Université Paris III - Sorbonne Nouvelle
Master de Traitement Automatique des Langues, parcours Recherche & Développement

Extraction de lexiques syntaxiques à partir d'un treebank - Le cas du treebank SUD du naija, un pidgin créole de l'anglais

SONG Yuchen

Mémoire dirigé par **Sylvain Kahane** et **Kim Gerdes**

Année universitaire 2019-2020

Attestation de non-plagiat

Déclaration sur l'honneur

Je, soussigné (e), déclare avoir rédigé ce travail sans aides extérieures ni sources autres que celles qui sont citées. Toutes les utilisations de textes préexistants, publiés ou non, y compris en version électronique, sont signalées comme telles. Ce travail n'a été soumis à aucun autre jury d'examen sous une forme identique ou similaire, que ce soit en France ou à l'étranger, à l'université ou dans une autre institution, par moi-même ou par autrui.

Date

Signature manuscrite de l'étudiant·e

Acknowledgement

Je remercie mes directeurs de stage et de mémoire, Sylvain Kahane et Kim Gerdes pour leur soutien constant tout au long de mon stage et de la rédaction du mémoire. Je leur remercie de m'avoir donné l'opportunité de travailler avec l'équipe du projet ANR NaijaSynCor, ce qui était un projet très intéressant et inspirant pour moi.

Abstract

Ce mémoire décrit un travail de recherche qui vise à extraire deux lexiques à partir d'un corpus annoté en syntaxe. Nous utilisons un corpus annoté en syntaxe de dépendance, également appelé *dependency treebank*. Ce treebank, constitué dans le cadre du projet NaijaSynCor, est un treebank de la langue de naija - un pidgin créole à base lexicale anglaise. À partir des annotations des treebanks, nous extrayons des informations sur les lexèmes et leur distribution au sein des structures de dépendance. Ce sont des informations nécessaires pour construire les deux lexiques, un lexique morphosyntaxique dont chaque entrée est associée à ses traits morphologiques et un lexique syntaxique dont chaque entrée est associée à ses cadres de sous-catégorisation.

Nous proposons également les modes d'emploi pour les deux lexiques, y compris la fouille des erreurs d'annotation à l'aide du lexique morphosyntaxique, le regroupement des cadres de sous-catégorisation, et l'analyse statistique des informations extraites. Ainsi, lors de l'extraction et la mise en application des lexiques, nous avons rencontré des problèmes intéressants qui motivent constamment notre travail : Quelle sont les phénomènes grammaticaux particuliers de la langue naija ? Comment traiter les erreurs d'annotation liées au mot qui correspond à plusieurs parties du discours ? Comment formaliser un cadre de sous-catégorisation ? Quelles sont les processus pour préparer des ressources linguistiques propres à exploiter ?

Table des Matières

Chapitre 1

Introduction

L'extraction des lexiques à partir des treebanks joue un rôle important dans le domaine du Traitement Automatique des Langues parce que les informations morphosyntaxiques et syntaxiques aident à améliorer la couverture et la précision des systèmes de TAL. Briscoe & Carroll (1993) estiment qu'environ la moitié des erreurs des analyseurs syntaxiques repose sur des informations insuffisantes concernant la structure argumentale. Carroll & Fang (2004) montre que le taux d'analyses correctes augmente de 15 % lorsque le parseur basé sur la grammaire HPSG (Head Driven Phrase Structure) est enrichi par des informations syntaxiques détaillées. Les lexiques sont également importants pour la génération automatique de textes (Danlos, 1985), la traduction automatique (Han et al., 2000), et l'extraction d'information, cf. (Surdeanu et al., 2003). L'extraction des lexiques est un processus de rassemblement et de regroupement sur les masses de données d'annotations du corpus annoté en syntaxe de dépendance. L'exploitation des lexiques extraits permet de mieux connaître le vocabulaire et l'usage des mots de la langue. En analysant statistiquement des entrées du lexique, nous pouvons savoir plus précisément quels sont les mots courants pour un corpus d'un contenu particulier, quels sont des usages fréquents du mot, et quels sont des phénomènes linguistiques rares de la langue. En plus, les lexiques peuvent aider à la fouille des erreurs d'annotation. En projetant des informations morphosyntaxiques du token du lexique à celles des treebanks, nous pouvons détecter des erreurs d'annotations potentielles et remplir des informations manquantes. Notre motivation principale est donc d'extraire deux lexiques à partir du treebank SUD du naija : un lexique morphosyntaxique dont chaque entrée est associée à ses traits morphologiques, et un lexique syntaxique dont chaque entrée est associée à ses cadres de sous-catégorisation. En plus, ces lexiques nous permet d'étudier le vocabulaire et l'usage du mot en naija, et de rechercher les erreurs d'annotation pour rendre le treebank propre.

Dans le chapitre 2, nous présenterons les notions clés en syntaxe, le projet NaijaSynCor et le treebank, ce sont des connaissances importantes pour dérouler nos travaux sur le treebank SUD du naija. Dans le chapitre 3, nous montrerons respectivement en détail les travaux sur les deux lexiques, y compris les méthodes d'extraction, l'analyse des résultats, l'application et les travaux futurs. Dans le chapitre 4, nous conclurons notre travail en discutant des limites de notre travail et ce qu'il reste à faire.

Chapitre 2

État de l'art

Dans ce chapitre, nous présentons les notions-clés en syntaxe. Nous commençons par la grammaire de base du schéma d'annotation du treebank - la grammaire de dépendance, nous présentons brièvement l'histoire de la grammaire universelle et comparons les différences entre la grammaire de dépendance et la grammaire de constituant. Ensuite, nous montrons le schéma d'annotation du treebank, SUD, une alternative à UD. Nous traitons principalement trois questions : Qu'est-ce que c'est le projet UD ? Qu'est-ce que c'est le schéma d'annotation SUD ? Et pour quelles raisons nous utilisons le SUD au lieu d'UD ? Et puis nous introduisons la notion de sous-catégorisation et de cadre de sous-catégorisation. C'est basé sur ces notions que nous extrayons deux lexiques à partir de notre treebank - un lexique morphosyntaxique et un lexique syntaxique. Enfin, nous présentons le projet NaijaSynCor dont les travaux du trebank font partie, et le format CONLL-U qui enregistre les annotations du treebank.

2.1 Grammaire universelle

Depuis le Moyen-Age, les linguistes s'attachent à établir une grammaire universelle (généralement supposée innée) et à saisir les points communs de différentes langues. L'idée d'une grammaire universelle remonte au 13e siècle, lorsque le philosophe anglais Roger Bacon souligne dans ses ouvrages *Overview of Grammar* (1245) et *Greek Grammar* (1268) que toutes les langues partagent une grammaire commune, même si elles ont leurs propres variations culturelles. En linguistique moderne, cette notion a été développée par Noam Chomsky dans les années 1960s. Selon Chomsky, la grammaire universelle est un ensemble de règles générales de grammaire que les êtres humains sont nés avec une capacité innée d'acquérir leurs connaissances linguistiques (V. J. Cook, 1985). En observant l'apprentissage du langage des enfants, Chomsky a établi la terminologie "pauvreté du stimulus" en 1980 selon laquelle le langage est inné à un certain degré vu que le langage que reçoit un enfant (le stimulus) ne suffit pas pour l'apprendre tous les complexités du grammaire de la langue. Généralement, les connaissances linguistiques acquises se réfèrent aux connaissances lexicales et phonétiques, alors que les connaissances linguistiques innées sont principalement syntaxiques et sémantiques (McGinn, 2015). C'est-à-dire qu'il est important de saisir les points communs de la syntaxe des différentes langues pour créer une grammaire universelle. Par exemple, la phrase verbale s'organise autour d'un verbe (conjugué à un mode personnel ou impersonnel), qui constitue le noyau de la phrase. En plus, il existe toujours des

règles syntaxiques particulières qui distinguent chaque langue que les linguistes doivent prennent en compte. Par exemple, en français, lorsque le COD ou le COI est remplacé par un pronom, celui-ci se trouve juste avant le verbe, mais en anglais, le pronom complément est toujours placé après le verbe.

La théorie de la grammaire universelle de Chomsky est une motivation pour les linguistes de trouver une grammaire universelle en identifiant tous les règles communes et les règles spécifiques limitées de différentes langues. La théorie de la grammaire universelle, qui n'est pas encore remplie, permet théoriquement de construire un schéma d'annotation syntaxique applicable à toutes les langues, le projet UD est l'un d'entre eux.

2.2. Grammaire de dépendance

Le schéma d'annotation UD propose des analyses syntaxiques de phrases en fonction de la grammaire de dépendance (Nivre et al., 2016). La grammaire de dépendance est une approche à la syntaxe fondée par Lucien Tesnière représentant les liens syntaxiques entre les mots d'une phrase. La dépendance syntaxique représente le fait que la présence d'un mot et tous ses dépendants est légitimée par un autre mot, son gouverneur (Kahane, 2001). La grammaire de dépendance est différente de la grammaire de constituants (phrase structure grammar). Ce dernier est formalisée par Chomsky dans les années 1950s, elle est basée sur la structure de constituants dans laquelle chaque (constituant) unité syntaxique est formées par un ou plusieurs mots fonctionnant ensemble, le constituant maximal peut être une phrase complète. Tandis que la grammaire de dépendance est basée sur la structure de dépendance, qui représente le lien un à un entre chaque élément dans une phrase, c'est-à-dire chaque mot fonctionne comme une unité syntaxique et il n'y a qu'une seule relation syntagmatique qui relie deux unités minimales.

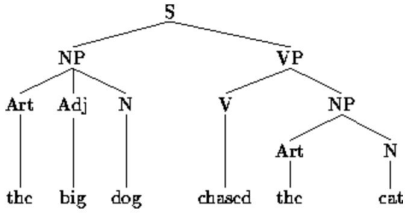

Figure 2.1 : Exemple de l'arbre de constituent
EAGLES, 1996. Recommendations for the Syntactic Annotation of Corpora

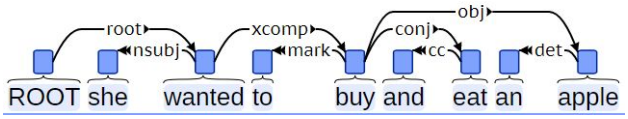

Figure 2.2 : Exemple de l'arbre de dépendance
https://universaldependencies.org/u/overview/syntax.html

En grammaire de constituants, une phrase est composée d'un groupe nominal (noun phrase NP) et d'un groupe verbal (verb phrase VP), qui joue respectivement le rôle de sujet et de prédicat. Alors qu'en grammaire de dépendance, elles sont liées par une relation parentale "nsubj". Nous pouvons dire que la structure de constituant décrit mieux la hiérarchie interne d'une phrase en montrant la relation entre les groupes lexicaux. Cependant, la structure de dépendance est plus concentrée (Ninio 2006, Hudson 2007, Osborne et al. 2011) puisque les dépendances sont les liens établis entre deux unités syntaxiques minimales d'une phrase, ce qui est plus simple et plus flexible pour les recherches de TAL.

Au cours de ces dernières années, de différents treebanks multilingues sont établis pour que les linguistes puissent étudier et améliorer la théorie de la grammaire de dépendance. Stanford Dependencies, lancé en 2005, est un projet d'annotation basée sur la grammaire de dépendance dans le but d'analyser les relations grammaticales entre les mots des phrases anglaises, puis il a été développé pour d'autres langues. En 2014, ce projet a été fusionné avec Google universal part-of-speech tags (Petrov et Slav, 2011) et Interset Interlingua for morphosyntactic tagsets (Zeman, 2008), formant un nouveau projet sous le nom de Universal Dependencies.

2.3. Projet Universal Dependencies

Le projet Universal Dependencies (UD) est un projet collaboratif visant à construire des treebanks de différentes langues et à proposer un schéma d'annotation syntaxique cohérent et cross-linguistiquement applicable afin de faciliter l'apprentissage d'analyseurs syntaxiques multilingues. UD est un ensemble de projets menés par plusieurs groupes de recherche à travers le monde qui rédigent les guides d'annotation adaptés aux différentes langues et construisent des treebanks selon les principes de base d'UD. Auu 15 mai 2020, 163 treebanks décrivant 92 langues ont été développés.

Le schéma d'annotation UD produit des analyses syntaxiques de phrases en fonction de la notion de dépendance issue de la grammaire de dépendance. La structure syntaxique UD est un arbre de dépendance dont les sommets sont les mots ou morphèmes d'une phrase, les arêtes sont habituellement étiquetées par des fonctions syntaxiques. Dans une phrase, il n'y a qu'un seul mot qui joue le rôle de la tête de la phrase, la tête est étiquetées par la fonction syntaxique "ROOT". Chaque mot a un seul gouverneur, mais zéro, un ou plusieurs dépendants. Par exemple :

(1).

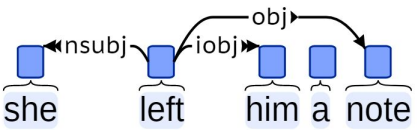

Figure 2.3 : Exemple du schéma d'annotation UD - 1
https://universaldependencies.org/u/overview/simple-syntax.html

Cet arbre montre que "she", "him" et "a note" sont les dépendants du verbe "left". Le pronom "she" est le sujet (nsubj) du verbe, le pronom "him" est un objet indirect (iobj) du verbe et le groupe nominal "a note" constitue un objet direct (obj) du verbe. Il existe encore une dépendance non indiquée dans cet arbre qui relie l'article indéfini "a" au nom "note", "a" est le déterminant (det) de "note".

(2).

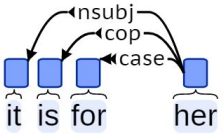

Figure 2.4 : Exemple du schéma d'annotation UD - 2
https://universaldependencies.org/u/overview/simple-syntax.html

Cet arbre identifie "it" comme le sujet (nsubj), "is" comme la copule (cop), et "for" comme un marqueur de cas (case), qui est considéré comme le dépendant de la tête de la phrase "her", qui est un pronom.

Le schéma d'annotation UD dispose aussi des conventions d'analyse spéciales. Premièrement, les relations de dépendance se tiennent principalement entre les mots lexicaux, plutôt que les relations indirectes indiquées par des mots grammaticaux (Nivre, 2015). Cela signifie que les gouverneurs sont principalement des têtes lexicales plutôt que des têtes fonctionnelles. Vu que UD vise à maximiser le parallélisme entre les langues, les mots lexicaux sont indispensable pour exprimer le sens dans les différentes langues, mais la nécessité des mots grammaticaux se varie selon la langue. Nous pouvons constater pleins de cas dans les configurations auxiliaire-verbe et adposition-complément.

Figure 2.5 : Exemple du schéma d'annotation UD - 3
https://universaldependencies.org/u/overview/syntax.html

Par exemple, en français, "On a dormi." a besoin d'un auxiliaire "avoir", mais la même phrase en anglais "We slept." n'en a pas. C'est la priorité des mots lexicaux

qui distingue le schéma d'annotation UD à la structure traditionnelle de la grammaire de dépendance.

Dans le schéma d'annotation UD, l'ordre d'annotation est les mots lexicaux, les mots grammaticaux et les ponctuations à la fin :

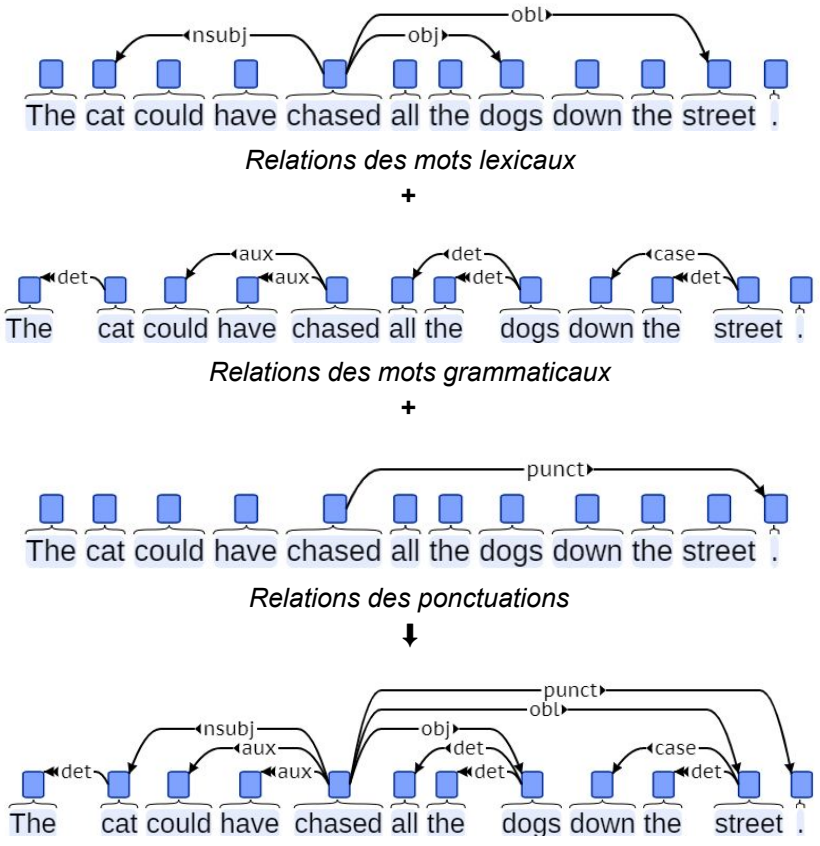

Figure 2.6 : Ordre d'annotation UD

https://universaldependencies.org/u/overview/syntax.html

Dans l'arbre de dépendance UD, les relations syntaxiques entre les mots lexicaux sont toujours supérieures que celles entre les mots grammaticaux parce qu'ils sont attachés à leur gouverneur - le mot lexical, comme ses dépendants directs.

Deuxièmement, la dépendance UD peut indique la catégorie fonctionnelle et la catégorie structurelle en même temps. De nombreuses étiquettes de relation syntaxique sont composées de deux parties, l'une représente la partie du discours du dépendant, l'autre montre la fonction syntaxique.

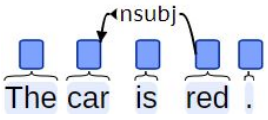

Figure 2.7 : Exemple de l'étiquette "nsubj" dans l'UD

https://universaldependencies.org/u/dep/nsubj.html

Par exemple "nsubj" (nominal subject) est composé de la partie du discours "NOUN" et de la relation syntaxique "subj". Ces étiquettes d'annotation sont héritées du parseur de Stanford Dependencies, un projet d'annotation basée sur phrase structure grammar. Ainsi, UD exprime non seulement les relations de dépendance, mais aussi les informations sur la structure de la phrase, qui diffèrent les unités syntaxiques ayant la même fonction syntaxique dans les différents types de phrases, par exemple, "nsubj" et "csubj" (clausal subject).

Il est vrai que le schéma d'annotation UD met l'accent sur le parallélisme entre les langues et ses étiquettes comporte plus d'informations syntaxiques et grammaticales que le schéma d'annotation SUD, cependant, sa complexité pourrait aussi poser des problèmes lors de l'analyse syntaxique pour certains projet. C'est pourquoi certains linguistes veulent améliorer ces insuffisances en développant un schéma d'annotation SUD basé sur UD.

