Désambiguïsation lexicale automatique des verbes du français

Vincent Segonne

Université de Paris

École doctorale de Sciences du Langage (ED 622)

Laboratoire de Linguistique Formelle (LLF)

French Verb Sense Disambiguation

Par Vincent Segonne

Thèse de doctorat de Linguistique

Dirigée par Benoît Crabbé

Présentée et soutenue publiquement le 16/12/2021

Devant un jury composé de :

Philippe Langlais	Professeur	Université de Montréal	Rapporteur
Emmanuel Morin	Professeur	Université de Nantes	Rapporteur
Marianna Apidianaki	Chercheur Senior	Université de Pennsylvanie	Examinatrice
Didier Schwab	Maître de conférence	Université de Grenoble Alpes	Examinateur
Marie Candito	Maître de conférence	Université de Paris	Co-encadrante
Benoît Crabbé	Professeur	Université de Paris	Directeur de thèse

Résumé long

Désambiguïsation automatique des verbes du français

Cette thèse traite de la désambiguïsation du sens des mots, en anglais *Word Sense Disambiguation* (WSD). Nous proposons plus précisément un cadre de travail pour la désambiguïsation des verbes du français. La tâche de WSD est une tâche du Traitement Automatique des Langues (TAL) qui consiste à prédire automatiquement le sens d'un mot compte tenu de son contexte et selon un inventaire de sens prédéfini. Prenons l'exemple de la phrase suivante :

L'avion a **volé** près de 12 000 km avant d'être à court de carburant.

Étant donné le mot cible *volé* à désambiguïser dans cette phrase et un inventaire de sens volontairement simplifié qui ne décrirait que deux sens (1) dérober quelque chose à quelqu'un et (2) se mouvoir dans l'air, la tâche consiste alors pour un système à prédire automatiquement l'un de ces deux sens en tenant compte du contexte. Bien que les humains puissent résoudre de telles ambiguïtés de manière presque inconsciente, cela reste une tâche difficile pour les ordinateurs, en particulier lorsqu'il s'agit de la désambiguïsation des verbes (Raganato et al., 2017b).

La désambiguïsation du sens des mots est un problème intéressant en soi, mais la résolution de cette tâche pourrait également profiter à différentes applications du traitement automatique des langues, comme la traduction automatique, la recherche d'informations ou les systèmes de questions-réponses. Même si la tendance actuelle en TAL est d'utiliser des méthodes de bout en bout basées sur des réseaux de neurones plutôt que de mettre au point des ensembles de systèmes spécifiques par tâche (étiquetage morpho-syntaxique, analyse syntaxique, co-référence, WSD, etc), produire une désambiguïsation explicite présente l'avantage d'apporter plus d'interprétabilité. Il s'agit là d'un aspect important en TAL, d'autant plus que la plupart des modèles d'apprentissage profond, souvent décrits comme des "boîtes noires", produisent des résultats qui restent encore très difficiles à interpréter.

Le succès de la tâche WSD repose en grande partie sur la disponibilité des données. Premièrement, il est nécessaire d'avoir accès à un inventaire de sens de grande qualité (c'est-à-dire avec une bonne couverture et des informations pertinentes) qui peut être facilement manipulé par les ordinateurs. Deuxièmement, les meilleurs systèmes de désambiguïsation utilisent des méthodes d'apprentissage automatique supervisées qui reposent sur des données annotées manuellement. De part leur coût en temps et en ressources humaines, ces deux types de ressources sont très difficiles à produire et, pendant des décennies, en raison du manque de données annotées dans la plupart des langues, la recherche en matière de WSD s'est essentiellement concentrée sur

l'anglais, tirant parti de l'inventaire des sens de Wordnet (Miller et al., 1990) et de SemCor (Miller et al., 1993) comme données annotées.

Dans cette thèse, nous proposons une méthode pour effectuer la désambiguïsation du sens des mots pour les langues dont les ressources en WSD sont limitées, en prenant le français comme exemple. De plus, nous avons choisi de nous concentrer sur les verbes pour plusieurs raisons : Premièrement, parmi toutes les parties du discours, les verbes restent les plus difficiles à désambiguïser et nous voulons faire un pas en avant vers la réussite de cette tâche. Deuxièmement, la désambiguïsation des verbes soulève des questions linguistiques intéressantes, en particulier en ce qui concerne le rôle de la sous-catégorisation des verbes : étant traditionnellement sous-catégorisés en fonction des caractéristiques syntaxiques de leurs arguments, les verbes sont de bons candidats pour étudier le rôle de la syntaxe dans la désambiguïsation. En effet, avant l'émergence des réseaux de neurones, la théorie traditionnelle pour la tâche de désambiguïsation des verbes, en anglais *Verb Sense Disambiguation* (VSD), s'est principalement appuyée sur la structure argumentale des verbes pour résoudre l'ambiguïté. Nous proposons dans cette thèse une analyse contrastive de la corrélation entre la syntaxe et le sens d'une part et les expériences avec les méthodes neuronales d'autre part.

Cette thèse s'organise en 4 chapitres. Les deux premiers chapitres ont pour but de donner au lecteur une bonne connaissance de la tâche de WSD et des différentes méthodes utilisées pour la résoudre tandis que les deux derniers décrivent les expériences que nous avons réalisées.

Dans le chapitre 2, nous proposons une présentation générale de la WSD. Le but étant d'introduire au lecteur le cadre de travail dans lequel cette thèse a été écrite. Il présente à la fois le domaine et les motivations qui ont mené à l'aboutissement de ce travail. Nous commençons par introduire la tâche de désambiguïsation lexicale, l'objectif ici n'est pas tant de rappeler tout l'historique et les méthodes de WSD mais plutôt de proposer une vue générale de la tâche et de ses enjeux. La deuxième partie de ce chapitre quant à elle se concentre sur l'objectif de cette thèse: la désambiguïsation des verbes du français. Nous discutons du cas particulier de la tâche de désambiguïsation des verbes qui par ailleurs a fait l'objet d'une étude à part entière. Enfin, nous abordons le problème de la désambiguïsation des verbes pour le français. Nous donnons les motivations qui nous ont poussé à travailler sur ce sujet mais aussi les diverses difficultés auxquelles nous devons faire face.

Le chapitre 3 propose un état de l'art de la tâche de WSD. Cette tâche s'appuyant grandement sur l'utilisation de ressources annotées, nous commençons par faire le tour des ressources disponibles anglaises et multilingues. Nous présentons deux types de ressources: D'un côté les inventaires de sens avec Wordnet pour l'anglais et BabelNet(Navigli and Ponzetto, 2010) pour le multilingue. D'un autre côté les données annotées en sens avec Semcor, WNGT, OMSTI(Taghipour and Ng, 2015a)

pour l'anglais et Eurosense(Bovi et al., 2017) pour le multilingue. Par ailleurs, nous détaillons également les différents jeux de données utilisés pour l'évaluation de la tâche principalement issus des campagnes d'évaluation SensEval et SemEval. Pour chacune de ces ressources, nous tentons de donner une description concise appuyée par des statistiques afin de donner au lecteur une bonne appréciation des avantages et inconvénients de chacune d'entre elles. Ensuite, nous nous concentrons sur la représentation du contexte, un élément clé pour la résolution de la tâche. Nous décrivons plusieurs méthodes de représentation du contexte principalement basées sur des réseaux neurones. La présentation de ces méthodes se fait de manière progressive: nous commençons par l'utilisation de "simples" word embeddings issus du modèle Word2vec (Mikolov et al., 2013b) puis nous présentons les réseaux de neurones récurrents de type RNN/LSTM (Hochreiter and Schmidhuber, 1997a) qui permettent d'encoder la séquence d'une phrase. Nous terminons par présenter les modèles de langues pré-entraînés basés sur les transformers (Vaswani et al., 2017; Devlin et al., 2018). Ces modèles très lourds (à la fois en nombre de paramètres et en données d'apprentissage) ont marqué un nouveau tournant (comme l'a pu être l'avènement des word embeddings) dans le domaine du TAL. Ils ont en effet permis d'améliorer de façon significative les performances dans la plupart des tâches, y compris en WSD où les représentations contextuelles jouent un rôle crucial. Néanmoins, ces modèles demeurent très opaques et il est assez difficile de bien comprendre pourquoi ceux-ci fonctionnent si bien. La fin de ce chapitre décrit les deux branches de méthodes les plus utilisées pour résoudre la tâche: la désambiguïsation supervisée et la désambiguïsation basée sur la connaissance. Pour chacune de ces branches nous décrivons le principe de la méthode et faisons le tour des systèmes les plus performants.

Dans le chapitre 4, nous nous concentrons sur la désambiguïsation des verbes, et en particulier sur l'interaction entre syntaxe et distinctions de sens. Plus précisément, notre but est d'évaluer le rôle de la structure argumentale des verbes dans la discrimination de leur sens. Les ressources annotées à la fois avec des informations syntaxiques et sémantiques étant très limitées dans la plupart des langues (ce qui est le cas du français où celles-ci sont quasi-inexistantes), les expériences réalisées dans ce chapitre portent sur l'anglais uniquement. La première étape de cette investigation consiste en une étude statistique de corpus dans laquelle nous étudions la corrélation entre la structure argumentale des verbes et leur sens. Les résultats de cette analyse de corpus montrent que la syntaxe seule ne suffit pas à faire la désambiguïsation des sens des verbes. En revanche, la combinaison d'informations syntaxiques et lexicales obtenues depuis les arguments syntaxiques des verbes semble corréler avec la distinction des sens. Ensuite, nous abordons la question de la syntaxe et de la désambiguïsation via l'utilisation de réseaux de

neurones. Nous étudions en particulier les réseaux de neurones basés sur des mécanismes d'attention et analysons leurs représentations internes afin de voir s'il existe une trace de l'encodage de la structure argumentale des verbes. En pratique, il s'agit de voir si la structure argumentale est représentée dans les poids d'attention. Nous utilisons deux types de modèles. Tout d'abord, un modèle très simple constitué d'une seule couche d'attention. L'objectif ici est d'utiliser un modèle qui puisse être le plus intelligible possible. Les résultats de cette expérience sont assez mitigés, il n'y pas de preuve significative que la structure argumentale des verbes soit encodée dans les poids d'attention de ce modèle. Nous comparons ensuite ces résultats à un modèle de type BERT (Devlin et al., 2018) qui est bien plus sophistiqué et plus performant mais également beaucoup plus difficile à analyser de par sa complexité. Même si quelques fonctions syntaxiques relevant de la structure argumentale (comme par exemple l'objet direct) semblent être bien identifiées, la plupart de ces fonctions capturées par le modèle sont en fait corrélées à des patterns beaucoup plus simples comme l'adjacence des mots.

Dans la dernière partie de ce chapitre, nous réalisons des expériences dont le but est d'intégrer des informations syntaxiques directement dans la représentation du contexte et d'évaluer son impact sur les performances d'un classifieur de WSD. Pour cela, nous proposons Dag2vec, un modèle qui apprend des représentations contextuelles à partir de structures syntaxiques via des réseaux de neurones récurrents de type LSTM (Hochreiter and Schmidhuber, 1997a) adaptés pour pouvoir encoder des graphes de dépendances. Après avoir entraîné Dag2vec sur un large corpus annoté automatiquement en arbres de dépendances, nous testons les représentations apprises sur la tâche de WSD. Nous n'observons aucune amélioration particulière sur les verbes (par rapport à un encodage "linéaire" classique). Toutes ces expériences nous ont mené à conclure qu'il existait bien un lien entre la distinction des sens des verbes et leur structure argumentale mais que l'encodage de celle-ci par un réseau de neurones restait encore à être trouvé.

Le chapitre 5 traite du problème de la rareté des données en WSD pour une langue autre que l'anglais, en prenant le cas du français comme exemple. Nous commençons par explorer diverses ressources multilingues qui ont été produites automatiquement. En particulier, nous nous intéressons à BabelNet et Eurosense. À travers une petite évaluation manuelle, nous avons conclu que celles-ci n'étaient pas en l'état exploitables pour notre tâche. Nous proposons à la place d'utiliser Wiktionary, la version dictionnaire multilingue de Wikipedia, qui offre à la fois un inventaire de sens et des exemples annotés en sens manuellement et ce pour un nombre substantiel de langues. Nous évaluons sa viabilité pour la tâche de désambiguïsation des verbes du français à l'aide d'un nouveau jeu de données d'évaluation que nous avons appelé FrenchSe-

mEval (FSE) (Segonne et al., 2019). Ce corpus est une contribution de cette thèse, il contient 3199 occurrences de verbes annotées manuellement avec les sens de l'inventaire de sens de Wiktionary. Nous avons réalisé plusieurs expériences sur FSE, évaluant un classifieur supervisé avec des représentations contextuelles différentes. En particulier, nous avons évalué les représentations de plusieurs modèles de langue pré-entrainés basés sur des transformers dont FlauBERT (Le et al., 2020), une adaptation pour le français de modèle BERT, auquel nous avons contribué. Les résultats, bien qu'encore en dessous d'un niveau acceptable de désambiguïsation (environ 50% seulement des occurrences sont correctement désambiguïsées), vont pouvoir servir de références pour des travaux futurs.

Pour conclure, dans cette thèse nous nous sommes intéressés au problème de la désambiguïsation lexicale et plus particulièrement à la désambiguïsation lexicale des verbes du français. Nous avons été confrontés à un sujet difficile puisque les recherches et les données pour cette tâche étaient quasi-inexistantes. Tout au long de ce manuscrit, nous avons abordé plusieurs aspects clés de la tâche de WSD: la représentation du contexte, les algorithmes de désambiguïsation, le lien entre syntaxe et sens, la rareté des données annotées. Les contributions de cette thèse sont multiples: Tout d'abord, nous avons produit des ressources, un jeu de données d'évaluation pour la tâche de désambiguïsation des verbes. Ce jeu de données distribué librement a trouvé sa place au sein de *Flue* (Le et al., 2020), un corpus regroupant divers jeux de données d'évaluation pour des tâches en lien avec la compréhension de la langue. Nous avons par ailleurs contribué à l'élaboration de FlauBERT, un modèle de langue pré-entraîné basé sur les transformers pour le français et l'avons évalué sur notre jeu de données. Ensuite, nous avons fait une étude poussée sur l'apport de la syntaxe pour la désambiguïsation des verbes. Cette étude, reposant à la fois sur des analyses statistiques de corpus et sur des expériences à base de réseaux de neurones a permis de mettre en lumière un certain nombre de phénomènes et de mieux comprendre l'interaction entre syntaxe et distinction de sens des verbes. Enfin, nous avons proposé une méthode qui, sans être complètement satisfaisante, permet d'initier le travail de recherche sur la désambiguïsation des verbes du français.

Mots clés: Traitement automatique des langues, TAL, désambiguïsation lexicale, WSD, apprentissage automatique, apprentissage profond, corpus, syntaxe, sémantique

Résumé court

Désambiguïsation automatique des verbes du français

La désambiguïsation lexicale est une tâche du traitement automatique des langues dont l'objectif est de prédire automatiquement le sens des mots en contexte, à partir d'un inventaire de sens prédéfini. La réussite de cette tâche repose en particulier sur l'utilisation de ressources lexicales et de données annotées en sens. Par ailleurs, le récent essor des méthodes d'apprentissage automatique par réseaux de neurones profonds a grandement amélioré les performances des systèmes de désambiguïsation.

Dans cette thèse, nous nous concentrons sur la désambiguïsation des verbes du français, une langue qui ne dispose pas ou peu, à priori, de données utilisables pour cette tâche. Pour commencer, nous faisons un état de l'art des principales méthodes neuronales de représentation du contexte ainsi que des méthodes de désambiguïsation.

Puis, nous nous intéressons à la question du rôle de la syntaxe pour la désambiguïsation des verbes. Pour cela, nous commençons par étudier en corpus la potentielle corrélation entre le sens et la structure argumentale des verbes. Nous tentons ensuite de voir si la structure argumentale des verbes est encodée dans les représentations contextuelles issues de réseaux de neurones. Nous proposons également un modèle qui apprend des représentations contextuelles étant données des structures syntaxiques de phrases obtenues *a priori* par un analyseur syntaxique et nous les testons sur la tâche de désambiguïsation.

Enfin, dans la dernière partie de cette thèse, nous abordons le problème de la disponibilité des données pour la tâche de désambiguïsation dans une langue autre que l'anglais en prenant le français pour exemple. Après avoir étudié diverses ressources produites automatiquement, nous proposons d'utiliser Wiktionary, une ressource libre et collaborative sur le modèle de Wikipédia, afin de produire FrenchSemEval, le premier corpus d'évaluation pour la tâche de désambiguïsation des verbes du français. Nous testons plusieurs systèmes de désambiguïsation sur ce jeu de données et obtenons les tout premiers résultats pour cette tâche.

Mots clés: Traitement automatique des langues, TAL, désambiguïsation lexicale, WSD, apprentissage automatique, apprentissage profond, corpus, syntaxe, sémantique

Abstract

French Verb Sense Disambiguation

Word Sense Disambiguation (WSD) is a Natural Language Processing (NLP) task which goal is to automatically predict the meaning of words in context, based on a predefined inventory of word senses. The success of this task relies on the use of lexical resources and sense annotated data. Moreover, the recent development of contextual representations based on learning with deep neural networks has greatly improved the performance of disambiguation systems.

In this thesis, we focus on the disambiguation of verbs in French, a language that has little or no viable data for this task. First, we review the state of the art of neural net based contextual representations and disambiguation methods.

Then, we investigate the role of syntax for the disambiguation of verbs. To do so, we first perform a corpus study exploring the potential correlation between the argument structures of verbs and their senses. We then study whether the argument structure of verbs is encoded in contextual representations obtained from attention-based neural networks. We also propose a model that learns contextual representations from syntactic structures of sentences provided *a priori* by a parser and test them on the disambiguation task.

Finally, in the last part of this thesis, we address the problem of data availability regarding the WSD task for any language other than English, using French as an example. After studying various automatically produced resources, we propose to use Wiktionary, a free and collaborative dictionary based on the Wikipedia model, and release FrenchSemEval, the first evaluation corpus for the French verb disambiguation task. We evaluate several disambiguation systems on this dataset and obtain the very first results for this task.

keywords: Natural Language Processing, NLP, Word Sense Disambiguation, WSD, machin learning, deep learning, corpus, syntax, semantics

Remerciements

Avant toute chose, je souhaite remercier mon directeur Benoit Crabbé et ma directrice Marie Candito pour leur formidable encadrement. J'ai bien conscience de la chance que j'ai eu d'avoir des encadrants aussi disponibles qui ont su me donner à la fois une liberté de recherche et un soutien infaillible tant sur le plan académique qu'émotionnel. Si cette thèse voit le jour c'est avant tout grâce à eux.

Je suis également très honoré d'avoir pu présenter mes travaux devant un jury de thèse si bienveillant. Je remercie en particulier Philippe Langlais et Emmanuel Morin d'avoir accepté de rapporter cette thèse avec intérêt et pertinence. D'autre part, je remercie Marianna Apidianaki, examinatrice et membre de mon comité de suivi, avec qui j'ai pu avoir des échanges critiques et constructifs, en particulier sur la WSD. J'en profite pour remercier Mathieu Constant également membre de mon comité de suivi.

Merci à Didier Schwab d'avoir participé à ce jury de thèse en tant qu'examinateur et de m'avoir si bien accueilli sur le projet FlauBERT avec l'équipe de l'Université Grenoble Alpes que je salue également.

Cette thèse a été faite au Laboratoire de Linguistique (LLF) Formelle à l'Université de Paris et je tiens à saluer et remercier tous les membres du laboratoire qui font de celui-ci un cadre de travail exceptionnel. En particulier, je remercie Olivier Bonami, directeur du LLF du temps de l'écriture de cette thèse, qui a su me donner les meilleures conditions de travail à un moment où le contexte sanitaire était très difficile. Merci également à Lucie Barque pour ses conseils, son expertise sur la sémantique lexicale et avec qui j'ai eu le plaisir d'avoir des discussions amicales enrichissantes.

Je salue tous mes collègues doctorants que j'ai pu croiser et avec qui j'ai apprécié partager mon bureau pendant ces quatre années. En particulier, je suis très heureux d'avoir pu discuter, partager mes doutes et mes repas avec Maximin, Sasha, Olga, Timothée, Charlotte, Justine, Juliette et Antoine.

Je souhaite remercier chaleureusement tous mes anciens professeurs (dont certains sont maintenant devenus des collègues et/ou des amis) du cursus de linguistique-informatique. Je remercie en particulier Pascal Amsili et Laurence Danlos de m'avoir donné envie d'évoluer dans ce domaine exceptionnel qu'est celui du TAL.

Un grand merci à ma famille et mes amis qui m'ont toujours soutenu et pour qui je serai sans doute toujours un éternel étudiant ! Mais finalement, ne sommes-nous pas tous un peu étudiant toute notre vie ?

Enfin, je te remercie, toi, qui crois toujours en moi et qui n'a cessé pendant ces quatre années de me montrer la lumière tel un phare dans la nuit, fut-elle noire ou blanche.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis deals with Word Sense Disambiguation (WSD). We propose a framework that focuses on the disambiguation of French verbs. The WSD task is a task in Natural Language Processing (NLP) which consists in automatically predicting the meaning of a word given its context and according to a predefined inventory of senses. Let's take the following sentence as an example:

I have **run** 20 kilometers every week for half of my life.

Given the target word *run* to be disambiguated in this sentence and a voluntarily simplified sense inventory that would describe only two senses (1) to cover a distance and (2) to operate a machine, then the task is for a system to predict one of these two senses by taking the context into account. Although humans can resolve such ambiguities mostly unconsciously, it remains a challenging task for computers, particularly when it comes to the disambiguation of verbs (Raganato et al., 2017b).

Word sense disambiguation is an interesting problem in itself, but solving this task could also benefit downstream tasks in NLP such as machine translation, information retrieval, and question-answering. Even though the current trend in NLP is to use neural net based end-to-end methods rather than ensembles of task-specific systems (PoS-tagging, parsing, coreference, WSD, etc), performing explicit WSD presents the advantage of bringing more interpretability. This is an important aspect in NLP, especially given the fact that most deep learning models, often described as "black boxes", produce results that still remain very difficult to interpret.

The success of the WSD task relies heavily on the availability of data. Firstly, it is necessary to have access to a sense inventory of high quality (i.e with a good coverage and relevant information) that can be easily manipulated by computers. Secondly, the best disambiguation systems use supervised machine learning methods that rely on manually sense annotated data. Due to their human and time cost, these two types of resources are very difficult to produce and for decades, because of the lack of annotated data in most of the languages, research in WSD has

essentially focused on English taking advantage of the sense inventory from Wordnet (Miller et al., 1990) and SemCor (Miller et al., 1993) as annotated data.

In this thesis, we propose a solution to perform word sense disambiguation for languages with limited WSD resources, taking French as an example. Moreover, we choose to focus on verbs for several reasons: First, among all parts of speech, verbs remain the most difficult to disambiguate and we want to take a step forward for the success of this task. Secondly, the disambiguation of verbs raises interesting linguistic questions, in particular regarding the role of verbs' subcategorization: being traditionally subcategorized according to their arguments' syntactic characteristics, verbs are good candidates to investigate the role of syntax for disambiguation. Indeed, the traditional VSD theory had mostly relied on verbs' argument structure to help performing their disambiguation, until the emergence of neural network-based WSD systems. We propose in this thesis a contrastive analysis of the correlation between syntax and meaning on the one hand and experiments with neural methods on the other hand.

This thesis is structured around 4 chapters which we briefly describe in the following paragraphs.

In Chapter 2 we start by introducing a minimum background of the word sense disambiguation task thus providing the knowledge necessary to understand the issues at stake. First, we present a formal description of the WSD task and introduce several methods to solve it. Having presented the general framework of the WSD task we then turn to the subject of our concern, the disambiguation of French verbs. We discuss with further details our motivations and the challenges that we will be confronted with.

In Chapter 3, we present an overview of the state of the art for the WSD task. Since WSD highly relies on data, we first present the most prominent existing resources (English and multilingual) for WSD. Then, we focus on context representation, a key element for success in this task. We describe various neural net based methods to encode the context of words starting from the simple use of word embeddings to the most recent and sophisticated attention-based pre-trained language models. Finally, we give details on the two most successful branches of disambiguation methods, namely knowledge-based and supervised methods.

Chapter 4 focuses on the disambiguation of verbs and in particular on the interaction between syntax and sense distinctions. More precisely, we aim at asserting the role of the argument structure of verbs in the discrimination of their senses. The first step consists in a corpus study investigating the correlation between the argument structure and the verbs' senses. Then, we confront the results from the corpus study to multiple experiments performed with neural networks. We focus on attention-based models and investigate whether the argument structure

is encoded within their inner representations (i.e if the argument structure of verbs is reflected in their attention weights). In the last part of this chapter, we evaluate whether a syntax-infused contextual representation can be beneficial to a WSD classifier. To do so, we propose Dag2vec, a model which learns contextual representations from syntactic structures using recurrent neural networks adapted to process dependency graphs. After training our model on a large corpus enriched with predicted dependency trees, we test the resulting syntax-based contextual representations on the WSD task.

In Chapter 5 we address the issue of data scarcity for languages other than English, studying the case of French. We first explore various multilingual resources that were automatically produced. We then investigate and propose to use Wiktionary, a multilingual dictionary version of Wikipedia, that can provide both a sense inventory and training annotated data, for a substantial number of languages. We assess its usability for the French VSD task on a new evaluation corpus we called FrenchSemEval (FSE) (Segonne et al., 2019) which consists of manual annotations of verb occurrences with senses from Wiktionary. We perform several experiments on FSE, using various contextual representations to provide the very first results on this dataset. In particular, we evaluate several transformer-based pre-trained language models, including FlauBERT (Le et al., 2020), an adaptation for French of the original BERT model (Devlin et al., 2018), which we contributed to.

Some of the work and experiments presented in this thesis have been previously published or submitted in the following articles:

- ”Using Wiktionary as a resource for WSD : the case of French verbs” Segonne et al., IWCS 2019
- ”FlauBERT: Unsupervised Language Model Pre-training for French” Le et al., LREC 2020
- ”Does Bert Encode The Argument Structure of Verbs ?” Segonne et al., submitted.

Chapter 2

Background

Contents

2.1 Introduction

The purpose of this chapter is to introduce the reader to the framework within which this thesis was written. It presents both the field and the motivations that led us to the completion of this work. In Section 2.2 we present the Word Sense Disambiguation (WSD) task. The intent here is not to recall the whole history and methods of WSD, but rather to propose a general overview of the task and the issues at stake. WSD is now well documented and we highly advise the reader to explore the complete survey written by Navigli (2009) for more details.

The second part of this chapter (Section 2.3) is an introduction to the objective of this dissertation: the disambiguation of French verbs. We shortly review the particularities of verbs and present the Verb Sense Disambiguation (VSD) task (Section 2.3.1).

Finally, in Section 2.3.2 we discuss the motivations and challenges that our approach entails.

2.2 Word Sense Disambiguation

2.2.1 Introduction

The human language is naturally **ambiguous**. Yet, in the vast majority of cases humans are able to solve that ambiguity based on context. At the word level, the ambiguity is exhibited by the possibility for a single word form to have multiple meanings according to the context of the sentence. Here is a minimal example with the noun *bass* frequently cited in the literature:

(1) a. He plays the *bass* in a rock band called Teacup Monster.
b. The *bass* of my new amplifier is way to loud!
c. I caught a *bass* in the lake early this morning.

In these sentences, the word *bass* is **polysemous** since it can refer to three different meanings: (1a) refers to the music instrument, (1b) to the low pitch and (1c) makes reference to a specific type of fish. Some senses may be semantically connected, for examples looking at (1a) and (1b) one can presume that both senses share a sort of common root sense related to a "low pitched sound". On the contrary, other senses may be completely unrelated, i.e we can not make any direct or indirect link between them (see (1a) and (1c) for example). In the literature, this latter case is usually referred to as **homonymy**. The ambiguity not only concerns nouns but words from all open-class part-of-speech (PoS). In fact, verbs are especially known to be highly polysemous (i.e with a high number of senses). Here is another example of polysemy with the verb *get*.

(2) a. Don't worry, I *get* it. (understand)
b. Did you *get* the newspaper on your way ? (obtain)
c. Eventually they *got* scared and runaway. (become)
d. We need to *get* moving now! (begin, start)
e. This movie *got* me so depressed.. (cause state)
f. Can you *get* the call ? (respond telephone, doorbell)

The examples proposed in (2) show a non-exhaustive list of the multiple meanings of the verb *get*. The number of senses, also called granularity of senses, may differ from a sense inventory (dictionaries, ontologies, thesaurus etc) to another. For example, the word "get" has 27 senses in the Oxford dictionary, 11 in the Cambridge dictionary, 34 in the Collins dictionary and 33 in Wiktionary (the dictionary version of Wikipedia). This is mainly due to the fact that the notion of meaning is not well defined, there is no simple, formal way to describe it. Linguistic tests have been designed as sufficient or necessary conditions for positing different senses (e.g. (Cruse, 2000)), but there is no consensus on these

nor on how to delimit senses. Homonymy cases aside, it is very often difficult to set boundaries between senses. In fact, the variation across the ensemble of existing sense inventories illustrates the complexity of the definition of senses.

The variation does not concern the granularity of senses only, coverage, i.e the vocabulary size, may differ from one resource to another. For example the Oxford dictionary has 273.000 entries while Wordnet (Miller et al., 1990) includes roughly 155.000 entries. These differences may be explained by several factors: the means employed to build the resource (private/academic/open-source projects), the type and features offered by the resource (dictionary, wordnet, ontology) and the targeted applications (learner/expert/encyclopedic dictionaries, specific domains etc.).

The WSD task As early as the 1950's Weaver (1955) with his primary work on machine translation (MT) noticed that resolving the ambiguity of words in an upstream process may help to perform better automatic translations. He also observed that the context of an occurrence is key to the success of the disambiguation of words. Since these observations researchers have gotten more and more interested in this sub-problem, which eventually led to the emergence of the Word Sense Disambiguation (WSD) task. This task is part of the natural language processing (NLP) field and aims at automatically identifying the correct sense of a word occurrence in context, given a predefined sense inventory. Although resolving the ambiguity of words is an interesting problem, it is not an end in itself but rather a pre-process that can benefit downstream tasks such as machine translation and other tasks like information retrieval (IR) or question-answering (QA).

There are several well identified obstacles that are inherent to the WSD task. First, as mentioned earlier, the definition of the meaning of a word is not consensual. As a result, the choice of the sense inventory may have a significant impact on WSD, especially when it has rather arbitrary sense distinctions. Secondly, the distribution of senses in natural text is heterogeneous, some senses are over-represented whereas others are very scarce. Besides, the frequency of the senses may vary depending on the corpus genre which may cause domain adaptation issues to WSD systems.

Last but not least, the disambiguation of words requires an access to a sophisticated linguistic knowledge with underlying semantic and world knowledge which is still difficult to infuse into computers. Furthermore, WSD has been formalized as an artificial intelligence problem (Mallery, 1988) and even if the task was identified almost 70 years ago, it still remains unresolved at the present time.

2.2.1.1 WSD Versus WSI

Word Sense Induction (WSI) is a task whose aim is to induce the meanings of words from unannotated data. It is strongly related to WSD since both tasks deal with the disambiguation of words in context but the methods differ drastically. For one thing, WSI does not assume the pre-existence of a sense inventory, instead the goal is to infer the senses directly from the data itself. This has two main benefits: Firstly, WSI systems are adaptable to any domain, task or language at hand. Then, because the senses are induced from the data, it is by nature representative of their natural distribution, thus avoiding some senses to be left out (Brody and Lapata, 2009). WSI methods mainly rely on unsupervised learning, they usually apply clustering algorithms to group up similar contexts into classes which should be representative of the senses The main disadvantage with WSI systems is their lack of interpretability. Indeed, the clusters output by the systems are generally difficult to understand for humans and, ironically, make no sense to us. Moreover, the evaluation or comparison of different WSI systems is not as easy as with WSD systems since they do not share the same set of classes. Two evaluation campaigns were dedicated to this task (Agirre and Soroa, 2007; Manandhar et al., 2010) and the recent years have witnessed an effort to improve the interpretability of the systems (Ruppert et al., 2015; Panchenko, 2016; Panchenko et al., 2017). In this dissertation, we will leave WSI aside to focus essentially on WSD.

