Approche supervisée à base de cellules LSTM bidirectionnelles pour la désambiguïsation lexicale

Loïc Vial — Benjamin Lecouteux — Didier Schwab

Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
{loic.vial,benjamin.lecouteux,didier.schwab}@univ-grenoble-alpes.fr

RÉSUMÉ. *En désambiguïsation lexicale, l'utilisation des réseaux de neurones est encore peu présente et très récente. Cette direction est pourtant très prometteuse, tant les résultats obtenus par ces premiers systèmes arrivent systématiquement en tête des campagnes d'évaluation, malgré une marge d'amélioration qui semble encore importante. Nous présentons dans cet article une nouvelle architecture à base de réseaux de neurones pour la désambiguïsation lexicale. Notre système est à la fois moins complexe à entraîner que les systèmes neuronaux existants et il obtient des résultats état de l'art sur la plupart des tâches d'évaluation de la désambiguïsation lexicale en anglais. L'accent est porté sur la reproductibilité de notre système et de nos résultats, par l'utilisation d'un modèle de vecteurs de mots, de corpus d'apprentissage et d'évaluation librement accessibles.*

ABSTRACT. *In Word Sense Disambiguation, there are still few usages of neural networks. This direction is very promising however, the results obtained by these first systems being systematically in the top of the evaluation campaigns, with a scope for improvement which seems still large. We present in this paper a new architecture based on neural networks for Word Sense Disambiguation. Our system is at the same time less difficult to train than existing neural networks, and it obtains state of the art results on most evaluation tasks in English. The focus is on the reproducibility of our system and our results, through the use of a word embeddings model, training corpora and evaluation corpora freely accessible.*

MOTS-CLÉS : *désambiguïsation lexicale, approche supervisée, LSTM, réseau neuronal.*

KEYWORDS: *Word Sense Disambiguation, Supervised Approach, LSTM, Neural Network.*

1. Introduction

La désambiguïsation lexicale (DL) est une tâche centrale en traitement automatique des langues (TAL) qui vise à attribuer le sens le plus probable à un mot donné dans un document, à partir d'un inventaire prédéfini de sens.

Il existe une multitude d'approches pour la DL, dont les approches supervisées, qui utilisent des méthodes d'apprentissage automatique couplées à de grandes quantités de données manuellement annotées, les approches à base de connaissances, qui s'appuient sur des ressources lexicales telles que des dictionnaires, des thésaurus ou des réseaux lexicaux par exemple, les approches semi-supervisées, non supervisées, ou encore les approches à base de graphes ou de similarités. Pour un état de l'art plus complet, le lecteur est invité à lire par exemple Navigli (2009).

Depuis la création des campagnes d'évaluation pour les systèmes de DL telles que SensEval-SemEval, les approches supervisées se retrouvent systématiquement dans les premières places en termes de scores obtenus (Iacobacci *et al.*, 2016). Alors que les utilisations de techniques d'apprentissage à base de réseaux de neurones se multiplient dans la plupart des champs de recherche du TAL, par exemple pour la représentation vectorielle des mots ou la traduction automatique, on retrouve aussi des approches supervisées à base de réseaux de neurones pour la désambiguïsation lexicale, et ce sont ces méthodes qui obtiennent aujourd'hui les résultats état de l'art (Yuan *et al.*, 2016 ; Kågebäck et Salomonsson, 2016 ; Raganato *et al.*, 2017).

Dans cet article, nous présentons une nouvelle approche supervisée de DL à base de réseaux de neurones, qui s'appuie sur les modèles existants et qui obtient des résultats état de l'art sur la plupart des tâches d'évaluation de la DL en anglais tout en étant moins complexe et difficile à mettre en place. De plus, nous utilisons pour la première fois l'ensemble des corpus annotés avec des sens provenant de la base lexicale WordNet (Miller, 1995), ce qui permet à notre système d'être plus robuste car plus généralisable à de nouvelles données.

En effet, les systèmes supervisés sont généralement uniquement entraînés sur le SemCor (Miller *et al.*, 1993), mais une demi-douzaine d'autres corpus annotés en sens et de grande taille existent, et ils ont été récemment regroupés dans une ressource libre nommée UFSAC[1] (Vial *et al.*, 2018). Par souci de comparaison avec les systèmes état de l'art, nous avons évalué notre approche à la fois en utilisant tous les corpus UFSAC disponibles, mais aussi en nous restreignant uniquement au SemCor.

Dans un premier temps, nous allons présenter une vue d'ensemble de la désambiguïsation lexicale ainsi que les architectures des systèmes neuronaux de DL de l'état de l'art, avec leurs avantages et inconvénients respectifs dans la section 2, avec une description et une comparaison des corpus d'entraînement pour les systèmes supervisés. Puis nous présenterons notre architecture dans la section 3. Nous décrirons ensuite le protocole expérimental suivi pour évaluer notre système dans la section 4 puis nous

1. `https://github.com/getalp/UFSAC`

détaillerons les résultats dans la section 5 avec une analyse des erreurs. Enfin nous présenterons un travail préliminaire d'amélioration de notre système de manière totalement non supervisée dans la section 6 et enfin nous conclurons dans la section 7.

2. Désambiguïsation lexicale

La désambiguïsation lexicale est une tâche visant à assigner le sens le plus probable des mots contenus dans un texte, à partir d'un inventaire prédéfini de sens. Par exemple, si l'on considère la phrase « La souris mange le fromage. », un bon système de désambiguïsation lexicale devrait plutôt assigner au mot « souris » le sens du rongeur, plutôt que le sens du périphérique informatique.

Cette façon de clarifier explicitement un texte en y assignant un label de sens à chacun de ces mots a montré son efficacité dans l'amélioration de diverses tâches. Dans les travaux de Zhong et Ng (2012) par exemple, un système de DL est intégré à un système de traduction automatique statistique et permet d'améliorer ses performances. Plus récemment, les travaux de Rios *et al.* (2017) et Liu *et al.* (2018) ont permis de montrer les difficultés pour un système de traduction automatique neuronale de faire un choix lorsqu'il est confronté à un mot polysémique, et ont intégré avec succès les prédictions d'un système de DL pour améliorer ses performances. Enfin, les travaux de Chan *et al.* (2007a) ont montré que l'utilisation des prédictions d'un système de DL pour la création d'un modèle de langue pour la recherche d'informations améliore de façon significative un système état de l'art sur cette tâche.

2.1. *Approches pour la désambiguïsation lexicale*

Les approches pour la désambiguïsation lexicale sont multiples et sont généralement classées en plusieurs catégories en fonction de la nature des ressources utilisées et de leur quantité. On distingue ainsi :

– les approches à base de connaissances, qui s'appuient principalement sur des ressources telles que des dictionnaires, réseaux lexicaux ou graphes par exemple. On retrouve ici les approches à base de mesures de similarité sémantiques comme celles exploitant l'algorithme de Lesk (Lesk, 1986), ou les mesures exploitant un graphe comme celui de BabelNet (Navigli et Ponzetto, 2010) ;

– les approches supervisées, qui exploitent un grand nombre d'exemples de mots annotés manuellement ou automatiquement en sens, et qui sont généralement liés à une méthode d'apprentissage automatique telle qu'un classifieur linéaire (Chan *et al.*, 2007b ; Zhong et Ng, 2010) ou plus récemment un réseau de neurones récurrents (Yuan *et al.*, 2016 ; Raganato *et al.*, 2017) ;

– les approches non supervisées, qui réalisent de l'induction de sens, c'est-à-dire que seuls des textes non annotés sont utilisés, et les différents sens des mots sont induits en fonction de leurs contextes (Brody et Lapata, 2008 ; Yarowsky, 1995).

Bien sûr les frontières entre ces catégories sont parfois floues, on retrouve par exemple des approches à base de connaissances améliorées à l'aide de corpus annotés

en sens (Vial *et al.*, 2016), une catégorie de méthodes dites faiblement supervisées qui visent l'utilisation d'un minimum de données annotées en sens, et des approches semi-supervisées qui exploitent des données annotées ainsi que des données non annotées dans leurs approches (Yuan *et al.*, 2016).

2.2. *Évaluation des systèmes de désambiguïsation lexicale*

L'évaluation des différents systèmes de DL est rendue possible notamment grâce à la création de la première campagne d'évaluation nommée SensEval (Kilgarriff, 1998), centrée sur la tâche de DL. Après deux autres éditions de SensEval, cette campagne a été renommée en SemEval et touche maintenant à des tâches diverses allant de la reconnaissance d'entités nommées à de l'analyse de sentiments. Les tâches de DL sont pour autant toujours régulièrement présentes (Edmonds et Cotton, 2001 ; Snyder et Palmer, 2004 ; Navigli *et al.*, 2007 ; Navigli *et al.*, 2013 ; Moro et Navigli, 2015).

Dans ces tâches de DL, les systèmes comparés doivent annoter en sens tous les mots d'un document ou une partie d'entre eux. Les annotations sont ensuite comparées aux références et les performances des systèmes sont mesurées selon les métriques couverture (C), précision (P), rappel (R) et F-mesure (F1).

