Alignement des embeddings des définitions et du contexte pour un assistant de lecture sensible au contexte

Ioana Ivan Nathan Chometton
Aix-Marseille Université, CNRS, LIS, Marseille, France
ioana.ivan@etu.univ-amu.fr, chometton.n@gmail.com

RÉSUMÉ
La désambiguïsation lexicale (word sense disambiguation ou WSD) est une tâche du traitement automatique de la langue qui vise à identifier, à partir d'un mot, de son contexte d'occurrence et d'une liste de sens possibles, le sens du mot le plus adapté. Cette tâche pourrait permettre le développement de liseuses plus sophistiquées que les liseuses actuelles en présentant, lorsqu'on clique sur un mot pour en connaître le sens, le sens le plus probable étant donné le contexte. Nous allons dans cette étude proposer quelques pistes pour aborder ce problème.

ABSTRACT
Matching contextual and definitional embeddings for a sense-aware reading assistant.

WSD is a branch of natural language processing which aims to identify, by means of the context of occurence and a list of possible senses, the most suitable sense of the word. This task could enable the development of more sophisticated e-readers which, while clicking on a word to find out its meaning, would present the reader with the most probable meaning given the context. In this study, we will propose some pointers to tackle this problem.

MOTS-CLÉS : Wiktionnaire, désambiguïsation lexicale, word2vec, fastText, flauBERT.

KEYWORDS: Wiktionary, WSD, word2vec, fastText, flauBERT.

1 Introduction

La désambiguïsation lexicale est une branche du domaine du traitement automatique de la langue qui vise a identifier, à partir du contexte d'apparition d'un mot et d'une liste des sens possibles, le sens du mot le plus adapté. Dans cet étude nous utilisons des méthodes sans apprentissage, fondées sur l'hypothèse qu'il existe une similarité entre le contexte d'apparition d'un mot et sa définition. L'implémentation de cette idée remonte à l'algorithme introduit par Michael Lesk en 1986, dont une version simplifiée consistait à compter le nombre de mots en commun entre le contexte et la définition d'un mot. Mais elle constitue aussi une source d'inspiration pour des approches modernes en WSD, comme GlossBERT (Huang *et al.*, 2019), un modèle neuronal qui apprend si une définition correspond ou pas à un contexte donné.

La tâche de désambiguïsation suppose que l'on dispose d'un corpus dans lequel les mots sont associés à leur sens en contexte. De tels corpus sont peu courants car très chers à développer. Une manière de contourner le problème consiste à utiliser un dictionnaire donnant pour chaque entrée une définition et pour chaque définition un ou plusieurs exemples. Ces exemples comportent une occurrence de l'entrée

qui est désambiguïsée (son sens est la définition sous laquelle l'exemple apparaît). L'ensemble des exemples constitue alors un corpus où chaque exemple comporte un mot désambiguïsé. Nous avons suivi Segonne *et al.* (2019) et avons utilisé Wiktionary. D'après les auteurs, cette ressource, malgré son caractère dynamique et collaboratif, permet d'obtenir de meilleures performances, parmi les différentes ressources existant en français, pour la tâche de désambiguïsation.

2 Le jeu de données

Notre jeu de données est obtenu à partir de l'extrait ontolex Dbnary du 29-05-2022 du Wiktionnaire français (Sérasset, 2015) [1]. Le fichier Dbnary a été traité pour obtenir un jeu de données organisé en 6 colonnes : lemme, définition, exemple, index du lemme dans l'exemple, partie du discours du lemme, registre(s). Dans ce jeu de données, chaque ligne correspond à un exemple, comme illustré dans le Tableau 1.

Mot	Définition	Exemple	Index	POS	Registres
pile	Précisément	Il est minuit pile	3	adv	Familier|Populaire

TABLE 1 – Exemple d'une entrée du jeu de données.

Les exemples correspondant aux mots monosémiques (qui ont un seul sens/définition dans le Wiktionary) ont été enlevés du corpus car ce cas de figure ne nous intéresse pas. En effet, au vu d'une application concrète d'aide à la lecture sur liseuse, ces mots ne posent aucun problème et leur traitement est trivial. Nous avons également retiré les mots outils ainsi que les lemmes associés à des définitions ne contenant pas d'exemples. Le jeu de données final se compose donc de **34 026 lemmes**, **102 138 définitions** et **188 000 exemples**.