2.2.2 WSD: A classification task

Word Sense Disambiguation is formally described as a classification task (Yarowsky, 2000) where given an input x, the occurrence of a word in context, the automatic system should predict the correct class $y \in Y$, a set of senses from a specific sense inventory. Let us see an example of such a configuration using the sentence from (2b):

$$
\begin{aligned}
x &= \text{“Did you } \underline{\textbf{get}} \text{ the newspaper on your way ?”} \\
\mathrm{Y} &= \{y_1 : \text{understand},\ y_2 : \text{obtain},\ y_3 : \text{become}\} \\
\hat{y} &= \textsc{classify}_{y \in Y}(x)
\end{aligned}
$$

In this toy configuration, x is the input, Y is the set of available senses for the target word *get* which includes only three possible senses (y_1, y_2 and y_3), y is the gold sense and $\hat{y}$ is the prediction output by the system. To resolve such a classification task, we usually proceed in two steps: First, we extract features from the input. In our case, it means extracting features from the context of the target occurrence (often called

context representation). For example, one of the simplest ways to represent the context is to select the co-occurring words on the left and right of the target word. This is referred to as the *context-window* and has been widely used in WSD. Other more sophisticated features may be based on syntax, morphology, semantics etc. Then, given the obtained context representation, a classifier assigns a class (i.e a sense) accordingly. As we mentioned previously, the particularity of WSD is that the success of the classification usually relies on underlying world knowledge that is embedded in the meaning of lexical items, but capturing it in computerized word meaning representations is still an open problem. This is why, as humans, we can resolve this task very easily while it is much more difficult for computers since they do not own that world knowledge. Nevertheless, we will see in the part dealing with the state-of-the-art (chapter 3) that the recent breakthrough of deep neural net models, trained on large amounts of data, leads the way to a better representation of the context. These powerful contextual representations somehow seem to capture this precious world knowledge and consequently improve the quality of the disambiguation.

2.2.3 WSD: methods

2.2.3.1 Supervised and semi-supervised methods

As a classification task, WSD fits particularly well with supervised methods. In a standard supervised WSD configuration, the automatic systems are trained and evaluated on sense annotated data whose senses (the classes to predict) are provided by a pre-defined sense inventory. This is the most widely used approach and it is generally known to outperform other methods (Navigli, 2009). Yet, the good results are obtained at the expense of a strong dependency towards the data which presents two major drawbacks: First, the resulting trained system is completely reliant on the training data, hence it is dependent on its domain and on the sense inventory from which the senses were used to annotate the training instances. As mentioned earlier, there is no consensus as to which sense inventory is the most suited, it rather depends on the targeted objective. Furthermore sense inventories can also evolve over time, making the annotated data and trained systems obsolete. This has been identified as the knowledge acquisition bottleneck in the literature (Gale et al., 1992).

Secondly, even if there was such thing as a consensual sense inventory, WSD supervised systems are likely to suffer from data sparsity since it is hardly reasonable to think, as a first approximation, that we can manually annotate even a minimum number of examples for every sense of every word from a given lexicon.

To overcome the problem of the scarcity of annotated data, some works based on supervised systems attempted to integrate more un-

labeled data. It opened up the way for a new branch in WSD supervised methods called *semi-supervised* methods. The general process of these methods is to automatically expand the senses from annotated data to raw texts in order to build a larger semi-automatically sense annotated corpus that can be used later on by supervised classifiers (Taghipour and Ng, 2015b; Camacho-Collados et al., 2016; Raganato et al., 2016). In various settings semi-supervised systems proved to perform as well as supervised systems (Raganato et al., 2017b).

2.2.3.2 Knowledge based methods

In opposition to supervised methods, knowledge-based systems do not rely on annotated data. Instead, they take advantage of manually curated lexical resources such as dictionaries or semantic networks to disambiguate words. Knowledge-based methods are not recent, Lesk (1986) developed an algorithm, the Lesk's algorithm, which performs word sense disambiguation using the context of the words and dictionaries. Very simply the algorithm compares the overlap between the words from the context of the target word and the words from its definitions found in a dictionary. Since then, several works have proposed more sophisticated versions of Lesk's algorithm (Banerjee, 2002; Basile et al., 2012; Chen et al., 2014). More Recently Luo et al. (2018) proposed a hybrid system between supervised and knowledge-based methods using deep neural networks to encode gloss definitions. Overall knowledge based methods generally obtain lower performances than supervised methods (Raganato et al., 2017b) but they have the serious advantage of being independent from annotated data.

2.3 French Verb Sense Disambiguation

In this section, we present the French Verb Sense Disambiguation task, a subpart of WSD focusing on verbs and applied to the French language. After a short discussion on the place of verbs in NLP, we describe the particularities of the Verb Sense Disambiguation (VSD) task. We then explain the motivations and challenges in French VSD.

2.3.1 Verb Sense Disambiguation

2.3.1.1 The specific case of verbs

The meaning of a sentence is substantially conveyed by verbs, which makes their disambiguation all the more necessary. Moreover, verbs have a particular link with syntax and semantics as their different senses are known to be related with their argument structures (i.g the number of arguments, the syntactic realizations, the semantic roles etc.). This has

been the subject of numerous works and has even led to the construction of dedicated lexical resources such as Verbnet (Kipper et al., 2005). Because verbs are at the junction of the syntax-semantics interface they hold a special place in NLP and a particular treatment may be beneficial, especially to natural language understanding tasks (NLU) such as Semantic Role Labeling (SRL), question-answering (QA), sentiment analysis (SA) and for our concern WSD.

2.3.1.2 From WSD to VSD

Within the word sense disambiguation task, verbs have proven to be very hard to disambiguate as WSD systems consistently reported poor performances on several evaluation campaigns (Kilgarrif, 1998; Pradhan et al., 2007). The main difficulty comes from the fact that verbs tend to have a higher degree of polysemy than other PoS. For example, the sense inventory from Wordnet (Miller et al., 1990) displays an average polysemy (mean number of senses per word type) for verbs of 3.41 while the second most polysemous words are nouns with an average of 2.79 senses. Furthermore, this phenomena is also reflected in natural texts, Raganato et al. (2017b) estimated the level of ambiguity per PoS in several WSD evaluation datasets and pointed out that verbs were much more ambiguous than any other PoS. Consequently, making the distinction between senses is more subtle and more complex. To make things even worse, some verbs with high polysemy degrees are also very frequent, for example one may think of auxiliaries and light verbs like *get*.

The first automatic systems in WSD did not make any distinction between the different part-of-speech, instead they disambiguate verbs just like any other word, consequently leading them to obtain rather bad results. To address this problem researchers working on WSD started to treat the case of verbs apart, eventually conducting to the emergence of a subpart in the WSD task called Verb Sense Disambiguation (VSD). In this subtask, only verbs are the target of disambiguation. Influenced by (Levin, 1993)'s classification of verbs, most of the traditional work in VSD relied on verbs' syntactic and semantic features of their arguments (Dligach and Palmer, 2008; Roberts and Kordoni, 2012; Kawahara and Palmer, 2014; Wagner et al., 2009). More recently the breakthrough of very sophisticated deep neural networks such as BERT (Devlin et al., 2018), successfully integrated into WSD systems (Luo et al., 2018; Du et al., 2019; Vial et al., 2019), has drastically improved the quality of disambiguation setting up a new state-of-the-art on the task, including on verbs. However, these recent models are not verb specific and thus question the need to favor special treatments for verbs.

2.3.2 French VSD: Motivations and challenges

2.3.2.1 WSD: A matter of English only ?

Most WSD systems are based on supervised methods since they are known to obtain the best results. Nevertheless, these methods strongly rely on manually sense annotated corpora. Semcor (Miller et al., 1993) is one of such corpus in which words are annotated with senses from the sense inventory of Wordnet (Miller et al., 1990). It is still to this day the most important manually built semantic resource for English. It gathers more than 200,000 sense annotations from all open-class part-of-speech. Because manually annotating the words of a corpus with their senses is time, effort and money consuming (Navigli, 2009), not to mention difficult, the vast majority of WSD works has essentially been focused on Semcor. As a result, for a long time WSD has remained a task restricted to the English language only.

2.3.2.2 French VSD Motivations and Challenges

As for French, the language of our concern in this dissertation, very little work has been done on WSD, mainly due to the lack of annotated data. French is a good example of languages which are rather well endowed in terms of resources (certainly less than English or Chinese but much more than Icelandic) and yet very poorly provided with sense annotated data for WSD. Again, as mentioned, data scarcity is one key issue in WSD.

The major contributions regarding French annotated data appeared in the multilingual WSD evaluation challenges (Navigli et al., 2013) which provided a small evaluation dataset annotated with senses from Babelnet (Navigli and Ponzetto, 2012), a multilingual semantic network (we present this resource and discuss its usability for French in section 3.2.2). Unfortunately only nouns were annotated and submitted for evaluation. Moreover the campaign aimed at investigating multilingual WSD rather than providing evaluation data for other languages than English.

The quasi non-existence of research in French WSD and particularly on verbs motivated the project of this thesis. In particular, our objective is three-fold: (1) Explore and compare the various multilingual resources available for other languages than English and assert their suitability to perform verb sense disambiguation taking French as an example. This includes both the investigation of an appropriate sense inventory and the development of training/validation data. (2) Investigate the role of syntax for the disambiguation of verbs, especially its impact on the representation of context. (3) Take advantage of state-of-the-art methods to propose a large scale system capable of automatically producing verb sense disambiguated data of good quality. These objectives are challenging given the difficulty of the task already proven on English and the lack of references and data. Nonetheless, with this work we hope to

increase the research interest in French WSD.

Chapter 3

State of the Art

Contents

3.1 Introduction

In this chapter, we propose an overview of the state-of-the-art for the word sense disambiguation task. We start the chapter (Section 3.2) by presenting the available resources for both English and multilingual WSD. In this section we also present the various existing evaluation WSD datasets used to evaluate and compare WSD systems.

In the second section (Section 3.3) we focus on the representation of context, a key element to succeed in the WSD task. We review the major state-of-the-art contributions from the emergence of the word embeddings (Bengio et al., 2003; Mikolov et al., 2013b) to the recurrent neural networks (Elman, 1990; Hochreiter and Schmidhuber, 1997a) and finish by presenting language-model-based architectures (Peters et al., 2018; Devlin et al., 2018).

Finally, the remaining of the chapter (Section 3.4) is dedicated to the disambiguation methods which can be divided into two main branches: the knowledge-based methods, relying on external semantic resources, and the supervised methods based on sense annotated data. This will be a good opportunity to present the most recent state-of-the-art WSD systems.

3.2 Data

We can distinguish two main kinds of resources for the WSD task: on the one hand, lexical resources which offer the sense inventory from which sense labels are extracted, e.g electronic dictionaries, ontologies, semantic networks etc. On the other hand, sense annotated corpora used both as training data for supervised systems and as evaluation data for testing. The success of the task depends on the availability and the good quality of these two types of resources.

Now, these resources exist for English, in particular Semcor (Miller et al., 1993), a manually sense annotated corpus with senses from the lexical database Wordnet (Miller, 1995). For other languages on the other hand, such high-quality resources are very rare or even non-existent. This is mainly due to the fact that both the manual construction of semantic resources and the annotation of corpora with senses are expensive and very difficult to obtain on a large scale. As a consequence, given these difficulties, researchers turned to automatic approaches to build multilingual lexical databases and large-scale multilingual disambiguated texts.

This section aims at presenting the major resources used by state-of-the-art systems. First, we focus on the main English resources, that is Wordnet and Semcor. We also introduce OMSTI (Taghipour and Ng, 2015b), an English, automatically built corpus containing one million sense annotations.

Then, in a second part dedicated to multilingual resources, we present Babelnet (Navigli and Ponzetto, 2012), a vast multilingual semantic network, and Eurosense (Bovi et al., 2017), a multilingual corpus automatically annotated with Babelnet's senses.

Finally, we describe the available WSD evaluation datasets used to compare and evaluate WSD systems.

3.2.1 English Resources

For decades, most of the WSD research has been focusing on the English language because of the availability of high quality resources. Indeed, on the one hand, the construction of Wordnet (Miller et al., 1990), a computationally efficient lexical knowledge database based on psycholinguistic theories, has benefited knowledge-based WSD methods. On

the other hand, the good quality and relative large size of SemCor (Miller et al., 1993), a corpus manually annotated with Wordnet's senses, allowed the development of supervised methods which have proven to obtain best results on the WSD task. Consequently, these two resources have been most widely used and are still the *de facto* reference resources for the English WSD task today.

In this section we first give a brief presentation of Wordnet. Then, we introduce three sense disambiguated corpora used by state-of-the-art-systems. In order to compare these corpora we sum up several statistics in Table 3.3.

3.2.1.1 Wordnet

Wordnet (Miller et al., 1990) is an English online lexical database which groups up verbs, nouns, adjectives and adverbs into sets of synonyms (called *synsets*) representing underlying meaning concepts. The resource is structured as a hierarchical semantic network in which synsets may be linked together through semantic and lexical relations such as antonymy, hyponymy, hyperonymy etc. Each synset is provided with a gloss definition and one or several examples. In Figure 3.1 we illutrate an example of an entry found in Wordnet (the latest version 3.0) for the verb *help*. For the sake of readibility we only show two of the many synsets (*help.v.01* and *help.v.02* in the example) it is related to. If we take a look at the synset *help.v.01*, it gathers multiple word forms including (but not exhaustively) *aid*, *assist* and *help*. Word forms belonging to the same synset are considered synonyms. In the figure, the synset is linked to other synsets *support.v.1* and *care.v.1* through relations of hyperonymy and hyponymy respectively.

Statistics Wordnet is still today the most important manual resource of its kind for English. It collects more than 150,000 word entries for a little less than 120,000 synsets showing thus a good lexical coverage that approximates that of standard dictionaries. Table 3.1 [1] details the number of word entries along with the number of synsets per part-of-speech in Wordnet. Statistics about polysemy is given in table 3.2. From a disambiguation perspective, one can notice that verbs, our primary concern, exhibit a higher degree of polysemy than the other part-of-speech whether or not we include monosemous words.

3.2.1.2 Semcor

SemCor (Miller et al., 1993) is the largest manually sense tagged corpus for English. It is composed of sentences from the Brown Corpus (Kucera

[1] We report the statistics provided by Princeton's wordnet official website: `https://wordnet.princeton.edu/documentation/wnstats7wn`

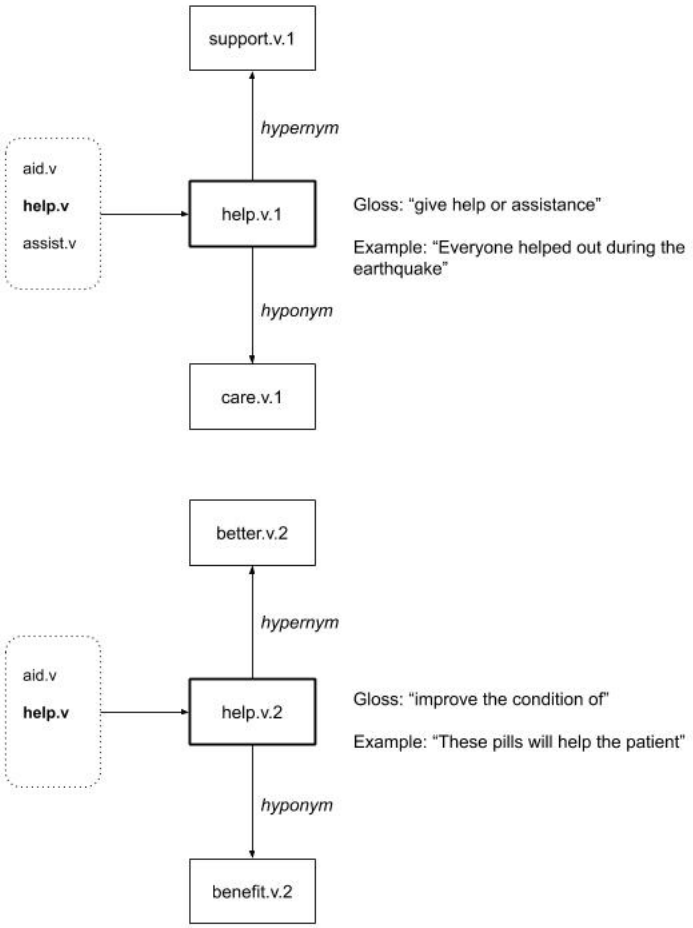

Figure 3.1: Example of an entry for the verb *help* in Wordnet.

pos	word entry	synsets	word-sense pairs
Noun	117798	82115	146312
Verb	11529	13767	25047
Adjective	21479	18156	30002
Adverb	4481	3621	5580
Totals	155287	117659	206941

Table 3.1: Statistics per part-of-speech in Wordnet.

et al., 1967; Francis et al., 1982) in which open-class word forms (i.e verbs, nouns, adjectives and adverbs) were annotated with Wordnet's senses. Because of its size (226,040 sense annotations) and its quality (manual annotations), Semcor has imposed itself as the standard reference training corpus for English WSD.

POS	Average Polysemy	
	including monosemous	excluding monosemous
Noun	1.24	2.79
Verb	2.17	3.57
Adjective	1.40	2.71
Adverb	1.25	2.50

Table 3.2: Average Polysemy per part-of-speech in Wordnet.

3.2.1.3 Princeton Annotated Gloss Corpus

As we've seen, Wordnet offers definitions and examples to describe the senses present in the resource. Since the word forms that compose the definitions, called "glosses", are manually linked to the appropriate context sense in Wordnet, they can thus be considered as disambiguated text. The Princeton Annotated Gloss Corpus (also often referred to as the WordNet Gloss Corpus (WNGT)) is a corpus, available since the 3.0 version of Wordnet, composed of the synset definitions found in Wordnet and in which word forms were manually or semi-automatically sense annotated with Wordnet senses. WNGT has 117,659 disambiguated glosses which represents 496,776 annotations. Although the corpus is of specific genre, as it is only composed of definition-like sentences, it has proven to boost up supervised trained WSD systems when combined with other sense annotated data such as SemCor (Vial et al., 2019; Bevilacqua and Navigli, 2019) .

3.2.1.4 OMSTI

Since manually sense annotating words is a fastidious task, researchers looked for various ways to automatically or semi-automatically produce sense annotated data with sufficient quality (Diab, 2004; Kübler and Zhekova, 2009; Zhong and Ng, 2009). One Million Sense-Tagged Instances (denoted OMSTI) (Taghipour and Ng, 2015a) is a corpus semi-automatically sense annotated with Wordnet's sense inventory. The corpus was built using a WSD method based on word alignment (Chan and Ng, 2005) and applied to a large English-Chinese parallel corpus (Eisele and Chen, 2010). Experiments with WSD supervised systems trained on OMSTI have shown competitive results on several WSD benchmarks despite the fact that the resource has been constructed semi-automatically (Taghipour and Ng, 2015a; Iacobacci et al., 2016).

Corpus	Tokens	Sentences	Annotations	Sense types	Word types
SemCor	802,443	37,176	226,036	33,362	22,436
WNGT	1,621,129	117,659	458,825	59,250	55,561
OMSTI	813,798	30,441,386	911,134	3,730	1,149

Table 3.3: Statistics of the three major sense annotated corpus for English WSD.

3.2.2 Multilingual Resources

Semantic resources like Wordnet and Semcor are very profitable for the WSD task. Nevertheless, they are extremely costly to produce and similar resources for other languages are very scarce or simply do not exist. There were various attempts to create manual multilingual lexical databases following Wordnet's design (Vossen, 1998; Pianta et al., 2002; Atserias et al., 2004) but the cost of building such resources (which has to be repeated for each language) is high and makes it difficult to be manually maintained in the long term. Besides, they often offer a poorer coverage. These difficulties urged researchers to use automatic techniques to obtain these resources. In this section we present two multilingual WSD resources built with such methods: BabelNet and EuroSense. The former is a vast multilingual semantic networks based on Wikipedia and Wordnet, and the latter is a collection of parallel texts automatically sense annotated with BabelNet's sense inventory.

3.2.2.1 BabelNet

BabelNet (Navigli and Ponzetto, 2012) is a very large multilingual semantic network, built automatically, which combines multiple resources such as Wikipedia and Wordnet into sets of concepts called babel synsets. The original version of BabelNet only mapped Wikipedia pages to Wordnet synsets and the mapping was essentially based on the comparison of context information retrieved from both the Wikipedia pages (sense labels, links, categories) and the Wordnet senses (synonyms, gloss etc.). Once English Wikipages are linked to Wordnet senses, the babel synsets are translated into other languages using the context of inter-language links from the wikipedia pages, sentences from Semcor and a state-of-the-art machine translation systems. We report in Figure 3.2 the illustration of a babel synset from (Navigli and Ponzetto, 2012). Since the first released version of Babelnet, more resources have been added (such as Wiktionary, VerbNet, FrameNet [2]) and BabelNet now accounts for more than 800 million babel synsets.

[2] `https://babelnet.org/about`

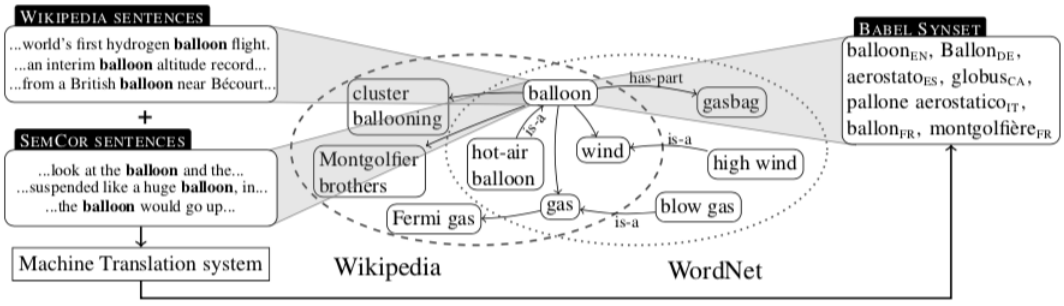

Figure 3.2: Illustration of a Babel synset (Navigli and Ponzetto, 2010)

3.2.2.2 EuroSense

EuroSense (Bovi et al., 2017) is a multilingual corpus automatically sense annotated with Babelnet synsets and based on the joint disambiguation of the parallel corpus EuroParl (Koehn, 2005), a collection of parallel texts from the proceedings of the European Parliament which was initially designed to serve as training data for the automatic machine translation task. Unlike previous works which treated the disambiguation of sentences in isolation (Taghipour and Ng, 2015a), (Bovi et al., 2017) fully took advantage of the parallel texts by performing the disambiguation of the sentences and their translations at the same time. This was made possible thanks to Babelfy (Moro et al., 2014), a graph-based multilingual system which performs both WSD and entity linking (EL) and Nasari (Camacho-Collados et al., 2016), a language-independant vector representation of concepts, to refine the annotations. Two versions of the corpus were released, a full version and a refined version where only annotations with a certain level of confidence were kept based on a coherence score. Overall, the corpus accounts for 21 languages. We report statistics for 4 languages in Table 3.4.

		EN	FR	DE	SP
Full	Annotations	26 455 574	22 214 996	16 888 108	21 486 5
	Lemma types	60 853	30 474	66 762	43 892
	Sense Types	138 115	65301	75 008	74 214
Refined	Annotations	15 441 667	12 955 469	9 165 112	12 193 260
	Lemma Types	42 947	23 603	50 681	31 980
	Sense Types	86 881	49 189	52 425	52 859

Table 3.4: Statistics for (full and refined versions) English, French, German and Spanish in EuroSense.

To measure the quality of the automatic annotations, intrisic and extrinsic evaluations were performed. The intrinsic evaluation was carried out through a manual evaluation of the annotation on four languages

(English, French, German and Spanish). Fifty sentences present in all four languages were randomly sampled and manually evaluated both before and after the refinement step. The results showed a good inter-annotator agreement, as the judges agreed 85% of the time and the average Kappa score (Cohen, 1968) was 67.7 %, and an improvement of the precision of the annotation after refinement at the expense of a lower coverage.

As for the extrinsic evaluation, Bovi et al. (2017) made experiments using EuroSense (the refined version only) as additional training data for supervised WSD systems evaluated on two standard English WSD evaluation datasets (SemEval-2013 task 12 (Navigli et al., 2013) and the SemEval-2015 task 13 (Moro and Navigli, 2015), see next section for more details). The authors compared results of the It Make Sense (IMS) system (Zhong and Ng, 2010) when trained on SemCor alone versus SemCor augmented with examples sampled from the high precision Eurosense corpus (up to 500 additional training examples per sense). They report a slight improvement in the latter case, the Fscore rising from 65.3 to 66.4 on SemEval-2013, and from 69.3 to 69.5 on SemEval-2015.

3.2.3 WSD Evaluation Datasets

The first initiative to evaluate WSD systems was proposed in the late 1990's leading up to the organization of a dedicated workshop called Senseval. At that time, there were many different programs capable of disambiguating words but no means to make a fair comparison between them, to analyse their strengths and weaknesses. The Senseval competition was created to address this need. It proposed a WSD framework in which participants were all given the same resources and were expected to evaluate their system on the task. Since then, with the success of the first Senseval workshop, several following challenges have been organized and each time the number of target languages has grown as well as the number of participants, showing the interest for the task. Over the years, other semantic tasks (such as lexical substitution, semantic role labeling, sentiment analysis) were integrated in the evaluation campaigns, and the Senseval competiton, which was originally WSD specific, has evolved to become Semeval, a more generic semantic evaluation competition.

These evaluation campaigns led to the construction of multiple manually sense annotated WSD datasets, each one with different features. We give a brief description of these datasets in the following paragraphs:

Senseval2 (Edmonds and Cotton, 2001). It was the second international WSD evaluation campaign and the first time that Wordnet was used as sense inventory. Three other languages than English were submitted for evaluation: Czech, Dutch and Estonian. It contains sense

annotations for nouns, verbs, adjectives and adverbs.

Senseval-3 task 1 (Snyder and Palmer, 2004). This dataset contains sense annotations of sentences extracted from the Penn Treebank (Marcus et al., 1993) and with specific genres: editorial, fiction and news.

SemEval-07 task 17 (Pradhan et al., 2007). This is the smallest dataset of the challenges. Sentences were extracted from the Wall Street Journal and the Brown Corpus and annotated with Wordnet 2.1.

SemEval-13 task 12 (Navigli et al., 2013). This dataset was built for multilingual WSD evaluation purposes. It uses BabelNet as sense inventory and the texts cover different domains from sports to financial news.

SemEval-15 task 13 This is the most recent WSD evaluation exercise. The goal of this evaluation was to promote the research towards multilingual disambiguation jointly with the entity linking task on a specific domain (biomedical).

Even if these datasets were all constructed to evaluate systems on the same task, they suffer from heterogeneity especially in terms of format, construction guidelines and sense inventories. For example the Senseval-3 task 1 and SemEval-07 task 7 datasets both share the common sense inventory of Wordnet but with different versions, respectively 1.7 and 2.1. Raganato et al. (2017b) have proposed a unified WSD framework gathering all the datasets presented above into a single standarized one. They use a pipeline which ensures that all the datasets are annotated with the latest version of Wordnet (3.0) and undergo the same pre-processing. Besides the framework also includes SemCor and OMSTI as training corpora. Not only does this framework make the comparisons between systems more reliable, but it also allows a direct quantitative confrontation since all systems can now be evaluated on all datasets at once.

We have seen in this section the main existing resources used in the WSD task. We will now turn to the presentation of state-of-the-art methods and start in the next section by the introduction of the context representation methods.

3.3 Context representations

As mentioned in the introduction, the meaning of a word depends on the context of the sentence it occurs in. Firth, particularly known for his work in distributional semantics, illustrated this phenomena with a

famous quotation: "You shall know a word by the company it keeps". Since the context of a word is crucial to determine its meaning, its representation should be of prime concern. In fact, Weaver (1955) noticed the importance of the representation of the context early on and since the first works in WSD, the algorithms have highly relied on cues found in the context of an occurrence to perform its disambiguation. Over the last two decades, we have witnessed several major breakthroughs in terms of word and sentence representations.

In this section, we propose to gradually review state-of-the-art context representation methods from the simplest to the most sophisticated ones. Our starting point is the introduction of word embeddings (Bengio et al., 2003; Mikolov et al., 2013a) (Section 3.3.1) which efficiently encode words on dense vectors while preserving some semantic features of words. While word embeddings can be directly combined to build context representations, they have furthermore greatly participated in the rise of neural networks in NLP as they provide a useful encoding of symbols to be used as input of the networks.

In Section 3.3.2, we take a step further towards context representation and present recurrent neural networks (RNN). Unlike word embeddings which give the same representations of words independently from the context, the RNNs are able to provide different representations of words given the context they occur in, which can be viewed as representations of full sequences.

Finally, we dedicate the last part of this section to the description of ELMO (Peters et al., 2018) and BERT (Devlin et al., 2018), two recent state-of-the-art models based on the language model objective, trained on large amounts of data in an unsupervised manner and fine-tuned on NLP downstream tasks. These models output contextual representations which have shown impressive results in several NLP benchmarks, including WSD.

3.3.1 Word Embeddings

Word embeddings are distributed representations of words in a real-value vector space. The key idea is to enable word similarity modeling in a vector space, instead of considering atomic word symbols. Ideally, these representations should reflect the natural semantic features of words. Bengio et al. (2003) were the first to propose a neural network for the language model task that provided word vector representations as a by-product. Yet, word embeddings were widely popularized later by Mikolov et al. (2013a)'s Word2vec methods which allowed to learn these representations more efficiently on large corpora. Word2vec consists in two different neural net models, namely Skip-gram and CBOW, which both aim at learning word embeddings but with different structural objective functions. The training objective of the Skip-gram model is to

predict the words of the context of a target word in a sentence while the CBOW model does the exact opposite, i.e to predict the target word given the words occurring in its context. We report the illustrations of the two models from (Mikolov et al., 2013b) in Figure (3.3). Yet, neural

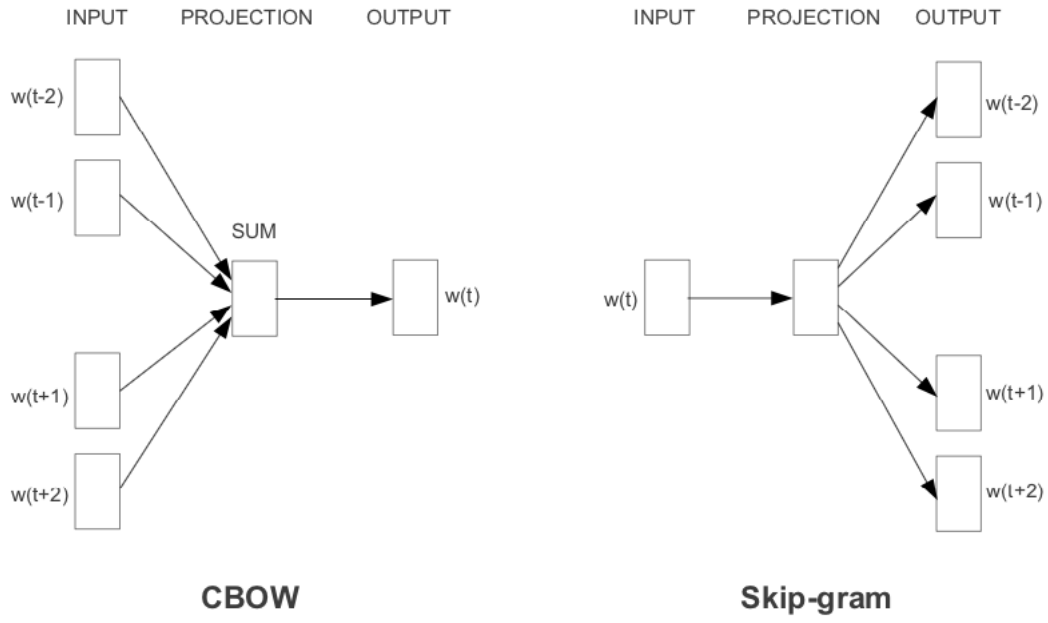

Figure 3.3: Mikolov's Skip-gram and CBOW models objectives.

networks outputing a probability distribution over a large vocabulary face computational challenges. The efficiency of the word2vec's methods comes from the introduction of a variant of the Negative Contrastive Estimation (Gutmann and Hyvärinen, 2012; Mnih and Teh, 2012), called Negative Sampling, as objective function to train the model. Instead of predicting a word from the whole vocabulary, the model is trained to distinguish true examples from artificially built noise using logistic regression resulting in a considerable speed up of the training process. The word embeddings learnt with the word2vec methods have proven to exhibit interesting semantic and syntactic word relationships (Mikolov et al., 2013b) and have become *de facto* standard word representations in NLP.

