Université de Montréal

Créer un corpus annoté en entités nommées avec Wikipédia et WikiData : de mauvais résultats et du potentiel

par

Lucas PAGÈS

Département d'Informatique et de Recherche Opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

15 avril 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Créer un corpus annoté en entités nommées avec Wikipédia et WikiData : de mauvais résultats et du potentiel

présenté par

Lucas PAGÈS

a été évalué par un jury composé des personnes suivantes :

Guy LAPALME
(président-rapporteur)

Philippe LANGLAIS
(directeur de recherche)

Stefan MONNIER
(membre du jury)

Résumé

Ce mémoire explore l'utilisation conjointe de WikiData et de Wikipédia pour créer une ressource d'entités nommées (NER) annotée : DataNER. Il fait suite aux travaux ayant utilisé les bases de connaissance Freebase et DBpedia et tente de les remplacer avec WikiData, une base de connaissances collaborative dont la croissance continue est garantie par une communauté active. Malheureusement, les résultats du processus proposé dans ce mémoire ne sont pas à la hauteur des attentes initiales. Ce document décrit dans un premier temps la façon dont on construit DataNER. L'utilisation des ancres de Wikipédia permet d'identifier un grand nombre d'entités nommées dans la ressource et le programme NECKAr permet de les classifier parmi les classes LOC, PER, ORG et MISC en utilisant WikiData. On décrit de ce fait les détails de ce processus, dont la façon dont on utilise les données de Wikipédia et WikiData afin de produire de nouvelles entités nommées et comment calibrer les paramètres du processus de création de DataNER. Dans un second temps, on compare DataNER à d'autres ressources similaires en utilisant des modèles de NER ainsi qu'avec des comparaisons manuelles. Ces comparaisons nous permettent de mettre en valeur différentes raisons pour lesquelles les données de DataNER ne sont pas d'aussi bonne qualité que celles de ces autres ressources. On conclut de ce fait sur des pistes d'améliorations de DataNER ainsi que sur un commentaire sur le travail effectué, tout en insistant sur le potentiel de cette méthode de création de corpus.

Mots-clés : Wikipédia, WikiData, Supervision distante, Entités nommées, Création de corpus

Abstract

This master's thesis explores the joint use of WikiData and Wikipedia to make an annotated named entities (NER) corpus : DataNER. It follows papers which have used the knowledge bases DBpedia and Freebase and attempts at replacing them with WikiData, a collaborative knowledge base with an active community guaranteeing its continuous growth. Unfortunately, the results of the process described in this thesis did not reach our initial expectations. This document first describes the way in which we build DataNER. The use of Wikipedia anchors enable us to identify a significant quantity of named entities in the resource and the NECKAr toolkit labels them with classes LOC, PER, ORG and MISC using WikiData. Thus, we describe the details of the corpus making process, including the way in which we infer more named entities thanks to Wikipedia and WikiData, as well as how we calibrate the making of DataNER with all the information at our availability. Secondly, we compare DataNER with other similar corpora using models trained on each of them, as well as manual comparisons. Those comparisons enable us to identify different reasons why the quality of DataNER does not match the one of those other corpora. We conclude by giving ideas as to how to enhance the quality of DataNER, giving a more personal comment of the work that has been accomplished and insisting on the potential of using Wikipedia and WikiData to automatically create a corpus.

Keywords: Wikipedia, WikiData, Distant Supervision, Named entities, Corpus creation

Table des matières

Liste des tableaux

Table des figures

Liste des sigles et des abréviations

NER	*Named Entity Recognition*, le terme anglais faisant référence à la reconnaissance d'entités nommées
NLP	*Natural Language Processing*, terme anglais pour le Traitement automatique du langage naturel
TALN	*Traitement Automatique du Langage Naturel*, étude du langage humain écrit
CRF	*Conditional Random Field*, un type de modèle probabiliste utilisé pour la classification de séquences de données
Bi-LSTM	*Bidirectional Long Short-Term Memory*, une architecture de modèle répandue dans la littérature du traitement automatique du langage naturel

Remerciements

Je tiens à remercier en premier lieu l'aide continue de mon superviseur de recherche, Philippe Langlais, pour son soutien tout le long de ce mémoire de maîtrise, ainsi que pour ses retours toujours pertinents sur les orientations et résultats de mon travail de recherche.

Je souhaite également remercier Abbas Ghaddar pour le temps qu'il a pris à de multiples reprises pour répondre à mes questions sur son travail sur la ressource WiNER, dont ce mémoire s'est inspiré.

Je souhaite aussi remercier Guy Lapalme dont la suggestion d'utiliser le programme *jq* m'a permis de créer une première approche pour aborder les données massives que j'ai dû manipuler lors de ce mémoire.

Je voudrais également remercier Joel Nothman qui a pris le temps de répondre à des questions quant aux données de son article (Nothman et al., 2013), cité à plusieurs reprises dans des articles liés au sujet de ce mémoire.

Finalement, je voudrais remercier les autres étudiants et étudiantes du RALI. Nos échanges virtuels et en personne m'ont aidé à persévérer dans mon expérience de ce diplôme en pleine pandémie.

Introduction

La reconnaissance d'entités nommées

La **reconnaissance d'entités nommées** (NER) est un sous-domaine du traitement automatique du langage naturel (TALN) dont le but est d'identifier puis de classifier des "entités nommées" dans des données textuelles. Une **entité nommée** peut être définie comme un élément d'importance dans un texte, c'est-à-dire un "objet" du texte. Le terme a été utilisé pour la première fois lors de la sixième *Message Understanding Conference* (Grishman and Sundheim, 1996). Un système de NER prend en entrée une séquence de tokens $s = \langle t_1, t_2, ..., t_N \rangle$ et y associe une séquence de prédictions $P = \langle c_1, c_2, ..., c_N \rangle$, avec $c_{1 \leq n \leq N}$ une des classes de NER définies par la tâche. La conférence MUC-6 (Grishman and Sundheim, 1996) définissait les classes *personne*, *organisation*, *emplacement géographique*, *date*, *pourcentage* et *monnaie*. Le domaine du NER a beaucoup étudié les classes *personne*, *organisation*, *emplacement géographique* et *misc* (autre) utilisées par le corpus anglais de la conférence CoNLL03 (Tjong Kim Sang and De Meulder, 2003), dont on peut voir un exemple de phrase sur la Figure 0.1.

> [ORG EU] rejects [MISC German] call to boycott [MISC British] lamb .

Figure 0.1 – Exemple de phrase annotée issue du corpus d'entraînement anglais proposé par la conférence CoNLL03 (Tjong Kim Sang and De Meulder, 2003). Les mots annotés sont entre crochets en bleu et précédé de leur classe dans les crochets.

Cependant, différentes ressources ont depuis été publiées dans un effort d'affiner l'ensemble de classes prédites en NER, comme avec le corpus OntoNotes (Pradhan et al., 2012) qui contient 18 classes, ou le système FIGER (Ling and Weld, 2012a) qui en utilise 112 et prend en compte la possibilité qu'une entité soit prédite dans plusieurs classes. Le NER est par ailleurs un élément important des systèmes de TALN répondant à d'autres tâches, comme la traduction automatique (Babych and Hartley, 2003) ou la recherche d'information (Guo et al., 2009; Raviv et al., 2016) qui peuvent profiter de la détection d'entités nommées dans leur entrée.

Différents types d'approches à la reconnaissance d'entités nommées existent dans le domaine qu'on distingue entre les approches dites traditionnelles et celles plus actuelles. Les premières sont partagées entre les approches par règles (utilisant des gazeteers (Etzioni et al., 2005) ou des schémas syntactico-lexicaux (Zhang and Elhadad, 2013)), les approches non supervisées comme la détection d'entités nommées basée sur des clusters de contextes similaires (Nadeau and Sekine, 2007) et les approches d'apprentissage supervisé basé sur des features (comme celles basées sur la morphologie (Mikheev, 1999)) données à des modèles d'apprentissage automatique comme les modèles de Markov cachés (Bikel et al., 1999). Les approches actuelles utilisent l'apprentissage de représentations contextuelles afin de produire des modèles très performants à partir de grandes quantités de données. On distingue par exemple différents modèles importants dans cette lignée comme ELMo (Peters et al., 2018) ou le modèle *transformer* BERT (Devlin et al., 2019) qui a marqué la littérature du domaine. Il existe maintenant une variété d'outils et programmes pour la tâche du NER comme NeuroNER (Dernoncourt et al., 2017), Stanford-NLP (Manning et al., 2014) ou encore Flair (Akbik et al., 2018).

Un des problèmes rencontrés par les méthodes supervisées est le manque de grandes quantités de données annotées pour le NER. Cela a amené à la multiplication d'approches utilisant Wikipédia comme matériel d'entraînement, comme (Nothman et al., 2013), (Al-Rfou et al., 2014) et (Ghaddar and Langlais, 2017), des articles dont les auteurs utilisent tous les ancres des liens de Wikipédia comme entités nommées pour produire un corpus de NER. Suivant ces travaux, ce travail de mémoire utilise également Wikipédia de cette façon et s'intéresse à l'utilisation de la méthode de la supervision distante dans ce procédé, comme l'avaient fait Ghaddar and Langlais (2017) avec Freebase.

La supervision distante

La **supervision distante**, concept introduit par (Mintz et al., 2009) pour l'extraction de relations, fait référence à l'utilisation de l'information encodée dans une base de connaissances pour automatiquement produire des décisions dans un processus d'annotation qui nécessiterait habituellement un intervenant humain ou un modèle entraîné. Les bases de connaissances constituent un type particulier de base de données encodant de l'information pour répondre à un besoin, comme contenir l'information d'une organisation ou informer le public. Avec cette méthode, le rôle de l'annotateur humain est remplacé par un processus automatique utilisant de l'information encodée dans une base de données, ce qui rend le processus plus rapide et moins coûteux. La supervision distante peut être utilisée pour fabriquer du matériel d'entraînement annoté pour le NER, comme l'ont fait Ghaddar and Langlais (2017) en utilisant Wikipédia et la base de connaissances Freebase. Comme proposé par (Nothman et al., 2013), on peut transformer les ancres des liens contenus dans les

articles Wikipédia en annotations d'entités nommées. Il faut par la suite catégoriser ces entités nommées dans un ensemble de classes de NER prédéfini. Ghaddar and Langlais (2017) proposent pour ce faire d'utiliser les pages de la base de connaissances Freebase afin de classifier ces ancres parmi les quatre classes LOC, PER, ORG et MISC (Tjong Kim Sang and De Meulder, 2003) et de produire ainsi la ressource WiNER. Dans leur article, l'utilisation de WiNER comme matériel d'entraînement permet d'améliorer les résultats de plusieurs modèles sur différents corpus de test du domaine, démontrant l'intérêt de la méthode. Zhu et al. (2019) proposent également une approche similaire en utilisant DBpedia — une autre base de connaissances, produite automatiquement à partir de Wikipédia — et obtiennent de bons résultats, continuant de prouver l'intérêt de la supervision distante pour cette application. La base de connaissances Freebase n'est plus à jour et continuer à l'utiliser introduit de ce fait un risque de biais d'information en n'incluant pas de données actuelles. En effet, Freebase a été officiellement fermé en 2015. Également, DBpedia est produit automatiquement à partir de Wikipédia, nécessitant de nombreux calculs pour garder la base de connaissances à jour. Pour ces raisons, l'utilisation de nouvelles bases de connaissances devient pertinente et c'est pour cette raison que ce mémoire s'intéresse à WikiData. Cette dernière ressource est une base de connaissances gérée par la fondation Wikimedia — en charge de Wikipédia — et suit leurs principes de données ouvertes et éditées collaborativement. De ce fait, WikiData reste continuellement à jour grâce à une communauté active d'éditeurs. Par ailleurs, cette base de connaissances est directement connectée à Wikipédia par des liens entre les pages qui parlent des mêmes concepts sur les deux ressources. WikiData contient maintenant 97M de pages ainsi que 1,3G de faits, permettant d'encoder une richesse d'information importante dans cette ressource. Pour les différentes raisons invoquées précédemment, le couple Wikipédia-WikiData constitue un choix intéressant pour produire un corpus annoté avec des entités nommées. On verra cependant dans le chapitre 6 de ce mémoire que l'approche amène à des performances décevantes.

Contributions et structure de ce mémoire

Ce mémoire s'intéresse à l'utilisation de la base de connaissances collaborative WikiData pour produire des annotations de classe NER pour les ancres contenues dans Wikipédia. Il inspecte les meilleures pratiques d'élaboration d'une ressource NER à partir de Wikipédia et WikiData. Ce couple de ressources n'a jamais été utilisé ensemble dans ce but précis. Cette piste de recherche nous paraît intéressante pour son exploration d'une base de connaissances ouverte et collaborative, avec une croissance garantie sur un temps long et encadré par un organisme avec de l'expérience de ce genre de projets. La contribution principale de ce mémoire est de ce fait d'explorer cette utilisation conjointe et d'étudier les meilleures pratiques à observer pour ce faire. Nous avons développé une infrastructure logicielle qui

permet de digérer un *dump* de Wikipédia et de WikiData et qui produit une ressource annotée en entités nommées. Cette architecture est disponible sur un dépôt GitHub [1].

Le premier chapitre rappelle les travaux reliés à notre sujet dans la littérature. Dans le deuxième chapitre, on décrit les ressources utilisées : Wikipédia et WikiData. Le troisième chapitre décrit les différentes façons dont on peut utiliser ces deux sources d'information pour la création d'une ressource NER. Dans le quatrième chapitre, on précise le protocole d'évaluation utilisé afin de situer les approches décrites dans ce travail. Le cinquième chapitre explore les paramètres du processus de création de DataNER une fois toutes les données récoltées. Finalement, le chapitre six compare la ressource produite à différentes ressources similaires et explique les mauvais résultats obtenus.

1. `https://github.com/LucasPages/dataner_creation`

Chapitre 1

Travaux reliés

1.1. Reconnaissance d'entités nommées

La reconnaissance d'entités nommées (NER) est un sous-domaine du traitement automatique du langage naturel (TALN) dont le but est d'identifier puis de classifier des "entités nommées" dans des données textuelles non structurées. Une **entité nommée** peut être définie comme un élément d'importance dans un texte, c'est-à-dire un "objet" du texte. Le terme a été utilisé pour la première fois lors de la sixième *Message Understanding Conference* (Grishman and Sundheim, 1996). La conférence définit alors la tâche du NER comme étant "identifier les noms des personnes, organisations et emplacements géographiques dans un texte" [1] Le but de cette conférence — organisée par le DARPA, une division de recherche de l'armée américaine — est alors d'extraire automatiquement de l'information dans du texte à travers différentes tâches, dont le NER. La conférence qualifie alors cette nouvelle tâche de "Named Entity Recognition and Classification" et la considère comme élément essentiel de l'Extraction d'Information. Comme ce dernier nom le suggère, la tâche du NER se divise en fait en deux étapes distinctes :

(1) identifier les entités nommées

(2) classifier les entités détectées dans un ensemble de classes déterminé

La conception d'un système de NER doit de ce fait passer par la sélection d'un ensemble de classes dans lesquelles les entités doivent être classifiées. La littérature du NER a graduellement diversifié les ensembles de classes auxquels elle s'intéressait avec l'introduction de différents corpus à travers son histoire.

1. Il s'agit d'une citation de l'article de MUC-6 : "the 'named entity' task, which basically involves identifying the names of all the people, organizations, and geographic locations in a text."

1.1.1. Les jeux de données en NER

Le domaine du NER s'est traditionnellement concentré sur des classes génériques dont MUC-6 a posé les bases : personnes, organisations et emplacements géographiques, types auxquels la conférence fait référence par le terme *enamex*. Ceux-ci servent de base largement utilisée dans le domaine. On peut en fait constater que les classes utilisées par les systèmes de NER ont évolué en suivant la publication de différents corpus de NER au travers de la vie du domaine. Ces corpus sont souvent issus de textes de presse (Tjong Kim Sang and De Meulder, 2003; Grishman and Sundheim, 1996), mais également d'articles de blogs ou des réseaux sociaux, marquant de ce fait une diversification des sources du texte présent dans la littérature, principalement après l'année 2005.

En 2003, la conférence CoNLL03 (Tjong Kim Sang and De Meulder, 2003) introduit deux corpus de NER issus de nouvelles de l'agence de presse Reuters. Ces corpus sont annotés avec les *enamex* et y ajoutent la classe MISC qui sert à classifier les entités qui ne tombent dans aucune de ces trois classes. Les données de cette conférence, ainsi que l'ensemble de classes qu'elles utilisent, ont depuis été très étudiés par la littérature. À partir de 2009, on constate la multiplication de corpus issus de Wikipédia, comme le corpus WikiGold (Balasuriya et al., 2009) qui est produit en annotant manuellement des pages Wikipédia avec l'ensemble de classes de la conférence CoNLL03 [2]. On peut également mentionner les corpus HYENA (Yosef et al., 2012) et WikiFiger (Ling and Weld, 2012b) qui introduisent des ensembles de classes plus fins avec respectivement 505 et 112 classes NER. On constate aussi la conception de données propres à des domaines spécifiques comme le domaine biomédical avec des classes comme "protéine" ou "ADN" dans le corpus GENIA (Ohta et al., 2002). La reconnaissance d'entités nommées dans des domaines précis répond à des besoins de gestion de données massives qui prendrait trop de temps à analyser à la main. Une variété de corpus issus du domaine bio-médical apparaissent entre 2004 et 2018 comme GENETAG (Tanabe et al., 2005) avec deux classes qui servent à identifier les gènes et les protéines ou le corpus NCBI-disease (Dogan et al., 2014) qui identifie les maladies au sein de résumés d'articles issus de la revue PubMed. Publiée en 2012, la dernière version du corpus OntoNotes (Pradhan et al., 2012) — avec 18 classes NER — marque la publication d'un corpus avec des entités annotées dans des classes fines et qui est également largement étudié dans la littérature.

Le domaine du NER s'est de ce fait intéressé à une variété de sujets et domaines depuis sa création dans les années 90. On peut constater la richesse de la littérature en données malgré la grande place qu'y prend le corpus CoNLL03.

2. LOC, PER, ORG et MISC.

1.1.2. Les approches

Le NER est le plus souvent traité dans la littérature comme un problème de prédiction d'une séquence de tags. Étant donné une séquence de tokens en entrée, on veut récupérer une séquence de tags — classes — en sortie. Ce principe est illustré sur la Figure 1.1 qui décrit la structure de la plupart des systèmes — *modèles* — de NER. On peut constater qu'on y associe en entrée une séquence de mots — ou *tokens* — à une séquence de tags de classes NER en sortie.

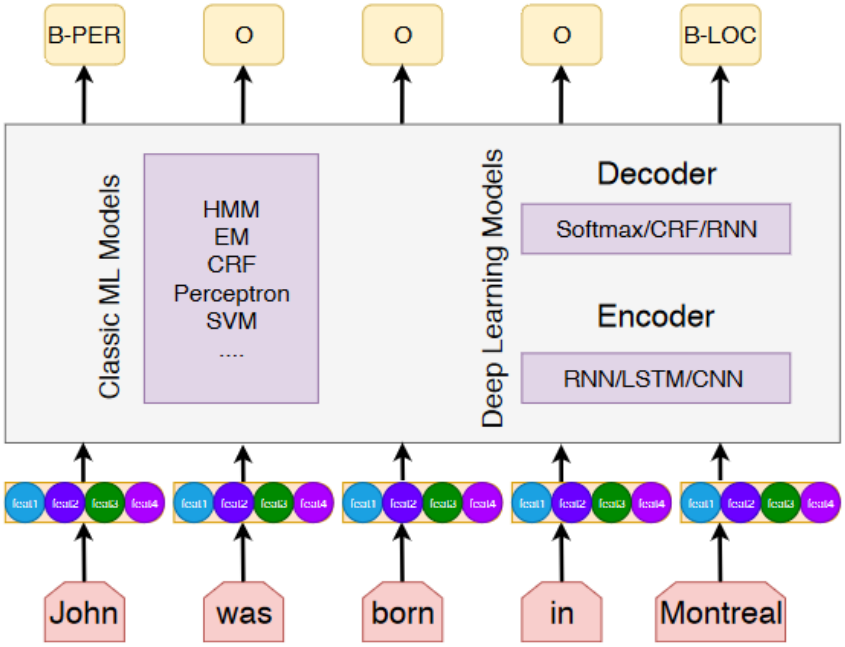


Figure 1.1 – Représentation du fonctionnement général des modèles de NER. Les tokens en entrée sont transformés en *features* qui sont données à un classificateur qui donne comme sortie une séquence de tags. Cette image est issue d'une figure de la thèse de doctorat d'Abbas Ghaddar (`https://papyrus.bib.umontreal.ca/xmlui/handle/1866/24799`)

Une variété d'approches a été utilisée pour la tâche du NER. Come on peut le voir au centre de la Figure 1.1, on peut distinguer les approches dites "traditionnelles" d'apprentissage automatique des approches actuelles qui utilisent l'apprentissage profond.

Les approches traditionnelles utilisent des méthodes d'apprentissage machine comme les Modèles de Markov Cachés (HMM) (Eddy, 1996), ou les modèles de type CRF (Finkel and Manning, 2009) — pour *Conditional Random Field*, un autre modèle probabiliste utilisé dans la prédiction de séquences de *tags*. Comme les modèles d'apprentissage profond, ces modèles "apprennent" à partir de données annotées, mais nécessitent cependant un grand travail de fabrication de *features*. Ce terme fait référence à des caractéristiques trouvées dans les données que l'on donne à un modèle prédictif pour représenter l'entrée, comme les parties

de discours ou les représentations vectorielles de mots — également appelées *embeddings*, ces vecteurs entraînés sur de grands corpus représentent les mots d'un texte dans un espace géométrique. C'est dans cet espace de features que le modèle doit ensuite trouver les règles qui lui permettront de généraliser sur de nouvelles données.

On peut distinguer une variété de features pour ces modèles telles que :

— des features de capitalisation qui caractérisent la casse des caractères d'un mot : AllCaps, AllLower ou UpperFirst

— des features qui indiquent les mots se trouvant dans le contexte direct d'un token, comme le fait Settles (2004)

— des features marquant la présence de certains mots du texte dans des lexiques, permettant de les lier entre eux (Settles, 2004)

— similairement, l'utilisation de *gazetteers* permet de fournir à un modèle des listes de mots importants pour un domaine précis. On les retrouve dans de nombreux travaux et ils peuvent être produits automatiquement (Toral and Muñoz, 2006)

Ces features sont principalement utilisées par les approches traditionnelles, mais on peut parfois les retrouver dans les approches d'apprentissage profond, comme c'est le cas avec les features de capitalisation (Lê, 2019a). Cependant, les modèles d'apprentissage profond restent relativement indépendants de ce genre de procédés. En effet, ces modèles sont basés sur l'utilisation de grandes quantités de données pour apprendre eux-mêmes des représentations de l'entrée à partir de leur architecture et de l'algorithme d'apprentissage utilisé. Pour ce faire, ces modèles utilisent une structure de type *encoder* / *decoder* pour créer une représentation de l'entrée avec l'*encoder* au travers des *couches* du modèle avant d'être traduit en prédiction par le *decoder* (voir Figure 1.1). Les *couches* de l'encoder, paramétrées pendant l'entraînement, permettent au modèle de développer une compréhension fine de l'entrée.