La principale langue cible de ces tâches d'évaluation est l'anglais, mais il existe aussi des tâches pour la désambiguïsation d'autres langues comme le français ou l'italien, par exemple (Navigli *et al.*, 2013 ; Moro et Navigli, 2015). Toutefois, une des raisons principales pour expliquer la prédominance de l'anglais comme langue cible de la désambiguïsation lexicale est l'existence de WordNet (Miller, 1995), une base de données lexicale pour l'anglais de grande qualité, utilisée souvent comme l'inventaire de sens de référence pour annoter les documents.

2.3. *Architectures neuronales pour la désambiguïsation lexicale*

On retrouve dans les approches neuronales pour la DL notamment trois travaux majeurs : le modèle de Kågebäck et Salomonsson (2016), le modèle de Yuan *et al.* (2016) et celui de Raganato *et al.* (2017).

Kågebäck et Salomonsson (2016) sont les premiers à mettre en œuvre un réseau de neurones à base de vecteurs de mots et de cellules récurrentes de type LSTM pour prédire le sens d'un mot cible. Dans leurs travaux, un modèle n'est capable de prédire le sens que d'un seul lemme du dictionnaire, et donc chaque lemme a son modèle propre de classification qui est entraîné séparément. Leur système est évalué sur les tâches de *lexical sample* des campagnes d'évaluation SensEval 2 et SensEval 3 dans lesquelles plusieurs instances d'un faible nombre de lemmes distincts sont à annoter en sens, mais il n'est pas évalué sur les tâches de désambiguïsation lexicale *all words* où tous les mots d'un document doivent être annotés en sens.

Le principal avantage de leur modèle est donc sa petite taille. En effet la couche de sortie de leur réseau est de la taille du nombre de sens pour le lemme cible, le

nombre de sens moyen pour les mots polysémiques dans WordNet étant d'environ 3[2]. Les couches cachées de cellules LSTM sont, elles aussi, très petites, avec seulement deux couches de taille 74 chacune. Il est cependant peu aisé d'entraîner ce système à annoter tous les mots d'un document car chaque lemme doit avoir son propre modèle.

Dans le modèle de Yuan *et al.* (2016), un réseau neuronal à base de cellules LSTM est utilisé comme modèle de langue, pour prédire un mot d'une séquence en fonction de son contexte. Un apprentissage supervisé sur des corpus annotés en sens est ensuite effectué pour que leur système apprenne à distinguer les différents sens d'un mot en fonction des mots prédits par leur modèle de langue. Dans un second temps, les auteurs proposent une méthode de propagation de labels pour augmenter leurs données annotées en sens et obtenir ainsi leurs meilleurs résultats. Cette méthode consiste à chercher dans des corpus non annotés de nouvelles phrases, proches des phrases de leur corpus annoté, en s'appuyant sur sur une mesure de similarité cosinus entre les représentations vectorielles de ces phrases. Les annotations en sens sont ensuite propagées de la phrase initialement annotée vers l'autre phrase.

Dans cet article, les auteurs comparent les performances de différents modèles, entraînés sur le SemCor ou l'OMSTI, avec et sans leur propagation de labels, et obtiennent des résultats état de l'art sur la plupart des tâches. Le principal problème de leur approche est la reproductibilité des résultats. En effet, leur modèle de langue est entraîné sur un corpus privé d'actualités (*news*) d'une taille de 100 milliards de mots, et ils ont utilisé pour leur propagation de labels des phrases prises aléatoirement sur le Web, sans en spécifier la source plus précisément.

Enfin, l'architecture de leur modèle de langue ne permet de prédire le sens que d'un seul mot à la fois pour une séquence donnée. Il est donc nécessaire d'exécuter leur modèle pour chaque mot d'une phrase afin de tous les annoter.

Raganato *et al.* (2017) proposent également un modèle à base de LSTM mais qui prédit directement un label pour chacun des mots donnés en entrée. Ce label fait partie d'un ensemble comprenant tous les sens d'un dictionnaire ainsi que tous les mots observés pendant l'entraînement. Ils augmentent ensuite leur modèle avec une couche d'attention et effectuent un entraînement multitâche dans lequel leur réseau prédit à la fois un sens ou un mot, un label de partie du discours et un label sémantique.

Cette architecture est la seule permettant d'annoter tous les mots d'une séquence en une passe et l'entraînement de leur modèle s'est effectué sur le SemCor uniquement. Leur réseau associe à un mot en entrée un label appartenant à l'ensemble des sens de leur inventaire de sens ainsi que l'ensemble des mots observés pendant l'entraînement. Cette approche permet à leur modèle d'apprendre à prédire un label de sens lorsque le mot est annoté dans le corpus d'entraînement, et un label de mot lorsque le mot n'est pas annoté (si c'est un mot-outil par exemple). L'inconvénient de l'approche est qu'elle n'est pas applicable pour l'apprentissage sur un corpus partiellement annoté en

2. `https://wordnet.princeton.edu/documentation/wnstats7wn`

sens. En effet, pour ce type de corpus, leur modèle va apprendre à recopier des mots non annotés, au lieu d'essayer de les annoter.

2.4. *Corpus annotés en sens pour l'entraînement de systèmes supervisés*

Les corpus annotés en sens sont la matière première des approches supervisées, et leur utilisation ou non en tant que données d'entraînement est ainsi un choix crucial.

Le SemCor (Miller *et al.*, 1993) est indéniablement le corpus le plus utilisé en tant que données d'entraînement dans les systèmes supervisés. Il est en effet la source principale de données annotées des meilleurs systèmes depuis la création des premières campagnes d'évaluation (Hoste *et al.*, 2001 ; Decadt *et al.*, 2004) jusqu'aux systèmes les plus récents (Yuan *et al.*, 2016 ; Raganato *et al.*, 2017).

D'autres corpus sont aussi utilisés dans un plus faible nombre de travaux. C'est le cas, par exemple, du DSO (Ng et Lee, 1997) qui est utilisé en plus du SemCor dans le système NUS-PT (Chan *et al.*, 2007b), le meilleur système de la campagne d'évaluation SemEval 2007 (Navigli *et al.*, 2007) ou de l'OMSTI (Taghipour et Ng, 2015), un corpus d'un million de mots annotés en sens, utilisé notamment par Iacobacci *et al.* (2016) en remplacement du SemCor et qui permet aux auteurs d'obtenir leurs meilleurs résultats. On peut aussi citer le MASC (Ide *et al.*, 2008), un corpus utilisé dans le récent travail de Yuan *et al.* (2016) pour l'évaluation de leur système, l'OntoNotes (Hovy *et al.*, 2006) ou encore le WordNet Gloss Tagged (Miller, 1995).

On remarque ainsi qu'aucun travail n'utilise l'ensemble des corpus annotés en sens disponibles, et rares sont ceux qui justifient l'emploi d'un corpus plutôt qu'un autre. Jusqu'à récemment, ces ressources pouvaient être difficiles d'accès et d'utilisation, notamment à cause des différents inventaires de sens utilisés (différentes versions de WordNet, ou même d'autres bases lexicales), ou bien des différents formats de fichiers. Cependant, les travaux de Vial *et al.* (2018) apportent une ressource nommée UFSAC contenant l'ensemble des corpus annotés en sens cités précédemment, dans un format et un inventaire de sens unifié, ce qui facilite leur utilisation.

Dans cet article, nous proposons pour la première fois d'entraîner un système supervisé à partir de ces données. Nous présentons aussi une synthèse des différences notables entre ces corpus dans le tableau 1. Ces chiffres font ainsi ressortir les caractéristiques propres de chaque corpus.

L'OMSTI est le plus grand corpus concernant le nombre de mots annotés et non annotés, mais il est celui qui a le plus faible ratio entre ces deux nombres : moins d'un mot sur trente est effectivement annoté en sens, là où pour le SemCor, près d'un mot sur quatre est annoté. L'OntoNotes, le DSO et le MASC ont aussi cette caractéristique d'avoir un faible ratio de mots annotés, alors que le WordNet Gloss Tagged a, comme pour le SemCor, un ratio de plus d'un mot sur cinq annotés.

