# L'évaluation des représentations vectorielles de mots en utilisant WordNet

Nourredine Aliane[1] Jean-Jacques Mariage[1] Gilles Bernard[1,2]
(1) Laboratoire LIASD Université Paris 8, 2 rue de la Liberté 93526 Saint-Denis cedex
(2) Instutit d'Enseignement à Distance
nourredine@ai.univ-paris8.fr, jjm@ai.univ-paris8.fr,
gilles.bernard@iedparis8.net

## RÉSUMÉ

Les méthodes d'évaluation actuelles des représentations vectorielles de mots utilisent généralement un jeu de données restreint et biaisé. Pour pallier à ce problème nous présentons une nouvelle approche, basée sur la similarité entre les synsets associés aux mots dans la volumineuse base de données lexicale WordNet. Notre méthode d'évaluation consiste dans un premier temps à classer automatiquement les représentions vectorielles de mots à l'aide d'un algorithme de clustering, puis à évaluer la cohérence sémantique et syntaxique des clusters produits. Cette évaluation est effectuée en calculant la similarité entre les mots de chaque cluster, pris deux à deux, en utilisant des mesures de similarité entre les mots dans WordNet proposées par NLTK (wup _similarity). Nous obtenons, pour chaque cluster, une valeur entre 0 et 1. Un cluster dont la valeur est 1 est un cluster dont tous les mots appartiennent au même synset. Nous calculons ensuite la moyenne des mesures de tous les clusters. Nous avons utilisé notre nouvelle approche pour étudier et comparer trois méthodes de représentations vectorielles : une méthode traditionnelle, WebSOM et deux méthodes récentes, word2vec (Skip-Gram et CBOW) et GloVe, sur trois corpus : en anglais, en français et en arabe.

## ABSTRACT

**Evaluating word representations using WordNet.**

Current evaluation methods for word representations generally use a restricted and biased dataset. To overcome this problem we present a new approach, based on the similarity between synsets associated with words in the large WordNet lexical database. Our evaluation method consists first of all in automatically arranging the vector representations of words in clusters with a clustering algorithm, and then to evaluate the semantic and syntactic coherence of the clusters produced. This evaluation is performed by calculating the similarity between the words of each cluster, taken two by two, using similarity measures between the words in WordNet proposed by NLTK (wup _similarity). We obtain, for each cluster, a value Between 0 and 1. A cluster whose value is 1 is a cluster whose words belong to the same synset. The average of the measurements of all the clusters is then calculated. We used our new approach to study and compare three vector representation methods : a traditional WebSOM method and two recent methods, word2vec (Skip-Gram and CBOW) and GloVe, on three corpora : in English, French and Arabic.

MOTS-CLÉS : Representations vectorielles de mots, word2vec, GloVe, WebSOM, WordNet, similarité entre mots, clustering.

KEYWORDS: word vector representations, word2vec, GloVe, WebSOM, WordNet, word similarity, clustering.