2.4. Surface-Syntactic Universal Dependencies

Surface-syntactic Universal Dependencies (SUD), similaire à UD, est un schéma d'annotation syntaxique établi en 2018 par les linguistes Kim Gerdes, Bruno Guillaume, Sylvain Kahane et Guy Perrier. SUD a hérité autant que possible des caractéristiques et des principes d'UD, y compris le format CONLL-U et les étiquettes. Quant aux relations syntaxiques dans le SUD, les relations de dépendance et leurs étiquettes sont définies en fonction de critères purement syntaxiques (Mel'čuk 1988). Le schéma d'annotation SUD est considéré comme alternative à UD et non comme un schéma d'annotation concurrent, ce qui signifie que le schéma d'annotation SUD doit porter autant d'informations que UD, mais d'une manière différente (Gerdes et al., 2018). Les treebanks SUD, utiles pour l'enseignement et les études typologiques, peuvent simplement être converti à partir des treebanks UD. Également, les annotations peuvent être faites directement dans SUD, et finalement converties en UD. Le SUD est actuellement utilisé pour développer un treebank de naija (Courtin et al., 2018 ; Caron et al., 2019), et des treebanks pour le français et le chinois sont aussi en cours de construction.

Le SUD est créé pour résoudre deux problèmes majeurs qui existent dans l'UD et qui posent des problèmes pour l'analyse syntaxique.

Premièrement, UD ne tient pas compte de la hiérarchie entre les mots grammaticaux, et la priorité des mots lexicaux tend à aplatir la structure syntaxique. Dans l'UD, les relations entre les mots lexicaux ont une hiérarchie, le verbe dont la relation est "root" est la tête de la phrase, le reste des mots lexicaux sont reliés par leur gouverneur avec la relation de dépendance qui indique leur fonction syntaxique,

mais les mots grammaticaux, en revanche, sont tous attachés aux mots lexicaux voisins au même niveau. L'annotation centrée sur le mot lexical pose également des problèmes pour la cohésion interne du treebank (Gerdes et Kahane, 2016) et elle marque une rupture avec les traditions syntaxiques, où la relation syntaxique entre gouverneur et dépendant est définie par la distribution de chaque mot (Bloomfield 1933). En ignorant cette tradition, la structure syntaxique UD manque certaines informations importantes, par exemple, dans la figure 2.6, la relation de la structure préposition-verbe "chase...down..." n'est pas marquée puisque la préposition "down" est directement liée au nom "street". Contrairement à UD, les mots grammaticaux tels que les prépositions, les conjonctions de subordination, les auxiliaires peuvent être identifiés comme la tête. Et puisque les mots grammaticaux peuvent avoir son dépendant, au lieu d'attacher à son mot lexical voisin au même niveau, la profondeur de structure syntaxique est étendue. SUD peut établir une relation directe entre les verbes et les prépositions, ce qui est plus utile pour l'analyse des verbes à particule et pour les tâches qui nécessitent une analyse syntaxique distributionnelle plus stricte que UD, comme la fouille des erreurs grammaticales.

Deuxièmement, les étiquettes de relation syntaxique UD, héritées des Stanford Dependencies, comprennent des informations de la partie du discours et de la fonction syntaxique, par exemple "nsubj" où le "n" indique la partie du discours du dépendant et "subj" signifie le rôle de sujet. De ce fait, UD crée de différentes étiquettes qui représentent la même fonction syntaxique. Étant donné que la partie du discours n'est pas toujours nécessaires pour l'analyse syntaxique, cela augmenterait la difficulté de l'analyse syntaxique. Dans les étiquettes de relations syntaxiques SUD, les mots qui occupent la même position syntaxique et qui ont la même fonction syntaxique, sont reliés à leur gouverneur par la même relation syntaxique. Plus précisément, SUD introduit quatre nouvelles relations : subj, comp, mod et unknown. Tous les sujets ont la fonction subj, regroupant "nsubj" et "csubj" de UD. Tous les autres arguments de l'adjectif et du verbe ont la fonction comp, regroupant "obj", "iobj", "xcomp" et "ccomp" d'UD. Donc, SUD dispose des étiquettes de dépendance plus flexibles que UD.

2.5. Introduction des notions de "sous-catégorisation" et de "cadre de sous-catégorisation"

En linguistique, la sous-catégorisation indique la capacité et la nécessité pour les éléments lexicaux d'exiger et autoriser la présence et les types des arguments syntaxiques avec lesquels ils coexistent, Chomsky (1965) est une des premières sources importantes sur le concept de sous-catégorisation. Par exemple, un verbe transitif doit être suivi d'un objet direct NP, contrairement aux verbes intransitifs. On peut dire que les verbes transitifs forment une sous-catégorie de la catégorie VERBE, en vertu du fait qu'ils doivent être suivis d'un complément NP. C'est la présence de l'objet qui donne lieu à la sous-catégorie des verbes transitifs.

Pour former une phrase syntaxique correcte, il ne suffit pas de sélectionner les mots appropriés et de les mettre en l'ordre qui exprime le bon sens. Les différentes sous-catégorisations du verbe montrent des exigences différentes de leurs arguments. Ce genre de sélection diversifiée du verbe peut s'expliquer par la notion de sous-catégorisation, par exemple (Anna Korhonen, 2002) :

(a). Sam put the book on the table
(b). * Sam put the book
(c). * Sam put on the table
(d). * Sam put

Le verbe "put" prend le complément du type NP-PP (a), mais n'accepte pas l'existence individuel du complément NP (b) ou PP (c), ni une variance intransitive (d). Pour être grammaticalement correct, le verbe "put" a besoin d'au moins de trois arguments syntaxiques : un sujet, un objet et un objet oblique.

La structure de sous-catégorisations est souvent présentée sous forme d'un cadre syntaxique appelé le cadre de sous-catégorisation (SCF : Subcategorization Frames). Celui-ci fournit une généralisation sur divers contextes syntaxiques requis par les verbes associés au même comportement syntaxique. Par exemple, le cadre NP-PP peut caractériser la structure de sous-catégorisation du verbe "put" dans (a). Dans une annotation développée par Chomsky dans les années 1960, les positions du verbe et de ses arguments dans un arbre de constituant pourrait montrées par un cadre de sous-catégorisation. C'est ce qui distingue les verbes intransitifs, transitifs et ditransitifs. Selon Maggie Tallerman (2011), les verbes qui ne prennent qu'un seul argument sont classés comme intransitifs, tandis que les verbes à deux et trois arguments sont classés respectivement comme transitifs et ditransitifs. Par exemple (Chomsky, 1965 & Janez Orešnik, 1966) :

- Le SCF pour un verbe intransitif : [+ __]
- Le SCF pour un verbe transitif : [+ __ NP]

- Le SCF pour un verbe transitif : [+ __ NP NP]

L'underscore représente l'élément lexical lui-même, il forme avec [+] un cadre complet. Donc, le cadre de sous-catégorisation du verbe ditransitif ci-dessus indique que le verbe prend deux compléments consécutifs NP à son droit.

Voici un exemple précis :

(1). I sometimes donate money to WWF because I love animals.

donate, V, [+ __ NP PP]

Ici, V signifie que cet élément lexical "donate" est un verbe qui doit être suivi par deux compléments consécutifs, un groupe nominal NP, et un groupe prépositionnel PP.

Le cadre de sous-catégorisation est la spécification du nombre et des types d'arguments d'un élément lexical (généralement le verbe). Un élément lexical peut avoir plusieurs cadres de sous-catégorisation qui montrent ses capacités diverses de se combiner avec ses compléments. Il existe aussi des éléments lexicaux qui n'ont pas de cadre de sous-catégorisation. Le cadre de sous-catégorisation est associé le plus étroitement aux verbes, mais cette notion peut également être appliquée à d'autres catégories des éléments lexicaux comme des adjectifs, par exemple, "fier de sa fille", l'adjectif "fier" se combine avec un syntagme prépositionnel en "de".

La notion de sous-catégorisation est similaire à la notion de valence, la sous-catégorisation trouve son origine dans "phrase structure grammars" du Chomsky (1965), et la valence trouve son origine dans la grammaire de dépendance du Lucien Tesnière (1959). La principale différence entre les deux notions concerne la présence du sujet. Initialement, la sous-catégorisation n'inclut pas le sujet, c'est-à-dire un verbe est sous-catégorisé par son ou ses complément(s) (objets directs et obliques), mais pas par son sujet. Cependant, de nombreuses théories modernes incluent maintenant le sujet dans le cadre de la sous-catégorisation. La valence, en revanche, a inclus le sujet dès le début (Tesnière a souligné que d'un point de vue syntaxique, le sujet est un actant tout comme l'objet). Actuellement, la notion de sous-catégorisation est plus proche de la notion de valence parce que de nombreux "phrase structure grammars" pense que la sous-catégorisation du verbe est influencée par le sujet et l'objet, le sujet fait partie du cadre de sous-catégorisation (Pollard et Sag 1994, Kaplan et Bresnan 1982, Cattell 1984).

Selon (Anna Kupsc, 2007), la représentation des cadres de sous-catégorisation sont différentes en fonction de différents approches. Certains modèles théoriques, comme LFG (grammaire lexicale fonctionnelle) privilégient une notation basée sur

les fonctions syntaxiques (a), d'autres comme le LADL(Laboratoire d'Automatique Documentaire et Linguistique) privilégient une notation basée sur les catégories (b), d'autres enfin, comme en HPSG (grammaire syntagmatique guidée par les têtes), ont une approche mixte (c) qui combine la catégorie et la fonction pour obtenir l'information plus riche. Par exemple,

laver :
a. <SUJ, OBJ>
b. N0 V N1
c. <SUJ :NP, OBJ :NP>

2.6. Travaux précédents du lexique de sous-catégorisation

Le lexique de sous-catégorisation est un lexique syntaxique qui contient l'information sur le potentiel combinatoire d'un prédicat (le verbe intransitif régit un seul argument, le sujet), mais aussi sur le type de ses arguments (groupe nominal ou phrastique). Dans le domaine du traitement automatique des langues, les informations de sous-catégorisation détaillées sont importantes dans la plupart des applications. Selon Briscoe et Carroll (1993), presque la moitié des erreurs du parseur sont dues aux informations insuffisantes de la structure argumentale. Le lexique de sous-catégorisation est essentiel pour améliorer la couverture et la précision des systèmes de traitement automatique de la langue. Par exemple, Briscoe et Carroll (1993) montrent qu'environ 50 % des échecs d'analyse sur des données nouvelles sont liés aux informations de sous-catégorisation. Selon Carroll et Fang (2004), l'enrichissement des informations de sous-catégorisation peut aider à augmenter 15 % du taux d'analyses correctes lorsque le parseur basé sur la grammaire HPSG.

Il existe déjà de nombreux travaux sur l'extraction du lexique syntaxique à partir de treebanks. Pour l'anglais, les premiers travaux sont menés dès le début des années 90s (Manning 1993, Brent 1993). Ces travaux sont toutefois limités quant au nombre de verbes considérés et de cadres de sous-catégorisation possibles, parce qu'ils reposent souvent sur des heuristiques locales, sans exploiter pleinement le corpus. Le système développé à l'Université de Cambridge (Briscoe et Carroll, 1997) est le premier qui permet une acquisition à large échelle de bonne qualité. Le lexique syntaxique le plus important pour l'anglais extrait à partir du Penn-II Treebank a été obtenu par (O'Donovan et al., 2004), comme une ressource supplémentaire de l'induction des grammaires lexicalisées, voici un exemple d'entrée pour le verbe "rembourser" :

```
reimburse:
        (VERB          :ORTH "reimburse"      :SUBC  ((NP-NP)
                                                      (NP-PP :PVAL ("for"))
                                                      (NP)))
```

Chaque verbe possède une étiquette :SUBC, précisant son comportement de sous-catégorisation. Par exemple, "reimburse" peut se combiner avec deux groupes nominaux (NP-NP), un groupe nominal et un groupe prépositionnel avec "for" (NP-PP :PVAL ("for")) ou un seul groupe nominal (NP).

Il y a aussi des lexiques de sous-catégorisations adaptés pour d'autres langues, Sarkar et Zeman (2000) présentent les résultats d'apprentissage automatique de cadres de sous-catégorisation à partir du corpus arboré du tchèque (Prague Dependency Treebank).

Pour le français, le Lefff est un lexique morphologique et syntaxique à large couverture (Sagot et al. 2006, Sagot 2010). L'architecture du Lefff est à deux niveaux : un lexique intensionnel et un lexique extensionnel. Dans le lexique intensionnel, chaque entrée correspond à un lexème, le lexique décrit pour chaque entrée lexicale son lemme, sa partie du discours, son cadre de sous-catégorisation et ses informations syntaxiques supplémentaires. Le lexique intensionnel est utilisé pour le développement de ressources lexicales. Le lexique extensionnel est compilé automatiquement à partir du lexique intensionnel, il associe chaque forme fléchie d'une entrée donnée à une structure détaillée qui représente ses informations morphologiques et ses comportements syntaxiques possibles. Le lexique extensionnel est directement utilisé par les outils de TAL tels que le parseur syntaxique. Par exemple, dans le lexique extensionnel du Lefff, l'entrée du verbe "manger" dans le Lefff est comme ci-dessous :

manger v [pred="manger_____1<Suj:(sn|cln),Obj:(sn|cla)>",@pers,cat=v,@W]

Nous pouvons bien distinguer la catégorie et la structure syntaxique, présentée entre crochets. La structure syntaxique comporte un « pred » à la LFG, compose d'un identifiant sémantique, souvent identique au lemme, et d'un cadre de sous-catégorisation, présentée entre <>. Ici, il s'agit d'un sujet nominal ou clitique obligatoire et d'un objet nominal ou clitique facultatif indiqués par les parenthèses. Enfin, la catégorie morphosyntaxique est indiquée par l'attribut cat, ainsi qu'une macro résumant l'étiquette morphologique (les macros sont introduites par « @ »).

Basé sur le Lefff, nous voulons reproduire nos propres entrées de cadres de sous-catégorisation, nous voulons que chaque entrée peuvent montrer la catégorie et les structures syntaxiques de l'élément lexical, et préciser ses arguments potentiels en indiquant la catégorie et la fonction syntaxique, de plus, nous voulons aussi calculer la fréquence de chaque cadre de sous-catégorisation et de chaque argument en fournissant pour chacun un exemple de phrase.

2.7. Projet NaijaSynCor

Les travaux de l'extraction des lexiques font partie du projet NaijaSynCor qui propose une étude exhaustive et approfondie de la structure du naija.

Le naija est un pidgin créole à base lexicale anglaise (Bakker, 2009), son origine se trouve dans le pidgin nigérian, un créole parlé au Nigeria. Cette langue est aujourd'hui parlée comme deuxième langue par plus de 100 millions de locuteurs, au Nigeria, un pays de 180 millions d'habitants, où environ 450 langues maternelles sont parlées avec trois langues dominantes (Igbo, Yoruba et Hausa). Le Naija, qui n'a pas de statut officiel ni d'orthographe standard, a pris une importance économique et culturelle au Nigeria, et les élites nigérianes l'utilisent pour la communication privée et informelle. Au fil du temps, le naija a développé de nouvelles structures, un nouveau vocabulaire, et probablement une nouvelle prosodie, qui le différencient du Pidgin nigérian. En tant qu'une langue avec sa propre grammaire et son lexique, le naija a un potentiel exceptionnel en faveur de la cohésion nationale et de l'intégration régionale puisqu'elle est ethniquement neutre. Malgré son importance croissante, le naija n'a guère retenu l'attention. C'est pourquoi le projet NaijaSynCor (NSC) a été lancé, une recherche sur le naija basée sur des corpus, financée par l'Agence nationale de la recherche ANR (Caron, 2017). L'objectif est d'examiner de manière exhaustive et approfondie la nature et les fonctions du naija aujourd'hui, afin d'établir le lien entre l'évaluation de structure et le changement d'utilisation et de fonction de la langue. Le corpus étudiera la parole naturelle afin d'évaluer la différence entre le naija et l'anglais nigérian par l'étude de l'intonation, de la structure de l'information, de la morphologie, de la micro-syntaxe et de la macro-syntaxe. Actuellement, le corpus du NSC compte 321 fichiers audio d'une durée moyenne de 5 minutes chacun et 319 locuteurs, ce qui représente un total de 500 000 mots collectés dans 11 lieux. Les contenus couvrent des histoires de vie, des discours, des programmes radio, des conversations, des recettes de cuisine, des commentaires sur l'actualité, etc.

Le corpus utilisé dans ce mémoire vient du projet NSC, les 88 fichiers de treebanks annotés dans le schéma SUD compte en total 9 235 phrases et 141 363 tokens. L'état actuel des treebanks est accessible sur le site https://github.com/surfacesyntacticud/SUD_Naija-NSC et sur le site http://match.grew.fr/?corpus=SUD_Naija%20NSC@dev. Voici quelque exemples de phrases qui présentent des caractéristiques spécifiques du naija, ces exemples viennent des Naija SUD Guidelines https://surfacesyntacticud.github.io/guidelines/pcm/.

En naija, "dey", "be" et "na" peuvent fonctionnent comme des copules et sont annotés comme un verbe. La copule est reliée au sujet par la relation syntaxique "subj" et au prédicat par la relation "comp:pred".

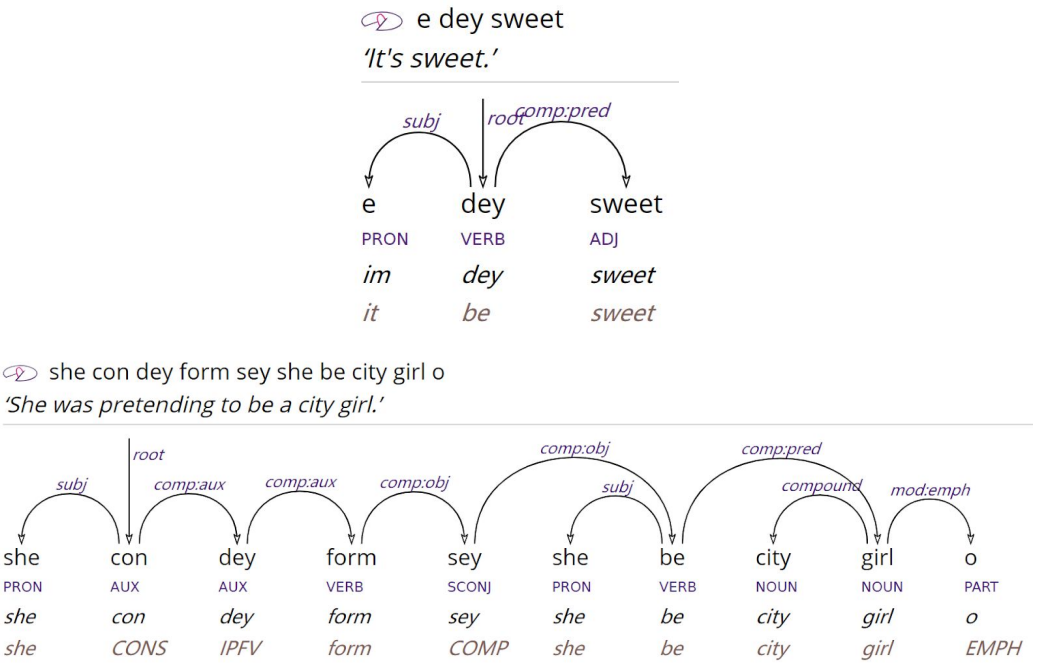

Figure 2.8 : Annotation de la copule en SUD - 1
https://surfacesyntacticud.github.io/guidelines/pcm/

Cependant, la copule n'est pas toujours nécessaire pour relier les sujets à leurs prédicats. Dans les cas où aucune copule n'est présente, le prédicat est relié à son sujet par la relation "subj".

