Exploration de traits pour la reconnaissance d'entités nommées du Français par apprentissage automatique

Yoann Dupont[1, 2]
(1) Laboratoire Lattice (CNRS, ENS, Université Sorbonne Nouvelle, PSL Research University, USPC)
1 rue Maurice Arnoux, 92120 Montrouge
(2) Expert System France, 207 rue de Bercy, 75012 Paris
yoa.dupont@gmail.com

RÉSUMÉ

Dans cet article, nous explorons divers traits proposés dans la littérature afin de fournir un détecteur d'entités nommées pour le Français appris automatiquement sur le French Treebank. Nous étudions l'intégration de connaissances en domaine, l'apport de la classification des verbes, la gestion des mots inconnus et l'intégration de traits non locaux. Nous comparons ensuite notre système aux récents réseaux de neurones.

ABSTRACT

Feature exploration for French Named Entity Recognition with Machine Learning

In this article, we explore features that have been described in the litterature to learn a named entity recognizer on the French TreeBank. We study the use of in-domain knowledge, the benefit of classifying verbs, the handling of unknown words and non-local features. We then compare our system to recent neural networks.

MOTS-CLÉS : Reconnaissance d'entités nommées, French Treebank, Apprentissage automatique, CRF, réseaux de neurones.

KEYWORDS: named entity recognition, French Treebank, machine learning, CRF, neural networks.

1 Introduction

La reconnaissance d'entités nommées (REN) est une tâche importante du TAL qui sert généralement de point de départ à d'autres tâches telles que l'extraction de relations (Bunescu & Mooney, 2005), l'entity linking, la résolution de coréférence (Hajishirzi *et al.*, 2013). De nombreux travaux ont été réalisés dans le domaine de l'apprentissage automatique sur l'anglais, mais assez peu sur d'autres langues, en particulier le français, pour lequel on peut citer Galliano *et al.* (2009); Gravier *et al.* (2012); Sagot *et al.* (2012). Les interactions entre les différents traits que l'on peut intégrer dans ces systèmes ont déjà fait l'objet d'études sur l'anglais Ratinov & Roth (2009); Tkachenko & Simanovsky (2012), mais à notre connaissance aucune n'a été faite sur le français. Ici, nous explorons divers traits utilisés dans la littérature afin d'en évaluer l'impact sur la REN pour le français. Nous partirons des traits proposés par Raymond & Fayolle (2010) sur le French Treebank (FTB) annoté en entités nommées (Sagot *et al.*, 2012), que nous enrichirons progressivement.

Nous étudions également comment organiser les ressources lexicales ainsi que la meilleure manière de les intégrer une fois cette classification faite, comme la classification des verbes et la gestion des mots inconnus.

2 La tâche et les corpus

Nous considérons la REN comme une tâche d'étiquetage de séquences. Les séquences à annoter sont les phrases d'un texte, où chaque élément est un *token*. Une entité pouvant correspondre à plusieurs tokens, nous avons utilisé le schéma d'annotation BIO (*Beginning*, *Inside Outside*), où un *token* est annoté `B-<Entité>` s'il est le premier *token* d'une entité, `I-<Entité>` s'il appartient à une entité sans être son premier élément et `O` s'il ne fait pas partie d'une entité. Nous avons principalement utilisé les CRF, particulièrement adaptés cette tâche, que nous comparons à des réseaux de neurones récents faisant partie des meilleurs systèmes sur l'anglais. Nous avons choisi le FTB annoté en entités nommées car il est le plus proche d'un point de vue typologique du corpus le plus utilisé pour l'anglais : le corpus CoNLL 2003 (Tjong Kim Sang & De Meulder, 2003).

2.1 les CRF

Les *Conditional Random Fields* (CRF) (Lafferty *et al.*, 2001) sont des modèles graphiques probabilistes (Koller & Friedman, 2009; Gaussier & Yvon, 2011). Ils prennent en entrée un ensemble structuré d'éléments x et fournissent en sortie un étiquetage structuré y de cet ensemble. Ils sont dits discriminants car ils modélisent la probabilité conditionnelle d'un ensemble d'étiquettes y selon une entrée x. Quand le graphe exprimant les dépendances entre étiquettes est linéaire, la distribution de probabilité d'une séquence d'annotations y selon une séquence observable x est donnée par :

$$p(y|x) = \frac{1}{Z(x)} \prod_t \exp\left[\sum_{k=1}^{K} \lambda_k f_k(t, y_t, y_{t-1}, x)\right] \quad (1)$$

où $Z(x)$ est un facteur de normalisation dépendant de x et où les K traits f_k sont des fonctions à valeur dans $\{0, 1\}$ fournies par l'utilisateur. Les poids λ_k associés aux différents traits f_k sont les paramètres du modèle déterminés par l'apprentissage. L'une des implémentations les plus efficaces des CRF linéaires est fournie par Wapiti [1] (Lavergne *et al.*, 2010), qui implémente la plupart des algorithmes d'entraînement couramment utilisés en plus de permettre la sélection de traits pertinents.

Dans nos expériences, nous nous concentrerons sur le French Treebank (FTB) (Abeillé *et al.*, 2003) annoté en entités nommées (Sagot *et al.*, 2012).

2.2 Le FTB

Le French Treebank, ou FTB (Abeillé *et al.*, 2003), est un recueil de phrases issues du journal Le Monde de 1989 à 1995 annotées en constituants. Sa taille est de 12351 phrases pour 350931 . Dans le cadre de cette expérience, nous avons utilisé sa version annotée en entités nommées fournie par

1. disponible depuis : https://github.com/Jekub/Wapiti

Sagot *et al.* (2012), dont les caractéristiques principales sont données dans le tableau 1. On y distingue sept types d'entités principaux : *Company* (les entreprises), *Location* (les lieux tels que les villes ou les pays), *Organization* (les organisations à but non lucratif), *Person* (les personnes réelles), *Product* (les produits), *FictionCharacter* (les personnages fictifs, de série TV ou bande dessinée par exemple) et finalement les *PointOfInterest* (les points d'intérêt tels que l'Opéra). Le découpage du corpus suit le protocole entraînement–développement–test défini par Crabbé & Candito (2008).

