# Évaluation et Production de Plongements de Mots à Partir de Contenus Web Français à Grande Échelle

H Abdine, C Xypolopoulos, M Kamal Eddine, M Vazirgiannis

# Évaluation et Production de Plongements de Mots à Partir de Contenus Web Français à Grande Échelle

H. Abdine[1], C. Xypolopoulos[1], M. Kamal Eddine[1], M. Vazirgiannis[1,2]

[1] École Polytechnique, LIX
[2] Athens University of Economics and Business, AUEB

{hadi.abdine, christos.xypolopoulos, moussa.kamal-eddine, michalis.vazirgiannis}@polytechnique.edu

## Résumé

*Les représentations distribuées de mots sont couramment utilisées dans de nombreuses tâches de traitement du langage naturel, en ajoutant que les vecteurs de mots pré-entraînés sur d'énormes corpus de textes ont atteint une performance importante dans de nombreuses tâches NLP. Cet article présente plusieurs vecteurs de mots de qualité supérieure pour la langue française, deux d'entre eux étant entraînés sur des données françaises massives recueillies au cours de cette étude et les autres sur un corpus français déjà existant utilisé pour pré-entraîner le model FlauBERT. Nous évaluons également la qualité de nos vecteurs de mots proposés et des vecteurs de mots français existants sur la tâche d'analogie de mots français. De plus, nous effectuons l'évaluation sur de multiples tâches NLU qui montrent l'amélioration importante des performances des vecteurs de mots pré-entraînés durant cette étude par rapport aux plongements existants et aléatoires. Enfin, nous avons créé une application web pour tester et visualiser les plongements de mots obtenus [1]. Ceux derniers sont disponibles au public, ainsi que le code d'affinage sur les tâches NLU [2]. et le code de la démonstration [3].*

## Mots-clés

*Plongements de mots, Word2Vec, pré-entraînement, apprentissage profond, NLP, FLUE, analogie de mots.*

## Abstract

*Distributed word representations are popularly used in many tasks in natural language processing, adding that pre-trained word vectors on huge text corpus achieved high performance in many different NLP tasks. This paper introduces multiple high-quality word vectors for the French language where two of them are trained on massive crawled French data and the others are trained on an already existing French corpus used to pretrain FlauBERT model. We also evaluate the quality of our proposed word vectors and the existing French word vectors on the French word analogy task. In addition, we do the evaluation on multiple NLU tasks that shows the important performance enhancement of the pre-trained word vectors during this study compared to the existing and random ones. Finally, we created a demo web application to test and visualize the obtained word embeddings[1]. The produced French word embeddings are available to the public, along with the fine-tuning code on the NLU tasks[2]. and the demo code[3].*

## Keywords

*Word embeddings, Word2Vec, pretraining, deep learning, NLP, FLUE, word analogy.*

## 1 Introduction

Les plongements de mots sont un type de représentation des mots qui est devenu très important et largement utilisé dans différentes applications du traitement du langage naturel. Par exemple, les plongements de mots pré-entraînés ont joué un rôle essentiel dans l'obtention de performances remarquables avec des modèles d'apprentissage profond sur des problèmes difficiles de compréhension du langage naturel. Ces vecteurs de mots sont produits à l'aide des modèles de réseaux de neurones non supervisés basés sur l'idée que la signification d'un mot est liée au contexte dans lequel il apparaît. Ainsi, des mots similaires en signification ont une représentation similaire. Plus précisément, ils ont une représentation proche dans l'espace. En conséquence, la qualité des vecteurs de mots est directement liée à la quantité et à la clarté du corpus sur lequel ils ont été entraînés. De nombreuses techniques et algorithmes ont été proposés depuis 2013 pour apprendre ces représentations distribuées de mots. Nous nous concentrons principalement sur Word2Vec [14], GloVe [18], et FastText [3] qui utilisent deux approches : CBoW et SkipGram. Ces méthodes ont été utilisées pour apprendre des vecteurs de mots à partir d'énormes corpus. La plupart des vecteurs de mots disponibles publiquement sont pré-entraînés sur des textes en anglais. Cependant, les vecteurs de mots pour les autres langues sont peu nombreux, inexistants ou formés à l'aide d'un très petit corpus qui ne peut produire des vecteurs de mots de bonne qualité. Dans cet article, nous nous intéressons à la création des plongements de mots français statiques avec un benchmark qui les compare avec d'autres plongements de mots. Ainsi, nous n'incluons pas les plongements de mots contextuels tels que FlaubBERT [12], CamemBERT [13] et BARThez [7] dans la comparaison. Ces modèles pré-entraînés basés sur