# 1 Introduction

Une méthode de représentation vectorielle a pour but d'associer à chaque mot dans un corpus textuel, un vecteur à valeurs réelles, tel que les composantes de ce vecteur décrivent le mieux possible le sens de ce mot dans son contexte. Cependant, la tâche la plus difficile est de vérifier la qualité des vecteurs produits par ces méthodes. L'évaluation se fait généralement par un travail manuel ou avec une évaluation directe, en utilisant un petit jeu de données comme : "WordSim-353" (Finkelstein *et al.*, 2001), "TOEFL" (Landauer & Dutnais, 1997) constitué de 80 questions à choix multiples, "Google's analogy dataset" qui comporte 19 544 analogies ou "MSR's analogy dataset" qui contient 8000 analogies morpho-syntaxiques. Une étude récente de (Baroni *et al.*, 2014), conduit un ensemble d'expériences en comparant la méthode de word2vec (Mikolov *et al.*, 2013) aux autres méthodes traditionnelles. (Levy *et al.*, 2015) ont toutefois trouvé des résultats différents, en utilisant le même jeu de données. Comme le rappellent (Claveau & Kijak, 2015), "*L'évaluation directe séduit par sa simplicité, mais pose la question de l'adéquation des lexiques utilisés comme références*". Afin de permettre d'utiliser un lexique de référence plus conséquent, nous proposons une nouvelle approche d'évaluation indirecte, basée sur l'idée d'exploiter les mesures de similarités entre les mots de WordNet (Miller, 1995). Ces mesures sont présentées dans (Pedersen *et al.*, 2004). Elles sont implémentées dans NLTK [1]. La base de données lexicale WordNet (Miller, 1995), plus volumineuse, offre des possibilités plus intéressantes. Elle contient plus de 200 000 mots, avec leurs relations sémantiques et lexicales. Nous décrivons notre approche plus en détail dans la section 4. Pour nos expériences, nous avons choisi trois méthodes de représentations vectorielles de mots : word2vec (Mikolov *et al.*, 2013), GloVe (Pennington *et al.*, 2014) et WebSOM (Kohonen *et al.*, 1998). Elles sont toutes fondées sur l'hypothèse de (Harris, 1954), selon laquelle les mots apparaissant dans des contextes similaires ont un sens similaire. Les modèles de représentation vectorielle de mots transforment l'analyse distributionnelle d'un corpus en espace vectoriel, dans lequel deux vecteurs proches géométriquement représentent deux mots sémantiquement proches.

# 2 Représentation vectorielle de mots

Dans ce paragraphe, nous présentons brièvement les trois méthodes de representation vectorielle de mots que nous avons choisies pour conduire nos expérimentations.

## 2.1 WebSOM

Cette technique de représentation a été utilisée dans le système WebSOM (Kohonen *et al.*, 1998). Dans un premier temps, il est associé à chaque mot $m_i$, un vecteur $x_i$, de dimension $d$, dont les composantes sont des nombres réels, initialisés aléatoirement entre 0 et 1. Dans un second temps, le mot $m_i$ sera représenté par un autre vecteur $X_i$, qui est déterminé de la façon suivante :

$$X_i = (p_N(x_i)...p_1(x_i)\ \epsilon.x_i\ s_1(x_i)...s_N(x_i))^T.$$

Où : $p_1(x_i)$ et $s_1(x_i)$ sont respectivement les vecteurs moyens des vecteurs qui correspondent à tous les mots prédécesseurs immédiats et successeurs immédiats du mot $m_i$ dans l'ensemble de corpus, ($p_1(x_i)$ et $s_1(x_i)$ sont également de dimension $d$). La fenêtre contextuelle est de $(2N + 1)$ mots

1. NLTK (Natural Language Toolkit) est une bibliothèque logicielle en Python.

(le mot courant, N mots précédents et N mots suivants). $\epsilon$ est un réel positif inférieur à 1. Il sert à contrôler le rôle du mot $m_i$ dans son contexte ($\epsilon = 1$ signifie que le vecteur $x_i$, initialisé aléatoirement au départ, et les autres vecteurs des contextes du mot $m_i$, sont de même importance). Le vecteur $X_i$ est de dimension $(2N + 1)d$

### 2.2 word2vec

Une méthode proposée par (Mikolov *et al.*, 2013) fondée sur les réseaux de neurones, a été implémentée dans un outil qui s'appelle word2vec. Deux modèles de représentation des mots sont implémentés dans word2vec. Ce sont le continuous bag-of-words (CBOW) et le Skip-Gram.

1. CBOW a pour objectif de prédire la probabilité d'un mot, à partir de ses contextes. Cette représentation de mots consomme moins de temps en entraînement que le skip-gram.
2. Skip-Gram, contrairement aux CBOW, vise à prédire la probabilité des contextes d'un mot à partir de ce mot.

