# Apprentissage de plongements lexicaux par une approche réseaux complexes

Victor Connes, Nicolas Dugué

# Apprentissage de plongements lexicaux par une approche réseaux complexes

Victor Connes[1, 2] Nicolas Dugué[2]
(1) Le Mans Université, LIUM, EA 4023, Laboratoire d'Informatique de l'Université du Mans
(2) LS2N Université de Nantes – faculté des Sciences et Techniques (FST) Bâtiment 34 2 Chemin de la Houssinière BP 92208, 44322 Nantes Cedex 3
victor.connes@univ-nantes.fr, nicolas.dugue@univ-lemans.fr

## RÉSUMÉ

La littérature des réseaux complexes a montré la pertinence de l'étude de la langue sous forme de réseau pour différentes applications : désambiguïsation, résumé automatique, classification des langues, *etc.* Cette même littérature a démontré que les réseaux de co-occurrences de mots possèdent une structure de communautés latente. Nous formulons l'hypothèse que cette structuration du réseau sous forme de communautés est utile pour travailler sur la sémantique d'une langue et introduisons donc dans cet article une méthode d'apprentissage de plongements originale basée sur cette hypothèse. Cette hypothèse est cohérente avec la proximité qui existe entre la détection de communautés sur un réseau de co-occurrences et la factorisation d'une matrice de co-occurrences, méthode couramment utilisée pour l'apprentissage de plongements lexicaux. Nous décrivons notre méthode structurée en trois étapes : construction et pré-traitement du réseau, détection de la structure de communautés, construction des plongements de mots à partir de cette structure. Après avoir décrit cette nouvelle méthodologie, nous montrons la pertinence de notre approche avec des premiers résultats d'évaluation sur les tâches de catégorisation et de similarité. Enfin, nous discutons des perspectives importantes d'un tel modèle issu des réseaux complexes : les dimensions du modèle (les communautés) semblent interprétables, l'apprentissage est rapide, la construction d'un nouveau plongement est presque instantanée, et il est envisageable d'en expérimenter une version incrémentale pour travailler sur des corpus textuels temporels.

## ABSTRACT

**Complex networks based word embeddings.**

Most of the time, the first step to learn word embeddings is to build a word co-occurrence matrix. As such matrices are equivalent to graphs, complex networks theory can naturally be used to deal with such data. In this paper, we consider applying community detection, a main tool of this field, to the co-occurrence matrix corresponding to a huge corpus. Community structure is used as a way to reduce the dimensionality of the initial space. Using this community structure, we propose a method to extract word embeddings that are comparable to the state-of-the-art approaches.

MOTS-CLÉS : Plongements lexicaux, réseaux complexes, détection de communautés.

KEYWORDS: Word embeddings, complex networks, community detection.

# 1 Introduction

Dans l'état de l'art de l'apprentissage de plongements lexicaux, on recense de nombreuses approches basées sur une matrice de co-occurrences termes-termes construite en utilisant de grands corpus (Pennington *et al.*, 2014; Levy *et al.*, 2015). Les auteurs factorisent ensuite cette matrice creuse de façon à obtenir un nouvel espace dans lequel chaque terme est représenté par un vecteur dense.

Dans le domaine des réseaux complexes, ces matrices de co-occurrences sont appelées *graphes* ou *réseaux*. L'étude du langage naturel par le prisme des réseaux complexes n'est pas une science nouvelle. L'état de l'art du domaine utilise également de grands corpus pour construire des réseaux $G = (V, E)$ tels que chaque nœud $u \in V$ du réseau représente un terme du vocabulaire, et un lien $(u, v) \in E$ entre deux nœuds représente une co-occurrence dans le corpus entre deux termes. Ces réseaux peuvent être dirigés, ou valués, on se dote alors d'une fonction $w$ qui associe un poids à chaque lien $w : E \to \mathbb{R}$.

Ces travaux ont notamment permis de révéler plusieurs propriétés de ces réseaux et ainsi de mieux comprendre la façon dont est construite la langue : ces réseaux sont petit-monde (i Cancho & Solé, 2001), sans-échelle avec une loi de puissance à deux vitesses (i Cancho & Solé, 2001) expliquée par le modèle de Dorogovtsev & Mendes 2001, et le poids des liens suit également une loi de puissance dans le cas des réseaux valués (Gao *et al.*, 2014; Masucci & Rodgers, 2006).