Pour ce qui est de la diversité des lemmes, le SemCor et le WNGT se démarquent là encore des quatre autres corpus. En effet, plus de 10 000 lemmes différents sont annotés dans le SemCor et près de 20 000 dans le WNGT, alors qu'à l'extrême opposé,

Corpus	SemCor	DSO	WNGT	MASC	OMSTI	OntoNotes	Tous
Nombre de mots, en milliers	779 (#5) (1,77 %)	2 705 (#2) (6,15 %)	1 635 (#4) (3,72 %)	585 (#6) (1,33 %)	35 800 (#1) (81,40 %)	2 476 (#3) (5,63 %)	43 980
Nombre de mots annotés en sens, en milliers	189 (#3) (10,94 %)	176 (#4) (10,17 %)	329 (#2) (19,02 %)	45 (#6) (2,58 %)	917 (#1) (52,93 %)	75 (#5) (4,35 %)	1 732
Ratio entre le nombre de mots annotés et le nombre de mots total	0,24 (#1)	0,07 (#4)	0,20 (#2)	0,08 (#3)	0,03 (#6)	0,03 (#5)	0,04
Nombre de lemmes distincts annotés en sens	11 646 (#2) (57,27 %)	191 (#6) (0,94 %)	19 150 (#1) (94,18 %)	3 767 (#3) (18,53 %)	1 140 (#5) (5,61 %)	2 124 (#4) (10,45 %)	20 334
Nombre d'exemples par lemme	16 (#5)	922 (#1)	17 (#4)	11 (#6)	804 (#2)	35 (#3)	85
Nombre d'exemples par sens	6,04 (#6)	175,46 (#2)	6,06 (#5)	9,24 (#4)	262,86 (#1)	20,10 (#3)	19,38
Nombre de sens différents par lemme *dans WordNet*	3,66 (#5)	10,62 (#1)	3,18 (#6)	4,41 (#4)	6,93 (#2)	5,35 (#3)	3,14
Nombre de sens différents par lemme *dans le corpus*	1,94 (#4)	7,35 (#1)	2,27 (#3)	1,25 (#6)	3,26 (#2)	1,69 (#5)	2,40
Ratio entre le nombre de sens différents par lemme dans le corpus et le nombre de sens différents par lemme dans WordNet	0,56 (#3)	0,74 (#1)	0,72 (#2)	0,36 (#6)	0,55 (#4)	0,42 (#5)	0,75

Tableau 1. *Statistiques des corpus d'entraînement de la ressource UFSAC. Les chiffres entre parenthèses préfixés par un # sont les rangs, ceux suivis par un % sont la part de ce corpus par rapport au corpus « Tous ». Le corpus « Tous » correspond à un corpus constitué de la concaténation de tous les autres.*

le DSO ne contient que 191 lemmes différents. À noter que le nombre de lemmes polysémiques dans WordNet 3.0 s'élève à 26 896[3].

Dans les cinquième, sixième et septième lignes du tableau, on fait émerger une des caractéristiques principales de ces corpus : le DSO et l'OMSTI sont en effet construits autour d'un nombre restreint de lemmes très fortement polysémiques, mais avec un très grand nombre d'exemples pour chaque lemme et chaque sens de ces lemmes. À l'opposé, le MASC, l'OntoNotes et surtout le SemCor et le WNGT sont construits plutôt autour d'un certain type de documents (actualités, définitions d'un dictionnaire...) et donc les lemmes sont plus variés, avec un nombre de sens plus variable et un nombre d'exemples par lemme plus réduit.

Enfin les deux dernières lignes font ressortir une dernière caractéristique de ces corpus : à quel point tous les sens de chaque lemme sont représentés au sein du corpus, par rapport à ceux présents dans WordNet. En effet, un ratio inférieur à 1 dans la dernière ligne indique que pour chacun des lemmes présents dans le corpus, tous ses différents sens existants dans le dictionnaire ne sont pas représentés dans le corpus. On peut même dire que pour le MASC et l'OntoNotes, par exemple, en moyenne moins de la moitié des sens possibles d'un lemme sont représentés dans le corpus, ce qui pourrait poser problème à un système supervisé pour généraliser à de nouvelles données ce qu'il a appris sur ces corpus. Le DSO et le WNGT ont eux en moyenne près de trois quarts des sens représentés par lemmes présents dans leurs données.

Outre le fait de mieux comprendre quelles sont les différences marquées entre les corpus d'entraînement de la ressource UFSAC, ces chiffres nous permettent de voir l'intérêt de tous les combiner pour l'apprentissage d'un système supervisé de désambiguïsation lexicale. En effet, la colonne « Tous » montre les mêmes statistiques vues précédemment, appliquées à cet ensemble constitué de tous les corpus et on voit bien, par exemple sur la quatrième ligne ou sur la dernière ligne, que cette combinaison constitue un ensemble riche de données annotées, avec plus de 20 000 lemmes uniques présents, et le plus grand ratio de polysémie moyenne dans le corpus des lemmes représentés, sur la polysémie moyenne de ces lemmes dans WordNet. Cependant, cette combinaison donne aussi un des ratios de nombre de mots annotés sur le nombre de mots total parmi les plus bas. Les phrases quasi entièrement annotées du SemCor et du WNGT se retrouvent donc un peu noyées dans les phrases faiblement annotées de l'OMSTI ou de l'OntoNotes.

Afin de voir l'impact du choix du corpus d'apprentissage sur les performances du système que nous proposons, nous allons utiliser comme données d'entraînement cet ensemble de tous les corpus UFSAC d'une part, mais nous allons aussi nous restreindre aux données du SemCor uniquement, afin de nous comparer aux systèmes de l'état de l'art entraînés uniquement sur ce corpus.

3. `https://wordnet.princeton.edu/documentation/wnstats7wn`

3. Nouvelle architecture neuronale pour la désambiguïsation lexicale

Notre approche est, comme pour Raganato *et al.* (2017), de considérer la désambiguïsation lexicale comme un problème de classification dans lequel un label est assigné à chaque mot. Nous simplifions cependant leur modèle en considérant qu'un label appartient uniquement à l'ensemble de tous les sens de notre inventaire de sens. L'architecture de notre réseau de neurones repose ainsi sur trois couches de cellules.

La première couche, ou couche d'entrée, permet d'associer à un mot une forme vectorielle à l'aide d'une table de correspondance construite séparément de notre système. On pourra utiliser ici n'importe quelle base de vecteurs de mots pré-entraînés telle que Word2Vec (Mikolov *et al.*, 2013) ou GloVe (Pennington *et al.*, 2014). Si un mot est absent de la base de vecteurs utilisée, alors on lui associe le vecteur nul.

Ensuite, la couche cachée, composée de cellules LSTM (Hochreiter et Schmidhuber, 1997) bidirectionnelles, va calculer une nouvelle représentation vectorielle pour chacun des mots en fonction des mots précédents et suivants. En effet, ces cellules dites « à mémoire » aussi appelées cellules « récurrentes » permettent de calculer une sortie en considérant non seulement l'élément courant de la séquence, mais aussi l'historique passé des cellules précédentes. Nous utilisons l'implémentation d'une cellule LSTM incluse dans l'outil PyTorch[4] formulée ainsi :

$$
\begin{array}{ll}
i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{(t-1)} + b_{hi}) & f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{(t-1)} + b_{hf}) \\
g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{(t-1)} + b_{hg}) & o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{(t-1)} + b_{ho}) \\
c_t = f_t c_{(t-1)} + i_t g_t & h_t = o_t \tanh(c_t)
\end{array}
$$

La sortie h_t d'une cellule LSTM à l'instant t est donc fonction de son entrée x_t, de ses états précédents $h_{(t-1)}$ et $c_{(t-1)}$, des matrices de poids W_i et W_h et des vecteurs de biais b_i et b_h. Ces poids et ces biais étant les paramètres qui vont être appris par notre modèle pendant l'entraînement. Enfin, σ est la fonction sigmoïde.

Pour obtenir une sortie de cellule LSTM bidirectionnelle, nous appliquons ces formules pour chacune des deux directions (t allant de 0 à n pour une direction, et de n à 0 pour l'autre direction), et nous concaténons leur sortie.

Enfin, la dernière couche de notre architecture, la couche de sortie, applique une transformation linéaire et une fonction softmax classique à la sortie du LSTM, afin de générer pour chacun des mots en entrée, une distribution de probabilité sur tous les sens possibles du dictionnaire. Nous suivons ainsi la formule $y = softmax(Ax + b)$, avec x la sortie du LSTM, A la matrice de poids et b le vecteur de biais, A et b étant des paramètres de notre modèle.

La fonction de coût à minimiser pendant la phase d'apprentissage est l'entropie croisée entre la couche de sortie et un vecteur de type *one-hot*, pour lequel toutes les composantes sont à 0 sauf à l'index du sens cible où elle est à 1. On cherche ainsi à minimiser la fonction $H(p, q) = -\sum_x p(x) \log q(x)$, où x est une composante du vecteur de la couche de sortie, p est la distribution de probabilité réelle et q la sortie

4. `https://pytorch.org`

de notre réseau de neurones. Comme toutes les valeurs de la distribution réelle sont à 0 sauf à l'index du sens correct, pour un exemple donné, on cherche ainsi à minimiser la formule $-\log q(s)$, où s est l'index du sens à prédire.

Notre modèle prédit toujours un sens en sortie pour chaque mot en entrée, même pour les mots-outils ou les mots qui n'ont pas été annotés dans le corpus d'entraînement. Cependant, dans ces cas-là, nous annotons le mot avec un symbole spécial `<skip>` qui sert de marqueur pendant l'entraînement en indiquant d'ignorer les prédictions faites par le modèle et de ne pas en tenir compte lors de la rétropropagation. Plus précisément, la fonction de coût n'est pas calculée quand ce marqueur est présent et donc le gradient n'est pas mis à jour.

Contrairement à l'approche proposée par Raganato *et al.* (2017), nous entraînons notre modèle non seulement sur des données entièrement annotées, comme c'est le cas avec le SemCor par exemple, mais également sur des données partiellement annotées, comme l'OMSTI, dans lequel un seul mot est annoté par phrase. Il est en effet capable d'apprendre à prédire les sens de tous les mots annotés dans une séquence en même temps, tout en ignorant les éléments non annotés.