1. nlp.polytechnique.fr/
2. github.com/hadi-abdine/WordplongementsEvalFLUE/
3. github.com/hadi-abdine/FrenchWordplongementsDemo/

l'attention, même s'ils ont de meilleures performances, utilisent plus de mémoire et de puissance de calcul. Ce travail présente des plongements de mots français entraînés sur un large corpus français collecté du web avec plus d'un million de noms de domaine. De plus, nous entraînons les plongements Word2Vec sur d'autres corpus et ressources françaises afin de les comparer avec ceux entraînés sur le corpus français collecté. Ensuite, nous évaluons ces vecteurs de mots sur l'ensemble de questions crées dans [9] pour la tâche d'analogie de mots. Enfin, nous évaluons les vecteurs de mots français sur certaines tâches du benchmark FLUE [12] et nous comparons les résultats avec ceux des vecteurs français FastText [9] et Word2Vec [8] existants produits à partir des corpus Common Crawl, Wikipedia et FrWac. Nous discutons par la suite la signification de la tâche d'analogie des mots sur la qualité de leurs plongements.

## 2 Travaux antérieurs

Depuis que les représentations distribuées des mots ont été introduites, de nombreux vecteurs de mots pré-entraînés pour de nombreuses langues sont produits. Par exemple, des vecteurs de mots anglais ont été appris à l'aide d'une partie de l'ensemble de données de Google News et publiés avec Word2Vec [14]. Ensuite, des représentations de mots ont été entraînées pour 157 langues, dont la française, en utilisant FastText [9] pour apprendre des vecteurs de mots sur le corpus Common Crawl et Wikipedia. En 2015, les auteurs de [8] ont pré-entraîné plusieurs modèles Word2Vec pour la langue française en utilisant le corpus FrWac [1] et le corpus FrWiki Dump (données françaises issues de Wikipedia).
Depuis l'apparition de Word2Vec, l'évaluation des vecteurs de mots était basée sur la tâche d'analogie des mots introduite dans [14]. Cette tâche évalue la relation linéaire entre les vecteurs de mots pour vérifier la qualité de leurs plongements. Suivant cette idée, une liste de questions d'analogie en français pour évaluer les représentations de mots est présentée dans l'article [9]. Enfin, dans [12], les auteurs ont présenté un benchmark général pour évaluer les systèmes de NLP français nommé FLUE contenant de diverses tâches pour évaluer les modèles de compréhension naturelle français (NLU).
Notre objectif principal est d'évaluer la qualité des plongements de mots statiques pour la langue française et de fournir de nouveaux plongements entraînés sur un grand ensemble de données diversifiées qui sont performants dans les tâches d'analogie de mots, ainsi que dans les tâches de compréhension de la langue. De nombreux articles montrent que les modèles basés sur l'attention pré-entraînée comme BERT [6] et ELMo [19] surpassent les plongements de mots statiques. Mais cette amélioration des performances par rapport aux plongements statiques s'est faite au prix d'une utilisation moins efficace des ressources informatiques et de temps d'apprentissage et d'inférence plus longs, ainsi que d'une moindre interprétabilité et d'une plus grande dégradation de l'environnement. Même s'ils ne sont pas aussi expressifs ou puissants que les modèles de plongements contextuels, ils restent utiles dans la recherche sur le traitement du langage naturel [10].

## 3 Crawling du web français

Pour entraîner les plongements de mots, nous avons décidé de collecter un énorme corpus brut français de multiples domaines sur le web en utilisant un crawler qui parcourt en permanence, de façon autonome et automatique, les différents sites et pages et sauvegarde leur contenu.

**Graines.** Nos graines initiales ont été sélectionnées en trouvant les pages Web les plus populaires sous le domaine ".fr". Pour ce faire, nous avons utilisé une liste publique [4] qui contient les noms de domaine de sites web de différents genres tels que les sites d'information, Wikipedia et les médias sociaux. La frontière a ensuite été mise à jour avec les nouveaux liens découverts pendant l'exploration.

**Crawler.** Pour le crawling, nous avons choisi Heritrix [5], un crawler web open-source supporté par Internet Archive. Notre configuration est constituée d'un seul nœud avec 25 threads crawlant pendant une période de 1,5 mois.