	Entraînement	Développement	Test
Phrases	9881	1235	1235
Entités	9235	1271	1173

TABLE 1 – Une vue d'ensemble du FTB annoté en entités nommées

Typiquement, les systèmes de type CRF sur des tâches de reconnaissance d'entités nommées intègrent de la connaissance en domaine à l'aide de lexiques. De plus, près de 40% des entités du corpus de test sont absentes du corpus d'entraînement. Nous commencerons donc par détailler l'intégration de ce type de connaissances dans notre système.

3 Intégrer de la connaissance en domaine

Nombre de systèmes par apprentissage intègrent des connaissances d'une façon ou d'une autre, ceux n'en intégrant aucune peinant à être compétitifs, comme l'a conclu Jungermann (2007). Cette connaissance peut autant être lexicale et morphologique (Raymond & Fayolle, 2010; Holat *et al.*, 2016), une représentation des mots calculée sur des volumes importants de données textuelles (Collobert & Weston, 2008; Ratinov & Roth, 2009; Lample *et al.*, 2016) ou le résultat de l'apprentissage joint de plusieurs tâches (Collobert & Weston, 2008; Luo *et al.*, 2015).

Une des premières sources de connaissances externes utilisée pour la REN est un ensemble de lexiques, généralement un par type de sortie. Les traits issus de l'utilisation d'un ensemble de lexiques, appelé « connaissances *a priori* », sont un point important de Raymond & Fayolle (2010), qui nous servira donc de point de départ. Ces traits sont en fait la combinaison de plusieurs éléments, que l'on peut rapprocher des motifs décrits par Holat *et al.* (2016). Ils sont générés en trois étapes :

1. Les connaissances *a priori* sont appliquées. Ici, chaque terme reconnu par un lexique est marqué avec l'identifiant de ce dernier.
2. Les mots dits « importants » (qui ont une forte information mutuelle avec une classe de sortie) sont laissés tels quels.
3. La partie du discours (*Part Of Speech*, POS) est utilisée pour les mots non reconnus dans les deux étapes précédentes.

Nous avons légèrement modifié ces traits ici : à la place des mots importants, nous avons créé des lexiques de termes déclencheurs pour chaque type d'entité principal. Il existe deux différences entre les mots importants et les termes déclencheurs. Là où les mots importants sont générés automatiquement, les lexiques de termes déclencheurs ont été constitués manuellement et enrichis à l'aide de noms communs récupérés dans le contexte proche des entités dans l'ensemble d'entraînement. La nature de l'information est également différente : pour un mot important, sa forme fléchie est utilisée, alors que nous utilisons l'identifiant du lexique est utilisé pour un terme déclencheur. L'utilisation

des formes fléchies pour les mots importants pose à notre avis deux problèmes. Premièrement cette liste est figée et toute modification impose de réapprendre le modèle, alors qu'un terme déclencheur peut simplement être ajouté à la liste correspondante. De plus, il n'est pas garanti que l'ensemble des mots importants présents dans l'ensemble d'apprentissage soit exhaustif. Si un mot important est absent du corpus d'apprentissage, les CRF seront incapables de l'utiliser pendant l'annotation. À l'inverse, les termes absents des lexiques peuvent être rajoutés au fur et à mesure, donnant aux CRF des informations qu'ils peuvent utiliser.

Le tableau 2 présente des traits générés par la procédure précédente. Les lignes commençant par « l>X » signifient que « X » a été utilisé pour les mots qu'aucun lexique n'a reconnu. Nous voulions évaluer l'utilisation de deux ressources en plus du POS, l'une plus précise — les mots — et l'autre plus générale — le chunking (Abney, 1991). L'intuition est que les mots permettent d'avoir des contextes plus forts, tandis que le chunking permet une plus grande généralisation — les entités nommées correspondant généralement à des chunks nominaux ou prépositionnels.

Mots	la	société	Warner	fondée	par	les	frères	Warner
lexiques	∅	company.trigger	last-name	∅	∅	∅	∅	last-name
l>mots	la	company.trigger	last-name	fondée	par	les	frères	last-name
l>POS	DT	company.trigger	last-name	ADJ	PRP	DET	NC	last-name
l>chunks	NP	company.trigger	last-name	AP	B-PP	I-PP	I-PP	last-name

TABLE 2 – Exemple de traits générés depuis un répertoire de lexiques.

Nous nous concentrerons ici sur la gestion des termes ambigus dans la source de connaissance, c'est-à-dire ceux qui apparaissent dans au moins deux lexiques différents. Nous évaluons pour cela deux méthodes de gestion de ces ambigüités, lesquelles ne figuraient pas dans les travaux originaux mais qui peuvent causer de grandes différences de résultats, comme nous le verrons dans la section 4.2. Il n'est pas rare qu'un terme puisse être ambigu, dans le sens où il apparaît dans plusieurs lexiques. C'est par exemple le cas de « Paris », qui peut référer à une ville (lieu) ou à un prénom (personne). Dans un tel cas, deux possibilités s'offrent à nous. La première consiste à effectuer une analyse ambigüe, où chaque terme reconnu par plusieurs lexiques se verra attribuer plusieurs classes. Ce type d'analyse a l'avantage de distinguer les termes sûrs de ceux ambigus et permet donc à l'algorithme d'effectuer une analyse plus fine, mais elle a également comme inconvénient que les ambigüités peuvent ne pas être observées à l'apprentissage, laissant le système démuni en phase d'annotation. Une seconde approche consiste à établir une relation d'ordre sur les lexiques, notée $>$, avec $x > y$ se décrivant comme « x est plus prioritaire que y ». Prenons deux lexiques x et y ainsi qu'un terme t tels que $t \in x \cap y$ et $x > y$. Lorsque nous rencontrons le terme t dans notre corpus, ce dernier prendra alors systématiquement la classe associée à x. Cette approche a l'avantage de ne laisser aucune ambigüité et de fournir des traits plus simples et moins silencieux au CRF, même si ces derniers sont moins précis. Par la suite, nous appellerons ces connaissances *a priori*, lorsqu'elles sont classées et triées, un *répertoire* de lexiques, dont nous étudierons l'intégration dans un CRF.