Le modèle BiLSTM-CRF popularisé par (Lample et al., 2016; Ma and Hovy, 2016b) est un modèle d'apprentissage profond toujours très largement utilisé dans la littérature de NER pour ses bonnes performances dans les tâches de prédiction de séquences de tags. Ce modèle combine un modèle LSTM (Hochreiter and Schmidhuber, 1997) — pour Long Short-Term Memory — bidirectionnel avec une couche CRF (Lafferty et al., 2001) sur la sortie qui permet de prédire la meilleure séquence de tags à choisir. L'architecture LSTM est un type de modèle récurrent en apprentissage profond qui est fait pour prendre en entrée des séquences de mots. Elle est devenue très populaire en NLP et son utilisation conjointe avec une couche CRF comme *decoder* (voir Figure 1.1) est maintenant fréquente. L'utilisation de la couche CRF explique les bonnes performances du modèle en considérant la séquence en entier et non pas les tokens un à un.

Les modèles d'apprentissage profond sont dépendants des représentations vectorielles des mots — ou *embeddings* — qu'on leur fournit et de leur interaction avec l'architecture du modèle. Ces représentations sont des vecteurs de dimension variable selon leur entraînement

permettant de représenter un mot donné dans un espace géométrique. Ces *embeddings* obtenus sur de grandes quantités de données permettent aux modèles, une fois passés à travers les couches de l'architecture du modèle, de ne pas avoir besoin de features et de pouvoir effectuer des prédictions directement depuis une entrée textuelle. Précisément, on peut fournir aux modèles :

— des représentations statiques — qui ne sont pas modifiées après leur entraînement — comme les embeddings Glove (Pennington et al., 2014) ou word2vec (Mikolov et al., 2013), très fréquemment utilisés,

— des représentations contextuelles — qui changent selon le contexte de chaque mot — comme ElMo (Peters et al., 2018), Flair (Akbik et al., 2018) ou BERT (Devlin et al., 2019)

— des représentations de caractères (Ma and Hovy, 2016b; Lample et al., 2016)

Les performances de ces modèles sont les meilleures quand ces différents types sont utilisés ensemble et concaténés de façon à créer des représentations multiples pour l'entrée du modèle. Comme le montrent Wang et al. (2020) il est possible d'obtenir des performances état de l'art en NER en travaillant sur la construction de ces représentations multiples sur un modèle BiLSTM-CRF. Cela indique l'importance de cette partie des modèles pour augmenter leurs performances et la très grande importance qu'a pris la recherche des meilleures représentations dans la littérature.

1.2. Supervision distante

Un des problèmes rencontrés dans le domaine du NER est le manque de données d'entraînement annotées de grande taille. Étant donné la popularité des modèles d'apprentissage profond et le nombre grandissant de domaines où les appliquer, apporter une solution à ce problème devient important.

La supervision distante est un paradigme qui peut répondre à cette situation. Ce concept décrit l'utilisation de données préexistantes afin de créer automatiquement — et donc à moindre coût — des données d'entraînement. Ce procédé permet de ne pas complètement se fier à des annotateurs humains pour produire des données annotées, comme un corpus annoté avec des entités nommées, ce qui est l'approche usuelle. Pour ce faire, on utilise des bases de connaissances, un type de base de données encodant de l'information selon un besoin informationnel donné, que ce soit de l'information limitée à un domaine, ou de l'information plus générale. On en distingue plusieurs, comme DBpedia (Lehmann et al., 2014), une base de connaissances produite automatiquement à partir des données structurées de Wikipédia, YAGO (Kasneci et al., 2009), une base de connaissances contenant des données issues de Wikipédia, et des bases de données comme WordNet (Fellbaum, 1998) et GeoNames

(Wick, 2015), ou encore WikiData (Vrandečić and Krötzsch, 2014), la base de connaissances collaborative de la Fondation Wikimedia.

Le domaine a souvent utilisé Wikipédia comme source de données partiellement annotées, comme l'a fait (Nothman et al., 2013), traçant les débuts d'une méthode populaire. Ce type de méthode utilise généralement la présence dans le texte de Wikipédia d'hyperliens vers d'autres pages de la ressource collaborative : des *ancres*. Nothman et al. (2013) utilisent cette structure afin de créer des entités nommées classifiées dans les catégories de la conférence CoNLL03 : LOC, PER, ORG et MISC. Pour ce faire, les auteurs classifient les pages Wikipédia dans ces catégories avec des règles et propagent les classes trouvées sur les hyperliens du texte de la ressource. Cependant, la faible quantité d'ancres dans Wikipédia cause une faible *couverture* — proportion de tokens annotés parmi tous les tokens — de la ressource. De ce fait, (Nothman et al., 2013) propose d'utiliser les pages de redirection pour générer plus d'entités nommées.

On remarque par ailleurs l'utilisation de Wikipédia pour générer un corpus avec des classes d'entités nommées fines (Ling and Weld, 2012a). Ces classes sont récupérées en associant chaque article Wikipédia avec un article Freebase. La ressource Freebase est une base de connaissances encodant de l'information grâce à des triplets. Une partie de l'information encodée par les pages Freebase est celle du type de la page. Cette dernière information est utilisée par Ling and Weld (2012a) afin de l'utiliser comme une classe NER. On retrouve l'utilisation de Freebase dans (Ghaddar and Langlais, 2017) qui utilise la base de connaissances pour classifier les articles de Wikipédia parmi quatre classes : LOC, PER, ORG et MISC, comme l'avaient fait Nothman et al. (2013). Ghaddar and Langlais (2017) proposent également l'utilisation d'information de coréférence du concept décrit par l'article Wikipédia (Ghaddar and Langlais, 2016) et le suivi des liens vers d'autres pages pour générer plus d'entités nommées au sein de leur ressource. En effet, les auteurs proposent de suivre les liens sur une page Wikipédia, et d'annoter la page initiale avec toutes les ancres contenues sur la page atteinte. Ce procédé permet de créer un grand nombre de nouvelles entités.

Plusieurs travaux se concentrent sur la structure des liens dans Wikipédia afin de générer plus d'entités nommées, comme le font West et al. (2015) en recherchant les liens manquants dans Wikipédia. Spécifiquement, leur méthode ajoute uniquement des liens qui améliorent la navigabilité de Wikipédia. Similairement, Raganato et al. (2016) produisent une version de Wikipédia enrichie de nouvelles ancres en trouvant des liens manquants dans Wikipédia grâce à des heuristiques propageant les mentions au sein de leur page, et de celles qui y sont connectées par des liens. L'approche trouve de nouvelles mentions en utilisant la ressource BabelNet (Navigli and Ponzetto, 2012) afin de déduire des mentions présentes dans des versions de Wikipédia écrites dans une langue autre que l'anglais.

Plus récemment, Zhu et al. (2019) ont utilisé DBpedia afin d'annoter les liens dans Wikipédia et un modèle de correction neuronal pour ajouter des liens manquants (faux négatifs)

afin d'augmenter le nombre d'entités nommées dans Wikipédia. Ce modèle de correction neuronale permet de produire la ressource AnchorNER et est entraîné sur la ressource DocRED (Yao et al., 2019) — un corpus destiné à l'extraction de relations produit conjointement par supervision distante en utilisant Wikipédia et WikiData et par des annotateurs humains.

1.2.1. Passer à WikiData

La littérature utilisant la supervision distante (Ghaddar and Langlais, 2017; Ling and Weld, 2012a) pour le NER a beaucoup utilisé Freebase, cependant cette base de connaissances ne peut plus être éditée et peut maintenant seulement être consultée. En effet, reconnaissant l'importance grandissante de WikiData, Google — qui détenait alors les droits de Freebase — a décidé de collaborer avec WikiData en migrant le plus de données possibles depuis Freebase vers la base de connaissances collaborative (Pellissier Tanon et al., 2016). La migration débute en 2014 et Freebase est officiellement fermé en 2015. C'est cette information qui motive l'utilisation de WikiData dans ce mémoire. En effet, WikiData est constamment éditée par sa communauté et reste de ce fait à jour, ce qui permet d'avoir un flot de nouvelles informations qui y sont encodées, contrairement à Freebase dont l'information n'est plus à jour depuis 2015.

Utiliser une base de connaissances à jour permet d'avoir accès à une quantité d'information dont la croissance ne s'arrête pas, en plus d'avoir accès à des données structurées qui correspondent à l'époque d'utilisation de la ressource. Par ailleurs, utiliser une base de connaissances qui suit des principes similaires à ceux de Wikipédia devient pertinent sur le plan de la cohérence des données, ces deux projets étant gérés par la Fondation Wikimedia. Finalement, on peut remarquer que malgré le fait que Freebase et WikiData ne partagent pas le même schéma de données, leur structure reste relativement similaire en tant que bases de connaissances, avec des pages encodant des concepts et des propriétés encodant l'information en les liant entre elles.

On peut de ce fait émettre l'hypothèse que les approches utilisant Freebase peuvent être adaptées à WikiData. Cependant, comme on le verra dans le chapitre 6 de ce mémoire, les performances de l'approche que nous proposons avec WikiData n'égalent pas celles des approches avec Freebase.

1.3. Métriques utilisées

On différencie plusieurs méthodes d'évaluation de la tâche du NER.

1.3.1. La correspondance exacte

La méthodologie communément utilisée dans la littérature est celle de la correspondance exacte entre les prédictions et les entités du corpus de test. Avec cette façon d'évaluer,

on prend en compte les deux sous-tâches du NER : la détection de l'intervalle exacte d'une entité nommée dans le texte et sa classification. De ce fait, une entité seulement partiellement détectée — comme "Gonzales" au lieu de "Chilly Gonzales" — n'est pas considérée comme correctement prédite, indépendamment du fait que la classe prédite puisse être la bonne.

Dans le but de comprendre les prédictions d'un modèle, on les classifie parmi les catégories suivantes :

— Vrai positif : une entité correctement prédite

— Faux positif : une entité prédite qui n'est pas présente dans le corpus de test

— Faux négatif : une entité non prédite, mais présente dans le corpus de test

Le nombre de d'éléments dans chacune de ces catégories permet de calculer les valeurs de *précision* et de *rappel* :

$$précision = \frac{\#VP}{\#VP + \#FP} \qquad rappel = \frac{\#VP}{\#VP + \#FN}$$

La précision et le rappel permettent de comprendre la qualité des prédictions d'un modèle, mais c'est cependant la **F-mesure** qui est utilisée dans la littérature pour mesurer les performances d'un modèle sur des corpus de test. La F-mesure est la moyenne harmonique de la précision et du rappel, qui associe un score aux prédictions en donnant la même importance aux deux valeurs :

$$F - mesure = 2 \times \frac{précision \times rappel}{précision + rappel}$$

Il est également possible d'évaluer la F-mesure d'un système de NER par classe en récupérant la précision, le rappel et la F-mesure pour chacune des classes.

Afin d'évaluer les performances du système à travers toutes les classes, on peut ensuite considérer soit :

— la F-mesure moyenne *macro* : calcule une moyenne des F-mesure de chaque classe,

— la F-mesure moyenne *micro* : calcule une F-mesure à partir des VP, FP et FN toutes classes confondues.

La F-mesure moyenne de type *micro* peut cependant être biaisée par une distribution déséquilibrée des entités parmi les classes du corpus de test. Le score macro est de ce fait plus souvent utilisé dans la littérature. Le domaine du NER utilise fréquemment le script *conlleval*[3] pour effectuer les évaluations de prédictions. Ce script permet de calculer la F-mesure macro d'un ensemble de prédictions ainsi que les F-mesure de chaque classe que le modèle prédit. Il prend en entrée les prédictions d'un modèle et donne le détail de ses performances en moyenne et par classe NER en F-mesure en sortie. L'utilisation de ce script rend

3. `https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt`

les résultats plus facilement reproduisibles comme le suggère l'article (Lignos and Kamyab, 2020), ce qui motive son utilisation dans ce mémoire.

1.3.2. Autres méthodes

On note également l'utilisation d'autres méthodologies d'évaluation moins fréquentes. La conférence MUC-6 (Grishman and Sundheim, 1996) introduit une méthode d'évaluation souple, qui considère une prédiction correcte si elle est partiellement détectée et que sa classe est correctement prédite. Un tel mode d'évaluation permet d'élargir l'éventail des prédictions correctes étant donné la nature parfois subjective des limites d'entités nommées.

Similairement, on dénote l'utilisation de la F-mesure à l'échelle des tokens. Cette version de la F-mesure évalue la détection et classification des tokens des entités et non des entités dans leur ensemble, cette mesure très proche de la F-mesure souple de MUC-6 est utilisée entre autres par (Ghaddar and Langlais, 2017). De la même façon, cette version de la mesure permet de rendre la détection des limites d'une entité nommée plus simple et de pardonner certaines erreurs de prédiction. Comme expliqué dans l'article mentionné, cette méthode permet entre autres d'évaluer un modèle sur des corpus de test ayant différentes définitions des limites d'une même entité nommée.

Finalement, des tentatives d'évaluation plus complexes des prédictions de modèles de NER ont pu être créées comme (Bernier-Colborne and Langlais, 2020) qui s'intéressent à l'évaluation de prédictions de modèles en se concentrant dans les données de test sur des entités non-vues à l'entraînement, ou dont la classe n'est pas la même que dans les données d'entraînement.

Chapitre 2

Les ressources utilisées

On présente ici les deux ressources utilisées dans ce mémoire : Wikipédia et WikiData, et on discute de la pertinence d'utiliser WikiData.

2.1. Wikipédia

Wikipédia est une encyclopédie en ligne collaborative créée en 2001 et contenant à présent des éditions dans 323 langues différentes. Cette encyclopédie couvre une variété de thèmes permettant de rendre accessible gratuitement des connaissances à tout internaute. Wikipédia est une ressource monolinguistique, cela veut dire que chaque version linguistique de Wikipédia est indépendante des autres et évolue selon les éditions de sa propre communauté de bénévoles. Depuis le travail de (Nothman et al., 2008), cette ressource a été utilisée à plusieurs reprises afin d'obtenir une grande quantité de texte pouvant servir à créer du matériel d'entraînement pour des modèles d'apprentissage automatique.

2.1.1. Accès aux données

Afin d'accéder à cette ressource, la fondation Wikimedia met à disposition chaque 20 et premier du mois l'ensemble des articles contenus dans Wikipédia dans des *dumps* disponibles dans différents formats, dont le XML ou le SQL. Dans le cadre de ce mémoire, le dump XML du Wikipédia anglais du 2021-08-01 est utilisé. Il s'agit d'un fichier XML compressé selon le protocole bz2, sa taille est de 6,3Go. Le programme WikiExtractor (Attardi, 2015) permet de manipuler ce dump et d'en extraire tous les articles dans un fichier JSON compressé au format bz2, cette fois de taille 5,4Go. Le fichier obtenu contient sur chaque ligne un document JSON décrivant un article du dump :

— id : son identifiant Wikipédia
— revid : l'identifiant de la dernière édition de l'article
— url : l'adresse URL de l'article
— title : son titre

— text : le texte de l'article.

Le dump du 2021-08-01 contient 15,8M d'articles Wikipédia, cependant 9,5M de ces articles ont un champ "text" vide (voir Figure 2.1). Tous ces articles n'ont pas été vérifiés [1] mais il semblerait qu'une grande partie d'entre eux correspondent à des pages de redirection. Une page Wikipédia peut disposer de plusieurs pages de redirection, ce qui peut expliquer la baisse importante du nombre d'articles après filtrage. Par exemple, l'article "Automated Alice" [2] contient 12 pages de redirection dans ce dump Wikipédia.

Nombre d'articles	Nombre d'articles vides	Nombre d'articles non-vides
15 888 834	9 574 154	6 314 680

Figure 2.1 – Statistiques sur le dump Wikipédia du 2021-08-01. Ici on parle d'articles "vides" quand le texte de ces articles est vide.

Ce sont uniquement les articles dont le champ "text" n'est pas vide qui sont conservés dans le cadre de ce travail de mémoire.

2.2. WikiData

WikiData est une base de connaissances dirigée par la Fondation Wikimedia, une ONG responsable de la gestion de différents projets dont Wikipédia. WikiData a été créé en 2012 avec l'intention de donner accès au public à de l'information structurée en respectant les principes du mouvement Wikimedia. Ce projet a vu jour à la suite de la réalisation que Wikipédia n'était pas une ressource efficace pour communiquer des données cohérentes et claires sur des sujets donnés, du fait de sa nature textuelle ainsi que du fait de l'existence de plusieurs ressources Wikipédia indépendantes écrites dans différentes langues. De ce fait, WikiData a pour but d'organiser en un seul lieu des données traitant du même sujet, proposant toutefois toujours l'information dans plusieurs langues. Comme Wikipédia, WikiData est un projet en ligne collaboratif dépendant de sa communauté pour évoluer. Cela veut dire que tout usager de la plateforme peut éditer les données stockées sur WikiData. Ce principe collaboratif a permis à la base de connaissances de grandement évoluer depuis sa création.

En 2014, Vrandečić and Krötzsch (2014) décrivaient que WikiData était le projet le plus édité de la Fondation Wikimedia et que son nombre de pages dépassait celui de Wikipédia. Par ailleurs, 90 % des éditions étaient effectuées par des *bots*, avec une moyenne de 1 million d'éditions humaines par mois. La même année, à la suite du succès de WikiData, Google décide de faire migrer les données de leur base de connaissances Freebase vers WikiData. Selon la description de la migration que font Pellissier Tanon et al. (2016), celle-ci comprenait différents défis tels que la qualité des données contenues dans Freebase et les droits d'auteurs

1. Au moins 3,4M de ces articles sont des pages de redirection.
2. L'url de la page : `https://en.wikipedia.org/wiki/Automated_Alice`

des données qui y étaient contenues. À la suite du transfert, un total de 14 millions de faits ont pu être intégrés à WikiData grâce aux données de Freebase. Ces faits permettent d'encoder de l'information dans la base de connaissances en liant différentes pages entre elles.

Les données de WikiData sont disponibles sur le site officiel de la ressource [3]. Le site permet de mettre à disposition du public les données structurées du projet à travers des pages décrivant des sujets donnés.

2.2.1. WikiData au lieu de Freebase

Ce travail de mémoire prend suite à différents travaux sur la supervision distante utilisant la ressource Freebase, dont le travail de (Ghaddar and Langlais, 2017).

Freebase est une base de connaissances collaborative créée en 2007. Elle se base sur les concepts de *objets*, *faits*, *propriétés* et *types*. Dans Freebase, les pages décrivent des concepts, comme une personne ou une organisation. Freebase encode de l'information sur ces pages en les liant par des faits, caractérisés par des propriétés.

Par exemple, la page **Barack Obama** a un fait utilisant la propriété **/government/us_president/presidency_number** dont la valeur est "44". La page "Barack Obama" peut utiliser cette propriété dans Freebase car elle dispose du type **/government/us_president**.

Ces quatre concepts constituent les bases du schéma de données de Freebase et lui permet une grande quantité d'information. En janvier 2014, Freebase contenait 44 millions de pages — appelées des *topics* — et 2,4 milliards de faits y encodant de l'information.

2.2.2. Structure

Une page WikiData — ou item WikiData — sert à stocker de l'information sur un sujet donné. Elle est souvent connectée à une (ou plusieurs) page(s) Wikipédia qui traite(nt) du même sujet. Par exemple, la page WikiData de *Chilly Gonzales* est connectée à 12 pages Wikipédia qui correspondent chacune à une version de Wikipédia dans une langue différente : portugais, anglais, espagnol... Une page contient différentes informations structurées qui permettent de décrire l'information qui y correspond :

— un identifiant : **Q[nombre]**

— des paires **propriété - valeurs**

Les paires propriété-valeurs sont la base de l'encodage de l'information sur WikiData. Les propriétés détiennent leur propre page sur WikiData et permettent de décrire une partie de l'information liée à une page grâce à leur valeur. Une page de WikiData bien fournie est donc formée d'une multitude de paires qui permettent de décrire en détail le sujet dont il est question. Une propriété peut par ailleurs détenir plusieurs valeurs. Par exemple, la

3. L'URL du site : `https://www.wikidata.org/wiki/Wikidata:Main_Page`

page WikiData de Chilly Gonzales contient la propriété "occupation" à laquelle sont entre autres associées les valeurs "pianist" et "record producer". Ces paires permettent d'encoder une information qui sur Wikipédia — ou une autre encyclopédie en ligne — devrait être exprimée par une phrase : "Chilly Gonzales is a pianist and a record producer."

Ces paires propriété-valeurs peuvent par ailleurs être enrichies par des propriétés précisant la source de l'information, ainsi que par des paires propriété-valeurs additionnelles permettant de préciser l'information qui y est encodée. Par exemple, sur la page de Chilly Gonzales, la propriété "social media followers" détient la valeur "46,953", et une sous-paire "point in time" : "2 January 2021" permet de préciser cette information : "Chilly Gonzales détenait 46,953 followers sur les réseaux sociaux, **à la date du 2 janvier 2021**."

2.2.3. Accès aux données

Les données contenues dans WikiData sont accessibles à tous selon différentes approches.

2.2.3.1. Le système de requêtes. Les données de WikiData peuvent être atteintes grâce au langage de requêtes SPARQL. Ce langage dispose d'une syntaxe similaire au SQL et permet d'écrire des requêtes sur des bases de données au format RDF, liant les différents éléments de la base RDF selon les exigences de la requête. WikiData met à disposition de tous ses usagers une page en ligne, le *Wikidata Query Service*[4], permettant d'effectuer des requêtes sur toutes les données de WikiData. Ce système permet d'obtenir des données structurées issues de WikiData sous différents formats.

Comme le SQL, le SPARQL est basé sur deux éléments syntaxiques, la clause SELECT qui définit les variables à récupérer dans la base de connaissances ainsi que la clause WHERE qui définit les contraintes émises sur ces variables. Un exemple de requête SPARQL est présenté à la Figure 2.2.

Dans le cadre de WikiData, on cherche alors à interroger des pages qui respectent des propriétés de la base de connaissances. Par exemple, considérons la requête en langage naturel "Qui sont les enfants de Johann Sebastian Bach ?". Cette requête est illustrée en SPARQL simplifiée sur la Figure 2.2, laquelle illustre une requête SPARQL qui récupère les variables liées à la page "Bach" par la propriété "father". Cependant, le SPARQL suit une syntaxe spécifique sur WikiData et il faut utiliser des mot-clés propres à cette base de connaissances pour récupérer ses données avec le WikiData Query Service.