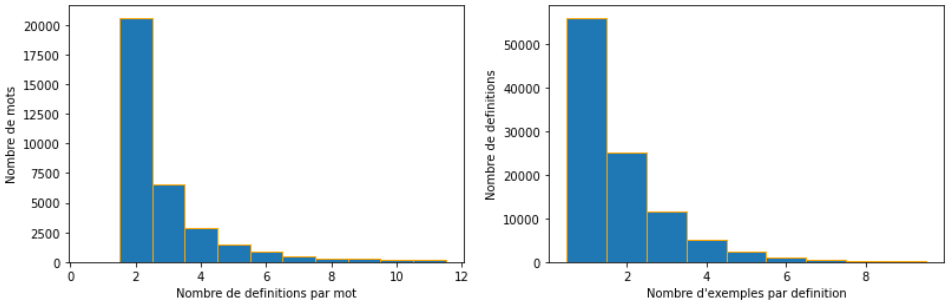

FIGURE 1 – **A gauche** : Histogramme de polysémie (nombre de définitions par lemme), **A droite** : Histogramme du nombre d'exemples par définition

Le nombre moyen de définitions par lemme est de **3** (Figure 1). Remarquons, pour commencer, que la plupart des lemmes ont 2 définitions, même si un certain nombre (764) compte plus de 10 définitions (allant jusqu'à 60 définitions différentes pour un même lemme). En ce qui concerne le nombre d'exemples par définition la moyenne est de **1,84**.

1. http://kaiko.getalp.org/about-dbnary/download/

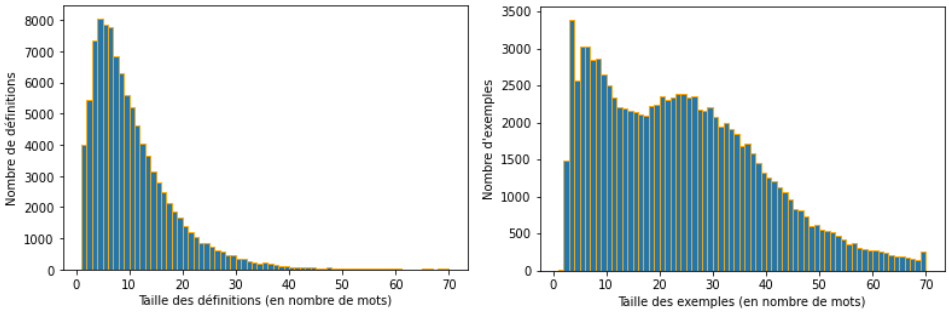

FIGURE 2 – **A gauche** : Histogramme du nombre de mots par définition, **A droite** : Histogramme du nombre de mots par exemple.

La longueur des exemples est assez élevée, avec une moyenne de **24 mots par exemple** (Figure 2). Ceci est probablement dû aux citations assez nombreuses dans notre base de données. La taille des définitions est plus courte avec une moyenne de **10 mots par définition**. On remarque qu'on compte très peu de définitions longues (15 mots ou plus), mais on compte plus de **30 000 définitions courtes** (5 mots ou moins).

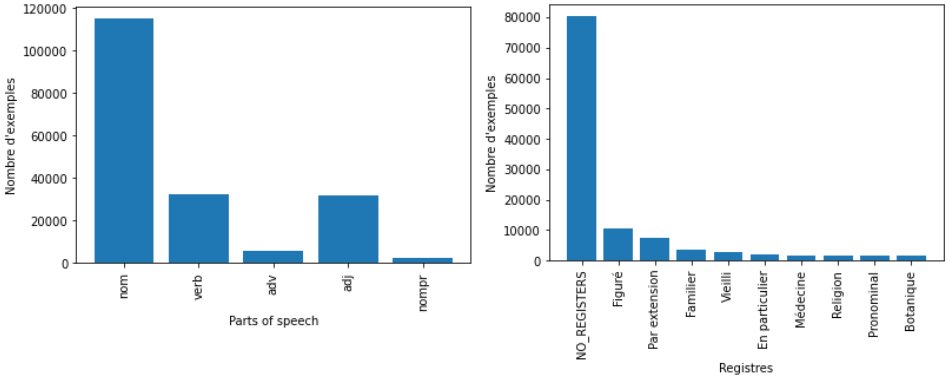

FIGURE 3 – **A gauche** : Histogramme des parties de discours, **A droite** : Histogramme des registres

Concernant les parties de discours, on constate une prédominance assez attendue des noms qui correspondent à **60%** (115 079) des exemples du corpus (Figure 3). Les adjectifs et les verbes représentent quant à eux autour de **30%** (64 726) des exemples. En ce qui concerne les registres, on observe que 42% des exemples ont au moins un registre associé. Les plus fréquents sont : "figuré", "par extension", "familier" et "vieilli".