La clasification utilisée pour la REN sur le FTB est donnée dans la figure 1. Nous avons également utilisé la classification des verbes de Dubois & Dubois-Charlier (1997), qui définit en fait deux classifications des verbes différentes : une générique (communication, don/privation, auxiliaires, etc. . .) et une sémantique (humain, animé, non-animé, etc. . .). Nous n'avons pas obtenu de meilleurs résultats en intégrant cette classification dans le *répertoire*, raison pour laquelle elle n'y figure pas.

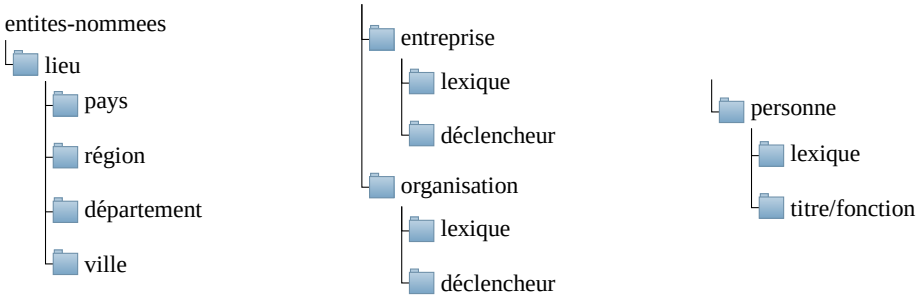


FIGURE 1 – clasification des lexiques utilisée dans les systèmes présentés. Plus un lexique est haut, plus ce dernier est prioritaire.

4 Expériences

4.1 Résultats préliminaires sur le FTB

Nous avons utilisé le répertoire décrit dans la section 3 pour établir les priorités de nos lexiques en cas d'ambigüité. Nous avons appliqué les lexiques en ignorant la capitalisation des mots, cette information étant plus pertinente à utiliser comme trait dans notre CRF. En effet, ne pas considérer la casse au moment d'appliquer les lexiques maximise leur couverture, ajouter les traits de capitalisation dans le CRF lui permettant alors d'apprendre à distinguer les reconnaissances bruitées.

L'ensemble des traits décrits dans ce papier ont été générés dans une fenêtre de $[-2, 2]$ autour du token courant. Notre CRF *étalon* utilise les mots, les lexiques comme traits spécifiques (un trait booléen par lexique), des préfixes et suffixes jusqu'à une taille de 5 caractères ainsi que divers traits booléens comme des informations de capitalisation. Dans notre expérience visant l'intégration du répertoire de lexiques, nous avons utilisé la configuration permettant d'obtenir les meilleurs résultats dans la section 4.2. Nous avons également intégré successivement les préfixes et suffixes des mots, les noms voisins ainsi que le verbe suivant (plus précisément, le premier verbe non suivi d'un autre verbe afin de ne pas capturer les auxiliaires des participes passés). Les classes de verbes sont celles définies par Dubois & Dubois-Charlier (1997). Nous avons utilisé la classe générique et la classe sémantique. Si un verbe avait plusieurs acceptions ayant des classes différentes, nous avons choisi la première, qui correspond à celle du verbe dans son sens premier.

Les résultats sont détaillés dans le tableau 3. Les expériences (a) à (c) montrent l'utilisation de mots et du répertoire uniquement. De façon assez prévisible, les mots donnent une très forte précision mais un mauvais rappel, tandis que les POS donnent un meilleur rappel. Les chunks semblent être à mi-chemin entre ces deux informations en termes de qualité, ce qui peut paraître étonnant étant donné qu'il s'agit d'une information plus générale que le POS. Cela vient du fait que la plupart des entités nommées correspondent à un chunk nominal sans le déterminant, la fin d'une entité correspondant à la fin d'un chunk nominal. La majorité des erreurs faites dans (c) sont sur des entités de taille 1 absentes des lexiques, où l'information du chunk n'est donc pas utilisable. Les expériences (d) et (e) montrent que le POS semble avoir un léger avantage par rapport au chunking en termes de qualité, particulièrement en termes de rappel. La combinaison des différentes expériences de (a) à (c) n'a pas donné d'amélioration significative du modèle, nous avons donc utilisé l'expérience (b) comme base

Expérience	Précision	Rappel	F-mesure
CRF *étalon*	85.89	76.88	81.13
a. o/mots	**89.42**	69.20	78.02
b. o/POS	85.4	76.88	80.92
c. o/chunking	88.95	74.83	81.28
d. (b) +préfixes/suffixes	86.48	78.58	82.34
e. (c) +préfixes/suffixes	87.26	77.73	82.22
f. (d) +noms voisins	85.86	78.75	82.15
g. (d) +prochain verbe (forme)	86.21	**78.92**	82.41
h. (d) +classes prochain verbe	85.89	**78.92**	82.26
(f) + (g)	86.03	78.84	82.28
(f) + (h)	86.77	**78.92**	**82.66**

TABLE 3 – Les premiers résultats obtenus sur le FTB. En gras sont marqués les meilleurs scores pour la colonne.

pour les autres expériences. L'ajout des verbes et classes de verbes n'a pas donné d'amélioration significative et donnait même plus souvent lieu à une dégradation des résultats.

De nombreux termes présent dans le répertoire sont ambigus, dans le sens où ils peuvent apparaître dans plusieurs lexiques. Dans la section suivante, nous détaillerons comment nous avons géré ces ambiguïtés et comment cette gestion peut influer sur la qualité finale du modèle.