3.3.1.1 Context Representation Through Word Embedding Linear Combination

As for the WSD task, pre-trained word embeddings can be either considered as inputs for more sophisticated context representation models such as RNN (see Section 3.3.2), used as features for WSD classifiers or they can be directly combined to build context representations. Iacobacci et al. (2016) investigated different methods to combine the word embeddings of the words occurring in the context window of a target word

(see Figure 3.4). A context window of size n for a target word at position i in a sentence is defined as the n-words on the left and the n-words on the right found in the immediate surroundings of the target word. In the examples below we give an illustration of such context window of size 3 (marked with brackets) for two target words (underlined) in context. Words colored in red (resp. blue) are part of the immediate left (resp. right) context. Notice that the target word is not included in the context window. Besides, the context window depends on the sentence boundaries which can result in asymmetrical left and right context size. For example, looking at examples (7) one can observe that, within the context window, there are three words on the left side of the target word and only two on its right.

(3) [The little boy plays an instrument loudly] enough to shake the walls .

(4) In the end we shall make thoughtcrime literally impossible, because there will be no words [in which to express it .][3]

Among the various methods explored by Iacobacci et al. (2016), a simple and intuitive method to build a context representation of a target word consists in averaging the embeddings of the words found in its context window. Formally, this means computing the centroïd of the word vector representations (Eq. 3.1).

$$e_i = \sum_{j=i-W}^{i+W} \frac{w_j}{2W} \tag{3.1}$$

Where w_j is the vector of the j^{th} word in the sentence. I is the position in the sentence of the target word and W is the size of the context window. This is referred to as Average Word Embeddings (AWE) and is often used as baseline due to its simplicity (Melamud et al., 2016).

Another approach explored by Iacobacci et al. (2016) relies on decay functions. The main idea is to weight the importance of a word vector representation based on its distance from the target word in the sentence. The closer a word in a context window is to the target word, the more importance it should have on the context representation. The Exponential decay strategy is based on that principle and weights the words of close context exponentially. It is defined as follows:

$$e_i = \sum_{j=I-W}^{I+W} w_{ij}(1-\alpha)^{|I-j|-1} \tag{3.2}$$

[3]Quotation extracted from *1984*, Georges Orwell

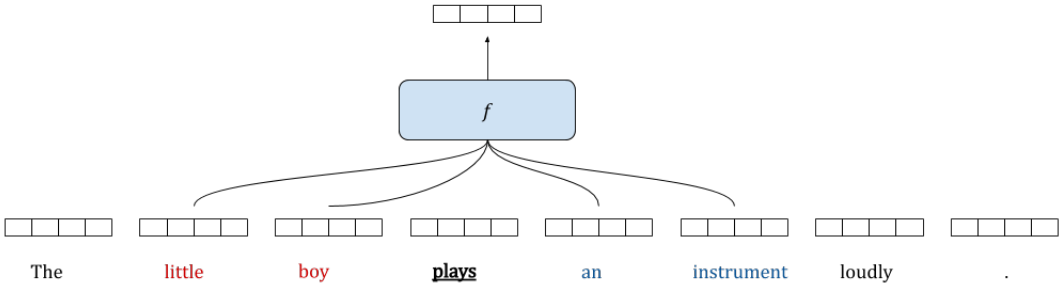

Figure 3.4: Building context representation using word embeddings. The context representation of the target word "plays" is obtained by combining the vector representations of the words in its context window of size 2 using a composition function f.

Where $\alpha = 1 - 0.1^{(W-1)^{-1}}$ is a decay parameter which controls to which extend close words in the context window contribute to the context representation. Among all composition functions presented in (Iacobacci et al., 2016), the strategies based on decay functions have proven to be the most efficient on the WSD task.

While building context representation through composition of word embeddings is of relative computational simplicity, Iacobacci et al. (2016) have shown that they obtain quite good results in the WSD task. Nevertheless, one of the main flaws of this method is that word order in the context representation is completely ignored. We present in the next section recurrent models whose main benefit is precisely to encode sequences.

3.3.2 Recurrent Neural Net Encoder

In this section we introduce recurrent neural network encoders designed to encode sequential information. We first present the original version of the RNN proposed by Elman (1990) and then describe two variants of the RNN: the LSTM (Hochreiter and Schmidhuber, 1997a) and the bi-directionnal LSTM.

In the last paragraph we present Context2vec (Melamud et al., 2016), a model derived from a bi-directionnal LSTM which learns word and contextual representations from a large corpus.

Recurrent Neural Net A Recurrent Neural Network (RNN) encoder is a neural net architecture capable of providing a fixed size vector representation from an arbitrary length sequential input, i.g a sequence of words in a sentence. In opposition to word embeddings which encode the input symbols independently, RNNs are able to encode the order of the sequence. The neural network initializes internal state vectors (called hidden states) and updates them while processing the input iteratively over the sequence. The input to an RNN is a sequence of vectors $v_1, v_2, v_3 \cdots v_n$. e.g static word embeddings. The simplest version of an RNN, originally proposed by Elman (1990), is composed of a simple hidden state vector h initialized at the begining of the sequence and updated recursively at each time step t of the sequence following the update rule :

$$\mathbf{h}_t = g\left(\mathbf{U} \cdot \mathbf{h}_{t-1} + \mathbf{W} \cdot \mathbf{v}_t + \mathbf{b}\right) \tag{3.3}$$

Where W, U (weight matrices) and b (a bias) are parameters to learn through backpropagation and g is a non-linear activation function. The matrices W and U make connections respectively for the input to the hidden state at timestep t and for the hidden state at time $t-1$ to the hidden state at time t. An illustration of such recursive cell is proposed in Figure 3.5. The output of the neural net are the hidden states at each

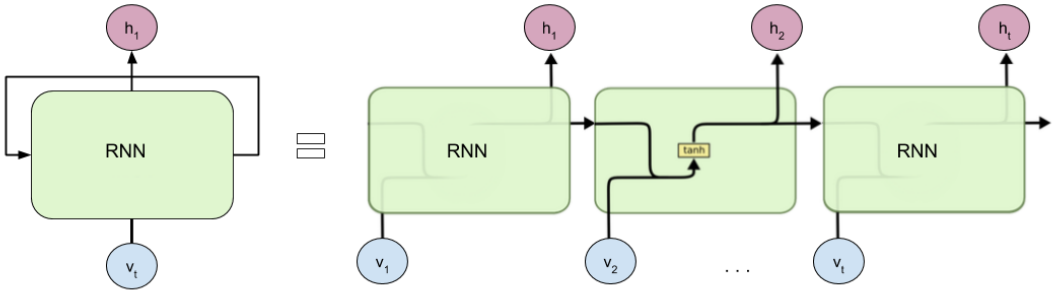

Figure 3.5: Illustration of the recursion of a RNN.[4]

time step t. Therefore, given a sequence of length n, the hidden state h_n output at time step n encodes the representation of the whole sequence. This early version of an RNN is efficient to encode small sequences. However, the training of such neural net becomes more difficult over long sequences as the repeated multiplications during backpropagation make the gradient vector either decay or explode exponentially. This problem is known as the vanishing or exploding gradient (Hochreiter, 1998; Bengio et al., 1994) and prevents simple RNNs from learning long-distance dependencies.

Long-Short Term Memory Hochreiter and Schmidhuber (1997a) proposed a more sophisticated version of Elman (1990)'s network called

[4]The illustration was adapted from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

Long-Short Term Memory (LSTM) to address the problem of the vanishing gradient. The LSTM is based on the original RNN architecture but introduces an additional structure called a *memory cell* whose aim is to maintain the internal state for longer periods of time. The general idea of the LSTM unit is to use the memory cell as a way to let the information flow through the iterations. The amount of information at each time step to be let through is regulated by four neural layers called *gates* composed of activation functions and point-wise operations. Following (Tai et al., 2015a) we present the recursive definition of an LSTM as follows:

$$
\begin{aligned}
i_t &= \sigma\left(W^{(i)}v_t + U^{(i)}h_{t-1} + b^{(i)}\right) \\
f_t &= \sigma\left(W^{(f)}v_t + U^{(f)}h_{t-1} + b^{(f)}\right) \\
o_t &= \sigma\left(W^{(o)}v_t + U^{(o)}h_{t-1} + b^{(o)}\right) \\
u_t &= \tanh\left(W^{(u)}v_t + U^{(u)}h_{t-1} + b^{(u)}\right) \\
c_t &= i_t \odot u_t + f_t \odot c_{t-1} \\
h_t &= o_t \odot \tanh\left(c_t\right)
\end{aligned}
\tag{3.4}
$$

Where c is the memory cell and i,f, o, u are the interactive gates. Each of the gates has a specific purpose: i, the input gate, first selects which values from the input vector should be let through while u computes new candidate values from the input using the tangent hyperbolic (*tanh*) activation function. The point-wise multiplication of i and u allows to control how much of the new information from the input should be stored in the cell state. The forget gate f selects the information to be left out from the previous state h_{t-1}. Finally o, the output gate, regulates the exposure of the memory cell. The illustration of the LSTM recursion is presented below in Figure 3.6. The LSTM unit is a powerful

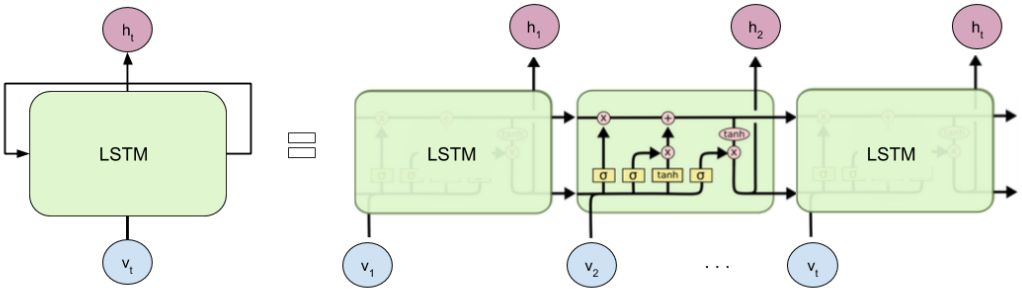

Figure 3.6: Illustration of the recursion of a LSTM.[5]

tool to encode sequence information. However, it can only encode information that has been seen prior to the current position. In the next paragraph, we present the bi-directional LSTM, a variant of the LSTM which addresses this problem.

[5]The illustration was adapted from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

Bidirectional-LSTM A Bidirectional LSTM (bi-LSTM) is an architecture made of two LSTM units which computes the sequential inputs in parallel (but with separated parameters): one in the natural order of the sequence and the other in the exact reversed order. The output of the bi-LSTM is the concatenation of the hidden layers of the two LSTM units at timestep t. Let us see an example with the encoding of a sentence. Given a sequence of words w_i and their corresponding vector representations v_i where i is the position of the word in the sequence. Then, given a bi-LSTM composed of two LSTMs, the first one (denoted lLS) runs the input sequence $v_1, v_2, v_3 \cdots v_n$ from left-to-right and the second one from right-to-left (denoted rLS). The representation of the target word at position i is thus defined as :

$$\text{biLS}(i) = h_i^{\text{L}} \oplus h_i^{\text{R}} \tag{3.5}$$

Where h_i^L (resp.^{R}) is the hidden vector at position i output by the lLS (resp. rLS) recurrent unit and $\oplus$ is the concatenation operation. In WSD, the context representations provided by bi-directional LSTMs have proven to be particularly efficient (Raganato et al., 2017a). This is mainly due to their ability to capture information anywhere in a sentence. It is an important feature since the clues for the disambiguation of a target word in a sentence may be found either before or after the target word.

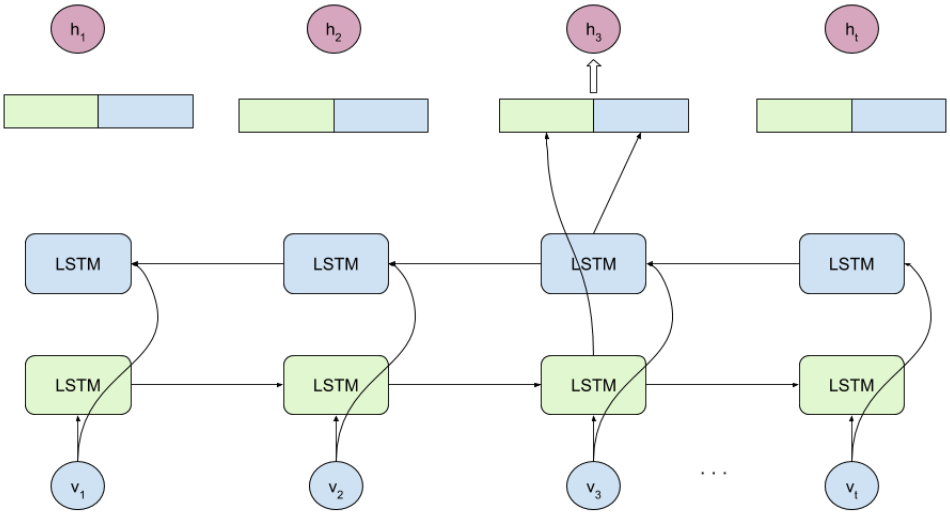

Figure 3.7: Illustration of a bi-LSTM.

Context2vec Context2vec (Melamud et al., 2016) is a neurnal network that learns a context representation function from large unlabeled corpora. The model is based on a bi-LSTM to represent sentential contexts and use word2vec's negative sampling objective function to learn both the word and context vector representations (Figure 3.8). Given a sentence s and a target word w_i where i is the position of the word in s, the

context vector representation $\mathbf{c}$ for w_i is computed in two steps. First the bi-LSTM is run on the input sentence and the outputs of the forward and backward are concatenated in such a way that the target word at position i is not part of the concatanated representation:

$$biLS(i) = h^l_{i-1} \oplus h^r_{i+1} \tag{3.6}$$

Notice that this is different from eq. 3.5 where w_i is included in the representation. Then a *Multi-Layered Perceptron* (MLP) projects the concatenated output into the word embedding vector space:

$$\text{MLP}(\mathbf{x}) = L_2\left(\text{ReLU}\left(L_1(\mathbf{x})\right)\right) \tag{3.7}$$

Where $L_i(\mathbf{x})$ are fully connected linear operations and ReLU is the Rectifier Linear Unit function activation. Thus $\mathbf{c}$ is defined as follows:

$$\mathbf{c} = MLP(biLS(i)) \tag{3.8}$$

The projection of the sentential context output by the Bi-LSTM into the word vector space allows the model to learn its parameters through the negative sampling objective function (Eq. 4.7) adapted from (Mikolov et al., 2013a) .

$$S = \sum_{t,c} \left(\log \sigma(\mathbf{e}(t) \cdot \vec{c}) + \sum_{j=1}^{k} \log \sigma(-\mathbf{e}(t_j) \cdot \vec{c})\right) \tag{3.9}$$

Where $\mathbf{e}(t)$ is the non-contextual word embedding of target word t, t_1 ... t_n are the negative samples and $\mathbf{c}$ is the context vector representation of t in the sentence. The summation goes over all the tokens of the training corpus.

3.3.3 Language-model-based Contextual Representations

Language modeling consists in building a model to estimate the probability of a sentence (or sequence). The seminal work of (Bengio et al., 2003) is the first neural language model, in which a neural network models the probability of words given a context. Since then, the primary use of language models has shifted towards obtaining vector representations of the contexts and has eventually led to interest into and massive use of the transfer learning paradigm. Indeed, the task of predicting a word given a context does not suffer from data scarseness since ”labeled” examples can be trivially extracted from tokenized texts. Thus, a network can be learned on very large corpora with a context-based word prediction objective, and its parameters are therefore good representations of this context. It appears that these contextual representations are very good starting points for representing the context of other language-related tasks. The pre-trained network can be used as the first block of a network for a downstream task, and the pre-trained context representations can be fine-tuned for that task. In this section we will present two

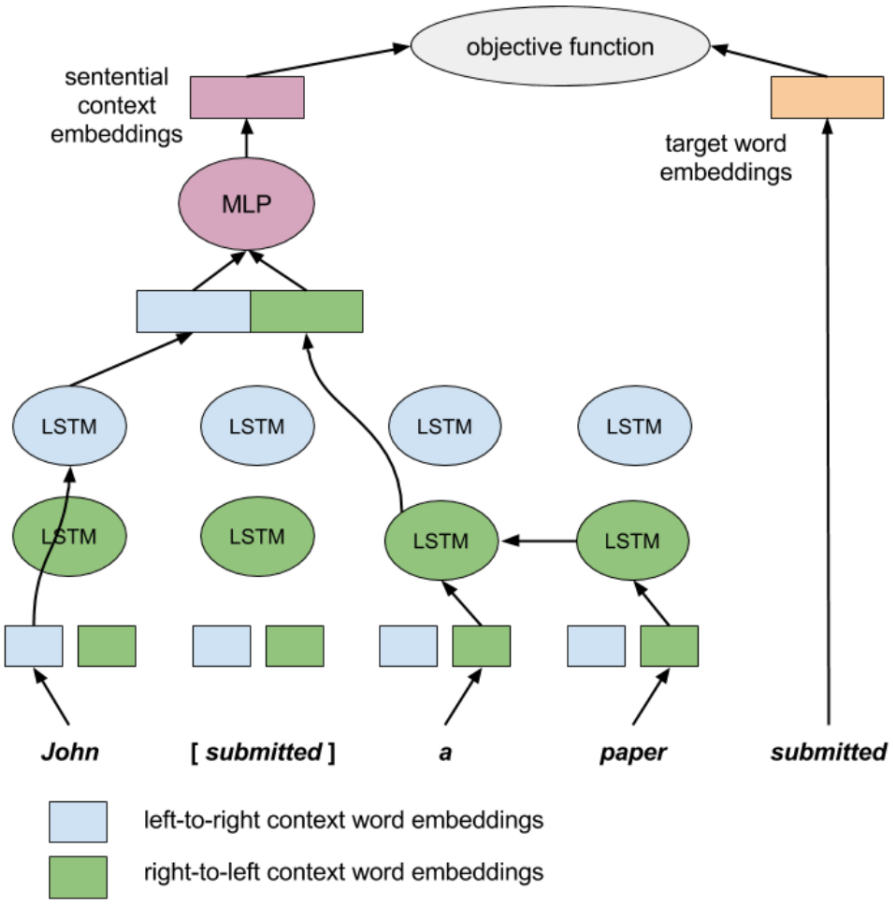

Figure 3.8: Illustration of the Context2vec model architecture. (Melamud et al., 2016)

popular models which fully take advantage of that paradigm: ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018). The former uses bi-LSTMs while the latter relies on more recent attention-based transformers architectures (Vaswani et al., 2017).

3.3.3.1 ELMo

Embeddings from language models (ELMo) (Peters et al., 2018) are word vector representations which are functions of the whole input sentence. The ELMo model provides context vector representations of words according to the sentence they occur in. To learn these representations, Peters et al. (2018) proposed an architecture based on a Bi-LSTM pretrained on large text corpora using a language model objective to be then interpolated in downstream NLP tasks. Except for the encoding of input words, based on a character convolution neural network (Kim et al., 2015), and the output token classification layer (Softmax layer), the parameters of the Bi-LSTM are kept separated. What makes ELMo particularly powerful is the fact that, while jointly trained with a supervised NLP tasks, the model learns linear combinations of the internal

states of the bi-LSTM, hence called deep contextual representations. Furthermore, Peters et al. (2018) among others (Dai and Le, 2015; Howard and Ruder, 2018; Radford et al., 2018) proved that pre-training models on large general-domain data and finetuning them on downstream tasks could be extremely beneficial. They pave the way towards the development of transfer learning, a paradigm that has been largely taken advantage of by the very recent and now prevalent transformer-based architectures.

3.3.3.2 BERT

The Bidirectional Encoder Representations from Transformers (BERT) proposed by (Devlin et al., 2018) is a language model based on multiple stacked layers of transformers, an architecture using self-attention mechanism allowing massive parallelization. The model is inherently bi-directional thanks to the use of a cloze task: predicting a masked word within a left and right context, instead of combining two uni-directional objectives (predicting each word given previous ones on the left (resp. on the right)). This is a major difference from the ELMo configuration where the parameters of the bi-LSTM are kept seperated. The BERT model was pre-trained on extremely large corpora (more than 3 billion words) to be finetuned on downstream tasks. In many NLP benchmarks such as GLUE (Wang et al., 2018) or SQUAD (Rajpurkar et al., 2018), the use of the contextualized representations (finetuned or not) from BERT allowed to outperform previous state-of-the-art models by a large margin, exhibiting once more the efficiency of transfer learning. As for WSD, using the BERT contextualised representations as input for WSD models has also shown outstanding results (Du et al., 2019; Huang et al., 2019; Vial et al., 2019; Scarlini et al., 2020). The following paragraphs aim at progressively presenting the transformer (the core unit of BERT) from the original attention mechanism introduced by Bahdanau et al. (2014) to the fully transformer block proposed in Vaswani et al. (2017).

Attention mechanism The attention mechanism was first integrated by (Bahdanau et al., 2014) in RNN encode-decoder models (Cho et al., 2014b; Sutskever et al., 2014) for the automatic machine translation task to cope with the problem of long sentences. Indeed, even LSTM have difficulties maintaining the information flow over very long sentences (Cho et al., 2014a). To address this problem, Bahdanau et al. (2014) proposed an extension (now called attention) of the RNN encoder-decoder which builds a context vector at each time step t of the sequence using all the hidden states output by the RNN. The underlying intuition of the attention function was to help the decoder generate the next translated word by relying on the most probable aligned word in the source sentence. To this end, the model builds a contextualised vector representation $\mathbf{c}_t$ of

the sequence at time t based on the weighted sum of the hidden states h_i of the RNN encoder, i.e the vector representation of the positions in the input sequence. The weights represent association scores between the previous state s_{t-1} of the model and the hidden states h_i of the encoder. Formally, the context vector $\mathbf{c}_t$ is computed as follows:

$$\mathbf{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} h_j \tag{3.10}$$

Where T_x is the length of the input sequence and α_{tj} are the weights at time step t for each position j in the sequence. The α_{tj} are probabilities resulting from the application of the $softmax$ function over the attention scores $a_{t,j}$ for each vector h_j. Bahdanau et al. (2014) proposed to use a feed forward neural net as attention function :

$$a_{t,j} = A\left(s_{t-1}, \mathbf{h}_j\right) = \mathbf{v}^T \tanh\left(\mathbf{W}\begin{bmatrix} \mathbf{s}_{t-1} \\ \mathbf{h}_j \end{bmatrix} + \mathbf{b}\right) \tag{3.11}$$

Where $\mathbf{v}$ and $\mathbf{W}$ are parameters to learn. While the attention mechanism was originally applied in machine translation, its formulation has been generalized and has been applied successfully in many NLP tasks. For example, in WSD (Raganato et al., 2017a) implemented a Bi-LSTM with an attention mechanism and outperformed most previous state-of-the art WSD models at the time.

Self-attention Self-attention is a specific case of attention introduced in the machine translation field and popularized by (Vaswani et al., 2017) as the main part of the transformer architecture. The aim of the self-attention model is to compute contextualized vectors $c_1, c_2 \cdots c_n$ by relating all positions in a single sentence of words $w_1, w_2 \cdots w_n$. Attention can be generalized as a function to weight the association of a query $\mathbf{q}$ to a set of keys $\mathbf{k}$ and values $\mathbf{v}$ pairs. In self-attention, the vector of the target word w_t is the query $\mathbf{q}$ and all the words $w_{t'}$ (including w_t) and their corresponding vector representations are used both as keys $\mathbf{k_t}$ and values $\mathbf{v_t}$. In opposition with the attention mechanism used in machine translation such as presented previously, in self-attention all the queries, keys and values come from the same input. In practice, the queries, keys and values can be packed together into matrices denoted respectively $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$. Using (Vaswani et al., 2017)'s *scale dot product attention* function, one can define the attention in a more compact notation:

$$\text{Attention}\left(\mathbf{Q}, \mathbf{K}, \mathbf{V}\right) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{3.12}$$

Where d_k is the dimension of the queries, keys and values vectors. The scaling factor $\sqrt{d_k}$ is essentially used to prevent gradient vanishing during training. Using Eq. 3.12, we can compute in a single step the ensemble of the context vectors $c_1, c_2 \cdots c_n$ (packed into a single matrix $\mathbf{C}$)

corresponding to the target words $w_1, w_2 \cdots w_n$:

$$\mathbf{C} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{3.13}$$

Multihead attention The multihead attention (Vaswani et al., 2017) is an ensemble version of the attention mechanism defined above. The idea is to use h additional parameter matrices, called heads, to project linearly the queries, keys and values into separated d-dimensional vector spaces respectively d_q, d_k and d_v. Then instead of performing a single attention function, we use the projected values to operate multiple attention functions in parallel. Thus, the output of each $head_i$ is defined as:

$$\text{head}_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{3.14}$$

Finally the output from all the $head_i$ is concatenated and then projected one last time using a fusion matrix W_O. The multihead attention is therefore described as:

$$\text{MultiHeadAttention}\ (Q, K, V) = \text{Concat}\,(\ \text{head}\ _1, \ldots,\ \text{head}\ _h)\, W^O \tag{3.15}$$

The use of multiple heads while performing attention allows the model to attend information from different subspace vector representations since each head's parameters are initialized independently.

Transformer The transformer architecture (Vaswani et al., 2017) is a model whose purpose is to map a sequence of word embeddings to a sequence of contextualized (transformed) embeddings. It is composed of two sub layers (Fig. 3.9) : a multi-head attention and a fully connected feed forward neural network (i.e a MLP as described in Eq. 3.7). Besides, each sub layer uses residual connections (He et al., 2016) followed by a layer normalization (Ba et al., 2016). The layer normalization prevents exploding or vanishing gradients and allows the model to stack multiple layers on top of each other. Let $\mathbf{c}$ be a contextualized vector output by a multi-head attention function then its normalization $c_{(z)}$ can be described as follows:

$$\mathbf{c}_{(z)} = \text{LayerNorm}(\mathbf{c}) = \gamma\frac{\mathbf{c} - \mu}{\sigma} + \mathbf{b} \tag{3.16}$$

Where μ and σ are respectively the mean and standard deviation values in c. The scaling factor γ and $\mathbf{b}$ are parameters to learn. Now given $\mathbf{X}$, a sequence of vectors compacted into a matrix, the transformer model is defined as:

$$\begin{aligned}
\text{Transformer}(\mathbf{X}) &= \text{LayerNorm}\left(\mathbf{F} + \mathbf{C}_{(z)}\right) \\
\mathbf{F} &= \text{MLP}\left(\mathbf{C}_{(z)}\right) \\
\mathbf{C}_{(z)} &= \text{LayerNorm}(\mathbf{C} + \mathbf{X}) \\
\mathbf{C} &= \text{MultiHeadAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})
\end{aligned} \tag{3.17}$$

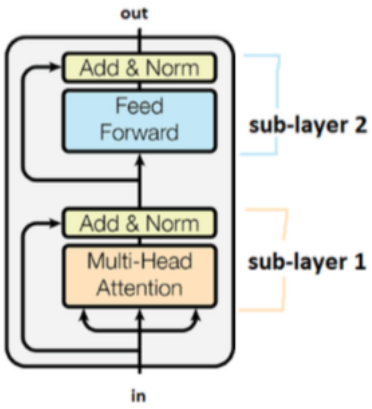

Figure 3.9: Illustration of Transformer block. The block is made of two sub-layers: a multihead attention and a feed forward neural network. Both sub-layers use a residual connection and layer normalization.

3.4 WSD methods

3.4.1 Introduction

Among the various methods proposed to resolve the task of WSD, we can distinguish roughly two main branches: on the one hand the methods based on external lexical resources named knowledge-based methods and on the other hand the supervised methods which highly rely on sense annotated data. In the first part of this section we briefly mention the most representative knowledge-based methods (Section 3.4.2). Then, we focus on supervised methods by first introducing the Knn-classifier and its application to WSD (Section 3.4.3.1). Finally we present classifiers following supervised learning approaches (Section 3.4.3.2): a svm-based classifier and neural network based classifiers .

3.4.2 Knowledge-based methods

The Knowledge-based (KB) WSD systems perform disambiguation by taking advantage of external manually-curated lexical database. As opposed to supervised methods, these techniques are independent from sense tagged corpora. KB methods are not recent, Lesk (1986) developed an algorithm, the Lesk's algorithm, which performs word sense disambiguation using the context of the words and dictionaries. Very simply the algorithm compares the overlap between the words from the context of the target word and the words from its definitions found in a dictionary. Since then, several works have proposed more sophisticated versions of the Lesk's algorithm (Banerjee, 2002; Basile et al., 2012; Chen et al., 2014). In a parallel branch, researchers proposed other KB methods making use of semantic graphs provided by lexical resources. These methods benefit from the structural particularities of the graphs and ap-

ply graph-based algorithms (Agirre and Soroa, 2009; Agirre et al., 2010; Ponzetto and Navigli, 2010; Guo and Diab, 2010; Agirre et al., 2014). Babely (Moro et al., 2014) is an example of such a method. The system makes use of the multilingual semantic network BabelNet and performs jointly the tasks of disambiguation and entity linking thanks to graph algorithms such as Random Walk with Restart (Tong et al., 2006). Babelfy was in particular used to produce Eurosense (see Section 3.2.2).

Very recently, Scarlini et al. (2020) proposed an approach to unify the expressive power of the pre-trained language model BERT and the vast knowledge contained in the semantic network BabelNet. Their method achieved state-of-the-art results or even outperformed most of recent supervised neural methods in the English WSD task. However, the experiments were carried out on nouns only.

3.4.3 Supervised Methods

3.4.3.1 Knn-based classifier

The *k*-nearest neighbors (Knn) algorithm is a supervised method used to solve both regression and classification problems. The algorithm is based on the assumption that similar objects should be close in a vector space. From a WSD perspective, this means that the occurrences of a given lemma sharing the same sense label should have similar vector representations. In a generic classification setting, the algorithm labels a test instance with the majority class of its *k*-closest neighbors from the training dataset where $k > 0$ is a hyperparameter. Since WSD is a classication task, the algorithm is fitted to address this problem and processes as follows: First it fetches every instance of the target word in the training dataset. Then, the Knn classifier tags the test instance with the sense of the training example whose context vector representation is the closest based on vector similarity (i.g cosine similarity or euclidean distances). This is the simplest implementation of the Knn algorithm with $k = 1$ also referred to as a *One*-nn classifier (Melamud et al., 2016; Scarlini et al., 2020).