Parmi les propriétés observées, ce papier se concentre sur la présence d'une structure de communautés dans ces réseaux (Newman, 2004). La structure de communautés d'un réseau est une partition des nœuds du réseau telle que pour chaque partie, les nœuds sont plus connectés entre eux qu'avec le reste du réseau (Newman & Girvan, 2004). Nous faisons l'hypothèse que cette structure de communautés permet de construire des plongements lexicaux.

Cette hypothèse se base sur deux constats. Le premier vient des exemples de Palla *et al.* 2005 qui semblent indiquer que les communautés encapsulent une partie de l'information sémantique. D'ailleurs, la définition de la structure de communautés vient appuyer ce constat : pour chaque partie (communauté) de la partition (structure de communautés), les nœuds sont plus connectés entre eux qu'avec le reste du réseau. Au regard de l'hypothèse de Firth "*a word is characterized by the company it keeps*", on comprend que chaque communauté sera constituée de mots qui seront utiles pour se caractériser les uns les autres. Le second constat vient de certains travaux de la littérature qui mettent en évidence les liens entre décomposition en valeur singulière (SVD) et détection de communautés (Sarkar & Dong, 2011). Or, appliquer une SVD à une matrice de co-occurrences pondérée par la *positive pointwise mutual information* est une méthode efficace pour aboutir à des plongements lexicaux (Levy *et al.*, 2015).

Nous présentons donc Section 2 notre approche basée sur la détection de communautés pour extraire des plongements. Cette approche considère chaque communauté comme une dimension, et les liens d'un nœud vers ces communautés permettent de calculer pour chaque dimension la valeur de la composante. Nous montrons Section 3 que les résultats expérimentaux démontrent la pertinence de l'approche, d'un point de vue qualitatif, mais également quantitatif. Enfin, nous discuterons Section 4 des avantages d'une telle approche. Tout d'abord, celle-ci permet d'espérer des dimensions interprétables. Ensuite, le calcul d'un plongement pour un terme est très rapide. Enfin, ce type d'approche ouvre des perspectives pour créer des plongements lexicaux évoluant dans le temps via des algorithmes de détection de communautés incrémentaux.

# 2 Méthode

**Données.** Les données utilisées sont les GoogleBooksNgram [1] anglais, corpus BristishEnglish et EnglishFiction. Les GoogleBooksNgram sont des recueils de co-occurrences de termes observées sur une grande bibliothèque de textes allant des années 1800 à 2008. Les co-occurrences sont fournies avec une fenêtre de contexte allant de 2 à 5. Pour nos expériences, nous avons conservé seulement les co-occurrences observées depuis 1980 aboutissant à un vocabulaire avant pré-traitements d'environ 380000 termes.

**Construction et pré-traitement du réseau.** Une fois le réseau créé en exploitant les co-occurrences d'un corpus textuel avec une fenêtre de taille $f$, on obtient alors $G = (V, E, w)$ comme décrit. Pour rappel, l'ensemble des nœuds $V$ est équivalent au *vocabulaire* considéré, l'ensemble des liens $E$ représente les co-occurrences entre les termes du vocabulaire, et on définit la pondération des liens de $E$ avec la fonction $w(u, v)$, qui vaut le nombre de co-occurrences observées entre les termes représentés par les nœuds $u$ et $v$ dans le corpus en considérant le paramètre $f$.

Dans le but de ne conserver que les co-occurrences ayant une valeur sémantique, nous supprimons les liens entre les nœuds qui ne révèlent pas une dépendance statistique significative en utilisant l'Éq. 1 :

$$ppmi(w, c) = max\left(0, log_2\left(\frac{p(u, v)}{p(v)p(u)}\right)\right) \quad (1)$$

Ce pré-traitement du réseau découle directement de ce qui est préconisé par l'état de l'art, notamment par Levy *et al.* 2015. Mais il semble également pertinent de l'appliquer pour simplifier le travail de l'algorithme de détection de communautés, dont les résultats s'améliorent avec des pré-traitements de type seuillage ou repondération (Yan *et al.*, 2018).