La Figure 2.3 illustre une requête SPARQL sur WikiData syntaxiquement correcte : la propriété ainsi que l'élément faisant référence à Bach ont changé et deux mot-clés sont apparus. **wdt :** signifie que l'élément suivant fait référence à une propriété, les éléments qui lient les items WikiData entre eux, et **wd :** signifie que l'élément suivant fait référence à un item, une page WikiData représentant un concept. **P22** fait référence à la propriété "père",

4. URL du Wikidata Query Service : https ://query.wikidata.org/

```
SELECT ?child
WHERE
{
  ?child father Bach.
}
```

Figure 2.2 – Exemple de "pseudo-requête" SPARQL illustrant sa syntaxe. Il s'agit d'une requête simplifiée pour récupérer les enfants de Bach : on veut récupérer toutes les variables (pages) "child" qui sont liées à la page Bach par la propriété "father".

```
SELECT ?child
WHERE
{
  ?child wdt:P22 wd:Q1339.
}
```

Figure 2.3 – Requête SPARQL syntaxiquement correcte permettant d'interroger des données issues de WikiData. Les mot-clés wdt : et wd : sont propres à WikiData.

dont la description française est "lien familial, parent direct de sexe masculin". Similairement, **Q1339** est l'identifiant de la page de Johann Sebastian Bach, "claveciniste, organiste et compositeur allemand". Cette requête permet de ce fait de récupérer l'ensemble des pages liées à l'item Q1339 par la propriété P22. Lorsque rentrée sur le WikiData Query Service, la requête renvoie 20 résultats en 438 ms. Les résultats correspondent aux identifiants WikiData des pages correspondant à la clause WHERE : les enfants de Johann Sebastian Bach. Ces résultats peuvent être téléchargés sous différents formats : JSON, TSV, CSV, HTML ou SVG. On s'intéresse principalement au format JSON, dans lequel les résultats sont structurés sous la forme d'un tableau de documents JSON, dans chacun un champ "child" a pour valeur le lien vers une page WikiData retournée, comme sur la Figure 2.4.

```
{
"child": "http://www.wikidata.org/entity/Q57225"
}
```

Figure 2.4 – Exemple de document JSON retourné par la requête SPARQL de la Figure 2.3. Le document donné ici en exemple indique la page de Johann Christoph Friedrich Bach (Q57225), un des enfants de Johann Sebastian Bach.

Ce système permet d'obtenir facilement un grand nombre de données issues de WikiData en utilisant la syntaxe SPARQL. Cependant, les requêtes effectuées sur le Wikidata Query Service sont effectuées sur des serveurs utilisés par tous les utilisateurs du service et ne peuvent de ce fait pas accepter un grand nombre de requêtes simultanées à la base de connaissances.

2.2.3.2. Les dumps. C'est ce besoin que viennent rencontrer les **dumps** Wikidata. Ces dumps sont similaires aux dumps Wikipédia en ce qu'ils contiennent toute — ou du moins une grande partie de — l'information contenue dans la base de données.

WikiData propose ces dumps sous différents formats, dont le format JSON que le site recommande d'utiliser. Ces dumps JSON sont émis à mesure d'un tous les deux jours et contiennent une sauvegarde des éléments contenus dans WikiData (ses pages — ou items – et ses propriétés). Un dump JSON de WikiData contient un document JSON par ligne. Un document représente une page de WikiData et contient différentes données issues de cette page, chacune stockée dans un champ différent du document. On compte plusieurs champs :

— ID WikiData
— claims : les propriétés
— alias
— titres de la page
— descriptions de la page
— la date de dernière révision de la page dans le dump
— le type de la page : item ou propriété
— les liens stockés sur la page : vers des pages Wikipédia, vers des bases de données...

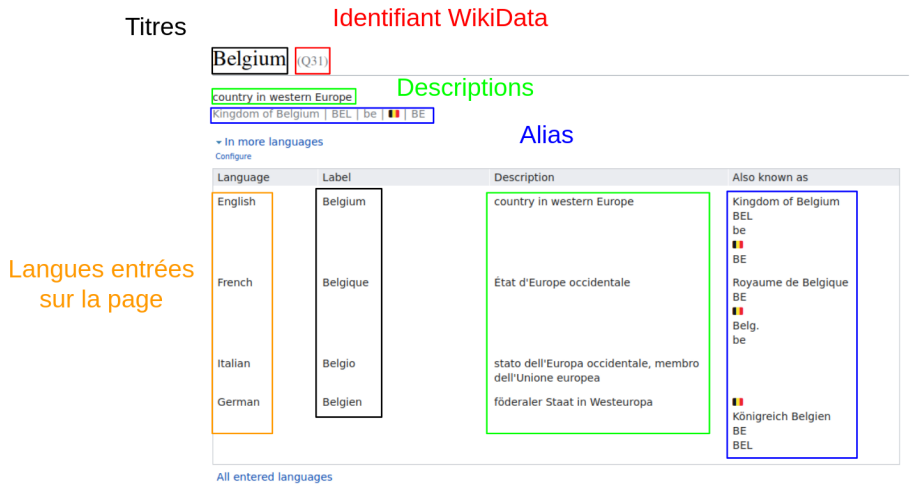
Excepté pour les liens, l'identifiant, la date de dernière révision ainsi que le type, tous ces champs peuvent contenir des valeurs dans plusieurs langues. Ils peuvent ainsi avoir pour valeur un sous-document JSON avec un champ pour chaque langue qui dispose d'une valeur.

Ces documents JSON constituent un parallèle direct avec les données trouvées sur la page WikiData en ligne, comme on peut le voir dans le cadre de la page WikiData de la Belgique sur la Figure 2.5.

L'information de propriété-valeurs, qui permet d'encoder l'information dans WikiData, est contenue dans le champ "claims" de ces objets. Ce champ contient un ensemble de sous-documents JSON qui contiennent l'information de chaque propriété dont dispose cette page. Par exemple, la page Belgique (identifiant Q31), dispose de la propriété P571 ("inception"), qui décrit la date de fondation d'un élément, en l'occurrence la date de création du pays. Associée à cette valeur se trouve une valeur de type date qui indique le jour de création de la Belgique. On peut trouver la propriété sur la page en ligne de la Belgique [5], mais également dans le dump JSON. De cette façon, le champ "claims" dispose d'une valeur décrivant la propriété P571 sous la forme d'un sous-document JSON détaillant la valeur associée à la propriété sur la page de la Belgique, ce sous-document JSON est détaillé sur la Figure 2.6.

Ces dumps permettent ainsi de sauvegarder les items et les propriétés de WikiData dans des fichiers JSON effectuant un parallèle exact avec la version en ligne de la base de connaissances. La grande taille des dumps amène cependant à des temps d'extraction de données longs du fait du format JSON compressé et invite de ce fait à utiliser le système de

5. https ://www.wikidata.org/wiki/Q31

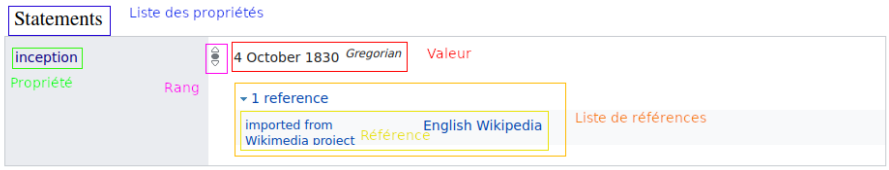


(a) Capture d'écran de la page WikiData de la Belgique. On y retrouve les données de la page dans plusieurs langues.

(b) Capture d'écran d'une version simplifiée document JSON du dump WikiData du 2021-08-02 sur la Belgique. On y affiche uniquement les données en français et en anglais pour simplifier l'affichage, et seulement deux alias sont listés par langue.

Figure 2.5 – Captures d'écrans d'une partie des données de WikiData sur la Belgique (Q31) (ne sont pas représentées ici les liens qui partent de la page et les propriétés). Les différents éléments affichés sur la page sont encadrées par les mêmes couleurs sur les deux documents. On peut voir la présence des mêmes données sur les deux types d'accès.

requêtes en ligne pour un besoin informationnel qui ne requiert qu'une seule requête (comme récupérer tous les noms des propriétés de WikiData). Il devient alors pertinent de considérer les dumps et le WikiData Query Service comme deux modes d'accès conjoints aux données de WikiData et non pas concurrents. En effet, pour un même besoin informationnel, une des deux méthodes peut être plus rapide, ou plus pratique.



(a) Capture d'écran modifiée de la liste des propriétés de la page WikiData de la Belgique. On y voit détaillée la propriété "Inception" avec sa valeur ainsi qu'une référence.

(b) Détail de la propriété "Inception" (P571) de la page WikiData de la Belgique dans le dump JSON WikiData du 2021-08-02. on y retrouve sous format JSON les éléments trouvés sur le site de WikiData : le couple propriété-valeur ainsi que sa référence (un autre couple propriété-valeur). Ce document JSON est en fait un sous-document contenu dans le champ "claims" du document de la Belgique (Q31) dans le dump.

Figure 2.6 – Captures d'écran des deux accès à la propriété "Inception" (P571) de la page WikiData de la Belgique.

2.2.3.3. Dump Wikidata utilisé. Dans le cadre de ce mémoire, le format JSON est retenu suivant la recommandation de WikiData et le dump du 2021-08-02 est utilisé. Ce dump contient 93,9M entités [6] WikiData, cependant seulement 8,8M d'entre elles contiennent un lien vers l'édition anglaise de Wikipédia (voir Table 2.1), cela correspond à environ 9,3 % des pages et 11,1 % des liens contenus dans le dump (voir Figure 2.7).

# items	# items en	# propriétés
93 934 581	8 813 945	9 056

Tableau 2.1 – Statistiques sur le dump WikiData du 2021-08-02. "items en" fait référence aux pages de WikiData connectées au Wikipédia anglais, cela représente 9,3 % des pages du dump. Les autres pages WikiData ont des liens menant vers d'autres projets de la Fondation Wikimedia, ou aucun lien.

6. On parle d'entités WikiData pour qualifier à la fois les propriétés et les items de la base de connaissances.

On peut par ailleurs noter que chaque page WikiData peut contenir des liens vers des pages de projets Wikimedia, "enwiki" — le Wikipédia anglais — est le site vers lequel le plus de liens pointent (voir Figure 2.7), mais on dénombre un total de 871 sites, incluant 323 éditions de Wikipédia dans différentes langues ainsi que d'autres sites de la Fondation Wikimedia tels que Wikimedia Commons [7] ou différentes versions linguistiques de Wiktionary [8].

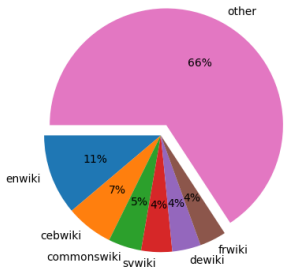


Figure 2.7 – Distribution des liens vers les sites Wikimedia dans le dump WikiData du 2021-08-02. Les sites totalisant moins de 3% du nombre de liens total ont été groupés dans "other". Chaque identifiant sur le graphe correspond au code d'un site de la fondation Wikimedia. Par exemple "enwiki" = Wikipédia anglais. "commonswiki" fait référence au site Wikimedia Commons de la Fondation Wikimedia, ce projet est une base de contenus multimédias libres de droit.

Ce travail de mémoire a pour objet la création d'une ressource en anglais annotée pour le NER, et se concentre de ce fait sur les pages WikiData liées au Wikipédia anglais, un sous-ensemble de WikiData concentrant 8,8M de pages (voir Figure 2.1). Cette sous-partie de WikiData a une taille de 28,5 Go non compressée et 6,9 Go lorsque compressée au format json.bz2. À la date du 2021-08-02, ce sous-ensemble de WikiData contient 8339 propriétés uniques, dont la plus fréquente est **instance of (P31)** avec 8 266 836 pages qui en ont une instance (voir Tableau 2.2). Cela correspond à 93,7 % des pages WikiData connectées au Wikipédia anglais.

Chaque page peut contenir plusieurs paires propriété-valeurs, la page avec le plus de propriétés est la page de **Madonna (Q1744)** avec 394 propriétés. Cependant, en moyenne, les pages contiennent 9,6 propriétés. Par exemple, la page de Chilly Gonzales (Q1532874) contient 54 propriétés, dont les propriétés fréquentes P27, P31 ou P106 (voir Table 2.2). La valeur de la propriété "instance of" (P31) sur cette page est l'item "human" (Q5), une autre page de WikiData.

7. Ressource collaborative contenant des contenus multimédias libres de droit : `https://commons.wikimedia.org/wiki/Main_Page`

8. Un dictionnaire en ligne collaboratif. Voir la version en anglais : `https://en.wiktionary.org/wiki/Wiktionary:Main_Page`

Identifiant	Propriété	Fréquence	Proportion
P31	instance of	8 266 836	93.79
P646	Freebase ID	4 198 406	47.63
P17	country	1 956 378	22.19
P21	sex or gender	1 856 842	21.06
P106	occupation	1 719 613	19.51
P569	date of birth	1 679 929	19.05
P373	Commons category	1 646 598	18.68
P18	image	1 586 642	18.00
P735	given name	1 492 132	16.92
P27	country of citizenship	1 411 875	16.01

Tableau 2.2 – Classement des 10 propriétés les plus fréquentes dans le sous-ensemble connecté au Wikipédia anglais du dump WikiData du 2021-08-02. La colonne "Proportion" décrit la proportion des pages du sous-ensemble de WikiData considéré qui contient la propriété en question. On peut constater que la propriété "instance of" (P31) est contenue dans la quasi-totalité des pages de notre sous-ensemble, avec 93,79 % des pages qui la contiennent. La prévalence de la propriété "Freebase ID" montre la grande proportion des pages de WikiData issues de la base de connaissances Freebase.

Chapitre 3

Utiliser Wikipédia et WikiData pour produire une ressource annotée avec des entités nommées

Dans le cadre de ce mémoire, les articles issus du Wikipédia anglais sont transformés en matériel d'entraînement de NER, la ressource *DataNER*. Pour ce faire, les ancres des liens hypertextes [1] contenues dans le texte du dump Wikipédia sont considérées comme des entités.

Gonzales collaborated with Jhené Aiko on the track "From Time" from Drake's third album Nothing Was the Same.

⇓

Gonzales collaborated with [Jhené Aiko] on the track ["From Time"] from [Drake]'s third album [Nothing Was the Same].

Figure 3.1 – Extrait de l'article Wikipédia de Chilly Gonzales. Les ancres soulignées en bleu (en haut) sont transformées en [entités] (en bas). Ces entités n'ont pas encore de classe NER qui permettent d'entraîner un classifieur.

Ce procédé, illustré sur la Figure 3.1, permet d'identifier un grand nombre d'ancres au sein du texte issu du dump du Wikipédia anglais. Ces entités n'ont cependant pas de classes NER à ce stade. WikiData est utilisé afin d'inférer, pour chaque entité contenue dans le texte issu de Wikipédia, une classe NER parmi une liste prédéfinie grâce à l'information encodée dans WikiData. Il s'agit alors d'effectuer plusieurs tâches :

(1) Lier chaque page Wikipédia à un item WikiData

(2) Classifier chaque item WikiData dans une classe NER grâce à l'information qu'elle encode

1. On entend par "ancre" le texte sur lequel est défini un lien hypertexte au sein d'un article Wikipédia.

(3) Reporter cette information de classe dans le dump Wikipédia et dans le matériel d'entraînement

Ces trois étapes invitent à manipuler deux ressources — un dump Wikipédia et un dump WikiData — de très grande taille, au format JSON. La capacité à facilement et rapidement manipuler et interroger ces dumps devient un élément important de ce travail.

3.1. Lier Wikipédia et WikiData

WikiData et Wikipédia sont deux ressources indépendantes, cependant il est possible de les lier en connectant leurs pages qui traitent d'un même sujet. Les pages WikiData contiennent des liens vers différents sites de la fondation Wikimedia, comme les différentes versions de Wikipédia, ou encore des sites annexes comme WikiBooks ou WikiVoyage. Cette information permet d'effectuer le liage des deux ressources. Grâce à celle-ci, il est possible de lier la page d'un item WikiData à sa page Wikipédia anglaise correspondante, si la page WikiData contient un lien vers le Wikipédia anglais. Effectuer cette correspondance sur tout le dump WikiData permet d'effectuer une association entre les identifiants WikiData contenus dans le dump WikiData et les liens des pages Wikipédia correspondantes. On dispose alors d'un mapping entre chacun des identifiants de pages WikiData (de la forme Q[nombre]) et des titres de pages Wikipédia.


```
"enwiki": {
   "site": "enwiki",
   "title": "Belgium"
 }
```


Figure 3.2 – Extrait du champ "sitelinks" du document JSON de la page WikiData de la Belgique (Q31). Ce sous-document décrit le lien vers la page correspondante dans le Wikipédia anglais. On y retrouve le champs "site" qui décrit la ressource vers laquelle le lien pointe, ainsi que le champ "title" qui indique le titre de la page Wikipédia. Il est alors possible de reconstituer l'URL de la page Wikipédia en question : https ://en.wikipedia.org/wiki/Belgium

Pour le dump Wikipédia du 2021-08-01, le liage de Wikipédia et WikiData a été effectué en 83 minutes de temps CPU [2]. Sur le total de 6 314 680 articles non-vides du dump, un identifiant WikiData a pu être associé à 6 277 897 d'entre eux, ce qui correspond à environ 99,4 % des articles Wikipédia de notre dump. Le dump WikiData du 2021-08-02 contient de ce fait une page WikiData pour quasiment tous les articles du dump Wikipédia anglais du 2021-08-01.

2. Sur un CPU Amd Ryzen 3900X 12 cœurs.

3.2. Étiquetage des pages WikiData

Le liage de Wikipédia et WikiData permet d'accéder, pour chaque page Wikipédia, à une richesse d'information structurée la concernant, contenue dans sa page WikiData. Ces pages WikiData contiennent des triplets qui nous permettent de les classifier dans des catégories NER de notre choix. En effet, WikiData n'encode pas directement une information de classe NER pour chaque page, de ce fait les triplets de WikiData sont ici utilisés pour la déduire.

3.2.1. Description conceptuelle

Les pages de WikiData représentent une variété d'éléments sur lesquels la base de connaissances possèdent des données. Cependant, certaines de ces pages décrivent des classes, des pages auxquelles d'autres pages appartiennent. Par exemple, la page "city or town" (Q27676416) — représentée sur la Figure 3.3 — est une classe dont Montréal (Q340) est une instance.

Ce concept de classes permet d'organiser WikiData selon une structure de graphe où des pages qui correspondent à un même concept (comme "city or town") peuvent être récupérées en interrogeant la base de connaissances. L'appartenance à une classe est encodée sur WikiData par la propriété "instance of" (P31), par exemple la page Montréal (Q340) est liée à la page "city or town" (Q27676416) par la propriété "instance of" (P31). On peut alors associer à une classe toutes les pages WikiData qui en sont une instance (c'est-à-dire qui y sont connectées par P31).

Cependant, il est à noter que WikiData contient des sous-classes de classes. Par exemple, la page "city or town" (Q27676416) est une sous-classe de "city / town" (Q7930989). De cette façon, on peut déduire que la page Montréal (Q340) appartient à la classe "city / town". Mais WikiData n'encode pas ce lien directement et il faut le déduire en suivant les propriétés que la base de connaissances utilise. Le lien de sous-classe est encodé par la propriété "subclass of" (P279) sur WikiData.

Les deux propriétés "instance of" (P31) et "subclass of" (P279) permettent d'encoder dans WikiData des liens d'appartenance nécessaires à l'association d'une page à une classe. En effet, il est possible, en utilisant conjointement ces deux propriétés, d'associer une page à toute classe dont elle est une instance, ou à toute l'arborescence de classes [3] dont cette classe fait partie (comme la classe "city / town" qui fait partie de l'arborescence de classes de la classe "city or town").

Cette conception des classes dans WikiData nous permet en fait de classifier les pages de WikiData dans des classes NER. En effet, comme le propose (Geiß et al., 2018), il est pertinent d'associer les classes NER désirées à des classes dites "racines" dans WikiData. On

3. Ici on fait référence par "arborescence" à toutes les classes dont une classe est une sous classe : si A est une sous-classe de B et B une sous-classe de C, "l'arborescence" de A est B et C.

peut ensuite récupérer toutes les sous-classes (et leurs sous-classes) de ces classes racines afin d'obtenir toutes les classes associées à cette catégorie NER, grâce à la propriété "subclass of" (P279). La classification de pages dans une catégorie NER est ensuite basée sur l'appartenance à une de ces sous-classes, encodée par la propriété "instance of" (P31). C'est ce procédé arborescent qui est illustré sur la Figure 3.3. Le défi principal de cette approche est alors l'association de chaque catégorie NER avec une classe racine dans WikiData et l'inspection des arborescences qui en découlent.

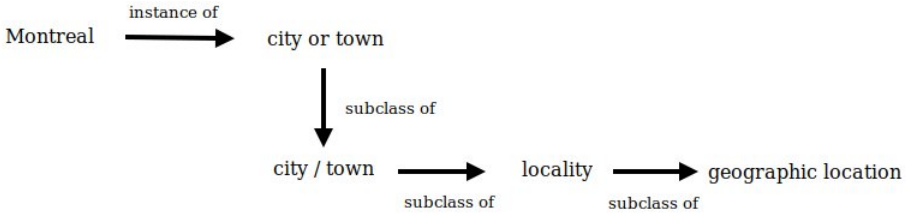


Figure 3.3 – Classification de la page WikiData Montréal (Q340) dans la catégorie LOC grâce au chemin entre les pages "Montréal" (Q340) et "geographic location" (Q2221906). Ce chemin est suivi grâce aux propriétés "instance of" (P31) et "subclass of" (Q21514624). Ce processus de classification est effectué à travers les propriétés de la page WikiData seulement.

3.2.2. Implémentation

Ce mémoire utilise l'outil que (Geiß et al., 2018) met à disposition afin de créer un mapping entre des classes NER et les pages du dump WikiData. Cet outil permet d'associer chaque identifiant WikiData du dump à une classe NER du corpus CoNLL03 : LOC, PER et ORG. Toute page non prédite par l'outil est associée à la classe MISC.

Comme décrit dans l'article (Geiß et al., 2018), la désignation de classes racines et la récupération de leurs sous-classes permet de déduire la classe d'un grand nombre de pages WikiData. Afin de récupérer toutes ces sous-classes, l'outil utilise le WikiData Query Service depuis des scripts Python.

L'outil NECKAr (Geiß et al., 2018) décrit différentes stratégies pour récupérer les bonnes sous-classes pour chaque catégorie NER (LOC, ORG et PER) :

— L'item "geographic location" (Q2221906) est désigné comme classe racine pour la catégorie NER LOC. Cette page permet de récupérer 29 137 [4] sous-classes pour LOC. Cependant, comme indiqué dans (Geiß et al., 2018), une partie de ces sous-classes sont erronées et doivent être retirées.

— Pour la catégorie ORG, la classe "organization" (Q43229) est utilisée. Cette page permet l'extraction de 28 217 sous-classes [5] pour identifier les instances de ORG sur WikiData.

4. A la date du 2021-09-01.
5. A la date du 2021-09-01

— Finalement, pour PER, aucune sous-classe n'est utilisée. C'est la page "human" (Q5) qui est utilisée pour classifier des pages dans la catégorie PER. Si une page est une instance de "human", elle est classifiée dans PER.

Considérant le grand nombre de requêtes que l'opération demande, une fois toutes les sous-classes récupérées, un dump WikiData est interrogé pour récupérer toutes les instances de ces sous-classes. Ainsi, pour chaque page WikiData contenue dans le dump, ses propriétés sont inspectées pour vérifier si cette page est une instance d'une des sous-classes identifiées pour chaque catégorie NER.

Ce procédé permet de classer certaines des classes d'un dump WikiData. En l'occurrence, la méthode est répétée pour chaque classe NER que l'outil utilise : LOC, PER et ORG. Il est cependant à noter que le grand nombre de requêtes effectuées par le code de NECKAr pour récupérer les identifiants WikiData pertinents à chaque classe a parfois amené à un blocage du processus par la plateforme de WikiData.