4.2 Gestion de l'ambiguïté des lexiques

Comme dit dans les parties précédentes, le choix des priorités quant aux différents types du répertoire est capital. Dans cette section, nous détaillerons les différences de résultats que ces changements peuvent amener. À cet effet, nous avons simplement évalué les différents ordonnancements ainsi qu'en effectuant une analyse ambigüe, où plusieurs éléments d'un répertoire peuvent reconnaître un même mot ou groupe de mots. Ici, nous évaluons l'impact sur les résultats obtenus sur le FTB en changeant l'ordre de priorité des lexiques au moment de générer les traits relatifs aux répertoires. Nous n'avons pas inclus l'influence entre organisation et entreprise car ces dernières n'avaient aucun terme en commun. L'analyse dite ambigüe consiste à expliciter l'ensemble des ambigüités présentes dans les lexiques, ce qui se fait ici en effectuant la concaténation des différents lexiques ayant reconnu un mot. Ainsi, pour chaque mot du texte, nous pouvons récupérer l'ensemble des classes auxquelles il peut appartenir et ainsi avoir des termes reconnus de façon non-ambigüe ainsi que des termes reconnus de façon ambigüe. Le contexte doit alors être utilisé pour trouver la classe la plus appropriée. Le tableau 4 détaille les résultats obtenus en modifiant l'ordre de priorité des lexiques ainsi qu'en effectuant une analyse ambigüe. Comme nous pouvons le remarquer, cet ordre influe de façon significative sur la qualité globale du résultat. Nous pouvons cependant déduire certaines tendances quant à l'ordre des lexiques. Par exemple, le lexique des lieux doit être prioritaire sur celui des personnes, et il en va de même pour le lexique des organisations et entreprises. Notre lexique des personnes ayant un bruit assez important, lui donner une priorité faible permet de privilégier les autres lexiques, plus sûrs. L'influence est moindre entre les lieux et les organisations/entreprises, car il y a peu d'intersection entre ces derniers.

Expérience	Précision	Rappel	F-mesure
$P > C\&O > L$	85.98	76.79	81.13
$P > L > C\&O$	85.58	76.96	81.04
$L > P > C\&O$	85.47	77.89	81.50
$L > C\&O > P$	85.80	78.49	81.98
$C\&O > P > L$	85.66	76.36	80.74
$C\&O > L > P$	**86.77**	**78.92**	**82.66**
Ambigüe	85.39	76.79	80.86

TABLE 4 – Les résultats selon la priorité accordée aux différents lexiques. P : Person, L : Location, C&O : Company&Organization. En gras sont marqués les meilleurs scores pour la colonne.

L'analyse ambigüe est celle dont les performances sont les plus mauvaises, autant en termes de précision que de rappel. Les problèmes principaux de l'analyse ambigüe, en comparaison avec le meilleur système, sont d'abord le silence sur les lieux, suivi d'erreurs de type puis de frontières, les entités proposées tendant à être plus courtes. Si l'on observe les poids présents dans le CRF, ces derniers diffèrent peu entre traits ambigus et non-ambigus dans les cas les plus simples. Les traits ambigus quant à eux tendent à suivre l'ordre de priorité ayant donné le meilleur résultat, favorisant les *Company* et *Organization*, puis les *Location* et finalement les *Person*. Les silences sont généralement dus à des ambigüités non-observées dans le corpus d'apprentissage. Partant de ce constat, nous pouvons proposer une méthode afin d'estimer un ordre proche de l'optimal. Pour ce faire, nous apprenons un modèle où une analyse ambigüe a été effectuée. En recherchant dans ce modèle les poids attribués par le CRF pour résoudre les cas spécifiquement ambigus, il est possible d'avoir une idée des lexiques plus ou moins prioritaires. L'inconvénient d'une analyse ambigüe demeure dans la combinatoire des possibles ambigüités, qui fait que toutes ne peuvent pas toujours être observées. Les cas ambigus absents du corpus d'apprentissage ne pourront pas se voir attribuer un poids par le CRF, qui sera donc incapable d'en tirer profit à l'annotation.

De manière générale, l'inconnu est l'une des plus grandes sources d'erreurs pour les systèmes par apprentissage. La source principale d'inconnu demeure encore les formes fléchies non observées dans le corpus d'apprentissage, en particulier si ces dernières ne font partie d'aucun lexique. Dans la section suivante, nous souhaitons évaluer l'intégration des mots inconnus dans notre CRF.

4.3 Gestion des mots inconnus

Typiquement, lorsqu'un algorithme d'apprentissage automatique rencontre un mot inconnu, il recourt à son contexte et/ou à sa morphologie afin de trouver une classe pertinente. Le caractère inconnu d'un mot offre une information intéressante pour l'analyse de ce dernier : en effet, de nombreuses entités nommées peuvent être déclenchées dans le contexte d'un mot inconnu. Afin de donner aux CRF l'information du caractère inconnu d'un mot, nous avons extrait le lexique des mots du corpus d'apprentissage et supprimé ceux trop peu fréquents, laissant ainsi le lexique des mots considérés comme connus. Un mot inconnu est alors un mot absent de ce lexique. Une fois ce lexique constitué, il est possible d'ajouter de plusieurs façons l'information « mot inconnu ». La première consiste à intégrer ce lexique directement dans le répertoire en lui accordant la plus faible priorité : ainsi, les CRF pourront obtenir une vision contextualisée d'un mot inconnu et pourra ainsi

mieux le désambigüiser. Une seconde serait de rajouter un trait booléen « mot inconnu » afin de l'utiliser comme tout autre trait booléen dans le CRF. Cela permet également, pour les traits discutés dans la section 3, de rajouter une information supplémentaire pouvant s'ajouter aux informations déjà présentes : par exemple, un « nom propre inconnu » sera une information plus pertinente que simplement un « mot inconnu ». Dans nos expériences, seul l'ajout d'un nouveau trait, où les mots appartenant au lexique des mots inconnus étaient remplacés par "_unknown_", a permis d'obtenir des gains intéressants, les meilleurs résultats étant obtenus en considérant inconnus les mots apparaissant un maximum de 4 fois dans le corpus, après avoir testé l'ensemble des valeurs entre 1 et 5. Nous avons appelé ce trait "hapax4" (inspiré de Pécheux *et al.* (2015)), dans le tableau 5.