### 2.3 GloVe

GloVe (Global Vectors for Word Representation) (Pennington *et al.*, 2014) est un modèle proposé par l'équipe NLP de l'université de Stanford. Cette méthode combine les avantages de la factorisation matricielle globale et des méthodes de contexte local. Le contexte est une fenêtre de longueur fixe d'éléments lexicaux centrés sur le mot. On cherche à représenter chaque mot $i$ et chaque mot $j$ apparaissant dans le même contexte par des vecteur $v_i$ et $v_j$ respectivement, de dimension $d$ tels que : $v_i.v_j + b_i + b_j = log(X_{ij})$. Où $X_{ij}$ représente le nombre de fois où le mot $j$ se produit dans le contexte du mot $i$. $b_i$ et $b_j$ sont des biais scalaires associés aux mots $i$ et $j$ respectivement.

## 3 WordNet et la similarité entre les mots

WordNet (Miller, 1995) est une grande base de données lexicale de l'anglais, développée par des linguistes de l'université de Princeton. Les mots y sont regroupés en ensembles de synonymes cognitifs (synsets). Les synsets sont interconnectés au moyen de relations conceptuelles-sémantiques et lexicales. WordNet est distribuée sous licence libre, la dernière version 3.1 répertorie plus de 200 000 mots. (Pedersen *et al.*, 2004) présentent plusieurs algorithmes de calcul de similarité entre mots, utilisant la structure et le contenu de WordNet. Après étude de ces différentes mesures de similarités, la wup_similarity semble la plus pertinante.

**wup_similarity** (Wu & Palmer, 1994) : renvoie un score entre 0 et 1, en fonction des profondeurs des deux mots et de celle de leur dernier ancêtre commun dans une taxonomie. Elle est définie par l'équation :

$$wup\_similarity(s_1, s_2) = \frac{2 * depth(lcs(s_1, s_2))}{depth(s_1) + depth(s_2)} \quad (1)$$

$lcs(s_1, s_2)$ : le dernier ancêtre commun entre $s_1$ et $s_2$ (pour l'anglais : least common subsumer). Il correspond au dernier noeud du graphe taxonomique à partir duquel divergent les branches de $s_1$ et $s_2$. $depth(s_i)$ est la profondeur de $s_i$ (le nombre d'arêtes entre la racine et $s_i$, $depth(racine) = 1$).

# 4 Méthodologie

La Figure 1 donne une vue générale de la méthode proposée :

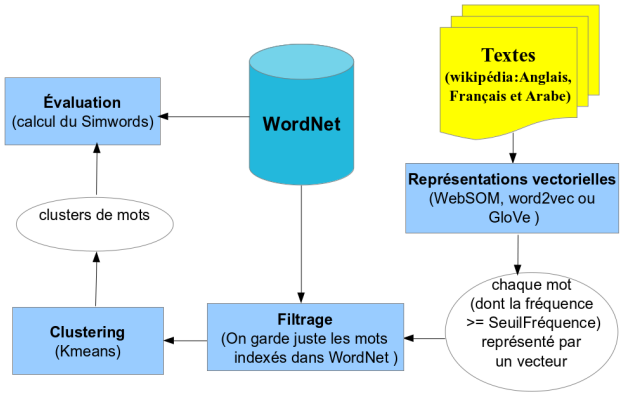


FIGURE 1 – Le principe de fonctionnement de notre système

**Représentation vectorielle de mots :** Nous appliquons les trois méthodes de représentation vectorielle de mots sur nos corpus, afin d'associer un vecteur à chaque mot dont la fréquence d'usage est supérieure à un seuil donné.

**Filtrage :** Cette étape consiste à sélectionner, parmi les mots qui ont été représentés par des vecteurs, ceux qui sont indexés dans WordNet ($w \in WordNet \equiv wordnet.synsets(w) \neq \emptyset$).

**Clustering :** Après la représentation vectorielle de mots, nous utilisons Kmeans++ (Arthur & Vassilvitskii, 2007) [2], afin de regrouper les mots qui ont une proximité sémantique ou syntaxique dans un même cluster.