L'outil stocke les résultats de cette classification dans une base de données locale MongoDB qui peut être interrogée pour récupérer la classe prédite de chaque page WikiData. Il est à noter que NECKAr prédit certaines pages WikiData dans plusieurs classes. Une grande partie de ces pages sont des pays qui sont catégorisés à la fois dans LOC et ORG. Ce phénomène est dû à l'intersection de certaines sous-classes dans les propriétés des pages WikiData.

3.2.3. Utilisation

Pour le dump WikiData du 2021-08-02, NECKAr a pris un temps d'exécution de 29 minutes de temps CPU [6] et a classifié 3 799 068 pages WikiData du dump du 2021-08-02 qui en contient 8 813 945. La distribution de ces prédictions peut être consultée sur la Figure 3.4. Il est à noter que NECKAr ne prédit pas la classe MISC, de ce fait toutes les pages WikiData que l'outil n'a pas prédites sont associées à cette classe.

Aussi, l'outil produit un total de 108 769 pages WikiData avec plusieurs classes NER prédites. Elles sont toutes prédites dans deux classes sauf les pages "Valentin Wolfenstein" (Q16649480) et "Li Guangchang" (Q15900712) qui sont prédites dans les trois classes. Pour ce travail de mémoire, on prend dans ces cas une classe au hasard parmi celles prédites [7].

3.3. Construction de DataNER

À ce stade de la création de la ressource, on dispose d'une base de données locale contenant le texte de chaque article Wikipédia, son identifiant sur WikiData ainsi que sa classe NER prédite. La base de données contient des informations sur 6,2M articles issus du Wikipédia

6. Sur un CPU AMD Ryzen 9 3900X 12-coeurs.

7. Ce choix tenait lieu de première approche, mais un manque de temps a mené à le conserver pour la version définitive de ce travail.

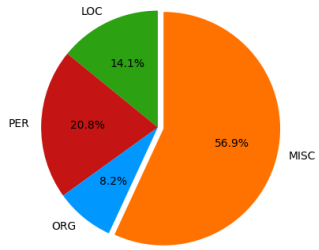


Figure 3.4 – Distribution des classes prédites grâce à NECKAr sur le dump WikiData du 2021-08-02. La majorité des pages WikiData sont associées à la classe MISC.

anglais. La Figure 3.6 décrit la distribution des classes NER parmi les articles de la base de données.

Afin de finaliser l'annotation de la ressource, il faut rapporter l'information de classe NER dans le texte des articles Wikipédia. Pour ce faire, on itère sur le texte de chaque article Wikipédia et, pour chaque ancre rencontrée, une requête à la base de données permet de récupérer sa classe, que l'on peut alors ajouter au texte (voir Figure 3.5).

Gonzales collaborated with [Jhené Aiko] on the track ["From Time"] from [Drake]'s third album [Nothing Was the Same].

⇓

Gonzales collaborated with [PER Jhené Aiko] on the track [MISC "From Time"] from [PER Drake]'s third album [MISC Nothing Was the Same].

Figure 3.5 – Illustration de l'annotation des entités dans le texte. Des crochets sont utilisés pour identifier et catégoriser les entités dans une classe NER. Pour chaque ancre rencontrée, on interroge la base de données locale afin de récupérer sa classe NER.

La base de données contient alors un ensemble de 6,2M articles Wikipédia dont le texte est annoté avec ses entités et leur type. Comme le montre la Figure 3.6 (a) près de la moitié des articles du dump sont classifiés dans MISC. La couverture[8] du dump est de 6.63 %. Cette valeur correspond à la proportion de tokens annotés par rapport au nombre total de tokens dans le corpus. Selon Zhu et al. (2019), ce paramètre est une partie intégrante de la qualité d'un jeu de données et en constitue une métrique pertinente. En comparaison, le corpus CoNLL03 a une couverture de 17 % (Zhu et al., 2019) et le corpus WiNER de Ghaddar and Langlais (2017) a une couverture de 15,19 %.

Ce dump Wikipédia permet donc de construire un corpus annoté avec des classes NER qui contient **127M phrases**, contenant un total de **2,8G tokens**, répartis dans **6,2M**

8. On parle également de RAT : *Ratio of Annotated Tokens*.

(a) (b)

Figure 3.6 – Ces deux graphes représentent différentes distributions des classes NER dans le dump manipulé. On remarque que l'ordre de proportion des classes est toujours le même, avec une **prédominance marquée de la classe MISC**.
(a) Distribution des classes NER parmi les articles du dump Wikipédia du 2021-08-01, contenant un total de 6 277 897 articles. (b) Distribution des classes NER parmi les 88M entités nommées du dump du 2021-08-01.

Phrases	Entités	Tokens	Tokens annotés	Couverture
127 838 651	88 003 368	2 814 043 890	186 657 784	6,63 %

Tableau 3.1 – Statistiques sur le corpus produit avec le dump Wikipédia du 2021-08-01. Chaque colonne correspond à une quantité sauf la dernière : la couverture correspond à la proportion de tokens annotés comme entités nommées par rapport au nombre total de tokens.

d'articles. Parmi ces tokens 186M sont annotés, ce qui correspond à 6,63 % de la totalité (voir Tableau 3.1). Sur ces 186M de tokens, on dénombre 88M d'entités nommées (voir Tableau 3.1).

En moyenne, les articles du corpus contiennent 20 phrases et 448 tokens. Également, les phrases du corpus contiennent en moyenne 22 tokens. On dénombre un vocabulaire contenant **7 889 036 mots uniques**. On peut voir que le corpus contient de courtes phrases, qui contiennent en moyenne 0,6 entités nommées. Ce qui correspond à une moyenne de 14 entités nommées par article du dump Wikipédia du 2021-08-01.

3.4. Augmentation de données

Après la création initiale du corpus à partir du dump Wikipédia, il est possible d'augmenter le nombre d'entités annotées en utilisant différentes techniques d'augmentation de données. Suivant le travail de Ghaddar and Langlais (2017), trois techniques d'augmentation de données sont utilisées :

— on utilise le titre afin d'inférer de nouvelles entités dans la page,

— les liens contenus dans un article du dump de Wikipédia sont utilisés pour récupérer les entités sur les pages cibles et les ajouter à l'article initial (voir Figure 3.9 pour illustration),
— pour chaque article Wikipédia du dump, on récupère ses alias sur sa page WikiData et on cherche et annote ces chaînes de caractères dans l'article Wikipédia.

L'ensemble des ancres et des entités nommées obtenues par augmentation de données permet ensuite de créer la ressource NER *DataNER*.

Dans la suite de cette partie, on prend la page Wikipédia de Chilly Gonzales pour illustrer les différentes méthodes d'augmentation de données.

3.4.1. Utiliser le titre

On peut utiliser le titre d'une page Wikipédia pour produire de nouvelles ancres dans son texte.

Une première approche de ce type propage directement le titre de la page dans le texte. Il faut alors rechercher dans le texte les occurrences exactes du titre. Toutes les occurrences sont alors annotées comme entités nommées avec la classe NER de la page en question. Cette méthode permet de produire **cinq** nouvelles entités nommées dans la page de Chilly Gonzales, comme celle illustrée sur la Figure 3.7.

In 2018 , [PER Chilly Gonzales] launched his own music school .

Figure 3.7 – Cette entité nommée est issue de l'augmentation par expansion du titre. Elle se trouve à la phrase 69 de l'article de Chilly Gonzales dans Wikipédia.

Une autre approche utilise conjointement le titre de la page et la page WikiData correspondante. Elle consiste en l'utilisation de WikiData pour récupérer des alias du titre de la page Wikipédia afin de les rechercher dans le texte de la page Wikipédia. Toutes les occurrences sont alors annotées avec la classe NER de cette page Wikipédia. Cette méthode permet de produire **trente** nouvelles entités nommées sur la page de Chilly Gonzales. La page WikiData de Chilly Gonzales[9] ne contient qu'un seul alias : "Gonzales". On peut en voir l'illustration sur la Figure 3.8.

Apart from his solo carreer , [PER Gonzales] is also a member of the Berlin - based hip - hop band Puppetmastaz .

Figure 3.8 – Cette entité nommée est issue de l'augmentation par alias du titre. Elle se trouve à la phrase 41 de l'article de Chilly Gonzales dans Wikipédia.

9. Page WikiData issue du dump du 2021-08-02.

3.4.2. Utiliser les liens

Il est possible d'utiliser la structure même de Wikipédia afin d'augmenter le nombre d'annotations d'entités dans le dump. En effet, on peut aller chercher des entités non annotées en suivant les liens contenus dans un article donné et en récupérant les entités annotées dans l'article cible. De cette façon, sur l'illustration de la Figure 3.9, on peut aller chercher l'annotation "France" en suivant le lien "Paris" jusqu'à sa page cible, car celle-ci contient l'ancre "France". Ce procédé permet d'augmenter rapidement le nombre d'annotations d'entités en récupérant des ancres supposément rapprochées par leur sujet (comme un pays et une de ses villes comme sur la Figure 3.9).

Figure 3.9 – Illustration du processus d'augmentation de données utilisant les liens des articles Wikipédia. Illustration issue de la présentation de thèse d'Abbas Ghaddar.

Ainsi, on inspecte chaque ancre sur un article donné et on récupère sa page Wikipédia. Une fois cette page obtenue, on récupère toutes les entités annotées que son texte contient, et on les ajoute au texte de la page Wikipédia à augmenter. Ce procédé permet d'obtenir un très grand nombre de nouvelles annotations.

On peut par cette méthode obtenir **398** nouvelles entités nommées sur la page Wikipédia de Chilly Gonzales. On en illustre une sur la Figure 3.10.

singer [PER Jane Birkin] and indie rocker Leslie Feist .

Figure 3.10 – Extrait de la phrase 40 de l'article de Chilly Gonzales. On peut y voir l'entité nommée "Jane Birkin" produite en suivant l'ancre "Feist" de la phrase 3 vers sa page Wikipédia.

3.4.3. Utiliser les alias WikiData

Un des champs contenus dans les pages WikiData contient des **alias**, des titres alternatifs pour une page WikiData donnée. Cette information permet d'augmenter le nombre d'entités

contenues dans un article. En effet, on peut ajouter à l'article Wikipédia des annotations des alias de ses ancres déjà annotées. Pour ce faire, on itère sur les ancres de l'article à augmenter, et pour chacune d'elle, on interroge la page WikiData correspondante et on y récupère ses alias. Une fois cette liste récupérée, on peut identifier et annoter les alias présents dans l'article original avec l'information de classe NER de l'ancre originale. Cette augmentation est effectuée indépendamment de l'augmentation avec liens, donc uniquement sur les ancres.

On peut par cette méthode créer **seize** nouvelles entités nommées sur la page Wikipédia de Chilly Gonzales. On en illustre une sur la Figure 3.11.

[ORG Apple] adapted the tune for electric guitar .

Figure 3.11 – Illustration d'une entité nommée obtenue par augmentation par alias dans la phrase 56. Cette entité est obtenue à partir de l'alias "Apple" de l'ancre "Apple Inc ." contenue dans la phrase 54 de l'article.

3.5. Les mentions d'entités nommées

Certaines des entités récupérées par augmentation de données sont sujettes à des conflits dans le texte entre entités nommées, ce qui motive l'appellation *entités candidates*. Il faut de ce fait choisir quelle entité utiliser en cas de conflit. On traite de la résolution de ces conflits dans le chapitre 5.

Les entités obtenues dans le processus d'augmentation de données sont de ce fait toutes considérées comme des entités candidates jusqu'à résolution des conflits existants. Il est à noter que certaines de ces entités candidates peuvent être définies sur la même portion de texte dans Wikipédia, mais sont obtenues grâce à différentes méthodes. Ces dernières entités sont conservées dans un souci d'intégrité des données, cependant ces conflits sont facilement résolus.

On compte une très grande quantité de ces conflits dans les mentions d'entités nommées récoltées. Un total de 1,3G mentions sont concernées par un conflit sur la même portion de texte — soit 92,6 % des mentions — et 708M mentions sont concernées par des conflits entre mentions définies sur des portions de texte différentes — soit 47,1 % des mentions. Ce constat de la très grande quantité de mentions concernées par ces conflits montre la nécessité d'y apporter une solution. Ce sujet est abordé dans le chapitre 5.

Méthode	Ancres	Alias-Titre	Expansion du titre	Outlinks	Alias
Quantité	88 003 368	3 209 166	12 888 762	1 341 453 596	57 494 961

Tableau 3.2 – Quantité de mentions d'entités nommées obtenues pour chaque méthode décrite dans la section 3.4. On différencie les ancres contenues dans le dump Wikipédia original des entités candidates obtenues par augmentations de données.

On compte un total de **1 415 046 485 entités candidates** et **88 003 368 ancres** dans les mentions d'entités nommées sur le texte de Wikipédia. On peut voir sur le Tableau 3.2 que la méthode *outlinks* — suivre les liens — permet d'obtenir 94,7 % des mentions d'entités candidates nommées de Wikipédia, et 89,2 % de toutes les mentions d'entités nommées en incluant les ancres. Cette méthode constitue de ce fait la source principale de mentions d'entités nommées de cette approche.

Chapitre 4

Protocole d'évaluation

Dans ce chapitre, nous décrivons la méthodologie utilisée pour évaluer les méta-paramètres du procédé décrit dans le chapitre 3. Dans ce but, plusieurs modèles de reconnaissances d'entités nommées sont entraînés sur la ressource et évalués sur différents jeux de test.

On précise que ces entraînements n'incluent pas de données *in-domain* récupérées dans des benchmarks du domaine du NER. On entend par données *in-domain* les données de la portion d'entraînement d'un corpus de test donné. Cette méthode d'entraînement est sélectionnée pour ce mémoire car on souhaite construire une ressource "générique" qui peut s'adapter à différentes situations de test. De ce fait, on utilise uniquement les données issues de DataNER pour entraîner un modèle sans y ajouter la portion d'entraînement du corpus utilisé pour l'évaluation. On remarque bien sûr qu'il reste possible d'ajouter des données in-domain à DataNER lorsqu'elles sont disponibles.

4.1. SpaCy

4.1.1. Modèle

La librairie SpaCy [1] est utilisée dans ce mémoire afin d'entraîner des modèles de NER. Il s'agit d'un outil de NLP multilingue qui contient une variété de composantes pouvant effectuer des tâches du domaine, comme la tokénisation ou la prédiction de *partie de discours*. SpaCy modélise le NER comme un problème de *transition-based parser*, dont le concept est décrit dans un article de blog [2] qui décrit une implémentation de Goldberg and Nivre (2013).

Afin de réaliser cette tâche, l'outil utilise un modèle neuronal qui prend en entrée des représentations vectorielles construites à partir de *features* lexicales, d'embeddings statiques ainsi que de convolutions qui prennent en compte le contexte. L'outil utilise des fichiers de

1. `https://spacy.io/`
2. `https://explosion.ai/blog/parsing-english-in-python`

configuration pour paramétrer les entraînements et indiquer quels composants de la librairie doivent être utilisés dans le modèle. On partage de ce fait en annexe le fichier de configuration utilisé dans le cadre de ce mémoire, afin que ces entraînements puissent être reproduits.

Dans ce mémoire, on utilise l'outil SpaCy pour les expériences de calibrage décrites dans le chapitre 5 étant donné la rapidité d'entraînement de ses modèles de NER ainsi que pour le peu de variabilité dans ses performances entre plusieurs entraînements. En effet, étant donné la grande quantité de combinaisons à évaluer, ces deux caractéristiques sont primordiales au déroulé de ces expériences.

4.1.2. Entraînement

Pour entraîner les modèles de calibrage, on utilise un sous-ensemble (s-e) de 5000 articles sélectionnés aléatoirement. Ces articles sont annotés avec des ancres et des entités obtenues par augmentation de données. Ce sous-ensemble de 5000 articles contient **108 054 phrases**, **2 395 840 tokens** et **73 871 ancres**. Cet ensemble d'articles détient de ce fait un nombre de phrases et de tokens supérieur aux corpus populaires du domaine comme CoNLL03, OntoNotes, I2B2 ou WNUT, comme on peut le voir sur la Table 4.1.

Corpus	# Phrases	# Tokens
S-e	108 054	2 395 840
CoNLL03	17 291	254 983
OntoNotes	68 452	1 236 227
I2B2	60 616	598 608
WNUT	4403	78 463

Tableau 4.1 – Statistiques sur des corpus populaires du domaine du NER. On ne compte ici que le nombre de phrases des portions d'entraînement et de développement de ces corpus. Ce choix est dû à une volonté de comparaison avec le nombre de phrases du sous-ensemble décrit dans cette partie, duquel n'est extrait aucun corpus de test.

4.2. BiLSTM-CRF

4.2.1. Modèle

L'architecture BiLSTM-CRF (Huang et al., 2015; Lample et al., 2016) (voir Figure 4.1) est utilisée lorsqu'on souhaite entraîner des modèles sur de grandes quantités de données, comme dans le chapitre 6 dans lequel on utilise l'intégralité de DataNER ou WiNER comme matériel d'entraînement. Ce travail de mémoire récupère le code de Ghaddar and Langlais (2017), disponible sur un dépôt GitHub public [3] et l'utilise avec les features décrites dans (Ghaddar and Langlais, 2018) :

3. `https://github.com/ghaddarAbs/NER-with-LS`

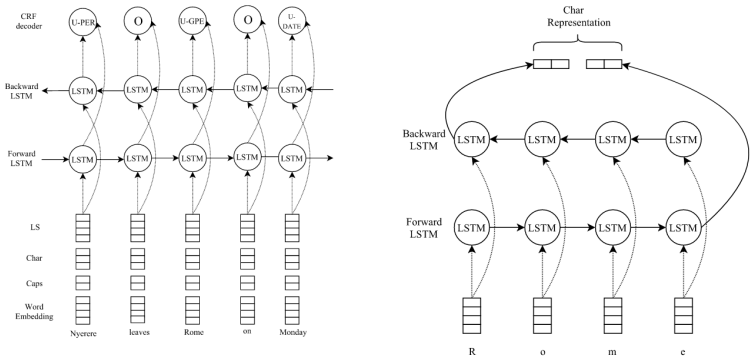


Figure 4.1 – (Gauche) Représentation de l'architecture BiLSTM-CRF (Droite) Représentation de la construction de l'embedding de caractères du mot "Rome". Ces deux images sont issues de l'article (Ghaddar and Langlais, 2018)

— des embeddings statiques. Ce sont les embeddings SSKIP (Ling et al., 2015) qui ont été choisis par les auteurs suite à des expérimentations. Il s'agit d'embeddings sensibles à la casse de dimension 100 entraînés par un modèle n-skip-gram sur 42G de tokens,
— des embeddings de caractères entraînés avec un modèle BiLSTM (Ma and Hovy, 2016a). Ceux-ci permettent de développer une représentation des mots à partir de leurs caractères. Ces features sont introduits pour ce modèle dans (Lample et al., 2016),
— des features de capitalisation qui permettent de caractériser la casse des mots considérés, comme décrits par Lê (2019b),
— des embeddings dits de "Similarité Lexicale" (LS), il s'agit de la contribution principale de Ghaddar and Langlais (2018). Ces embeddings pré-entraînés avant l'entraînement du modèle permettent d'encoder la similarité de chaque mot du vocabulaire avec des classes NER prédéterminées.

4.2.2. Entraînement

Les embeddings LS sont obtenus à partir d'un modèle FastText entraîné sur la totalité de la ressource. On utilise ensuite une sélection aléatoire de 100 000 phrases comme matériel d'entraînement pour le modèle BiLSTM-CRF. Ces phrases sont retirées des données fournies au modèle FastText. On choisit ce nombre de phrases car on peut ainsi avoir un corpus qui tient en mémoire et qui est plus grand que plusieurs des corpus de NER les plus connus (voir Tableau 4.1).

Étant donné un corpus d'entraînement, on entraîne cinq modèles différents sur les mêmes données afin d'obtenir une moyenne des performances du modèle considérant que les données fournies au modèle BiLSTM-CRF sont ordonnées aléatoirement dans le jeu de données.

Les données fournies au modèle FastText ne sont mélangées qu'une fois avant les cinq entraînements des modèles BiLSTM-CRF, étant donné qu'on garde les mêmes embeddings LS pour tous ces entraînements. Entraîner cinq modèles permet d'effectuer une moyenne des performances de chaque modèle étant donné la variabilité des performances liées à l'aspect stochastique de l'entraînement.

Les hyperparamètres par défaut fixés dans le code du modèle sont conservés.

4.3. Données de test

Les modèles obtenus sont par la suite évalués sur quatre corpus de test populaires dans la littérature.

- **CoNLL03** est un corpus créé pour la conférence CoNLL de 2003 (Tjong Kim Sang and De Meulder, 2003) en utilisant des articles de l'agence de presse Reuters. Il s'agit d'un corpus très étudié dans la littérature dont le jeu de quatre classes NER est très répandu : LOC, PER, ORG et MISC.
- **OntoNotes** est un corpus, dont la dernière version 5.0 — celle utilisée dans ce mémoire — a été introduite lors de la conférence CoNLL12 (Pradhan et al., 2012). Le corpus est construit à partir d'une variété de sources textuelles : articles de presse, articles de magazine, nouvelles télévisées, conversations télévisuelles, données issues du web et données de conversations orales. Ces données sont annotées avec un jeu de 18 classes NER.
- **WikiGold** est un corpus créé par Balasuriya et al. (2009) en annotant manuellement 149 articles Wikipédia anglais issus du dump du 22 mai 2008. Le corpus est annoté avec les classes LOC, PER, ORG et MISC.
- Le corpus de **Pages Web** de Ratinov and Roth (2009). Il s'agit d'un corpus de 20 pages web annotées manuellement. Ces pages sont des pages d'accueil de conférences d'informatique, des pages académiques ainsi que des pages personnelles. Le schéma d'annotation utilise le jeu de classe de CoNLL03 : LOC, PER, ORG et MISC.

Il est à préciser que contrairement aux corpus CoNLL03 et OntoNotes, les corpus WikiGold et Pages Web ne sont pas divisés en portions d'entraînement/développement/test. On utilise de ce fait l'intégralité de ces deux corpus pour évaluer des prédictions de modèle.

Tous ces corpus sauf OntoNotes utilisent le jeu de classes LOC, PER, ORG et MISC. On mappe les classes des entités de OntoNotes vers le jeu de classes de CoNLL03 afin de

pouvoir l'utiliser comme corpus de test pour la sortie du modèle utilisé. Pour ce faire, on suit la catégorisation des classes décrite dans (Nothman et al., 2013) [4] [5].

4.3.1. Statistiques

On inspecte ici des statistiques de bases sur les différents corpus de test utilisés. On peut constater en table 4.2 que les corpus WikiGold et Pages Web sont relativement plus petits que CoNLL03 et OntoNotes, avec une petite taille marquée du corpus Pages Web. C'est le corpus OntoNotes qui contient le plus de tokens.

Corpus	# Phrases	# Tokens	# Mentions
CoNLL03	3453	46 435	5648
OntoNotes	8262	152 728	7545
WikiGold	1696	39 007	3558
Pages Web	617	7545	781

Tableau 4.2 – Statistiques sur les corpus de test utilisés. Dans les titres des colonnes, le symbole "#" signifie "nombre de". Dans le cas du corpus OntoNotes, on s'intéresse ici à la version mappée du corpus.