**Sortie.** La sortie générée par Heritrix suit le format de fichier WARC, avec des fichiers fractionnés à 1 Go chacun. Ce format a été traditionnellement utilisé pour enregistrer les "web crawls" sous forme de séquences de blocs de contenu recueillis sur le World Wide Web. Les blocs de contenu d'un fichier WARC peuvent contenir des ressources dans n'importe quel format, par exemple des images binaires ou des fichiers audiovisuels qui peuvent être intégrés ou liés à des pages HTML.

**Extraction.** Pour extraire le texte intégré dans la sortie du Heritrix, nous avons utilisé un outil warc-extractor. [6]. Cet outil est utilisé pour analyser les enregistrements avec le WARC et ensuite analyser les pages HTML. Au cours de cette étape, nous avons également intégré le module de détection de langue FastText [7] qui nous permet de filtrer le texte HTML par langue.

**Dédoublonnage.** Pour éliminer les données redondantes dans le corpus, nous avons utilisé l'outil de déduplication d'Isaac Whitfield [8], le même que celui utilisé sur le corpus Common Crawl [16] basé sur un algorithme de hachage très rapide et résistant aux collisions. Le corpus final après déduplication a une taille totale de 33 Go, (171,701,319) lignes, (5,073,407,023) mots et (12,464,568) unigrammes uniques.

**Ethique.** Heritrix est conçu pour respecter le fichier robots.txt (fichier écrit par les propriétaires de sites Web pour donner des instructions sur leur site aux robots d'exploration), les directives d'exclusion et les balises de META nofollow.

4. Sites web les plus populaires avec le domaine .fr
5. https://github.com/internetarchive/heritrix3
6. https://github.com/alexeygrigorev/warc-extractor
7. https://github.com/vinhkhuc/JFastText
8. https://github.com/whitfin/runiq

| Plongements | # Vocab | Outil | Méthode | Corpus | Dimension |
|---|---|---|---|---|---|
| fr_w2v_web_w5 | 0.8M | Word2Vec | cbow | fr_web | 300 |
| fr_w2v_web_w20 | 4.4M | Word2Vec | cbow | fr_web | 300 |
| fr_w2v_fl_w5 | 1M | Word2Vec | cbow | flaubert_corpus | 300 |
| fr_w2v_fl_w20 | 6M | Word2Vec | cbow | flaubert_corpus | 300 |
| cc.fr.300 | 2M | FastText | skip-gram | CC+wikipedia | 300 |
| frWac_200_cbow | 3.6M | Word2Vec | cbow | frWac | 200 |
| frWac_500_cbow | 1M | Word2Vec | cbow | frWac | 500 |
| frWac_700_sg | 184K | Word2Vec | skip-gram | frWac | 700 |
| frWiki_1000_cbow | 66K | Word2Vec | cbow | wikipedia dump | 1000 |

TABLE 1 – Résumé des modèles utilisés dans nos expériences

# 4 Apprentissage des plongements de mots

Afin d'apprendre les plongements de mots français, nous avons utilisé Word2Vec de Gensim pour produire quatre modèles CBoW (Continuous Bag of Words) de vecteurs de mots. Deux de ces modèles ont été entraînés sur une portion de 33 Go du corpus français utilisé pour pré-entraîner FlauBERT[12]. Ce corpus est composé de 24 sous-corpus collectés à partir de différentes sources, avec des sujets et des styles d'écriture variés. Le premier modèle Word2Vec est noté fr_w2v_fl_w5 entraîné en utilisant Word2Vec CBoW avec une taille de fenêtre de 5 et une fréquence de coupure de 60. En d'autres termes, nous ne considérons la formation d'un plongement pour un mot, que s'il apparaît au moins 60 fois dans le corpus. Le second est fr_w2v_fl_w20 formé en utilisant Word2Vec CBoW avec une taille de fenêtre de 20 et une fréquence de coupure de 5. Les deux autres modèles ont été entraînés sur le corpus français dédupliqué de 33 Go collecté sur le Web (section 3). Le premier est noté fr_w2v_web_w5 entraîné sur Word2Vec CBoW encore avec une taille de fenêtre de 5 et une fréquence de coupure de 60. Le second est fr_w2v_web_w20 entraîné sur Word2Vec CBoW avec une taille de fenêtre de 20 et une fréquence de coupure de 5. Tous les modèles examinés dans cet article ont un dimension de plongement de 300. Le tableau 1 contient les détails de chaque plongement de mots utilisée dans cette étude : cc.fr.300 représente les vecteurs de mots Fasttext français [9] de dimension 300 formés sur le corpus Common Crawl [16], les modèles frWac_200_cbow, frWac_500_cbow et frWac_700_sg [8] représentent les vecteurs Word2Vec français de dimensions 200, 500 et 700 respectivement qui sont entraînés sur le corpus frWac [2] (un corpus de 1,6 milliards de mots construit à partir du Web en limitant le crawl au domaine .fr) et finalement le modèle frWiki_1000_cbow [8] qui représente les vecteurs Word2Vec de dimension 1000 entraînés sur la partie française du jeu de données frWiki dump (une copie complète de tous les wikis de Wikimedia écrits en français).