**Évaluation :** Nous calculons la similarité entre les mots de chaque cluster, pris deux à deux en utilisant la $wup\_similarity$. Nous définissons $Simwords(C_i)$ [3] la similarité entre les mots du cluster $C_i$ par l'équation (2).

2. kmeans++ est implémenté dans Scikit-learn (est une bibliothèque libre Python dédiée à l'apprentissage automatique).
3. https ://github.com/nourredinealiane/simwords.

$$Simwords(C_i) = \frac{\sum_{k=1}^{n_{C_i}-1} \sum_{j=k+1}^{n_{C_i}} wup\_simMax(m_k, m_j)}{n_{C_i}(n_{C_i}-1)/2} \quad (2)$$

$$wup\_simMax(m_k, m_j) = \underset{k \in wordnet.synsets(m_k),\ j \in wordnet.synsets(m_j)}{ArgMax}\ wup\_similarity(k, j) \quad (3)$$

Où les mots $m_k$ et $m_j$ appartiennent au clusters $C_i$. Nous calculons ensuite la moyenne des mesures de tous les clusters, par l'équation :

$$Simwords = \frac{\sum_{i=1}^{k} Simwords(C_i)}{k} \quad (4)$$

Où $n_{C_i}$ est le nombre de mots du cluster $C_i$, k est le nombre de clusters.

*Note* : Pour l'arabe et le français, NLTK utilise une traduction automatique, par exemple :

*wordnet.synsets(m, lang='fra')*, renvoie les synonymes de la traduction en anglais du mot français $m$ (pour l'arabe, *lang = 'arb'*). Le script Python ci-dessous montre comment calculer $wup\_simMax$ entre le mot «chat» et le mot «chien».

```
>>> from nltk.corpus import wordnet
>>> from itertools import product
>>> syns1 = wordnet.synsets('chat', lang='fra')
>>> syns1
[Synset('computerized_tomography.n.01'), Synset('cat.v.01'),
Synset('felis.n.01'), Synset('cat.n.01'), Synset('tom.n.02'),...]
>>> syns2 = wordnet.synsets('chien', lang='fra')
>>> syns2
[Synset('dog.n.01'), Synset('pooch.n.01'), Synset('hound.n.01'),
Synset('andiron.n.01'), Synset('pawl.n.01'),...]
>>> wup_simMax = max((wordnet.wup_similarity(s1, s2) or 0, s1, s2)
                              for s1, s2 in product(syns1, syns2))
>>> wup_simMax
(0.8571428571428571, Synset('cat.n.01'), Synset('dog.n.01'))
#le dernier ancetre commun est :
>>> wup_simMax[1].lowest_common_hypernyms(wup_simMax[2])
[Synset('carnivore.n.01')]
```

# 5 Corpus

Pour évaluer et comparer les trois méthodes de représentations vectorielles de mots choisies, nous avons sélectionné trois corpus textuels (Wikipédia dump 2017) en trois langues différentes : anglais, français et arabe. Les propriétés détaillées de ces corpus textuels sont indiquées dans le tableau 1.

| Corpus | Nombre de mots | Vocabulaire : nombre de mots uniques | Fréquence des mots | Nombre de mots indexés dans WordNet |
|---|---|---|---|---|
| anglais | 2 409 291 852 | 9 045 033 | $\geq$ 600 = 96 967 | 47 118 |
| français | 804 476 834 | 4 348 227 | $\geq$ 300 = 86 697 | 18 600 |
| arabe | 117 472 209 | 2 140 757 | $\geq$ 300 = 33 974 | 1 646 |

TABLE 1 – Propriétés des corpus

# 6 Expérimentations et résultats

Pour chaque méthode, nous avons déterminé empiriquement les paramètres (taille de la fenêtre, dimension des vecteurs, nombre d'itérations, k le nombre de clusters, ...) qui donnent le meilleur résultat (Simwords maximum). Pour chaque corpus, nous avons utilisé le même nombre de clusters pour les trois méthodes de vectorisation. Les meilleurs résultats obtenus sont présentés dans le tableau 2 ci-dessous.