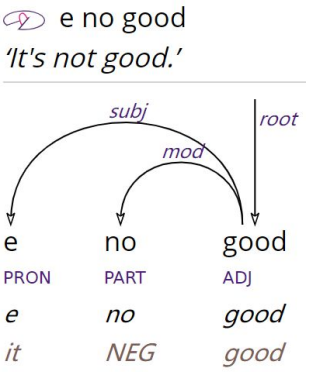

Figure 2.9 : Annotation de la copule en SUD - 2
https://surfacesyntacticud.github.io/guidelines/pcm/

Le schéma d'annotation de NaijaSUD utilise fréquemment la relation "compound". Cette relation présente systématiquement la relation entre deux noms dans lesquelles l'un d'eux fonctionne comme un modifieur. En ce sens, la fonction "compound" est similaire à la relation "mod", sauf qu'elle relie deux noms entre eux plutôt qu'un nom à un adjectif.

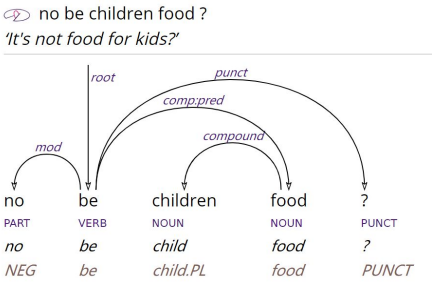

Figure 2.10 : Annotation de la relation "compound" en SUD - 1
https://surfacesyntacticud.github.io/guidelines/pcm/

La relation "compound" est également utilisée dans certaines relations entre noms et adjectifs qui sont considérés comme des expressions figées dont le sens ne peut être comprise directement à partir des constituants, comme "dry cleaner" dans la figure 2.11.

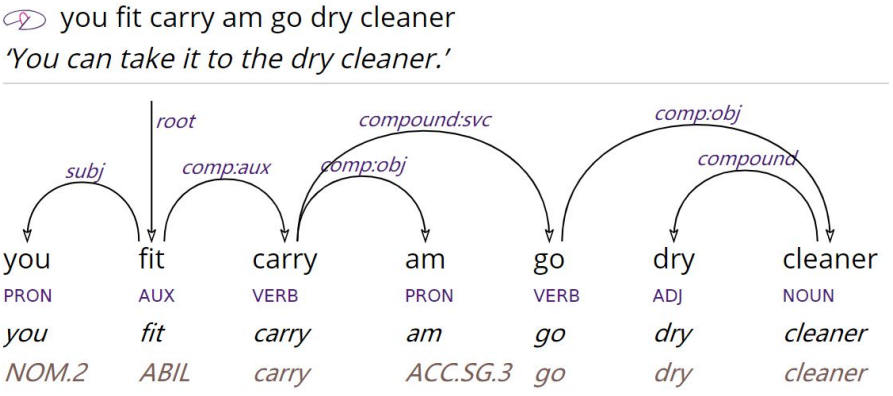

Figure 2.11 : Annotation de la relation "compound" en SUD - 2
https://surfacesyntacticud.github.io/guidelines/pcm/

2.8. Le format CONLL-U

Les treebanks sont disponibles au format CONLL-U. Le format CONLL-U est un format tabulaire dans lequel les différents types d'annotations d'un token sont écrits dans une ligne et sont séparés par 10 colonnes. Une phrase comporte les lignes d'annotation du token et les lignes de commentaires commençant par un dièse (#). Et chaque phrase est séparée par une ligne vide.

```
# sent_id = ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__1
# sound_url = http://www.tal.univ-paris3.fr/trameur/lTrameur-naija/mp3/ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG.mp3
# speaker_id = Sp3
# text = good morning > my people //
# text_en = Good morning, guys.
# text_ortho = Good morning, my people.
1  good    good    ADJ    _  _           2  mod       _  AlignBegin=2092|AlignEnd=2251|Gloss=good
2  morning morning NOUN   _  _           0  root      _  AlignBegin=2251|AlignEnd=2553|Gloss=morning
3  >       >       PUNCT  _  _           5  punct     _  AlignBegin=2553|AlignEnd=2583|Gloss=PUNCT
4  my      my      PRON   _  Number=S     5  mod:poss  _  AlignBegin=2583|AlignEnd=2751|Gloss=SG.1.POSS
5  people  people  NOUN   _  Number=F     2  vocative  _  AlignBegin=2751|AlignEnd=3148|Gloss=people.PL
6  //      //      PUNCT  _  _           2  punct     _  AlignBegin=3148|AlignEnd=3178|Gloss=PUNCT
```

Figure 2.12 : Exemple du format CONLL-U du corpus SUD_Naija-NSC

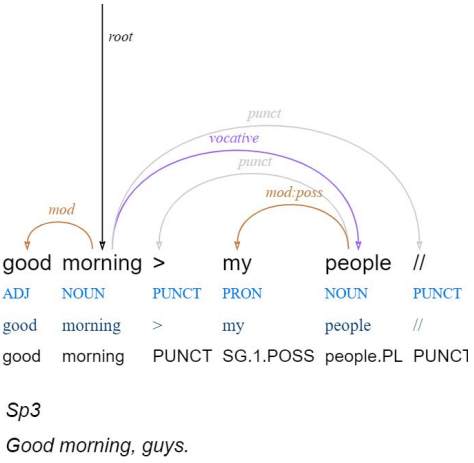

Figure 2.13 : Arbre de dépendance correspondant du corpus SUD_Naija-NSC

Pour chaque phrase, les commentaires sont utilisés pour stocker six méta-traits qui portent sur toute la phrase :

- sent_id : le nom du corpus et le numéro de la phrase
- sound_url : la transcription phonétique
- speaker_id : la référence du locuteur
- text : la phrase en naija
- text_en : la traduction anglaise de la phrase
- text_ortho : la phrase orthographique en naija

L'annotation de chaque token de la phrase est composée de 10 colonnes qui apparaissent dans une ligne :

1. Identifiant du token : nombre entier commençant à 1 pour chaque nouvelle phrase
2. Forme : forme du token ou la ponctuation.
3. Lemme
4. UPOS : partie du discours universelle
5. XPOS : partie du discours spécifique ; underscore si non disponible.
6. FEATS : chaque trait est annoté en suivant la convention trait = valeur, et les traits sont séparés par le symbole |, underscore si non disponible.
7. HEAD : identifiant du gouverneur, qui est soit la valeur d'identifiant du token, soit zéro.
8. DEPREL : relation syntaxique par rapport au gouverneur
9. DEPS : relations appartenant à la représentation UD "augmentée" (qui fournit des informations sur le graphe de syntaxe profonde)
10. MISC : informations supplémentaires que les personnes ayant constitué le corpus souhaitent ajouter : gloses, alignements temporels, commentaires, etc.

2.9. Résumé

Dans ce chapitre, nous avons mieux compris comment la théorie de la grammaire universelle a influencé les recherches syntaxiques au fil des années. Dans le but de faciliter les recherches syntaxiques multilingues, le schéma d'annotation d'UD cross-linguistiquement applicable a été créé sur la base de Stanford Dependencies. De plus, pour surmonter les désavantages des étiquettes et de la priorité des mots lexicaux dans l'UD, le schéma d'annotation SUD est créé comme alternative à l'UD. Nous avons aussi pris connaissance des notions de sous-catégorisations, du projet NaijaSynCor et du format d'enregistrement du corpus - le format CONLL-U. Ce sont toutes les connaissances nécessaires pour dérouler les travaux d'extraction des lexiques. Dans le chapitre suivant, nous présenterons le corpus utilisé et les méthodes d'extraction des deux lexiques, nous proposerons aussi des analyses sur les entrées des lexiques et les travaux à faire dans l'avenir.

Chapitre 3

Corpus, méthodes et analyses des résultats

Dans ce chapitre, nous présenterons les processus d'extraction des lexiques. L'extraction des lexiques est un travail syntaxique qui vise à préparer les ressources linguistiques pour le projet NaijaSynCor. Le but principal est de produire deux lexiques (un lexique morphosyntaxique dont chaque entrée est associée à ses traits morphologiques et un lexique syntaxique dont chaque entrée est associée à ses cadres de sous-catégorisation) et de fournir un corpus propre qui pourrait être utilisé pour les différentes tâches de TAL dans le cadre du projet NaijaSynCor, dont l'entraînement du parseur, la conversion SUD-UD, etc. Dans les sections suivantes, nous présentons brièvement le corpus utilisé dans ce mémoire, et nous montrons respectivement en détail les processus d'extraction des deux lexiques, y compris les méthodes d'extraction, les résultats, et les travaux futurs.

3.1. Corpus

L'extraction des deux lexiques se fait à partir du corpus du projet NSC, y compris les 88 fichiers de treebanks au format CONLL-U annotés dans le schéma SUD. Les contenus couvrent des histoires de vie, des discours, des programmes radio, des conversations, des recettes de cuisine, des commentaires sur l'actualité, etc. Le corpus compte en total 9 235 phrases et 141 363 tokens. L'état actuel des treebanks est accessible sur le site https://github.com/surfacesyntacticud/SUD_Naija-NSC et sur le site http://match.grew.fr/?corpus=SUD_Naija%20NSC@dev.

3.2. Lexique morphosyntaxique

3.2.1. Introduction du lexique morphosyntaxique

Le lexique morphosyntaxique comporte 6 colonnes, les cinq premières colonnes contiennent les informations sur la forme, la partie du discours, les traits morphologiques, le lemme et la glose du token. La sixième est la colonne de commentaire où nous pouvons partager nos diverses opinions pour obtenir le meilleur résultat.

FORME	POS	TRAITS MORPH	LEMME	GLOSE	COMMENTAIRE
accommodate	VERB	_	accommodate	accommodate	
accommodating	VERB	Tense=Pres|VerbForm=Part	accommodate	accommodate.PRS.PTCP	
...	...	...	...	...	
Zuba	PROPN	_	Zuba	Zuba	
zugi	NOUN	_	zugi	intervention	

Figure 3.1 : Tableau du lexique morphosyntaxique

L'exploitation du lexique morphosyntaxique permet de mieux connaître le vocabulaire et l'usage des mots de la langue. En analysant statistiquement des données du lexique, nous pouvons savoir plus précisément quels sont les mots importants pour un corpus d'un contenu particulier, quels sont des usages fréquents du mot, et quels sont des cas vraiment rares dans cette langue. En plus, les lexiques peuvent aider à la fouille des erreurs d'annotation. En projetant des étiquettes d'annotation d'un token du lexique à celles des treebanks, nous pouvons détecter des erreurs d'annotations et remplir des informations manquantes afin d'avoir des corpus assez propres.

3.2.2. Méthode d'extraction du lexique morphosyntaxique

Comme présenté dans la section 2.8, les annotations du treebank sont enregistrées au format conll, qui est un format tabulaire dans lequel les informations relatives à un token apparaissent sur une ligne qui comporte 10 colonnes.

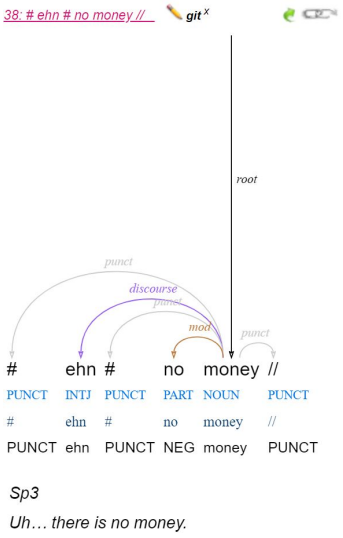

Figure 3.2 : Arbre de dépendance de la phrase du corpus de NSC

```
# sent_id = ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__38
# sound_url = http://www.tal.univ-paris3.fr/trameur/iTrameur-naija/mp3/ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG.mp3
# speaker_id = Sp3
# text = # ehn # no money //
# text_en = Uh··· there is no money.
# text_ortho = Ehn... no money.
1 #      #      PUNCT _ _                  5 punct    _ AlignBegin=95460|AlignEnd=95809|Gloss=PUNCT
2 ehn    ehn    INTJ  _ _                  5 discourse _ AlignBegin=95809|AlignEnd=95999|Gloss=ehn
3 #      #      PUNCT _ _                  5 punct    _ AlignBegin=96029|AlignEnd=96400|Gloss=PUNCT
4 no     no     PART  _ Polarity=Neg       5 mod      _ AlignBegin=96400|AlignEnd=96581|Gloss=NEG
5 money  money  NOUN  _ _                  0 root     _ AlignBegin=96581|AlignEnd=96790|Gloss=money
6 //     //     PUNCT _ _                  5 punct    _ AlignBegin=96790|AlignEnd=96820|Gloss=PUNCT
```

Figure 3.3 : Phrase annotée au schéma SUD enregistrée au format conll - 1

Voici les processus de l'extraction du lexique morphosyntaxique :

(1). Pour chaque phrase, identifier les lignes d'annotation (au gris), chaque ligne contient des annotations morphosyntaxiques d'un token.

```
# text = # ehn # no money //
# text_en = Uh··· there is no money.
# text_ortho = Ehn... no money.
1 #      #      PUNCT _ _                  5 punct    _ AlignBegin=95460|AlignEnd=95809|Gloss=PUNCT
2 ehn    ehn    INTJ  _ _                  5 discourse _ AlignBegin=95809|AlignEnd=95999|Gloss=ehn
3 #      #      PUNCT _ _                  5 punct    _ AlignBegin=96029|AlignEnd=96400|Gloss=PUNCT
4 no     no     PART  _ Polarity=Neg       5 mod      _ AlignBegin=96400|AlignEnd=96581|Gloss=NEG
5 money  money  NOUN  _ _                  0 root     _ AlignBegin=96581|AlignEnd=96790|Gloss=money
6 //     //     PUNCT _ _                  5 punct    _ AlignBegin=96790|AlignEnd=96820|Gloss=PUNCT
```

Figure 3.4 : Phrase annotée au schéma SUD enregistrée au format conll - 2

(2). Pour chaque token, localiser les colonnes (au bleu) qui contiennent respectivement les annotations morphosyntaxiques sur la forme, le lemme, la partie du discours, les traits morphologiques et la glose.

Figure 3.5 : Phrase annotée au schéma SUD enregistrée au format conll - 3

(3). Extraire ces données et écrire dans le lexique en l'ordre de forme - partie du discours - traits morphologiques - lemme - glose.

FORME	POS	TRAITS MORPH	LEMME	GLOSE	COMMENTAIRE
#	PUNCT	_	#	PUNCT	
ehn	INTJ	_	ehn	ehn	
#	PUNCT	_	#	PUNCT	
no	PART	Polarity=Neg	no	NEG	
money	NOUN	_	money	money	
//	PUNCT	_	//	PUNCT	

Figure 3.6 : Extraction des informations morphosyntaxiques de la phrase

Il faut faire attention à la dixième colonne du treebank au format CONLL-U, où il y a deux types d'informations : l'alignement temporelle et la glose. Ici, il faut séparer ces deux informations et n'extraire que la glose.

AlignBegin=95460|AlignEnd=95809|Gloss=PUNCT ➞ Gloss=PUNCT

(4). Combiner les différentes occurrences dans une seule entrée. Dans le tableau ci-dessus, les mêmes annotations sur "#" apparaît deux fois, donc nous ne gardons une seule entrée dans le lexique.

FORME	POS	TRAITS MORPH	LEMME	GLOSE	COMMENTAIRE
#	PUNCT	_	#	PUNCT	
ehn	INTJ	_	ehn	ehn	
no	PART	Polarity=Neg	no	NEG	
money	NOUN	_	money	money	
//	PUNCT	_	//	PUNCT	

Figure 3.6 : Extraction des informations morphosyntaxiques de la phrase - 2

(5). Vérifier le lexique morphosyntaxique extrait, corriger les fausses annotations et ajouter les informations manquantes. Puisque le corpus est annoté par un parseur ou par plusieurs linguistes, un parseur ne cesse d'être amélioré et des linguistes ont parfois des opinions divergentes, donc il existe toujours des erreurs d'annotation ou manque d'annotations. De ce fait, après l'extraction du lexique, il est important de le vérifier manuellement pour rendre toutes les informations correctes et complètes.

Une fois que nous obtenons le "golden" lexique morphosyntaxique, nous pouvons l'utiliser pour analyser le vocabulaire, corriger le corpus annoté, etc.

3.2.3. Analyse des résultats d'extraction du lexique morphosyntaxique

(1). Vocabulaire

Le lexique morphosyntaxique dispose de 5 259 formes et 4 116 lemmes. La taille du lexique est 162 KB.

(2). Distribution des parties du discours du lexique

La partie du discours annotée est basée sur la liste de "Universal POS tags" de UD (https://universaldependencies.org/u/pos/), qui est une liste fixe contenant 17 catégories grammaticales, les divisant en 3 groupes : classe lexicale ouverte (mot lexical), classe lexicale fermée (mot grammatical), et autres. Il est possible que certaines étiquettes ne soient pas utilisées dans certaines langues. Cependant, la liste ne peut pas être étendue pour couvrir des extensions spécifiques à une langue. Selon le lexique morphosyntaxique, nous constatons qu'il y a en total 16 catégories dans le corpus, l'étiquette "SYM" (symbol) de Universal POS tags n'est pas utilisée.

	Universal POS tags	**Sens**	**Fréquence**
Classe lexicale ouverte	NOUN	Nom	2144
	VERB	Verbe	1016
	PROPN	Nom propre	664
	ADJ	Adjectif	492
	X	Inconnu	253
	ADV	Adverbe	172
	INTJ	Interjection	108

Classe lexicale fermée	PRON	Pronom	107
	ADP	Préposition	70
	AUX	Auxiliaire	53
	PUNCT	Punctuation	43
	NUM	Adjectif numéral	37
	SCONJ	Conjonction de subordination	34
	DET	Déterminatif	27
Autres	PART	Particule	26
	CCONJ	Conjonction de coordination	13
En total	16		5259

Figure 3.7 : Tableau des étiquettes de POS en naija

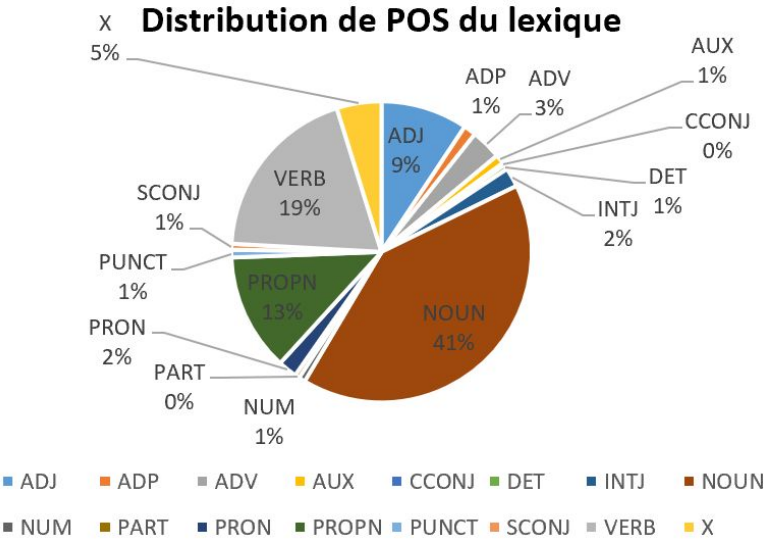

Figure 3.8 : Distribution des étiquettes de POS pour les lexèmes en naija

(3). Traits morphologiques (Features)

Les traits morphologiques sont des informations supplémentaires correspondant au mot. Chaque trait est sous forme de "Attribut = Valeur" et chaque mot peut avoir zéro, un ou plusieurs traits, et les traits sont séparés par le symbole "|", par exemple "Case=Acc|Person=2|PronType=Prs". Selon le lexique morphosyntaxique extrait à partir du corpus du projet NSC, il y a en total X attributs avec 44 valeurs. "PartType" est le seul attribut différent de Universal features UD (https://universaldependencies.org/u/feat/index.html), il a deux valeurs différentes possibles, "Cop" et "Disc".