# 5 Évaluation avec l'analogie des mots

Dans ce travail, la première méthode d'évaluation des vecteurs de mots est la tâche d'analogie [14]. Dans cette tâche, étant donné trois mots, A, B et C, nous pouvons prédire un mot D tel que la relation entre A et B est la même entre C et D, en supposant qu'une relation linéaire entre les vecteurs de paires de mots indique la qualité de leurs plongements. Par exemple, si la relation entre les représentations des mots **king** et **man** est similaire à la relation entre les représentations de **queen** et **woman**, cela impliquera de bonnes plongements de mots. Par exemple, dans cette évaluation, nous utilisons les vecteurs de mots $x_A$, $x_B$ et $x_C$ de trois mots A, B et C pour calculer le vecteur $x_B - x_A + x_C$, et le vecteur le plus proche dans le dictionnaire du vecteur résultant sera considéré comme le vecteur du mot D. La performance des vecteurs de mots est finalement calculée en utilisant la précision moyenne sur le jeu de données de test. Pour évaluer nos plongements de mots français présentées dans la section 3, nous utilisons le jeu de données d'analogie française créé dans [9]. Ce dernier contient (31,688) questions d'analogie. Nous comparons également les résultats de nos vecteurs de mots avec ceux du FastText français (cc.fr.300), également produit dans [9] et les vecteurs de mots français entraînés sur frWacky et frWiki.

| Plongements | Accuracy |
|---|---|
| fr_w2v_web_w5 | 41.95% |
| fr_w2v_web_w20 | 52.50% |
| fr_w2v_fl_w5 | 43.02% |
| fr_w2v_fl_w20 | 45.88% |
| cc.fr.300 | 63.91% |
| frWac_500_cbow | **67.98%** |
| frWac_200_cbow | 54.45% |
| frWac_700_sg | 55.52% |
| frWiki_1000_cbow | 0.87% |

TABLE 2 – Précision sur la tâche d'analogie

**Résultats.** Le tableau 2 montre la précision de la tâche d'analogie pour tous les modèles. Les résultats indiquent que les vecteurs de mots FastText et les plongements de mots entraînés sur frWacky français sont meilleurs que les vecteurs de mots CBoW Word2Vec produits au cours de cette étude dans la tâche d'analogie. Cependant, les auteurs du [11] ont prouvé que la propriété de relation géométrique par rapport aux analogies ne tient pas en général et qu'elle est probablement fortuite plutôt que systématique. En conclusion, nous avons décidé d'évaluer les vecteurs de mots sur des tâches de compréhension du langage naturel (NLU) telles que la

classification de textes, la paraphrase et l'inférence linguistique qui sont présentées dans [12] au lieu de s'appuyer uniquement sur l'analogie de mots comme d'autres études de plongements de mots statiques.

**Application web.** Pour visualiser et examiner nos plongements de mots, nous avons créé une application web qui contient les outils suivants :

1. Examen de l'analogie des mots : il prend en entrée trois mots, A, B et C, et il calcule les dix vecteurs les plus proches de $x_B - x_A + x_C$, puis affiche les mots correspondants comme nous pouvons le voir sur la figure 1.
2. Un calculateur de similarité en cosinus : il prend deux mots et calcule la similarité en cosinus entre les vecteurs correspondants.
3. Un outil de mots similaires : il trouve les dix mots les similaires à un mot d'entrée.
4. Un outil de visualisation : en utilisant T-SNE et k-means, cet outil affiche dans un espace vectoriel 2-D les **n** mots les plus proches du mot **w** distribués en **k** clusters, où n, w et k sont choisis par l'utilisateur. Par exemple, dans la figure 2, nous voyons une partie du graphe pour n=200, w=paris et k=8.