Expérience	Précision	Rappel	F-mesure
a. CRF (section 4.2)	86.77	78.92	82.66
b. (a) +concat(mot_i,mot_{i+1}) avec i $\in$ {-2,1}	87.59	79.52	83.36
c. (a) +concat($hapax4_i$,$hapax4_{i+1}$) avec i $\in$ {-2,1}	88.15	79.95	83.85
d. (c) +concat($hapax4_i$,o/POS_0) avec i $\in$ {-2,-1,1,2}	**88.41**	**80.03**	**84.05**
(c) + mots inconnus = classe	86.80	79.69	83.10

TABLE 5 – Les résultats en intégrant les mots inconnus. En gras sont marqués les meilleurs scores pour la colonne.

L'un des inconvénients principaux des CRF réside dans leur nature fondamentalement locale. Afin de pallier ce manque inhérent au modèle, nous avons testé deux approches pour améliorer la consistance des annotations au niveau global.

4.4 Consistance des annotations

L'un des défauts des systèmes par apprentissage vient du fait que l'inférence se fait de façon purement locale, une hypothèse d'indépendance étant faite afin de rendre le modèle calculable en pratique. Divers travaux ont été effectués afin de modéliser des dépendances non-locales et assurer la consistance des annotations, modélisant généralement des dépendances à l'échelle du document et du corpus (Krishnan & Manning, 2006; Ratinov & Roth, 2009). La première approche que nous utilisons est une simple propagation des annotations : après l'annotation du CRF, nous récupérons le lexique de chaque type d'entité retrouvé par le CRF. Pour les entrées figurant dans plusieurs lexiques, nous prenons alors l'entité qui lui est la plus fréquemment attribuée. Deux heuristiques peuvent alors être employées. La première consiste à n'appliquer ces lexiques que sur des portions non-annotées du texte, celles proposées par le CRF étant considérées comme meilleures de façon systématique. La seconde consiste à ne mettre à jour les annotations du CRF que dans le cas où une chaîne plus longue a été trouvée, aucun retypage des séquences de même taille n'étant effectué. Cette approche a l'avantage d'être très simple à mettre en place et d'être intuitivement sous-optimale : elle permet donc d'établir une référence des gains obtenables par cette approche. Nous avons également testé l'approche en deux passes à l'aide des traits *token majority* et *entity majority* tels que décrits dans Krishnan & Manning (2006); Mao *et al.* (2007); Ratinov & Roth (2009). Le trait *token majority* considère la classe majoritairement associée à chaque token indépendamment, en ignorant les le schéma BIO. Par exemple, si « Paris » apparait deux fois en tant qu'organisation et une en tant que lieu, alors le trait *token majority* attribuera la valeur « organisation » à toutes les occurrences de « Paris ». Le trait *entity majority* est analogue à *token majority*, mais considère les entités retrouvées

par le premier CRF. Si par exemple « Calvin Klein » a été annoté trois fois en tant qu'entreprise et deux fois en tant que personne, alors toutes les occurences de « Calvin Klein » seront annotées en tant qu'entreprise. Les égalités ont été résolues en utilisant les priorités établies dans la section 4.2 et les chevauchement ont été gérés selon la règle de « la première chaine la plus longue ».

Expérience	Précision	Rappel	F-mesure
CRF (section 4.3)	**88.41**	80.03	84.05
heuristique1	87.89	**82.34**	**85.02**
heuristique2	87.80	82.26	84.94
deux passes	87.72	81.66	84.58

TABLE 6 – Les résultats selon les différentes méthodes de propagation. En gras sont marqués les meilleurs scores pour la colonne.

Le tableau 6 résume les résultats obtenus avec les différentes méthodes de propagation. L'heuristique sans mise à jour des annotations du CRF donne des résultats sensiblement meilleurs que celle pouvant modifier les annotations du CRF. En observant les annotations, nous avons remarqué que cette différence était principalement due à des inconsistances d'annotation dans le gold standard pour certaines organisations, les résultats sur les autres entités étant meilleurs. Nous observons cependant une baisse de précision à l'échelle globale, cela vient des erreurs de bruits et des types ambigus (ex : lieux contre organisations) qui se propagent par cette méthode. L'approche en deux passes telle que décrite dans Krishnan & Manning (2006); Mao *et al.* (2007); Ratinov & Roth (2009) ne nous a pas offert d'amélioration supplémentaire par rapport à notre post-traitement plus simple.

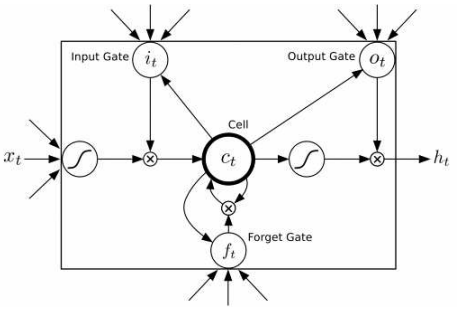
Depuis quelques années en particulier, les réseaux de neurones sont des concurrents sérieux aux CRF. Dans la section suivante, nous comparerons le système construit jusqu'ici avec l'une de ses variantes les plus efficaces, appelée le LSTM-CRF bidirectionnel.

5 Comparaison avec Bi-LSTM-CRF

Récemment, le domaine du TAL a vu un essor des réseaux de neurones *récurrents*. Ces derniers ont été créés pour traiter des séquences de données selon un principe simple : pour chaque élément d'une séquence, son résultat est injecté dans l'élément suivant de la séquence, permettant ainsi, de proche en proche, d'incorporer le contexte à un instant donné. Le réseau récurrent le plus simple est le réseau d'Elman (Elman, 1990), où la couche cachée d'un réseau communique avec elle-même. Cependant, ces réseaux n'arrivaient pas à modéliser des dépendances longue distance et leur apprentissage était très coûteux (Bengio *et al.*, 1994).