A variant of this classifier consists in comparing the test instance to sense representations, also called sense embeddings. For example, Yuan et al. (2016) computed sense vector representations by averaging the context vector of the training examples belonging to the same sense label. A more sophisticated method consists in using clustering techniques to gather sense annotated data into sense clusters (Van de Cruys and Apidianaki, 2011; Erk and Padó, 2008). Representing the meaning through vector representations has been the subject of many works and further details can be found in Camacho-Collados and Pilehvar (2018)'s survey dedicated to sense embeddings. Once the sense representations are computed, the prediction then follows the same principle but instead of comparing the test instance to each of its related training examples,

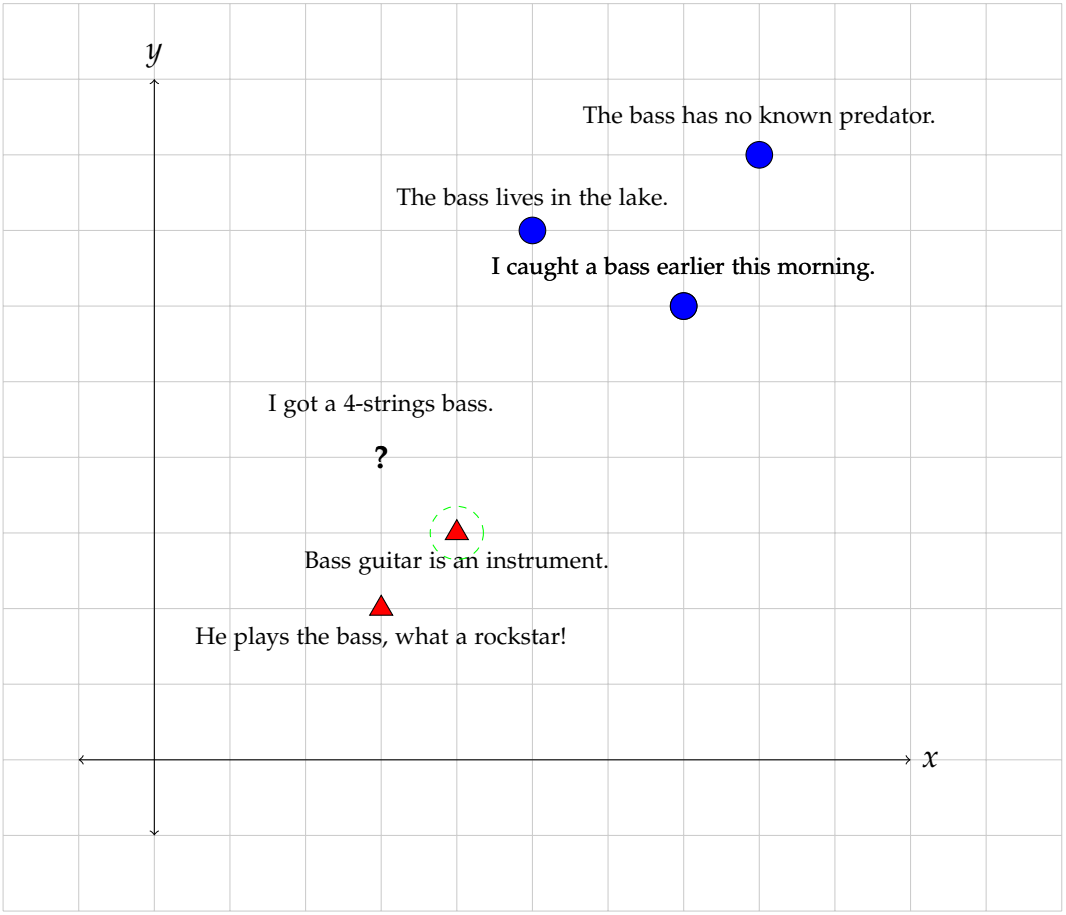

Figure 3.10: Illustration of the Knn classifier applied to WSD. In this examples, the points are vector representations of the occurrences of the words *bass*. The instance to tag is displayed with a question mark and the training examples are labeled with two senses: s_1 the music instrument (red triangle) and s_2 the fish (blue circle). The dotted green circle shows the closest training instance to the target instance and thus indicates that it will be tagged with s_1.

it is compared to the sense representations and is finally sense tagged accordingly.

3.4.3.2 Supervised Learning methods

The WSD systems based on supervised learning methods train a classifier on manually sense annotated data. Over the years they have proven to achieve the best result on the task (Raganato et al., 2017b) but at the expense of a strong dependency towards the availability and good quality of annotated data. Now, we can identify two categories of WSD models: those that are lemma-specific, i.e each lemma has its own classifier, and those handling all lemmas at once using a single classifier. The former approach has the advantage of making the disambiguation easier for the classifier since it searches through a restricted number of classes. This

is particularly fitted for highly polysemous words such as verbs where finer distinction between senses is required. The second method on the other hand shares the parameters among all lemmas which allows a better generalization over the data and offers some interesting properties. First, the models is able to predict the same sense for two different words which could be seen as a way to induce synonymy. Then, since the classification is not based on particular words, it is possible to tag words which have not been seen during the training or even words which are not part of the sense inventory (one may think of neologisms, spelling mistakes etc..) (Vial et al., 2019).

We propose in the next paragraphs to describe some of the most popular state-of-the-art models.

SVM-based classifiers Zhong and Ng (2010) proposed a supervised framework for WSD called "It Makes Sense" (IMS) which integrates different modules for pre-processing, feature extractions and classifiers. The original implementation of IMS was based on a linear support vector machines (SVM) as classifier and used three types of features as input: the PoS tags of the surrounding words, the surrounding words themselves and local collocations. The framework was designed to be very flexible so that anyone can integrate their own modules into the framework from the pre-processing to the machine learning classifiers. For example, Iacobacci et al. (2016) integrated word embeddings as input features into the framework and use the linear classifier to perform WSD. Although the linear classifier has been recently outperformed by transformer-based models, it has still proven to perform well in WSD, especially when combined with word embeddings, despite its simplicity.

Neural-net-based classifier Exploiting neural networks into WSD supervised architectures has become a standard to achieve state-of-the-art results (Raganato et al., 2017a). The typical configuration consists in adding a linear classifier on top of a neural network and train the whole model in a supervised manner: for each target word of a given sense annotated corpus, the model outputs a probability distribution over the senses through a softmax activation function and predicts the sense whose probability is the highest. During the training process, the learning of the model parameters is done by minimizing the cross-entropy loss between the true sense label and the distribution of probabilities, iterating over all the examples of the training data.

Bi-LSTM based architectures K\aagebäck and Salomonsson (2016) were the first to train a bi-LSTM specifically on the WSD task. Their model consists in three components, a word embedding layer to encode the input words (they used the Glove pre-trained word embeddings (Pennington et al., 2014)) which is used as input to feed a second layer made

of a bi-LSTM. The last layer is a linear classifier which, given the output of the bi-LSTM, outputs probabilities over senses.

Another approach proposed by Raganato et al. (2017a) considered the WSD classification task as a sequence learning problem. Instead of making distinct predictions on target words independently, they choose to handle the classification at the sentence level and use a single model to cope with all the predictions. To this end, the authors proposed three bi-LSTM-based WSD taggers. The first model is the simpler version of the three and is very similar to (K\aagebäck and Salomonsson, 2016). There are two main differences: first, instead of using a single bi-LSTM layer, they use multiple stacked bi-LSTM layers. Secondly, the model uses a single classifier to make predictions over all words of the training data (i.e labeled and unlabeled words). The second model is the same as the first one but enhanced with attention mechanism such as described in (Bahdanau et al., 2014). The last model is based on a encoder-decoder architecture inspired by (Sutskever et al., 2014), adapted for sequence-to-sequence WSD and where the encoder and decoder are both composed of bi-LSTMs. The three models were trained in a multitask learning set up (Caruana, 1997) with two auxiliary tasks, POS tagging and Coarse-grained semantic labeling. The results of the experiments showed an advantage to the use of the bi-LSTM with attention mechanism which indicates (1) the usefulness of attention mechanism in WSD and (2) that seq-to-seq architectures may be sub-optimal for the WSD task.

Transformer-based architectures The most recent state-of-the-art models include the use of transformers in the WSD architectures. Du et al. (2019) proposed to finetune the pre-trained BERT model on the WSD task. To this end, they use BERT as an encoder to obtain contextualized representations of the target words and use these representations as input for a 2-layer MLP classifier which second layer is lemma-specific.

Vial et al. (2017) pushed the use of transformers even further by building an architecture similar to (Raganato et al., 2017a) with two key differences: they replace the word embedding layer by the BERT encoder and instead of using a bi-LSTM layer, the replace it with a stacked transformers layer. Overall the transformer-based architectures are at the time of this writing the new state-of-the-art for the WSD task.

3.5 Conclusion

In this chapter we have presented a state-of-the-art of the Word Sense Disambiguation task. We have first introduced the data used by state-of-the-art method to perform WSD. While for decades most of the resources were only available for English, the construction of BabelNet, a large multilingual semantic network, made possible new approaches

to build multilingual sense annotated data such as EuroSense. We will see in chapter 5 that while these resources seem promising at first, we eventually haven't found it suitable in our work on French VSD.

In the meantime, the rapid development of neural nets and especially the shift towards the transfer learning paradigm pushed further and further the limits of the representations of context, a key element to a successful disambiguation. In particular, the recent transformer-based architectures such as BERT have obtained outstanding results on the task. Nevertheless, the success of these models remains unclear due to their high complexity and there is still much to investigate to understand what these models actually learn.

Chapter 4

Syntax for Verb Sense Disambiguation

Contents

4.1 Introduction

In this chapter we address the question of the role of syntax for verb sense disambiguation, starting from the traditional hypothesis that syntax is a key element to the disambiguation of words.

Former VSD sytems took advantage of the argument structure of verbs to perform their disambiguation (Dligach and Palmer, 2008; Roberts and Kordoni, 2012; Kawahara and Palmer, 2014). In practice, it consisted in representing the context of verbs by extracting features from the words within the sentence which were considered as their syntactic arguments. This is based on the hypothesis that the syntactic behaviour of verbs is strongly correlated to their senses.

Then, with the arrival of more sophisticated neural networks, first the RNNs and now the transformers, context representations based on the argument structure of verbs have been left aside in favor of new representations capable of encoding a broader context. In fact, the task of VSD itself took a backseat as most WSD systems context representations all take the same form, independently from the parts-of-speech of the target word to disambiguate. Nevertheless, as powerful as these new representations are, they still remain quite unintelligible and it is very unclear what kind of features are indeed encoded.

In this chapter, we put syntax back on the table and investigate its impact on VSD through three angles: in Section 4.3, we analyse the correlation between argument structure and verb sense through a corpus study. In Section 4.4, we focus on attention-based neural models and study whether they do encode argument structure of verbs. Finally, in Section 4.5, we propose a model to learn contextual representations from syntactic trees and assess its use for WSD.

Before detailing these three sections, we start by presenting the data (Section 4.2) on which our experiments were performed.

4.2 Data

To study the role of syntax for the disambiguation of verbs it is necessary to have at one's disposal data that is both annotated with senses and syntactic features. Since we will focus on the argument structure of verbs, we chose to use dependency tree representations, from which head-argument relations are easy to extract. Besides, in order to make viable experiments, these annotations must be in sufficient quantity and with the highest possible quality. To our knowledge, such corpus that gathers manual annotations for these semantic and syntactic features does not exist, most likely because of the expensiveness of manual annotations. Therefore, we decided to use an existing manually sense annotated corpus, namely SemCor (Miller et al., 1993), and parsed it with a state-of-the-art dependency parser.

In this section, we first present the corpus and the selection of the data. We give some statistics regarding the distribution of verbs and senses in the extracted dataset and further describe a method to balance the distribution of senses. Finally, we provide syntactic details on the

parsing of the data.

4.2.1 Corpus

As mentioned in Chapter 3, SemCor (Miller et al., 1993) is the most important manually sense annotated corpus in use, it has imposed itself as standard resource for the WSD task on English. Hence, it was naturally the best candidate as support data for our experiments. With more than 200 000 manual annotations, the corpus fills both the quantitative and qualitative requirements for the experiments. Moreover, the corpus was included in (Raganato et al., 2017b) 's WSD framework, making it very handy to use and process.

Data selection Since our objective is to study the role of the argument structure of verbs for their disambiguation, we only kept the verb lemmas having at least two distinct annotated senses in corpus, the monosemous ones being irrelevant for the experiments. Yet, compared with nouns or adjectives, verbs are mostly polysemous and the filtering of the monosemics only removed around 10% of the annotations. We present in Table 4.1 some statistics about the filtered dataset.

Corpus	# Annotations	# Target Verbs	Mean number of examples	
			per sense	per lemma
SemCor	80109	1859	12.6	43.0

Table 4.1: Several statistics on the polysemous verbs in SemCor.

Verbs distribution We represented in Figure 4.1 the distribution of the number of verb occurrences. As one can see, the distribution is relatively heterogeneous, all verbs do not have quite the same number of annotated occurrences. For example, most of the verbs have between 10 and 50 occurrences while very few (roughly 50 of them) occur more than 200 times. Therefore, depending on the number of examples, the difficulty of the task may vary across verbs.

Sense distribution Let us have a look at the distribution of the senses in the data. First, we are interested in the distribution of senses among verbs. This is shown in Figure 4.2 and Figure 4.3. The former displays the number of verbs per number of senses while the latter reports the number of occurrences of verbs per number of senses. As we can see, the vast majority of verbs roughly have between two and four different senses and are also the most frequent in the data[1]. The figures also reveal

[1]The over representation of verbs with 11 senses is due to the verb "be" which for obvious reasons is very frequent in the data.

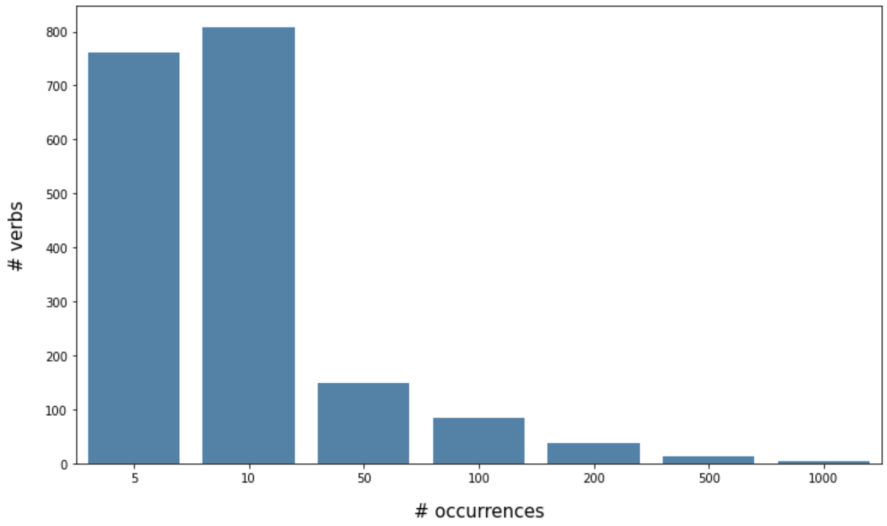

Figure 4.1: Distribution of polysemous verbs frequencies in SemCor. Each bin gathers verbs which have at most n occurrences. For example, the second bin on the left shows that there are roughly 800 verb lemmas that occur at least 10 times.

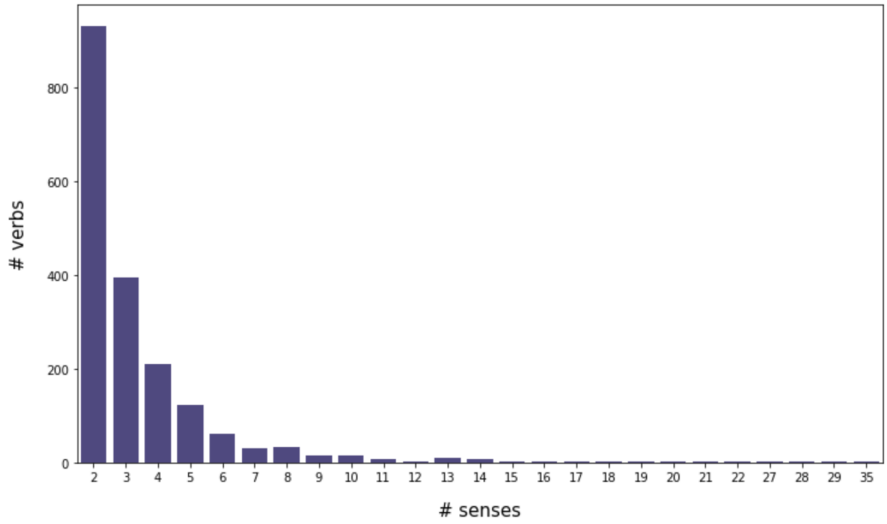

Figure 4.2: Distribution of verbs per number of senses in SemCor. For instance, more than 800 verbs have only 2 different senses as shown by the very first bar on the left. At the extreme opposite, only few verbs have 35 senses.

the fine level of granularity that can be found in Wordnet. Indeed, some verbs can have up to 35 senses. Nevertheless, as shown in Figure 4.3, the most polysemous verbs very scarcely occur in the data which makes them even harder to disambiguate. This reinforces the previous claim that the difficulty of the disambiguation is highly heterogeneous among verbs.

Finally, if we have a closer look at how senses are distributed among occurrences per lemma (Figure 4.4), the most striking observation is the

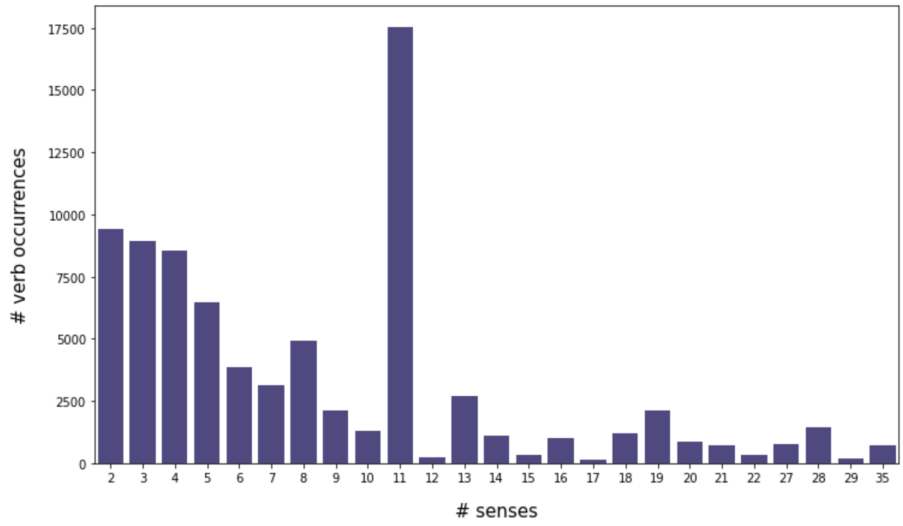

Figure 4.3: Distribution of verbs occurrences per number of senses. For example, there are roughly 5,000 occurrences whose verbs have 8 distinct senses (fourth bar from the left).

consistent prevalence of a sense over the others, no matter what the degree of polysemy is. This actually reflects a well known phenomenon, which has led to a very strong baseline dubbed "most frequent sense", consisting in simply systematically tagging a verb occurrence with its most frequent sense.

Balancing As shown in the previous paragraph, the distribution of senses shows a certain level of heterogeneity and highlights the predominance of the most frequent sense pattern. Observations made using the natural sense distribution of the data will mix syntactic and frequency effects. In order to better pinpoint the disambiguating power of syntactic features, we will systematically provide observations both on the original corpus, with the "natural" sense distribution, and on a "balanced" corpus, in which the senses per verbs are equally distributed.

The balancing was performed using a method described in Algorithm 1. The main idea is to calculate a target number N of instances per sense (for a given lemma) and then to over/sub-sample the occurrences of the senses to fit that number. The oversampling was performed by randomly duplicating existing instances of the target sense. As for the subsampling we did quite the opposite and removed random instances of the target sense from the dataset.

In our experiments we selected N for a given verb as the integer portion of the division of the total number of its instances by the number of its senses. Note that due to the selection of N, which is based on a floor division, the number of annotations may very slightly differ between the original and the balanced distribution of the dataset. In the remaining of this chapter, we will refer to the original dataset as $\text{Semcor}_{\text{Natural}}$ and

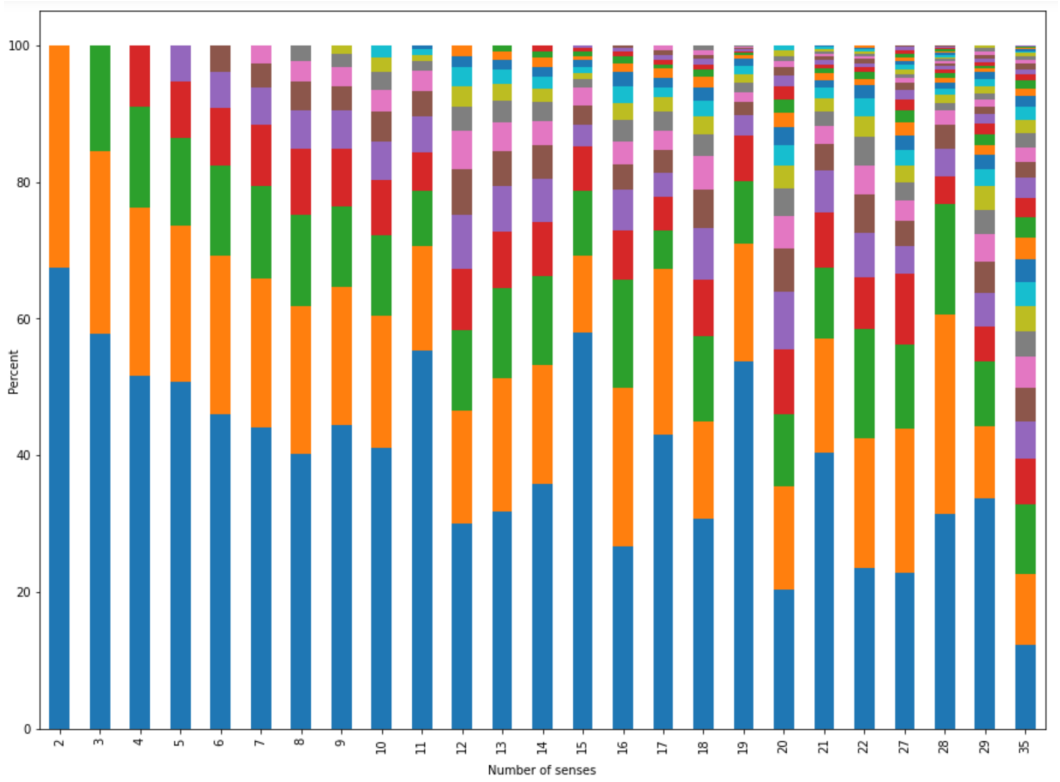

Figure 4.4: Distribution of senses per polysemy degree in SemCor. We have gathered in groups the verbs by their number of senses (x-axis) and averaged then normalized the distribution of senses within groups (y-axis) where each color represents a sense. For example, among the verbs having two senses annotated in the corpus, the average proportion of their most frequent sense is approx. 70.

to its balanced version as $\text{Semcor}_{\text{Balanced}}$.

4.2.2 Syntactic details

Parsing The SemCor corpus contains sense annotations only and was not initially provided with syntactic annotations. We therefore applied a state-of-the-art dependency parser to obtain dependency trees. We used the en_core_web_lg[2] model from SpaCy[3], an industrial API providing state-of-art and easy to deploy NLP tools. We provide an example of an output parse in Figure 4.5.

Argument functions Spacy's English models are trained on dependency trees converted from constituency trees, with labels from the NLP4J project[4]. From this particular set of labels, we introspectively selected a restricted list of syntactic functions that may correspond to

[2] https://spacy.io/models/en#en_core_web_lg
[3] https://spacy.io/
[4] https://emorynlp.github.io/nlp4j/

Algorithm 1: Sense Balancing

```
Result: balanced data
foreach lemma L do
    N = integer portion of (total number of instances of L /
     number of senses s of L) ;
    foreach sense s ∈ L do
        n = number of instances of s ;
        if n < N then
            oversample(s)
        else if n > N then
            subsample(s)
        else
            continue
        end
    end
end
```

syntactic arguments of verbs. We present a concise description of these syntactic functions in Table 4.2.

It is worth mentioning that we have included the "prep" label although the parser does not reliably distinguish between adjunct and argumental prepositional phrases. Furthermore, we bypassed the preposition, considering direct arcs from verbs to the object of the preposition, and concatenating the preposition to the label name. Hence for instance in Figure 4.5, the two arcs $stole \xrightarrow{prep} at \xrightarrow{pobj} school$ would be collapsed into a single $stole \xrightarrow{prep\_at} school$ arc. The result of such modification is illustrated in Figure 4.6.

Function	Description
nsubj	Nominal subject
csubj	Clausal subject
dobj	Direct Object
ccomp	Clausal complement
attr	Attribute
acomp	Adjectival complement
xcomp	Open clausal complement
expl	Expletive
dative	Dative
prt	Verb particle
prep_X	Prepositional modifier or argument

Table 4.2: List of syntactic functions potentially corresponding to syntactic arguments.

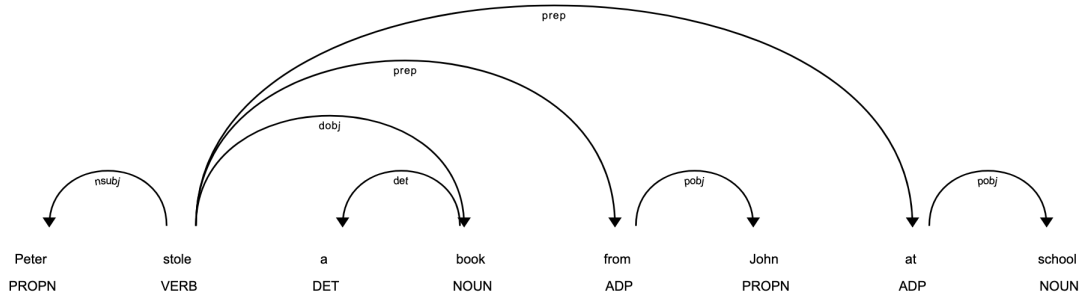

Figure 4.5: Example of dependency parsing obtained with the en_core_web_lg from SpaCy.

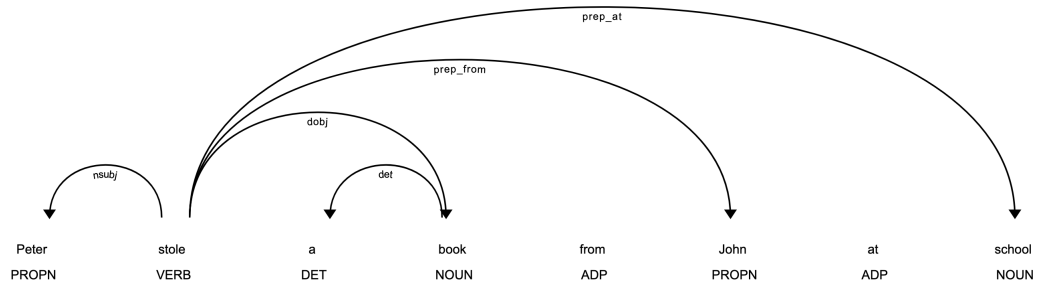

Figure 4.6: Example of the modified dependency parsing obtained with the bypassing of the prepositions.

4.3 Does argument structure discriminate verb senses?

In this section, we try to quantify to which extent the argument structure of verbs allows their sense disambiguation. Our intuition let us think that, to some degree at least, some verb senses can be potentially disambiguated based on their syntactic realization. This idea has inspired traditional work on verb sense disambiguation (Dligach and Palmer, 2008; Roberts and Kordoni, 2012; Kawahara and Palmer, 2014). Here is a simple illustration with two occurrences of the verb "run" exhibiting two different meanings and syntactic configurations. In (5) the verb is intransitive while (6) the argument structure exhibits a subject and an object.

(5) Everyday Sarah **runs** at 8 o'clock.

(6) They **ran** twenty blood tests on me and they still don't know what's wrong.

Of course this is not systematic, there are pairs of examples where two different senses share a common syntactic structure. For example, let us take another example with the verb *run*:

(7) I **ran** the whole race without being short of breath.

Now, comparing (6) and (7) one can easily observe that it is the lexicon rather than the syntax which allows to interpret the different meaning of the verb *run*.

While these examples were drawn from pure introspection, we now provide a study of the sense discrimination achievable when representing verbs through their argument structure.

To this end, we first designed two vector representations to encode the argument structure of a verb in context. The first one is based on syntax only while the second one integrates syntactic and lexical features. Both versions are described in Section 4.3.1.

We then use these argument structure representations for two investigations: in Section 4.3.2 we measure to which extent a (dis)similarity of the argument structure of two verb occurrences correlates with the (dis)similarity of their annotated sense. In Section 4.3.3, we further investigate the performance of purely supervised VSD classification when the sole input representation of the verb is its argument structure.

4.3.1 Argument structure vector representations

We propose two different ways to encode the argument structure of an occurrence of a verb by means of real valued vectors. The first one relies on syntactic features only while the second one combines syntactic and lexical information into the same vector representation. In what follows, we will refer to these vector representations of the argument structure as ”syntax-based argument vector” and ”lexical-based argument vector” (although the latter is also syntactic).

Syntax-based argument vector The vector is built using binary values (1 or 0) which stands for the fulfillment (or lack thereof) of an argument of the verb in context. It is of size n, the number of possible syntactic functions as verb arguments, and where each dimension corresponds to a given function. An example of such representations is given in Figure 4.8

Lexical-based argument vector The lexical-based argument vector (which is actually lexico-syntactic) makes use of pre-trained word embeddings. It is constructed the same way as the syntax-based argument vector, but instead of binary values, we use the word embeddings of the syntactic head words of the arguments. More precisely, if a verb has a certain dependent d labeled with an argumental label l, then the portion of input for label l is the word embedding of d, otherwise it is the null vector. Its size is therefore $n \times d$ where d is the size of the word embeddings. Injecting word embeddings into the argument vector representation allows to combine both syntactic and lexical information in the same represen-

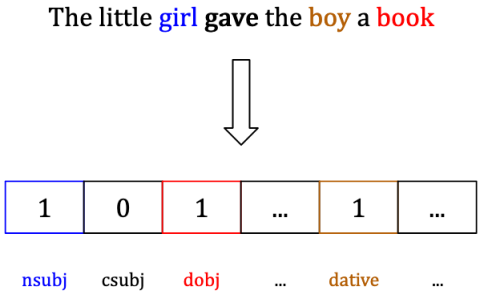

Figure 4.7: Illustration of the syntax-based argument vector. For the "gave" instance, only *nsubj*, *dobj* and *dative* are present in the argument structure. Therefore, the values corresponding to these syntactic functions are set to 1 and all the others to 0.

tation. An illustration of the lexical-based argument vector is shown in Figure 4.8.