De plus, notre modèle a l'avantage par rapport à celui de Raganato *et al.* (2017) d'avoir un nombre de paramètres réduit à la taille du vocabulaire de sortie. En effet, notre modèle prédit un label appartenant à l'ensemble des sens vus pendant l'entraînement, ce qui correspond à un ensemble de taille 26 215 en entraînant sur le SemCor, et 70 131 en utilisant tous les corpus UFSAC. L'approche de Raganato *et al.* (2017), quant à elle, inclut dans le vocabulaire de sortie du modèle, en plus de tous les sens, tous les *mots* vus pendant l'entraînement et qui sont présents dans le modèle de vecteurs de mots utilisé en entrée. Les auteurs ne fournissent pas de chiffre ni leur code, mais nous estimons en utilisant le même modèle de vecteurs de mots qu'eux, et en filtrant sur les mots présents dans le SemCor (leur corpus d'entraînement), que ce sont 34 980 labels qui sont ajoutés à leur vocabulaire de sortie. En appliquant cette méthode sur tous les corpus d'entraînement de UFSAC, 144 308 labels supplémentaires seraient ajoutés.

Notre architecture est enfin très différente de celles de Yuan *et al.* (2016) ou de Kågebäck et Salomonsson (2016), notamment car elles ne permettent pas d'annoter tous les mots en entrée de leurs modèles en une seule passe.

4. Protocole expérimental

Dans cette section, nous détaillons le processus que nous avons suivi concernant l'apprentissage et l'évaluation de notre système de DL à base de réseau neuronal.

4.1. *Corpus d'entraînement, de développement et d'évaluation*

Nous avons tiré parti du travail de Vial *et al.* (2018) et décidé d'utiliser pour la première fois les six corpus d'entraînement de la ressource UFSAC décrits dans la sous-section 2.4, à savoir : le SemCor, le DSO, le WordNet Gloss Tagged, l'OMSTI,

le MASC et l'OntoNotes. De plus, afin de comparer l'architecture que nous proposons avec l'état de l'art, et notamment Raganato *et al.* (2017) et Yuan *et al.* (2016), qui utilisent uniquement le SemCor comme corpus d'apprentissage supervisé, nous avons aussi évalué notre approche en limitant l'apprentissage du modèle à ce corpus.

Nous avons utilisé le corpus de la tâche 13 de SemEval 2015 (Moro et Navigli, 2015) comme corpus de développement durant l'apprentissage. Le modèle ayant obtenu le meilleur score F1 (sur le développement) a été utilisé sur les corpus de SensEval 2 (Edmonds et Cotton, 2001), SensEval 3 (Snyder et Palmer, 2004), les tâches 7 et 17 de SemEval 2007 (Navigli *et al.*, 2007 ; Pradhan *et al.*, 2007), et enfin la tâche 12 de SemEval 2013 (Navigli *et al.*, 2013).

Dans certains corpus, les mots peuvent être annotés avec plusieurs sens WordNet, soit parce que l'annotateur a trouvé qu'ils étaient tous applicables, ou bien parce qu'ils ont été initialement annotés avec un autre dictionnaire puis les sens ont été convertis en sens WordNet. Dans ce cas nous supprimons entièrement l'annotation du mot pour ne garder au final que ceux qui ont été annotés avec un seul sens dans notre corpus d'apprentissage. Pour le SemCor, le DSO, le WordNet Gloss Tagged et l'OMSTI, ce cas ne concerne respectivement que 0,38 %, 0,02 %, 0 % et 0,02 % des annotations. Cependant pour le MASC et l'OntoNotes, qui ont été initialement annotés avec un dictionnaire distinguant moins finement les sens que WordNet, ce sont respectivement 55,98 % et 65,93 % des annotations qui sont enlevées.

Afin de mesurer l'impact du mélange des corpus d'entraînement ayant des proportions de mots annotés très différentes, nous avons mené l'expérience qui consiste à répartir les annotations du SemCor en plusieurs phrases, et de mesurer les répercutions sur notre système. Nous avons créé deux nouvelles versions du SemCor. Une version « éclatée » à 100 % dans laquelle pour chaque phrase contenant n annotations, nous dupliquons cette phrase n fois et annotons un seul mot différent dans chacune d'elles. Puis une version « éclatée » à 50 % dans laquelle nous appliquons cette opération sur seulement la moitié des phrases du SemCor, puis mélangeons toutes les phrases. Nous avons ainsi entraîné huit modèles sur chacun de ces corpus, et les résultats de nos systèmes sur le corpus de développement se trouvent dans le tableau suivant :

Données d'entraînement	Nombre de phrases	Score F1 (%)	Écart-type
SemCor original	36 321	72,24	± 0,64
SemCor « éclaté » à 50 %	132 945	71,50	± 0,62
SemCor « éclaté » à 100 %	227 993	71,72	± 0,61

Bien qu'une tendance se dégage en faveur de la version originale du SemCor, même en se plaçant dans le cas extrême d'un seul mot annoté par phrase, l'impact sur le score final est négligeable. Notre architecture est donc capable d'apprendre sur des phrases partiellement annotées, sans dégradation notable.

4.2. *Hyperparamètres du modèle neuronal*

En entrée de notre réseau, nous avons utilisé les vecteurs de GloVe (Pennington *et al.*, 2014) pré-entraînés sur Wikipedia 2014 et Gigaword 5 disponibles librement[5]. La taille des vecteurs est de 300, la taille du vocabulaire est de 400 000 et tous les mots sont en minuscules. Nous avons choisi ces vecteurs pour la petite taille de leur modèle pré-entraîné et pour leur qualité démontrée par les auteurs sur les tâches de similarité de mots et d'analogie de mots. Ce sont aussi ces vecteurs qui sont utilisés en entrée du réseau décrit par Kågebäck et Salomonsson (2016).

Pour la couche cachée de neurones récurrents, nous avons choisi des cellules LSTM de taille 1 000 par direction (donc 2 000 au total). C'est, à peu de choses près, la taille qui est utilisée dans les travaux de Yuan *et al.* (2016) (une couche de taille 2 048), et c'est environ deux fois moins que ce qui est utilisé par Raganato *et al.* (2017) (deux couches de cellules LSTM bidirectionnelles de taille 1 024).

Enfin, entre la couche cachée et la couche de sortie, nous avons appliqué une régularisation *Dropout* (Srivastava *et al.*, 2014) à $50\,\%$. Cette méthode vise à empêcher le surapprentissage pendant l'entraînement afin de rendre le modèle plus robuste.

Système	Vecteurs en entrée	Couche LSTM	Taux de *Dropout*	Score F1 (%)	Écart-type
(Raganato *et al.*, 2017)	Word2Vec	2*1 024	0	69,2	-
Notre système (SemCor, moyenne de huit modèles entraînés séparément)	Word2Vec	2*1 024	0	68,76	± 0,77
	Word2Vec	2*1 024	0,5	68,47	± 0,57
	Word2Vec	1*1 000	0	69,06	± 0,71
	Word2Vec	1*1 000	0,5	69,25	± 0,48
	GloVe	2*1 024	0	71,21	± 0,49
	GloVe	2*1 024	0,5	72,08	± 0,60
	GloVe	1*1 000	0	71,37	± 0,35
	GloVe	1*1 000	0,5	72,24	± 0,64

Tableau 2. *Résultats obtenus par notre système entraîné sur le SemCor et celui de Raganato et al. (2017), sur notre corpus de développement (SemEval 2015 tâche 13) en fonction des hyperparamètres du modèle neuronal.*

Dans le tableau 2, nous montrons l'impact de ces hyperparamètres sur l'apprentissage de notre modèle en faisant varier chacun d'eux entre les valeurs que nous avons choisies et celles utilisées par Raganato *et al.* (2017).

Comme nous pouvons le voir, à paramètres équivalents, c'est-à-dire en utilisant le même modèle de vecteurs de mots, avec deux couches de LSTM et sans *Dropout*, notre système obtient des résultats qui ne sont pas significativement différents de ceux de Raganato *et al.* (2017). De même, réduire de moitié le nombre de couches de cellules LSTM ne fait pas varier significativement les résultats en diminuant pourtant largement le nombre de paramètres du modèle. La différence de résultats est cependant

5. `https://nlp.stanford.edu/projects/glove/`

notable lorsque l'on utilise les vecteurs GloVe plutôt que Word2Vec, avec systématiquement entre 2 % et 4 % d'amélioration. Enfin, l'utilisation du *Dropout* améliore les résultats de presque 1 % lorsque l'on utilise GloVe.

4.3. *Entraînement du modèle neuronal*

Les paramètres utilisés pour l'apprentissage sont les suivants : la méthode d'optimisation est Adam (Kingma et Ba, 2015), avec les mêmes paramètres par défaut tels que décrits dans leur article, la taille de mini-lots utilisée est de 30, les phrases sont tronquées à 50 mots, pour faciliter l'entraînement tout en minimisant la perte d'informations (moins de 5 % des mots annotés dans nos données d'entraînement sont perdus), enfin, les séquences sont remplies de vecteurs nuls depuis la fin de façon à ce qu'elles aient toutes la même taille au sein d'un mini-lot.