La couche cachée *Long Short-Term Memory* (Hochreiter & Schmidhuber, 1997), appelée par la suite LSTM, s'est distinguée par sa capacité à capturer des dépendances de longue portée. Il s'agit d'une couche cachée particulière d'un réseau d'Elman utilisant une mémoire interne capable de retenir les informations pertinentes, en ajoutant et oubliant des informations au fur et à mesure selon un système de portes permettant de n'activer un ajout ou un oubli qu'au moment opportun. Une illustration de la variante *peephole* (Gers & Schmidhuber, 2000) d'une cellule de LSTM ainsi que les formules des différentes portes sont données dans la figure 2. Nous comparerons notre CRF avec une variante de ce réseau faisant partie des systèmes les plus performants, appelée LSTM-CRF bidirectionnel (Bi-

LSTM-CRF) (Huang *et al.*, 2015; Lample *et al.*, 2016), qui sont parmi les systèmes état-de-l'art. Il s'agit d'une variante de réseau de neurones à laquelle est ajoutée un CRF, une de ses grandes forces étant sa capacité de généralisation (Augenstein *et al.*, 2017), leur permettant d'obtenir de très bons scores sur les entités inconnues. Dans cette partie, nous souhaitons effectuer une comparaison entre les différents CRF proposés ici et le réseau Bi-LSTM-CRF.



$$
\begin{aligned}
f_t &= \sigma(W_f \times x_t + U_f \times c_{t-1} + b_f) \\
i_t &= \sigma(W_i \times x_t + U_i \times c_{t-1} + b_i) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_c \times x_t + b_c) \\
o_t &= \sigma(W_o \times x_t + U_o \times c_{t-1} + b_o) \\
h_t &= o_t \odot \sigma(c_t)
\end{aligned}
\tag{2}
$$

FIGURE 2 – représentation d'une cellule de LSTM et les formules des différentes portes. σ représente la fonction sigmoïde. $\odot$ est le produit d'Hadamard. $\times$ est le produit matriciel. Illustration tirée de Graves *et al.* (2013)

Pour notre comparaison, nous avons utilisé l'implémentation de LSTM de Lample *et al.* (2016), qui a deux particularités. La première est qu'il utilise la variante de LSTM donnée sur la figure 2 et fusionne les porte d'oubli (f_t) et d'entrée (i_t). La seconde combine en réalité deux LSTM bidirectionnels. Un premier Bi-LSTM est employé pour donner une représentation des mots étant donné leur contexte gauche et droit dans la phrase, cela permet donc de capturer des informations de nature plus syntaxique. À cette représentation contextuelle des mots est alors concaténée une seconde, endogène, à l'aide d'un Bi-LSTM au niveau des caractères d'un mot, permettant de modéliser des informations relatives à la morphologie dudit mot. Ainsi, pour chaque mot, ce réseau modélise des informations à la fois syntaxiques et morphologiques, le rendant très intéressant pour des tâches telles que la REN. Un schéma de ces deux niveaux de représentation est donné dans la figure 3.

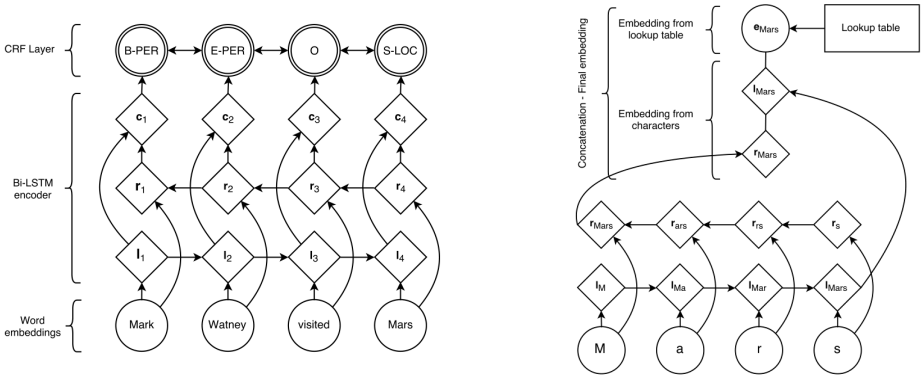


FIGURE 3 – À gauche : le Bi-LSTM-CRF pour une phrase. À droite : Bi-LSTM pour un mot. Illustrations tirées de Lample *et al.* (2016)

5.1 Expériences de comparaison

Pour cette comparaison, nous avons utilisé le Bi-LSTM-CRF proposé par Lample *et al.* (2016) [2]. Afin d'obtenir une comparaison complète, nous l'avons tout d'abord entraîné sans préentraînement puis en utilisant des représentations préappris sur des corpus issus du web [3]. Ils ont été appris à l'aide de word2vec (Mikolov *et al.*, 2014) sur le corpus FrWac du projet WaCky (Baroni *et al.*, 2009) et sur un dump de Wikipedia français. Ces derniers donnant des résultats significativement moins bons que ceux appris sur FrWac, nous ne les détaillerons pas ici. Les représentations appris sur le corpus FrWac ont une taille de 200, les mots apparaissant moins de 100 fois ont été considérés comme inconnus.

Pour notre Bi-LSTM-CRF, nous avons utilisé des vecteurs de taille 200 pour les mots et de taille 25 pour les caractères. Nous avons utilisé un dropout de 0,5. Les modèles ont été entraînés pendant 50 itérations selon une descente de gradient stochastique, le modèle final étant celui ayant maximisé la F1-mesure sur les entités dans le corpus de développement. Nous avons également évalué le gain obtenu en utilisant les traits de Raymond & Fayolle (2010) en tant qu'information supplémentaire de notre réseau, en leur attribuant une taille de 32, taille donnant les meilleurs résultats dans nos expériences. Les résultats comparatifs entre les CRF et les LSTM sont donnés dans le tableau 7.