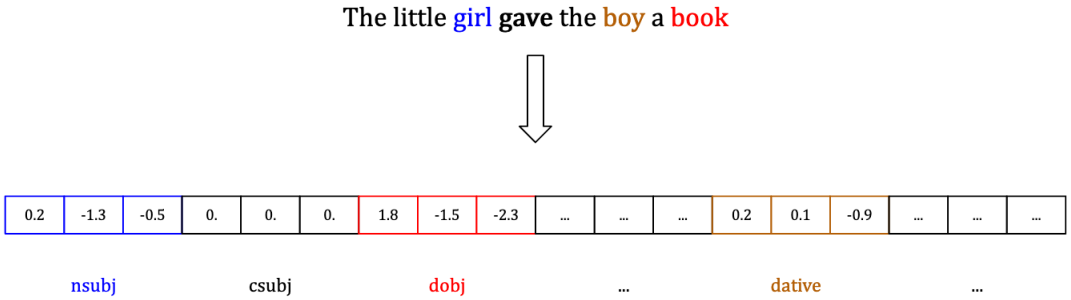

Figure 4.8: Illustration of the lexical-based argument vector. The vector is composed of n slots dedicated to the n argument functions. Each slot is filled with a word vector corresponding to the lexicalisation of the argument function. In this example, the slots specific to the *nsubj*/*dobj* and *dative* syntactic functions contains the word vectors for "girl","boy" and "book" respectively. All other slots (that is unfullfilled argument functions) are set with zero vectors.

4.3.2 Corpus study: correlation between argument structure similarity and sense annotation

Now that we have defined a method to encode the argument structure, we propose two experiments based on vector similarity whose aim is to measure to what extent these representations can discriminate verb senses.

The first experiment, referred to as "inter-sense test" measures, given a verb lemma, how dissimilar pairs of senses are from the point of view of argument structure. The more dissimilar pairs of senses are, the more the argument structure is actually a discriminant feature for verb senses.

The second experiment denoted "intra-sense test" measures the quality of the clusters provided by the annotation of verbs' senses. More precisely, it measures to what extent the set of occurrences of a verb sense is more homogeneous with respect to argument structure representation than the set of all occurrences of that verb.

Both experiments are described in the next two sections.

4.3.2.1 Notations

In the following we denote an occurrence of verb as o, its lemma as $lemma(o)$, and its annotated sense as $sense(o)$. We denote the set of annotated senses of a lemma l as $senses(l)$.

We will note v_o for the argument structure vector of o (either syntax-based or lexical-based). By extension, we will note v_{xxx} the average vector of all the occurrences matching xxx, so v_s is the average vector of all occurrences annotated with sense s, and v_l is the average vector of all the occurrences of the lemma l.

4.3.2.2 Inter-Sense Test

This test will allow us to measure the similarity of the senses of the same lemma, through the prism of the argument vectors. Our hypothesis for this test is that the more argument vectors of distinct senses are dissimilar, the more syntactic information actually helps to discriminate between senses.

Method For every verb lemma l of our dataset, we calculate the cosine similarity between each pair of senses s_i, s_j of l. The result is a distribution of cosine similarity values for every pair of senses of the same lemma.

In this experiment, we compared the syntax-based and the lexical-based argument vectors on both $\text{Semcor}_{\text{Natural}}$ and $\text{Semcor}_{\text{Balanced}}$ datasets, giving rise to four configurations.

Results The results of the inter-sense test are given in Figure 4.9. They are several histograms representing the similarity distributions in the different configurations.

Our first observation concerns the balancing of the data. As we can see, it doesn't seem to affect the experiment since the figures on the top and bottom rows are pretty much alike showing no significant difference.

Now turning to the analysis of the similarity distributions, the results show two interesting trends. On the one hand, the distribution obtained using the syntax-based argument vectors shows that syntax alone is not sufficient to discriminate verb senses. Indeed, most pairs have a cosine

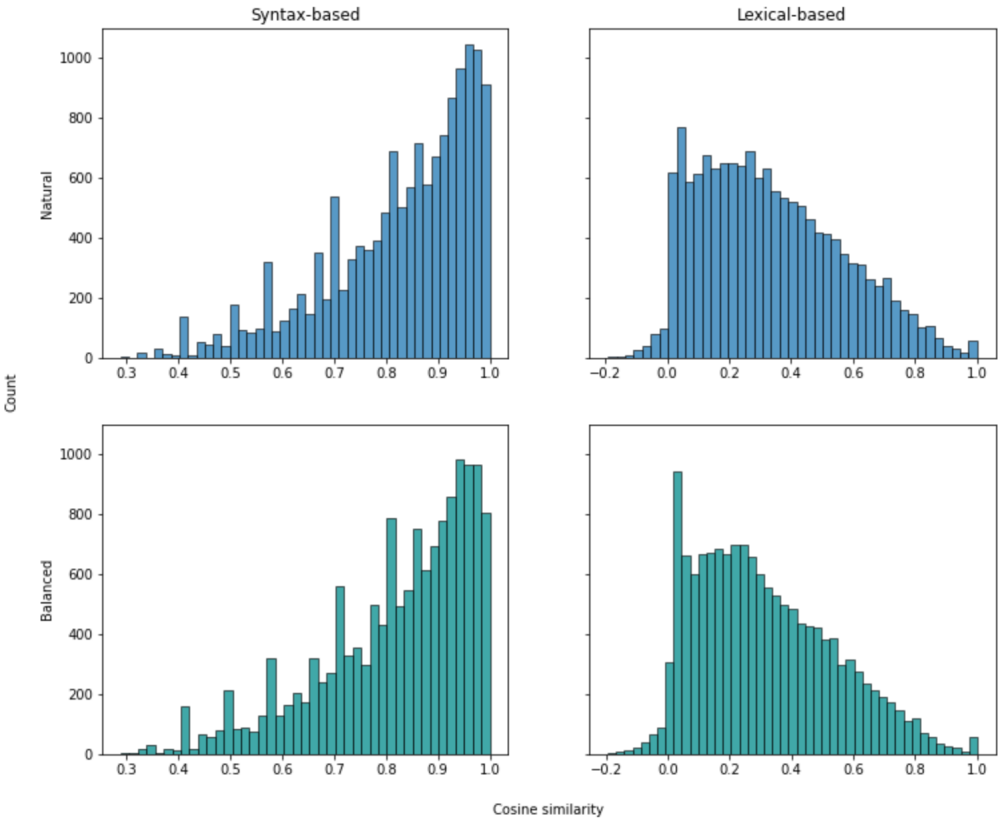

Figure 4.9: Distribution of the cosine similarity values of pairs of senses resulting from the inter-sense experiment. Columns gather the experiments per argument vector types and rows show the datasets on which they were performed. For example, the figure in the top left corner displays the results of the inter-sense experiment on the $\text{Semcor}_{\text{Natural}}$ dataset using the syntax-based argument vector representations .

similarity close to 1 indicating that they have very similar syntax-based argument vectors. On the other hand, the distribution is reversed with the lexical-based argument vector, the similarities greatly decrease, with very few pairs having a cosine close to 1, and a lot more pairs having a cosine below 0.5.[5]. This suggests that the lexical content of arguments does discriminate between verb senses.

4.3.2.3 Intra-Sense Test

This second test broaches in a different way the question of the discrimination of the meanings of verbs by their argument structure. In this experiment we aim at observing the quality of the clustering induced by sense annotations in the corpus with regard to the argument structure of verbs. The main idea is to look at whether the argument vector of an occurrence is closer to its sense vector than to its lemma vector.

[5]Note that in the lexical case, since word embeddings may have negative values, the cosine is valued in [-1,1]. On the other hand, the syntax-based argument vector is made of binary values which thus constraints the cosine values into the 0,1 range.

Argument vector	*Natural*		*Balanced*	
	cos_S	cos_L	cos_S	cos_L
Syntax-based	0.87	0.84	0.89	0.82
Lexical-based	0.47	0.41	0.59	0.43

Table 4.3: Results of the intra-sense experiment on the $\text{Semcor}_{\text{Natural}}$ and $\text{Semcor}_{\text{Balanced}}$ datasets for the syntax-based and lexical-based argument vectors. The score cos_S denotes the average cosine similarities between occurrences and their senses while cos_L is the average similarities between the occurrences and a more global vector based on the lemma occurrences.

Method To perform this test, we compare (i) the average similarity between an occurrence vector and its sense vector, with (ii) the average similarity between an occurrence vector and its lemma vector. More precisely we compute cos_S and cos_L defined as:

$$cos_S = \frac{1}{|corpus|} \sum_{o \in corpus} cos(v_o, v_{sense(o)})$$

$$cos_L = \frac{1}{|corpus|} \sum_{o \in corpus} cos(v_o, v_{lemma(o)})$$

We compare cos_S to cos_L using the syntax-based and the lexical-based arguments vectors, and using the natural and the balanced sets of occurrences, hence leading to four settings.

Results The results are presented in Table 4.3. In all configurations, we can see that cos_S is higher than cos_L, which is the expected result should argument structure be helpful to discriminate between senses. Yet, the difference is much smaller when using the syntax-based argument structure than with the lexical-based representation. **This strongly suggests that differences in pure syntax do not explain much of the sense distinctions**. This supports the results obtained in the previous experiment and highlights once again the importance of the lexical information in the representation of the meaning of verbs. Moreover, the differences between cos_S and cos_L are smaller when using the natural corpus than when using the balanced data. **This suggests that argument structure is less discriminating in the case of frequent senses than in the general case.**

4.3.3 Supervised VSD using argument vectors only

The last two experiments revealed that the verb sense distinctions correlate much better with argument structures representations when these encode not only the grammatical functions of the arguments but also

their lexical semantics. While these experiments were based on corpus statistics, we are now interested in testing the argument vectors as sole input representation in a supervised VSD setup as a way to confront the previous results. To this end, we trained a neural net based classifier for the VSD task using the argument vectors as input features and compared the performances to simpler bag of words representations. In what follows, we first expose the experimental protocol and then present and discuss the results of the experiment.

4.3.3.1 Experimental protocol

Model For this experiment we used, for each lemma, a single linear layer with softmax output. The model takes argument vectors as input and outputs a probability distribution over the senses of the target lemma. The predicted sense is the one with the highest probability.

Data We used the natural and balanced versions of SemCor in our experiments. For each lemma, we made a 75/25 split for the training and validation datasets respectively. Statistics on the datasets are given in Table 4.4.

Distribution	Dataset	# Annotations	Mean number of examples	
			per sense	per lemma
$SemCor_{Natural}$	Train	62893	9.02	21.06
	Dev	20976	5.83	10.92
$SemCor_{Balanced}$	Train	64754	9.28	21.68
	Dev	21986	5.81	11.45

Table 4.4: Statistics on the train/dev splits for the natural and balanced version of SemCor.

Experiments In our experiments, we trained the model using the syntax-based and lexical-based argument vectors as input vectors. We compared our model to the most frequent sense baseline for the experiment on $\text{Semcor}_{\text{Natural}}$ and to a random prediction baseline on $\text{Semcor}_{\text{Balanced}}$. We also compared the argument vector representations to a standard bag-of-word (denoted BOW_{all}) representation where we computed the sum of the word embeddings of all words in the sentence. Furthermore we experimented a specific version of the BOW representation in which we only considered the argument of the verbs denoted BOW_{arg}.

In every experiment implicating word embeddings, we used the GloVe (Pennington et al., 2014) pre-trained word embeddings which we kept fixed during the training. All models were trained for 50 epochs using Adadelta with a learning rate of 0.5. At each epoch we performed an evaluation on the validation dataset.

4.3.4 Results

The results are shown in 4.5. Let us first analyze results in the balanced corpus. The first observation is that lexical-based argument vectors are the best inputs (acc=44.7, a 28 point increase over the random baseline), while the syntax-based argument vectors are the worst (acc=36.8), which supports the observations made in section 4.3.2: purely syntactic argument vectors are much less effective than lexico-syntactic ones. Interestingly enough though, results when using BOW_{all} and BOW_{arg} as input suggest that it is the combination of lexical and syntactic information that is the most effective: there is an almost 5-point drop when switching from lexical-based argument vectors to BOW_{arg} vectors.
The picture is a bit blurred when looking at results on the natural corpus. Unsurprisingly, sense distribution helps the classifiers, hence all results are higher. While the lexical-based vectors still yield the best results, all three other settings do perform better than the most frequent sense baseline. This suggests that the contribution of argument structure to VSD is less crucial, but still useful, when the natural sense distribution is kept.

Vectors	Accuracy	
	Natural	Balanced
Syntax-based	62.3	36.8
Lexical-based	65.5	44.7
BOW_{all}	60.1	39.8
BOW_{arg}	63.1	40.0
MFS	57.0	-
Random	-	27.0

Table 4.5: VSD accuracies on the development split of Semcor.

4.4 Syntax in attention-based models

In this section we continue our investigation on the role of the argument structure of verbs in the VSD task from another angle: we seek the presence of the argument structure encoding in attention-based models.

Attention mechanism, since originally proposed in (Bahdanau et al., 2014), has been successfully integrated in many NLP systems and has set a new state-of-the-art in a wide range of tasks. At the time of writing of this thesis, the best WSD models are based on attention mechanism (Raganato et al., 2017b; Vial et al., 2019). Self-attention, in particular, has been used to build the Transformer (Vaswani et al., 2017), a neural net architecture which is the core of the BERT (Devlin et al., 2018) model, the new NLP standard for sentence and token encoding.

Not only do attention mechanisms appear very efficient, but they also have a very interesting interpretability capacity. By looking at the weights put on the words of the sentence we can have an insight on what words are important to solve the targeted task (Clark et al., 2019; Voita et al., 2019; Htut et al., 2019).

We want to make use of this interpretability to assess the role of the argument structure in verb sense disambiguation by using attention based models on the VSD task. Our hypothesis is that if the argument structure is indeed decisive to disambiguate verbs, then we should expect it to be significantly attended to in the model.

We conducted two experiments to investigate this hypothesis. First, we trained a simple VSD classifier based on a single self-attention layer and studied the output attention weights. Secondly, we used a state-of-the-art WSD model based on the pre-trained Bert (Devlin et al., 2018) model and following previous works, we looked for particular heads that would focus on the argument structure. Both experiments are presented in the next two sections.

4.4.1 Simple attention-based VSD

The aim of this experiment is to see whether a simple self-attention based VSD model can obtain satisfactory results, and if so, to observe the amount of attention paid to the argument structure by the model. We voluntarily seek for the model to be as simple as possible since we are not so much interested in performance as in intelligibility.

Model architecture Our model architecture is composed of three components (illustrated in Figure 4.10):

- An embedding layer : converts the words w_i of a sentence into d-dimensional real value vectors v_i. We used the GloVe (Pennington et al., 2014) pre-trained word embeddings as input word vectors which we kept fixed during the training process.
- An attention layer : a self-attention mechanism as proposed in (Vaswani et al., 2017) which computes contextualized vector $\vec{c}_i$ for each v_i based on the weighted sum of all the v_j. Following (Vaswani et al., 2017) we used three parameters matrices of size $|d| \times |d|$ to project the input v_i into the query, key and value vector spaces. More details on self-attention formulas can be found in Chapter 3-Section 3.3.3.2.
- A lemma-specific fully connected linear layer with softmax activation function to turn the output vector $\vec{c}_i$ into a distribution of probability over the senses of the target lemma. The model predicts the sense which has the highest probability.

Note that, parameter matrices aside, the context representation output by the self-attention mechanism can be seen as a weighted bag of words representation.

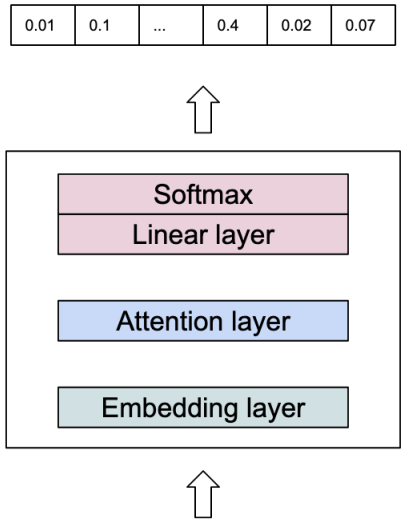

Figure 4.10: Illustration of the architecture of our simple attention-based VSD model.

Experiments The model was trained for 50 epochs using Adadelta as optimizer with a 0.5 learning rate and we kept the parameters of the model that provided the lowest loss on the development set during training. We first compare the level of performance obtained by our attention-based classifier with a simple BOW model.

We then calculated the proportion of the attention weight put on the argument structure (using the dependency tree) as follows: for each instance to disambiguate, we summed the attention weights of syntactic heads belonging to the argument structure and then averaged the overall sum by the total number of instances. We compared this proportion with weights of randomly selected tokens in the sentence following the same process. We also considered the proportion of attention weights put on the whole span of the argument structure instead of the heads alone and similarly compared with random weights accordingly.

Results The performance of the attention-based model is shown in Table 4.6. Firstly, the attention model outperformed the MFS and Random baseline as well as the model based on the BoW representation in all configurations highlighting the efficiency of the self-attention mechanism despite its relative simplicity.

Secondly, we represented the proportion of attention weights on the argument structure in Figure 4.11. The weights on the heads tokens of

Model	Accuracy	
	natural	balanced
Attention	62.7	39.8
BOW	60.1	35.1
MFS	57.0	-
Random	-	27.0

Table 4.6: Performance (accuracy) of the simple attention-based VSD classifiers on both configurations (natural and balanced) of the development dataset.

the argument structure remained relatively low since it only concentrates 12% of the attention. When using the whole span of the argument structure, the proportion of attention is significantly higher reaching nearly 40%. We can observe that the weights of the argument structure compared with randomly selected weights are superior when selecting either heads or spans of arguments. Nevertheless, when we constrained the random selection to tokens that share the same PoS with the tokens of the argument structure, the margin becomes much smaller (shown as the green bars in Figure 4.11). This indicates that the increase in attention proportion for argument heads / spans is actually due to the distribution of POS in argument heads / spans, and not to the argument status.

Considering the fact that the proportion of attention on the heads of the argument structure remains quite low and that the random weight proportion is very close to the one of the argument structure, we can deduce that the model does not particularly attend to the argument structure to succeed in the task. Therefore, the role of the argument structure remained very limited in this configuration.

4.4.2 Syntax in transformers-based VSD models

In the previous section, we initiated our investigation of the role of syntax in attention-based models using a minimal architecture example. We now continue our study further and focus on much more sophisticated attention-based models. In particular, we are interested in the study of BERT(Devlin et al., 2018), a pre-trained language model based on the transformer architecture (Vaswani et al., 2017) which optimally takes advantage of the attention mechanism. BERT has been successfully integrated into WSD models achieving outstanding results(Du et al., 2019; Vial et al., 2019).

Since traditional work has shown the importance of argument structure for VSD (Dligach and Palmer, 2008; Roberts and Kordoni, 2012; Kawahara and Palmer, 2014) , we may wonder whether transformer-

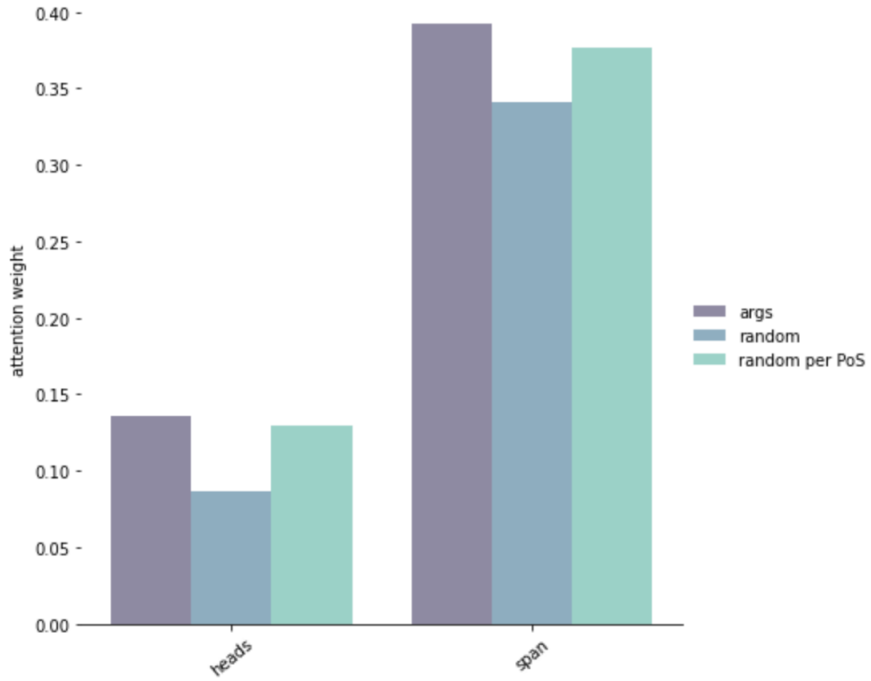

Figure 4.11: Proportion of attention weights on the argument structure per heads and span compared with random words in the sentences. The green bar indicates the proportion of attention for randomly selected tokens which share the PoS found in the argument structure of the verb occurrence.

based pre-trained language models also encode it in some way.

Several results suggested that syntactic features are captured by some attention heads at least (Clark et al., 2019; Voita et al., 2019; Htut et al., 2019). In particular, some of the heads pay maximal attention to specific syntactic relations, some of which correspond to arguments of verbs. Yet, comparing most-attended-to words to syntactic dependents only gives a partial picture: a head paying maximal attention to subjects may also attend to simpler but correlated patterns.

In this section, we first experiment the finetuning of BERT on the VSD task. We then investigate the most-attended-to word classification method (Clark et al., 2019), in the context of verb sense disambiguation data. Finally, as a contribution we show that simultaneously looking at how a given head attends to syntactic, lexical and positional patterns reveals a much more contrasted picture concerning the syntactic abilities of attention heads.

4.4.2.1 Finetuning BERT on the VSD task

BERT is a multi-layer transformer based model trained on the masked language modeling and next sentence prediction tasks. Although the model provides powerful contextual representations from its pre-training, best results on various NLP tasks were achieved by finetuning, that is to

say adding a classifier on top of the pre-trained model and training both the classifier and BERT as a whole model on the task.

In our first experiment we finetuned BERT on the VSD task and compared the results with keeping BERT's parameters fixed. We found that the finetuning did not significantly improve the performance of the classifier. The next paragraphs describe the experimental protocol and the results.

Setup The finetuning was performed using a simple classifier similar to the one described in Section 4.4.1 replacing the two first layers with the BERT model. We trained and evaluated the model on the same data as specified in 4.3.3.

We trained both versions of the model for 50 epochs and used Adadelta as optimizer with learning rates of 0.5 and 0.005 for the pre-trained and finetuned models respectively.

Model	Precision	
	Natural	Balanced
Bert_{frozen}	73.4	58.5
$\text{Bert}_{finetuned}$	73.3	57.3
MFS	57.0	-
Random	-	27.0

Table 4.7: Performances (precision) of the Bert model (finetuned and frozen) on the development dataset in the natural and balanced configurations.

Results The results are presented in Table 4.7. They reveal no significant difference of performance between the pre-trained and the finetuned models on the verb sense disambiguation task. We are not aware of any research in WSD which has studied the impact of finetuning on this task to support this observation.

Our hypothesis is that the disambiguation task is very similar to the masked language model task and it is possible that most of the features may have been already learnt during the pre-training of the model but this remains to be further investigated.

Given these results and for the sake of efficiency, we decided to use the non-finetuned version of BERT for the attention study that follows.

4.4.2.2 Extracting Attention Maps

We ran the model on the data and extracted for each verb occurrence v the self-attention weight vector of each attention head in each layer. In the remaining of this section, we will refer to a particular head as L-H

with L the layer number, and H the head number in L, e.g. 7-1 denotes the first head of the seventh layer.

BERT tokenizes words into tokens (subwords). We follow Clark et al. (2019) in order to recover word-word attention weights from token-token weights: for attention to a split word, we sum the attention weights over its tokens. For attention from a split word, we average the weights over its tokens. This ensures that the attention from a word sums to 1.

4.4.2.3 Maximum Attention Classifier

Before detailing the method, let us first define how we will designate some of the positions in the verb context, hereafter called "**patterns**". For a target verb occurrence v, we denote its absolute position as i and the absolute position of its direct dependents in the parse tree by the name of the dependent's dependency label: for instance *nsubj* denotes the absolute position of the verb's nominal subject.

For relative positions, we write $p \pm k$ to target the position p with a shift of k tokens. Thus *i+1* denotes the position right after the verb, and the *nsubj≠i-1* pattern designates the *nsubj*, for instances in which it is not right before the verb.

Method For each verb instance v and each attention head h, we consider the head's prediction as being the word that received the most attention from v, apart from v itself[6]. Then for each syntactic label l, we define the accuracy of head h for label l as the proportion of verbal instances for which the most-attended-to word equals the dependent[7] labeled l (the proportion being computed among the instances having such a dependent). Finally, we keep the best predicting head $\hat{h}(l)$ for each label l. For instance in Table 4.8, head 8-1 is the best at identifying nominal subjects: for 74.9% of the verb instances having a *nsubj* dependent, the most-attended-to word by head 8-1 is the *nsubj*.

For each label l, we compare the accuracy obtained by $\hat{h}(l)$ with that obtained by a relative position baseline classifier. More precisely, for a relative position $p = i \pm k$, we count among the verb instances having a dependent labeled l, how often that dependent is at position p. We let k vary from 1 to 5 and retain the best relative position accuracy as baseline for label l. For instance, a majority of nominal subjects are at position *i-1*, hence *i-1* is the baseline for the *nsubj* label.

Results As previously observed (e.g. (Kovaleva et al., 2019)), the first striking result when studying BERT's self-attention weights is that in many cases, attention heads put most of their attention to the [CLS] and

[6] We also excluded BERT's [CLS] and [SEP] special tokens to focus on inter-words relations

[7] Since our prime interest is the representation of verbs, we only considered the verb→dependent direction and the 10 labels for syntactic arguments of verbs.

Label	Head	Acc.	Baseline	Δ
expl	3-2	98.1	81.0 (-1)	+21%
acomp	6-7	88.9	61.4 (1)	+45%
prt	2-4	84.5	87.1 (1)	-3%
dative	6-5	77.9	60.2 (1)	+29%
attr	7-5	76.5	37.2 (2)	+105%
nsubj	8-1	74.9	58.2 (-1)	+27%
dobj	7-5	69.9	38.8 (2)	+80%
xcomp	7-9	59.2	60.7 (2)	-2%
csubj	6-4	38.7	14.2 (-4)	+173%
ccomp	7-5	21.8	15.2 (3)	+43%

Table 4.8: For each syntactic label: best head ($h(\hat{l})$) using the maximal attention method, accuracy of the best head, accuracy of the best relative position, and best relative position (within brackets).

[SEP] special tokens used as classification token and sentence separator. Indeed, we calculated that roughly half of the attention heads put more than 30% attention on these tokens.

Turning to the predicted syntactic dependencies, only 25 out of the 144 heads obtained results superior to 20% of the relative position baseline. In other words, most of the attention heads either attend to BERT's special tokens or to the tokens in the close context of the target word, as already observed

The best performing heads for the syntactic dependency prediction task are presented in Table 4.8. Some of those heads beat the relative position baselines by a large margin. In particular, the heads specialized in predicting nominal subjects (nsubj), direct objects (obj), adjectival complements (acomp) and attributes (attr) achieved high performances. These results seem to support the claim that some heads are indeed good at tracking syntactic relations and thus exhibit BERT's ability to capture syntax from self supervision only.

4.4.2.4 Attention study

While the previous experiment informs us about the heads' capacity to capture syntactic behaviors, it doesn't plainly reveal how the heads distribute their attention among all the words of the sentence, and among the different patterns. For instance, Table 4.8 shows that head 7-5 is good at identifying direct objects, but firstly it is also good at identifying *attr* dependents, and secondly, it is unclear yet whether other, non syntactic, patterns are also attended to, although not with maximal attention.

Average attention proportions To refine the analysis, we compute the average attention proportion for a given pattern p and a given attention head: for the $n(p)$ verbal instances for which p exists (e.g. verbs having a *nsubj*), we sum the attention to p and normalize by $n(p)$. We computed the attention proportions for all heads, and for various patterns, including syntactic dependencies, relative positions, the special tokens [CLS] and [SEP], plus patterns targeting the correlation between syntax versus adjacency, such as *nsubj≠i-1*. We provide the results as a heatmap in Figure 4.12.

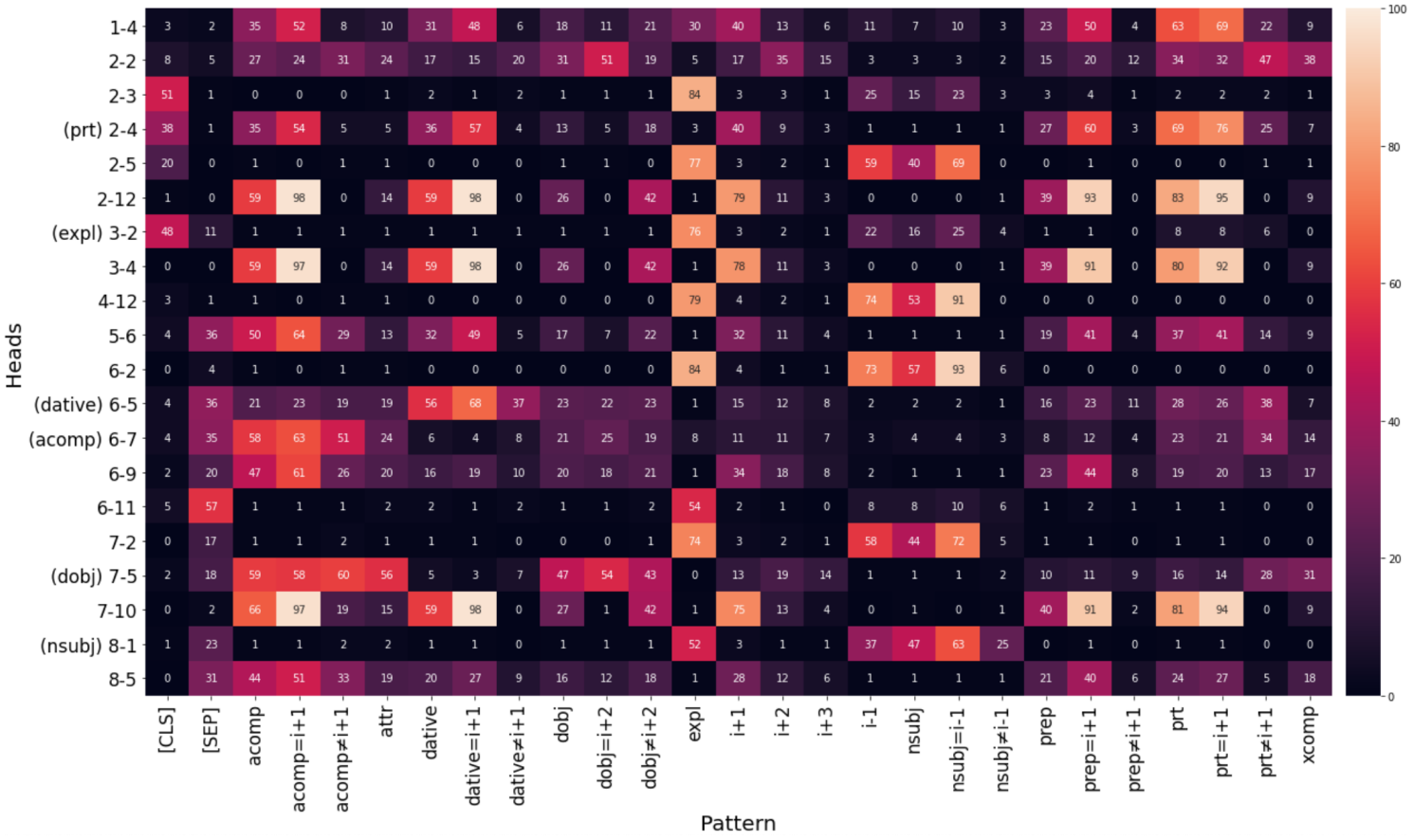

Figure 4.12: For each attention head, percentage of attention paid to certain patterns, when the pattern is instantiated for the verb instance. We only showed the heads having at least one pattern (other than CLS / SEP) holding 50% or more of the attention, and we excluded the patterns that did not receive more than 15% in any of the heads.