NaijaSUD Features		
Attribute	**Value**	**Meaning**
Aspect	Cons	Consecutve
	Imp	Imperfective
	Perf	Perfect
	Prosp	Prospective
Case	Acc	Accusative
	Gen	Genitive
	Nom	Nominative
Definite	Def	Definite
	Ind	Indefinite
	Spec	Specific
Degree	Cmp	Comparative
	Sup	Superlative
Gender	Fem	Feminine
	Masc	Masculine
	Neut	Neuter
Mood	Cond	Conditional
	Ind	Indicative
	Nec	Necessitative
	Opt	Optative
	Pot	Potential

NumType	Card	Cardinal
	Ord	Ordinal
Number	Plur	Plural
	Sing	Singular
PartType	Cop	Copula
	Disc	Discursive
Person	1	1st (person)
	2	2nd (person)
	3	3rd (person)
Polarity	Neg	Negative
Poss	Yes	Possessive
PronType	Art	Article
	Dem	Demonstrative
	Int	Question (Pronoun)
	Prs	Personal (Pronoun)
	Rel	Relative (Pronoun)
Reflex	Yes	Reflexive (Pronoun)
Tense	Past	Past
	Pres	Present
VerbForm	Fin	Finite
	Inf	Infinitive
	Part	Participle
VerbType	Cop	Copula
Voice	Pass	Passive

Figure 3.9 : Tableau des traits morphologiques en naija

(4). Analyse quantitative

Le naija est un pidgin créole à base lexicale anglaise avec sa propre grammaire et son lexique, nous voulons montrer les différences entre le naija et anglais en comparant les informations morphosyntaxiques de façon quantitative. À cette fin, deux corpus anglais sont impliqués : Oxford English Corpus (OEC) (2011) et UD_English-GUM (Zeldes et Amir, 2017).

Nous avons calculé la fréquence du token de l'ensemble de corpus de NSC selon laquelle nous pouvons déterminer les mots de base et mesurer le niveau d'utilisation du mot. Le tableau ci-dessous montre les 10 mots les plus fréquents du corpus (les tokens avec l'étiquette "PUNCT" et "X" sont exclus) :

	NaijaSUD			**OEC**	
RANK	**TOKEN**	**POS**	**FREQUENCE**	**TOKEN**	**POS**
1	I	PRON	3168	the	Article
2	dey	AUX	3157	be	Verb
3	di	DET	2768	to	Preposition
4	you	PRON	2593	of	Preposition
5	go	AUX	2217	and	Conjonction
6	na	PART	2024	a	Article
7	no	PART	1816	in	Preposition
8	for	ADP	1760	that	Conjunction et al.
9	sey	SCONJ	1712	have	Verb
10	e	PRON	1562	I	Pronoun

Figure 3.10 : Tableau des mots fréquents en naija et en anglais

En comparant avec les 100 mots anglais les plus fréquents de l'OEC qui contient plus de 2 milliards de mots, il y a deux points qui nous intéressent, la fréquence faible du déterminant et la fréquence élevée de l'auxiliaire. Nous avons ensuite calculé la distribution des mots basée sur les parties du discours du corpus de NSC, et nous avons la comparée avec la distribution des parties du discours en anglais du UD_English-GUM. Les ponctuations, les symboles et les inconnus sont exclus.

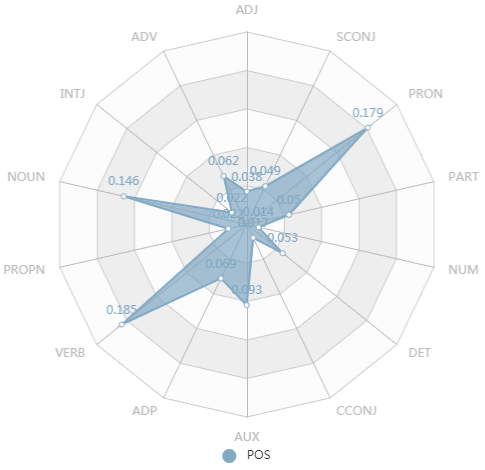

Figure 3.11 : Fréquence relative des POStag de NSC

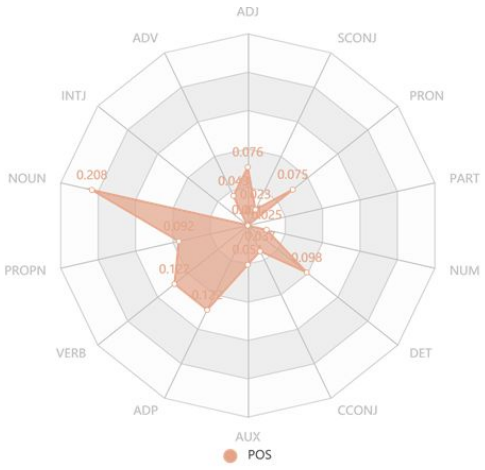

Figure 3.12 :Fréquence relative des POStag de UD_English-GUM

La fréquence des déterminants est plus faible en naija (5,3%) qu'en anglais (9,8%). Une phrase naija telle que "daddy na reverend, mumsie na pastor" ne nécessiterait pas de déterminants précis devant le nom "reverend" et "pasto", alors que son équivalent anglais, "My daddy is a reverend, my mommy is a pastor", le ferait.

La fréquence des auxiliaires est deux fois plus fréquente en naija (9,3%) qu'en anglais (5,1%). Nous avons ensuite regardé précisément la distribution de ces auxiliaires, les auxiliaires qui apparaissent plus fréquents sont ceux qui ne sont pas partagés avec l'anglais, y compris "dey" (38%), "go" (24%), "come" (15%), don (8%), fit (4%), l'auxiliaire "will" apparaît 77 fois et l'auxiliaire "for" apparaît 39 fois, 10% pour les autres mots qui sont annotés comme auxiliaire dont la plupart sont des mots issus directement de l'anglais, comme "have", "get", "be", etc.

Étant donné qu'un auxiliaire est un verbe qui se combine à un verbe principal pour constituer ainsi un temps composé ou périphrastique, nous avons calculé le ratio verbe sur auxiliaire. Le score de naija est plus bas que celui d'anglais, ce qui signifie que la nécessité du verbe de combiner avec un auxiliaire dans les phrases de naija est plus élevée que celle d'anglais.

	Ratio Verb / Auxiliaries
Naija	1.989
Anglais	2.392

Figure 3.13 : Ratio Verb / Auxiliaries en naija et en anglais

Nous avons également remarqué une grande proportion des verbes en naija (18%) alors qu'en anglais ce sont les noms qui occupent la plus grande proportion (21%), nous attribuons cela au fait qu'en naija il existe plein de constructions des verbes en série, c'est-à-dire deux ou plusieurs verbes ou expressions verbales sont enchaînées en une seule clause. Par exemple, dans la phrase

"ABJ_GWA_08_David-Lifestory_MG__2", les verbes "wan" et "carry", "go" et "train" sont enchaînés.

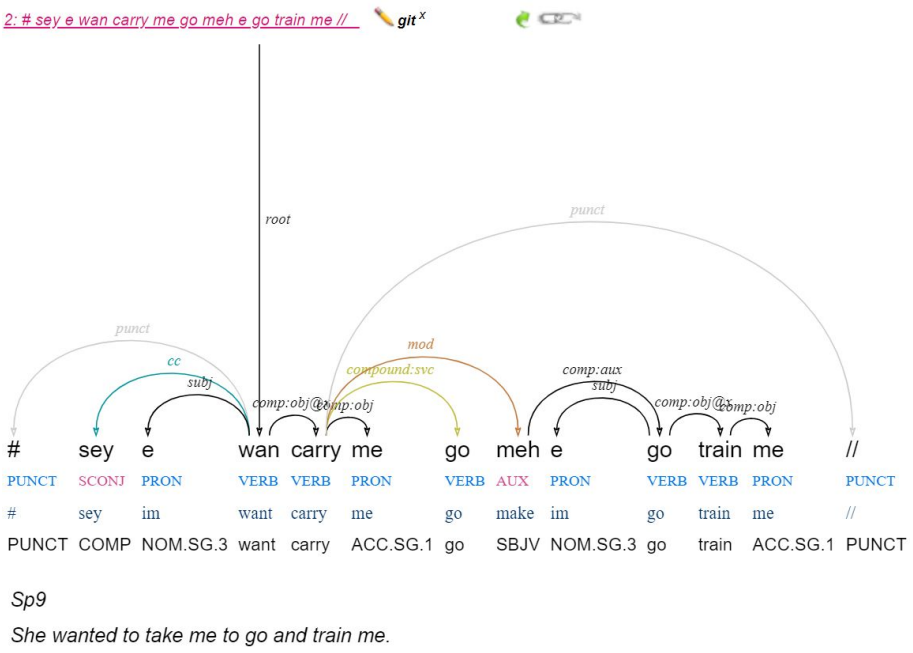

Figure 3.14 : Arbre de la phrase ABJ_GWA_08_David-Lifestory_MG__2

3.2.4. Applications du lexique morphosyntaxique

Dans cette section, nous présentons principalement comment rendre des treebanks propres à l'aide du lexique morphosyntaxique. Nous parlons d'abord le lien entre les treebanks et le lexique morphosyntaxique. Ensuite, nous montrons deux étapes principales qui sont nécessaires pour achever cet objectif : la fouille des erreurs et l'ajout des informations morphosyntaxiques manquantes. Finalement, nous analysons les erreurs détectées.

(1). Lien entre les treebanks et le lexique morphosyntaxique

Généralement, il existe des erreurs d'annotations dans les treebanks annotés par un parseur ou par des linguistes. Nous voulons que le lexique morphosyntaxique extrait peut aider non seulement à analyser le vocabulaire en naija, mais aussi à rendre les treebanks propres. Les étapes ci-dessous montre le workflow entre les treebanks et le lexique morphosyntaxique :

(i). Sélectionner et télécharger les treebanks au format CONLL-U.

- À ce moment-là, ces treebanks ont des erreurs d'annotations, en plus, les annotations sur le lemme et la glose sont souvent omis.

(ii). Extraire un lexique morphosyntaxique à partir de ces treebanks.

(iii). Vérifier le lexique et compléter les informations manquantes.

- Le lexique contient des fausses informations morphosyntaxiques du token, donc il faut que les linguistes le vérifient et corrigent manuellement. En plus, les informations sur le lemme et la glose ne sont pas complètes, surtout pour la glose, les linguistes ont complété les lemmes et créée des règles pour former automatiquement la glose. À la fin, nous obtenons le "golden" lexique morphosyntaxique prêt à utiliser.

(iv). Rendre tous les treebanks propres à l'aide du lexique morphosyntaxique.

- Fouiller des erreurs d'annotations sur les token, les parties du discours et les traits morphologiques.
- Corriger certaines erreurs qui peuvent être fixées automatiquement.
- Et puis, les linguistes corrigent manuellement le reste des erreurs potentielles détectées.
- Ensuite, ajouter les informations manquantes du lemme et de la glose dans les lignes d'annotation du token dans les treebanks.
- Les treebanks sont tous propres, le "golden" corpus est ainsi formé.

(v). Mettre en ligne le "golden" corpus au projet NSC.

(2). Fouille des erreurs d'annotation à l'aide du lexique morphosyntaxique

La fouille des erreurs d'annotation se concentre sur trois informations morphosyntaxiques : la forme, la partie du discours et les traits morphologiques. Puisque les treebanks sont au format CONLL-U, il faut donc localiser les lignes d'annotation de chaque phrase (au bleu) et les regarder ligne par ligne.

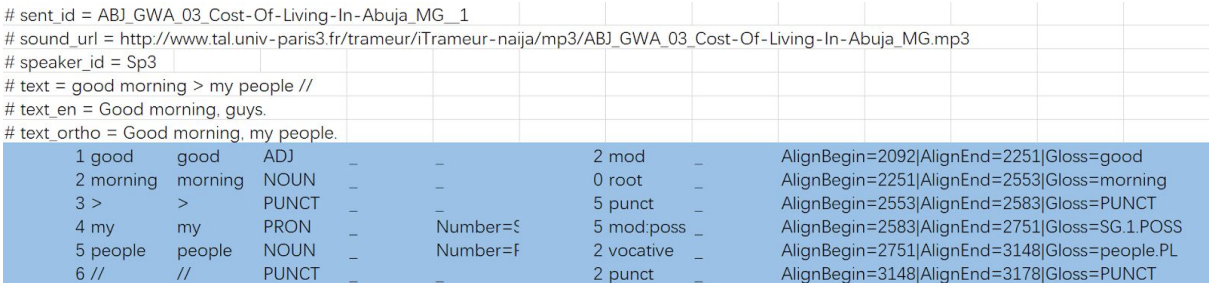

Figure 3.15 : Format CONLL-U de la phrase ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__1

Pour chaque ligne d'annotation du token, il faut vérifier les trois annotations morphosyntaxiques une par une dans l'ordre de forme, partie du discours et traits

morphologiques. Donc, nous devons localiser les annotations de la forme, de la partie du discours et des traits morphologiques qui se trouvent respectivement au 2e, 4e et 6e colonnes, et les projeter dans le lexique morphosyntaxique.

Une ligne d'annotation du treebank ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__1

	FORME	LEMME	POS		TRAITS MORPH				ALIGNEMENTS TEMPORELS + GLOSE
4	my	my	PRON	_	Number=Sing|Person=1|Poss=Yes	5	mod:poss	_	AlignBegin=2583|AlignEnd=2751|Gloss=SG.1.POSS

⬇

my PRON Number=Sing|Person=1|Poss=Yes

⬇

comparer la version des treebanks à celle du lexique

my PRON Number=Sing|Person=1|Poss=Yes

(version treebanks)

LEXIQUE

FORME	POS	TRAITS MORPH	LEMME	GLOSE
my	ADJ	Number=Sing|Person=1|Poss=Yes	my	SG.1.POSS
my	PRON	Number=Sing|Person=1|Poss=Yes	my	SG.1.POSS

(version lexique)

Figure 3.16 : Processus de la fouille des erreurs d'annotation

Si ces trois informations du token des treebanks sont cohérentes avec celles du lexique, elles seront considérées comme correctes, sinon, elles sont considérées comme les erreurs à traiter.

(i). Fouille des erreurs sur la forme

La vérification de la forme est la première étape de la fouille des erreurs d'annotation. Ici nous ignorons la case du mot.

TREEBANK		LEXIQUE
FORME		FORME
my	= ou ≠	my
		my

Figure 3.17 : Vérification de la forme du token

Si l'annotation de la forme localisée du treebank existe dans la colonne FORME du lexique morphosyntaxique, nous pensons que cette annotation est correcte, et nous ensuite vérifier la partie du discours et les traits morphologiques.

Si nous ne trouvons pas cette forme dans la colonne FORME du lexique morphosyntaxique, nous arrêtons les processus de vérification et marquons cette annotation comme une erreur orthographique. Dans ce cas-là, nous ne pouvons pas corriger automatiquement la forme, il faut donc enregistrer la position de l'erreur et les informations complètes de la phrase pour que les linguistes puissent corriger manuellement après.

(ii). Fouille des erreurs sur la partie du discours

La vérification de la partie du discours est basée sur la justesse de la forme parce que si la forme est incorrecte, il est impossible de localiser les informations de la partie du discours du lexique par la forme. Nous localisons toutes les parties du discours possible d'un token à l'aide de la forme.

TREEBANK			LEXIQUE	
FORME	POS		FORME	POS
my	ADJ	= ou ≠	my	ADJ
			my	PRON

Figure 3.18 : Vérification de la partie du discours du token

Si l'annotation de la partie du discours du token du treebank existe dans la colonne POS du lexique morphosyntaxique, nous pensons que cette annotation est correcte, et nous pouvons passons à l'étape finale qui est la vérification des traits morphologiques.

Sinon, nous pensons qu'il s'agit d'une erreur d'annotation. Il y a deux cas possibles :

- Si ce token a une seule partie du discours dans le lexique, c'est-à-dire ce token est unique dans le lexique, il correspond à une seule partie du discours et à une seule série des traits, donc l'algorithme va corriger automatiquement sa partie du discours et ses traits en localisant les informations correctes et uniques de ce token dans le lexique à l'aide de sa forme. Et la fouille des erreurs est finie pour cette ligne d'annotation.
- Si ce token a plusieurs parties du discours dans le lexique, il s'agit un cas ambigu que nous ne pouvons pas corriger automatiquement parce que l'algorithme ne peut pas juger laquelle est correcte. Il faut les linguistes vérifier manuellement.

L'important c'est que pour toutes les erreurs d'annotation sur la partie du discours, il est nécessaire de reconsidérer le rôle syntaxique que ce token joue dans la phrase, Donc il faut enregistrer cette erreur et les informations complètes de la phrase et fournir toutes les informations de ce token du lexique pour que les linguistes puissent ré-examiner si cet arbre de dépendance doit être ré-annoté manuellement.

Il est à noter que cette étape a deux défauts :

- Si un token a plusieurs parties du discours, par exemple, dans le lexique morphosyntaxique, "my" peut être un adjectif ou un pronom.

LEXIQUE				
FORME	POS	TRAITS MORPH	LEMME	GLOSE
my	ADJ	Number=Sing|Person=1|Poss=Yes	my	SG.1.POSS
my	PRON	Number=Sing|Person=1|Poss=Yes	my	SG.1.POSS

Figure 3.19 : Informations de "my" dans le lexique

Et si dans une phrase des treebanks, "my" fonctionne comme un pronom mais il est annoté comme un adjectif, notre algorithme est incapable de détecter ce genre d'erreur.

- La correction automatique de la partie du discours est effectuée lorsque nous considérons que le lexique contient toutes les parties du discours d'un token par défaut. Par exemple, si "my" sont tous annotés comme "ADJ" dans les treebanks et certains devraient fonctionner comme "PRON", "my" occupe une seule ligne dans le lexique extrait.

LEXIQUE				
FORME	POS	TRAITS MORPH	LEMMA	GLOSS
my	ADJ	Number=Sing|Person=1|Poss=Yes	my	SG.1.POSS

Figure 3.20 : Informations de l'adjectif "my" dans le lexique

Dans ce cas-là, même les linguistes peuvent enrichir la partie du discours "PRON" lors de la vérification du lexique, les pronoms "my" qui sont faussement comme "ADJ" ne peuvent pas être détectés.

Pour résoudre ces problèmes, il faut prendre en compte les relations syntaxiques entre le token et son gouverneur, cela nous ramènerait à construire un parseur basé sur des règles. En plus, plus il y a de données dans le lexique, moins il y a ce genre d'erreurs.

(iii). Fouille des erreurs sur les traits morphologiques

La vérification des traits morphologiques est basée sur la justesse de la forme et de la partie du discours. En naija, un mot peut avoir plusieurs parties du discours, mais chacune correspond à une seule série des traits morphologiques. Nous pouvons donc localiser les informations des traits du token du lexique à l'aide de la combinaison de forme et partie du discours.

TREEBANK				LEXIQUE		
FORME	POS	TRAITS MORPH		FORME	POS	TRAITS MORPH
my	ADJ	Number=Sing|Person=1|Poss=Yes	= ou ≠	my	ADJ	Number=Sing|Person=1|Poss=Yes

Figure 3.21 : Vérification de la forme du token

Si l'annotation des traits du treebank et les informations des traits du lexique sont toutes cohérentes, nous pensons que cette annotation est correcte. Dans ce cas-là, la fouille des erreurs sur la forme, la partie du discours et les traits morphologiques est finie, aucune erreur n'est détectée.