Système	Connues			Inconnues			Global		
	P	R	F	P	R	F	P	R	F
CRF *étalon* (section 4.1)	95.04	92.34	93.67	68.68	53.53	60.17	85.89	76.88	81.13
CRF (section 4.3)	**97.21**	93.90	95.53	72.63	59.20	65.17	**88.41**	80.03	84.05
+heuristique1 (section 4.4)	96.83	**95.46**	**96.14**	72.46	62.53	67.13	87.89	82.34	85.02
LSTM-CRF (*base*)	96.10	94.33	95.20	64.21	54.18	58.77	84.53	78.33	81.31
+ o/POS	95.95	94.04	94.99	70.13	59.13	64.27	86.56	80.20	83.26
LSTM-CRF (FrWac)	96.25	94.61	95.42	69.44	60.81	64.84	86.30	81.14	83.64
+ o/POS	96.11	94.61	95.35	**74.50**	64.45	69.12	88.16	82.59	85.29
+ heuristique1	95.98	94.89	95.44	73.09	**67.45**	**70.16**	87.23	**83.96**	**85.57**
Dupont & Tellier (2014)	?	?	?	?	?	?	86.38	80.30	83.23

TABLE 7 – comparaison entre les différents CRF et LSTM-CRF. En gras sont marqués les meilleurs scores pour la colonne.

Les seuls résultats à notre connaissance sur la reconnaissance d'entités nommées sur le FTB sont ceux de Dupont & Tellier (2014), qui utilisent un CRF dont les traits sont proches de ceux de notre CRF *étalon*. Malgré la qualité inférieure de notre CRF *étalon* par rapport au leur, l'utilisation des traits détaillés ici nous a permis d'obtenir une qualité finale supérieure, ce qui montre bien leur pertinence. Nos Bi-LSTM-CRF intégrant des informations extérieures, autant sous la forme des traits détaillés dans la section 3 que de représentations préappris, témoignent également de l'efficacité des réseaux de neurones récurrents.

Les deux systèmes ont des f-mesures globales comparables, avec cependant un léger avantage pour le Bi-LSTM-CRF, qui se distingue surtout sur les entités inconnues où il obtient une qualité significativement supérieure à celle du CRF, résultats cohérents avec ceux décrits par Augenstein *et al.* (2017). L'ajout du répertoire de lexiques a permis au réseau de neurones d'obtenir une représentation plus

2. disponible à l'adresse : https://github.com/glample/tagger
3. disponibles à l'adresse : http://fauconnier.github.io

générale, en témoigne le gain de 5 à 6 points de f-mesure sur les entités inconnues, et de moins de 0,5 points sur les entités connues. L'ajout de règles de post-traitement simples a permis l'amélioration des deux meilleurs modèles respectifs, le CRF ayant plus bénéficié de ce gain que le LSTM-CRF. Cette différence s'explique par le côté plus précis du CRF, qui aura plus tendance à n'annoter une entité que dans un contexte sûr, une même entité n'étant alors annotée qu'à certains endroits. Une autre source de gain du Bi-LSTM-CRF vient des représentations préentraînés sur le corpus FrWac. Il est possible d'ajouter des représentations des mots dans un CRF, typiquement via l'entraînement de clusters de Brown (Brown *et al.*, 1992). Nous avons intégré 1000 clusters de Brown appris sur un dump de Wikipédia français, mais ces derniers n'ont pas amélioré nos résultats. Nous n'avons pas pu les entraîner sur le corpus FrWac, le coût en temps étant prohibitif.

L'ajout de la consistance des annotations en tant que post-traitement simple n'améliore pas significativement les résultats obtenus par les Bi-LSTM-CRF, en raison d'une forte baisse de la précision (-0.93). Cela vient du fait que le réseau de neurones est un système ayant tendance à être plus bruyant qu'un CRF, qui lui aura plus tendance à être silencieux. Cela se remarque également dans l'amélioration du rappel, moins importante pour le Bi-LSTM-CRF (+1.37) que pour le CRF (+2.31), autant sur les entités connues qu'inconnues. Cela suggère que le CRF a besoin d'un contexte plus fort pour annoter une entité et que Bi-LSTM-CRF a tendance à être plus consistant dans ses annotations.

6 Conclusion

Dans cet article, nous avons exploré diverses méthodes pour intégrer de la connaissance en domaine ainsi que des règles basiques dans un CRF. Nous avons utilisé comme point de départ les traits définis par (Raymond & Fayolle, 2010), que nous avons ensuite progressivement enrichis avec de nouvelles informations. Nous avons comparé cette méthode avec celle, classique, de l'ajout de lexiques à l'aide de traits booléens dissociés et avons constaté que leur combinaison améliorait les résultats car elle permet de contextualiser l'information des lexiques. Nous avons également étudié la classification des verbes ainsi que la meilleure manière des les intégrer au CRF. Nous avons aussi exploré la gestion des mots inconnus, leur ajout a permis d'obtenir une améliotation dans certaines conditions. Nous avons ensuite comparé notre CRF avec un Bi-LSTM-CRF ayant obtenu de meilleurs résultats avec moins de traits. L'une des premières tâches à effectuer serait d'ajouter les traits du CRF manquant au Bi-LSTM-CRF.

Nous prévoyons d'enrichir les lexiques que nous avons constitués ainsi que les traits de notre CRF. Pour cela, nous prévoyons d'ajouter des règles à notre CRF, autant en tant que traits qui seront pondérés à l'apprentissage qu'en tant que contraintes afin de forcer, ou d'interdire, le déclenchement d'une entité ou d'une annotation incohérente. Une de ces contraintes serait d'appliquer un poids négatif arbitrairement grand aux transitions inconsistantes, comme deux étiquettes `I-<Entité>` successives de deux entités différentes. Une autre idée serait d'utiliser des systèmes comme mXs (Nouvel *et al.*, 2013) qui disposent de nombreux lexiques et règles d'annotations que nous pourrions intégrer directement en tant que traits dans notre CRF. Nous comptons également explorer d'autres types de réseaux de neurones comme les *Gated Recurrent Units* (Cho *et al.*, 2014), plus légers, ainsi que de remplacer le Bi-LSTM au niveaux des caractères par une convolution.

Nous voulons également vérifier notre approche en l'appliquant sur d'autres langues et corpus comme le CoNLL (Tjong Kim Sang & De Meulder, 2003), où les systèmes actuels sont des plus compétitifs (Passos *et al.*, 2014; Luo *et al.*, 2015; Lample *et al.*, 2016).

Références

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

ABNEY S. (1991). Parsing by chunks. In *Principle-Based Parsing*, p. 257–278 : Kluwer Academic Publishers.