# 6 Évaluation sur les tâches de FLUE

FLUE est un benchmark d'évaluation de la compréhension de la langue française [12], (French Language Understanding Evaluation), créé pour évaluer les performances des modèles NLP français. Il contient de nombreux jeux de données qui varient en termes de sujets, de niveau de difficulté, de taille et de degré de formalité. Cette section présente les résultats d'affinage de nos différents plongements Word2Vec et les compare avec les résultats d'affinage des autres plongements de mots mentionés dans le tableau 1 sur certaines tâches de FLUE. Nous incluons également un plongement sans pré-entraînement (initialisation aléatoire de la couche de plongements des mots) pour la comparaison.

## 6.1 Classification des textes

**Description des données.** Dans cette tâche, le jeu de données utilisé est la partie française de CLS (Cross Lingual Sentiment) [20] qui se compose d'avis Amazon divisés en trois sous-ensembles : livres, DVD et musique.
Suivant [12], chaque sous-ensemble est binarisé en considérant tous les avis supérieurs à trois étoiles comme positifs, ceux inférieurs à trois étoiles comme négatifs, et en éliminant ceux qui ont trois étoiles. En outre, chaque sous-ensemble contient un ensemble équilibré de formation et de test contenant environ 1000 critiques positives et 1000 critiques négatives chacun. Enfin, un jeu de données de validation est formé pour chaque sous-ensemble en prenant une division aléatoire de 20 % des données d'entraînement.

**Description du modèle.** Le modèle utilise une couche de plongements de mots dont les poids initiaux et la taille des plongements varient en fonction de plongements de mots évalués. Cette couche est suivie d'un Bi-LSTM à deux couches de 1500D (par direction). Enfin, nous utilisons la tête de classification utilisée dans [6] qui est composée des couches suivantes : une couche de dropout avec un taux de dropout de 0.1, une couche linéaire, une couche d'activation à tangente hyperbolique, une seconde couche de dropout avec le même taux de dropout et une seconde couche linéaire avec une dimension de sortie de deux (identique au nombre de classes).
Nous entraînons le modèle pendant 30 époques pour les différents plongements de mots tout en effectuant une recherche par quadrillage sur trois taux d'apprentissage différents : 5e-5, 2e-5 et 5e-6. Le modèle le plus performant sur le jeu de données de validation est choisi pour l'évaluation sur le jeu de données de test.

| Plongements | Livres | Musique | DVD |
|---|---|---|---|
| w2v_0 | 67.20% | 67.20% | 62.15% |
| fr_w2v_web_w5 | 79.38% | 79.49% | 80.29% |
| fr_w2v_web_w20 | 81.55% | 79.75% | 80.75% |
| fr_w2v_fl_w5 | 82.16% | 81.00% | 79.64% |
| fr_w2v_fl_w20 | **82.38%** | **82.58%** | **82.43%** |
| cc.fr.300 | 75.30% | 74.35% | 72.43% |
| frWac_500_cbow | 81.30% | 80.80% | 79.59% |
| frWac_200_cbow | 79.65% | 75.40% | 80.75% |
| frWac_700_sg | 77.55% | 75.20% | 78.43% |
| frWiki_1000_cbow | 66.30% | 70.05% | 60.94% |

TABLE 3 – Précision sur le CLS français

**Résultats.** Le tableau 3 présente la précision sur le jeu de données de test pour chaque plongement de mots : w2v_0 représente les plongements des mots aléatoires. Les résultats démontrent l'importance des modèles pré-entraînés. Comme nous le voyons, tous les plongements de mots pré-entraînés surpassent de loin les plongements non pré-entraînés (aléatoires) avec une différence qui peut aller jusqu'à plus de 20% en termes de précision. En outre, nous voyons clairement une contradiction dans les résultats entre la tâche d'analogie et l'analyse des sentiments. Autrement dit, une meilleure performance sur la tâche d'analogie ne signifie pas une meilleure performance sur la tâche NLU.

## 6.2 Paraphrase

**Description des données.** Cette tâche utilise la partie française de PAWS-X [22] qui étend le jeu de données adversariales multilingues pour l'identification de paraphrases. Cette tâche vise à identifier si une paire de phrases avec des variations dans le choix des mots et la grammaire ont le même sens ou non. Le jeu de données est obtenu à partir de la traduction de paires de phrases en anglais provenant de Wikipédia et de Quora, avec un taux de chevauchement lexical élevé et jugé par des humains. Le jeu de données utilisé contient (49,401) échantillons d'entraînement, (1,992) échantillons de validation et (1,985) échantillons de test.