Sinon, il existe des erreurs d'annotations sur les traits. Grâce à la combinaison de forme et de partie du discours, il est possible de corriger automatiquement les fausses annotations des traits de ce mot en cherchant ses informations correctes des traits dans le lexique.

Il faut faire attention à l'ordre des traits qui sont organisés par ordre alphabétique dans le lexique, mais dans le treebank, l'ordre pourrait être différent, il faut donc réorganiser des traits par l'ordre alphabétique.

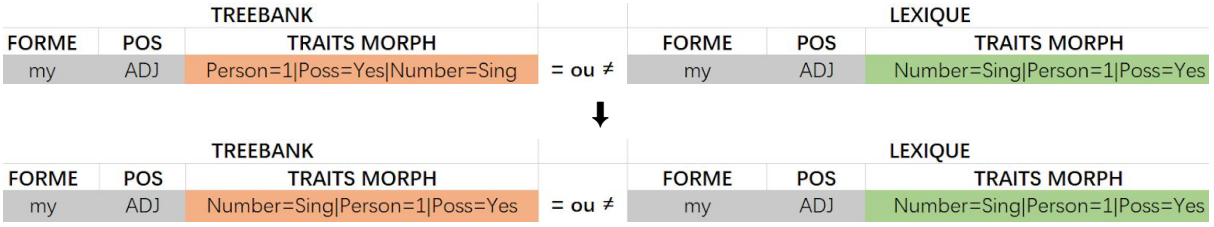

Figure 3.22 : La mise en l'ordre des traits morphologiques

Après la fouille des erreurs, la plupart des erreurs d'annotation seraient corrigées. Il reste encore quelques erreurs à regarder manuellement par des linguistes. Dès que les linguistes finissent la vérification, toutes les informations sur la forme, la partie du discours et les traits devraient être correctes.

(iv). Inventaire des erreurs détectées

Nous avons classifié et compté les erreurs détectées, voici les résultats.

(a). Erreurs sur la forme

Nous avons enregistré 81 erreurs sur la forme, et nous avons les classifié en deux types :

- Erreurs orthographiques : manque des lettres ; lettres supplémentaires ; manque de "-" (les mots dont la partie du discours est "X") ; espace ou symbole supplémentaire, etc.
- Erreurs de fusion : le mot et la ponctuation ont fusionné ensemble.

Erreurs orthographiques		
Forme incorrecte	**Forme correcte**	**Occurrence**
accomodation	accommodation	2
adaci	adashi	1
afihafereboh	afiafhereboh	1
assessement	assessment	1
becauase	because	1
bittter	bitter	1
caburator	carburetor	1

dats	dat's	3
dont	don't	3
extent	extend	1
gaskett	gasket	1
gentleman	gentlemen	1
harrass	harass	1
hygenic	hygienic	1
intergovermental	intergovernmental	1
knak	knack	1
liablity	liability	1
millilon	million	1
millon	million	1
mtchew	mstchew	6
mtshew	mstchew	2
oclock	o'clock	1
Portharcourt	Port-Harcourt	1
restructring	restructuring	1
spinnach	spinach	1
sulphiric	sulphuric	1
sulphric	sulphuric	1
their	there	2
Theophilllus	Theophilus	1
universties	universities	1
untill	until	1
whats	what's	1
?//]	?//]	1
?//+	?//	1
[[eng	[eng	1
[haus	[hau	1

ac-	AC	1
al	al-	1
co	co-	1
coun	coun-	1
ea	Ea-	1
inf	inf-	1
lafs	lafs-	1
ni-	ni	1
of-	of	1
sab	sab-	1
semi	semi-	1
Socio	Socio-	1
st	st-	2
TOTAL		**60**

Figure 3.23 : Tableau des erreurs orthographiques détectées

Erreurs de fusion		
Forme incorrecte	**Forme séparée**	**Occurrence**
//[eng	//	1
	[eng	
eng]//	//	15
	[eng	
eng//]	//	1
	eng]	
eng]]//	//	1
	[eng	
eng]|c	|c	1
	eng]	
eng]<	<	1

	eng]	
hau]<	<	1
	hau]	
TOTAL		**21**

Figure 3.24 : Tableau des erreurs de fusion détectées

(b). Erreurs sur la partie du discours

Erreurs sur la partie du discours	**Total**	**corrigées automatiquement**	**corrigées manuellement**
Nombre de tokens concernés	1 977	1 766	211
Nombre de types concernés	1 543	1 392	151
Occurrence	3 987	3 569	418

Figure 3.25 : Tableau des erreurs sur la partie du discours

Erreurs sur la partie du discours corrigées automatiquement				
	Forme	**POS non existante**	**POS correcte**	**Occurrence**
1	&	PUNCT	X	124
2	all	DET	ADV	97
3	take	AUX	VERB	67
4	state	PROPN	NOUN	43
5	to	PART	ADP	42
6	pepper	PROPN	NOUN	35
7	out	ADV	ADP	35
8	everybody	NOUN	PRON	33
9	oya	PART	INTJ	30
10	all	PRON	ADV	28
11	are	VERB	AUX	28
...	...	...	...	....
1766	allocate	NOUN	VERB	1

Figure 3.26 : Tableau des erreurs sur la partie du discours corrigées automatiquement (Erreurs_pos_unique.txt)

Erreurs sur la partie du discours corrigées manuellement				
	Forme	**POS non existante**	**POS possibles**	**Occurrence**
1	back	ADP	VERB / ADV / NOUN	34
2	own	NOUN	ADJ / VERB	16
3	naim	PART	ADV / SCONJ	16
4	my	DET	ADJ / PRON	12
5	own	PRON	ADJ / VERB	10
6	pidgin	NOUN	ADJ / PRON	10
7	sick	VERB	ADJ / NOUN	8
8	o	X	NOUN / PART	7
9	a	X	PRON / DET	6
10	nigerian	NOUN	ADJ / PROPN	6
...	...	...	...	...
207	book	PROPN	NOUN / VERB	1

Figure 3.27 : Tableau des erreurs sur la partie du discours corrigées manuellement (Erreurs_pos_ambigu.txt)

Voici quelques exemples avant et après la correction de la partie du discours :

1). sent_id = IBA_07_Na-Love_DG__165
text = di two of us < we do wedding //
text_en = Both of us had our wedding.

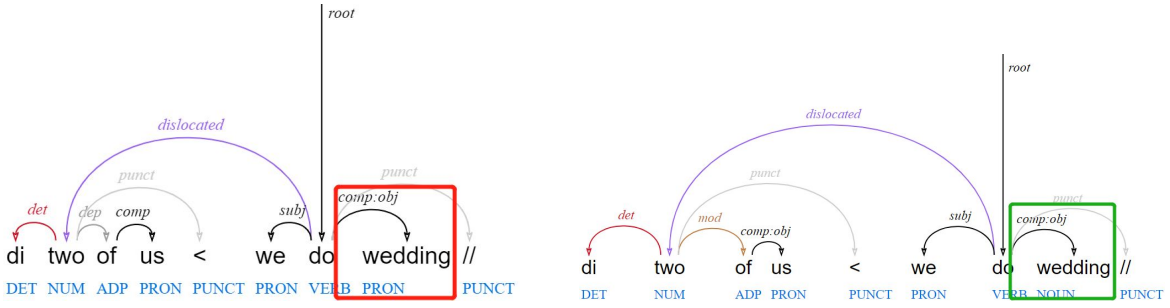

Figure 3.28 : Arbre de dépendance de la phrase "IBA_07_Na-Love_DG__165" avant et après la correction de la partie du discours de "wedding"

Ici, "wedding", qui est un nom, est faussement annoté comme un pronom. Puisque ce mot correspond à une seule partie du discours dans le lexique, la correction est automatiquement effectuée. Après la vérification manuelle par les linguistes, aucune relation syntaxique a été modifiée.

2). sent_id = ABJ_GWA_02_Market-Food-Church_DG__174
text = as in any upcoming programme ?//
text_en = As in any upcoming program?

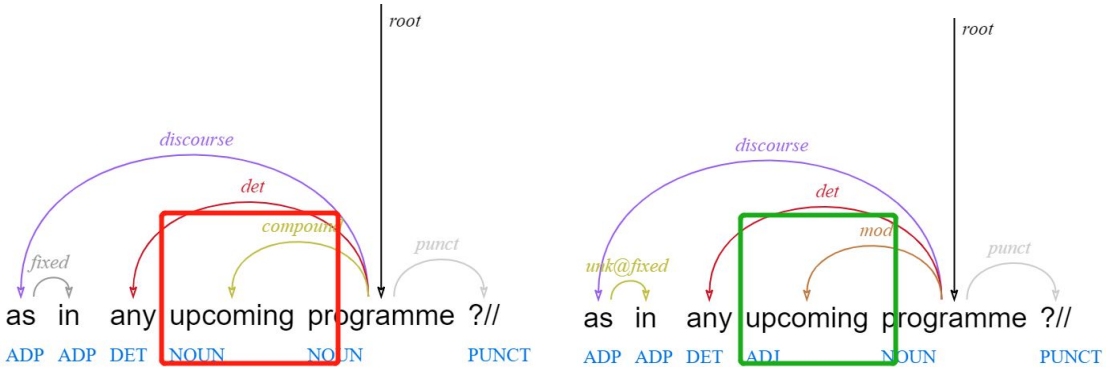

Figure 3.29 : Arbre de dépendance de la phrase "ABJ_GWA_02_Market-Food-Church_DG__174" avant et après la correction de la partie du discours de "upcoming"

Ici, "upcoming", qui est un adjectif, est faussement annoté comme un nom. Puisque ce mot correspond à une seule partie du discours dans le lexique, la correction est automatiquement effectuée. Au début, la relation syntaxique entre "upcoming" et son gouverneur "programme" est annotée comme "compound" parce que "compound" présente systématiquement la relation entre deux noms dans lesquelles "upcoming" fonctionne comme un modifieur. Puisque la partie du discours correct est ADJ, nous modifions cette relation en "mod".

3). sent_id = WAZL_15_MC-Abi_MG__136
text = naim I take spu- || naim I take speak am //
text_en = So I took spu-… that was how I recited it.

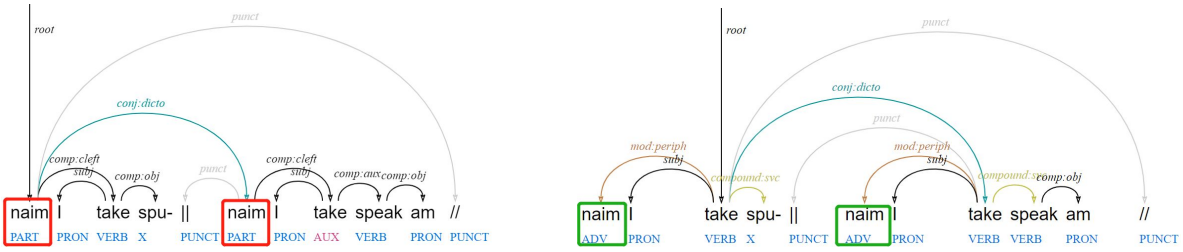

Figure 3.30 : Arbre de dépendance de la phrase "WAZL_15_MC-Abi_MG__136" avant et après la correction de la partie du discours de "naim"

Ici, "naim", qui est un adverbe, est faussement annoté comme un participe. En plus, ce mot correspond à plusieurs parties du discours dans le lexique (ADV/SCONJ), il faut que les linguistes vérifier manuellement. Dans cette phrase, "naim" est faussement considéré comme la tête de la phrase, il est en fait le modifieur du verbe "take" qui devrait être la tête de la phrase. Dans ce cas-là, la structure syntaxique de la phrase doit être reconstruire.

(c). Erreurs sur les traits morphologiques

Nous avons enregistré 15 855 lignes d'annotation dont les traits du token ne sont pas correctement annotés. Nous avons classifié ces erreurs en deux cas :

- Traits discordants : manque de traits ; traits superflus ; traits incohérents, etc.

FORME	POS	TRAITS DU TREEBANK INCORRECTS	TRAITS DU LEXIQUE CORRECTS	COMMENTAIRES
don't	AUX	Polarity=Neg|Tense=Pres	Mood=Ind|Polarity=Neg|Tense=Pres|VerbForm=Fin	manque de trait "Mood=Ind" et "VerbForm=Fin"
her	ADJ	PronType=Pers|Gender=Fem|Number=Sing|Poss=Yes|Person=3	Gender=Fem|Number=Sing|Person=3|Poss=Yes	traits superflus, supprimer "PronType=Pers"
hersef	PRON	Case=Acc|Gender=Masc|Number=Sing|Person=3|PronType=Prs|Reflex=Yes	Case=Acc|Gender=Fem|Number=Sing|Person=3|PronType=Prs|Reflex=Yes	traits incohérents, "her" est un pronom féminin

Figure 3.31 : Exemples des traits discordants

- Traits mal orthographiés :

Traits actuels	Correction à effectuer	Commentaires
Asp=Imp	Aspect=Imp	
Tense=Prosp	Aspect=Prosp	
Aspect=Real	Aspect=Cons	
Case=NOM	Case=Nom	
Def=Ind	Definite=Ind	
Degree=Comp	Degree=Cmp	
Degree=Poss	Degree=Pos	
Adj=Yes Aspect=Ind Definite=Sing Mood=Inf Number=Ind PersType=Prs	Ignorer	Corrigé manuellement
Aspect=Jus Mood=Jus	Mood=Sub	
Nuber=Plur Num=Plur Number=PLur Number=P_ Number=Pllur Number=Plural	Number=Plur	
Nuber=Sing Number=Spec	Number=Sing	
PronType=Card	NumType=Card	
Pers=1	Person=1	
Pers=3	Person=2	
Number=3	Person=3	
Prontype=Art	PronType=Art	
PronType=Prox	PronType=Dem	
PronTye=Prs PronTyp=Prs PronType=Pers	PronType=Prs	

PronounType=Prs Prontype=Prs		
PronType=Yes	Reflex=Yes	
Tense=Pst	Tense=Past	
Tense=Prs	Tense=Pres	
VerbFor=Fin Verbform=Fin verbForm=Fin	VerbForm=Fin	
VebForm=Part VerForm=Part VerbType=Part Verbform=Part VerbForm=Ger	VerbForm=Part	

Figure 3.32 : Tableau des traits mal orthographiés

(3). Ajout des informations morphosyntaxiques manquantes

Au début, les annotations sur le lemme et la glose ne sont pas complètes dans les treebanks, beaucoup de mots manque d'informations du lemme et il n'y a pas de glose. C'est pourquoi il faut d'abord compléter les informations des lemmes et des gloses dans le lexique morphosyntaxique extrait, et ensuite l'utiliser pour remplir les lignes d'annotation du treebank où il manque du lemme et de la glose du token.

Les lemmes sont remplis par les linguistes nigérians. Et avec leur aide, nous avons créé des règles qui permettent de produire automatiquement la glose du token basée sur leurs informations morphosyntaxiques.

En linguistique, une glose est un commentaire linguistique pour expliquer un mot. Pour nous, la traduction du mot (indiquée par le lemme et la partie du discours) et les informations morphologiques (indiquée par les traits morphologiques) sont les deux composantes importantes de la glose. Dans notre schéma d'annotation SUD, les traits morphologiques sont indépendants des autres informations morphosyntaxiques. Cela nous donne l'occasion de produire une glose mot à mot qui satisfera les typologues, via une glose de type Leipzig Glossing Rules (Bernard Comrie et Martin Haspelmath) produite automatiquement à partir des traits morphologiques.

Il y a deux étapes principales, la création de la glose de type SUD et la transformation de la glose de type SUD en celle de type Leipzig, par exemple :

writing	VERB	Tense=Pres|VerbForm=Part	write

⬇

(a). Créer la glose de type SUD

⬇

writing	VERB	Tense=Pres|VerbForm=Part	write	write.Tense=Pres.VerbForm=Part

⬇

(b). Transformer la glose de type SUD en celle de type Leipzig

⬇

writing	VERB	Tense=Pres|VerbForm=Part	write	write.PRS.PTCP

Figure 3.33 : Processus de la transformation de la glose

(a). Créer la glose de type SUD basée sur les informations morphosyntaxiques

(i). Mots grammaticaux dont la partie du discours est ADP, AUX, CONJ, SCONJ, DET, PART, PRON et ADJ possessif (Poss=Yes) :

- Si le mot a un ou plusieurs traits, la glose est représentée par la combinaison des traits. Par exemple :

FORME	POS	LEMME	TRAITS MORPH	CLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
con	AUX	con	Aspect=CONS	Aspect=CONS	Traits	CONS
the	DET	the	Definite=Def|PronType=Art	Definite=Def|PronType=Art	Traits	DEF.ART
no	PART	no	Polarity=Neg	Polarity=Neg	Traits	NEG
dat	PRON	dat	Number=Sing|PronType=Dem	Number=Sing|PronType=Dem	Traits	SG.DEM
dem	ADJ (Poss=Yes)	dem	Number=Plur|Person=3|Poss=Yes	Number=Plur|Person=3|Poss=Yes	Traits	PL.3.POSS

Figure 3.34 : Exemples de la création de la glose de type SUD - 1

- Si le trait est vide ("_"), la glose est le lemme. Par exemple :

FORME	POS	LEMME	TRAITS MORPH	CLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
about	ADP	about	_	about	Lemme	about
may	AUX	may	_	may	Lemme	may
both	CONJ	both	_	both	Lemme	both
because	SCONJ	because	_	because	Lemme	because

another	DET	another	_	another	Lemme	another
sebi	PART	sebi	_	sebi	Lemme	sebi
anybody	PRON	anybody	_	anybody	Lemme	anybody
intelligent	ADJ (Poss=Yes)	intelligent	_	intelligent	Lemme	intelligent

Figure 3.35 : Exemples de la création de la glose de type SUD - 2

(ii). Noms propres (PROPN):

- Si le mots a un ou plusieurs traits, la glose est représentée par la combinaison du lemme et des traits. Par exemple :

FORME	POS	LEMME	TRAITS MORPH	GLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
Americans	PROPN	Americans	Number=Plur	Americans.Number=Plur	Lemme.Traits	American.PL

Figure 3.36 : Exemples de la création de la glose de type SUD - 3

- Si le trait est vide ("_"), la glose est le lemme. Par exemple :

FORME	POS	LEMME	TRAITS MORPH	GLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
America	PROPN	America	_	America	Lemme	America

Figure 3.37 : Exemples de la création de la glose de type SUD - 4

(iii). Les mots dont la partie du discours inconnue, marqué "X" : la glose est aussi "X".

FORME	POS	LEMME	TRAITS MORPH	GLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
Adigo	X	Adigo	_	X	X	X

Figure 3.38 : Exemples de la création de la glose de type SUD - 5

(iv). Les ponctuations : la glose est notée comme "PUNCT".

FORME	POS	LEMME	TRAITS MORPH	GLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
eng]	PUNCT	eng]	_	PUNCT	PUNCT	PUNCT

Figure 3.39 : Exemples de la création de la glose de type SUD - 6

(v). Le reste des mots :

- Les mots anglais :
  - Si le mots a un ou plusieurs traits, la glose est représentée par la combinaison du lemme et des traits.
  - Si le trait est vide ("_"), la glose est le lemme.

FORME	POS	LEMME	TRAITS MORPH	GLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
eight	NUM	eight	NumType=Card	eight.NumType=Card	Lmme.Traits	eight.CARD
alive	ADJ	alive	_	alive	Lemme	alive

Figure 3.40 : Exemples de la création de la glose de type SUD - 7

- Les mots du naija (non-anglais) : créer manuellement la glose.