Nous avons construit notre réseau neuronal à l'aide de l'outil PyTorch[6] et nous avons effectué l'apprentissage pendant 20 cycles. Un cycle correspondant à une passe complète sur nos données d'entraînement. Nous avons évalué périodiquement (tous les 2 000 mini-lots et à la fin de chaque cycle) notre modèle sur le corpus de développement, et conservé uniquement celui ayant obtenu le plus grand score F1. Le code source de notre système ainsi que nos modèles pré-entraînés sont disponibles à l'adresse suivante : `https://github.com/getalp/disambiguate`.

4.4. *Processus de désambiguïsation*

Pour réaliser la désambiguïsation d'une séquence de mots en utilisant le réseau entraîné, chaque mot est d'abord transformé en vecteur à l'aide du modèle de vecteurs de mots, puis donné en entrée au réseau. En sortie, une distribution de probabilité sur tous les sens observés pendant l'apprentissage est retournée pour chaque élément de la séquence. Nous assignons le sens le plus probable en suivant cette distribution, parmi les sens possibles du mot dans WordNet, en fonction de son lemme et de sa partie du discours. Ces deux informations étant systématiquement données pendant les campagnes d'évaluation de la DL. Si aucun sens n'est assigné, une stratégie de repli est effectuée. La plus courante, et celle que nous utilisons, est d'assigner au mot son sens le plus fréquent dans WordNet.

Le processus d'apprentissage est forcément stochastique. En effet non seulement les poids du modèle sont initialisés aléatoirement par la bibliothèque sous-jacente, mais le corpus d'apprentissage est également mélangé à chaque début de cycle. Nous avons entraîné ainsi huit modèles séparément pour chacune de nos configurations, puis pour chacune de nos évaluations, nous donnons deux résultats :

– la moyenne des scores obtenus par chacun des huit modèles, ainsi que leur écart-type. Nous pouvons ainsi nous comparer avec les autres travaux de l'état de l'art qui n'entraînent qu'un seul modèle tout en ayant un moyen d'estimer la variabilité et la significativité des résultats ;

6. `http://pytorch.org/`

– le score d'un système « ensemble », c'est-à-dire un système qui moyenne les prédictions faites par ces huit modèles afin d'obtenir une nouvelle distribution de sens plus stable et généralement de meilleure qualité.

Pour le système « ensemble », nous avons utilisé une moyenne géométrique sur les prédictions faites par les modèles. Cette pratique couramment utilisée (par exemple Sutskever *et al.* (2014)) permet non seulement d'avoir un système moins sensible au bruit et donc plus robuste, mais aussi un système de meilleure qualité. En effet, un modèle peut être individuellement bloqué dans un minimum local pendant l'entraînement et avoir un bon score sur le corpus de développement, mais être incapable de généraliser, alors qu'il est improbable que ce problème arrive à l'ensemble de modèles.

5. Résultats

Nous avons évalué notre modèle sur tous les corpus d'évaluation des tâches de DL des campagnes d'évaluation SensEval-SemEval. Les scores obtenus par notre système comparés à ceux des systèmes semblables de l'état de l'art à base de réseaux de neurones (Yuan *et al.*, 2016 ; Raganato *et al.*, 2017), ainsi que l'étalon du sens le plus fréquent se trouvent dans le tableau 3.

On remarque d'abord qu'en termes de scores F1 avec le repli, il y a très peu de différences entre les modèles entraînés sur le SemCor et ceux entraînés sur tous les corpus UFSAC. Nos meilleurs résultats sur les tâches de SensEval 3 et de SemEval 2007 sont même obtenus par le système qui est entraîné sur le SemCor uniquement. Le SemCor possède pourtant seulement environ 10 % des mots annotés dans UFSAC.

Cependant, lorsque l'on compare les scores de désambiguïsation d'un système avec et sans repli, la différence entre ces deux scores est bien plus grande avec le système entraîné sur le SemCor qu'avec celui entraîné sur UFSAC. Ceci s'explique par la couverture du SemCor qui est moins importante que celle des corpus UFSAC réunis. Pour le système appris sur le SemCor, la couverture est en effet de 91 % sur SensEval 2, 97 % sur SensEval 3, 93 % sur SemEval 2007 (07), 98 % sur SemEval 2007 (17), 91 % sur SemEval 2013 et 95 % sur SemEval 2015. Pour celui appris sur tout UFSAC, la couverture est respectivement de 98 %, 99 %, 99 %, 99 %, 98 % et 99 %.

Ces résultats démontrent la grande qualité du SemCor, c'est en effet lorsque sa couverture sur les tâches d'évaluation est la plus proche de 100 % que notre système appris sur ce seul corpus obtient les meilleurs résultats. Les autres corpus UFSAC permettent quand même d'annoter un bien plus grand nombre de sens sans stratégie de repli, et nos meilleurs résultats sur SensEval 2, SemEval 2013 et SemEval 2015 sont obtenus avec le système appris sur tous les corpus réunis.

Le premier système de Yuan *et al.* (2016) obtient des résultats comparables aux nôtres mais comme nous l'avons souligné dans la section 2, le caractère privé de leur corpus d'entraînement contenant 100 milliards de mots pour leur modèle de langue rend très difficile la reproductibilité de leurs résultats.

Système	SE2	SE3	SE07 (07)	SE07 (17)	SE13 (12)	SE15 (13)
Sens le plus fréquent	65,6	66,0	78,89	54,5	63,8	67,1
Yuan *et al.* (2016) (LSTM)	73,6	69,2	82,8	**64,2**	67,0	72,1
Yuan *et al.* (2016) (LSTM + LP)	**73,8**	**71,8**	**83,6**	63,5	**69,5**	**72,6**
Raganato *et al.* (2017) (BLSTM)	71,4	68,8	-	*61,8	65,6	69,2
Raganato *et al.* (2017) (BLSTM + att. + LEX + POS)	72,0	69,1	83,1	*64,8	66,9	71,5
Notre système (SemCor, individuel, sans repli)	⋆67,72 (± 0,31)	⋆68,99 (± 0,40)	⋆79,62 (± 0,33)	⋆60,07 (± 0,87)	⋆62,94 (± 0,35)	⋆*68,25 (± 0,67)
Notre système (UFSAC, individuel, sans repli)	⋆71,08 (± 0,70)	⋆67,27 (± 0,65)	⋆82,79 (± 0,46)	⋆58,68 (± 1,38)	⋆66,67 (± 0,83)	⋆*72,57 (± 0,43)
Notre système (SemCor, ensemble, sans repli)	68,36	69,78	80,14	60,51	63,10	*68,63
Notre système (UFSAC, ensemble, sans repli)	72,54	69,57	83,05	59,63	67,47	*73,30
Notre système (SemCor, individuel, avec repli)	⋆73,18 (± 0,30)	⋆70,74 (± 0,40)	83,49 (± 0,32)	⋆61,54 (± 0,86)	67,55 (± 0,33)	⋆*72,24 (± 0,64)
Notre système (UFSAC, individuel, avec repli)	⋆72,30 (± 0,69)	⋆67,99 (± 0,65)	83,51 (± 0,46)	⋆58,84 (± 1,38)	68,07 (± 0,82)	⋆*73,34 (± 0,43)
Notre système (SemCor, ensemble avec repli)	73,71	**71,51**	**83,99**	**61,98**	67,70	*72,60
Notre système (UFSAC, ensemble avec repli)	**73,75**	70,27	83,77	59,78	**68,86**	***74,07**

Tableau 3. *Scores F1 (%) obtenus par notre système sur les tâches de DL des campagnes d'évaluation SensEval-SemEval. Les résultats préfixés par un astérisque (*) sont obtenus sur le corpus utilisé pour le développement pendant l'apprentissage. Les résultats préfixés par une étoile (⋆) sont significativement différents (entre le système entraîné sur le SemCor et le système entraîné sur tout UFSAC) selon un test de Student (p-valeur $< 0,01$). Les résultats en gras sont les meilleurs obtenus par notre système et par les systèmes de l'état de l'art. Ceux en rouge sont les meilleurs de l'état de l'art.*

Leur deuxième système (LSTM + LP) ajoute une étape de propagation de labels, dans laquelle ils augmentent automatiquement leurs données d'entraînement annotées en sens, en recherchant dans une grande quantité de textes non annotés des phrases similaires aux phrases annotées, et en portant les labels de sens depuis les phrases annotées, vers les phrases non annotées. Cette méthode apporte de meilleurs résultats sur la plupart des tâches, cependant ils récupèrent, pour leurs données non annotées, 1 000 phrases prises aléatoirement sur le Web pour chaque lemme, sans plus de précisions, ce qui rend la reproductibilité des résultats encore plus difficile.

Le système de Raganato *et al.* (2017) qui est, quant à lui, très semblable au nôtre obtient des résultats moins élevés alors qu'ils utilisent deux couches de cellules LSTM bidirectionnelles de taille 2 048 (1 024 par direction), donc un total de 4 096 unités cachées, ce qui est deux fois plus que notre modèle.

Pour leur second système (BLTM + att. + LEX + POS), les auteurs ont ajouté une couche d'attention à leur réseau, et ils effectuent de l'apprentissage multitâche, c'est-à-dire que leur réseau apprend à la fois à prédire un label de mot ou de sens, ainsi que la partie du discours (POS) du mot, et son label sémantique dans WordNet (LEX).