3 Expériences et résultats

Pour réaliser la désambiguïsation, nous avons utilisé des méthodes sans apprentissage, fondées sur la similarité cosinus entre une représentation vectorielle du contexte d'occurrence d'un mot et la représentation vectorielle de sa définition. Notre modèle *baseline* consiste simplement à additionner les embeddings des mots de la définition d'une part et ceux du contexte d'autre part, et comparer les deux embeddings résultants.

Nous avons réalisé les expériences sur un jeu de données test constitué de 18 800 exemples, soit 10% du jeu de données. Les exemples du jeu de test sont prises aléatoirement dans le jeu de données complet. Nous gardons 90% des données pour pouvoir appliquer des méthodes fondées sur l'apprentissage à l'avenir.

Nous avons commencé en utilisant des embeddings **fastText** de taille 300 fournis par Grave *et al.* (2018). En appliquant notre modèle baseline, nous avons obtenu un score de **35.69%**. Étant donné la faiblesse de ce résultats, nous l'avons comparé au choix aléatoire. Pour déterminer le score du choix aléatoire, nous avons choisi au hasard pour chaque exemple une des définitions du mot-cible et nous avons calculé l'exactitude. Nous avons obtenu une exactitude de **32.59 %**.

Considérant que notre baseline améliorait très faiblement les résultats du choix aléatoire, nous avons tenté de l'améliorer en donnant plus de poids à certains mots dans la phrase. Dans ce but, nous avons conçu deux modèles, basés sur les hypothèses suivantes : i) dans une définition, les mots qui aident le plus à différencier le sens sont les premiers mots de la phrase (il s'agit souvent d'hyperonymes) et ii) dans un exemple, ce sont les mots positionnés le plus proche du mot-cible qui aident le plus à distinguer le sens.

Les détails des deux modèles sont donnés ci-dessous :

1. **Modèle début de la définition.** Le poids du mot dépend de la longueur de la phrase et décroît linéairement avec la position dans la phrase, pondérée par un facteur α.

$$w_1(i) = n - i\alpha,$$

où $n \in \mathbb{N}^*$ est le nombre de mots de la phrase, $\alpha \in (0, 1]$ est une constante fixée et $i \in [0, n)$ est la position du mot dans la phrase.

2. **Modèle contexte dans l'exemple.** Les voisins du mot cible ont un poids égal à la taille de la phrase et le poids décroît linéairement vers les deux extrémités de la phrase.

$$w_2(i) = n + 1 - \alpha * |i_c - i|,$$

où $n \in \mathbb{N}^*$ est le nombre de mots de la phrase, $\alpha \in [0, 1)$ est une constante fixée, i_c est la position du mot cible et $i \in [0, n)$ est l'index du mot

Modèle	Exactitude
Aléatoire	32.59%
Baseline	**35.69%**
Début déf. ($\alpha = 0.1$)	35.65%
Contexte ex. ($\alpha = 0.3$)	**35.79%**

TABLE 2 – Exactitude des modèles avec des embeddings fastText

En étudiant le tableau 2 qui présente les résultats, nous remarquons une amélioration très faible (0.1%) lors de la pondération, mais globalement les scores sont très proches du modèle baseline. L'exactitude ne semble pas s'améliorer significativement en changeant les poids en accord avec ces hypothèses. En effet, ni les exemples ni les définitions ne semblent pas suivre une des hypothèses que nous avons formulé.

Au vu de ces résultats, nous avons décidé de garder le même modèle baseline et de changer de type d'embedding. Nous avons choisi deux autres types d'embeddings : des embeddings non contextuels provenant de word2vec, un précurseur de fastText, et des embeddings contextuels provenant de FlauBERT.

Dans le cas de word2vec, nous avons considéré plusieurs modèles pré-entraînés sur le corpus frWac2Vec fournis par Fauconnier (2015). Nous avons choisi celui qui maximise l'écart entre la similarité cosinus entre les exemples avec leurs définitions associées d'un coté, et la similarité des exemples avec une définition du mot tirée au hasard d'un autre coté. Le meilleur modèle de ce point de vue a été le modèle cbow avec des vecteurs de taille 200.