Corpus	% LOC	% PER	% ORG	% MISC
CoNLL03	29,5	28,6	29,4	12,4
OntoNotes	33,9	26,3	23,8	16,0
WikiGold	28,5	26,3	25,2	20,0
Pages Web	12,2	35,3	28,6	23,9

Tableau 4.3 – Distribution des classes NER parmi les entités nommées contenues dans les corpus de test. Les statistiques sont exprimées sous forme de pourcentage.

Par ailleurs, on remarque en table 4.3 que la classe MISC est généralement la moins répandue dans ces corpus, avec une proportion bien moindre que les autres classes, à l'exception du corpus Pages Web qui contient plus de MISC que de LOC. C'est généralement la classe LOC qui est la plus répandue dans ces corpus.

4.4. Métriques utilisées

Étant donné que l'on utilise quatre corpus de test, on souhaite obtenir pour un modèle donné quatre scores de F-mesure décrivant les performances du modèle sur chacun de ces

4. L'article donne une table de correspondances entre différentes classes par niveau de finesse. Les données — anciennement disponibles à `http://schwa.org/resources` — sont maintenant disponibles à `https://figshare.com/articles/dataset/Learning_multilingual_named_entity_recognition_from_Wikipedia/5462500`.

5. Pour des détails sur le mapping, voir l'annexe A.1

corpus ainsi qu'une moyenne de ces scores afin de quantifier les performances du modèle en général.

Pour ce faire, on utilise le script *conlleval*[6] pour récupérer une valeur de F-mesure qui évalue les performances du modèle pour toutes les classes sur un corpus donné. Le script permet de calculer les valeurs de précision, rappel et F-mesure par classe et en moyenne (voir Figure 4.2). Ce sont les valeurs de F-mesure calculées avec *conlleval* que nous rapportons dans ce mémoire.

```
processed 46435 tokens with 5648 phrases; found: 743 phrases; correct: 625.
accuracy:  85.00%; precision:  84.12%; recall:  11.07%; FB1:  19.56
              LOC: precision:  74.38%; recall:   7.13%; FB1:  13.02  160
             MISC: precision:  36.84%; recall:   2.99%; FB1:   5.53  57
              ORG: precision:  83.24%; recall:   8.67%; FB1:  15.70  173
              PER: precision:  96.60%; recall:  21.09%; FB1:  34.62  353
```

Figure 4.2 – Exemple de sortie du script *conlleval* sur le corpus de test de CoNLL03. On peut y voir les valeurs de précision, rappel et F-mesure (FB1) par classe, ainsi qu'en moyenne sur la deuxième ligne. C'est la valeur de F-mesure qui se trouve à la fin de cette deuxième ligne (en rouge) qui est celle que l'on veut garder dans ce mémoire. Il s'agit de la macro F-mesure à correspondance exacte.

6. https://www.clips.uantwerpen.be/conll2000/chunking/conlleval.txt

Chapitre 5

Calibrage du processus de création de DataNER

Dans ce chapitre, on inspecte le calibrage de DataNER. En effet, on dispose d'un grand nombre d'entités nommées issues des ancres de Wikipédia et de procédés d'augmentation de données. On cherche ici à identifier les meilleures pratiques pour obtenir une ressource de qualité. Étant donné la grande quantité d'entités ainsi que la diversité des méthodes avec lesquelles on les obtient, il est en effet pertinent de questionner quelles sont les meilleures pratiques de sélection des entités candidates au vu des performances d'un modèle entraîné sur les différentes versions possibles de DataNER.

Dans cette exploration des meilleures pratiques, on utilise 5000 articles tirés aléatoirement de Wikipédia. Ces 5000 articles sont les mêmes tout le long de l'expérience de calibrage.

5.1. Le procédé de calibrage

Afin d'obtenir un corpus de NER annoté, il faut traiter toutes les entités nommées dont nous disposons. En effet, celles-ci correspondent à des *ancres* issues de Wikipédia ou à des entités obtenues par un procédé d'augmentation de données, auxquelles on fait référence par le terme *entités candidates*. On utilise pour ce faire le modèle SpaCy décrit précédemment.

Ce processus de choix des entités candidates se fait en plusieurs étapes :

(1) on choisit quelles entités candidates doivent être conservées,

(2) on applique certaines règles pour affiner la sélection,

(3) on effectue des choix entre les entités restantes qui se superposent dans le texte

Ces trois étapes constituent le *pipeline* d'extraction qui permet de construire DataNER. Ces différents choix d'utilisation des entités candidates constituent les *configurations* que l'on utilise. On distingue différentes valeurs de configuration pour chacun de ces paramètres. Le détail de ces configurations peut être inspecté en annexe.

Dans cette partie, on inspecte les façons dont on peut calibrer ce pipeline.

5.1.1. Sélection des entités candidates

La première partie du pipeline consiste à sélectionner les entités candidates à conserver. On effectue ce choix selon les méthodes d'augmentation de données qui ont créé ces entités — s'il s'agit d'une ancre présente dans le dump Wikipédia, si l'entité a été récupérée en suivant un lien... on prend pour exemple sur la Figure 5.1 une phrase issue d'une version de Wikipédia utilisant une configuration qui n'extrait pas les entités obtenues par la technique d'alias du titre. Deux entités candidates ne sont pas extraites de la phrase avec cette sélection, résolvant par ailleurs un conflit entre deux entités candidates. Après cette sélection, deux conflits sont toujours dans la phrase, qui seront résolus plus tard dans le pipeline.

[PER Gonzales] broadcasts a [MISC web series] " [MISC Pop [MISC Music]] Masterclass " on WDR , the documentary " Classical Connections " on [ORG [ORG BBC] Radio 1] , " The History of [MISC Music] " on Arte , and " [MISC Music] 's Cool with [PER Chilly [PER Gonzales]] " on [ORG [ORG Apple] [MISC Music]] 's Beats1 radio show .

⇓

Gonzales broadcasts a [MISC web series] " [MISC Pop [MISC Music]] Masterclass " on WDR , the documentary " Classical Connections " on [ORG [ORG BBC] Radio 1] , " The History of [MISC Music] " on Arte , and " [MISC Music] 's Cool with [PER Chilly Gonzales] " on [ORG [ORG Apple] [MISC Music]] 's Beats1 radio show .

Figure 5.1 – Exemple du procédé de sélection sur la phrase 4 de l'article Wikipédia de Chilly Gonzales. On sélectionne ici toutes les techniques sauf "alias-title", ce qui a pour effet de supprimer deux entités candidates.

Par ailleurs, dans le cas des entités obtenues en suivant des liens contenus dans Wikipédia et celles obtenues par alias, on inspecte également comment les utiliser dans l'article. En effet, on veut inspecter sur quelle portion du texte utiliser ces entités candidates obtenues par augmentation de données : dans tout l'article, ou uniquement dans la phrase depuis laquelle elles ont été récupérées. Effectuer cet examen permet d'inspecter si ces entités sont plus fiables lorsqu'elles sont proches de l'entité de laquelle elles sont issues.

5.1.2. Règle de filtrage

Ici on inspecte les combinaisons possibles pour les valeurs prises lors de la configuration des règles appliquées aux entités sélectionnées à l'étape de sélection. Il s'agit du deuxième composant du pipeline d'extraction, qui applique une heuristique à évaluer.

Précisément, on s'intéresse à une règle : si une entité candidate et une entité issue d'une ancre se superposent, on choisit toujours l'ancre. Cette règle traduit la plus grande confiance dans les entités nommées issues des ancres que celles issues de l'augmentation de données.

On l'illustre sur la Figure 5.2 avec une phrase dont la qualité des étiquettes est améliorée par la règle. Après l'application de cette règle, la phrase de l'exemple ne contient plus de conflits.

Gonzales played in the songs " [MISC Give Life Back to [MISC Music]] " and " Within " on [ORG Daft Punk] ' s fourth studio [MISC album] , " [MISC Random Access Memories] " , which won a [MISC Grammy Award for [MISC Album] of the Year] .

⇓

Gonzales played in the songs " [MISC Give Life Back to Music] " and " Within " on [ORG Daft Punk] ' s fourth studio [MISC album] , " [MISC Random Access Memories] " , which won a [MISC Grammy Award for Album of the Year] .

Figure 5.2 – Exemple du procédé de règle de filtrage sur la phrase 43 de l'article Wikipédia de Chilly Gonzales. Cette règle a pour effet de supprimer deux entités candidates de la phrase. En effet, ces deux entités candidates sont incluses dans les ancres "Give Life Back to Music" et "Grammy Award for Album of the Year".

Afin d'évaluer cette méthode, on considère deux choix : l'utiliser et ne pas l'utiliser.

5.1.3. Conflits

La dernière étape du processus de calibrage est la sélection des entités qui sont en conflit : étant donné une paire d'entités candidates qui se superposent dans le texte, on veut faire un choix entre les deux. En effet, même après l'application des règles de l'étape précédente, il est possible que des entités candidates obtenues lors de l'augmentation de données se superposent toujours. Il faut de ce fait choisir une entité au sein de ces paires d'entités superposées. Plusieurs méthodes sont possibles pour effectuer cette tâche, cependant on choisit d'effectuer ce choix de façon aléatoire. En effet, on choisit de limiter le nombre de combinaisons sur ce paramètre et on ne garde pour celui-ci qu'une seule valeur. Ce choix est également motivé par la difficulté d'établir une règle générale quant à la meilleure pratique de résolution de conflit étant donné le grand nombre de cas de conflits différents présents dans les données [1].

On illustre ce procédé sur la Figure 5.3 dont la phrase ne contient plus aucun conflit après application de cette méthode.

5.2. Expérimenter avec le calibrage

Afin d'identifier les meilleures pratiques pour calibrer DataNER, on utilise les 5000 articles sélectionnées aléatoirement dans Wikipédia et on en extrait des corpus différents selon

1. On a également expérimenté avec d'autres méthodes — choisir l'entité candidate la plus longue, ou la plus courte notamment — sans succès, et nous avons de ce fait conservé la méthode de résolution aléatoire.

Gonzales broadcasts a [MISC web series] " [MISC Pop [MISC Music]] Masterclass " on WDR , the documentary " Classical Connections " on [ORG [ORG BBC] Radio 1] , " The History of [MISC Music] " on Arte , and " [MISC Music] 's Cool with [PER Chilly Gonzales] " on [ORG [ORG Apple] [MISC Music]] 's Beats1 radio show .

⇓

Gonzales broadcasts a [MISC web series] " [MISC Pop Music] Masterclass " on WDR , the documentary " Classical Connections " on [ORG BBC] Radio 1 , " The History of [MISC Music] " on Arte , and " [MISC Music] 's Cool with [PER Chilly Gonzales] " on [ORG Apple Music] 's Beats1 radio show .

Figure 5.3 – Illustration du procédé de résolution de conflit par sélection aléatoire sur la phrase 4 de l'article Wikipédia de Chilly Gonzales. Le choix aléatoire entre mentions en conflit permet de supprimer les derniers conflits présents dans la phrase. Étant donné l'aspect aléatoire de la méthode, on peut constater des erreurs — "BBC" au lieu de "BBC Radio 1".

les configurations que l'on souhaite évaluer. On cherche alors à comparer les résultats du modèle SpaCy décrit dans le chapitre 4 afin d'identifier les configurations qui amènent aux meilleurs scores sur les quatre corpus de test. On entraîne et on évalue de ce fait un modèle sur chaque version du corpus : la version avec les ancres uniquement et toutes les configurations pour sélectionner les entités candidates. On décompte pour chaque partie du pipeline [2] :

(1) 36 combinaisons de sélection d'entités candidates,

(2) 2 combinaisons de filtrage par règle,

(3) 1 combinaison de résolution de conflits.

On dénombre de ce fait $36 \times 2 \times 1 = 72$ configurations différentes du pipeline d'extraction de DataNER. Il faut de ce fait entraîner 72 modèles, chacun correspondant à une configuration possible du pipeline d'extraction défini dans ce chapitre.

Précisément, on cherche à identifier les configurations qui amènent aux meilleures F-mesure moyennes sur les quatre corpus de test. Ce sont ces configurations qui seront conservées après les expériences de calibrage de DataNER. Étant donné la grande quantité de configurations que cette expérience utilise, la totalité des résultats ne sera décrite qu'en annexe.

5.3. Résultats du calibrage

On rapporte maintenant les résultats des expériences de calibrage.

2. Voir cette annexe pour inspecter le détail des combinaisons dont il est ici question.

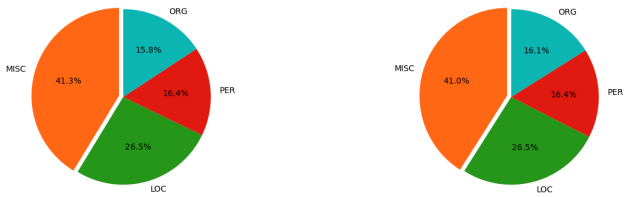
5.3.1. DataNER de base : utiliser seulement les ancres

Dans un premier temps, on inspecte les performances de la configuration du pipeline qui n'utilise que les ancres contenues dans le dump Wikipédia original.

Cette version des 5000 articles sélectionnés aléatoirement contient 73 871 entités nommées et 2 395 840 tokens. Ces entités sont définies sur un total de 156 798 tokens, ce qui signifie que cette version de ce sous-ensemble de DataNER détient une couverture — soit la proportion de tokens annotés sur le total de tokens du corpus — de **6,5 %**, très similaire à celle de la ressource intégrale, de 6,6 %.

On peut par ailleurs constater sur la figure 5.4 la similarité entre la distribution des classes NER dans DataNER dans son intégralité et dans le sous-ensemble de 5000 articles utilisés pour le calibrage.

Ces deux constats laissent présager que ce sous-ensemble est représentatif de la ressource.



(a) (b)

Figure 5.4 – (a) Distribution des classes NER parmi les entités nommées contenues dans le corpus de 5000 articles extrait uniquement avec les ancres. (b) Distribution des classes NER parmi les ancres contenues dans l'intégralité de la ressource DataNER.

On peut consulter les performances du modèle entraîné sur ces données sur la Tableau 5.1. Ces résultats constituent une base que les modèles utilisant des données d'entraînement avec augmentation de données doivent dépasser.

	CoNLL03	OntoNotes	WikiGold	Pages Web	*AVG*
Configuration ancres	19,56	17,39	33,29	17,74	21,99

Tableau 5.1 – F-mesures du modèle SpaCy NER entraîné sur la configuration avec ancres uniquement et évalué sur les quatre corpus de test.

5.3.2. Les variantes de DataNER

On inspecte maintenant les résultats des différentes configurations qui ont été expérimentées. Dans la suite de ce chapitre, on nommera les configurations selon la première lettre

de chacune de ses valeurs de configuration. On donne des exemples de nommage dans le Tableau 5.2.

alias-titre	exp-titre	outlinks	alias	heuristique	Nommage
OUI	OUI	article	article	OUI	O-O-A-A-O
NON	OUI	phrase	article	NON	N-O-P-A-N
OUI	NON	NON	NON	OUI	O-N-N-N-O

Tableau 5.2 – Exemples de configurations utilisées dans l'expérience de calibrage de DataNER et leur nommage dans ce chapitre.

La Table 5.3 décrit les résultats des modèles entraînés sur les cinq meilleures configurations, les trois pires et sur celle qui n'utilise que les ancres.

Configuration	CoNLL03	OntoNotes	WikiGold	Pages Web	*AVG*
ancres	19,56	17,39	33,29	17,74	21,99
O-O-A-A-O	**53,97**	**47,49**	**48,48**	**38,45**	**47,09**
N-O-A-A-O	52,10	43,36	47,03	34,80	44,32
O-O-A-P-O	48,42	43,65	47,31	35,11	43,62
N-N-A-A-O	50,22	45,05	45,61	32,41	43,32
O-N-A-A-O	49,61	43,76	47,28	30,98	42,90
N-N-N-P-N	22,89	23,26	31,98	20,89	24,75
O-N-N-N-N	19,85	16,65	32,58	19,78	22,21
O-N-N-N-O	19,93	17,76	32,84	16,65	21,79

Tableau 5.3 – Valeurs de F-mesure pour différentes configurations de DataNER. On liste d'abord les performances de la configuration avec les ancres uniquement, puis les cinq meilleures et les trois pires.

On peut commencer par constater qu'une même configuration a obtenu les meilleures performances sur tous les corpus de test avec une F-mesure moyenne de 47,09. Il s'agit de la configuration qui utilise les deux techniques d'augmentation du titre et qui propage les entités obtenues en suivant des liens et par alias dans tout l'article. Cette configuration permet d'obtenir une amélioration moyenne de 25,10 points en comparaison avec la version n'utilisant que les ancres (voir Tableau 5.3).

Cette version du sous-ensemble contient 179 468 entités nommées, soit 77 092 entités de plus que la version avec les ancres uniquement. On note par ailleurs une couverture de 12,4 %, soit une augmentation de 6,1 points. On voit sur la Figure 5.5 que les entités "MISC" constituent maintenant la majorité des entités nommées contenues dans le sous-ensemble.

Les quatre autres meilleures configurations ont comme point en commun la propagation systématique des entités obtenues par lien dans tout l'article, ainsi que la quasi systématique propagation dans tout l'article des entités obtenues par alias. Par ailleurs, les cinq meilleures configurations utilisent toutes l'heuristique de priorité des ancres et toutes sauf une utilisent au moins une forme d'augmentation de données utilisant le titre.

On peut établir de ce fait que l'intuition de l'heuristique sur la priorité donnée aux ancres semble être bonne : les ancres constituent des entités nommées de confiance, et il semblerait que garder — propager — le plus d'entités nommées possibles amène aux meilleures performances.

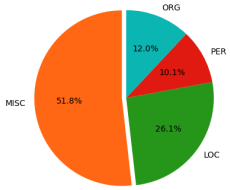


Figure 5.5 – Distribution des classes NER parmi les entités nommées contenues dans la version des 5000 articles qui permet d'entraîner le meilleur modèle. Il s'agit de la configuration O-O-A-A-O.

Les trois pires configurations n'utilisent pas les valeurs qui semblent profiter aux meilleures configurations. Malgré l'utilisation de l'augmentation de données par alias du titre, deux des pires configurations n'utilisent pas l'heuristique de priorité donnée aux ancres et les trois utilisent peu de ces paramètres, avec beaucoup de valeurs "NON". On remarque qu'aucune des trois configurations n'utilisent d'entités issues de l'augmentation par lien ni d'entités issues de l'expansion du titre.

On inspecte par ailleurs des statistiques sur les différentes configurations mises en valeur dans cette section sur la Table 5.4. On peut y voir que les baisses de performance correspondent généralement à une baisse du nombre d'entités nommées. De cette façon, la meilleure configuration — O-O-A-A-O — dispose du plus grand nombre d'entités nommées annotées. Cependant, on peut également voir que la configuration O-O-A-P-O performe mieux que la configuration O-N-A-A-O malgré que la première dispose de 7 139 entités nommées de moins que la deuxième, ce qui suggère que le nombre d'entités nommées n'est pas le seul élément qui impacte les performances de chaque configuration.

5.3.3. Évaluer les paramètres

On peut considérer cinq paramètres à évaluer dans ces configurations : alias-titre, expansion-titre, outlinks, alias et l'heuristique sur la priorité des ancres. Ces paramètres sont les différents champs des configurations que l'on peut inspecter dans l'annexe. Afin de comprendre leurs effets sur les performances, on inspecte les moyennes des performances des groupes de configurations qui utilisent les différentes valeurs de chaque paramètre.

5.3.3.1. Augmentation par le titre. On inspecte ici à la fois l'augmentation par expansion du titre et par alias du titre. Pour ce faire, on regarde l'effet de ces deux paramètres en

Configuration	# Annotations	# Tokens annotés	Couverture	F-mesure moyenne
ancres	73 871	156 798	6.3 %	21.99
O-O-A-A-O	179 468	300 084	12.5 %	47.09
N-O-A-A-O	178 097	297 555	12.4 %	44.32
O-O-A-P-O	169 842	288 632	12.0 %	43.62
N-N-A-A-O	175 515	290 865	12.1 %	43.32
O-N-A-A-O	176 981	293 825	12.2 %	42.90
N-N-N-P-N	88 399	173 406	7.2 %	24.75
O-N-N-N-N	76 408	161 243	6.7 %	22.21
O-N-N-N-O	76 410	161 332	6.7 %	21.79

Tableau 5.4 – Statistiques sur différentes configurations dont les performances ont été inspectées. La colonne "Couverture" fait ici référence à la couverture de tokens annotés dans le corpus. La colonne "F-mesure moyenne" indique la moyenne des F-mesure sur les quatre corpus de test pour cette configuration.

comparant les performances des configurations qui l'utilisent avec les configurations correspondantes qui ne l'utilisent pas. Pour ce faire, on effectue une moyenne des performances de chaque combinaison selon ces deux paramètres.

alias-titre	expansion-titre	F-mesure moyenne	# LOC	# PER	# ORG	# Entités
NON	NON	32,04	31 309	14 042	14 786	118 167
OUI	NON	31,28	31 562	14 414	15 127	120 061
NON	OUI	34,24	32 544	15 478	15 566	124 908
OUI	OUI	35,21	32 777	15 783	15 865	126 537

Tableau 5.5 – Moyenne des performances des configurations de l'expérience selon les paramètres d'augmentation par titre. Les trois colonnes de droite indiquent la moyenne des quantités d'entités de chaque classe (LOC, PER et ORG) pour chaque type de configuration. La dernière colonne indique également le nombre total moyen d'entités dans chaque groupe de configurations.

La Table 5.5 nous indique que les combinaisons que l'on peut effectuer avec ces paramètres ont un impact assez clair sur les performances. On peut constater que conserver toutes les entités obtenues de cette façon permet d'augmenter les performances moyennes de 3,17 points, ce qui milite pour le fait de systématiquement garder toutes ces entités. Cependant, on peut également noter que n'utiliser que les entités obtenues en utilisant des alias du titre amène à des baisses de performance en comparaison aux configurations qui n'en utilisent pas. Une inspection des données indique que l'utilisation des alias du titre produit l'apparition de nombreux conflits résolus aléatoirement qui peuvent introduire du bruit. En effet, de nombreux alias sont des versions plus courtes du titre, ce qui produit un plus grand nombre de conflits. La Figure 5.6 illustre ce phénomène avec une entité mal identifiée à cause d'un conflit entre une entité candidate issue d'un alias du titre et une entité candidate issue de liens.

[PER Pablo González Velázquez] (1664 - 1727) was a Spanish [MISC Baroque] sculptor .
(a) Configuration N-N-A-A-O

[PER Pablo González] Velázquez (1664 - 1727) was a Spanish [MISC Baroque] sculptor .
(b) Configuration O-N-A-A-O

Figure 5.6 – Comparaison de la première phrase de l'article Wikipédia de Pablo González Velázquez entre (a) une configuration qui n'utilise pas les alias du titre et (b) une configuration qui les utilise sans l'expansion du titre. La configuration (a) obtient de meilleures performances que la configuration (b). On peut voir que la configuration (b) identifie mal la première entité.