**Description du modèle.** Pour affiner les vecteurs de mots sur PAWS-X, nous utilisons le modèle d'inférence séquentielle amélioré (ESIM) [4]. Ce modèle est formé de trois composants essentiels : l'encodage des entrées, la modélisa-


FIGURE 1 – Exemple d'analogie de mots en utilisant l'outil d'analogie de notre application web (avec fr_w2v_web_w20).

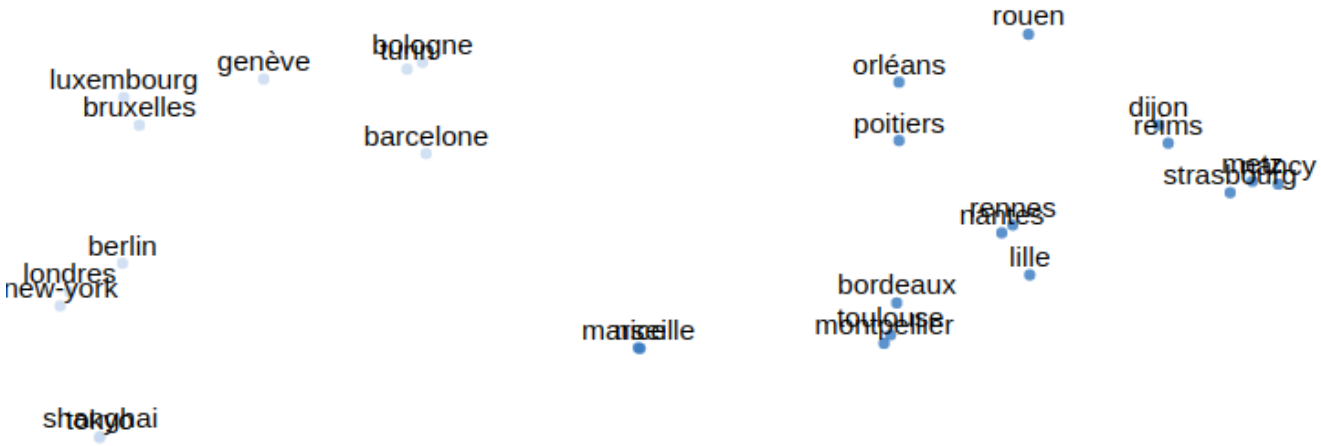


FIGURE 2 – 2 des 8 clusters de la requête : "Les 200 mots les plus proches de **paris**" en utilisant l'outil de visualisation de notre application web (avec fr_w2v_web_w20)

tion des inférences locales et la composition des inférences. Après la couche de plongements des mots, la couche d'encodage d'entrée est composée d'un BiLSTM 1500D à une couche qui encode les informations d'inférence locale des phrases A et B d'une longueur respective de $l_a$ et $l_b$ pour donner la sortie $\overline{A}$ et $\overline{B}$. Le composant suivant est le modèle d'inférence locale où :

1. Nous calculons la matrice de similarité E entre les deux phrases représentant les poids d'attention, telle que $e_{ij} = \overline{a}_i^T \overline{b}_j$.
2. Nous calculons la pertinence locale entre les deux phrases où chaque mot sera représenté comme une somme pondérée des informations pertinentes. De manière plus détaillée, $\tilde{a}_i = \sum_{j=1}^{l_b} softmax(e_{ij})\overline{b_j}$ et $\tilde{b_j} = \sum_{i=1}^{l_a} softmax(e_{ij})\overline{a_i}$.
3. Nous nous retrouvons l'amélioration de l'information d'inférence locale par une concaténation présentée par les auteurs qui prévoient d'améliorer l'inférence locale. Par exemple, les vecteurs résultants seront $m_a = [\overline{a}; \tilde{a}; \overline{a} - \tilde{a}; \overline{a} \odot \tilde{a}]$ et $m_b = [\overline{b}; \tilde{b}; \overline{b} - \tilde{b}; \overline{b} \odot \tilde{b}]$.

La quatrième couche est une couche de projection qui est composée d'une couche linéaire avec une dimension cachée de 1500D, d'une activation ReLU et d'une couche de dropout avec un taux de 0.1
Ènfin, la couche de composition d'inférence est composée d'une couche BiLSTM 1500D à 1 couche qui donne les vecteurs $V_a$ et $V_b$ en sortie. Nous avons utilisé la même tête de classification que dans la tâche de classification de texte dont l'entrée est le vecteur concaténé de «average pooling» et de «max pooling» de $V_a$ et $V_b$.