En comparaison avec ces autres systèmes, les nôtres obtiennent des scores supérieurs à ceux de Raganato *et al.* (2017) dans la majorité des cas, malgré une complexité réduite au niveau de l'architecture. Nous obtenons des scores similaires ou légèrement inférieurs à ceux de Yuan *et al.* (2016), mais en utilisant largement moins de données pour l'apprentissage, et surtout des données librement accessibles.

5.1. *Analyse des erreurs*

Afin de mieux comprendre l'impact du choix du corpus d'entraînement sur la qualité des annotations produites par notre système, nous proposons dans cette section une analyse plus fine des sorties, en nous appuyant sur les statistiques du tableau 4. Ces chiffres sont calculés avec les sorties de nos systèmes sur l'ensemble des corpus d'évaluation (sans le corpus de développement) et sans la stratégie de repli.

Tout d'abord la première chose que l'on peut remarquer, dans les deux premières sections du tableau, est que sur les 8 492 mots à annoter, le système appris sur l'ensemble des corpus UFSAC permet d'annoter 441 mots de plus que le système appris sur le SemCor, soit une couverture supplémentaire de plus de 5 % des mots à annoter. Sur les autres métriques, la précision du système appris sur plus de données est légèrement supérieure (moins de 1 %), et d'une manière générale le rappel et la F-mesure sont largement meilleurs (plus de 2 % de F1 du fait de la plus grande couverture).

La troisième section du tableau, « Moyenne du nombre de sens dans WordNet », montre une tendance visible sur les deux systèmes : les mots mal annotés sont généralement aussi les plus polysémiques. Cette statistique permet de rappeler que même pour les systèmes de désambiguïsation les plus performants, la distinction extrêmement « fine » entre les sens de WordNet reste un des défis principaux de cette tâche. Pour rappel, le verbe « *make* » a par exemple 49 sens différents dans WordNet.

	Notre système (SemCor)	Notre système (UFSAC)
Nombre de mots		
total des mots à annoter	8 492	8 492
mots non annotés	582	141
mots bien annotés	5 770	6 145
mots mal annotés	2 140	2 206
Métriques		
Couverture (C)	93,15 %	98,34 %
Précision (P)	72,95 %	73,58 %
Rappel (R)	67,95 %	72,36 %
F-mesure (F1)	70,36 %	72,97 %
Moyenne du nombre de sens dans WordNet		
pour les mots bien annotés	5,47	5,04
pour les mots mal annotés	8,58	8,86
Nombre moyen d'exemples du lemme cible dans le corpus d'entraînement		
pour les mots bien annotés	347	1 146
pour les mots mal annotés	215	1 489
Nombre moyen d'exemples du sens attendu dans le corpus d'entraînement		
pour les mots bien annotés	92	399
pour les mots mal annotés	21	206
Équilibre de la représentation des différents sens du lemme cible dans le corpus d'entraînement		
pour les mots bien annotés	1,64	2,00
pour les mots mal annotés	0,70	1,05
Nombre de mots mal annotés dont le sens attendu n'est jamais représenté dans le corpus d'entraînement	546	81

Tableau 4. *Analyse des erreurs commises sur l'ensemble des corpus d'évaluation (hormis le corpus de développement) par notre système entraîné sur le SemCor d'une part, et sur tous les corpus UFSAC d'autre part.*

Dans la quatrième section, « Nombre moyen d'exemples du lemme cible », on peut voir un indice supplémentaire pour montrer la qualité du SemCor par rapport aux autres corpus annotés en sens. En effet, la colonne de gauche montre que le système appris sur ce corpus annote généralement mieux les lemmes dont il a vu le plus d'exemples pendant l'entraînement, ce qui peut sembler évident pour un système supervisé. En revanche, le deuxième système affiche des chiffres opposés : les mots mal annotés ont été observés en moyenne un plus grand nombre de fois que les mots bien annotés. On peut ainsi émettre l'hypothèse que parfois, un trop grand nombre d'exemples dans le corpus d'entraînement va nuire à l'apprentissage des systèmes supervisés en ajoutant plus de bruit que d'informations pertinentes.

Le nombre moyen d'exemples du sens attendu dans le corpus d'entraînement montre que les sens les plus vus dans les données d'apprentissage sont généralement

plus souvent choisis par le système. Autrement dit si un sens est souvent représenté, que ce soit dans un contexte pertinent ou bien dans un contexte bruité, le système aura plus souvent tendance à le sélectionner.

L'avant-dernière section « Équilibre de la représentation des différents sens » met en avant une statistique particulière : pour les mots bien et mal annotés, est-ce que les différents sens du lemme cible étaient représentés de manière équilibrée dans le corpus d'entraînement ? Ces chiffres s'appuient sur la formule suivante :

$$\frac{nombre\ d'exemples\ du\ sens\ cible}{nombre\ d'exemples\ du\ lemme\ cible} * nombre\ de\ sens\ du\ lemme\ cible$$

Plus le chiffre est proche de 1, et plus la représentation du sens est « équilibrée », c'est-à-dire qu'il n'est pas sur-représenté ni sous-représenté par rapport aux autres sens possibles du lemme. Sur nos deux systèmes, les chiffres montrent qu'en général, les mots bien annotés ont en moyenne un sens qui est quasiment deux fois plus représenté que les autres sens, et les sens mal assignés sont largement moins représentés dans le cas du SemCor, et assez équilibrés dans le cas de l'ensemble de corpus.

Enfin, les derniers chiffres du tableau mettent en avant le fait que, dans de nombreux cas, sur ces corpus d'évaluation, même si le système est capable d'annoter un mot en sens, il ne pourra jamais sélectionner le sens attendu, tout simplement parce qu'il ne l'a jamais observé pendant l'apprentissage. C'est le cas pour 546 mots, pour le système appris sur le SemCor, ce qui correspond à plus de 6 % du total des mots à annoter, qu'aucun système supervisé appris sur ces données ne serait capable d'annoter correctement. Pour le deuxième système ce chiffre tombe à moins de 1 % du total des mots à annoter, cela montre une fois de plus l'intérêt d'utiliser l'ensemble des données disponibles pour l'entraînement de systèmes supervisés plus robustes.

Pour conclure, ces statistiques permettent de mieux se rendre compte de l'impact des corpus choisis pour l'entraînement sur les performances de notre système, mais aussi de l'importance de la qualité et de la quantité des données annotées en sens pour l'entraînement de systèmes supervisés en général.

En effet, si la majorité des systèmes supervisés sont entraînés sur le SemCor, très peu sont les travaux qui justifient son utilisation plutôt qu'un autre corpus. Avec l'ensemble des corpus annotés en sens WordNet maintenant facilement accessibles et dans un format unifié, il est désormais plus facile d'identifier et de sélectionner des corpus ou même seulement des parties de corpus qui seront bénéfiques aux systèmes de désambiguïsation supervisés.

6. Vers une amélioration non supervisée

Dans cette section, nous présentons une approche visant l'amélioration non supervisée de notre système, en s'appuyant sur des corpus non annotés. Nous mettons ainsi en avant des pistes qui pourraient être approfondies dans de futurs travaux.

6.1. *Approche*

L'approche que nous avons suivie est en partie inspirée de la méthode de propagation de labels de Yuan *et al.* (2016), dans laquelle les auteurs transfèrent des annotations de sens de leur corpus manuellement annoté vers des phrases non annotées, pour étendre leurs données d'apprentissage.

Notre approche est aussi inspirée des méthodes d'apprentissage par mimétisme telles que Buciluǎ *et al.* (2006), dans lesquelles un ou plusieurs modèles « enseignant » vont transférer leurs connaissances à un modèle « élève » en lui montrant comment effectuer une tâche. L'élève va ainsi apprendre à recopier ce que font les enseignants, observer des exemples dans de nouveaux contextes et ainsi apprendre à mieux généraliser.

Dans le contexte de la DL, les modèles enseignants sont des modèles capables d'annoter n'importe quelle séquence de mots en sens, et le modèle élève sera un nouveau modèle qui va être entraîné sur des données produites par les enseignants.

Intuitivement, on peut comprendre comment le modèle élève sera capable de généraliser les connaissances de l'enseignant grâce à l'exemple suivant : imaginons que les données d'entraînement utilisées par le système enseignant contiennent la phrase « la souris mange le fromage » annotée en sens, et que dans les données d'évaluation, on retrouve la phrase « la souris *dévore* le *gruyère* ». Les distances successives entre les mots « mange » et « *dévore* », puis entre « fromage » et « *gruyère* » dans l'espace vectoriel des mots sont peut être suffisamment élevées pour que le modèle ayant appris à annoter la première phrase ne soit pas capable d'annoter correctement la seconde.

Si par contre le modèle enseignant annote un ensemble de données dont la phrase « la souris mange le *gruyère* », ici seul le mot « *gruyère* » est différent par rapport à la phrase contenue dans les données d'entraînement et donc le mot « souris » a plus de chances d'être annoté correctement. Le modèle élève va ensuite pouvoir considérer cette dernière phrase annotée pendant son apprentissage, et par transfert, comme cette phrase est très proche de celle sur laquelle l'enseignant se trompait auparavant, il va cette fois être mieux capable de l'annoter correctement.

