a un impact statistiquement significatif ($p = 0.015$)[11] sur les performances de l'analyseur de BKY. Par contre ajouter toute l'information morphologique (TREEBANKMAX), dégrade significativement les résultats pour l'analyseur BKY.

C'est une solution intermédiaire, le corpus TREEBANK+, qui permet d'obtenir les meilleurs résultats, en sélectionnant uniquement le mode des verbes et certains traits de sous-catégories (table 2). L'intuition derrière le choix de ce jeu de préterminaux est analogue à celle indiquée dans (Schluter & van Genabith, 2007) : on a ici guidé l'apprentissage en distinguant les catégories verbales selon le mode, qui a un impact considérable pour capturer l'ordre des mots autour du verbe dans une grammaire. Le résultat pour TREEBANK+ est une amélioration statistiquement significative des résultats par comparaison avec TREEBANKSUBCAT ($p = 0.002$).

CORPUS	PRÉCISION	RAPPEL	F_1-SCORE	COUVERTURE	TAG ACCY	F_1-SCORE (TNT/LNCKY)
TreebankMin	85.25	84.46	84.85	99.59	97.35	69.68
TreebankMax	84.17	84.08	84.13	99.69	92.20	74.76
TreebankSubcat	85.91	85.58	85.74	99.59	96.63	72.54
Treebank+	86.57	86.25	**86.41**	99.79	96.83	75.02

TAB. 3 – Évaluation de l'impact de la richesse du jeu de préterminaux

Lexicalisation / Flexion Pour évaluer l'impact de la flexion riche du français, soulignée section 2, nous comparons l'utilisation de formes fléchies versus l'utilisation de symboles regroupant des formes fléchies. Un regroupement par lemme semble trop grossier : nous testons plutôt un regroupement selon les catégories du jeu TREEBANK+. Pour ce faire, nous remplaçons une forme fléchie par (tag+lemme)[12], y compris dans les phrases de test, ce qui induit un tagging parfait[13]. Ce cas est donc à comparer à un équivalent en formes fléchies non regroupées, en tagging parfait, simulé en remplaçant une forme fléchie par (tag+forme). Les résultats sont donnés table 4. Le regroupement de formes a effectivement un impact positif, quoique faible (F-score de 0.39 point supérieur à l'équivalent en formes fléchies, et meilleur nombre moyen de croisements). La moindre dispersion des données a plus d'effet que la perte des marques d'accord [14].

Fonctions syntaxiques Nous avons testé d'encoder dans les non terminaux les fonctions syntaxiques annotées dans le treebank. Cela crée de manière prévisible une dispersion des données,

[11] En utilisant un t-test unidirectionnel d'écarts à la moyenne pour échantillons appariés.

[12] Par exemple, les différentes formes de *manger* sont remplacées par 6 symboles selon le jeu TREEBANK+ : manger-VINF, manger-VPP, manger-VPR, manger-V, manger-VS, manger-VIMP. Les deux formes d'un nom commun sont remplacées par une seule, etc...

[13] Le lissage lexical est modifié en conséquence. Nous envisageons une autre solution, sans le biais du tagging parfait, qui consisterait à disposer d'un tagger donnant en entrée à l'analyseur des séquences de couples tag+lemme.

[14] Ces résultats sont à comparer avec le remplacement simple des mots par leur tag, ce qui donne une version purement non lexicalisée, en tagging parfait. On obtient 86.28, ce qui montre bien le peu d'utilisation de la lexicalisation par BKY (dit *non lexicalisé*, mais où la lexicalisation intervient par division des catégories lexicales).

TERMINAUX	PRÉCISION	RAPPEL	F-SCORE	COUVERTURE	TAG ACCY	NB MOY. CROISEMENTS
tag+forme	87.85	87.73	87.79	99.89	99.74	0.86
tag+lemme	86.90	88.16	88.18	99.89	99.80	0.82
tag seul	86.28	86.27	86.28	99.89	100	0.95

TAB. 4 – Regroupement de formes fléchies sur TREEBANK+ (en tagging parfait simulé)

dégradant ainsi les résultats (F-score=78.73 pour TREEBANK+).

Evaluation sur corpus anglais comparable À titre indicatif et pour estimer la dépendance à la langue, nous avons construit un échantillon du WSJ formellement analogue au corpus français[15], et comparons les résultats. Pour l'analyseur TNT/LNCKY, on obtient F=71.84 pour l'anglais (vs F=75.02 pour le français en TREEBANK+). Alors que le contraste est inversé avec BKY : F=88.61 pour l'anglais, versus F=86.41 pour le français. Pour l'anglais, on peut également remarquer que l'échantillon utilisé, qui divise la taille du WSJ par trois, ne fait baisser les résultats que de 1.5 point : (Petrov *et al.*, 2006) obtiennent 90.15 sur la totalité du WSJ.