| Plongements | Accuracy |
|---|---|
| w2v_0 | 73.49% |
| fr_w2v_web_w5 | 77.12% |
| fr_w2v_web_w20 | **78.02%** |
| fr_w2v_fl_w5 | 75.2% |
| fr_w2v_fl_w20 | 74.24% |
| cc.fr.300 | 62.80% |
| frWac_500_cbow | 71.47% |
| frWac_200_cbow | 74.80% |
| frWac_700_sg | 67.39% |
| frWiki_1000_cbow | 69.76% |

TABLE 4 – Précision sur le PAWS-X français en utilisant l'ESIM

**Résultats.** La précision finale de chaque vecteur de mots est indiquée dans le tableau 4. Une fois encore, on peut constater que non seulement les résultats sont opposés entre les tâches NLU et la tâche d'analogie, mais aussi que les vecteurs de mots aléatoires surpassent les vecteurs de mots qui ont obtenu la deuxième meilleure précision dans la tâche d'analogie.

## 6.3 L'interprétation en langage naturel

**Données et description du modèle.** La tâche d'interprétation en langue naturelle (ou Natural Language Inference)

française de FLUE utilise la partie française du jeu de données XNLI [5]. Il contient des données NLI pour 15 langues. Chaque paire de phrases dans les jeux de données de test et de validation est annotée manuellement par des humains avec trois classes d'inférence : entaillement, neutre, et sans entaillement (contradiction). La partie française de jeu de données d'apprentissage de XNLI est obtenue par traduction automatique. Dans cette tâche, l'objectif est de déterminer s'il existe une relation d'implication, de contradiction ou de neutralité entre une phrase p appelée prémisse et une autre phrase h appelée hypothèse. Notez que le même échantillon peut être utilisé deux fois avec un ordre inverse entre les deux phrases avec un classe différent. Le jeu de données se compose de (92,702) échantillons d'entraînement, (2,491) échantillons de validation et (5,010) échantillons de test.
Le modèle utilisé pour affiner les vecteurs de mots dans cette tâche est ESIM avec les mêmes configurations et paramètres que ceux décrits dans la section 6.2.

| Plongements | Accuracy |
|---|---|
| w2v_0 | 61.37% |
| fr_w2v_web_w5 | 67.71% |
| fr_w2v_web_w20 | 68.27% |
| fr_w2v_fl_w5 | 69.41% |
| fr_w2v_fl_w20 | **69.57%** |
| cc.fr.300 | 64.70% |
| frWac_500_cbow | 63.82% |
| frWac_200_cbow | 63.74% |
| frWac_700_sg | 60.78% |
| frWiki_1000_cbow | 61.34% |

TABLE 5 – Précision sur le XNLI français en utilisant l'ESIM

**Résultats.** Nous rapportons les précisions finales des différents vecteurs de mots sur le jeu de données français de XNLI dans le tableau 5. Les résultats continuent de montrer les avantages des poids pré-entraînés et la surperformance des plongements de mots CBoW Word2Vec produits dans cette étude par rapport aux vecteurs de mots déjà existants.

### 6.4 Désambiguïsation du sens du nom

**Description de données** Cette tâche (NSD) est proposée par [12] pour la désambiguïsation du sens des mots (WSD) en français qui cible uniquement les noms. Elle est basée sur la partie française de la tâche WSD dans [15] pour créer le jeu de données de test composé de 306 phrases et (1,445) noms français annotés avec les clés de sens de WordNet et vérifiés manuellement. Le jeu de données d'apprentissage est obtenu en utilisant le meilleur système de traduction automatique anglais-français de l'outil fairseq [17] pour traduire en français le corpus SemCor et WordNet Glosses.
**Description du modèle** Nous avons utilisé les mêmes classifieurs que ceux présentés par [21] qui transmettent la sortie de nos vecteurs de mots dans une pile de 6 couches d'encodeurs de transformer et prédisent le sens du mot par une couche softmax à la fin. Nous avons utilisé les mêmes paramètres que dans [12].

| | Single | | |
|---|---|---|---|
| Plongements | Mean | Std | Ensemble |
| w2v_0 | 47.85% | ±1.17 | 52.87% |
| fr_w2v_web_w5 | 50.76% | ±1.4 | **53.77%** |
| fr_w2v_web_w20 | 50.28% | ±0.92 | 53.2% |
| fr_w2v_fl_w5 | 50.16% | ±1.41 | 53.36% |
| fr_w2v_fl_w20 | 50.65% | ±1.62 | 52.46% |
| cc.fr.300 | 49.28% | ±1.5 | 52.39% |