6.2. *Protocole expérimental*

Nous avons utilisé comme modèle enseignant le système de DL qui a obtenu le meilleur score F1 (voir le tableau 3) sur notre corpus de développement uniquement (SemEval 2015) afin d'éviter tout biais, c'est-à-dire celui entraîné sur toutes les données UFSAC avec la stratégie de repli.

Nous avons annoté avec ce système un million de phrases prises sur les données anglaises monolingues des campagnes d'évaluation de la traduction automatique WMT, et plus précisément le premier million de phrases du corpus « News Crawl 2016 » accessible sur le site de la campagne d'évaluation WMT17[7].

7. `http://data.statmt.org/wmt17/translation-task/news.2016.en.shuffled.gz`

Nous avons ensuite entraîné un nouveau modèle avec la même architecture sur ces données automatiquement annotées, en suivant le même protocole décrit dans la section 4, puis nous avons poursuivi l'entraînement de ce modèle, mais cette fois-ci sur les corpus UFSAC manuellement annotés, toujours pendant vingt cycles et en conservant le modèle avec le meilleur score sur le corpus de développement.

Nous avons réitéré cette dernière étape jusqu'à obtenir huit modèles différents afin d'évaluer cette méthode, comme pour le système original, à la fois en calculant la moyenne et l'écart-type des modèles pris individuellement, et en moyennant les prédictions d'un ensemble de modèles.

Dans le tableau 5 se trouvent les statistiques du corpus annoté en sens par le système enseignant, afin de le positionner par rapport aux autres corpus UFSAC. Tout d'abord nous pouvons voir que, comparativement à l'OMSTI qui est aussi un corpus d'environ un million de phrases, le nombre de mots de ce corpus est bien inférieur. En effet, en moyenne, les phrases de l'OMSTI sont constituées de 44 mots alors que les phrases de ce corpus ont une longueur moyenne de 22 mots. En ce qui concerne les nombres d'annotations en sens, ce nouveau corpus devient le plus densément annoté.

Concernant les lemmes distincts annotés, on retrouve plus de trois quarts du vocabulaire des corpus UFSAC qui est représenté dans ce million de phrases, et un très grand nombre d'exemples par lemmes et par sens, avec des chiffres comparables à ceux de l'OMSTI et du DSO.

Enfin, par rapport à la polysémie présente dans le corpus, et le ratio entre le nombre de sens présents dans le corpus par rapport au nombre de sens présents dans WordNet, on retrouve des chiffres similaires à ceux du SemCor.

	1M News 2016
Nombre de mots, en milliers	19 616
Nombre de mots annotés en sens, en milliers	7 762
Ratio entre le nombre de mots annotés et le nombre de mots total	0,40
Nombre de lemmes distincts annotés en sens	17 469
Nombre d'exemples par lemme	444
Nombre d'exemples par sens	181
Nombre de sens différents par lemme *dans WordNet*	3,26
Nombre de sens différents par lemme *dans le corpus*	1,60
Ratio entre le nombre de sens différents par lemme dans le corpus et le nombre de sens différents par lemme dans WordNet	0,53

Tableau 5. *Statistiques du corpus constitué du premier million de phrases du corpus « News Crawl 2016 » annoté automatiquement en sens par le système enseignant.*

6.3. *Résultats*

Nous avons évalué le système « élève » avec repli sur les mêmes tâches que précédemment, et nous avons comparé ses scores avec ceux obtenus par le système « enseignant », et avec le système de Yuan *et al.* (2016). Les résultats sont dans la tableau 6.

Système	SE2	SE3	SE07 (07)	SE07 (17)	SE13 (12)	SE15 (13)
Yuan *et al.* (2016) (LSTM)	73,6	69,2	82,8	64,2	67,0	72,1
Yuan *et al.* (2016) (LSTM + LP)	73,8	71,8	83,6	63,5	69,5	72,6
Système « enseignant » (UFSAC, ensemble)	73,75	**70,27**	83,77	59,78	68,86	*74,07
Système « élève » à l'initialisation (1M News 2016, individuel)	71,25	68,38	82,93	61,10	68,80	*71,53
Système « élève » (UFSAC + 1M News 2016, individuel)	73,50 (± 0,68)	68,74 (± 0,66)	84,39 (± 0,50)	59,62 (± 0,95)	68,95 (± 0,81)	***74,43** (± 0,57)
Système « élève » (UFSAC + 1M News 2016, ensemble)	**74,58** (+ 0,83)	69,35 (-0,92)	**84,96** (+ 1,19)	**60,66** (+ 0,88)	**69,77** (+ 0,91)	*74,17 (+ 0,10)

Tableau 6. *Scores F1 (%) obtenus par le système élève sur les tâches de DL des campagnes d'évaluation SensEval-SemEval. Les résultats préfixés par un astérisque (*) sont obtenus sur le corpus utilisé pour le développement pendant l'apprentissage. La différence entre le meilleur système élève et le système enseignant est affichée entre parenthèses. Le meilleur score entre l'élève et l'enseignant est affiché en gras. Le meilleur score de l'état de l'art est affiché en rouge.*

Sur toutes les tâches d'évaluation, à part celle de SensEval 3, le système élève obtient ainsi des scores significativement supérieurs à ceux du système enseignant, et il obtient même des scores surpassant l'état de l'art sur les tâches de SensEval 2, SemEval 2007 (07), SemEval 2013 et SemEval 2015.

À travers ces résultats, on voit à quel point la mise en place de ce type d'apprentissage par transfert de connaissances peut s'avérer efficace pour la construction d'un système de DL robuste et de bonne qualité. Notre système ainsi entraîné obtient en effet des scores supérieurs à notre système original et à l'état de l'art sur la plupart des tâches d'évaluation, alors que nous avons uniquement utilisé comme ressource supplémentaire des phrases en anglais non annotées provenant d'un corpus libre d'accès.

Afin d'avoir plus de détails sur les performances de notre système « élève », nous proposons dans le tableau 7 une analyse d'erreurs semblable à celle du tableau 4. Ces statistiques sont similaires à celles du système enseignant (voir tableau 4), mais les tendances sont plus marquées. L'écart se creuse entre le nombre d'exemples du sens attendu pour les mots bien annotés et pour les mots mal annotés, ainsi que pour l'équilibre de la représentation des sens. Une tendance change cependant, c'est le nombre d'exemples du lemme cible pour les mots bien annotés et mal annotés. Dans tous ces cas, on a maintenant une bien plus forte proportion d'exemples de lemmes et de sens des mots bien annotés par rapport à ceux mal annotés.

	Système « élève »
Nombre de mots	
total des mots à annoter	8 492
mots non annotés	141
mots bien annotés	6 188
mots mal annotés	2 163
Métriques	
Couverture (C)	98,34 %
Précision (P)	74,10 %
Rappel (R)	72,87 %
F-mesure (F1)	73,48 %
Moyenne du nombre de sens dans WordNet	
pour les mots bien annotés	5,08
pour les mots mal annotés	8,82
Nombre moyen d'exemples du lemme cible dans le corpus d'entraînement	
pour les mots bien annotés	13 743
pour les mots mal annotés	11 893
Nombre moyen d'exemples du sens attendu dans le corpus d'entraînement	
pour les mots bien annotés	4 311
pour les mots mal annotés	1 255
Équilibre de la représentation des différents sens du lemme cible dans le corpus d'entraînement	
pour les mots bien annotés	2,32
pour les mots mal annotés	0,96
Nombre de mots mal annotés dont le sens attendu n'est jamais représenté dans le corpus d'entraînement	75

Tableau 7. *Analyse des erreurs commises sur l'ensemble des corpus d'évaluation (hormis le corpus de développement) par notre système « élève » pré-entraîné sur notre corpus automatiquement annoté et entraîné sur tous les corpus UFSAC.*

Cette approche est un premier pas pour l'amélioration d'un système de DL comme le nôtre, sans utiliser de données annotées manuellement supplémentaires. Elle l'aide effectivement à mieux généraliser, mais elle souffre encore de défauts, en témoigne la baisse de résultats sur la tâche de SensEval 3. Afin de l'améliorer, nous prévoyons de sélectionner plus finement les données à annoter par le système enseignant, afin de s'adapter à la tâche sur laquelle on souhaite s'évaluer, et de sélectionner les annotations produites par le système enseignant, pour éviter de reproduire les erreurs du modèle neuronal qui peuvent être facilement détectées, par exemple à l'aide d'une mesure de confiance s'appuyant sur sa couche de sortie. Enfin, des architectures neuronales plus évoluées, comme par exemple les réseaux récurrents à base d'arbres (Tai *et al.*, 2015), pourraient permettre d'améliorer notre système en tenant mieux compte de la structure syntaxique des phrases en entrée.

7. Conclusion

Nous présentons dans cet article une nouvelle architecture de réseau neuronal pour la désambiguïsation lexicale à base de cellules LSTM. Ces cellules récurrentes sont largement utilisées dans les réseaux de neurones traitant des séquences tels que les systèmes *sequence-to-sequence* pour la traduction automatique ou les systèmes utilisant un modèle de langue prédisant la prochaine entrée d'une suite de mots. Notre modèle est composé d'une couche d'entrée qui prend une séquence de vecteurs de mots construits séparément, puis une couche cachée de cellules LSTM bidirectionnelles, et enfin une couche de sortie entièrement connectée de la taille du nombre de sens possibles dans le dictionnaire utilisé.