| | Simwords | | | | | |
|---|---|---|---|---|---|---|
| | anglais | | français | | arabe | |
| | k = 2400 | k = 1200 | k = 800 | k = 400 | k = 300 | k = 150 |
| skip-gram | **0,603** | **0,542** | 0,517 | 0,452 | 0,376 | 0,338 |
| CBOW | 0,581 | 0,522 | **0,536** | **0,467** | **0,412** | **0,382** |
| GloVe | 0,548 | 0,486 | 0,529 | 0,461 | 0,396 | 0,355 |
| WebSOM | 0,471 | 0,453 | 0,472 | 0,438 | 0,370 | 0,354 |

TABLE 2 – Évaluation des méthodes de représentations vectorielles de mots pour les trois langues : anglais, français et arabe, k est le nombre de clusters.

Il apparaît que word2vec, pour les deux modèles, représente mieux les mots selon la mesure proposée. skip-gram est plus performant pour le corpus anglais et CBOW pour le français et l'arabe. Nous ne prenons pas en compte les clusters singleton dans le calcul de Simwords, car ils augmentent les valeurs sans pour autant signifier une bonne méthode de vectorisation des mots. Par exemple, en segmentant les vecteurs produits par la méthode WebSOM avec le corpus français, on obtient plusieurs clusters singletons (429/800). Si nous les comptons, Simwords = 0,75522 au lieu 0.47218. En revanche, avec la méthode word2vec (CBOW), on obtient 166 clusters singleton sur 800 clusters, ce qui donne Simwords = 0,63301 au lieu de 0.53693 sans les compter. L'augmentation du nombre de clusters, augmente la valeur de Simwords (similarité entre mots) (voir la figure 2).

L'augmentation de la dimension des vecteurs pour word2vec et GloVe augmente les performances (figure 3, à gauche). L'augmentation de la taille de la fenêtre pour word2vec-CBOW, influence négativement les performances. Et pour GloVe, une fenêtre de 10 fait mieux que celles de 4 ou de 14 (figure 2, à droite). Les trois méthodes produisent quelques clusters parfaits (dont les mots appartiennent au même synset). Voici quelques exemples produits par CBOW avec Wikipédia en français : [façon, manière], [intersection, carrefour, croisement], [tremblement, séisme], [voyage, visite, séjour], [cour, tribunal], [organisme, corps, organe], [régler, résoudre], [résumé, synopsis].

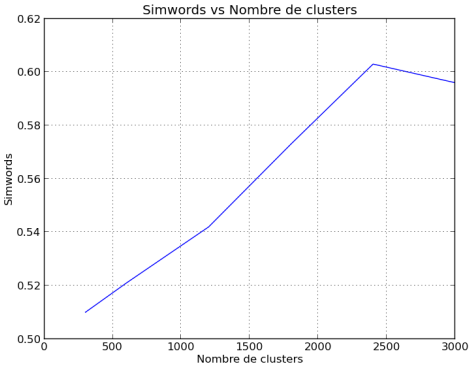


FIGURE 2 – Simwords vs Nombre de clusters : corpus anglais, avec word2vec (skip-gram), taille de la fenêtre = 8, dimension des vecteurs = 200.