Le fait que le groupe de combinaisons qui utilisent à la fois les alias du titre et l'expansion du titre obtient de meilleurs résultats suggère que l'augmentation des conflits de ces alias avec le "titre correct" augmente les chances de produire des résolutions aléatoires correctes en augmentant le nombre d'entités candidates. Ce dernier constat suggère une utilisation conjointe de ces deux techniques.

On finit par remarquer que la croissance des performances moyennes semble être lié à la croissance du nombre d'entités total, excepté pour les configurations qui utilisent la combinaison OUI-NON. Ce dernier constat continue de singulariser ce groupe de configurations. Par ailleurs, la proportion de chaque classe dans les configurations reste stable à travers les différentes combinaisons.

5.3.3.2. Augmentation avec liens et alias. On regarde ici conjointement les effets des paramètres outlinks et alias dans le processus de calibrage de DataNER. Ces deux paramètres peuvent chacun avoir trois valeurs, ce qui amène à neuf combinaisons différentes en les considérant ensemble. On commence par inspecter les moyennes des performances des configurations en regardant ces neuf combinaisons.

outlinks	alias	F-mesure moyenne	# LOC	# PER	# ORG	# Entités
article	article	42.53	45 768	16 961	19 123	177 752
article	phrase	40.78	42 523	16 146	16 623	168 108
article	NON	39.63	42 230	16 049	15 959	165 813
phrase	article	31.66	31 396	15 157	16 697	112 968
phrase	phrase	30.53	28 865	14 430	14 186	107 483
phrase	NON	28.61	26 516	14 014	12 897	96 124
NON	article	27.89	26 328	14 493	16 237	98 815
NON	phrase	30.29	24 008	13 834	13 837	94 273
NON	NON	25.05	20 798	13 278	12 465	80 426

Tableau 5.6 – Moyenne des performances des configurations utilisant ces combinaisons pour les paramètres outlinks et alias. On indique également le nombre d'entités total et par classes.

On constate qu'on obtient les meilleures performances lorsque l'on utilise les deux techniques en même temps et qu'on les propage dans tout l'article. On peut voir sur le Tableau 5.6 que l'augmentation des performances correspond en général à l'augmentation du nombre d'entités dans la configuration. On note une seule exception au phénomène entre le groupe de configurations NON-article et NON-phrase qui singularise à nouveau l'utilisation seule des alias dans la création d'entités nommées.

On peut voir que les performances sont fortement impactées par l'utilisation des entités obtenues par lien uniquement sur leur phrase d'origine. On constate dans ce cas-là une différence de scores moyen de 10,71 points, ce qui suggère une propagation systématique de ces entités dans tout l'article. Cela montre également l'importance de ce paramètre dans le processus d'extraction.

On peut par ailleurs voir que propager les alias dans tout l'article au lieu de simplement les propager dans leur phrase d'origine permet en général d'obtenir de meilleures performances. Ce qui suggère que propager les alias dans tout l'article soit une meilleure pratique.

Ce sont les combinaisons qui propagent les deux types d'entités dans tout l'article qui permet d'obtenir les meilleures configurations et il convient de ce fait d'utiliser cette combinaison.

5.3.3.3. La priorité donnée aux ancres. On commence par inspecter la moyenne des performances des modèles entraînés sur les données utilisant l'heuristique sur les ancres et les performances de ceux qui ne l'utilisent pas.

Ce faisant, on veut inspecter l'effet de cette heuristique sur les performances des modèles.

	F-mesure moyenne	# LOC	# PER	# ORG	# Entités
avec	34.67	31 802	14 947	15 645	122 221
sans	31.77	31 854	14 932	15 426	122 180

Tableau 5.7 – Moyenne des scores sur les quatre corpus de test et en moyenne des modèles entraînés sur toutes les configurations avec et sans l'heuristique de priorité aux ancres. On indique également le nombre d'entités par classe et au total pour chaque groupe de configurations.

On peut constater sur la Table 5.7 que l'heuristique permet d'obtenir en moyenne une amélioration des résultats toutes configurations confondues. Exactement, on note une augmentation moyenne du score de **+2,9** points. Par ailleurs, le nombre d'entités est plus grand en moyenne parmi les configurations qui utilisent l'heuristique, cependant la différence moyenne est minime : 41 entités nommées en moyenne. On peut de ce fait avancer que l'heuristique augmente la qualité des entités nommées contenues dans les corpus produits.

L'heuristique amène à une baisse de performances moyenne dans 5 configurations, pouvant aller jusqu'à une baisse de 2,78 points (voir Tableau 5.8). Cet ensemble de configurations contient les trois configurations n'utilisant que l'augmentation par les deux techniques

Configuration	F-mesure moyenne	# LOC	# PER	# ORG	# Entités
N-O-N-A-O	29.08	27 206	15 263	16 670	102 767
N-O-P-N-O	27.00	26 659	14 707	13 765	98 941
O-O-N-N-O	25.92	21 957	14 380	13 238	86 673
N-O-N-N-O	25.49	21 665	14 056	12 911	84 759
O-N-N-N-O	21.79	19 962	12 563	12 071	76 411

Tableau 5.8 – Configurations où l'utilisation de l'heuristique de priorité aux ancres amène à une baisse de score.

de titre. On peut de ce fait constater qu'utiliser les entités obtenues par augmentation du titre amène à des baisses de performance quand on utilise l'heuristique et que l'on n'utilise pas également les autres méthodes d'augmentations. Les deux autres configurations qui enregistrent une baisse de performances avec l'heuristique n'utilisent également pas tous les paramètres d'augmentation, ce qui laisse supposer que cette heuristique permet d'obtenir de meilleures performances quand on utilise le plus d'entités candidates possibles.

Il semble de ce fait que l'heuristique peut amener à une baisse de performance si elle est utilisée dans les configurations qui extraient moins d'entités candidates. Comme le montre le Tableau 5.8, les configurations qui ne bénéficient pas de l'heuristique ont en effet un nombre d'entités total plus faible que la moyenne des configurations utilisant l'heuristique (voir Tableau 5.7). On peut supposer que ce phénomène est dû au fait que le peu d'entités candidates que ces configurations extraient peut amener à une plus grande quantité d'entités correctement identifiées qui se superposent avec des ancres.

De la même façon, on suppose que les configurations qui extraient plus d'entités candidates disposent de suffisamment d'entités proprement identifiées qui ne sont pas superposées avec des ancres, et qui compensent cette potentielle perte, en plus d'améliorer le score en filtrant les faux positifs.

Les résultats montrent de manière générale que cette heuristique est un apport positif au pipeline d'extraction de DataNER.

5.4. Conclusion

On peut constater différents schémas dans la façon dont les performances des modèles entraînés sont impactées par les différentes configurations. En effet, en général utiliser tous les paramètres en même temps permet d'obtenir de meilleures performances sur les corpus de test. On remarque également des erreurs de segmentation des entités nommées dans les données liées à la résolution aléatoire des conflits des mentions d'entités nommées. Ce phénomène se remarque principalement dans les configurations qui extraient moins d'entités de la ressource.

Plus précisément, on constate que ce sont les configurations qui permettent d'extraire le plus d'entités nommées qui amènent à de meilleures performances. On peut constater ce phénomène en observant que la propagation des entités obtenues par lien et par alias dans tout l'article amène aux meilleures performances, ainsi qu'en observant que l'augmentation par titre amène aux meilleures performances en utilisant les deux techniques d'augmentation conjointement.

Les données montrent également certains phénomènes isolés propres à certaines combinaisons spécifiques, comme les meilleures performances obtenues en propageant les alias dans les phrases si on n'utilise pas les liens ou la baisse de performance provoquée par l'utilisation de la technique d'alias du titre seule.

Finalement, on remarque que les performances sont en moyenne meilleures lorsqu'on utilise l'heuristique qui conserve les ancres en cas de conflit avec une autre entité. Cette heuristique augmente les performances des configurations qui utilisent toutes les méthodes d'augmentation simultanément et doit de ce fait être conservée.

En conclusion, l'expérience de calibrage a permis d'identifier l'impact des différentes valeurs des paramètres des configurations. La meilleure configuration a également pu être identifiée — O-O-A-A-O — et sera conservée comme meilleure version de DataNER pour la suite de ce mémoire.

Chapitre 6

Comparaison de DataNER à d'autres ressources

Ce chapitre est dédié à la comparaison de DataNER à d'autres ressources similaires. Il s'agit de ressources également annotées par supervision distante qui sont décrites dans la suite de ce chapitre.

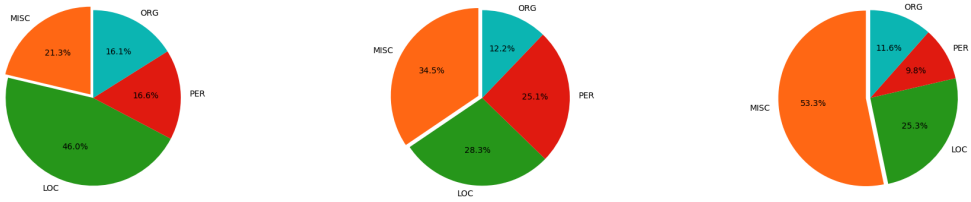
6.1. Les ressources

On décide de comparer DataNER à deux ressources : la ressource WiNER (Ghaddar and Langlais, 2017) et AnchorNER (Zhu et al., 2019). Ces deux ressources ont été construites par des procédés de supervision distante utilisant Wikipédia et des bases de connaissances — respectivement Freebase et DBpedia. Il est alors naturel de les comparer à DataNER étant donné leur proximité. C'est la configuration O-O-A-A-O de DataNER que l'on utilise dans cette partie, car c'est celle qui a obtenu les meilleures performances dans les expériences de calibrage.

	# Articles/Résumés	# Phrases	# Tokens	# Entités	Couverture
WiNER	3,2M	54M	1,3G	106M	15,20 %
AnchorNER	5,2M	12M	268M	31M	22,26 %
DataNER	6,2M	127M	2,8G	210M	12,57 %

Tableau 6.1 – Statistiques sur les ressources WiNER, AnchorNER et DataNER.

On peut voir sur la Figure 6.1 que les trois ressources ont des distributions de classes NER différentes. On note par exemple la très grande proportion de MISC dans DataNER, une classe moins importante dans les deux autres ressources. Ces deux autres ressources ont une distribution des classes plus équilibrée. Les classes ORG et PER sont systématiquement les moins fréquentes à travers toutes les ressources. Par ailleurs, le Tableau 6.1 montre que les trois ressources diffèrent entre elles en termes de taille et de proportion du matériel annoté (*couverture*), avec AnchorNER qui constitue la ressource la plus annotée pour sa taille.



(a) AnchorNER (b) WiNER (c) DataNER

Figure 6.1 – Distribution des classes NER parmi les entités nommées des trois ressources.

6.1.1. WiNER

La ressource WiNER est un corpus annoté avec des entités nommées réparties dans les classes LOC, PER, ORG et MISC. Cette ressource est construite à partir d'un dump du Wikipédia anglais de 2013, contenant 3,2M d'articles.

Les auteurs récupèrent tout le texte issu des articles et utilisent l'hypothèse déjà suivie par (Nothman et al., 2013; Al-Rfou et al., 2014) de transformer les *ancres* — le texte des liens — en entités nommées. Les auteurs effectuent ensuite une association entre chaque article Wikipédia et sa page Freebase, permettant d'extraire un type Freebase pour chaque article Wikipédia. Ces types permettent de classifier les articles dans les quatre classes NER utilisées et par le même procédé les liens contenus dans le texte vers ces articles. Ce procédé d'annotation permet de produire un corpus annoté avec des entités nommées.

Les auteurs définissent également des procédés d'augmentation de données utilisant la structure des liens de Wikipédia, ainsi qu'un système de coréférence (Ghaddar and Langlais, 2016) du concept principal des articles Wikipédia. Grâce à ces procédés, les auteurs créent WiNER, une ressource contenant 3,2M articles, 54M phrases, 1,3G tokens ainsi que 106M annotations d'entités nommées (voir Tableau 6.1). Comme on peut le voir sur la Figure 6.1, WiNER est la ressource avec les classes les plus équilibrées.

6.1.2. AnchorNER

AnchorNER est une ressource annotée avec les classes LOC, PER, ORG et MISC. Elle est constituée du texte des résumés — *abstracts* — des articles Wikipédia anglais d'un dump datant de 2017. Les auteurs utilisent également les ancres contenues dans le texte de Wikipédia, mais les classifie grâce à la base de connaissances DBpedia. Ils utilisent par la suite un *modèle de correction neuronale* qui leur permet d'y ajouter des entités nommées qui n'y étaient pas présentes. Ce modèle est entraîné grâce à la ressource manuellement annotée DocRED (Yao et al., 2019).

Ce procédé permet de créer une ressource NER contenant 5,2M résumés [1] d'articles Wikipédia, 12M de phrases, 268M tokens ainsi que 31M d'entités nommées. Cette ressource dispose d'une couverture des tokens de 22,26 %, plus importante que les deux autres ressources (voir Tableau 6.1). Par ailleurs, on constate sur la Figure 6.1 qu'AnchorNER n'a pas le même ordre de fréquence des classes que les deux autres ressources — MISC, LOC, PER, ORG — mais que la classe LOC y est plus fréquente que la classe MISC.

6.2. Comparaison

Afin de comparer ces trois ressources, on utilise le modèle BiLSTM-CRF décrit plus haut. Lors de cette comparaison, on utilise uniquement les ressources elle-mêmes comme matériel d'entraînement, aucun matériel d'entraînement issu de benchmarks de NER n'est utilisé.

Pour chaque ressource, on donne l'intégralité du corpus au modèle. On sélectionne aléatoirement 100 000 phrases dans ce corpus comme matériel d'entraînement et le reste de la ressource est utilisée pour produire les embeddings de similarité lexicale utilisés comme *features* pour ce modèle. On rapporte les résultats dans le Tableau 6.2.

	CoNLL03	OntoNotes	WikiGold	Pages Web	*AVG*
WiNER	**60,60**	49,54	53,05	**37,78**	49,31
min-max	59,32 - 61,85	48,85 - 50,49	51,88 - 53,81	36,51 - 39,14	
AnchorNER	56,30	**55,74**	**61,64**	37,66	**52,83**
min-max	55,55 - 57,04	55,38 - 56,00	59,98 - 62,83	35,10 - 39,32	
DataNER	52,57	40,46	41,13	27,17	40,33
min-max	51,25 - 53,99	38,49 - 41,94	37,39 - 42,99	24,28 - 28,57	

Tableau 6.2 – F-mesure moyennes des cinq modèles entraînés sur chaque ressource pour chaque corpus de test. Les lignes *min-max* indiquent pour chaque ressource et chaque corpus de test la F-mesure minimale et maximale.

On constate que les performances sont quasiment toujours les meilleures avec AnchorNER : sur tous les corpus de test sauf CoNLL03, sur lequel WiNER est la meilleure ressource. On peut constater que ces deux ressources obtiennent une F-mesure moyenne proche dans cette expérience, avec une différence de 2,53 point en faveur de AnchorNER. DataNER obtient cependant un score moyen beaucoup plus bas que ces deux autres ressources.

Étant donné les meilleures performances de WiNER et d'AnchorNER en comparaison avec DataNER, on choisit de continuer de comparer les performances de DataNER avec WiNER spécifiquement. Ce choix est effectué car l'approche suivie pour construire WiNER est plus proche de celle de DataNER étant donné l'utilisation de tout Wikipédia — et non pas juste les résumés d'article — ainsi que du fait qu'une ressource annotée manuellement — DocRED — est utilisée dans la création d'AnchorNER.

1. Les auteurs justifient l'utilisation des résumés uniquement en constatant que ceux-ci contiennent plus d'ancres, ce qui impacterait positivement les performances.

6.3. Analyse des résultats

Dans cette section, on tente d'expliquer les différences de performances constatées entre l'approche proposée dans ce mémoire et celle de WiNER (Ghaddar and Langlais, 2017). Dans le but de comprendre ces différences de performances (voir Tableau 6.2), on compare les procédés d'étiquetage des entités nommées dans chacune de ces deux ressources.

6.3.1. Performances de modèles

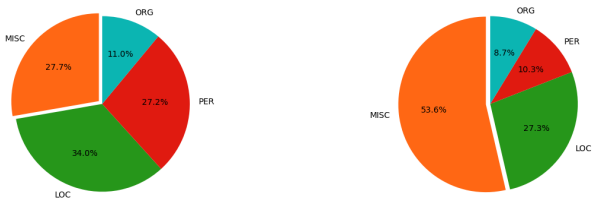
Dans le but de n'évaluer que cette partie du procédé de création des ressources, on produit deux versions du corpus WiNER :

— une version dont les classes d'entités nommées sont issues de Freebase,

— une version dont les classes d'entités nommées sont issues de WikiData.

On souhaite ensuite comparer les performances de deux modèles entraînés sur chaque version du corpus. Les résultats d'une telle expérience permettront de ce fait de ne considérer que l'impact de la méthode d'étiquetage sur les performances des modèles.

Afin de produire ces deux versions de WiNER, on en récupère le texte annoté et on en supprime toutes les entités nommées que l'on ne trouve pas dans WikiData. On produit ensuite deux versions du corpus : la version originale utilisant la méthode d'étiquetage de Freebase, et une nouvelle version qui étiquète les entités nommées avec l'outil NECKAr utilisant WikiData. On justifie ce choix d'utiliser le texte de WiNER car on peut facilement récupérer l'étiquetage effectué par Freebase en conservant les entités nommées originales de WiNER.



(a) (b)

Figure 6.2 – Distribution des classes NER parmi les entités nommées de (a) la version de WiNER avec les classes originales de WiNER (issues de Freebase) (b) la version de WiNER avec les classes issues de WikiData en utilisant NECKAr.

On peut voir sur la Figure 6.2 que les distributions des classes NER parmi les entités nommées de chaque version de WiNER suivent celles des deux ressources originales (voir Figure 6.1). On remarque de ce fait que la version de WiNER utilisant Freebase a une distribution des classes beaucoup plus équilibrée que celle utilisant WikiData, avec une majorité d'entités

nommées de classe MISC. On peut avancer qu'une distribution des classes équilibrée est un avantage ici étant donné que l'on considère des ressources sans un domaine d'application précis en tête qui peut définir la distribution réelle des classes parmi les entités nommées. On dénote par ailleurs 28M entités qui ont une classe différente entre les deux versions de ce corpus. Parmi ces entités, 23,4M sont étiquetées MISC par WikiData, ce qui continue de souligner le biais de cette méthode avec cette classe NER.

Ces deux versions de WiNER contiennent toutes les deux **78 795 841 entités nommées**, soit une perte de 41M entités nommées pour lesquelles aucune page WikiData n'a été trouvée.

Comparer les performances de modèles entraînés sur ces deux versions de WiNER permet de ce fait de ne comparer que la méthode d'étiquetage utilisée, ce dernier facteur étant la seule variable entre ces deux corpus.

	CoNLL03	OntoNotes	Pages Web	WikiGold	*AVG*
WiNER/WikiData	50,62	42,39	44,79	28,96	41,69
min-max	48,62 - 52,19	41,28 - 43,82	43,12 - 46,51	26,87 - 31,42	
WiNER/Freebase	**57,64**	**46,41**	**52,22**	**31,19**	**46,86**
min-max	56,25 - 58,74	45,02 - 47,19	51,35 - 53,02	28,77 - 32,72	

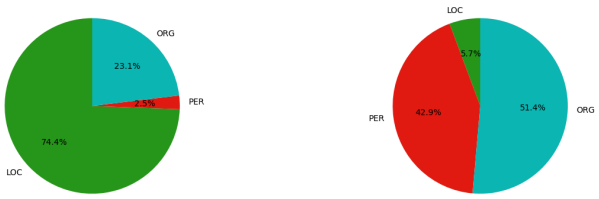
Tableau 6.3 – Comparaison des performances entre les deux versions de WiNER, avec supervision de Freebase ou WikiData

On peut constater sur le Tableau 6.3 que les performances sont systématiquement meilleures avec le modèle entraîné sur la version du corpus dont la classe des entités nommées est issue de Freebase, soit de WiNER. Cette comparaison montre que la méthode d'étiquetage de NECKAr utilisée dans ce mémoire ne permet pas d'obtenir d'aussi bons étiquetages des entités nommées que celle utilisée par Ghaddar and Langlais (2017) avec Freebase.

6.3.2. Comparaison manuelle

On continue de comparer WiNER et DataNER en inspectant des entités nommées pour lesquelles la classe NER étiquetée n'est pas la même dans les deux ressources.

Pour ce faire, on considère à nouveau les deux versions de WiNER précédemment décrites : une avec des entités dont les classes sont étiquetées avec Freebase, et une autre version utilisant WikiData à travers NECKAr. On compte un total de 78M entités nommées dans ces deux versions de WiNER, dont 28M ont une classe NER différente entre les deux versions de la ressource. Afin de continuer à comparer ces deux ressources, on choisit ici aléatoirement 1000 entités nommées dans cette version de WiNER, et on inspecte manuellement quelle est la meilleure classe parmi les deux choix. Pour effectuer cette expérience, on exclut les entités de classe MISC de la comparaison. En effet, on a remarqué au travers de ce mémoire que notre méthode de typage étiquète un trop grand nombre d'entités MISC qui semblent noyer les entités dans une mauvaise distribution des classes NER. On souhaite de ce fait comparer



(a) WikiData (b) Freebase

Figure 6.3 – Distribution des classes parmi les prédictions de chaque méthode de typage pour les 1000 entités sélectionnées aléatoirement.

WikiData et Freebase hors MISC pour considérer les capacités de l'outil utilisé à identifier les autres classes. La Figure 6.3 nous montre que la distribution des classes NER parmi les 1000 entités sélectionnées aléatoirement est très différente entre les deux méthodes de typage, avec par exemple une très grande représentation de la classe LOC avec NECKAr que l'on ne retrouve pas avec Freebase, qui ne prédit cette classe que dans 5,7 % des cas — soit 57 entités.

On distingue de ce fait trois possibilités pour chaque entité nommée évaluée : Freebase a raison, NECKAr a raison ou aucun des deux. Cette dernière option correspond aux entités nommées qui ne tombent dans aucune des classes proposées par ces versions de WiNER.

Ce faisant, on souhaite évaluer manuellement quelle méthode de typage amène aux meilleurs résultats, ainsi qu'identifier des tendances entre les deux approches.

	Freebase	WikiData
Quantité	677	288

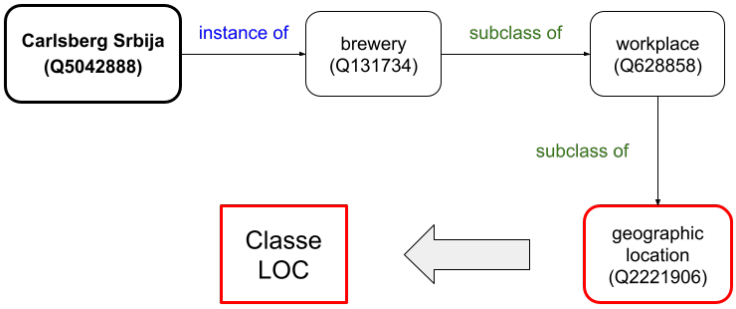
Tableau 6.4 – Quantité d'entités nommées correctement étiquetées par chaque approche.