FORME	POS	LEMME	TRAITS MORPH	GLOSE	COMMENTAIRES SUR GLOSE	GLOSE LEIPZIG
pikin	NOUN	pikin	_	child	Glose différent de lemma	child

Figure 3.41 : Exemples de la création de la glose de type SUD - 7

(b). Transformation de la glose de type SUD en celle de type Leipzig

Nous utilisons les étiquettes de Leipzig Glossing Rules (Leipzig, 2015) au lieu des étiquettes des traits de SUD. Nous faisons un tableau qui sert à la transformation des traits.

NaijaSUD Feature	Leipzig feature	Meaning
Aspect=Imp	IPFV	Imperfective
Aspect=Perf	PRF	Perfect
Aspect=Prosp	PROSP	Prospective
Aspect=Cons	CONS	Consecutive
Case=Acc	ACC	Accusative
Case=Nom	NOM	Nominative
Definite=Def	DEF	Definite
Definite=Ind	INDF	Indefinite
Definite=Spec	SPEC	Specific
Degree=Cmp	CMPR	Comparative
Degree=Pos	POS	Positive
Degree=Sup	SUP	Superlative
Gender=Fem	F	Feminine
Gender=Masc	M	Masculine
Gender=Neut	NT	Neuter
Mood=Cond	COND	Conditional
Mood=Imp	IMP	Imperative
Mood=Ind	INDF	Indicative
Mood=Sub	SBJV	Subjunctive
NumType=Card	CARD	Cardinal
NumType=Ord	ORD	Ordinal
Number=Plur	PL	Plural
Number=Sing	SG	Singular
PartType=Cop	COP	Copula
PartType=Disc	DISC	Discursive
Person=1	1	1st (person)
Person=2	2	2nd (person)
Person=3	3	3rd (person)
Polarity=Neg	NEG	Negative
Polarity=Pos	POS	Positive
Poss=Yes	POSS	Possessive
PronType=Art	ART	Article
PronType=Dem	DEM	Demonstrative
PronType=Det	DET	Determinant
PronType=Int	Q	Question (Prononoun)
PronType=Prs	PS	Personal (Pronoun)
PronType=Rel	REL	Relative (Pronoun)
Reflex=Yes	REFL	Reflexive (Pronoun)
Tense=Past	PST	Past
Tense=Pres	PRS	Present
VerbForm=Fin	FIN	Finite
VerbForm=Inf	INF	Infinitive
VerbForm=Part	PTCP	Participle
VerbType=Cop	COP	Copula
Voice=Pass	PASS	Passive

Figure 3.42 : Tableau de la transformation des traits en Leipzig

Après que les lemme et les gloses sont complets dans le lexique, nous pouvons l'utiliser pour ajouter ces informations dans les treebanks.

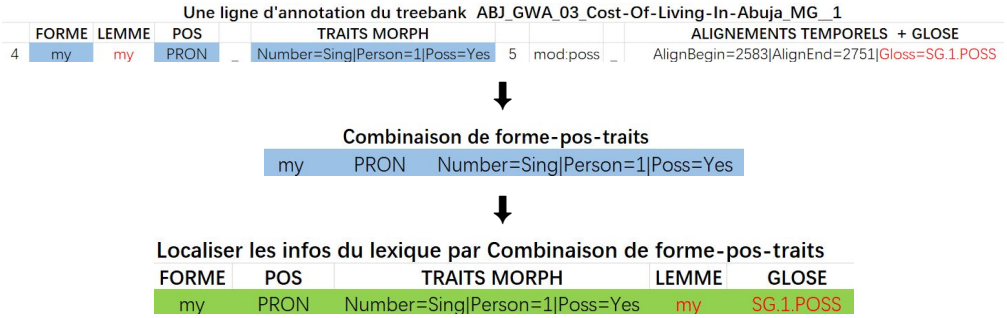

Figure 3.43 : Processus d'ajout des informations morphosyntaxiques manquantes

Puisque la combinaison de "forme - partie du discours - traits" est unique pour un token, lors de la vérification d'un token du corpus, nous pouvons utiliser cette combinaison pour localiser directement les informations correspondantes de ce token dans le lexique, dont son lemme et ses gloses corrects. Que les annotations soient erronées ou manquantes, nous pouvons les remplacer directement par les informations correctes de lexique au lieu de les vérifier. Voici une comparaison avant et après l'achèvement de l'information.

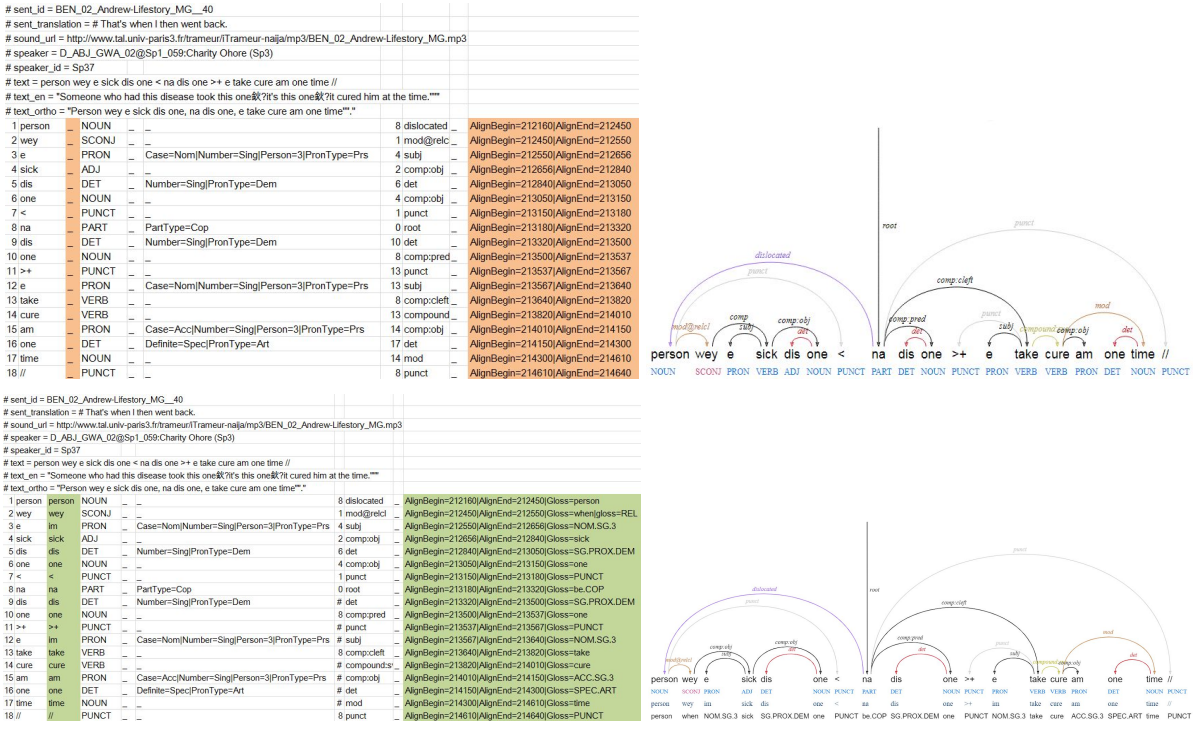

Figure 3.44 : Avant et après l'ajout du lemme et de la glose

À travers la fouille des erreurs sur la forme, la partie du discours et les traits morphologiques et l'ajout des lemmes et des gloses manquants, toutes les informations morphosyntaxiques du corpus devraient être correctes, nous obtenons enfin le "golden" corpus.

3.3. Lexique de cadres de sous-catégorisation

3.3.1. Introduction du lexique de cadres de sous-catégorisation

Le lexique de cadre de sous-catégorisations est un lexique syntaxique qui propose des entrées de cadres de sous-catégorisation du mot. Chaque entrée est un élément lexical qui indique les informations de la catégorie et des structures syntaxiques selon lesquelles nous pouvons savoir les arguments potentiels acceptés par un mot, la catégorie et la fonction syntaxique de ces arguments. De plus, nous voulons aussi calculer la fréquence de chaque cadre de sous-catégorisation et de chaque argument en fournissant pour chacun un exemple de phrase.

Basé sur les entrées du Lefff, nous voulons créer nos propres entrées de cadres de sous-catégorisation basée sur la fonction des arguments à partir du "golden" corpus de NSC. Chaque entrée concerne un lemme qui correspond à une partie du discours, nous utilisons le lemme au lieu de la forme parce que nous voulons neutraliser les différences dues à la flexion. Par exemple, le mot "clean", qui correspond à deux parties du discours "VERB" et "ADJ", peut se produire deux entrées.

```
clean                                          VERB                                                    4
-------------------------------------------------------
__frames :
<gov=comp:obj, comp:obj>                          2 # use to clean it |c # and di rest //
<gov=conj:coord, comp:obj,comp:obj>               1 # den we go come go { dust all di body |c clea
<gov=conj:coord, comp:obj>                        1 # oya { now |r now } de don go middle of ocear
-------------------------------------------------------
__arguments :
gov=comp:obj                                      2 [(ADP:2)]
gov=comp:obj                                      2 [(to:2)]
comp:obj                                          5 [(PRON:3) (ADJ:1) (NOUN:1)]
comp:obj                                          5 [(am:3) (dirty:1) (IT:1)]
gov=conj:coord                                    2 [(NOUN:1) (VERB:1)]
gov=conj:coord                                    2 [(body:1) (bring:1)]
-------------------------------------------------------

clean                                          ADJ                                                     5
-------------------------------------------------------
__frames :
<gov=conj:coord, comp:obj,comp:obj>               1 # go con { wash everywhere inside di house
<gov=comp:obj@x>                                  1 # put water > two of dem //= # be shaking i
<gov=comp:pred>                                   2 # we dey clean //
<gov=parataxis:conj, subj>                        1 no just take eye look tyre say [ ah di teet
-------------------------------------------------------
__arguments :
gov=conj:coord                                    1 [(VERB:1)]
gov=conj:coord                                    1 [(wash:1)]
comp:obj                                          2 [(NOUN:2)]
comp:obj                                          2 [(house:1) (place:1)]
gov=comp:obj@x                                    1 [(VERB:1)]
gov=comp:obj@x                                    1 [(shake:1)]
gov=comp:pred                                     2 [(AUX:1) (VERB:1)]
gov=comp:pred                                     2 [(be:1) (dey:1)]
gov=parataxis:conj                                1 [(VERB:1)]
gov=parataxis:conj                                1 [(sharp:1)]
subj                                              1 [(NOUN:1)]
subj                                              1 [(tyre:1)]
-------------------------------------------------------
```

Figure 3.45 : Exemple d'une entrée du lexique de cadres de sous-catégorisation

Chaque entrée est composée de trois parties : le mot lexical, les cadres et les arguments, et chaque partie est séparée par la ligne pointillée "------".

clean	VERB	4

Figure 3.46 : Première partie de l'entrée

La première partie indique le lemme et la partie du discours du mot, et le nombre de fois ce mot apparaît dans le corpus. Ici, le verbe "clean", qui apparaît 4 fois dans notre corpus, est une entrée du lexique de cadres de sous-catégorisation.

__frames :		
<gov=comp:obj, comp:obj>	2	# use to clean it |c # and di rest //
<gov=conj:coord, comp:obj,comp:obj>	1	# den we go come go { dust all di body |c clean am up ehn |c con carry am come back } ba
<gov=conj:coord, comp:obj>	1	# oya { now |r now } de don go middle of ocean |c # go { bring am out |c clean am } //

Figure 3.47 : Deuxième partie de l'entrée

La deuxième partie "__frames" regroupe tous les cadres de sous-catégorisation possibles. Chaque cadre de sous-catégorisation donne les informations sur les arguments potentiels du mot, y compris le gouverneur, le sujet, et les compléments. Nous calculons aussi le nombre d'occurrences de chaque cadre de sous-catégorisation dans l'ensemble de corpus, et nous collectons tous ses exemples concernant le même cadre de sous-catégorisation et montrons un exemple le plus court à la fin.

__arguments :		
gov=comp:obj	2	[(ADP:2)]
gov=comp:obj	2	[(to:2)]
comp:obj	5	[(PRON:3) (ADJ:1) (NOUN:1)]
comp:obj	5	[(am:3) (dirty:1) (IT:1)]
gov=conj:coord	2	[(NOUN:1) (VERB:1)]
gov=conj:coord	2	[(body:1) (bring:1)]

Figure 3.48 : Troisième partie de l'entrée

La troisième partie "__arguments" regroupe les relations syntaxiques des arguments potentiels en calculant pour chacune sa fréquence. Nous fournissons également les informations sur le POS et le lemme (valeurs entre parenthèses encadrées par "[ ]"), nous les mettons en l'ordre décroissant et ne gardons que 10 valeurs les plus fréquentes.

3.3.2. Méthode d'extraction du lexique de cadres de sous-catégorisation

Puisque l'entrée du lexique est un ensemble de cadres de sous-catégorisation avec les informations syntaxiques et la fréquence, il est donc nécessaire de traiter tous les tokens de toutes les phrases du corpus et regrouper les informations identiques pour le même token. L'extraction est constituée de deux étapes principales : l'extraction des arguments potentiels avec toutes ses informations syntaxiques et morphologiques pour chaque token du corpus, le regroupement des informations de sous-catégorisations identiques du même token en une seule entrée.

(1). Extraire les arguments potentiels avec toutes ses informations syntaxiques et morphologiques

L'idée c'est que les informations extraites sont détaillées et structurées, ce qui permettent d'intégrer plus facilement les informations identiques afin de former les entrées de cadres de sous-catégorisation. Vue que chaque entrée concerne un lemme qui correspond à une partie du discours, pour chaque token des treebanks, nous extrayons 6 types d'informations : son lemme, sa partie du discours, son gouverneur, son sujet, son(ses) complément(s), et le texte de la phrase comme un exemple. Le symbole "_" signifie les informations vides. Et nous utilisons 5 types d'étiquettes en majuscules qui aident à localiser les différentes informations. La tabulation est utilisée comme le séparateur.

LEMMA&POS:	lemme du token	pos du token	
GOV:	relation syntaxique entre le token et son gouverneur	pos du gouverneur	lemme du gouverneur
SUBJ:	relation syntaxique entre le token et son sujet	pos du sujet	lemme du sujet
COMP1:	relation syntaxique entre le token et son 1er complément	pos du complément	lemme du complément
...	...	...	...
COMPn:	relation syntaxique entre le token et son n-ième complément	pos du complément	lemme du complément
EXEMPLE	texte de la phrase		

Figure 3.49 : Exemple d'un ensemble d'informations des arguments potentiel du token

Regardons un exemple pour le verbe "do" :

```
# sent_id = ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121
# sound_url = http://www.tal.univ-paris3.fr/trameur/iTrameur-naija/mp3/ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG.mp3
# speaker_id = Sp3
# text = she do everyting for me //
# text_en = She did everything for me.
# text_ortho = She do everyting for me.
1 she she PRON _ Case=Nom|Gender=Fem|Number=Sing|Person=3|PronType=Prs 2 subj _ AlignBegin=347075|AlignEnd=347285|Gloss=NOM.F.SG.3
2 do do VERB _ _ 0 root _ AlignBegin=347285|AlignEnd=347652|Gloss=do
3 everyting everyting PRON _ _ 2 comp:obj _ AlignBegin=347652|AlignEnd=348355|Gloss=everything
4 for for ADP _ _ 2 comp:obl _ AlignBegin=348355|AlignEnd=348605|Gloss=for
5 me me PRON _ Case=Acc|Number=Sing|Person=1|PronType=Prs 4 comp:obj _ AlignBegin=348605|AlignEnd=348875|Gloss=ACC.SG.1
6 // // PUNCT _ _ 2 punct _ AlignBegin=348875|AlignEnd=348905|Gloss=PUNCT
```

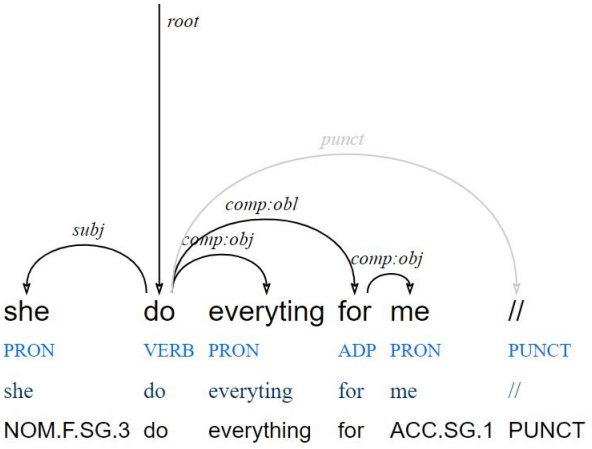

Figure 3.50 : Format CONLL-U et arbre de dépendance de la phrase ABJ_GWA_03_Cost-Of-Living-In-Abuja__MG__121

Le verbe "do" a un sujet et deux compléments, un complément d'objet direct et un complément d'objet indirect connecté par la préposition "for". En plus, "gov=root" montre qu'il est la racine de la phrase. Nous pouvons donc savoir l'usage de ce mot est "do sth for sb" et nous extrayons pour "do" les informations ci-dessous :

LEMMA&POS:	do	VERB	
GOV:	gov=root	_	_
SUBJ:	subj	PRON	she
COMP1:	comp:obj	PRON	everything
COMP2:	comp:obl	ADP	for <comp:obj PRON me>
EXEMPLE	she do everyting for me //		

Figure 3.51 : L'ensemble d'informations des arguments potentiels du verbe "do"

Maintenant nous expliquons en détail la méthode d'extraction pour chaque type d'arguments.

(a). Le gouverneur (GOV:)

Pour tous les tokens, nous voulons extraire son gouverneur, parce que le gouverneur fait partie du cadre de sous-catégorisation, surtout pour les modifieurs (ADJ, ADP, ADV, SCONJ). Si le token est la racine de la phrase, nous mettons seulement "gov=root".

(b). Le sujet (SUBJ:)

Il y a deux étiquettes qui indiquent la relation syntaxique de sujet dans notre corpus : "subj" et "subj@expl". Pour les verbes, les auxiliaires et les adjectifs qui se comportent comme un verbe dans la grammaire de Naija, nous voulons extraire le sujet s'il existe parmi les dépendants du token, par exemple, dans la phrase "ABJ_GWA_14_Mary-Lifestory_MG__82" :

```
# sent_id = ABJ_GWA_14_Mary-Lifestory_MG__82
# sound_url = http://www.tal.univ-paris3.fr/trameur/iTrameur-naija/mp3/ABJ_GWA_14_Mary-Lifestory_MG.mp3
# speaker_id = Sp15
# text = so e good as government do am //
# text_en = So that government procedure is good.
# text_ortho = So e good as government do am.
1 so so ADV _ _ 3 discourse _ AlignBegin=220820|AlignEnd=221270|Gloss=so
2 e im PRON _ Case=Nom|Number=Sing|Person=3|PronType=Prs 3 subj@expl _ AlignBegin=221270|AlignEnd=221590|Gloss=NOM.SG.3
3 good good ADJ _ _ 0 root _ AlignBegin=221590|AlignEnd=221820|Gloss=good
4 as as ADP _ _ 3 comp:obj@ag _ AlignBegin=221820|AlignEnd=222000|Gloss=as
5 governmer governmer NOUN _ _ 6 subj _ AlignBegin=222000|AlignEnd=222320|Gloss=government
6 do do VERB _ _ 4 comp:obj _ AlignBegin=222320|AlignEnd=222450|Gloss=do
7 am am PRON _ Case=Acc|Number=Sing|Person=3|PronType=Prs 6 comp:obj _ AlignBegin=222450|AlignEnd=222560|Gloss=ACC.SG.3
8 // // PUNCT _ _ 3 punct _ AlignBegin=222560|AlignEnd=222590|Gloss=PUNCT
```

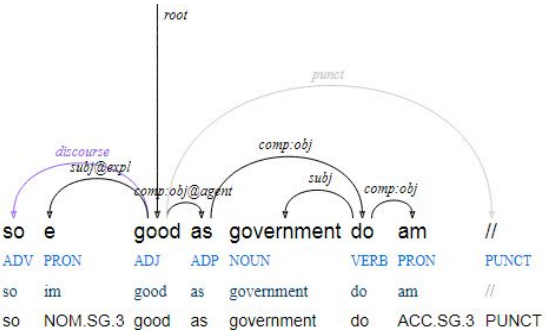

Figure 3.52 : Format CONLL-U et arbre de dépendance de la phrase ABJ_GWA_14_Mary-Lifestory__MG__121

Nous avons extrait pour le verbe "good" :

LEMMA&POS:	good	ADJ	
GOV:	gov=root	_	_
SUBJ:	subj	PRON	im
COMP1:	comp:obj@agent	ADP	<comp:obj VERB do>
EXEMPLE	so e good as government do am //		

Figure 3.53 : L'ensemble d'informations des arguments potentiels de l'adjectif "good"

- Ici good est un verbe statif. Nous avons décidé de l'annoter ADJ plutôt que VERB, mais il se comporte comme un verbe. Les adjectifs sont souvent prédicatifs, et c'est un point très important de la grammaire du naija.