AUGENSTEIN I., DERCZYNSKI L. & BONTCHEVA K. (2017). Generalisation in named entity recognition : A quantitative analysis. *arXiv preprint arXiv :1701.02877*.

BARONI M., BERNARDINI S., FERRARESI A. & ZANCHETTA E. (2009). The wacky wide web : a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, **43**(3), 209–226.

BENGIO Y., SIMARD P. & FRASCONI P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, **5**(2), 157–166.

BROWN P. F., DESOUZA P. V., MERCER R. L., PIETRA V. J. D. & LAI J. C. (1992). Class-based n-gram models of natural language. *Computational linguistics*, **18**(4), 467–479.

BUNESCU R. C. & MOONEY R. J. (2005). A shortest path dependency kernel for relation extraction. In *Proceedings of the conference HLT/EMNLP*, p. 724–731 : ACL.

CHO K., VAN MERRIËNBOER B., GULCEHRE C., BAHDANAU D., BOUGARES F., SCHWENK H. & BENGIO Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, p. 1724–1734.

COLLOBERT R. & WESTON J. (2008). A unified architecture for natural language processing : Deep neural networks with multitask learning. In *Proceedings of ICML*, p. 160–167 : ACM.

CRABBÉ B. & CANDITO M. H. (2008). Expériences d'analyse syntaxique statistique du français. In *Actes de TALN'08*.

DUBOIS J. & DUBOIS-CHARLIER F. (1997). Synonymie syntaxique et classification des verbes français. *Langages*, p. 51–71.

DUPONT Y. & TELLIER I. (2014). Un reconnaisseur d'entités nommées du français. In *TALN 2014*, p. 40–41.

ELMAN J. L. (1990). Finding structure in time. *Cognitive science*, **14**(2), 179–211.

GALLIANO S., GRAVIER G. & CHAUBARD L. (2009). The ESTER 2 evaluation campaign for the rich transcription of french radio broadcasts. In *Interspeech*, volume 9, p. 2583–2586.

GAUSSIER É. & YVON F. (2011). *Modèles statistiques pour l'accès à l'information textuelle*. Lavoisier.

GERS F. A. & SCHMIDHUBER J. (2000). Recurrent nets that time and count. In *Proceedings of the IEEE-INNS-ENNS IJCNN'00*, volume 3, p. 189–194 : IEEE.

GRAVES A., MOHAMED A.-R. & HINTON G. (2013). Speech recognition with deep recurrent neural networks. In *ICASSP 2013*, p. 6645–6649 : IEEE.

GRAVIER G., ADDA G., PAULSON N., CARRÉ M., GIRAUDEL A. & GALIBERT O. (2012). The ETAPE corpus for the evaluation of speech-based TV content processing in the french language. In *LREC-Eighth international conference on Language Resources and Evaluation*, p. n/a.

HAJISHIRZI H., ZILLES L., WELD D. S. & ZETTLEMOYER L. S. (2013). Joint coreference resolution and named-entity linking with multi-pass sieves. In *EMNLP*, p. 289–299.

HOCHREITER S. & SCHMIDHUBER J. (1997). Long short-term memory. *Neural computation*, **9**(8), 1735–1780.

HOLAT P., TOMEH N., CHARNOIS T., BATTISTELLI D., JAULENT M.-C. & MÉTIVIER J.-P. (2016). Fouille de motifs et CRF pour la reconnaissance de symptômes dans les textes biomédicaux. In *Proceedings of TALN*.

HUANG Z., XU W. & YU K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv :1508.01991*.

JUNGERMANN F. (2007). Named entity recognition without domain-knowledge using conditional random fields. In *Workshop Notes of the Machine Learning for Natural Language Processing Workshop*, p. 16–17 : Citeseer.

KOLLER D. & FRIEDMAN N. (2009). *Probabilistic graphical models : principles and techniques*. MIT press.

KRISHNAN V. & MANNING C. D. (2006). An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proceedings of the 21st COLING and the 44th annual meeting of the ACL*, p. 1121–1128 : ACL.

LAFFERTY J., MCCALLUM A. & PEREIRA F. (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML 2001*, p. 282–289.

LAMPLE G., BALLESTEROS M., SUBRAMANIAN S., KAWAKAMI K. & DYER C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv :1603.01360*.

LAVERGNE T., CAPPÉ O. & YVON F. (2010). Practical very large scale CRFs. In *Proceedings of ACL'2010*, p. 504–513 : Association for Computational Linguistics.

LUO G., HUANG X., LIN C.-Y. & NIE Z. (2015). Joint named entity recognition and disambiguation. In *Proc. EMNLP*.

MAO X., XU W., DONG Y., HE S. & WANG H. (2007). Using non-local features to improve named entity recognition recall. In *PACLIC*.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2014). word2vec.

NOUVEL D., ANTOINE J.-Y., FREEBURGER N. & SOULET A. (2013). Fouille de règles d'annotation partielles pour la reconnaissance des entités nommées. In *Proceedings of TALN*.

PASSOS A., KUMAR V. & MCCALLUM A. (2014). Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv :1404.5367*.

PÉCHEUX N., ALLAUZEN A., LAVERGNE T., WISNIEWSKI G. & YVON F. (2015). Oublier ce qu'on sait, pour mieux apprendre ce qu'on ne sait pas : une étude sur les contraintes de type dans les modèles CRF. In *Conférence sur le Traitement Automatique des Langues Naturelles*.

RATINOV L. & ROTH D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the 13th CoNLL*, p. 147–155 : ACL.

RAYMOND C. & FAYOLLE J. (2010). Reconnaissance robuste d'entités nommées sur de la parole transcrite automatiquement. In *TALN'10*.

SAGOT B., RICHARD M. & STERN R. (2012). Annotation référentielle du corpus arboré de paris 7 en entités nommées. In *Traitement Automatique des Langues Naturelles (TALN)*, volume 2.

TJONG KIM SANG E. F. & DE MEULDER F. (2003). 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*, p. 142–147 : ACL.

TKACHENKO M. & SIMANOVSKY A. (2012). Named entity recognition : Exploring features. In *KONVENS*, p. 118–127.