Relative position heads Some of the heads are strictly bound to relative positions. For example, heads 2-12 and 3-4 both highly attend to *i+1* which, for verbs, matches the predominant positions of e.g. particles (*prt*) and adjectival complements (*acomp*). But when these dependents are not at position *i+1*, the attention proportion drastically drops (for instance for head 2-12, the attention proportion is 95% when *prt=i+1*

but drops to 0.12% for *prt≠i+1*). This assesses that this attention head remains fixed on a specific relative position.

Lexical heads Some of the syntactic patterns seem to be well predicted by some heads, but prove to exhibit a low lexical diversity. This is the case for instance for the *expl* dependent and head 3-2. The *expl* label is used to label the dependency between *there* and the verb *be*, hence its predominant position is *i-1*. When *expl* is present in the verb's context, the head puts 76% of its attention on it, while overall, the i-1 position receives little attention (22%). Yet, given the lexical specificity of *expl*, it is likely that the 3-2 head has learned a lexical rather than syntactic pattern.

The head 6-7, obtained good results on the prediction of the adjectival complement (cf. 88.9% for *acomp* in Table 4.8). Moreover, the attention proportions for various patterns also suggest that this head is able to capture the *acomp* syntactic dependency: the attention proportion is high on the *acomp* pattern (58%), while its baseline position *i+1* remains overall relatively little attended to in the general case (11%). Furthermore, the 6-7 head consistently focuses on *acomp* even when it is not at *i+1* (51% of the attention goes to *acomp* for the pattern *acomp≠ i+1*). Yet, verbs having a *acomp* dependent have low lexical diversity, with *be* representing 86% of the instances. Hence, it seems that the head has learned to focus on *i+1* specifically when the verb is stative or be. And indeed, when the verb is be, the attention proportion to *acomp* is 58%.

Subject We now investigate to what extent the model does more than learning a shallow positional heuristic for locating subjects. The 8-1 head performs very well at predicting the nominal subject (accuracy is 74.9 in Table 4.8, a +27% increase over the *i-1* positional baseline). Yet, firstly, the attention proportion drastically drops when the subject does not occupy its baseline position (from 63% for *nsubj=i-1* to 25% for *nsubj≠i-1* in Figure 4.12).

Secondly, as shown in Table 4.9, the 8-1 head fails to predict the subject unless it is at *i-1* or *i-2*: accuracy drops to 29.3% when the subject is neither at *i-1* nor *i-2*. The only syntactic ability of the head over the positional baseline seems to be the relatively high accuracy when the subject is at *i-2* (59.8%). Note though that for these instances, the attention proportion on the *i-1* position remains high: for the *nsubj=i-2* instances, 8-1 head pays 37% of its attention to the *i-2* position (hence to the subject), but still 29% on the *i-1* position (Figure 4.13). Furthermore, in the *nsubj=i-2* instances, the *i-1* position mostly corresponds to modals, auxiliaries or adverbs, which exhibit a rather low lexical diversity.

Direct Object For direct objects we find a head that apparently learns a more sophisticated pattern. The head 7-5 outperforms the direct ob-

Pattern	Head	Accuracy	#occ
nsubj (all)	8-1	74.9	44261
nsubj $\neq$ i-1	8-1	43.4	18651
nsubj $\neq$ i-1/-2	8-1	29.3	10029
nsubj $=$ i-2	8-1	59.8	8625

Table 4.9: Accuracy of the best head for *nsubj* using the maximal attention method (8-1), computed on instances for which the subject is in various positions.

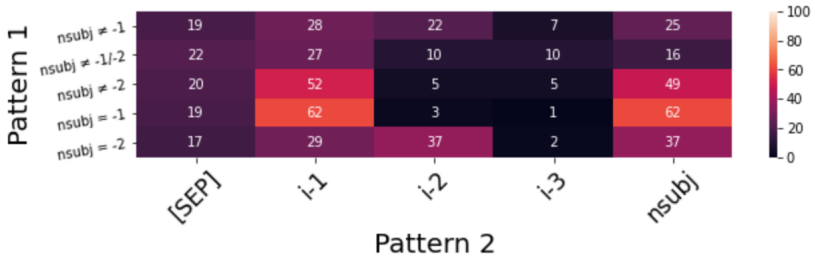

Figure 4.13: Attention proportions of head 8-1 for verb instances showing pattern1 (lines), for various patterns (columns). For instance: among the verb instances such as *nsubj=i-2*, 29% of the attention goes to *i-1*.

ject baseline (*i*+2) by a very large margin (69.9 vs 38.8). Moreover, the baseline is relatively low, indicating that direct objects can be found in multiple positions relative to the verb.

Indeed, the attention percentages for that head in Figure 4.12 show that the attention is distributed homogeneously among the relative positions at the right of the verb rather than focused on a single position. Furthermore, as shown in Figure 4.14, the proportion of attention varies according to the position of the *dobj*. Finally, contrary to the subject head, the 7-5 head succeeds in predicting direct objects, even when these are not in their baseline position as shown in Table 4.10. This confirms its ability to track the verb-object dependency.

Relation	Head	Accuracy	#occ
dobj (all)	7-5	69.9	25288
dobj $==$ 2	7-5	77.1	9652
dobj $!=$ 2	7-5	64.9	15636
dobj $==$ 1	7-5	64.6	7105
dobj $==$ 3	7-5	72.8	4698
dobj $>$ 3	7-5	56.0	3833

Table 4.10: Accuracy of the best head for *dobj* using the maximal attention method (7-5), computed on instances for which the direct object is in various positions

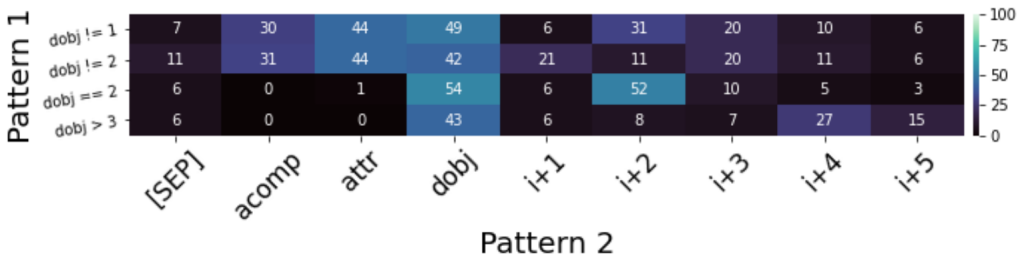

Figure 4.14: Attention proportions of head 7-5 for verb instances showing pattern1 (lines), for various patterns (columns). For instance: among the verb instances such as *dobj>3*, 27% of the attention goes to *i+4*.

4.4.2.5 Conclusion

We have taken advantage of the interpretability of attention mechanisms to evaluate the importance of argument structure for verb disambiguation.

To do so, we first evaluated a VSD classifier based on standard word vector representations with a self attention layer. While the model outperformed several baselines, the study of its attention weights did not reveal any particular interest to the argument structure. Indeed, the weights put on the argument structure remained close to those distributed among random selected words. Besides, selecting either the heads or the whole span did not seem to change the results.

We also analyzed BERT's attention heads on contextualized representations of verbs to assess the ability of the model to capture syntactic dependencies related to verbs' argument structures. Our results suggest that although some heads are good predictors of specific syntactic relations, a closer look reveals that some of them are in fact strongly correlated to simpler patterns such as adjacency or recurrent lexical patterns. Actually, we found that among the dependency labels for verb arguments, only the head specialized in picking the direct object seems to capture a true syntactic dependency.

Given the results of these two experiments, it remains unclear to what extend the argument structure plays a role within attention-based models. It might be the case that, to some degree, some syntactic functions such as direct objects contribute to the context representations of verbs, but our results suggest that there must be some other features at stake as well.

4.5 Learning Contexual Representations from Structured Data

In this last section, we investigate whether providing structural information (obtained *à priori*) to a WSD system can be beneficial. More precisely, as a contribution we propose a new model we called Dag2vec which learns contextual representations from syntax-driven graph based structure of sentences and test it as input representations for WSD. The model is inspired from Context2vec (Melamud et al., 2016) and makes use of recurrent neural models adapted to tree structure input obtained from external parsers.

We first introduce our notations for direct acyclic graphs (DAGs) (Section 4.5.1). Then, we propose a method based on Tree-LSTM (Tai et al., 2015b) to encode the DAG (Section 4.5.2) and give a formal definition of the Dag2vec model (Section 4.5.3).

In Section 4.5.4 we evaluate the model on the WSD task using a simple Knn classifier and compare its results with state-of-the art models.

Finally, we discuss both the computational and theoretical limitations of Dag2vec in Section 4.5.5.

4.5.1 Directed Acyclic Graphs

In this work, we are interested in taking advantage of structured input, in particular syntactic structures. Instead of plain trees though, we will generalize our formalization using Directed Acyclic Graphs (DAG) structures, so that the bottom-up and top-down traversals of the structure be symetric. These structures can encode a syntactic dependency tree but also semantico-syntactic dependency graphs. In what follows we will thus consider Directed Acyclic Graphs (DAGs).

A DAG is a finite directed graph composed of vertices and directed egdes such that it's impossible to start from node v and follow a sequence of edges that loops back to v (i.e a cycle). Every node i in a DAG may have zero, one or several dependents as well as governors. We will note $\textsc{Deps}(i)$ the immediate dependents of i, and $\textsc{Govs}(i)$ its immediate governors of i. Finally edges between nodes may or may not have labels.

Our interest is in the DAG representation of sentences. To obtain these representations, we make use of dependency trees[8], a specific form of DAG. In that configuration, given a sentence S and G its associated DAG, we will denote $w_{i:n}$ the words of S in linear order and $v_{i:n}$ its associated nodes in G. Furthermore, we will add two artificial non-lexical nodes namely dLeaf and dRoot as opposed to lexical nodes (i.e nodes associated to one token). Any node i has either at least one lexical gover-

[8]Our formalization below works for any DAG and thus could be used for semantic dependency graphs, but we leave this for future work.

nor node (resp. dependent node) or is attached to dLeaf (resp. dRoot)[9]. Finally, the labeled edges of the DAG are directly derived from the dependency tree of the sentence. In Figure 4.15 is an illustration of the DAG representation for the sentence "The little dog sleeps peacefully."

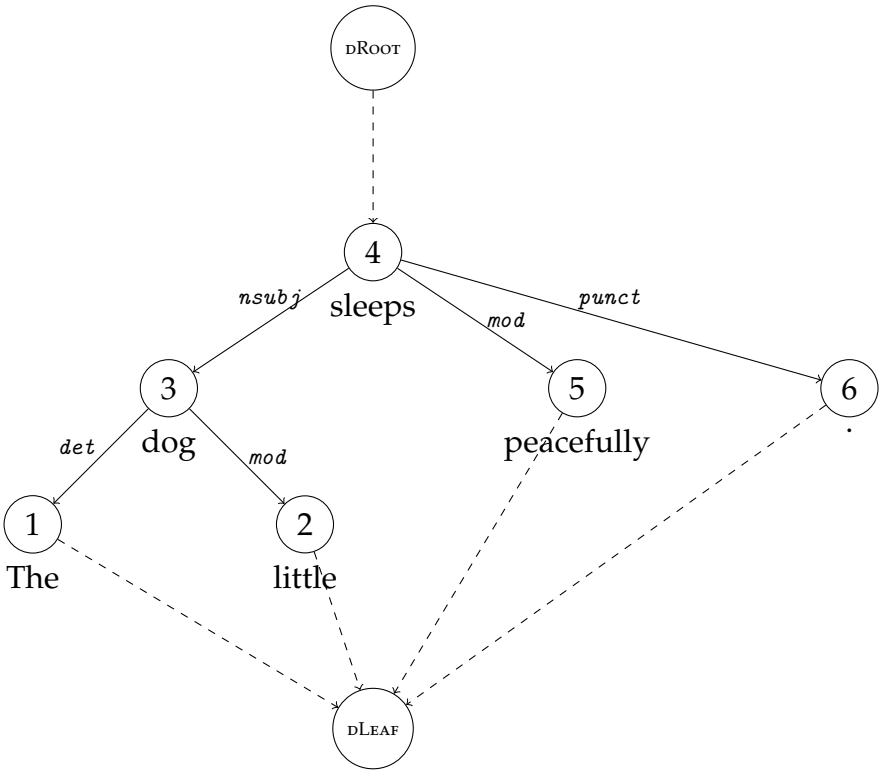

Figure 4.15: An example of DAG representation for the sentence "the little dog sleeps peacefully."

4.5.2 DAG Encoding

We now consider the problem of encoding structural contextual hidden vectors on DAG nodes. For any lexical node i, we note $\mathbf{e}(w_i)$ the non-contextual word embedding for word w_i.

Our encoding of the context of a node i in the DAG will make use of two hidden vectors, $\mathbf{h}_i^{\text{B}}$ and $\mathbf{h}_i^{\text{T}}$, computed in a bottom-up and in a top-down recursive fashion respectively.
The recursive computation of $\mathbf{h}_i^{\text{B}}$ starts from the dLeaf node, to which we associate a null hidden vector (Eq. 4.1)

$$\mathbf{h}_{dLeaf}^{\text{B}} = 0 \tag{4.1}$$

The recurrence is then the following:

$$\mathbf{h}_i^{\text{B}} = \phi(\mathbf{e}(w_i), \textsc{Deps}(i))$$

[9]Though not strictly necessary, these dummy nodes simplify the recursive definitions that we will use below.

The computation of $\mathbf{h}_i^{\mathrm{T}}$ uses the same function ϕ, but traverses the nodes starting from dRoot, and then from heads to their dependents:

$$\mathbf{h}_{dRoot}^{\mathrm{T}} = 0 \tag{4.2}$$

$$\mathbf{h}_i^{\mathrm{T}} = \phi(\mathbf{e}(w_i), \mathrm{Govs}(i))$$

There are multiple candidates to fill the ϕ function. In the next section, we propose and describe one possible solution which makes use of recurrent neural networks.

4.5.2.1 Tree-LSTM

The proposed recursive definitions recall the recurrent encoding of context offered by well-known existing neural nets such as RNNs and LSTMs (more detail on these neural nets can be found in Chapter 3). Let us first recall the equations of the LSTM (Hochreiter and Schmidhuber, 1997b) (Eq. 4.3).

$$\begin{aligned}
i_t &= \sigma\left(W^{(i)}v_t + U^{(i)}h_{t-1} + b^{(i)}\right) \\
f_t &= \sigma\left(W^{(f)}v_t + U^{(f)}h_{t-1} + b^{(f)}\right) \\
o_t &= \sigma\left(W^{(o)}v_t + U^{(o)}h_{t-1} + b^{(o)}\right) \\
u_t &= \tanh\left(W^{(u)}v_t + U^{(u)}h_{t-1} + b^{(u)}\right) \\
c_t &= i_t \odot u_t + f_t \odot c_{t-1} \\
h_t &= o_t \odot \tanh\left(c_t\right)
\end{aligned} \tag{4.3}$$

As one can observe from the equations, we cannot directly use the LSTM as such for the encoding of our DAG representations since it is to be run on linear sequence input. The model needs to be adapted to take structured data as input.

Tai et al. (2015b) proposed two variants of the LSTM that process trees as input. The first one, called the N-ary Tree-LSTM is used in tree structures where the branching factor is stable and where children nodes are ordered. This is not well suited to our dependency tree representations since the number of children can vary greatly (i.g there is no theoretical limit to the number of modifiers). The second version of the Tree-LSTM is called the Child-sum Tree-LSTM (denoted further as CSTL) and is described by the authors as particularly fitted to encode dependency trees. The particularity of the CSTL model is that it computes the hidden state of any node i based on the sum of the hidden states of its dependents

(thus child-sum). We report the equation transitions in Eq. 4.4

$$
\begin{aligned}
\tilde{h}_j^{\mathrm{B}} &= \sum_{k \in \textsc{Deps}(j)} h_k^{\mathrm{B}}, \\
i_j &= \sigma\big(W^{(i)}\mathbf{e}(w_j) + U^{(i)}\tilde{h}_j^{\mathrm{B}} + b^{(i)}\big), \\
f_{jk} &= \sigma\big(W^{(f)}\mathbf{e}(w_j) + U^{(f)}h_k^{\mathrm{B}} + b^{(f)}\big), \\
o_j &= \sigma\big(W^{(o)}\mathbf{e}(w_j) + U^{(o)}\tilde{h}_j^{\mathrm{B}} + b^{(o)}\big), \\
u_j &= tanh\big(W^{(u)}\mathbf{e}(w_j) + U^{(u)}\tilde{h}_j^{\mathrm{B}} + b^{(u)}\big), \\
c_j &= i_j \odot u_j + \sum_{k \in \textsc{Deps}(j)} f_{jk} \odot c_k, \\
h_j^{\mathrm{B}} &= o_j \odot tanh(c_j)
\end{aligned}
\tag{4.4}
$$

Where σ denotes the logistic sigmoid function, and $\odot$ denotes element-wise multiplication. Note that the forget gate (f_{jk} in Eq. 4.4) is applied to every child of the current node. This can be seen as a way to filter the information from the children nodes. This is different from the standard RNN cell where the forget gate impacts the hidden representation of the sentence at the $t-1$ step.

4.5.2.2 Dependency labels

We propose a variant of the CSTL recursive function such that dependency labels are taken into account. To do so, we modify the hidden representation of a child k of j, using the dependency label between j and k, (leading to a representation $h_{kj}^{\mathrm{B}'}$ depending both on k and j)

$$h_{kj}^{\mathrm{B}'} = tanh(W^{(d)}\mathbf{e}(d_{j,k}) + U^{(d)}h_k^{\mathrm{B}} + b^{(d)})$$

and, in Eq 4.4, instead of :

$$\tilde{h}_j^{\mathrm{B}} = \sum_{k \in \textsc{Deps}(j)} h_k^{\mathrm{B}}$$

we use:

$$\tilde{h}_j^{\mathrm{B}} = \sum_{k \in \textsc{Deps}(j)} h_{kj}^{\prime\mathrm{B}}$$

4.5.3 Dag2vec

We now present Dag2vec, a variant of the Context2vec model (presented in Section 3.3.2) which aims at learning context vector representations from DAGs instead of linear input (Figure 4.16). We replaced the biLSTM of the original Context2vec model with two CSTL encoders (which we further note as a "biCSTL") computed in a bottom-up and top-down fashion. Given a sentence $w_{1:n}$ and its associated dependency graph $G_{1:n}$ the context vector representation $\vec{c}$ for w_i is defined as the following

vector concatenation:

$$\text{biCSTL}(i) = \sum_{j \in \text{DEPS}(i)} h_j^{\text{B}} \oplus \sum_{k \in \text{GOVS}(i)} h_j^{\text{T}} \tag{4.5}$$

Following Melamud et al. (2016) we obtain a final vector representation c of the context of i by applying an MLP to the biCSTL representation, in order for the final vector c to have the same dimensionality as the lexical embeddings:

$$\vec{c} = \text{MLP}(\text{biCSTL}(i)) \tag{4.6}$$

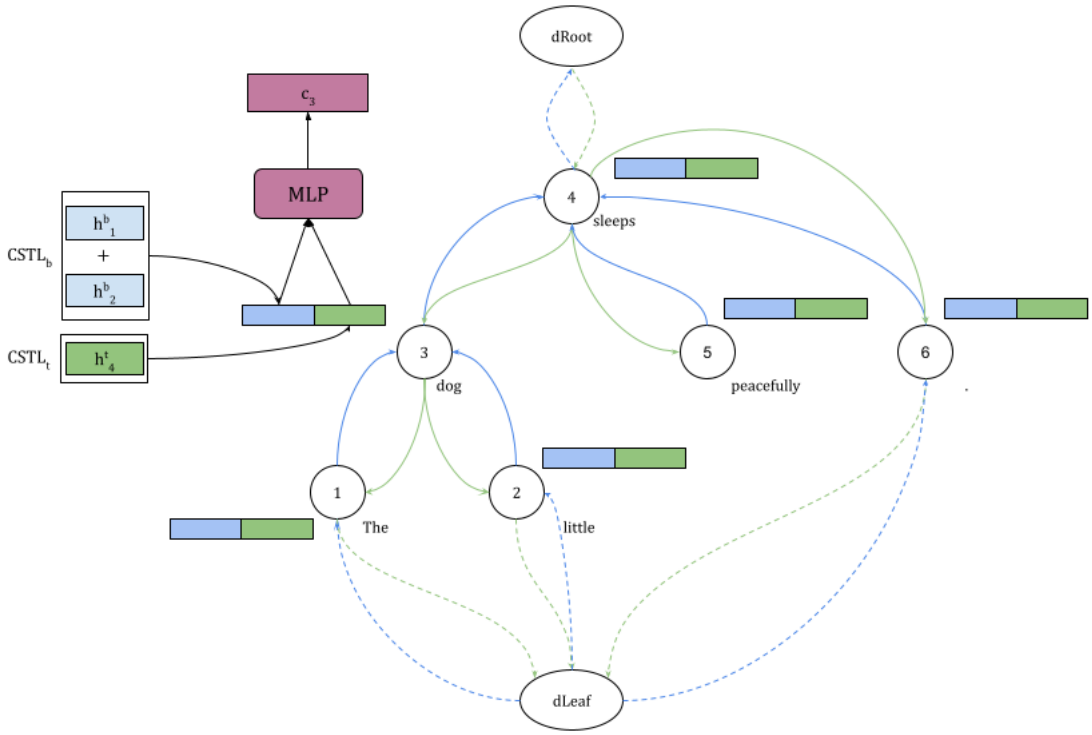

Figure 4.16: Illustration of the Dag2vec encoding. In this illustration, the contextual vector for the word "dog" is obtained merging the output representations of the bottom-up and top-down CSTL encoders through a multi layered perceptron.

It is important to note that the identity of the word i itself is not used to compute biCSTL(i). Only the nodes from i to DLEAF and from i to DROOT are used. This trait will allow to train these contextual representations in a self-supervised way, in which the objective is to predict whether a pair (c, t) is actually a context, target pair.

4.5.3.1 Training Dag2vec with Negative Sampling

As Context2vec, the Dag2vec model can be trained on large corpora using the negative sampling objective function, itself inspired by the skip-gram with negative sampling word2vec model. More precisely, the objective function to maximize, for a given target word t, its context c and n words $t_1, ..., t_n$ sampled to form negative examples:

$$S = \sum_{t,c} \log \sigma(\vec{t} \cdot \vec{c}) + \sum_{i=1}^{k} \log \sigma(-\vec{t_i} \cdot \vec{c}) \tag{4.7}$$

Where $\mathbf{e}(t)$ is the non-contextual word embedding of target word t, t_1 ... t_n are the negative samples and $\vec{c}$ is the context vector representation of t output by the Dag2vec model.

4.5.4 Evaluating Dag2vec on WSD

We evaluated the Dag2vec model on the word sense disambiguation task. To do so, we first trained the model on a large corpus to learn generic contextual representations using the negative sampling objective function defined in Equation 4.7. Once the model was trained, we evaluated its contextual representations on the WSD task using a simple Knn classifier. As a way of comparison we performed the same evaluation using contextual representations obtained from Context2vec and BERT.

4.5.4.1 Training setup

The model was trained using the same set up as the original Context2vec model proposed in (Melamud et al., 2016). We trained the model on the ukWaC corpus, a two billion words British English corpus gathering texts from the .uk web domain. To speed up the training process, sentences with lengths higher than 64 were discarded removing roughly 10% of the original sentences. As the Dag2vec model takes DAGs as input, the corpus was parsed with the `en_core_web_lg` model from the spaCy API[10]. The vocabulary of the model is based on the word frequence distribution of the corpus. We lowercased all texts and kept only the tokens whose frequencies were higher than 100, the remaining were considered as unknown words. Sentences were gathered by lengths and the training took approximatively 40 hours using batches of 500 sentences on a single GPU GTX 1080 ti. The training hyper-parameters are given in 4.11.

[10] `https://spacy.io/models/en`. Spacy actually outputs dependency trees. We leave as future work to test the system using predicted dependency graphs.

Hyper-parameters	
optim	Adam
lr	0.0001
emb dim	300
hid dim	600
out dim	600

Table 4.11: Training hyper-parameters of the Dag2V model on the ukWaC corpus where *emb dim* corresponds to the size of the word embeddings, *hid dim* the hidden units size of the CSTL encoders and *out dim* is the size of the resulting contextual embedding output by the MLP.

4.5.4.2 WSD evaluation

The evaluation was made on the English all-words WSD task available in the unified WSD evaluation framework (Raganato et al., 2017b). We considered SemCor as training data, performed development on SemEval-2007 (Pradhan et al., 2007), and made the evaluation on the remaining datasets: Senseval2 (Edmonds and Cotton, 2001), Senseval3 (Snyder and Palmer, 2004), SemEval-2013 (Navigli et al., 2013), SemeEval-2015 (Moro and Navigli, 2015).

Method To assess our model, we used a Knn classifier such as described in Section 3.4.3.1, more precisely a 1-NN (other k values revealed detrimental): The method basically consists in (1) computing sense vector representations from the training data (2) comparing the vector representation of an example with the sense representations and selecting the one whose cosine similarity is the highest (3) Evaluating the predictions using standard precision, recall, f-score metrics.

We compared the contextual representations of the Dag2vec model with those of Context2vec and BERT.

Results The results of the evaluations are shown in Table 4.12. First, we can see that the Dag2vec model substantially outperformed the most frequent sense baseline in all datasets.

Then, we observe that the contextual representations of BERT consistently achieved the highest performances in all configurations, sometimes by a large margin (e.g more than 6 points in SE13 dataset).

Focusing on the comparison between Dag2vec and Context2vec, both models roughly obtained the same results although overall the latter performed slightly better. Interestingly enough, Dag2vec is better at disambiguating adjectives and adverbs and the reasons for this success remain unclear to us.

Finally, one would have expected Dag2vec to perform better on verbs since the model integrates syntactic features which were traditionally

used in verb sense disambiguation but the results contradict this hypothesis. Indeed, Dag2vec and Context2vec obtained the exact same results on the disambiguation of verbs. Furthermore, we briefly made a comparison of error analysis on verbs between the two models and we could not draw any significant conclusions.

	Dev	Test Datasets				All Test Datasets per PoS				
	SE07	SE2	SE3	SE13	SE15	Nouns	Verbs	Adj.	Adv.	All
Dag2vec	62.5	70.9	68.6	65.4	70.5	70.8	57.4	77.3	87.7	68.9
Context2vec	61.3	71.8	69.1	64.7	71.9	71.2	57.4	75.2	82.7	69.6
BERT	65.7	74.8	71.5	71.9	74.2	75.8	60.3	78.8	89.7	73.1
MFS	54.5	65.6	66.0	63.8	67.1	67.7	49.8	73.1	80.5	65.5

Table 4.12: F-score results of the evaluation of the Dag2vec model on the WSD English All-words task. The table also includes the performances of the Context2vec and BERT models as well as the most frequent sense baseline (MFS) for comparison.

4.5.5 Limitations

We identified several limitations to the Dag2vec model which are of different types. First of all, there are computational difficulties inherent to the use of structured data. Indeed, the model requires the access to dependency parsed sentences implying a preprocess step of the data. This is time and memory consuming depending on the size of the corpus and the speed of the parser in use. Besides, even though recent state-of-the-art models provide high quality parsing, there remains an error rate that can propagate through the data and affect the training of the Dag2vec model *in fine*.

Another computational limitation concerns the training of the model itself. As mentioned earlier, because of the inherent dependency tree representation, the number of dependents per node is highly variable which makes the batching of the data difficult. To face this problem, one can either perform additional operations or use padding to insure that the inputs within batches share the same size. Either way, this makes the training process slower than it is with more standard recurrent neural net based models.

Moreover, there are theoretical limitations to the encoding provided by the Dag2vec model, which may explain the absence of performance improvement in the WSD downstream task. First, the function to aggregate the children nodes potentially has a great impact on the resulting representations. Due to time restrictions we could only experiment with one function, namely the sum of the children nodes, but other functions (i.g average, neural nets) could work as well and should be explored in future work. Secondly, while in linear bidirectional recurrent represen-

tations, all sentence tokens participate to the encoding of a token i, this is not the case in Dag2vec. Indeed, because of the structure of the input and the bottom-up and top-down encoding methods, a whole part of the DAG may be ignored: the context of a node i only includes the nodes intervening in paths from i to DRoot and from i to DLeaf: nodes sharing the same parent or ancestor as i are ignored. For instance, consider the encoding of "dog" in the sentence given in Figure 4.16, its contextual representation completely discards both the final punctuation and the word "peacefully". This could be intensified in real world sentences where structures are much more complex. It could explain the disappointing results on the WSD task where wider context may be necessary to succeed in the task.

4.6 Conclusion

In this chapter we have investigated the potential role of syntax for the disambiguation of verbs through multiple perspectives. First of all, we put a focus on the argument structure of verbs and intended to understand to what extent it plays a part in the representation of their context. Our study based on corpus concludes that sole syntax does not allow proper sense disambiguation of verbs but a combination of syntax and lexical content of syntactic arguments does correlate well with sense distinctions.

Secondly, we tackled the question of the encoding of the argument structure within attention-based models. We first performed experiments using a minimal attention-based architecture and although it obtained satisfactory results on the task, we could not find any significant evidence of emphasis on the argument structure within the attention weights. Then, we took a step further and made experiments with BERT, a state-of-the-art attention-based pre-trained model, checking whether a given BERT head does specialize to attend to specific grammatical functions. While very few argumental functions, such as direct object, seemed to be well identified, most syntactic functions captured by the model were in fact biased towards simpler patterns such as adjacency.

Finally, we proposed Dag2vec, a model that builds contextual representations from syntactic structures of sentences provided a priori. We tested these representations on the WSD task and could not observe any significant improvement, over a linear encoding of contexts, especially on verbs.

All these observations lead us to conclude that, while the argument structure of verbs does correlate with sense distinctions when augmented with lexical content, the optimal way to encode it in neural networks still remains to be found.

Chapter 5

Verb Sense Disambiguation for French

Contents

5.1 Introduction

In this chapter, we investigate to what extent we can create a verb sense disambiguation framework for a language other than English that does not have any sense annotated data. In particular, we take the example of French, a language for which WSD data is almost non-existent.

We start by exploring several resources that could potentially be used for the French verb disambiguation task (Section 5.2). Our first step is to investigate Eurosense (Bovi et al., 2017), a multilingual corpus, including

a subpart in French, automatically sense annotated with synsets from the BabelNet (Navigli and Ponzetto, 2010) semantic network. Our goal is to check whether this resource can be viable to perform supervised WSD even though it was obtained through automatic methods. While investigation came up inconclusive, it led us to explore other resources starting with sense inventories. Indeed, senses inventories generally come with examples, which can serve as annotated data, albeit the resulting annotated corpus does not respect the natural distribution of senses.

In Section 5.2.2 we explore and compare two of them for our task: (1) Babelnet (Navigli and Ponzetto, 2010), a multilingual semantic network based on Wordnet (Miller, 1995) aggregating several semantic resources via automatic translation methods, and (2) Wiktionary, an open source and collaborative multilingual dictionary owned by the Wikimedia foundation. Due to its reasonable granularity and presence of annotated examples, the latter appeared to us as the best candidate for our task.

Having selected Wiktionary as sense inventory and the Wiktionary examples as training corpus, we still needed an evaluation corpus for the verb disambiguation task. We thus developed as a contribution the FrenchSemEval dataset (FSE), the first evaluation dataset for the French verb disambiguation task. FSE gathers occurrences of verbs manually annotated with Wiktionary senses, while preserving the natural sense distribution of these verbs. We present FSE in Section 5.3: we first present the data selection, then we describe the annotation process. Based on a descriptive study, we also propose a comparison with the WSD English datasets.