On voit dans le Tableau 6.4 que l'approche utilisant Freebase est bien plus fiable que celle utilisant WikiData, avec 67,7 % des 1000 entités sélectionnées bien étiquetées par Freebase. La majorité de ces 1000 entités sont ainsi mieux classifiées avec Freebase. On note également 35 cas où aucune des deux approches ne permet de correctement classifier une entité nommée. Parmi cette dernière catégorie, 24 entités appartiennent en fait à la classe MISC avec des entités qui correspondent à des groupes de gens qui ne sont pas des organisations comme "Sioux", des entités qui sont des éléments culturels comme "Broadway theatre" ou encore des langues comme "Negombo".

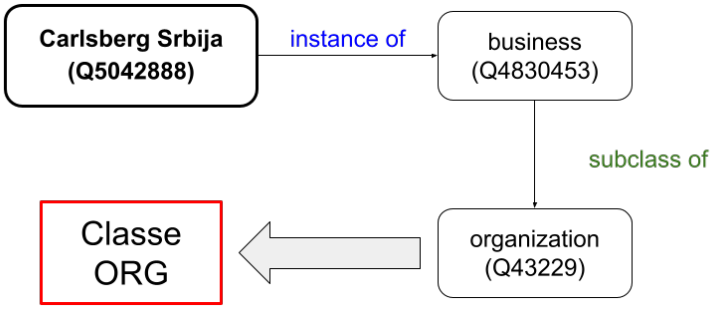
En inspectant les données, on peut constater différentes tendances dans les prédictions effectuées par l'outil utilisant WikiData. Premièrement, parmi les 677 cas où Freebase a raison, 595 — soit 87,6 % — de ces entités sont incorrectement prédites LOC par la méthode utilisant NECKAr, ce qui suggère un biais de cette approche pour cette classe. Les classes

"... through its local arm [LOC Carlsberg Srbija] ."

(a) Phrase dans WiNER étiquetée par NECKAr



(b) Chemin WikiData menant à LOC



(c) Chemin WikiData menant à ORG

Figure 6.4 – Exemple d'ancre mal étiquetée par la méthode utilisant WikiData. NECKAr l'associe à "LOC" et "ORG" mais la méthode suivie choisit aléatoirement la classe LOC. Ce sont les propriétés WikiData instance of et subclass of qui permettent de suivre ce chemin.

ORG et PER ne constituent de ce fait que 82 des cas où NECKAr a tort (voir Tableau 6.5). On peut voir sur ce tableau que la classe LOC est en fait sous-représentée parmi ces 677 entités et que Freebase en distribue correctement la majorité vers les classes ORG et PER. Ce biais pour LOC peut être illustré avec l'exemple de l'ancre "Carlsberg Srbija" (voir Figure 6.4) — une brasserie serbe — étiquetée LOC au lieu de ORG. En inspectant les données, il s'avère que cette ancre est associée aux deux classes LOC et ORG, mais qu'un choix aléatoire entre les deux l'associe à LOC. On remarque que parmi les 595 entités incorrectement étiquetées LOC par la méthode utilisant WikiData, 319 sont associées à deux classes par NECKAr dont celle correctement identifiée par Freebase. Ce constat signifie que 53,6 % de ces étiquetages sont incorrectement identifiés à cause de la méthode de choix aléatoire dans le cas où plusieurs classes sont trouvées par NECKAr.

Le Tableau 6.6 indique que la majorité des entités incorrectement classifiées dans la classe LOC par WikiData appartiennent à la classe ORG. Parmi ces entités, on distingue des établissements scolaires ou des équipes de football auxquelles on fait référence par le

Quantité	LOC	ORG	PER
WikiData	595	65	17
Freebase	19	434	224

Tableau 6.5 – Nombre de prédictions pour chaque méthode de typage dans chacune des trois classes considérées lorsque Freebase a raison. On regarde de ce fait ici les 677 entités correctement classifiées par Freebase.

	ORG	PER
Quantité	426	169

Tableau 6.6 – Quantité d'entités correctement classifiées ORG et PER par Freebase et qui sont classifiées LOC par WikiData.

nom de leur ville (voir la Figure 6.5 pour deux exemples). On peut avancer que WikiData ne gère pas bien les cas ambigus au vu du fait que certaines de ces entités sont principalement des ORG mais peuvent être également des LOC, comme des établissements scolaires ou des musées. Ces cas ne représentent cependant qu'une partie de ces 677 entités et on distingue également des cas non ambigus comme l'entité "Warner Bros.", une compagnie de production de films.

"... a sorority founded in 1874 at [ORG Colby College] ..."

"... he extended his loan deal at [ORG Watford] ..."

Figure 6.5 – Exemple d'entités ORG incorrectement classifiés LOC par WikiData : un établissement scolaire et une équipe de football.

Quantité	LOC	ORG	PER
WikiData	131	151	6
Freebase	36	72	180

Tableau 6.7 – Nombre de prédictions pour chaque méthode de typage dans chacune des trois classes considérées lorsque WikiData a raison. On regarde de ce fait ici les 288 entités correctement classifiées par WikiData.

En inspectant cette fois les cas où WikiData a raison (voir Tableau 6.7), on constate que c'est principalement les classes LOC et ORG que WikiData classifie bien. Ce constat est logique étant donné le peu de prédictions que WikiData fait pour cette classe (voir Figure 6.3(a)). Parallèlement, Freebase effectue plus de prédictions incorrectes dans cette classe que WikiData.

On peut voir dans les données que Freebase classifie souvent les entités ORG dans la classe PER (voir Tableau 6.8(a)). Parmi ces entités classifiées PER, on retrouve souvent des groupes musicaux comme "Village People" ou "The Offsprings" mais également des organisations de

"Series Six was broadcasted in the [LOC United States] on [PER PBS] ..."

"... after their concert with [PER Iron Maiden] that night ..."

Figure 6.6 – Exemple d'entités ORG incorrectement classifiées PER par Freebase : un réseau de télévision et un groupe musical.

presse ou de productions comme "PBS" ou "Marvel Comics" (voir deux exemples sur la Figure 6.6).

(a) ORG

	LOC	PER
Quantité	33	118

(b) LOC

	ORG	PER
Quantité	69	62

Tableau 6.8 – Quantité de prédictions effectuées dans les deux autres classes par Freebase dans le cas des entités correctement classifiées ORG et LOC par WikiData.

Finalement, Freebase prédit les 132 entités LOC dans les classes ORG et PER de façon égale (voir Tableau 6.8(b)). Ces cas correspondent souvent à des pays ou états comme "Pennsylvania" ou "New Zealand", ou à des cas ambigus comme "Russian Orthodox Church" qui fait référence à une église en particulier et non pas à une religion organisée.

"Village has a [ORG Russian Orthodox Church] ..."

"... rock band formed in Jönköping, [PER Sweden] , in 1992 ..."

Figure 6.7 – Exemple de cas où Freebase ne prédit pas correctement des entités LOC : une église et un pays.

6.3.3. L'étiquetage multi-classes

Afin de comprendre l'importance de la présence d'entités avec plusieurs classes prédites dans DataNER, on inspecte la proportion des mentions d'entités dans DataNER qui en ont plusieurs, et les couples de classes concernés. On cherche ici à comprendre les cas où plusieurs classes apparaissent ainsi que l'importance du problème dans les données dans leur intégralité — et pas seulement dans l'échantillon inspecté dans la comparaison manuelle. On cherche de ce fait à compter le nombre de mentions dans DataNER dont la page WikiData a été associée à deux ou trois classes, et qui ont de ce fait été étiquetées avec un choix aléatoire entre ces classes. On précise que ce choix aléatoire est effectué au niveau de la page WikiData et pas de la mention d'entité. De ce fait, toutes les mentions correspondant à une même page WikiData sont étiquetées avec la même classe.

Les statistiques sur ces mentions d'entités indiquées sur le Tableau 6.9 montrent qu'il s'agit d'un phénomène très important au sein de DataNER. On voit que la majorité de ces

	LOC - ORG	ORG - PER	LOC - PER	LOC - PER - ORG	**Total**
Quantité	409 855 085	1 359 525	139	17	411 214 766

Tableau 6.9 – Quantité d'entités candidates et ancres dans DataNER dont la page WikiData a été associée à plusieurs classes. On distingue les différentes combinaisons de classes trouvées.

mentions sont associées aux étiquettes LOC et ORG, ce qui est souvent le cas avec des pays, ou des institutions comme des musées ou des établissements scolaires. Un total de 411M de mentions de la ressource ont une page WikiData à laquelle NECKAr associe deux ou trois classes. Cela constitue 27,3 % de la totalité des mentions récupérées sur le texte de Wikipédia. On avance des solutions à ce problème dans la conclusion de ce mémoire.

6.3.4. Conclusion

On constate de ce fait que les deux expériences effectuées dans cette section indiquent qu'on obtient de meilleures performances avec Freebase. Dans la comparaison manuelle, malgré des tendances identifiées où Freebase produit des erreurs que WikiData ne fait pas ou peu, la méthode de typage utilisant Freebase produit de bien meilleurs résultats que celle utilisant WikiData. On note que 67,7 % des 1000 entités sélectionnées aléatoirement sont correctement étiquetées par Freebase dans la comparaison manuelle. Par ailleurs, la distribution des classes produites par Freebase parait beaucoup plus équilibrée que celle de WikiData, qui produit un nombre disproportionné d'entités étiquetées LOC. Lorsqu'on garde les étiquettes MISC, on continue de constater leur proportion bien trop grande, ce qui suggère d'y apporter une solution. On peut suggérer un sous-échantillonnage aléatoire des entités MISC dans DataNER, ou un affinage de l'outil NECKAr pour produire plus de pages WikiData associées aux classes LOC, PER et ORG.

On peut de ce fait conclure que la méthode de typage utilisant WikiData produit de moins bons résultats que celle utilisant Freebase même lorsqu'on ne considère pas la classe MISC. C'est la méthode de typage elle-même qui doit être modifiée afin de produire de meilleures étiquettes pour les entités nommées de Wikipédia. Ce résultat doit être mis à la lumière de la plus grande complexité du processus de typage dans WikiData que dans Freebase. En effet, cette dernière base de connaissances propose des types sur ces pages qui facilitent la création de mappings directs vers les classes NER utilisées. Cette fonctionnalité n'existe pas dans WikiData et la méthode qui utilise cette ressource cherche des chemins dans le graphe de la base de connaissances, ce qui facilite le mauvais étiquetage de certaines pages WikiData, considérant le grand nombre de ces chemins.

Étant donné les constats effectués sur le système de typage proposé dans ce mémoire, plusieurs suggestions peuvent être effectuées pour l'améliorer. Il apparaît que le choix aléatoire d'une classe lorsqu'une page WikiData est associée à plusieurs classes ne constitue pas

une approche satisfaisante pour gérer ces cas. Il semble de ce fait pertinent de suggérer une modification de cette méthode. Rendre le système de typage conscient du contexte pourrait aider à choisir la bonne étiquette pour une ancre donnée. En effet, on pourrait imaginer qu'un pays, associé à la fois aux étiquettes LOC et ORG pourrait être étiqueté différemment dans le texte selon les mots présents dans la phrase où il apparaît. L'utilisation de *gazeteers* associés à chaque classe pourrait par exemple permettre de désambiguïser le choix de l'étiquette en se basant sur les mots adjacents à l'ancre. On peut également imaginer implémenter des filtres quant à l'étiquetage des pages WikiData, afin d'empêcher la production d'une classe incompatible avec une propriété présente sur la page.

Conclusion

DataNER

Nous avons créé DataNER, une ressource de NER, en utilisant la supervision de Wikipédia et WikiData, deux ressources en ligne collaboratives. L'utilisation commune de ces deux ressources offre un grand potentiel par la quantité importante de données que l'on peut en extraire, ainsi que leur croissance continue sans date de fermeture prévue. La supervision distante a été fructueuse pour créer des ressources de NER avec Wikipédia et DBpedia (Zhu et al., 2019) ou Freebase (Ghaddar and Langlais, 2017), d'autres bases de connaissances disponibles en ligne. Utiliser WikiData au sein de cette approche semble donc de ce fait naturel.

Faire progresser cette méthode de supervision distante permet de produire une grande quantité de données annotées pouvant servir de données d'entraînement de NER. On peut imaginer des applications n'utilisant qu'un sous-ensemble du corpus produit, se concentrant sur un domaine précis, comme les articles Wikipédia traitant du droit, ou de la médecine. On peut également imaginer produire un autre ensemble de classes NER permettant de caractériser d'autres types d'entités, comme l'ont fait Ling and Weld (2012a).

La grande quantité d'ancres contenues dans le dump de Wikipédia que l'on manipule associée au grand nombre de faits contenus dans WikiData fait de DataNER un corpus contenant un plus grand nombre d'entités que les ressources auxquelles on se compare. Cette ressource contient 210M d'entités nommées définies sur 353M de tokens au sein d'un texte de 2,8G de tokens, ce qui signifie que 12,5 % des tokens de la ressource sont annotés. Parmi toutes ces entités, 88M sont des ancres issues du dump Wikipédia et 1,3G sont issues du procédé d'augmentation de données *outlinks*, ce qui en fait une partie fondamentale du processus de création de DataNER. La distribution des classes parmi les entités nommées de DataNER est déséquilibrée avec plus de la moitié des entités étiquetée avec la classe MISC.

Ce dernier point différencie DataNER des autres ressources auxquelles on la compare et introduit également la différence de qualité qu'on note entre DataNER et ces ressources. En effet, entraîner des modèles de NER permet de mettre en valeur de moins bonnes performances avec DataNER, allant quasiment jusqu'à une différence moyenne de 10 points de

F-mesure. Ces résultats démontrent malheureusement l'échec de la méthode proposée dans ce mémoire à produire des données annotées de qualité équivalente ou supérieure aux méthodes similaires auxquelles on se compare.

Un processus imparfait

Ces mauvaises performances amènent à questionner le procédé de création de DataNER décrit dans ce mémoire. Différentes inspections des données produites ont permis d'y identifier plusieurs problèmes.

On note en premier lieu un problème d'identification incorrecte de certaines entités nommées dans Wikipédia. En effet, certains mots comme "pop" ou "music" sont incorrectement identifiés comme entités nommées dans le texte. Ce problème vient des méthodes d'augmentation de données qui produisent un grand nombre d'entités — dont du bruit, comme ces mots — et de la présence de liens qui ne correspondent pas à des entités dans le dump Wikipédia. On remarque par ailleurs la grande proportion d'entités étiquetées MISC dans DataNER, comptabilisant 53,3 % des entités nommées du corpus.

On note en second lieu des problèmes avec les processus aléatoires utilisés dans la création de DataNER, comme l'inefficacité de la sélection aléatoire d'une classe dans le cas de mentions étiquetées avec plusieurs classes par NECKAr. En effet, cette méthode de sélection d'étiquette amène à un grand nombre d'entités incorrectement étiquetées, comme on a pu le constater dans la comparaison manuelle effectuée dans le chapitre 6. Ce dernier constat suggère l'utilisation de méthodes alternatives pour désambiguïser les mentions associées à plusieurs classes. De plus, on remarque également des problèmes de segmentation d'entités nommées avec la méthode de résolution aléatoire des conflits existant entre mentions d'entités nommées lors de la création de DataNER. En effet, certaines entités ne sont ainsi que partiellement identifiées dans certains cas, comme "Apple" au lieu de "Apple Music". Cette méthode est utilisée dans 42,9 % de la résolution des conflits existants — le reste étant résolu grâce à l'heuristique qui donne la priorité aux ancres — ce qui suggère qu'un nombre non négligeable de résolutions de conflits sont erronées.

Perspectives d'améliorations

Il est ainsi pertinent d'identifier des façons d'améliorer ce processus de création de corpus — dont le potentiel est toujours présent vu la grande quantité d'information présente dans WikiData — en le rendant plus déterministe là où c'est possible, et en y appliquant des heuristiques de correction.

En premier lieu, on peut suggérer de régler les problèmes de mots incorrectement identifiés comme des entités nommées en utilisant des listes de filtrage qui permettent de supprimer les entités qui y apparaissent. Cette méthode simple pourrait peut-être améliorer la qualité des

données, au moins partiellement. Il semble également possible d'améliorer les performances de modèles entraînés sur DataNER en diminuant la proportion d'entités MISC dans la ressource, soit par sous-échantillonnage aléatoire, ou en retravaillant le fonctionnement de NECKAr pour que l'outil étiquette plus de pages WikiData avec les classes LOC, PER et ORG.

Une autre voie d'amélioration est la modification de la gestion des mentions d'entités multi-classes. Plusieurs choix apparaissent quand une mention est identifiée dans plusieurs classes. On peut utiliser la phrase elle-même pour choisir une étiquette, avec des *gazeteers* pour utiliser la présence de certains mots comme indices quant à la bonne classe par exemple, ou on peut choisir la classe avec qui la page WikiData de la mention détient le plus grand nombre de chemins. Finalement, on peut également imaginer choisir aléatoirement une classe NER au niveau de la mention d'entité nommée — et non pas de la page WikiData — diluant ainsi les erreurs dans le texte parmi toutes les occurrences d'entités nommées liées à une même page.

Il serait également pertinent de trouver une meilleure méthode de résolution des conflits que la sélection aléatoire, qui produit parfois des entités mal segmentées. Cependant, étant donné le grand nombre de cas de conflits différents, la tâche n'est pas aisée. On suggère de ce fait de travailler vers une réduction du nombre de conflits. On peut par exemple imaginer filtrer certaines entités candidates parmi celles récupérées lors de l'augmentation de données — si elles sont une sous-chaîne d'une autre entité par exemple — ou établir un classement de confiance entre les méthodes d'augmentation de données. Cette dernière suggestion permettrait d'introduire un aspect déterministe dans la résolution des conflits, rendant les données cohérentes au sein d'un processus fixe. Un tel classement deviendrait alors un élément avec lequel expérimenter dans le *pipeline* d'extraction de DataNER. On peut considérer conceptuellement un tel classement comme une généralisation de l'heuristique de priorité aux ancres, suggérant qu'une telle méthode pourrait être utile.

Perspective sur le travail accompli

Malgré les résultats négatifs de ce mémoire, une grande quantité de travail a pu être accomplie dans le cadre de ce travail de recherche. En premier lieu, une contribution technique d'exploration et de manipulation des données décrites dans ce mémoire a été effectuée. En effet, lors de ce mémoire, une infrastructure de manipulation des données massives de Wikipédia et WikiData a due être développée. Le pipeline d'extraction de DataNER a également été le sujet d'un gros travail de développement logiciel. Ce travail technique qui a constitué une grande partie du temps de ce mémoire est rendu disponible dans un dépôt GitHub [2]

2. `https://github.com/LucasPages/dataner_creation`

contenant le code qui a permis de créer DataNER, ainsi qu'un article de blog [3] décrivant comment manipuler ces données avec le système de base de données *MongoDB*.

En second lieu, l'analyse des performances de DataNER et de ses données ont permis d'identifier que le problème principal provient du système de typage utilisant NECKAr. Ce dernier constat constitue une contribution sur l'utilisation des données de WikiData, mettant en valeur la difficulté de la production d'étiquetage NER pour les mentions d'entités nommées de Wikipédia. On note ainsi que ce travail constitue une première approche de cette méthode et on avance qu'il faut continuer le travail sur le système de typage et affiner NECKAr.

De manière générale, on remarque la complexité du travail impliqué dans la création d'une ressource telle que DataNER par supervision distante. On a dû effectuer un travail important de développement d'infrastructure pour la production et le calibrage de la ressource, suivi par une analyse poussée des données produites pour pouvoir former un avis sur la ressource produite. C'est uniquement grâce à ce travail accompli que nous avons été en mesure de revenir sur le système de typage utilisé, qui constitue le problème principal de DataNER.

La quantité de travail accomplie et les défauts identifiés dans les résultats sous-entendent de ce fait que DataNER n'est que la première ébauche d'une méthode qui doit être améliorée. L'utilisation et l'observation des données massives de WikiData avec Wikipédia ainsi que les améliorations potentielles précédemment décrites ont convaincu l'auteur de ce mémoire que le potentiel sous-jacent de cette ressource est grand et doit être affiné de façon à être utilisé proprement. Un tel travail futur permettra à WikiData et Wikipédia de former des corpus de qualité, dépassant les performances de DataNER et compétitifs dans le domaine du NER. Alors que le NLP nécessite de plus en plus de données, la supervision distante utilisant des données ouvertes à toutes et à tous comme WikiData devient une piste prometteuse.

3. `https://medium.com/@lucas.pages123/manipulating-wikipedia-and-wikidata-dumps-thanks-to-mongodb-2f74c239c28f`

Références bibliographiques

Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1139.

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. POLYGLOT-NER : massive multilingual named entity recognition. *CoRR*, abs/1410.3791, 2014. URL http://arxiv.org/abs/1410.3791.

Giuseppe Attardi. Wikiextractor. https://github.com/attardi/wikiextractor, 2015.

Bogdan Babych and Anthony Hartley. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other language technology tools, Improving MT through other language technology tools, Resource and tools for building MT at EACL 2003*, 2003. URL https://aclanthology.org/W03-2201.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP : Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-3302.

Gabriel Bernier-Colborne and Phillippe Langlais. HardEval : Focusing on challenging tokens to assess robustness of NER. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1704–1711, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.211.

Dan Bikel, Richard Schwartz, and Ralph Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 34, 02 1999. doi : 10.1023/A:1007558221122.

Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. Neuroner : an easy-to-use program for named-entity recognition based on neural networks, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi : 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Rezarta Dogan, Robert Leaman, and Zhiyong lu. Ncbi disease corpus : A resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47, 01 2014. doi : 10.1016/j.jbi.2013.12.006.

Sean R Eddy. Hidden markov models. *Current Opinion in Structural Biology*, 6(3) :361–365, 1996. ISSN 0959-440X. doi : https://doi.org/10.1016/S0959-440X(96)80056-X. URL https://www.sciencedirect.com/science/article/pii/S0959440X9680056X.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised named-entity extraction from the web : An experimental study. *Artificial Intelligence*, 165(1) :91–134, 2005. ISSN 0004-3702. doi : https://doi.org/10.1016/j.artint.2005.03.001. URL https://www.sciencedirect.com/science/article/pii/S0004370205000366.

Christiane Fellbaum. *WordNet : An Electronic Lexical Database*. Bradford Books, 1998.

Jenny Rose Finkel and Christopher D. Manning. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies : The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/N09-1037.

Johanna Geiß, Andreas Spitz, and Michael Gertz. Neckar : A named entity classifier for wikidata. In Georg Rehm and Thierry Declerck, editors, *Language Technologies for the Challenges of the Digital Age*, pages 115–129, Cham, 2018. Springer International Publishing. ISBN 978-3-319-73706-5.

Abbas Ghaddar and Phillippe Langlais. Coreference in Wikipedia : Main concept resolution. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 229–238, Berlin, Germany, August 2016. Association for Computational Linguistics. doi : 10.18653/v1/K16-1023. URL https://aclanthology.org/K16-1023.

Abbas Ghaddar and Phillippe Langlais. WiNER : A Wikipedia annotated corpus for named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, pages 413–422, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I17-1042.

Abbas Ghaddar and Phillippe Langlais. Robust lexical features for improved neural network named-entity recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1896–1907, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1161.