Dans le cas de FlauBERT, nous avons utilisé les embeddings de taille 786 fournis par Le *et al.* (2020). Pour les définitions, ce sont les embeddings CLS qui ont été utilisés, et pour les exemples, les embeddings contextuels générés pour les tokens dans la position du mot cible.

Une comparaison des résultats du modèle baseline avec les trois types d'embeddings est montrée dans le tableau 3. Les embeddings word2vec ont la meilleure exactitude et dépassent considérablement leur compétiteurs. Les résultats de FlauBERT sont surprenants, car ils se situent bien en deçà de word2vec et sont très proches du choix aléatoire. Cela pourrait indiquer que les embeddings CLS et les embeddings contextuels ne sont pas proches du point de vue de la similarité cosinus et qu'une autre façon d'utiliser les embeddings FlauBERT obtiendrait des résultats plus satisfaisants.

Modèle	Exactitude
Aléatoire	32.59%
FlauBERT	33.06%
fastText	35.69%
word2vec	**48.24%**

TABLE 3 – Exactitude du baseline avec des embeddings différents

4 Analyse

Pour mieux comprendre nos résultats, nous avons regardé la performance du meilleur modèle : celui basé sur les embeddings word2vec, sur les parties de discours et les registres des mots. Nous avons remarqué que les noms propres ont la meilleure exactitude - **62%**, alors que les adjectifs et les verbes sont les plus difficiles à identifier, avec une exactitude autour de **44%** (Tableau 4).

De manière similaire, nous avons regardé la performance du modèle par rapport au registre (tableau 5) : les mots sans registre ont la meilleure exactitude - **45%**. Parmi les registres les plus représentés, le registre *Par extension* semble être un des plus difficiles à identifier.

Embedding	Noms	Adjectifs	Verbes	Adverbes	Noms propres
Aléatoire	32.38 %	35.27 %	29.12 %	34.50 %	47.77 %
word2vec	50.22 %	44.03 %	44.88 %	45.22 %	**61.94 %**

TABLE 4 – Exactitude part of speech

Embedding	Pas de registre	Figuré	Par extension	Familier
Aléatoire	32.33 %	29.32%	26.78%	34.29%
word2vec	45.07 %	45.17%	41.88%	43.51%

TABLE 5 – Exactitude registres

5 Conclusions

Les pistes d'amélioration et d'analyse sont nombreuses. On aimerait, entre autres, déterminer si le corpus a une granularité adéquate en terme de nombre de définitions par mot et comprendre pourquoi les embeddings flauBERT n'obtiennent pas de bons résultats. Étendre nos méthodes à des réseaux de neurones en bénéficiant de la supervision du Wiktionary est aussi une des pistes envisagées.

Remerciements

Nous remercions nos professeurs, Alexis Nasr et Carlos Ramisch pour leur aide et leur bienveillance.

Références

FAUCONNIER J.-P. (2015). French word embeddings.

GRAVE E., BOJANOWSKI P., GUPTA P., JOULIN A. & MIKOLOV T. (2018). Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

HUANG L., SUN C., QIU X. & HUANG X. (2019). GlossBERT : BERT for word sense disambiguation with gloss knowledge. 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)*, p. 3509–3514, Hong Kong, China : Association for Computational Linguistics. DOI : 10.18653/v1/D19-1355.

LE H., VIAL L., FREJ J., SEGONNE V., COAVOUX M., LECOUTEUX B., ALLAUZEN A., CRABBÉ B., BESACIER L. & SCHWAB D. (2020). Flaubert : Unsupervised language model pre-training for french. In *Proceedings of The 12th Language Resources and Evaluation Conference*, p. 2479–2490, Marseille, France : European Language Resources Association.

SEGONNE V., CANDITO M. & CRABBÉ B. (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*, p. 259–270, Gothenburg, Sweden : Association for Computational Linguistics. DOI : 10.18653/v1/W19-0422.

SÉRASSET G. (2015). DBnary : Wiktionary as a Lemon-Based Multilingual Lexical Resource in RDF. *Semantic Web – Interoperability, Usability, Applicability*, **6**(4), 355–361. DOI : 10.3233/SW-140147, HAL : hal-00953638.