Finally in Section 5.4, we investigate the usability of Wiktionary for the French VSD task. To do so, we evaluate a Knn-based WSD system on the FSE dataset using the examples from Wiktionary as training data providing thus the very first results on the task. Moreover, in our experiments we use various context representations including those from French pre-trained language models based on the BERT model (Devlin et al., 2018).

5.2 Exploring suitable resources for French VSD

Manually sense annotated data is very rare, especially for languages other than English. This is mainly due to the fact that semantic annotation is a difficult and costly task. As for French, the very little data of that kind can be found in the SemEval-10 (Lefever and Hoste, 2010) and SemEval-13 (Navigli et al., 2013) evaluation datasets. Yet, because these datasets were meant to be used for evaluation only, they are very small. Besides, only nouns were annotated which makes them unusable for our verb sense disambiguation task. Thus, we place ourselves in a configuration where there is no manual data available at all. This section

aims at presenting our search for solutions to the problem of the lack of data.

In Section 5.2.1 we first investigate the use of Eurosense for the French verb disambiguaion task.

Since Eurosense's quality proved too low to serve as the sole source of training data, we then investigated whether sense inventories examples could be a useful source of supervision. In particular, we explored two potential candidates: Babelnet and Wiktionary (Section 5.2.2).

5.2.1 Investigating EuroSense as annotated data

Building a manually sense annotated corpus such as SemCor (Miller et al., 1993) for other languages is hardly conceivable. Indeed it would require too much energy and time and the task would need to be repeated for all languages. One way to get around this issue may be to translate the corpus into the desired language. However, we can find two main drawbacks (at least) to this method. First, the process is highly dependent on the quality of the automatic translation and the word alignment (since a sense label should be attached to a particular word). Secondly, it necessarily implies using an English sense inventory, namely Wordnet.

For these reasons, using automatically or semi-automatically sense tagged corpora might be a preferable path to follow. EuroSense (Bovi et al., 2017) is a corpus of that kind: it is a multilingual corpus automatically sense annotated with Babelnet synsets (more details on Eurosense and Babelnet are provided in Chapter 3). It seemed promising for many reasons: First of all, Babelnet is multilingual. For many languages (500 in the latest version[1]) the lexicon size is substantial, hence if the investigation reveals effective for French, it is likely to be a viable solution for many other languages too. Secondly, the Eurosense corpus is of consequent size and offers very good coverage. Thirdly, the sense annotations are of rather good quality. Indeed, the authors performed a manual evaluation (called "intrinsic" evaluation) of the annotations on randomly selected sentences from the corpus for four languages (including French). It revealed a good inter-annotator agreement (they agreed 85% of the time) as well as a good Kappa score (67.7% on average). We report results of the evaluation for English and French in Table 5.1.

Finally, experiments on English WSD showed that using EuroSense as additional training data on top of SemCor slightly increased WSD performance.

Nevertheless, as for our French VSD task, these results have to be tempered since (i) the precision of the annotations are lower on French than on English, (ii) the disambiguation of verbs has proven to be more difficult than any other part of speech (indeed, Raganato et al. (2017b)'s

[1] `https://babelnet.org/statistics`

	EN		FR	
	Prec.	Cov.	Prec.	Cov.
EuroSense$_{Full}$	80.3	100	67.9	100
EuroSense$_{Refined}$	81.5	75.0	71.8	63.5

Table 5.1: Results of EuroSense intrinsic evaluation on English and French presented in (Bovi et al., 2017). EuroSense$_{Full}$ is the original corpus with all sense annotations. EuroSense$_{Refined}$ is a version where only the annotations predicted with a score superior to a confidence threshold were kept. The evaluation was measured by two metrics: the precision (Prec.) of the automatic annotations compared with the manual annotations considered as gold standard. The coverage (Cov.) which compares the number of actual annotations, assuming that each word in the sampled sentences are targets to be disambiguated.

evaluation framework consistently reports a lowest score on verbs) , (iii) the WSD experiments only concerned English and moreover EuroSense would be used as primer corpus and not additional data for French since no other resource is available.

To have a better idea of EuroSense's usability for our task, we made a manual evaluation of the quality of the automatic sense tagging. The evaluation, which we describe in the next paragraph, is similar to that of Bovi et al. (2017) but it focuses on French verbs.

Evaluation of EuroSense's French verbs We followed the process as described in (Bovi et al., 2017)'s intrinsic evaluation. First, we sampled 50 sentences from the French version of the EuroSense corpus. We selected the high coverage version of the corpus because the gain of precision was too weak (less than 3 points absolute) compared with the loss of coverage (from 100% to 63.5%) (see Table 5.1). We then extracted the non-auxiliary occurrences (160) from the sampled sentences and split them into three sets. Each set was individually annotated by two judges and adjudicated by the third one. The judges were asked to answer "correct" if the sense tag seemed appropriate, even if another sense tag from the BabelNet sense inventory could be more precise. Notice that this is a binary task which is easier than (Bovi et al., 2017)'s original evaluation where annotators were asked to find the adequate sense tag among all available tags. Yet, although the Kappa-cohen score was 0.67, the agreement score of the judges was lower (0.72) than the evaluation performed originally on English. This highlights the difficulty for the judges to decide whether the automatically sense tags were correct.

More importantly, the evaluation revealed that the judges found the annotations correct in only 44% of all 160 verbal occurrences. In the light of these results, we concluded that using EuroSense as sole resource for

French VSD wouldn't be suitable for the rate of correct annotations is too low and does not allow to perform supervised WSD with good quality.

5.2.2 Exploring Potential Sense Inventories

During the evaluation of Eurosense, our annotators encountered some difficulties related to Babelnet which raised the question of the choice of sense inventory for our task. Unlike English, French does not has at its disposal a high quality resource such as Wordnet. The WOLF (Sagot and Fišer, 2008) is the resource that comes closest for French. It was automatically built from Wordnet, using translation techniques. Nevertheless, the coverage is rather low (roughly 30,000 synsets overall in the WOLF versus more than 115,000 in Wordnet) and moreover it has been partly manually validated (only 100 lemmas were manually evaluated). In what follows, we first describe the problems we encountered with the BabelNet sense inventory during the evaluation of Eurosense. Then we introduce Wiktionary and discuss to what extent it could be suitable for the French verb sense disambiguation task.

5.2.2.1 BabelNet

While evaluating Eurosense, we were able to have a better idea of the usability of BabelNet's sense inventory for the task of French VSD. Even though we acknowledged the effort to aggregate multiple multilingual semantic resources in one single network, allowing thus an access to a very vast semantic and lexical knowledge, we found two major obstacles to use it for our purpose.

First, the sense inventory comprises many senses. Indeed, we measured that the number of senses per verb type occurring in the samples sentences used for our evaluation of Eurosense is high (15,5 on average).

Secondly, on a more qualitative level, we found that the boundary between the various senses was often very difficult to grasp. Indeed, the gloss of the senses are often blurry, either made of synonyms or translations which does not help the description of the sense.

Let us illustrate these problems with the verb *affecter*. The entry in BabelNet (v5)[2] for this verb is composed of 14 different senses where only two are drawn from existing resources (Figure 5.1), the remaining being the results of machine translations. Following the order, the first sense (identified as bn:00082426v) occuring in the list is described by an English gloss extracted from Wordnet *Have an effect upon* and a single synonym *toucher*. As one can see, this is a relatively poor description of that particular meaning of *affecter*. Furthermore, one needs to be able to read English to understand that sense. On the contrary, the second sense

[2] `https://babelnet.org/search?word=affecter&lang=FR`

(bn:00082858v) is much more explicit since it provides a full definition in French and two synonyms.

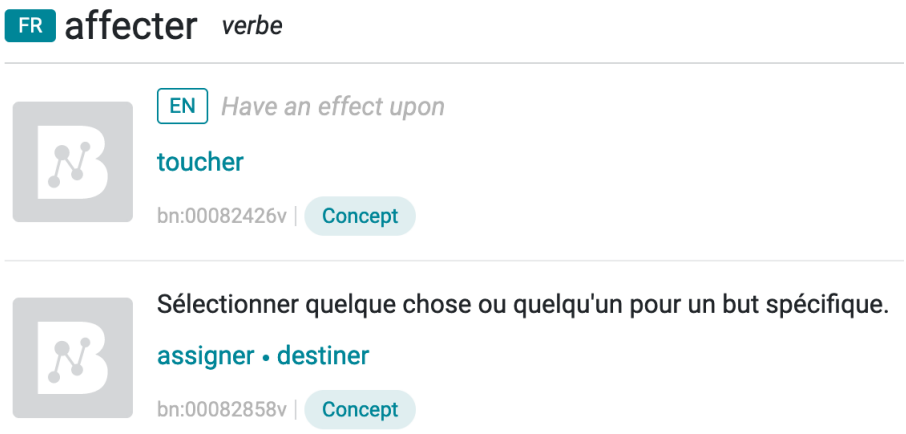

Figure 5.1: The first two senses of the entry for *affecter* in BabelNet.

Now turning to the senses obtained through machine translations (Figure 5.2), some may be easier to understand than others, as long as one can read English. For example, the sense bn:00082429v shows an English gloss, multiple French synonyms and a picture which as whole helps for the description of the meaning. But then, on the one hand there are cases such as bn:00082428v where it is very difficult if not impossible to guess the meaning of the entry based on the sole English gloss. Moreover, there are different senses with very similar descriptions (i.g bn:00086745v and bn:00086746v) which makes them difficult to tell apart.

Overall, even if we discard the senses resulting from machine translation, the obvious lack of intelligibility for the descriptions of the senses made us doubt the usability of BabelNet as sense inventory for our task.

5.2.2.2 Wiktionary

Once we realized that BabelNet would not be suitable for our task we decided to explore the sense inventory provided by Wiktionary, a dictionary version of Wikipedia.

Wiktionary is a collaboratively edited, open-source multilingual online dictionary, hosted by the Wikimedia Foundation. It is an interesting open-source resource and several studies already showed its usefulness for various NLP tasks (e.g. lemmatization (Liebeck and Conrad, 2015)), especially in the lexical semantic field, for extracting or improving thesauri (Navarro et al., 2009; Henrich et al., 2011; Miller and Gurevych, 2014).

Wiktionary's main advantages are that it is entirely open-source, multilingual and has a good coverage for a substantial number of languages

Figure 5.2: Senses obtained from machine translations in BabelNet for the verb *affecter*.

(according to current Wiktionary statistics[3], 22 languages have more than 50, 000 Wiktionary entries each). Each entry consists of a definition and one or several examples, either attested or created, each example being a potential sense-annotated example for the lemma at hand. Definitions and examples point to other Wiktionary pages, which can be useful, although not as useful as if links to Wiktionary senses (not pages) were provided. Furthermore, the structured nature of Wiktionary makes it possible to extract word networks rather easily (as was done for English, German and French by (Sérasset, 2012), in the RDF format).

These advantages come at the cost of Wiktionary's main potential drawback, namely its crowd-sourced nature. Firstly, this means that it is constantly evolving, since any user can edit pages at any time (unless pages that users with more editing rights might have protected). Indeed, new pages are created every day while already existing pages are deleted, modified, merged (note though that every change occurring in the resource is kept in track). Secondly, this means that the resource is not curated by skilled lexicographers only, and the "guidelines" are

[3] https://en.wiktionary.org/wiki/Wiktionary:Statistics

themselves collaboratively built. Despite this potential disadvantage, several features of Wiktionary (good coverage, sound granularity, manual examples provided with sense definition) seemed particularly suitable for the task of WSD and this, combined with the fact that sense-annotated data for French verbs are non-existent, makes it a serious candidate for a new resource of WSD.

Wiktionary for French VSD Our interest for Wiktionary rose after studying random verbal entries for French: we could observe that in general the granularity level is rather "natural" and that the sense distinctions are easy to grasp. Let us have look at the Wikionary page [4] for the verb *affecter* to compare it with BabelNet. It displays 10 different senses, all of them described by a French explicit definition. The words of the definitions can also be hyperlink to other Wiktionary pages (altough not disambiguated). For example the first word *Destiner* of the first sense redirects to its own page in Wiktionary. Furthermore, some definitions may be introduced by a topic word which helps even more to understand the sense (i.g *Programmation* in the definition shown in Figure 5.4). Finally, each sense may be illustrated by one or several examples.

Verbe [modifier le wikicode]

affecter \a.fɛk.te\ *transitif* 1er groupe (VOIR LA CONJUGAISON)

1. Destiner et appliquer une chose ou une personne à un certain usage.
   - *Le coq, ses aides et les hommes **affectés** au service des cuisines, porteront constamment la vareuse et le pantalon de fatigue ; […].* — (Joseph Grégoire Casy, *Organisation du personnel d'un vaisseau*, Paris : Carilian-Goeury & Vr Dalmont, 1840, p.227)
   - ***Affecter** un fonds de terre pour l'entretien, à l'entretien d'une école. — **Affecter** une rente au paiement d'une dette.*
2. Feindre ou exagérer certains sentiments, certaines qualités.
   - *Ceux-ci, d'ailleurs, un peu hautains et dédaigneux, **affectaient** souvent de regarder comme indignes d'eux les amusements habituels des gosses.* — (Louis Pergaud, *Deux Veinards*, dans *Les Rustiques, nouvelles villageoises*, 1921)
   - *Aussitôt l'intrus recula, baissa les yeux et, relevant les pans de sa pèlerine, **affecta** de se chauffer.* — (Francis Carco, *Brumes*, Éditions Albin Michel, Paris, 1935, page 87)
   - *M'autorisez-vous, maintenant, s'informa-t-il, **en affectant** une rondeur pleine de suffisance, à évoquer l'atmosphère du crime ?* — (Francis Carco, *L'Homme de minuit*, Éditions Albin Michel, Paris, 1938)
   - *Malgré sa corpulence excessive, l'autorité de M. Hector sur ses subordonnés n'est guère contestable. Il la doit surtout à sa placidité étudiée, au ton solennel et ampoulé qu'il **affecte** en s'exprimant, […].* — (Jean Rogissart, *Passantes d'Octobre*, Librairie Arthème Fayard, Paris, 1958)
   - *Les papes suivants **affecteront** en revanche d'être offusqués par tant de poitrines opulentes, tant de zigounettes ainsi impudiquement dévoilées. Clément VIII, Paul III et Paul IV songèrent à détruire la fresque...* — (Jean-Michel Renault, *Censure et caricatures: les images interdites et de combat de l'histoire de la presse en France et dans le monde*, éditions Pat à Pan, 2006, p. 23)
3. Marquer une prédilection excessive pour certaines choses.
   - ***Affecter** certains mots, certaines façons de parler, certains airs, certains gestes.*
   - *Il y a là le grand fils de Joux, le fermier des Basses-Fosses, un gaillard aux yeux durs, bon ouvrier déjà, qui a toujours un mégot pendu à la lèvre et **affecte** d'enfoncer ses mains dans ses poches, à la parisienne.* — (Roger Martin du Gard, *Vieille France*, Gallimard, 1933 ; éd. Le Livre de Poche, p. 138.)
   - *Il **affecte** le genre anglais.*

Figure 5.3: The first three senses found on the Wiktionary page for the verb *affecter*. Each sense is described by a definition and one or several examples.

[4] `https://fr.wiktionary.org/wiki/affecter`

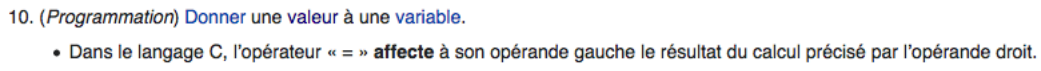

Figure 5.4: The 10th sense of the verb *affecter* in Wiktionary. The definition is introduced by a topic, *Programmation*.

On the quantitative level, we report in Table 5.2 several statistics for the French Wiktionary, in which it can be seen that the resource is large (we will see in the next section that the coverage in corpus is good indeed).

POS	Nb of entries	Nb of senses	Mean nb of senses per entry	Nb of examples
Noun	81099	112428	1.39	1511517
Verb	27271	41207	1.51	55206
Adj	25865	33732	1.30	46212
Adv	5904	6012	1.29	5904

Table 5.2: Statistics from French Wiktionary of the 04-20-2018 dump available via the tool of Sérasset (2012)

After a few manual investigations, double-blind manual annotation would help us quantifying Wiktionary's quality, by checking whether wiktionary's inventory has sufficient coverage and is sound enough to allow for consistent annotation. We conducted such manual annotation and found that Wiktionary was indeed viable to be used for verb sense annotation. The result of this annotation process is a new evaluation resource for the French verb disambiguation task which we present in the next section.

5.3 FrenchSemEval: A New Evaluation Corpus for French VSD

Since the first Senseval evaluation series in 1998 (Kilgarrif, 1998), a various number of evaluation frameworks have been proposed to assess different WSD tasks, but only a few include French test datasets (Lefever and Hoste, 2010; Navigli et al., 2013) and unfortunately these only focus on nouns. In this section we present FrenchSemEval, a new French dataset in which verb occurrences were manually annotated with Wiktionary senses. Our objective was to find out whether Wiktionary's sense inventory is operational for humans to sense-annotate a corpus, and if so, to use it as evaluation data for WSD experiments.

This section is structured as follows: we first describe the selection of

data (Section 5.3.1), the annotation process (Section 5.3.2) and we provide several statistics about the resulting dataset and the quality of the annotations. Then we compare the sense distribution characteristics of FSE, a lexicographic dataset FSE, with that of SemCor, a "natural text" corpus.

5.3.1 Selection of data: A lexical selection

To build FrenchSemEval, we chose to focus on verbs neither too rare nor too frequent, and neither monosemous nor too polysemous. Rare verbs are often monosemous, and very frequent verbs tend to be very polysemous and extremely difficult to disambiguate (we thus left these out for future work). FrenchSemEval was built using the following steps: we first selected a vocabulary of verbs based on their frequency in corpus. We selected verbs appearing between 50 and 1000 times on the French Wikipedia (dumped on 2016-12-12 hereafter fr-Wikipedia), which contains a little more than 45 million sentences.

To get a sense of the performed filtering, we provide in Table 5.3 some statistics about verb occurrences on the French Wikipedia.

Frequence	number of verb lemmas
$n < 50$	14363
$50 > n < 1000$	9929
$n > 1000$	2521

Table 5.3: Number of verb lemmas per frequence on the French Wikipedia. These numbers have to be taken with precaution given the fact that PoS-tagging was automatically performed and hence might have propagated errors.

Secondly, we extracted from this pre-selected list of verbs those having a number of senses comprised between two and ten in Wiktionary's sense inventory in order to discard the monosemics and leave out those that are too polysemous (hence more difficult) for future work. We chose to randomly extract 50 occurrences primarily from corpora comprising other annotations (the French TreeBank (FTB) (Abeillé and Barrier, 2004)) and the Sequoia Treebank (Candito and Seddah, 2012), supplementing the corpus when necessary by occurrences sampled from fr-Wikipedia, in order to reach 50 occurrences per verb.

5.3.2 Annotation process

The annotation was performed by three students over a period of nearly one month. We used WebAnno (Yimam et al., 2014; de Castilho et al., 2016), an open-source adaptable annotation tool. Sentences had already

Number of sentences	3121
Number of annotated verb tokens	3199
Number of annotated verb types	66
Mean number of annotations per verb type	48.47
Mean number of senses per verb type	3.83

Table 5.4: Among the 3121 sampled sentences, 71% comes from the FTB and Sequoia corpora and 29% from the French Wikipedia.

been pre-processed into CoNLL format (Nivre et al., 2007) with the Mind The Gap (MTG) parser (Coavoux and Crabbé, 2017) and were plugged in WebAnno. We were thus able to provide files (one file per verb) containing sentences in which occurrences of the specific verb were marked for annotation. The annotators were asked to annotate only the marked occurrences. We integrated in WebAnno the sense inventory from Wiktionary, including definitions and examples of senses, and added two extra tags: "OTHERPOS" and "MISSINGSENSE". The former was to use when an occurrence was wrongly tagged as verb, and the latter was to use when the sense of an occurrence did not exist in the sense inventory. As Wiktionary is constantly evolving through time, we used the 04-20-2018 dump available via Dbnary (Sérasset, 2012). The annotation was performed in double annotation and curation.

Resulting resource Table 2 reports various statistics about the resulting dataset. It contains 3199 occurrences for 66 different verbs, which means nearly 50 annotated instances per verb (about 100 OTHERPOS occurrences were discarded). The annotators agreed more than 70% of the time and obtained a Kappa score of 0.68 which is good according to the literature (a Kappa between 0.61 and 0.80 indicates a strong agreement). We believe that these metrics indicate an annotation quality which may not be extremely high but still sufficient to validate the coherence of the Wiktionary sense inventory, definitions and examples, despite its non-expert crowd-sourced nature. Finally, 111 occurrences were annotated with the tag "MISSINGSENSE" which indicates a good coverage of the sense inventory.

5.3.3 Comparing English and French Datasets through a Descriptive Study

The best-suited data for training a supervised WSD system is a corpus with sense tags for all content words. Training on such a corpus benefits from basic frequency information found in the corpus. This is particularly striking for WSD, as the "most frequent sense" baseline is known

to be very high. In the case of French, as for the majority of languages, we lack such a corpus, and turn to the Wiktionary examples to serve as training examples for a significant portion of the lexicon. Yet, because senses' distribution differ in the lexicographic examples found in Wiktionary with respect to natural text, we first provide some statistics for a running text sense-annotated corpus such as SemCor (for English) versus a lexicographic training set such as Wiktionary examples (for French).

5.3.3.1 Comparison of the sense distribution in training examples

We study here the distribution of the annotated senses in training data, namely SemCor for English and Wiktionary for French. We first summed up in Table 5.5 general statistics on the two datasets.

	Semcor	Fr-Wiktionary
Nb of sentences	35398	55206
Nb of annotated tokens	88334	55206
Number of annotated verb types	4665	27271
Mean number of annotations per verb type	18.9	1.51

Table 5.5: Comparison of general statistics on verbs between SemCor and FR-Wiktionary.

Now when looking at the number of training examples per sense, we obtain an average of 9.6 and a mean absolute deviation of 11.9 for SemCor, whereas the average is only 1.4 for FR-Wiktionary, and the mean absolute deviation is 1.0. This highlights the lexicographic aspect of the Wiktionary resource where the average number of examples per sense is small but more stable compared with SemCor where it may highly vary because of the natural distribution of the corpus.

5.3.3.2 Evaluation of the task difficulty: comparison of ambiguity rates

We now turn to comparing the difficulty of the WSD task, when tested on English SenseEval datasets versus on FrenchSemEval. Note that performance of WSD systems cannot be used for that purpose, given that it is not comparable across languages and datasets. For a corpus, that is a list of tokens $t_1 \dots t_N$, we rather compute the average ambiguity rate that a WSD system has to face, in two settings:

- **token_AMBIG_fullSI**: the ambiguity rate per token, using the full sense inventory:

$$\frac{1}{N} \sum_{i=1}^{N} \text{n\_senses}(t_i)$$

Language	Corpus (# annotations)	AMBIG_trainSI		AMBIG_fullSI	
		type	token	type	token
English	SemCor (88334)	1.97	7.91	3.24	10.94
	SenseEval2 (517)	4.90	6.7	7.58	10.28
	SemEval 2007 (296)	5.15	6.89	7.78	10.17
	SenseEval 2015 (251)	5.69	6.25	8.48	9.16
French	Wiktionary (55206)	1.66	5.49	1.74	5.68
	FSE (3199)	6.02	6.74	6.15	6.91

Table 5.6: Ambiguity rates for verbs, in the English usual training set (SemCor) and usual evaluation sets, and in the French training set (Wiktionary) and evaluation set (FSE). **AMBIG_trainSI** corresponds to using for the number of senses the sense inventory in the corresponding training corpus, whereas **AMBIG_fullSI** corresponds to using the full sense inventory.

- **token_AMBIG_trainSI**: the ambiguity rate per token, using the sense inventory found in the training corpus

$$\frac{1}{N}\sum_{i=1}^{N} \text{attested\_n\_senses}(t_i)$$

For further information, although not directly measuring the corpus WSD difficulty, we also provide the ambiguity rate per verb type, both using the full inventory or that attested in the training set (shown in the "type" columns in Table 5.6).

We report these metrics in Table 5.6. When studying the difference between the "fullSI" versus "trainSI" modes, namely when using the full sense inventory versus that found in the training set, we have a different trend for the English corpora (containing natural text) and the French ones: for SemCor and the English evaluation sets, there is a drop of ambiguity in trainSI mode. For example, the ambiguity rate in SemCor drops from 10.94 to 7.91 when passing from the fullSI to the trainSI mode. As for the English evaluation datasets the trend is similar with a drop of roughly 3 points in average. This illustrates the usual difficulty to cover rare senses in a corpus of natural, but at the same time this bias towards non rare senses will improve supervised WSD overall performance. Note though that for the French corpora, based on the Wiktionary inventory, there is almost no difference between the two modes of computation (i.g from 5.68 to 5.49 in Wiktionary and from 6.91 to 6.74 in FSE). , illustrating that almost all senses have examples in Wiktionary. This is an interesting feature since sense coverage is an important part of the difficulty of the task. It is indeed very difficult to combine quality and coverage in the same resource.

When comparing, for each language, the figures for the training corpora (SemCor and Wiktionary examples) and for the evaluation datasets, it can be noted that the average ambiguity per token is similar for the training and evaluation datasets, but the average ambiguity per type is much smaller for the training corpora (3.24 for SemCor, and 1.74 for Wiktionary). This is because the lexicon covered in the training corpora is much larger, and contains many more monosemic verbs.

As far as training corpora are concerned, it can be seen that the overall average ambiguity is higher for SemCor than for Wiktionary (e.g. in fullSI mode, 10.94 per token ambiguity for SemCor, versus 5.68 for Wiktionary). It shows that the sense inventory for Wiktionary is less ambiguous than Wordnet's (both for the senses found in SemCor, and overall).

5.4 Assessment of Wiktionary's usability

To investigate the suitability of using Wiktionary for supervised WSD on French verbs, we evaluated several supervised WSD systems on FrenchSemEval, considering the examples of Wiktionary's senses as training data. In our experiments, the supervised WSD systems consist in (1) an encoding method for the context representation and (2) a 1Knn classifier to perform the disambiguation.

In this section, we first describe the various methods we used to obtain the context representations of the verbs' instances and provide details on the disambiguation algorithm (Section 5.4.1). We then propose two sets of experiments (Section 5.4.3): in the first one we evaluate the performance of our WSD systems on FSE using the examples from Wiktionary. In the second row of experiments, we performed "in-domain" training, that is directly using examples from FSE as training instances in order to investigate both quantitatively and qualitatively the impact of the training examples. Finally, we present and discuss the results of these experiments in Section 5.4.4.

5.4.1 Context representations

The difference between the various WSD systems we employed in our experiments is the type of contextuals representations. As it was highlighted in this thesis, the representation of the context is a key element in the disambiguation process and has a great impact on the performance of a WSD system. In our experiments, we explored several methods to encode the context of verbs (whether they are taken from training or evaluation instances) ranging from simple word embeddings (Mikolov et al., 2013b) to state-of-the-art transformer based pre-trained language models like BERT (Devlin et al., 2018). In what follows we describe and give relevant details on these methods.

5.4.1.1 Average Word Embeddings

We implemented a simple model for the context representation that we use as baseline. We first trained a word2vec (Mikolov et al., 2013b) model on fr-Wikipedia[5] to obtain non contextual word vectors. Hyper-parameters training details are provided in Table 5.7. We then represent the context of an occurrence by averaging the vectors of the words found in its context window, which we defined as the 5 words on the left and 5 words on the right of the target word.

Training hyper-parameters	
Optimizer	SGD
Learning rate	1.0
Embedding dim	50
Window size	5
Negative samples	10
NS power	0.75

Table 5.7: Hyper-parameters for the training of the word2vec model on the French Wikipedia corpus.

5.4.1.2 Context2vec

Context2vec(Melamud et al., 2016) is a recurrent neural model that learns a function mapping the context around a target word to a vectorial representation using Mikolov's Negative Sampling objective function (Mikolov et al., 2013a). The Context2vec model represents the context using a bi-directional recurrent neural network (Hochreiter and Schmidhuber, 1997a) that allows to take the sequence of words of the sentence into account, thus contrasting with AWE. The model's architecture was previously presented in the earlier chapters hence we invite the reader to consult Chapter 3 for further details.

All codes and implementations are available publicly so we only adapted them and trained the model on the fr-Wikipedia. Following Melamud et al. (2016), we performed a pre-process step on the training corpus, lowercasing all texts and filtering all sentences whose length was superior to 64 words. We also restrained the model's vocabulary based on words frequency keeping only those that have a at least 100 occurrences in the training corpus. Training details are provided in Table 5.8. Once the model was trained on the fr-Wikipedia, we then ran the learnt model on the Wiktionary and FSE data to obtain contextual representations of the target verb occurrences.

[5]We used the fr-Wikipedia dump of 10-20-2017

Training hyper-parameters	
Optimizer	Adagrad
Learning rate	1e-3
Embedding dim	300
Hidden dim	600
Output dim	300

Table 5.8: Hyper-parameters for the training of the Context2vec model on the French Wikipedia corpus.

5.4.1.3 Transformer-based Pre-trained Language Models

As mentioned in the previous chapter, pre-trained transformer-based language models have become the current leading standard in many NLP tasks, including WSD (Luo et al., 2018; Du et al., 2019; Vial et al., 2019). The BERT model (Devlin et al., 2018) has proven to achieve state-of-the-art whether being finetuned on the task (Du et al., 2019) or used to produce contextual representations which serve for a Knn classifier (Scarlini et al., 2020). The latter method is particularly interesting for our VSD framework as the number of training instances is relatively low which would not allow a proper finetuning.

Given the impact of the BERT model on NLP downstream tasks in English, several works have recently released pre-trained models based on a similar architecture for other languages such as Arabic (Antoun et al., 2020), Dutch (de Vries et al., 2019; Delobelle et al., 2020), Finnish (Virtanen et al., 2019), Italian (Polignano et al., 2019), Portuguese (Souza et al., 2019), Russian (Kuratov and Arkhipov, 2019), Spanish (Cañete et al., 2020), and Vietnamese (Nguyen and Nguyen, 2020). As for French, two models were proposed: CamemBERT and FlauBERT (Le et al., 2020). In our experiments on FSE, we used these two French models as well as the multilingual version of the BERT model since it integrated French texts during its pre-training. The following paragraphs provide details on these three pre-trained language models and present a sum up comparison in Table 5.9

FlauBERT The FlauBERT project, which we contributed to[6], was led by the team of the Université Grenoble Alpes with the collaboration of several researchers from various organizations [7]. The aim of the project was to train a BERT model for French using the new CNRS Jean Zay su-

[6]Although we did not programmatically take part in the training of the model, we participated in the project in multiple ways: providing data, discussing training issues, evaluating the model on the verb disambiguation task, contributing to the writing of the paper.

[7]Université Paris Diderot, E.S.P.C.I, CNRS LAMSADE, PSL Research University

percomputer[8]. The project also led to the creation of FLUE (Le et al., 2018), a benchmark similar to the GLUE (Wang et al., 2018) English dataset that gathers several French NLP comprehension tasks, including the FSE dataset for the verb disambiguation task. All models and data were released publicly[9]. Due to the complexity of the model and because deep learning libraries such as HuggingFace[10] were not as developed as they are now, training and evaluating such model was not an easy task at the time. In what follows, we give brief details on the FlauBERT model that are relevant to the understanding of the issues of our experiments. Nevertheless, for further details we invite the reader to refer to the article on FlauBERT (Le et al., 2020).

The FlauBERT French training corpus consists of 24 sub-corpora gathered from different sources, covering diverse topics and writing styles, ranging from formal and well-written text (e.g Wikipedia and books) [11] to random text crawled from the Internet (e.g Common Crawl).[12]. All texts were tokenized with the Moses tokenizer (Koehn et al., 2007).