Yoav Goldberg and Joakim Nivre. Training deterministic parsers with non-deterministic oracles. *Transactions of the Association for Computational Linguistics*, 1 :403–414, 2013. doi : 10.1162/tacl_a_00237. URL https://aclanthology.org/Q13-1033.

Ralph Grishman and Beth Sundheim. Message understanding conference-6 : A brief history. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, page 466–471, USA, 1996. Association for Computational Linguistics. doi : 10.3115/992628.992709. URL https://doi.org/10.3115/992628.992709.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. Named entity recognition in query. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 267–274, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584836. doi : 10.1145/1571941.1571989. URL https://doi.org/10.1145/1571941.1571989.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8) :1735–1780, nov 1997. ISSN 0899-7667. doi : 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging, 2015. URL https://arxiv.org/abs/1508.01991.

Gjergji Kasneci, Maya Ramanath, Fabian Suchanek, and Gerhard Weikum. The yago-naga approach to knowledge discovery. *SIGMOD Rec.*, 37(4) :41–47, mar 2009. ISSN 0163-5808. doi : 10.1145/1519103.1519110. URL https://doi.org/10.1145/1519103.1519110.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558607781.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *CoRR*, abs/1603.01360, 2016. URL http://arxiv.org/abs/1603.01360.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, and Christian Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6, 01 2014. doi : 10.3233/SW-140134.

Constantine Lignos and Marjan Kamyab. If you build your own NER scorer, non-replicable results will come. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 94–99, Online, November 2020. Association for Computational Linguistics. doi : 10.18653/v1/2020.insights-1.15. URL https://aclanthology.org/2020.insights-1.15.

Wang Ling, Yulia Tsvetkov, Silvio Amir, Ramón Fermandez, Chris Dyer, Alan W Black, Isabel Trancoso, and Chu-Cheng Lin. Not all contexts are created equal : Better word representations with variable attention. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1367–1372, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi : 10.18653/v1/D15-1161. URL `https://aclanthology.org/D15-1161`.

Xiao Ling and Daniel Weld. Fine-Grained Entity Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 26(1) :94–100, 2012a. ISSN 2374-3468. URL `https://ojs.aaai.org/index.php/AAAI/article/view/8122`. Number : 1.

Xiao Ling and Daniel S. Weld. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, page 94–100. AAAI Press, 2012b.

Th Anh Lê. A deep neural network model for the task of named entity recognition. *International Journal of Machine Learning and Computing*, 9, 02 2019a. doi : 10.18178/ijmlc.2019.9.1.758.

Th Anh Lê. A deep neural network model for the task of named entity recognition. *International Journal of Machine Learning and Computing*, 9, 02 2019b. doi : 10.18178/ijmlc.2019.9.1.758.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf, 2016a.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 1064–1074, Berlin, Germany, August 2016b. Association for Computational Linguistics. doi : 10.18653/v1/P16-1101. URL `https://aclanthology.org/P16-1101`.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014. URL `http://www.aclweb.org/anthology/P/P14/P14-5010`.

Andrei Mikheev. A knowledge-free method for capitalized word disambiguation. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 159–166, College Park, Maryland, USA, June 1999. Association for Computational Linguistics. doi : 10.3115/1034678.1034710. URL `https://aclanthology.org/P99-1021`.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013. URL `http://arxiv.org/abs/1310.4546`.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual*

Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pages 1003–1011, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/P09-1113.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30, 08 2007. doi : 10.1075/li.30.1.03nad.

Roberto Navigli and Simone Paolo Ponzetto. Babelnet : The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193(0) :217 – 250, 2012. ISSN 0004-3702. doi : 10.1016/j.artint.2012.07.001. URL http://www.sciencedirect.com/science/article/pii/S0004370212000793.

Joel Nothman, James R. Curran, and Tara Murphy. Transforming Wikipedia into named entity training data. In *Proceedings of the Australasian Language Technology Association Workshop 2008*, pages 124–132, Hobart, Australia, December 2008. URL https://aclanthology.org/U08-1016.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194 : 151–175, 2013. ISSN 0004-3702. doi : https://doi.org/10.1016/j.artint.2012.03.006. URL https://www.sciencedirect.com/science/article/pii/S0004370212000276. Artificial Intelligence, Wikipedia and Semi-Structured Resources.

Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. The genia corpus : An annotated research abstract corpus in molecular biology domain. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, page 82–86, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

Thomas Pellissier Tanon, Denny Vrandečić, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. From freebase to wikidata : The great migration. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 1419–1428, Republic and Canton of Geneva, CHE, 2016. International World Wide Web Conferences Steering Committee. ISBN 9781450341431. doi : 10.1145/2872427.2874809. URL https://doi.org/10.1145/2872427.2874809.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe : Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi : 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi :

10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. Conll-2012 shared task : Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, CoNLL '12, page 1–40, USA, 2012. Association for Computational Linguistics.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. Automatic construction and evaluation of a large semantically enriched wikipedia. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2894–2900. AAAI Press, 2016. ISBN 9781577357704.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL https://aclanthology.org/W09-1119.

Hadas Raviv, Oren Kurland, and David Carmel. Document retrieval using entity-based language models. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, page 65–74, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450340694. doi : 10.1145/2911451.2911508. URL https://doi.org/10.1145/2911451.2911508.

Burr Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. *Proceedings of the Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, 01 2004. doi : 10.3115/1567594.1567618.

Lorraine K. Tanabe, Natalie Xie, Lynne H. Thom, Wayne Matten, and W. John Wilbur. Genetag : a tagged corpus for gene/protein named entity recognition. *BMC Bioinformatics*, 6 :S3 – S3, 2005.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task : Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, page 142–147, USA, 2003. Association for Computational Linguistics. doi : 10.3115/1119176.1119195. URL https://doi.org/10.3115/1119176.1119195.

Antonio Toral and Rafael Muñoz. A proposal to automatically build and maintain gazetteers for named entity recognition by using wikipedia. In *Workshop On New Text Wikis And Blogs And Other Dynamic Text Sources*, 2006.

Denny Vrandečić and Markus Krötzsch. Wikidata : A free collaborative knowledgebase. *Commun. ACM*, 57(10) :78–85, September 2014. ISSN 0001-0782. doi : 10.1145/2629489. URL https://doi.org/10.1145/2629489.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. Automated concatenation of embeddings for structured prediction. *CoRR*, abs/2010.05006, 2020. URL https://arxiv.org/abs/2010.05006.

Robert West, Ashwin Paranjape, and Jure Leskovec. Mining missing hyperlinks from human navigation traces. *Proceedings of the 24th International Conference on World Wide Web*, May 2015. doi : 10.1145/2736277.2741666. URL http://dx.doi.org/10.1145/2736277.2741666.

Marc Wick. Geonames ontology, 2015. URL http://www.geonames.org/about.html.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. DocRED : A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy, July 2019. Association for Computational Linguistics. doi : 10.18653/v1/P19-1074. URL https://aclanthology.org/P19-1074.

Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. Hyena : Hierarchical type classification for entity names. In *COLING*, 2012.

Shaodian Zhang and NoÃ©mie Elhadad. Unsupervised biomedical named entity recognition : Experiments with clinical and biological texts. *Journal of Biomedical Informatics*, 46(6) : 1088–1098, 2013. ISSN 1532-0464. doi : https://doi.org/10.1016/j.jbi.2013.08.004. URL https://www.sciencedirect.com/science/article/pii/S1532046413001196. Special Section : Social Media Environments.

Mengdi Zhu, Zheye Deng, Wenhan Xiong, Mo Yu, Ming Zhang, and William Yang Wang. Towards open-domain named entity recognition via neural correction models. *CoRR*, abs/1909.06058, 2019. URL http://arxiv.org/abs/1909.06058.

Annexe A

Informations Additionnelles

A.1. Mapper OntoNotes

Le corpus OntoNotes v5 (Pradhan et al., 2012) contient des annotations de NER dans 18 classes différentes. Cependant, on l'utilise dans ce mémoire comme un corpus de test pour des prédictions sur les 4 classes du corpus CoNLL03 : LOC, PER, ORG et MISC.

On choisit de ce fait de mapper le corpus de test de la ressource OntoNotes, suivant de ce fait les travaux de (Ghaddar and Langlais, 2017) ou (Zhu et al., 2019).

Classe OntoNotes	Proportion	Description	Classe CoNLL03
GPE	19.9 %	Pays, villes, états	LOC
PERSON	17.7 %	Personnes, même fictives	PER
ORG	15.9 %	Entreprises, agences, institutions, etc	ORG
DATE	14.2 %	Dates absolues ou relatives, ou périodes	
CARDINAL	8.3 %	Nombres	
NORP	7.5 %	Nationalités, groupes religieux ou politiques	MISC
PERCENT	3.1 %	Pourcentage	
MONEY	2.8 %	Valeurs ou unités monétaires	
TIME	1.9 %	Temps inférieurs à 1 journée	
ORDINAL	1.7 %	Nombres ordinaux	
LOC	1.6 %	Emplacements autre que "GPE"	LOC
WORK_OF_ART	1.5 %	Titres d'oeuvres	MISC
FAC	1.2 %	Bâtiments, aéroports, etc	LOC
QUANTITY	0.9 %	Mesures	
PRODUCT	0.7 %	Produits qui ne sont pas des services	MISC
EVENT	0.6 %	Évènements	MISC
LAW	0.4 %	Documents de loi	MISC
LANGUAGE	0.2 %	Langues	MISC

Tableau A.1 – Informations sur les différentes classes du corpus de test d'OntoNotes v5.0. La colonne Classe CoNLL03 contient les classes CoNLL03 correspondantes, une cellule noire correspond à une classe supprimée par le mapping.

A.1.1. Le corpus OntoNotes

Le schéma d'annotations du corpus OntoNotes contient 18 classes de NER. Ces classes servent à catégoriser avec finesse des entités nommées issues de textes de différentes sources.

On compte un total de **11 257 entités nommées** dans le corpus de test de la ressource, dont 4 830 entités uniques.

Les classes du corpus sont relativement déséquilibrées, avec la classe la plus fréquente GPE à 19,9% des entités nommées et la classe la moins fréquente LANGUAGE à 0,2% (voir Table A.1).

A.1.2. Le mapping

Le mapping est défini à partir des ressources publiées avec l'article (Nothman et al., 2013) [1]. Il est décrit dans la Table A.1.

Classe CoNLL03	# Classes OntoNotes
NON	7
MISC	6
LOC	3
PER	1
ORG	1

Tableau A.2 – Nombre de classes OntoNotes mappées vers chaque classe CoNLL03. La ligne *NON* fait référence aux classes OntoNotes qui sont supprimées.

Les 18 classes d'OntoNotes sont mappées vers les quatre classes de CoNLL03. Parmi ces classes, 7 sont supprimées, 6 sont mappées vers MISC, 3 vers LOC, 1 vers ORG et 1 vers PER (voir Table A.2).

On obtient alors une nouvelle version du corpus de test d'OntoNotes qui contient **7 545 entités nommées**, dont 2 953 entités uniques.

Classe	Quantité	Proportion
LOC	2 554	33,9%
PER	1 988	26,3%
ORG	1 795	23,8%
MISC	1 208	16,0%

Tableau A.3 – Quantité et proportion de chaque classe CoNLL03 d'entités nommées dans la version mappée du corpus de test d'OntoNotes v5.0.

La nouvelle distribution des classes du corpus peut être inspectée sur la Table A.3.

On peut par ailleurs constater une perte de **3 712 entités nommées** après le mapping du corpus.

1. Voir `https://figshare.com/articles/dataset/Learning_multilingual_named_entity_recognition_from_Wikipedia/5462500`

A.2. Détails des configurations de calibrage

On détaille ici les configurations utilisées pour sélectionner les entités candidates — issues de l'augmentation de données — dans le processus de calibrage de DataNER.

On inspecte ces combinaisons une technique à la fois.

A.2.1. Augmentation par titre

Deux méthodes sont utilisées pour obtenir des entités en utilisant le titre :

— en propageant le titre dans l'article

— en propageant les alias du titre dans l'article

Pour chacune de ces deux méthodes, on dispose de deux choix : l'utiliser ou ne pas l'utiliser. Ce qui amène à 4 combinaisons, incluant celle où aucune des deux techniques n'est utilisée.

alias-titre	expansion-titre
non-utilisée	utilisée
utilisée	non-utilisée
utilisée	utilisée
non-utilisée	non-utilisée

Tableau A.4 – Les 4 choix possibles pour les méthodes d'augmentation par titre.

A.2.2. Augmentation par alias et par liens

Les augmentations par alias et par liens disposent des mêmes trois valeurs possibles dans le cadre des configurations de calibrage : non-utilisée, phrase et article.

"Phrase" correspond à la propagation de l'entité dans la phrase d'où on l'a récupère seulement, là où "article" signifie que l'on propage l'entité dans tout l'article.

On dispose de ce fait de 3 choix possibles pour chacune de ces méthodes.

A.3. Fichier de configuration SpaCy

En dessous, on partage le fichier de configuration qui permet de paramétrer l'entraînement d'un modèle NER dans SpaCy.

Il s'agit d'un fichier *.cfg* dont les sections décrivent chacune un aspect du modèle ou de l'entraînement. Les champs "train" et "dev" de la section "paths" doivent être assignés respectivement par les corpus d'entraînement et de développement au format BIO2 à utiliser. Le champs "vectors" de cette même section doit être assigné l'emplacement des embeddings statiques à utiliser.

```
[paths]
train = null
dev = null
raw = null
init_tok2vec = null
vectors = null

[system]
gpu_allocator = null
seed = 0

[nlp]
lang = "en"
pipeline = ["ner"]
tokenizer = {"@tokenizers":"spacy.Tokenizer.v1"}
disabled = []
before_creation = null
after_creation = null
after_pipeline_creation = null
batch_size = 1000

[components]

[components.ner]
factory = "ner"
moves = null
update_with_oracle_cut_size = 100

[components.ner.model]
@architectures = "spacy.TransitionBasedParser.v1"
state_type = "ner"
extra_state_tokens = false
hidden_width = 128
maxout_pieces = 3
use_upper = true
nO = null

[components.ner.model.tok2vec]
```

```
@architectures = "spacy.Tok2Vec.v1"

[components.ner.model.tok2vec.embed]
@architectures = "spacy.MultiHashEmbed.v1"
width = ${components.ner.model.tok2vec.encode.width}
attrs = ["NORM","PREFIX","SUFFIX","LOWER"]
rows = [5000,2500,2500,2500]
include_static_vectors = true

[components.ner.model.tok2vec.encode]
@architectures = "spacy.MaxoutWindowEncoder.v1"
width = 128
depth = 4
window_size = 1
maxout_pieces = 3

[corpora]

[corpora.dev]
@readers = "spacy.Corpus.v1"
path = ${paths.dev}
max_length = 0
gold_preproc = true
limit = 0
augmenter = null

[corpora.train]
@readers = "spacy.Corpus.v1"
path = ${paths.train}
max_length = 0
gold_preproc = true
limit = 0
augmenter = null

[training]
train_corpus = "corpora.train"
dev_corpus = "corpora.dev"
seed = ${system:seed}
```

```
gpu_allocator = ${system:gpu_allocator}
dropout = 0.1
accumulate_gradient = 1
patience = 1600
max_epochs = 30
max_steps = 20000
eval_frequency = 200
frozen_components = []
before_to_disk = null

[training.batcher]
@batchers = "spacy.batch_by_words.v1"
discard_oversize = false
tolerance = 0.2
get_length = null

[training.batcher.size]
@schedules = "compounding.v1"
start = 100
stop = 1000
compound = 1.001
t = 0.0

[training.logger]
@loggers = "spacy.ConsoleLogger.v1"
progress_bar = true

[training.optimizer]
@optimizers = "Adam.v1"
beta1 = 0.9
beta2 = 0.999
L2_is_weight_decay = true
L2 = 0.01
grad_clip = 1.0
use_averages = true
eps = 0.00000001
learn_rate = 0.001
```

```
[training.score_weights]
ents_per_type = null
ents_f = 1.0
ents_p = 0.0
ents_r = 0.0

[pretraining]

[initialize]
vectors = "${paths:vectors}"
init_tok2vec = ${paths.init_tok2vec}
vocab_data = null
lookups = null
before_init = null
after_init = null

[initialize.components]

[initialize.tokenizer]
```

Annexe B

Résultats des expériences de calibrage

Dans cette partie, on détaille les résultats des expériences de calibrage dans leur intégralité. Précisément, on rapporte la F-mesure des prédictions des 72 modèles SpaCy entraînés sur les 4 corpus de test utilisés.

On met en gras les résultats les plus haut par colonne entre les deux tables.

alias-titre	exp-titre	outlinks	alias	CoNLL03	OntoNotes	WikiGold	Pages Web	*AVG*
NON	NON	article	article	50,22	45,05	45,61	32,41	43,32
OUI	NON	article	article	49,61	43,76	47,28	30,98	42,90
NON	OUI	article	article	52,1	43,36	47,03	34,80	44,32
OUI	OUI	article	article	**53,97**	**47,49**	**48,48**	**38,45**	**47,09**
NON	NON	phrase	article	34,62	25,5	38,55	26,61	31,32
OUI	NON	phrase	article	34,55	29,24	38,61	27,26	32,41
NON	OUI	phrase	article	38,33	30,72	42,22	28,38	34,91
OUI	OUI	phrase	article	38,93	30,67	41,57	30,74	35,47
NON	NON	article	phrase	47,19	42,25	44,34	34,3	42,02
OUI	NON	article	phrase	46,71	43,03	44,41	28,79	40,73
NON	OUI	article	phrase	48,11	43,16	45,58	32,98	42,45
OUI	OUI	article	phrase	48,42	43,65	47,31	35,11	43,62
NON	NON	phrase	phrase	32,21	28,56	38,94	25,21	31,23
OUI	NON	phrase	phrase	29,91	28,12	37,17	23,14	29,58
NON	OUI	phrase	phrase	36,45	31,97	40,86	29,11	34,59
OUI	OUI	phrase	phrase	34,74	31,55	40,62	29,91	34,20
NON	NON	NON	article	26,3	20,95	34,98	25,83	27,01
OUI	NON	NON	article	26,86	23,34	34,85	22,71	26,94
NON	OUI	NON	article	30,00	23,15	37,77	25,42	29,08
OUI	OUI	NON	article	30,63	23,73	38,08	29,03	30,36
NON	NON	article	NON	46,04	39,05	43,16	27,5	38,93
OUI	NON	article	NON	46,96	42,76	45,17	28,37	40,81
NON	OUI	article	NON	48,43	42,32	45,74	28,76	41,31
OUI	OUI	article	NON	49,94	42,32	45,74	28,76	41,31
NON	NON	NON	phrase	24,71	23,37	34,64	21,00	25,93
OUI	NON	NON	phrase	24,42	23,91	34,05	20,66	25,76
NON	OUI	NON	phrase	49,04	41,98	45,1	32,32	42,11
OUI	OUI	NON	phrase	48,48	42,45	45,86	31,44	42,05
NON	NON	phrase	NON	27,30	23,17	37,83	22,49	27,69
OUI	NON	phrase	NON	30,44	24,00	37,87	26,05	29,59
NON	OUI	phrase	NON	27,46	22,31	37,65	20,55	26,99
OUI	OUI	phrase	NON	33,35	26,25	42,72	29,57	32,97
OUI	NON	NON	NON	19,93	17,76	32,84	16,65	21,79
NON	OUI	NON	NON	24,53	18,74	35,88	22,82	25,49
OUI	OUI	NON	NON	24,2	19,23	36,87	23,40	25,92

Tableau B.1 – Résultats des expériences de calibrage avec l'heuristique de choix en faveur des ancres. Le choix de la résolution aléatoire des conflits est toujours utilisé. Dans ce tableau "OUI" fait référence à l'utilisation de la technique de la colonne pertinente, "NON" fait référence à sa non-utilisation. "article" et "phrase" dans les colonnes "outlinks" et "alias" font référence à l'échelle de propagation des entités obtenues par ces méthodes pour la combinaison de cette ligne.

alias-titre	exp-titre	outlinks	alias	CoNLL03	OntoNotes	WikiGold	Pages Web	*AVG*
NON	NON	article	article	47,55	42,85	41,91	29,77	40,52
OUI	NON	article	article	45,82	40,83	41,76	25,61	38,50
NON	OUI	article	article	50,77	43,92	44,58	30,91	42,54
OUI	OUI	article	article	50,04	42,44	42,99	28,81	41,07
NON	NON	phrase	article	31,21	23,95	32,38	20,56	27,02
OUI	NON	phrase	article	32,52	25,20	35,18	21,15	28,51
NON	OUI	phrase	article	35,77	27,28	38,27	27,05	32,09
OUI	OUI	phrase	article	36,98	27,33	36,53	25,42	31,56
NON	NON	article	phrase	46,17	42,92	41,71	26,44	39,31
OUI	NON	article	phrase	45,16	41,30	40,42	29,97	39,21
NON	OUI	article	phrase	48,02	39,69	43,54	28,26	39,87
OUI	OUI	article	phrase	46,67	41,66	40,52	27,42	39,06
NON	NON	phrase	phrase	27,13	25,79	33,30	17,17	25,84
OUI	NON	phrase	phrase	30,42	29,37	34,07	22,65	29,12
NON	OUI	phrase	phrase	31,82	27,10	36,13	23,17	29,55
OUI	OUI	phrase	phrase	32,15	29,25	35,60	23,47	30,11
NON	NON	NON	article	26,02	21,34	33,08	18,72	24,79
OUI	NON	NON	article	27,21	21,52	32,54	20,51	25,44
NON	OUI	NON	article	31,82	27,10	36,13	23,17	29,55
OUI	OUI	NON	article	32,05	23,73	35,03	29,11	29,98
NON	NON	article	NON	44,16	40,34	41,23	23,80	37,38
OUI	NON	article	NON	45,16	39,92	41,35	24,49	37,73
NON	OUI	article	NON	45,97	39,39	41,78	24,25	37,84
OUI	OUI	article	NON	47,10	42,32	43,81	27,66	40,22
NON	NON	NON	phrase	22,89	23,26	31,98	20,89	24,75
OUI	NON	NON	phrase	23,49	22,54	32,32	22,66	25,25
NON	OUI	NON	phrase	28,09	24,51	35,48	24,19	28,06
OUI	OUI	NON	phrase	28,12	24,91	34,16	26,46	28,41
NON	NON	phrase	NON	27,14	19,82	35,03	20,06	25,51
OUI	NON	phrase	NON	25,66	22,17	34,92	23,45	26,55
NON	OUI	phrase	NON	28,83	24,18	39,08	25,67	29,44
OUI	OUI	phrase	NON	29,89	23,86	38,41	28,32	30,12
OUI	NON	NON	NON	19,85	16,65	32,58	19,78	22,21
NON	OUI	NON	NON	25,17	18,43	36,70	24,44	26,18
OUI	OUI	NON	NON	26,92	20,63	37,88	29,44	28,71

Tableau B.2 – Résultats des expériences de calibrage sans l'heuristique de choix en faveur des ancres. Le choix de la résolution aléatoire des conflits est toujours utilisé. Dans ce tableau "OUI" fait référence à l'utilisation de la technique de la colonne pertinente, "NON" fait référence à sa non-utilisation. "article" et "phrase" dans les colonnes "outlinks" et "alias" font référence à l'échelle de propagation des entités obtenues par ces méthodes pour la combinaison de cette ligne.