De plus, s'il s'agit un sujet indirect, il faut remonter une ou plusieurs fois au gouverneur supérieur pour trouver le sujet, par exemple, dans la phrase " ABJ_GWA_14_Mary-Lifestory_MG__116" :

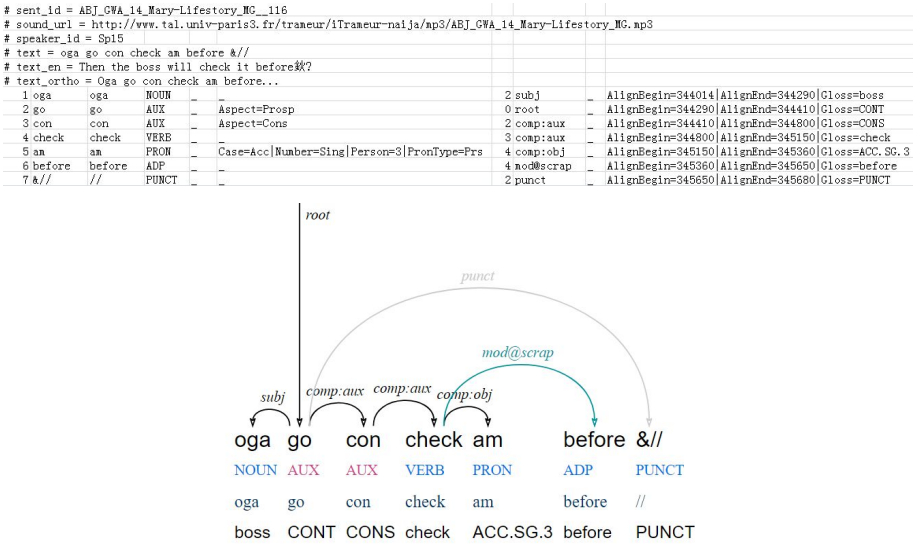

Figure 3.54 : Format CONLL-U et arbre de dépendance de la phrase ABJ_GWA_14_Mary-Lifestory_MG__116

Nous avons extrait pour le verbe "check" :

LEMMA&POS:	check	VERB	
GOV:	gov=comp:aux	AUX	con
SUBJ:	subj	NOUN	oga
COMP1:	comp:obj	PRON	am
EXEMPLE	oga go con check am before &//		

Figure 3.55 : L'ensemble d'informations des arguments potentiels du verbe "check"

- Le verbe "check" prend un complément d'objet direct, et il a un sujet indirect "oga", donc ici il faut remonter deux fois jusqu'au gouverneur de son gouverneur "con" pour trouver son sujet indirect qui est "oga".

(c). Les compléments (COMPn:)

Les compléments du token se trouvent parmi les dépendants dont la relation syntaxique est de type "comp", y compris sept étiquettes :

- comp : pour les compléments d'une préposition ou d'une conjonction
- comp:obj : complément d'objet direct
- comp:obl : complément oblique
- comp:caus : complément d'un causatif
- comp:aux : pour les relations entre un auxiliaire TAM (Tense-Aspect-Mood) et le verbe complet
- comp:pred : pour les relations entre deux prédicats qui partagent un argument.
- compound:svc : compound:svc est utilisé pour les constructions de verbes en série, qui sont typiques pour le Naija

S'il aucune relation syntaxique des dépendants du token est de type "comp", ce mot ne prend aucun complément, nous mettons alors "_".
En plus, si le dépendant direct du token dont la relation syntaxique est de type "comp" est une préposition (ADP) ou une conjonction de subordination (SCONJ), nous avons aussi besoin des informations du complément de ce dépendant, parce que ce complément, lié par une préposition ou une conjonction de subordination, est un complément oblique du token. Par exemple, dans la phrase "ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121" :

```
# sent_id = ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121
# sound_url = http://www.tal.univ-paris3.fr/trameur/iTrameur-naija/mp3/ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG.mp3
# speaker_id = Sp3
# text = she do everyting for me //
# text_en = She did everything for me.
# text_ortho = She do everyting for me.
1 she she PRON _ Case=Nom|Gender=Fem|Number=Sing|Person=3|PronType=Prs 2 subj _ AlignBegin=347075|AlignEnd=347285|Gloss=NOM.F.SG.3
2 do do VERB _ _ 0 root _ AlignBegin=347285|AlignEnd=347652|Gloss=do
3 everyting everyting PRON _ _ 2 comp:obj _ AlignBegin=347652|AlignEnd=348355|Gloss=everything
4 for for ADP _ _ 2 comp:obl _ AlignBegin=348355|AlignEnd=348605|Gloss=for
5 me me PRON _ Case=Acc|Number=Sing|Person=1|PronType=Prs 4 comp:obj _ AlignBegin=348605|AlignEnd=348875|Gloss=ACC.SG.1
6 // // PUNCT _ _ 2 punct _ AlignBegin=348875|AlignEnd=348905|Gloss=PUNCT
```

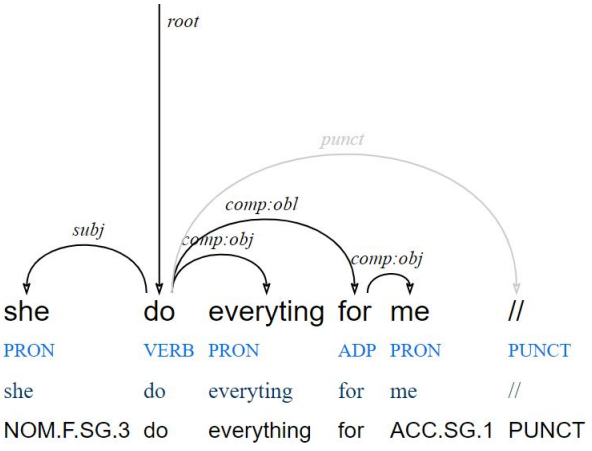

Figure 3.56 : Format CONLL-U et arbre de dépendance de la phrase ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121

Nous avons extrait pour le verbe "do" :

LEMMA&POS:	do	VERB	
GOV:	gov=root	_	_
SUBJ:	subj	PRON	she
COMP1:	comp:obj	PRON	everything
COMP2:	comp:obl	ADP	for <comp:obj PRON me>
EXEMPLE	she do everyting for me //		

Figure 3.57 : L'ensemble d'informations des arguments potentiels du verbe "do"

- Le verbe "do" a deux compléments, un complément d'objet direct et un complément d'objet indirect connecté par la préposition "for". L'usage de ce mot est "do sth for sb". Pour localiser les informations du complément d'objet indirect "me", il faut regarder le dépendant de la préposition "for″ dont la relation syntaxique est "comp:obj".

(d). Extraction complète d'une phrase (ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121)

```
# sent_id = ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121
# sound_url = http://www.tal.univ-paris3.fr/trameur/iTrameur-naija/mp3/ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG.mp3
# speaker_id = Sp3
# text = she do everyting for me //
# text_en = She did everything for me.
# text_ortho = She do everyting for me.
1  she       she       PRON  _  Case=Nom|Gender=Fem|Number=Sing|Person=3|PronType=Prs  2  subj       _  AlignBegin=347075|AlignEnd=347285|Gloss=NOM.F.SG.3
2  do        do        VERB  _  _                                                      0  root       _  AlignBegin=347285|AlignEnd=347652|Gloss=do
3  everyting everyting PRON  _  _                                                      2  comp:obj   _  AlignBegin=347652|AlignEnd=348355|Gloss=everything
4  for       for       ADP   _  _                                                      2  comp:obl   _  AlignBegin=348355|AlignEnd=348605|Gloss=for
5  me        me        PRON  _  Case=Acc|Number=Sing|Person=1|PronType=Prs            4  comp:obj   _  AlignBegin=348605|AlignEnd=348875|Gloss=ACC.SG.1
6  //        //        PUNCT _  _                                                      2  punct      _  AlignBegin=348875|AlignEnd=348905|Gloss=PUNCT
```

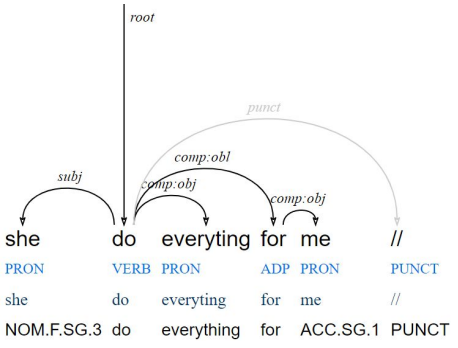

Figure 3.56 : Format CONLL-U et arbre de dépendance de la phrase ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121

Cette phrase est composée de six tokens, nous avons extrait six ensembles d'informations.

LEMMA&POS:	she	PRON	
GOV:	gov=subj	VERB	do
SUBJ:	_	_	_
COMP_NO:	_	_	_
EXEMPLE:	she do everyting for me //		

LEMMA&POS:	do	VERB	
GOV:	gov=root	_	_
SUBJ:	subj	PRON	she
COMP1:	comp:obj	PRON	everyting
COMP2:	comp:obl	ADP	for
EXEMPLE:	she do everyting for me //		

LEMMA&POS:	everyting	PRON	
GOV:	gov=comp:obj	VERB	do
SUBJ:	_	_	_
COMP_NO:	_	_	_
EXEMPLE:	she do everyting for me //		

LEMMA&POS:	for	ADP	
GOV:	gov=comp:obl	VERB	do
SUBJ:	_	_	_
COMP1:	comp:obj	PRON	me
EXEMPLE:	she do everyting for me //		

LEMMA&POS:	me	PRON	
GOV:	gov=comp:obj	ADP	for
SUBJ:	_	_	_
COMP_NO:	_	_	_
EXEMPLE:	she do everyting for me //		

LEMMA&POS:	//	PUNCT	
GOV:	gov=punct	VERB	do
SUBJ:	_	_	_
COMP_NO:	_	_	_
EXEMPLE:	she do everyting for me //		

Figure 3.58 : six ensembles d'informations extraits de la phrase ABJ_GWA_03_Cost-Of-Living-In-Abuja_MG__121

Dès que la première étape est finie, nous obtenons un fichier d'une taille de 23 MB qui contient des informations détaillées des arguments de tous les tokens du corpus. Ensuite, il faut regrouper les informations identiques du même token pour former des entrées des cadres de sous-catégorisation.

(2). Regrouper des informations de sous-catégorisations et produire des entrées des cadres de sous-catégorisation

Cette étape vise à identifier les ensembles d'informations extraits qui appartient au même token de la même catégorie, et regrouper des informations de sous-catégorisations identiques du même token en une seule entrée. Nous revenons à l'exemple de l'entrée du verbe "clean".

```
clean                                                VERB                                                4
-----------------------------------------------------
__frames :
<gov=comp:obj, comp:obj>                              2 # use to clean it |c # and di rest //
<gov=conj:coord, comp:obj,comp:obj>                   1 # den we go come go { dust all di body |c clea
<gov=conj:coord, comp:obj>                            1 # oya { now |r now } de don go middle of ocear
-----------------------------------------------------
__arguments :
gov=comp:obj                                          2 [(ADP:2)]
gov=comp:obj                                          2 [(to:2)]
comp:obj                                              5 [(PRON:3) (ADJ:1) (NOUN:1)]
comp:obj                                              5 [(am:3) (dirty:1) (IT:1)]
gov=conj:coord                                        2 [(NOUN:1) (VERB:1)]
gov=conj:coord                                        2 [(body:1) (bring:1)]
-----------------------------------------------------
```

Figure 3.59 : Exemple de l'entrée du verbe "clean"

Pour produire une telle entrée, nous devons premièrement localiser tous les ensembles d'informations dont le lemme est "clean" et le POS est "VERB". Nous en avons identifié quatre :

LEMMA&POS:	clean	VERB	
GOV:	gov=comp:obj	ADP	to
SUBJ:	_	_	_
COMP1:	comp:obj	ADJ	dirty
EXEMPLE:	# dat one < # e use to clean { di dirty wey dey inside # || di dust wey is inside } //		

LEMMA&POS:	clean	VERB	
GOV:	gov=comp:obj	ADP	to
SUBJ:	_	_	_
COMP1:	comp:obj	NOUN	IT
EXEMPLE:	# use to clean it |c # and di rest //		

LEMMA&POS:	clean	VERB	
GOV:	gov=conj:coord	NOUN	body
SUBJ:	_	_	_
COMP1:	comp:obj	PRON	am
COMP2:	comp:obj	PRON	am
EXEMPLE:	# den we go come go { dust all di body |c clean am up ehn |c con carry am come back } base on wetin { me |c and you } fit buy //		

LEMMA&POS:	clean	VERB	
GOV:	gov=conj:coord	VERB	bring
SUBJ:	_	_	_
COMP1:	comp:obj	PRON	am
EXEMPLE:	# oya { now |r now } de don go middle of ocean |c # go { bring am out |c clean am } //		

Figure 3.60 : 4 ensembles d'informations extraits du verbe "clean"

Il est facile de produire la première partie de l'entrée du verbe "clean" qui apparaît quatre fois dans le corpus.

```
clean                    VERB                    4
```

Figure 3.61 : Première partie de l'entrée du verbe "clean"

Quant à la deuxième partie "__frames", pour chaque ensemble d'information, nous utilisons les relations syntaxiques qui relient le token et ses arguments pour construire un cadre de sous-catégorisation, et puis nous calculons le nombre de fois qu'un même cadre apparaît. Nous devons également rassembler tous les exemples d'un même cadre et choisir un exemple court.

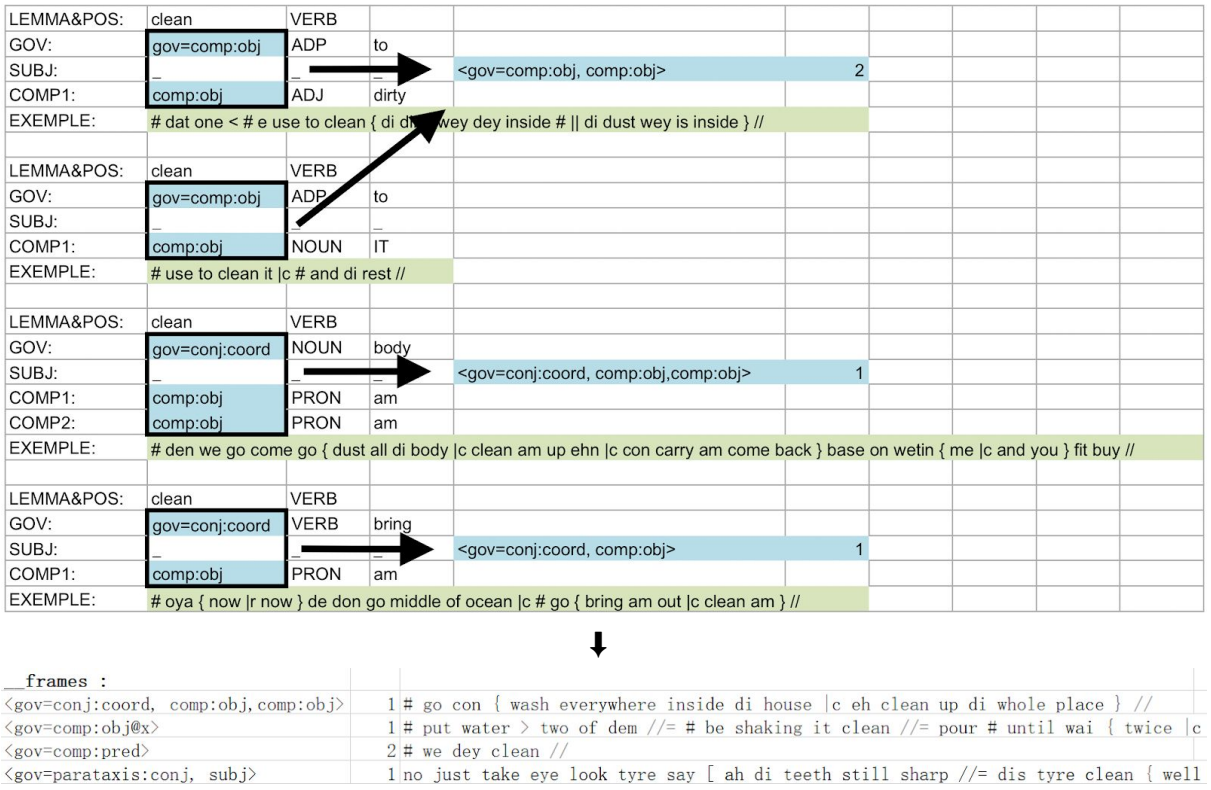

Figure 3.62 : Création de la deuxième partie de l'entrée du verbe "clean"

Pour la troisième partie "__arguments", nous avons besoin de lister toutes les relations syntaxiques des arguments potentiels du token en accompagnant les parties du discours et les lemmes correspondants, nous calculons la fréquence pour la même valeur. Pour la lisibilité, nous mettons la fréquence des parties du discours et des lemmes en l'ordre décroissant, si une relation syntaxique correspond à trop de lemmes ou de parties du discours, nous ne gardons 10 valeurs les plus fréquentes.