Regarding the model architecture, FlauBERT is very similar to BERT (Devlin et al., 2018), whose core is a multi-layer bidirectional Transformer (Vaswani et al., 2017). Following Devlin et al. (2018). Two versions of the model were released:

- FlauBERT$_{base}$: $L = 12, H = 768, A = 12$,
- FlauBERT$_{large}$: $L = 24, H = 1024, A = 16$,

where L, H and A respectively denote the number of Transformer blocks, the hidden size, and the number of self-attention heads. FlauBERT's vocabulary is made of 50k sub-word units obtained through the Byte Pair Encoding (BPE) algorithm (Sennrich et al., 2016). Finally, the training was performed using the masked language model (MLM) objective, a task in which the model learns to predict randomly masked tokens. The original BERT model added a next sentence prediction task to the training objectives where basically the goal is to predict if a sentence (B) actually follows a given sentence (A). Yet, recent studies showed that removing this task does not affect the performance of the model on downstream tasks (Yang et al., 2019; Lample and Conneau, 2019; Liu et al., 2019) and hence it was not included in the training of the FlauBERT model.

CamemBERT CamemBERT (Martin et al., 2020) is an alternative model to FlauBERT proposed by the Facebook AI, Inria and Almanach research teams. It is a monolingual transformer-based language model for French based on the RoBERTa architecture (Liu et al., 2019), a variant of the

[8] http://www.idris.fr/jean-zay/jean-zay-presentation.html
[9] https://github.com/getalp/Flaubert
[10] https://huggingface.co/transformers/
[11] http://www.gutenberg.org
[12] http://data.statmt.org/ngrams/deduped2017

original BERT (Devlin et al., 2018) model. It was trained on the French subpart of the multilingual OSCAR corpora (Suárez et al., 2019) which is composed of texts extracted from Common crawl, an open repository of web data.

While CamemBERT and FlauBERT are similar in their architectures, there are nevertheless some key differences between the two models: (1) the type of data used for the training of the models - CamemBERT is trained exclusively on random text crawled from the internet while the training corpora of the FlauBERT models also include different sources with diverse topics and writing styles. Besides, the size of the training data for CamemBERT is almost twice as big as FlauBERT's. (2) CamemBERT uses SentencePiece for tokenization and Whole-word masking (WWM) as pre-training objective, that is to say masking whole words instead of subwords only, as it is the case in FlauBERT. The WWM method was proposed in (Joshi et al., 2020) and has proven to improve the performances of the pre-trained models which implemented it.

Although multiple versions of the CamemBERT model have now been released (i.g including different model sizes), we performed the experiments with the CamemBERT base version as it was the only available model at the time.

Multilingual BERT In opposition to language-specific models, another trend considers one model estimated for several languages at once with a shared vocabulary. The release of multilingual BERT pioneered this approach.[13]. This multilingual version of BERT is based on the same architecture as the original model and follows the same training objectives (i.e the MLM and NSP tasks). The only difference is that is has been pre-trained on a collection of multilingual corpora. More precisely, the model has been trained on the first 104 top languages with the largest Wikipedia data, including French. Yet, because the sizes of the Wikipedia corpora vary greatly across languages, and in order to keep a certain balance in terms of language representation within the model, the data has been sampled using exponentially smooth weighting with respect to the Wikipedia data sizes. As a result, over-represented languages such as English (with the largest Wikipedia overall) were subsampled while languages with smaller Wikipedia data (i.g Icelandic) were oversampled.

5.4.2 Supervised disambiguation algorithm

All our WSD systems used the same method to perform the disambiguation which relies on a 1-NN classifier. We replicated the supervised WSD method used in (Yuan et al., 2016): a sense representation is learned from

[13] `https://github.com/google-research/bert`

	Multilingual BERT	CamemBERT	FlauBERT$_{base}$/ FlauBERT$_{large}$
Training data	N/A	138 GB†	71 GB‡
Pre-training objectives	NSP/MLM	MLM	MLM
Total parameters	110M	110 M	138 M/ 373 M
Tokenizer	WordPiece 30k	SentencePiece 32K	BPE 50K
Masking strategy	Static + Sub-word masking	Dynamic + Whole-word masking	Dynamic + Sub-word masking

†, ‡: 282 GB, 270 GB before filtering/cleaning.

Table 5.9: Comparison between FlauBERT, CamemBERT and multilingual BERT.

the training data by averaging the context vector representation of its instances, in our case the examples taken from Wiktionary or from FSE in the "in-domain" experiments. Then each test instance is sense tagged with the sense whose representation is the closest, based on cosine similarity, hence the 1-NN classifier.

5.4.3 Experiments

We now describe the experiments we performed on the disambiguation of French verbs using Wiktionary and FrenchSemEval.

5.4.3.1 Wiktionary experiment

Our first experiment aimed at assessing the suitability of using Wiktionary in a supervised configuration for the French verb disambiguation task. To do so, we used the annotated examples of the senses in the Wiktionary sense inventory as training data and then, using the different context representations described in Section 5.4.1, we evaluated the performance of our disambiguation algorithm on the evaluation examples in the FSE dataset.

5.4.3.2 In domain experiments

In order to better identify the potential error sources, we also performed experiments with "in-domain" training instances, namely instances directly taken from FSE. To evaluate the impact of the number of training examples per sense, a property that is quite different for a lexicographic training set and for a corpus-based training set, we performed experiments on different sets, using N_{max} a varying maximum number of examples per sense. More precisely, for each verb we sampled respectively 1, 2, 5 and 10 maximum training examples per sense from the dataset and evaluated the disambiguation on the remaining examples.

5.4.4 Results and Analysis

Results of the Wiktionary experiment The results of our experiments using Wiktionary's sense annotated examples as training data are presented in Table 5.10.

Model	Accuracy
AWE	40.0
C2V	43.0
mBERT	49.83
CamemBERT	50.02
FlauBERT$_{base}$	43.92
FlauBERT$_{large}$	**50.48**
MFS	0.30

Table 5.10: Accuracy scores (%) on the FSE evaluation dataset using Wiktionary's sense annotated examples as training data.

We compared our multiple WSD systems with a Most Frequent Sense baseline (MFS) which is computed in a supervised manner, tagging each test instance with the most frequent sense found in the training data. Because of the natural distribution of the senses in corpus, the MFS has been known to be a rather challenging baseline for the WSD task (Raganato et al., 2017b) (i.g 49% F-score on English verbs). Indeed, as seen in Chapter 4, our corpus study performed on SemCor (Miller et al., 1993) has shown that very often a particular sense tends to be over-represented in the distribution of senses, which explains why the MFS baseline is difficult to beat. Besides, many occurrences in the English WSD datasets are easy to disambiguate since they are monosemous[14]. Now, the result of the MFS baseline in the Wiktionary experiment is relatively low (with only 30% accuracy) and to understand why MFS is a rather weak predictor in our setup , we have to recall that contrary to Semcor and the Senseval/Semeval evaluation datasets, FSE is built following a particular perspective: the sentences are sampled in a non-natural way and monosemic words in particular were excluded, hence the poor result of the MFS.

Turning to the performances of the different WSD systems, we can observe that they all outperformed the MFS baseline, sometimes by a large margin.

First, comparing the C2V and AWE context representations, we can see the gain when using recurrent neural net based models (43% for C2V versus 40% for AWE) even though the performance on the French verbs

[14]The MFS on verbs decrease from 49% to 43% F-score when we only consider the polysemous ones.

remains modest[15]. It supports Melamud et al. (2016)'s observations, especially regarding the fact that Context2vec succeeds better in capturing context information than the common averaging of word vector representations.

Now, focusing on the transformer-based contextual representations, all models except FlauBERT$_{base}$[16] significantly widened the gap with C2V and AWE since they roughly improved the performance by 7 points. In particular, CamemBERT and FlauBERT$_{large}$ obtained similar results, the latter achieving a new state-of-the-art on the French verb disambiguation task with an accuracy of 50.48%. Finally, it is worth mentioning that the multilingual BERT model has also more than honorably succeeded on the task, keeping up with the French monolingual models. This is particularly interesting since it demonstrates the effectiveness of multilingual transformer-based models and reinforces the prospect of using these models for languages with smaller resources.

Yet, as powerful as the transformer-based pre-trained language model can be, using only Wiktionary examples to perform supervised disambiguation on verbs remains a rather adversarial setup since, at best, only one out of two verb occurrences are correctly disambiguated. We believe that apart from the lack of training examples, a potential source of error may come from the fact that very often Wiktionary's examples are either complex sentences taken from the literary genre or very short definition-like sentences (an example of such discrepancy is given in Figure 5.5 with the verb *affecter*), which may cause some trouble to the WSD systems when facing instances taken from other domains such as newspapers as it is the case in FSE.

3. Marquer une prédilection excessive pour certaines choses.
   - ***Affecter** certains mots, certaines façons de parler, certains airs, certains gestes.*
   - *Il y a là le grand fils de Joux, le fermier des Basses-Fosses, un gaillard aux yeux durs, bon ouvrier déjà, qui a toujours un mégot pendu à la lèvre et **affecte** d'enfoncer ses mains dans ses poches, à la parisienne.* — (Roger Martin du Gard, *Vieille France*, Gallimard, 1933 ; éd. Le Livre de Poche, p. 138.)
   - *Il **affecte** le genre anglais.*
   - *Il **affecte** toujours de dire des choses flatteuses.*
   - *Il **affecte** de dire en grand secret des choses insignifiantes.*
   - *Il **affecte** l'air distrait.*
   - *Il **affecte** de grands airs.*

Figure 5.5: Description of one of the senses for the verb *affecter*. Some sentences are very brief and look like definitions or imply the use of pronouns (first and third examples) while other are much more complex taken from the literary genre (second example).

[15] Although the English and French WSD setups can not be comparable, for information purposes only, the results for supervised WSD for English verbs are around 0.55 in the benchmark of Raganato et al. (2017b)

[16] We were not able to explain why it did not perform as well as the other BERT models and further investigation should be made.

Results of the "in domain" experiments In these experiments we investigated two potential ways to leverage the difficulty of the initial setup: the size of the training data and domain adaptation. To study the effect of the amount of training instances, since Wiktionary is limited in terms of number of examples per sense, we switched to using FSE both for training and testing. We used a variable number of maximum training examples per sense from $N_{max} = 1$ to $N_{max} = 10$ and we used the remaining examples as test set[17]. Due to time restrictions we only experimented the C2V and AWE context representations, although we think that using other models for context representations may reveal the same trends. The results of these experiments are summarized in Table 5.11 and illustrated in Figure 5.6 .

Models	N_{max} = 1	N_{max} = 2	N_{max} = 3	N_{max} = 5	N_{max} = 10
MFS	0.32	0.38	0.45	0.52	0.70
AWE	0.44	0.53	0.58	0.64	0.70
C2V	0.5	0.57	0.62	0.68	**0.74**
Mean nb of training ex. per sense	1	1.81	2.61	3.86	6.30
Mean nb training ex. per verb	3.83	6.95	9.81	14.8	24.15
Mean nb test ex. per verb	44.63	41.51	38.65	33.66	24.31

Table 5.11: Training on FSE examples, with varying maximum number of examples per sense (N_{max}).
Top: WSD accuracies. Bottom: training / test sets statistics.

Let us observe first the impact of the amount of training data. The mean number of examples in Wiktionary for the verbs occurring in FSE is $N_{avg} = 3.1$ and the results show that all classifiers dramatically improve when the available training examples per verb increases up to $N_{max} = 10$. Actually, the results suggest that performance would continue to improve as Nmax grows, but performance begins to be acceptable even with Nmax=10. This means that if we were able to expand with absolute certainty the small amount of examples in Wiktionary, we could get a much higher disambiguation performance.

Secondly, using the same setup we can compare the behaviour of the classifiers when predicting out of domain (Table 5.10) with in domain predictions (Table 5.11). It is worth recalling that Wiktionary examples are often either long literary sentences or very concise examples whereas the test instances are sampled from Le Monde newspaper or Wikipedia.

[17]We say that we use N_{max} as a maximum number of training examples because some senses may have only $K < N_{max}$ annotated instances in the whole data set. The actual average number of senses for each Nmax is provided in table 5.11

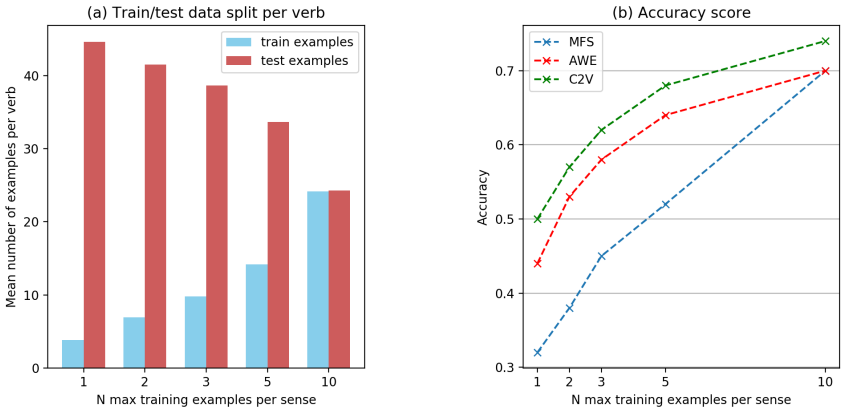

Figure 5.6: Illustration of the results reported in table 5.11. On the left, an histogram that represent the sizes of the training and test data given N, the maximum number of selected training examples. On the right, the accuracies of the models with respect to N.

Again as Wiktionary has $N_{avg} \approx 3$ training examples per sense we can see that the domain adaptation effect is worth roughly 20 points in accuracy. Indeed, when comparing the results of the wiktionary and in-domain experiments with the context2vec model, one can observe an improvement from 0.43 to 0.62 points. This observation gives us an interesting lead to follow for future improvement as the gain regarding the domain adaptation is significant.

5.4.5 Conclusion

Our experiments using the annotated examples of Wiktionary as training data to perform supervised VSD on FSE led to modest but encouraging results. Keeping in mind that the examples are provided freely from Wiktionary users, we believe this is a proof that Wiktionary examples are a valuable source of supervision for VSD. We were also able to quantify the gain in performance that could be obtained by adding a moderate number of seed instances which may be an interesting lead to follow. As for now, we will consider these results as a first baseline for FSE dataset.

5.5 Conclusion

In this chapter we have proposed a method to create a verb sense disambiguation framework for languages with limited or non-existent sense annotated data.

We first explored Eurosense as a resource for sense annotated data. Our study showed us that it is not, in its current state, suitable for our task since our manual evaluation revealed that the automatic sense-

tagging was not accurate enough (less than 50%). Moreover, Babelnet, the sense inventory used in Eurosense's annotations, turned out to be either unintelligible (sense distinctions proved quite difficult to draw in an annotation experiment) or too redundant due to the automatic translation methods employed.

Instead, we proposed to use Wiktionary for three reasons: (1) It is open source and multilingual, it exists with substantial size for a large number of languages (2) the granularity of the sense inventory is more reasonable and (3) its contains sense manually annotated examples that can be used as training data for VSD supervised systems.

As a contribution, we created FrenchSemEval, the first manually sense annotated dataset for the evaluation of French verbs. It is a corpus composed of 3199 verb occurences manually annotated with senses from Wiktionary. It has been built in order to evaluate VSD supervised systems on the French verb sense disambiguation task.

Using FSE, we were then able to assess the usability of Wiktionary's annotated examples as training data for our task. We experimented a WSD system based on contextual representations and a 1-NN disambiguation algorithm, providing the first results for the French verb disambiguation task with the FSE dataset. Furthermore, we participated in the development of FlauBERT, a pre-trained transformer-based language model for French and performed an evaluation on FSE. We also compared FlauBERT with two other transformer-based models, namely CamemBERT and Multilingual BERT. Overall, the results of our experiments made FlauBERT the new state-of-the-art on the French VSD task and highlighted the efficiency of transformer-based contextual representations.

Finally, in the course of "in domain" experiments we evaluated the impact of the training examples both on a quantity and quality level and found that minimally supplementing the Wiktionary examples with specific domain-adapted examples would lead to significant improvements.

Bibliography

Anne Abeillé and Nicolas Barrier. 2004. Enriching a French Treebank. In Proceedings of LREC 2004, Lisbon, Portugal.

Eneko Agirre, Oier López de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. Computational Linguistics, 40(1):57–84. Publisher: MIT Press.

Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In Proceedings of the 4th International Workshop on Semantic Evaluations, pages 7–12. Association for Computational Linguistics.

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), pages 33–41.

Eneko Agirre, Aitor Soroa, and Mark Stevenson. 2010. Graph-based word sense disambiguation of biomedical documents. Bioinformatics, 26(22):2889–2896. Publisher: Oxford University Press.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. arXiv preprint arXiv:2003.00104.

Jordi Atserias, Luıs Villarejo, German Rigau, Eneko Agirre, John Carroll, Bernardo Magnini, and Piek Vossen. 2004. The meaning multilingual central repository. In 2nd International Global Wordnet Conference, January 20-23, 2004: proceedings, pages 23–30. Masaryk University.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. arXiv preprint arXiv:1607.06450.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

Satanjeev Banerjee. 2002. Adaptating the Lesk Algorithm for Word Sense Disambiguation to Wordnet. Thesis.

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In LREC

2012, Eighth International Conference on Language Resources and Evaluation.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. Journal of machine learning research, 3(Feb):1137–1155.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2):157–166. Publisher: IEEE.

Michele Bevilacqua and Roberto Navigli. 2019. Quasi Bidirectional Encoder Representations from Transformers for Word Sense Disambiguation. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019), pages 122–131.

Claudio Delli Bovi, Jose Camacho-Collados, Alessandro Raganato, and Roberto Navigli. 2017. Eurosense: Automatic harvesting of multilingual sense annotations from parallel text. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2:594–600.

Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, pages 103–111. Association for Computational Linguistics.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In to appear in PML4DC at ICLR 2020.

Jose Camacho-Collados and Taher Pilehvar. 2018. From Word to Sense Embeddings: A Survey on Vector Representations of Meaning. arXiv preprint arXiv:1805.04032.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. Artificial Intelligence, 240:36–64. Publisher: Elsevier.

Marie Candito and Djamé Seddah. 2012. Le corpus Sequoia: annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In TALN 2012-19e conférence sur le Traitement Automatique des Langues Naturelles.

Rich Caruana. 1997. Multitask learning. Machine learning, 28(1):41–75. Publisher: Springer.

Richard Eckart de Castilho, Eva Mujdricza-Maydt, Seid Muhie Yimam, Silvana Hartmann, Iryna Gurevych, Anette Frank, and Chris Biemann.

2016. A web-based tool for the integrated annotation of semantic and syntactic structures. In Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH), pages 76–84.

Yee Seng Chan and Hwee Tou Ng. 2005. Scaling up word sense disambiguation via parallel texts. In AAAI, volume 5, pages 1037–1042.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A unified model for word sense representation and disambiguation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1025–1035.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Maximin Coavoux and Benoit Crabbé. 2017. Incremental Discontinuous Phrase Structure Parsing with the GAP Transition. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pages 1259–1270, Valencia, Spain. Association for Computational Linguistics.

D Alan Cruse. 2000. Aspects of the micro-structure of word meanings. Polysemy: Theoretical and computational approaches, pages 30–51.

Tim Van de Cruys and Marianna Apidianaki. 2011. Latent semantic word sense induction and disambiguation. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pages 1476–1485. Association for Computational Linguistics.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In Advances in neural information processing systems, pages 3079–3087.

Pieter Delobelle, Thomas Winters, and Bettina Berendt. 2020. Robbert: a dutch roberta-based language model. arXiv preprint arXiv:2001.06286.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Mona Diab. 2004. Relieving the data acquisition bottleneck in word sense disambiguation. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04), pages 303–310.

Dmitriy Dligach and Martha Palmer. 2008. Novel semantic features for verb sense disambiguation. In Proceedings of ACL-08: HLT, Short Papers, pages 29–32.

Jiaju Du, Fanchao Qi, and Maosong Sun. 2019. Using bert for word sense disambiguation. arXiv preprint arXiv:1909.08358.

Philip Edmonds and Scott Cotton. 2001. SENSEVAL-2: overview. In Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems, pages 1–5.

Andreas Eisele and Yu Chen. 2010. MultiUN: A Multilingual Corpus from United Nation Documents. In Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), Valletta, Malta. European Language Resources Association (ELRA).

Jeffrey L. Elman. 1990. Finding structure in time. Cognitive science, 14(2):179–211. Publisher: Wiley Online Library.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 897–906.

Winthrop Nelson Francis, Henry Kucera, Henry Kučera, and Andrew W Mackie. 1982. Frequency analysis of English usage: Lexicon and grammar. Houghton Mifflin.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1992. A method for disambiguating word senses in a large corpus. Computers and the Humanities, 26(5-6):415–439.

Weiwei Guo and Mona Diab. 2010. Combining orthogonal monolingual and multilingual sources of evidence for all words wsd. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1542–1551.

Michael U. Gutmann and Aapo Hyvärinen. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. The journal of machine learning research, 13(1):307–361. Publisher: JMLR. org.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778.

Verena Henrich, Erhard Hinrichs, and Tatiana Vodolazova. 2011. Semi-automatic extension of GermaNet with sense definitions from Wiktionary. In Proceedings of the 5th Language and Technology Conference (LTC 2011), pages 126–130.

Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 6(02):107–116. Publisher: World Scientific.

Sepp Hochreiter and Jürgen Schmidhuber. 1997a. Long short-term memory. Neural computation, 9(8):1735–1780.

Sepp Hochreiter and Jürgen Schmidhuber. 1997b. Long short-term memory. Neural computation, 9(8):1735–1780. Publisher: MIT Press.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019. Do Attention Heads in BERT Track Syntactic Dependencies? arXiv preprint arXiv:1911.12246.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. arXiv preprint arXiv:1908.07245.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for word sense disambiguation: An evaluation study. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 897–907.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. Transactions of the Association for Computational Linguistics, 8:64–77.

Mikael K\aagebäck and Hans Salomonsson. 2016. Word sense disambiguation using a bidirectional lstm. arXiv preprint arXiv:1606.03568.

Daisuke Kawahara and Martha Palmer. 2014. Single Classifier Approach for Verb Sense Disambiguation based on Generalized Features. In LREC, pages 4210–4213.

Adam Kilgarrif. 1998. Senseval: An exercise in evaluating word sense disambiguation programs. In Proceedings of the first international conference on language resources and evaluation (LREC 1998), Granada, Spain, pages 581–588.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2015. Character-aware neural language models. arXiv preprint arXiv:1508.06615.

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2005. Extending VerbNet with Novel Verb Classes. page 6.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In MT summit, volume 5, pages 79–86. Citeseer.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions, pages 177–180.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the Dark Secrets of BERT. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Henry Kucera, Henry Kučera, and Winthrop Nelson Francis. 1967. Computational analysis of present-day American English. Brown university press.

Yuri Kuratov and Mikhail Arkhipov. 2019. Adaptation of deep bidirectional multilingual transformers for russian language. arXiv preprint arXiv:1905.07213.

Sandra Kübler and Desislava Zhekova. 2009. Semi-supervised learning for word sense disambiguation: Quality vs. quantity. In Proceedings of the International Conference RANLP-2009, pages 197–202.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In Advances in neural information processing systems.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbé, Laurent Besacier, and Didier Schwab. 2020. FlauBERT: Unsupervised Language Model Pre-training for French. In Proceedings of the 12th Language Resources and Evaluation Conference, pages 2479–2490, Marseille, France. European Language Resources Association.

Minh Le, Marten Postma, Jacopo Urbani, and Piek Vossen. 2018. A Deep Dive into Word Sense Disambiguation with LSTM. In Proceedings of the 27th International Conference on Computational Linguistics, pages 354–365.

Els Lefever and Veronique Hoste. 2010. Semeval-2010 task 3: Cross-lingual word sense disambiguation. In Proceedings of the 5th International Workshop on Semantic Evaluation, pages 15–20. Association for Computational Linguistics.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In Proceedings of the 5th annual international conference on Systems documentation, pages 24–26.

Beth Levin. 1993. English verb classes and alternations: A preliminary investigation. University of Chicago press.

Matthias Liebeck and Stefan Conrad. 2015. IWNLP: Inverse Wiktionary for Natural Language Processing. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 414–418, Beijing, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

Fuli Luo, Tianyu Liu, Qiaolin Xia, Baobao Chang, and Zhifang Sui. 2018. Incorporating glosses into neural word sense disambiguation. arXiv preprint arXiv:1805.08028.

John C. Mallery. 1988. Thinking about foreign policy: Finding an appropriate role for artificially intelligent computers. In Master's thesis, MIT Political Science Department. Citeseer.

Suresh Manandhar, Ioannis P Klapaftis, Dmitriy Dligach, and Sameer S Pradhan. 2010. SemEval-2010 task 14: Word sense induction & disambiguation. In Proceedings of the 5th international workshop on semantic evaluation, pages 63–68. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. Computational Linguistics, 19(2):313–330.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and

Benoît Sagot. 2020. CamemBERT: a tasty French language model. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7203–7219, Online. Association for Computational Linguistics.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In Proceedings of the 20th SIGNLL conference on computational natural language learning, pages 51–61.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pages 3111–3119.

George A Miller. 1995. WordNet: a lexical database for English. Communications of the ACM, 38(11):39–41. Publisher: ACM.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: An on-line lexical database. International journal of lexicography, 3(4):235–244. Publisher: Oxford University Press.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A Semantic Concordance. In Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993.

Tristan Miller and Iryna Gurevych. 2014. WordNet—Wikipedia—Wiktionary: Construction of a Three-way Alignment. In LREC, pages 2094–2100.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. arXiv preprint arXiv:1206.6426.

Andrea Moro and Roberto Navigli. 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015), pages 288–297.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. Transactions of the Association for Computational Linguistics, 2:231–244. Publisher: MIT Press.

Emmanuel Navarro, Franck Sajous, Bruno Gaume, Laurent Prévot, Hsieh ShuKai, Kuo Tzu-Yi, Pierre Magistry, and Huang Chu-Ren. 2009. Wiktionary and NLP: Improving synonymy networks. In Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources, pages 19–27. Association for Computational Linguistics.

Roberto Navigli. 2009. Word sense disambiguation: A survey. ACM computing surveys (CSUR), 41(2):1–69. Publisher: ACM New York, NY, USA.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. Semeval-2013 task 12: Multilingual word sense disambiguation. In Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013), pages 222–231.

Roberto Navigli and Simone Paolo Ponzetto. 2010. BabelNet: Building a Very Large Multilingual Semantic Network. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 216–225, Uppsala, Sweden. Association for Computational Linguistics.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial Intelligence, 193:217–250.

Dat Quoc Nguyen and Anh Tuan Nguyen. 2020. Phobert: Pre-trained language models for vietnamese. arXiv preprint arXiv:2003.00744.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL).

Alexander Panchenko. 2016. Best of both worlds: Making word sense embeddings interpretable. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), pages 2649–2655.

Alexander Panchenko, Eugen Ruppert, Stefano Faralli, Simone Paolo Ponzetto, and Chris Biemann. 2017. Unsupervised does not mean uninterpretable: the case for word sense induction and disambiguation.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. arXiv preprint arXiv:1802.05365.

Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. 2002. Developing an aligned multilingual database. In Proceedings of Global WordNet Conference.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. 2019. AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019), volume 2481. CEUR.

Simone Paolo Ponzetto and Roberto Navigli. 2010. Knowledge-rich word sense disambiguation rivaling supervised systems. In Proceedings of the 48th annual meeting of the association for computational linguistics, pages 1522–1531.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. Semeval-2007 task-17: English lexical sample, srl and all words. In Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007), pages 87–92.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2016. Automatic Construction and Evaluation of a Large Semantically Enriched Wikipedia. In IJCAI, pages 2894–2900.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017a. Neural sequence learning models for word sense disambiguation. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1156–1167.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017b. Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pages 99–110, Valencia, Spain. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. arXiv preprint arXiv:1806.03822.

Will Roberts and Valia Kordoni. 2012. Using Verb Subcategorization for Word Sense Disambiguation. In LREC, pages 829–832.

Eugen Ruppert, Manuel Kaufmann, Martin Riedl, and Chris Biemann. 2015. JoBimViz: a web-based visualization for graph-based distributional semantic models. In Proceedings of ACL-IJCNLP 2015 System Demonstrations, pages 103–108.

Benoît Sagot and Darja Fišer. 2008. Building a free French wordnet from multilingual resources. In OntoLex.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. SensEmBERT: Context-Enhanced Sense Embeddings for Multilingual Word Sense Disambiguation. In AAAI, pages 8758–8765.

Vincent Segonne, Marie Candito, and Benoît Crabbé. 2019. Using Wiktionary as a resource for WSD : the case of French verbs. In Proceedings of the 13th International Conference on Computational Semantics - Long Papers, pages 259–270, Gothenburg, Sweden. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1715–1725.

Benjamin Snyder and Martha Palmer. 2004. The English all-words task. In Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, pages 41–43.

Fabio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2019. Portuguese named entity recognition using bert-crf. arXiv preprint arXiv:1909.10649.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In Advances in neural information processing systems, pages 3104–3112.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In 7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7). Leibniz-Institut für Deutsche Sprache.

Gilles Sérasset. 2012. Dbnary: Wiktionary as a LMF based Multilingual RDF network. In Language Resources and Evaluation Conference, LREC 2012.

Kaveh Taghipour and Hwee Tou Ng. 2015a. One Million Sense-Tagged Instances for Word Sense Disambiguation and Induction. In Proceedings of the Nineteenth Conference on Computational Natural Language Learning, pages 338–344, Beijing, China. Association for Computational Linguistics.

Kaveh Taghipour and Hwee Tou Ng. 2015b. One million sense-tagged instances for word sense disambiguation and induction. In Proceedings of the Nineteenth Conference on Computational Natural Language Learning, pages 338–344.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015a. Improved semantic representations from tree-structured long short-term memory networks. arXiv preprint arXiv:1503.00075.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015b. Improved semantic representations from tree-structured long short-term memory networks. arXiv preprint arXiv:1503.00075.

Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. In Sixth international conference on data mining (ICDM'06), pages 613–622. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, \Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2017. Représentation vectorielle de sens pour la désambiguïsation lexicale à base de connaissances. In 24ème Conférence sur le Traitement Automatique des Langues Naturelles.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2019. Sense vocabulary compression through the semantic knowledge of wordnet for neural word sense disambiguation. arXiv preprint arXiv:1905.05677.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: Bert for finnish. arXiv preprint arXiv:1912.07076.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Piek Vossen. 1998. Introduction to eurowordnet. In EuroWordNet: A multilingual database with lexical semantic networks, pages 1–17. Springer.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A dutch bert model. arXiv preprint arXiv:1912.09582.

Wiebke Wagner, Helmut Schmid, and S. Schulte Im Walde. 2009. Verb sense disambiguation using a predicate-argument-clustering model. In Proceedings of the CogSci Workshop on Distributional Semantics beyond Concrete Concepts, pages 23–28.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461.

Warren Weaver. 1955. Translation. Machine translation of languages, 14:15–23. Publisher: Cambridge: Technology Press, MIT.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In Advances in neural information processing systems.

David Yarowsky. 2000. Word Sense Disambiguation. In The Handbook of Natural Language Processing.

Seid Muhie Yimam, Chris Biemann, Richard Eckart de Castilho, and Iryna Gurevych. 2014. Automatic annotation suggestions and custom annotation layers in WebAnno. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 91–96.

Dayu Yuan, Julian Richardson, Ryan Doherty, Colin Evans, and Eric Altendorf. 2016. Semi-supervised word sense disambiguation with neural models. arXiv preprint arXiv:1603.07012.

Zhi Zhong and Hwee Tou Ng. 2009. Word sense disambiguation for all words without hard labor. In Twenty-First International Joint Conference on Artificial Intelligence.

Zhi Zhong and Hwee Tou Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In Proceedings of the ACL 2010 system demonstrations, pages 78–83.