Nous avons entraîné un système sur six corpus au format UFSAC, à savoir le SemCor, le DSO, le WNGT, l'OMSTI, le MASC et l'OntoNotes, mais aussi un système sur le SemCor uniquement, et nous les avons évalués sur les tâches de DL des campagnes d'évaluation SensEval-SemEval. Les résultats montrent que nos systèmes obtiennent des scores équivalents à ceux des meilleurs systèmes neuronaux de l'état de l'art. Seul le système de Yuan *et al.* (2016) augmenté par les données issues de leur propagation de labels obtient des scores plus élevés. Cette augmentation indépendante de leur architecture neuronale s'appuie cependant sur l'utilisation de grandes quantités de textes pris aléatoirement sur le Web, ce qui rend la reproductibilité difficile.

Nous avons présenté une analyse fine des résultats de nos deux systèmes afin de comprendre l'impact du choix des corpus d'entraînement pour l'apprentissage. Nous montrons ainsi les effets positifs d'utiliser davantage de données annotées en sens, mais aussi les problèmes que peuvent apporter des données de moins bonne qualité. Nous espérons ainsi voir se développer des approches supervisées utilisant d'autres corpus que le SemCor.

Enfin, nous avons présenté une amélioration de notre système à l'aide d'une approche par transfert de connaissances pour laquelle seulement un million de phrases initialement non annotées étaient ajoutées aux données d'entraînement afin d'obtenir un modèle plus robuste et performant. Nous avons présenté des résultats avec ce système qui surpassent significativement l'état de l'art sur la plupart des tâches d'évaluation de la DL, et nous avons proposé quelques pistes d'amélioration futures pour continuer dans cette voie.

Les études sur les réseaux de neurones pour la DL sont encore très récentes comme en atteste le faible nombre de systèmes existants pour le moment. C'est cependant une direction prometteuse, tant les résultats obtenus par ces nouveaux systèmes ont montré leur qualité sur les campagnes d'évaluation.

8. Bibliographie

Brody S., Lapata M., « Good Neighbors Make Good Senses : Exploiting Distributional Similarity for Unsupervised WSD », *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, Manchester, UK, p. 65-72, 2008.

Buciluǎ C., Caruana R., Niculescu-Mizil A., « Model compression », *Proceedings of the 12th international conference on Knowledge discovery and data mining*, p. 535-541, 2006.

Chan Y. S., Ng H. T., Chiang D., « Word Sense Disambiguation Improves Statistical Machine Translation », *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, p. 33-40, 2007a.

Chan Y. S., Ng H. T., Zhong Z., « NUS-PT : Exploiting Parallel Texts for Word Sense Disambiguation in the English All-words Tasks », *Proceedings of the 4th International Workshop on Semantic Evaluations*, Association for Computational Linguistics, p. 253-256, 2007b.

Decadt B., Hoste V., Daelemans W., Bosch A. V. d., « GAMBL, genetic algorithm optimization of memory-based WSD », *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, 2004.

Edmonds P., Cotton S., « SENSEVAL-2 : Overview », *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, SENSEVAL '01, Association for Computational Linguistics, Stroudsburg, PA, USA, p. 1-5, 2001.

Hochreiter S., Schmidhuber J., « Long Short-Term Memory », *Neural Computation*, vol. 9, n° 8, p. 1735-1780, 1997.

Hoste V., Kool A., Daelemans W., « Classifier Optimization and Combination in the English All Words Task », *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, p. 83-86, 2001.

Hovy E., Marcus M., Palmer M., Ramshaw L., Weischedel R., « OntoNotes : The 90% Solution », *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume : Short Papers*, p. 57-60, 2006.

Iacobacci I., Pilehvar M. T., Navigli R., « Embeddings for Word Sense Disambiguation : An Evaluation Study », *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 897-907, 2016.

Ide N., Baker C., Fellbaum C., Fillmore C., Passonneau R., « MASC : the Manually Annotated Sub-Corpus of American English », *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, 2008.

Kågebäck M., Salomonsson H., « Word Sense Disambiguation using a Bidirectional LSTM », *5th Workshop on Cognitive Aspects of the Lexicon (CogALex)*, 2016.

Kilgarriff A., « SENSEVAL : An Exercise in Evaluating Word Sense Disambiguation Programs », *First international conference on language resources and evaluation*, 1998.

Kingma D. P., Ba J., « Adam : A Method for Stochastic Optimization », *Proceedings of the 3rd International Conference for Learning Representations*, 2015.

Lesk M., « Automatic sense disambiguation using MRD : how to tell a pine cone from an ice cream cone », *Proceedings of SIGDOC '86*, ACM, New York, NY, USA, p. 24-26, 1986.

Liu F., Lu H., Neubig G., « Handling Homographs in Neural Machine Translation », *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, p. 1336-1345, 2018.

Mikolov T., Sutskever I., Chen K., Corrado G. S., Dean J., « Distributed Representations of Words and Phrases and their Compositionality », *Advances in Neural Information Processing Systems 26*, p. 3111-3119, 2013.

Miller G. A., « WordNet : a lexical database for English », *Communications of the ACM*, vol. 38, n° 11, p. 39-41, 1995.

Miller G. A., Leacock C., Tengi R., Bunker R. T., « A semantic concordance », *Proceedings of the workshop on Human Language Technology*, HLT '93, Association for Computational Linguistics, Stroudsburg, PA, USA, p. 303-308, 1993.

Moro A., Navigli R., « SemEval-2015 Task 13 : Multilingual All-Words Sense Disambiguation and Entity Linking », *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, p. 288-297, 2015.

Navigli R., « WSD : a Survey », *ACM Computing Surveys*, vol. 41, n° 2, p. 1-69, 2009.

Navigli R., Jurgens D., Vannella D., « SemEval-2013 Task 12 : Multilingual Word Sense Disambiguation », *Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, p. 222-231, 2013.

Navigli R., Litkowski K. C., Hargraves O., « SemEval-2007 Task 07 : Coarse-Grained English All-Words Task », *SemEval-2007*, Prague, Czech Republic, p. 30-35, June, 2007.

Navigli R., Ponzetto S. P., « BabelNet : Building a very large multilingual semantic network », *48th annual meeting of the association for computational linguistics*, p. 216-225, 2010.

Ng H. T., Lee H. B., « DSO Corpus of Sense-Tagged English », 1997.

Pennington J., Socher R., Manning C. D., « GloVe : Global Vectors for Word Representation », *Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532-1543, 2014.

Pradhan S. S., Loper E., Dligach D., Palmer M., « SemEval-2007 Task 17 : English Lexical Sample, SRL and All Words », *Proceedings of the 4th International Workshop on Semantic Evaluations*, p. 87-92, 2007.

Raganato A., Delli Bovi C., Navigli R., « Neural Sequence Learning Models for Word Sense Disambiguation », *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, p. 1167-1178, 2017.

Rios A., Mascarell L., Sennrich R., « Improving Word Sense Disambiguation in Neural Machine Translation with Sense Embeddings », *Proceedings of the Second Conference on Machine Translation, Volume 1 : Research Papers*, Copenhagen, Denmark, 2017.

Snyder B., Palmer M., « The English all-words task », *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, 2004.

Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., « Dropout : A Simple Way to Prevent Neural Networks from Overfitting », *J. Mach. Learn. Res.*, vol. 15, n° 1, p. 1929-1958, 2014.

Sutskever I., Vinyals O., Le Q. V., « Sequence to Sequence Learning with Neural Networks », *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, MIT Press, Cambridge, MA, USA, p. 3104-3112, 2014.

Taghipour K., Ng H. T., « One Million Sense-Tagged Instances for Word Sense Disambiguation and Induction », *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, Association for Computational Linguistics, p. 338-344, 2015.

Tai K. S., Socher R., Manning C. D., « Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks », *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, p. 1556-1566, 2015.

Vial L., Lecouteux B., Schwab D., « UFSAC : Unification of Sense Annotated Corpora and Tools », *Language Resources and Evaluation Conference (LREC)*, 2018.

Vial L., Tchechmedjiev A., Schwab D., « Extension lexicale de définitions grâce à des corpus annotés en sens », *23ème Conférence sur le Traitement Automatique des Langues Naturelles*, 2016.

Yarowsky D., « Unsupervised Word Sense Disambiguation Rivaling Supervised Methods », *33rd Annual Meeting on Association for Computational Linguistics*, p. 189-196, 1995.

Yuan D., Richardson J., Doherty R., Evans C., Altendorf E., « Semi-supervised Word Sense Disambiguation with Neural Models », *COLING 2016*, 2016.

Zhong Z., Ng H. T., « It Makes Sense : A Wide-coverage Word Sense Disambiguation System for Free Text », *Proceedings of the ACL 2010 System Demonstrations*, p. 78-83, 2010.

Zhong Z., Ng H. T., « Word Sense Disambiguation Improves Information Retrieval », *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, p. 273-282, 2012.