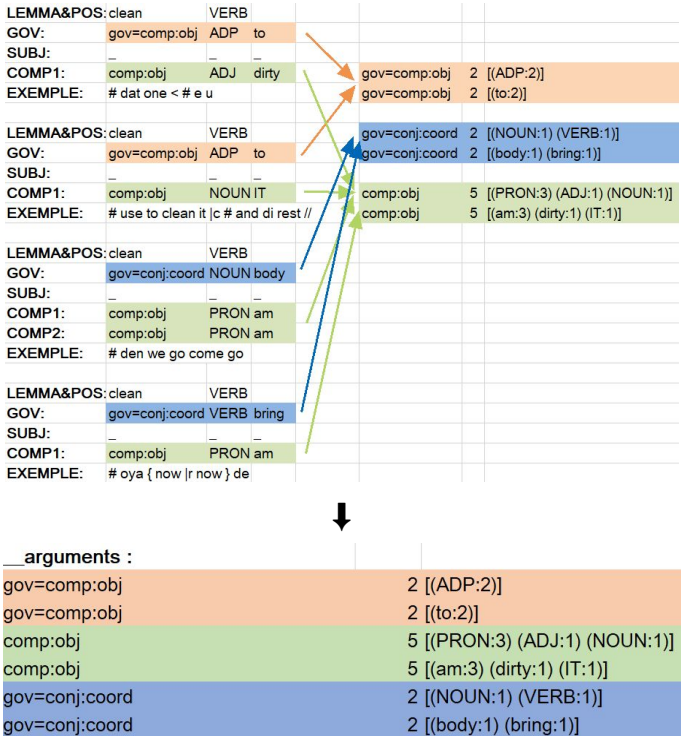

Figure 3.63 : Création de la troisième partie de l'entrée du verbe "clean"

Il est à noter que s'il s'agit du cas d'un complément d'objet indirect introduit par une préposition (ADP) ou une conjonction de subordination (SCONJ), il faut combiner les informations de cet argument indirect et de son gouverneur (ADP ou SCONJ) pour présenter la relation "comp:obl". Voici un exemple du verbe "communicate" :

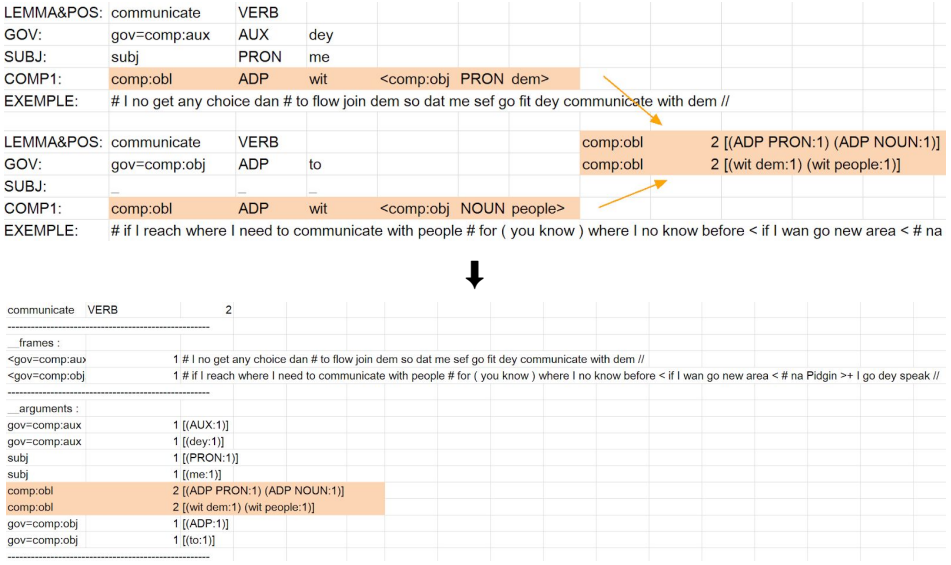

Figure 3.64 : Création de l'entrée du verbe "communicate"

3.3.3. Les travaux futurs du lexique de cadres de sous-catégorisation

Le lexique de cadres de sous-catégorisation dispose 4 360 entrées. La taille du lexique est 3.35 MB. Le nombre total de cadres est 14 522, avec 868 cadres différents, soit 3.33 cadres par lemme d'une catégorie.

Les résultats du lexique de cadres de sous-catégorisation que nous avons présentés dans ce mémoire ne représentent pas les résultats finaux. L'extraction de ce lexique n'est qu'un début, le lexique a encore besoin d'être vérifié et nettoyé par les linguistes pour obtenir un fichier "golden" qui est prêt à exploiter.

À l'avenir, en analysant statistiquement des cadres et des arguments potentiels de chaque mot, nous pourrons mieux savoir les usages grammaticaux du mot en naija, y compris les collocations, les locutions, les expressions figées, etc. Nous pourrons également classifier et mettre en ordre les mots et ses usages au niveau de la fréquence afin de décrire les mots courants et les mots rares de cette langue. Actuellement, il n'existe pas encore un dictionnaire de naija contenant un vocabulaire relativement complet, nous espérons utiliser ce lexique extrait pour construire un dictionnaire consultable de naija basée sur le format de Wiktionnaire. Ce dictionnaire devrait être capable de définir et de décrire les mots en naija, le mot peut être présenté par les éléments suivants dans l'ordre :

- La forme en naija
- La partie du discours
- La traduction en anglais
- Les dérivés
- Les mots apparentés
- Les usages grammaticaux courants et rares
- Les exemples en naija avec la traduction en anglais
- etc.

3.4. Résumé

Dans ce troisième chapitre, nous avons fait une brève introduction du corpus que nous avons utilisé, ainsi qu'une présentation détaillée sur les processus d'extraction des deux lexiques. Nous avons analysé statistiquement le lexique morphosyntaxique et montré quelques phénomènes grammaticaux intéressants dans la langue de naija. Nous avons montré comment fouiller des erreurs d'annotation à l'aide du lexique morphosyntaxique afin d'avoir des treebanks propres. En plus, nous avons présenté comment construire des cadres de sous-catégorisation du mot à partir des treebanks annoté en SUD.

Les résultats des deux lexiques et le corpus propre présentés dans ce mémoire ne représentent pas les résultats finaux, parce que les treebanks du projet NaijaSynCor sont toujours en train d'être améliorés, et les deux lexiques ont besoin d'être mis à jour simultanément. À l'avenir, de plus en plus des ressources linguistiques seront utilisées pour enrichir le corpus de naija et produire des lexiques de meilleure qualité et plus complet.

Chapitre 4

Conclusion

6.1. Résumé

Ce mémoire décrit l'extraction de deux lexiques à partir du treebank du projet NaijaSynCor. D'une part, nous avons analysé statistiquement les entrées du lexique morphosyntaxique pour mieux connaître le vocabulaire du naija. Nous avons également utilisé ce lexique pour la fouille des erreurs d'annotation sur le treebank. En effet, l'extraction du lexique, la fouille des erreurs et la correction constituent un processus circulaire qui pourrait se répéter plusieurs fois afin d'obtenir la version propre du lexique et le golden treebank. D'autre part, nous avons extrait les arguments potentiels des mots avec les informations morphologiques et syntaxiques, et nous les avons regroupés pour produire un lexique de cadres de sous-catégorisation.

6.2. Reste à faire

- **Vérification du lexique de cadres de sou-catégorisation**

  En raison du manque de temps, nous n'avons pu que produire des entrées de cadres de sous-catégorisation pour notre lexique syntaxique, il reste encore un processus de vérification par des linguistes. Il faut davantage réfléchir à ce qui doit être ou ne pas être dans le cadre de sous-catégorisation. Dans notre étude, nous avons mis pour chaque mot son gouverneur, son sujet et ses compléments de type "comp" possibles dans le cadre de sous-catégorisation. Est-ce que le gouverneur doit également être inclus dans le cadre sous-catégorisation de tous les verbes ? Est-ce que tous les compléments extraits font partie du cadre de sous-catégorisation d'un mot. Ce sont des questions nécessaires à réfléchir pour rendre le lexique de cadres de sous-catégorisation propre. Ensuite, nous pourrons mener une analyse quantitative sur le lexique de cadres de sous-catégorisation, nous pouvons calculer les occurrences de chaque cadre de sous-catégorisation, nous pouvons aussi savoir quels mots partagent les mêmes cadres de sous-catégorisation et quels mots ont plus de cadres que les autres mots. À

travers ces analyses, nous pourrons avoir une connaissance plus profonde sur les usages grammaticaux des mots en naija.

- **Réduction des travaux manuels sur la correction de la partie du discours**

  Comme mentionné dans la section 3.2.4, chaque fois qu'une erreur potentielle sur la partie du discours est détectée, que ce soit elle puisse être corrigée automatiquement ou non, il est nécessaire pour les linguistes de reconsidérer la structure syntaxique de l'arbre de dépendance de la phrase, cela cause pas mal de corrections manuelles. Pour réduire ces travaux de vérification manuelle, un parseur basé sur des règles, qui peut prendre en compte les informations morphosyntaxiques du mot et les relations syntaxiques entre le gouverneur et son dépendant en même temps, doit être construit.

- **Fusionner les deux lexiques en un lexique syntaxique plus idéal**

  Actuellement, nos deux lexiques sont plutôt séparés, mais il est possible d'établir un lien entre eux, afin de construire un lexique syntaxique plus idéal. Ce lexique doit contenir un vocabulaire relativement complet de la langue ; il peut définir le mot par ses informations morphosyntaxiques, telles que la catégorie du mot, le sens du mot, la flexion verbale, les dérivés, etc. ; il est aussi capable de montrer les usages du mot par ses cadres de sous-catégorisation en fournissant des exemples à côté de chaque cadre. La forme des entrées doit être unifiée et structurée, ce qui permettrait de fouiller les données plus facilement. Et chaque fois il y a de nouvelles données, le lexique doit être mis à jour immédiatement.

- **Tester les méthodes d'extraction sur d'autres langues**

  À l'avenir, nous espérons que les méthodes d'extraction et les idées proposées dans ce mémoire peuvent être étendues à d'autres langues. Étant donné que tous les treebanks Universal Dependencies partagent le même schéma d'annotation que le treebank sur lequel nous avons travaillé (par le biais de la conversion UD vers SUD), nos méthodes devraient fonctionner même si certaines langues ont des règles d'annotations différentes que celles du treebank du naija. Dans ce cas-là, certains paramètres de l'algorithme ont besoin d'être modifiés pour mieux s'adapter à une autre langue. Voici quelques modifications possibles qui ont besoin d'être testées dans le futur :

    - Extraction du lexique morphosyntaxique : nos méthodes d'extraction devraient fonctionner pour tous les treebanks annotés dans le schéma

d'annotation SUD et UD, puisqu'ils sont tous encodés au format CONLL-U.

- Extraction du lexique de cadres de sous-catégorisation : nos méthodes d'extraction devraient fonctionner sur les treebanks d'autres langues annoté dans le schéma d'annotation SUD. Cependant, notre outil devra être plus générique si nous voulons l'utiliser avec d'autres ensembles de relations que celui du treebank du naija. Il faudrait avoir la possibilité de paramétrer le jeu de relations sans devoir entrer dans le code. Par exemple, si nous traitons un treebank qui est annoté dans le schéma d'annotation UD, les paramètres relatifs aux relations syntaxiques doivent être modifiés, parce que les étiquettes syntaxiques ne sont pas toutes présentées de la même façon en SUD et en UD. Dans ce cas-là, certaines étiquettes de la relation syntaxique en UD qui n'existent pas en SUD doivent être paramétrées, comme "nsubj", "csubj", "ccomp", "xcomp", "iobj", etc.

Bibilographie

Anna Korhonen, 2002. Subcategorization acquisition, 17–65.

Anna Kupsc, 2007. Extraction automatique de cadres de sous-catégorisation verbale pour le français à partir d'un corpus arboré. TALN 2007, Toulouse, France.

Ben Ohiọmamhẹ Elugbe, Augusta Phil Omamor, 1991. Nigerian Pidgin : background and prospects. Ibadan : Heinemann Educational Books Nigeria PLC.

Bernard Caron, 2017. NaijaSynCor : A corpus-based macro-syntactic study of Naija (Nigerian Pidgin). naijasyncor.huma-num.fr (September 2017).

Bernard Caron, Marine Courtin, Kim Gerdes, et Sylvain Kahane, 2019. A surface-syntactic ud treebank for naija. In Proceedings of Universal Dependencies Workshop, Paris.

Brent M. R.,1993, « From Grammar to Lexicon : Unsupervised Learning of Lexical Syntax », Computational Linguistics, vol. 19, p. 203-222.

Briscoe T. & Carroll J., 1993. Generalised probabilistic LR parsing for unification-based grammars. Computational linguistics.

Briscoe T. & Carroll J., 1997, « Automatic Extraction of Subcategorization from Corpora », Proceedings of the 5th ACL Conference on Applied Natural Language Processing, Washington, DC., p. 356-363.

Carroll J. & Fang A., 2004. The automatic acquisition of verb subcategorisations and their impact on the performance of an hpsg parser. In Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP), p.107–114, Sanya City, China.

Cattell, Ray. 1984. Composite predicates in English. Syntax and Semantics Vol 17. Sydney : Academic Press.

Chomsky, N., 1965. Aspects of the Theory of Syntax. Cambridge, MA : MIT Press.

Claire Gardent, Bruno Guillaume, Guy Perrier et Ingrid Falk, 2006, Extraction d'information de sous-catégorisation à partir des tables du LADL.

Davies, Mark, 2010. The Corpus of Contemporary American English as the First Reliable Monitor Corpus of English. Literary and Linguistic Computing. 25 (4): 447–65. doi:10.1093/llc/fqq018.

Deryle Lonsdale, 2009. A Frequency Dictionary of French: Core Vocabulary for Learners.

Eagles, 1996. Preliminary Recommendations on. Subcategorisation. EAG---CLWG---SYNLEX/P. Version of Aug, 1996.

Eagles, 1996. Recommendations for the Syntactic Annotation of Corpora EAGLES DOCUMENT EAG—TCWG—SASG/1.8 Version of 11th March 1996.

Gardent C., Guillaume B., Perrier G. & Falk I., 2005. Extracting subcategorisation information from maurice gross' grammar lexicon. Archives of Control Sciences, 15(LI), 253–264.

Geert-Jan M. Kruijff, 2002. Formal & Computational Aspects of Dependency.

GJanez Orešnik, 1994. On N. Chomsky's strict subcategorization of verbs.

Han C., Lavoie B., Palmer M., Rambow O., Kittredge R., Korelsky T., Kim N., Kim M. (2000). Handling structural divergences and recovering dropped arguments in a KoreanEnglish machine translation system. In Proc. of the Conference of the Association for Machine Translation in the Americas, LNCS 1934, Berlin/New York: Springer, 40-53.

Hudson, R. 2007. Language Networks: The New Word Grammar. Oxford University Press.

Igor Mel'čuk, 1988. Dependency Syntax : Theory and Practice, SUNY Press.

J. Bresnan, 1982. The mental representation of grammatical relations, 173-281. Cambridge, MA : MIT Press.

Joakim Nivre, 2015. Towards a Universal Grammar for Natural Language Processing. CICLing 2015 : 16th International Conference on Intelligent Text Processing and Computational Linguistics, 3–16. https://doi.org/10.1007/978-3-319-18111-0_1

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, Daniel Zeman, 2016. Universal Dependencies v1: A Multilingual Treebank Collection

Kaplan, R. et J. Bresnan, 1982. Lexical Functional Grammar : A formal system of grammatical representation.

Kim Gerdes, Bruno Guillaume, Sylvain Kahane, Guy Perrier, 2018. SUD or Surface-Syntactic Universal Dependencies : An annotation scheme near-isomorphic to UD.

Kim Gerdes, Sylvain Kahane, 2016. Dependency annotation choices : Assessing theoretical and practical issues of universal dependencies. Proceedings of the 10th Linguistic Annotation Workshop held in conjunction with ACL 2016 (LAW-X 2016). "La difficulté d'annoter de manière cohérente des prépositions complexes qui contiennent généralement un mot de contenu".

Laurence Danlos, 1985. Automatic generation of texts in natural language

Leipzig, 2015. The Leipzig Glossing Rules : Conventions for interlinear morpheme-by-morpheme glosses.

Lucien Tesnière, 1959. Éléments de syntaxe structurale. Chapter 51, paragraph 13.

Maggie Tallerman, 2011. Understanding Syntax. Oxford: Hodder Education. 39-41.

Manning C. D., 1993. Automatic acquisition of a large subcategorization dictionary from corpora. In Proceedings of the 31th Meeting of the ACL, p. 235–242, Columbus, Ohio.

Marine Courtin, Bernard Caron, Kim Gerdes, Sylvain Kahane, 2018. Establishing a language by annotating a corpus : The case of naija, a post-creole spoken in nigeria. In Proceedings of the workshop on Annotation in Digital Humanities (An-nDH), pages 7–11, Sofia.

Matthews, P., 2014. subcategorization. In The Concise Oxford Dictionary of Linguistics. Oxford University Press.

McGinn, Colin, 2015. Inborn Knowledge : the Mystery Within, MIT Press.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL '03, pages 8–15, Stroudsburg, PA, USA. Association for Computational Linguistics

Nicholas Faraclas, 1989. A grammar of Nigerian Pidgin. PhD thesis, University of California at Berkeley.

Nick Cercone, 2010. Introduction to Computational Linguistics.

Ninio, A. 2006. Language and the learning curve: A new theory of syntactic development. Oxford: Oxford University Press.

O'Donovan, Michael Burke, Aoife Cahill, Josef van Genabith, Andy Way,2004. Large-scale induction and Evaluation of Lexical Resources from the Penn-II Treebank.

Ogechi Agbo et Ingo Plag ,2006. The relationship of Nigerian Pidgin English and Standard English in Nigeria : Evidence from copula constructions.

Osborne, T., M. Putnam, and T. Groß 2011. Bare phrase structure, label-less trees, and specifier-less syntax: Is Minimalism becoming a dependency grammar? The Linguistic Review 28, 315–364.

Oxford English Corpus, 2011. The Oxford English Corpus : Facts about the language. OxfordDictionaries.com. Oxford University Press. What is the commonest word?. Archived from the original on December 26, 2011. Retrieved June 22, 2011.

Paul Kroeger, 2005. Analyzing Grammar : An Introduction

Peter Bakker, 2009. Pidgins versus Creoles and Pidgincreoles. The Handbook of Pidgin and Creole Studies, John Wiley & Sons, 130–157.

Petrov, Slav. 2011. A Universal Part-of-Speech Tagset.

Pollard, C., Sag, I.A., 1994. Head-Driven Phrase Structure Grammar. Chicago : University of Chicago Press

Sagot, B., Clément, L., de La Clergerie, E.V. and Boullier, P., 2006. The Lefff 2 syntactic lexicon for French: architecture, acquisition, use

Sagot, Benoît. "The Lefff, a freely available and large-coverage morphological and syntactic lexicon for French." 2010.

Sarkar A. & Zeman D., 2000. Automatic extraction of subcategorization frames for Czech. In Proceedings of Colling 2000.

Sylavin Kahane, 2001. Grammaires de dépendances formelles et théorie sens-texte

V. J. Cook, 1985. Chomsky's Universal Grammar and Second Language Learning, 2–18

Zeldes, Amir, 2017. The GUM Corpus : Creating Multilayer Resources in the Classroom. Language Resources and Evaluation 51(3), 581–612.

Zeman, Daniel. 2008. Reusable tagset conversion using tagset drivers. In The International Conference on Language Resources and Evaluation (LREC) 2008, 213–218. Marrakech.

Annexe

Tous les documents sont accessibles sur mon github :
https://github.com/songyuchencyan/Lexiques_NSC

Fichiers des deux lexiques :

- Lexique morphosyntaxique.xlst
- Lexique de sous-catégorisation.tsv
- Ensemble_infos_arguments_potentiels.tsv

Scripts :

- Extraction_lexique_morphosyntaxique.py
- Fouille_des_erreurs_forme_pos_traits.py
- Ajout_infos_lemme_glose.py
- Ensemble_infos_arguments_potentiels.py
- Extraction_lexique_cadres_souscategorisation.py

Fichiers d'enregistrement des erreurs

- erreurs_forme.xlsx
- erreurs_pos_unique_avec_phrase.txt
- frequence_erreur_pos_unique.xlsx
- erreurs_pos_ambigu_avec_phrase.txt
- frequence_erreur_pos_ambigu.xlsx
- erreurs_trait.xlsx