TABLE 6 – F1 score (%) sur la tâche de NSD

**Résultats.** Pour chaque modèle de plongements de mots, nous reportons dans le tableau 6 les valeurs de la moyenne et de l'écart-type des scores F1 (%) des 8 modèles individuellement et le score F1 (%) de l'ensemble des modèles en faisant la moyenne de la sortie de la couche Softmax de tous les modèles. Nous observons dans cette tâche que, même si fr_w2v_web_w5 surpasse légèrement les autres vecteurs de mots, nous avons un score F1 (%) très similaire pour tous les modèles. Nous pouvons dire que, pour cette tâche, les vecteurs de mots pré-entraînés n'améliorent pas le score final. Nous pensons que ces vecteurs ont la même nature quelque soit le contexte du mot alors que le but de la tâche est d'étudier les différentes significations d'un mot, est la raison de la similitude des scores entre ces modèles.

## 7 Conclusion

Dans ce travail, nous contribuons des plongements de mots entraînés sur des données extraites du web français et d'autres vecteurs de mots entraînés sur un ensemble de données mixtes formées à partir de sources et de sujets divers et variés en français, y compris Common Crawl et Wikipedia. Ces vecteurs de mots peuvent être utilisés comme poids initiaux pour divers modèles d'apprentissage profond, ce qui peut améliorer considérablement les performances par rapport à l'utilisation de poids aléatoires dans la couche de plongements des mots. En outre, ces plongements de mots dans des domaines généraux pourraient être pré-entraînées en plus sur des données de domaines spécifiques tels que la santé et le domaine juridique afin d'adapter les poids à un contexte approprié. De plus, nous avons utilisé un benchmark de quatre tâches NLP pour comparer la qualité de quatre plongements de mots français produites dans cette étude et de cinq autres plongements lexicaux existantes. Toutes les ressources présentées sont disponibles pour la communauté des chercheurs via notre application web. Enfin, nous avons montré que la tâche d'analogie de mots n'était pas fiable pour juger de la qualité des plongements de mots et de leur capacité à être performantes dans les tâches NLU. La raison des résultats sur la tâche d'analogie de mots pourrait faire l'objet d'une étude future.

## Remerciements

Cette recherche a été soutenue par la chaire ANR AML/HELAS (ANR-CHIA-0020-01).

# Références


[1] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The wacky wide web : A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43 :209–226, 09 2009.

[2] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The wacky wide web : A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43 :209–226, 09 2009.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5 :135–146, Dec 2017.

[4] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 1657–1668, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[5] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli : Evaluating cross-lingual sentence representations. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[7] Moussa Kamal Eddine, Antoine J. P. Tixier, and Michalis Vazirgiannis. Barthez : a skilled pretrained french sequence-to-sequence model, 2021.

[8] Jean-Philippe Fauconnier. French word embeddings, 2015.

[9] Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).

[10] Prakhar Gupta. Learning computationally efficient static word and sentence representations. Technical report, EPFL, 2021.

[11] Sammy Khalife, Leo Liberti, and Michalis Vazirgiannis. Geometry and analogies : A study and propagation method for word representations. In Carlos Martín-Vide, Matthew Purver, and Senja Pollak, editors, *Statistical Language and Speech Processing*, pages 100–111, Cham, 2019. Springer International Publishing.

[12] Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoit Crabbe, Laurent Besacier, and Didier Schwab. FlauBERT : Unsupervised Language Model Pre-training for French. In *LREC*, Marseille, France, 2020.

[13] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. CamemBERT : a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online, July 2020. Association for Computational Linguistics.

[14] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[15] Roberto Navigli, David Jurgens, and Daniele Vannella. 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, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics.

[16] Pedro Javier Ortiz Suarez, Benoit Sagot, and Laurent Romary. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. In Piotr Bański, Adrien Barbaresi, Hanno Biber, Evelyn Breiteneder, Simon Clematide, Marc Kupietz, Harald Lungen, and Caroline Iliadi, editors, *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019*, pages 9 – 16, Mannheim, 2019. Leibniz-Institut fur Deutsche Sprache.

[17] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq : A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[18] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe : Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

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

[20] Peter Prettenhofer and Benno Stein. Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1127, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[21] Loïc Vial, Benjamin Lecouteux, and Didier Schwab. Sense Vocabulary Compression through the Semantic Knowledge of WordNet for Neural Word Sense Disambiguation. In *Global Wordnet Conference*, Wroclaw, Poland, 2019.

[22] Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. Paws-x : A cross-lingual adversarial dataset for paraphrase identification. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.