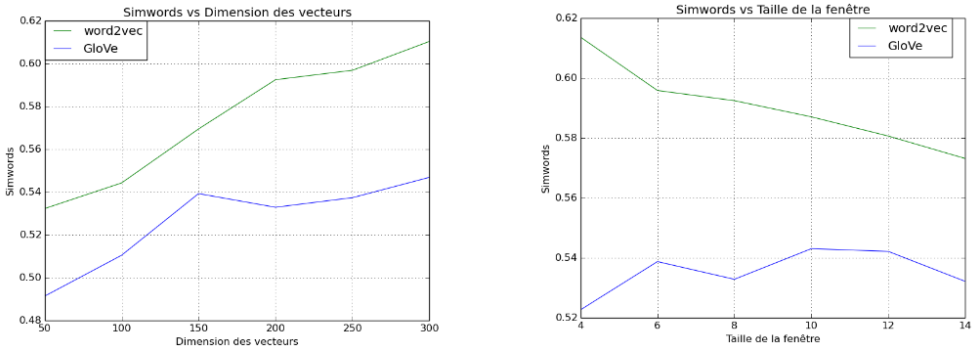


FIGURE 3 – À gauche, Simwords vs Dimension des vecteurs ; à droite, Simwords vs Taille de la fenêtre ; échantillon de 5000 vecteurs du corpus anglais, avec word2vec (CBOW) et GloVe, nombre de clusters = 800.

# 7 Conclusion

Nous avons proposé une solution d'évaluation de representations vectorielles de mots qui s'appuie sur un large lexique de référence. Elle permet de mesurer concrètement les performances d'une méthode de représentation vectorielle de mots, ou d'en optimiser les paramètres afin d'augmenter ses performances. Notre approche présente l'avantage d'être plus générale que les méthodes existant précédemment, grâce à la richesse sémantique qu'apportent les synonymes de WordNet et à la possibilité de l'utiliser avec différentes langues. Les 200 000 mots indexés dans WordNet offrent la possibilité de calculer à peu près 20 milliards de similarités entre mots. En revanche, avec "WordSim-353" par exemple, nous ne disposons que de 350 similarités (nombre de paires de mots). Ces résultats encourageants montrent l'efficacité de notre méthode et permettent d'envisager l'utilisation d'autres mesures de similarités dans WordNet et de comparer l'efficacité d'autres méthodes de vectorisation de mots.

# Références


ARTHUR D. & VASSILVITSKII S. (2007). K-means++ : the advantages of careful seeding. In *In Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*.

BARONI M., DINU G. & KRUSZEWSKI G. (2014). Don't count, predict ! a systematic comparison of context-counting vs. context-predicting semantic vectors. *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, **1**, 238–247.

CLAVEAU V. & KIJAK E. (2015). Thésaurus distributionnels pour la recherche d'information et vice-versa. In *Conférence en Recherche d'Information et Applications*, Actes de la conférence CORIA 2015, Paris, France.

FINKELSTEIN L., GABRILOVICH E., MATIAS Y., RIVLIN E., SOLAN Z., WOLFMAN G. & RUPPIN E. (2001). Placing search in context : The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, p. 406–414, New York, NY, USA : ACM.

HARRIS Z. S. (1954). Distributional structure. *Word*, **10**, 146–162.

KOHONEN T., KASKI S., LAGUS K. & SALOJARVI J. (1998). Websom - self-organizing maps of document collection. *Helsinki University of Technologie, Finland.*

LANDAUER T. K. & DUTNAIS S. T. (1997). A solution to plato's problem : The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *PSYCHOLOGICAL REVIEW*, **104**(2), 211–240.

LEVY O., GOLDBERG Y. & DAGAN I. (2015). Improving distributional similarity with lessons learned from word embeddings. *TACL*, **3**, 211–225.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*.

MILLER G. A. (1995). Wordnet : A lexical database for english. *Commun. ACM*, **38**(11), 39–41.

PEDERSEN T., PATWARDHAN S. & MICHELIZZI J. (2004). Wordnet : :similarity : Measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, HLT-NAACL–Demonstrations '04, p. 38–41, Stroudsburg, PA, USA : Association for Computational Linguistics.

PENNINGTON J., SOCHER R. & MANNING C. D. (2014). Glove : Global vectors forword representation. In *EMNLP*, volume 14, p. 1532–1543.

WU Z. & PALMER M. (1994). Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, p. 133–138, Stroudsburg, PA, USA : Association for Computational Linguistics.