6 Conclusion et perspectives

Cet article montre qu'il est possible, à partir du FTB et avec un algorithme non lexicalisé, d'obtenir un analyseur syntaxique statistique satisfaisant pour le français sur corpus journalistique. Il est obtenu sur TREEBANK+ et donne un F-Score de 86.41, le meilleur à ce jour à partir du FTB.

Les désavantages potentiels du FTB sont contournés par un algorithme non lexicalisé relativement indépendant du schéma d'annotation : une markovisation horizontale radicale ($h = 0$) diminue l'effet de dispersion des règles, dû à la petite taille du corpus et à la faible compacité de la grammaire sous-jacente. Cette hypersimplification initiale est contre-balancée dans un second temps par un algorithme de fusion/séparation des symboles qui maximise le degré de granularité (et atténue les effets des hypothèses d'indépendance conditionnelles) de la grammaire induite.

On remarque également que tirer parti de l'information supplémentaire encodée dans le treebank (traits `subcat,mph` et `lemma`) a un impact sur les performances de l'analyseur syntaxique non lexicalisé. On pense prolonger ce travail en augmentant ce premier analyseur appris d'un algorithme de reranking (Charniak & Johnson, 2005) spécifique au français intégrant des traits non locaux. Il sera en particulier intéressant d'étudier comment exploiter d'avantage l'information lexicale dans cette seconde passe. Les premiers résultats en fonctions syntaxiques sont considérés comme très encourageants. Nous envisageons tester différentes techniques d'étiquetage fonctionnel et d'extraction de dépendances fonctionnelles en sortie d'analyse. Cela permettrait d'une part la comparaison avec d'autres analyseurs syntaxiques pour le français, et d'autre part une évaluation interne plus fine, par type de dépendance.

Les résultats obtenus pour l'anglais sur un échantillon du WSJ formellement analogue au FTB, montrent qu'il reste certainement une marge de progression : à corpus formellement comparables, les résultats pour l'anglais sont 2 points au-dessus du français. L'hypothèse que cette

[15]Cet échantillon a été constitué par une procédure aléatoire augmentée de deux contraintes : (1) l'échantillon anglais comporte le même nombre de tokens que TREEBANK+ et (2) la procédure fait converger les distributions en longueur des phrases. L'algorithme d'échantillonnage garantit la convergence des moyennes. Suivant la pratique courante nous avons supprimé les traces et les annotations fonctionnelles du Penn Treebank.

différence provient d'une flexion plus riche du français apparaît comme insuffisante : le gain obtenu en minimisant la dispersion par flexion est décevant. Pour tenter d'expliquer cet écart, nous pensons investiguer l'utilisation de modifications structurelles automatisables.

Remerciements Les auteurs tiennent à remercier Anne Abeillé, Laurence Danlos, Slav Petrov, Natalie Schluter et Djamé Seddah pour leurs conseils lors de la réalisation de ce travail ainsi que l'université Paris 7 (Prix Diderot innovation) pour son soutien financier.

Références

ABEILLÉ A. & BARRIER N. (2004). Enriching a french treebank. In *Proceedings of Language Ressources and Evaluation Conference (*LREC*)*, Lisbon.

ABEILLÉ A., CLÉMENT L. & TOUSSENEL F. (2003). *Treebanks*, chapter Building a treebank for French. Kluwer : Dordrecht.

ARUN A. & KELLER F. (2005). Lexicalization in crosslinguistic probabilistic parsing : The case of french. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, p. 306–313, Ann Arbor, MI.

BOURIGAULT D. & FABRE C. (2000). Approche linguistique pour l'analyse syntaxique de corpus. *Cahiers de grammaires*, **25**.

BRANTS T. (2000). Tnt – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference (ANLP)*, Seattle-WA.

CHARNIAK E. & JOHNSON M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, Ann Arbor (MI).

COLLINS M. (2003). Head-driven statistical models for natural language parsing. *Computational Linguistics*, **29**(3).

GILDEA D. (2001). Corpus variation and parser performance. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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

KLEIN D. & MANNING C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*.

MARCUS M. P., SANTORINI B. & MARCINKIEWICZ M. A. (1994). Building a large annotated corpus of english : The penn treebank. *Computational Linguistics*, **19**(2), 313–330.

MATSUZAKI T., MIYAO Y. & TSUJII J. (2005). Probabilistic cfg with latent annotations. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, p. 75–82.

NASR A. (2006). Grammaires de dépendances génératives probabilistes. modèle théorique et application à un corpus arboré du français. *Traitement Automatique des Langues*, **46**(1).

PETROV S., BARRETT L., THIBAUX R. & KLEIN D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia : Association for Computational Linguistics.

SCHLUTER N. & VAN GENABITH J. (2007). Preparing, restructuring, and augmenting a french treebank : Lexicalised parsers or coherent treebanks ? In *Proceedings of PACLING 07*.