Dans le but d'alléger le réseau avec un filtre basse-fréquence, nous appliquons l'algorithme 1 qui permet d'obtenir le $k$-cœur du réseau (Matula & Beck, 1983) : il s'agit de supprimer tous les nœuds ayant moins de $k$ voisins de manière récursive jusqu'à que tous les nœuds restant dans le réseau soient connectés à au moins $k$ voisins.

Enfin, dans le but de supprimer les mots vides et de limiter l'influence des hautes fréquences (comme dans Glove (Pennington *et al.*, 2014) ou Word2vec (Mikolov *et al.*, 2013), nous choisissons de supprimer les $ntop$ nœuds de plus haut degré. Les meilleurs résultats sont empiriquement obtenus pour $ntop$ = 200 et $k$ = 10. Après pré-traitement, nous aboutissons à un vocabulaire de 135.000 mots dans le cas de notre corpus.

**Détection de communautés.** Une fois le réseau généré et pré-traité, la seconde étape consiste à détecter les communautés qui serviront par la suite de dimensions aux vecteurs de plongement lexicaux. On dit que $C$ est une partition de $V$ telle que $C = \{C_0, C_1, ..., C_n\}$ avec $\cup_i C_i \in C = V$. S'il n'existe pas de définition unique du concept de communauté, une structure de communautés est souvent définie comme une partition du réseau telle que les nœuds de chaque partie sont plus connectés entre eux qu'avec le reste du réseau.

1. http ://storage.googleapis.com/books/ngrams/books/datasetsv2.html

**Algorithm 1** Extraction du K-cœur

**Require:** $G = (V, E)$ graphe, $k$ entier
  $convergence \leftarrow False$
  **while** $convergence$ **do**
    $convergence \leftarrow True$ ; $V' \leftarrow \{\}$
    **for** $\forall n \in V$ **do**
      **if** $degres(n) < k$ **then**
        $V' \leftarrow V' \cup \{n\}$ ; $convergence \leftarrow False$
      **end if**
    **end for**
    $V \leftarrow V \setminus V'$
  **end while**
  **return** G

De nombreuses méthodes existent pour réaliser l'extraction de ces communautés. Nous avons choisi l'algorithme 2 de propagation de labels introduit par Raghavan *et al.* 2007, dont la complexité est quasi-linéaire en $O(|V|)$, et qui génère *théoriquement* des communautés dont les tailles suivent une distribution permettant d'éviter d'avoir en grand nombre des communautés trop grandes (fourre-tout) ou trop petites (trop spécifiques) (Dao *et al.*, 2018). En pratique, on constate Figure 1 un très grand nombre de petites communautés. Il s'agirait de considérer des adaptations de l'algorithme de propagation de labels pour éviter cet écueil.

**Algorithm 2** Propagation de labels

**Require:** $G = (V, E, w)$ graphe
  $\forall n \in V, c(n) \leftarrow n$
  *On parle de convergence lorsque la communauté de chaque nœud est la communauté majoritaire de ses voisins.*
  **while** $check\_convergence(G, C, w)$ **do**
    **for** $\forall n \in V in\ random\ order$ **do**
      *La communauté du nœud devient la communauté majoritaire des voisins.*
      $C(n) \leftarrow countmax(\{c(v),\ \forall v \in voisins(n)\}, w)$
    **end for**
  **end while**
  **return** C

**Extraction des plongements lexicaux.** Une fois les communautés extraites, il reste à construire les plongements pour notre vocabulaire. Pour ce faire, nous considérons la distribution des liens de chaque nœud à travers les communautés. Néanmoins, il s'agit de prendre en compte l'influence du degré du nœud et de celui de ses voisins. Prenons un exemple pour clarifier : celui des mots *escroc* et *aigrefin*. Ces deux mots sont proches d'un point de vue sémantique. Par contre, *escroc* est plus fréquent qu'*aigrefin*. Il sera donc mécaniquement d'un degré pondéré plus élevé. Une fois cette remarque faite, on se rend compte que si on considère seulement la distribution d'*escroc* dans les communautés pour créer son plongement, la norme de son vecteur sera plus grande que celle d'*aigrefin*.

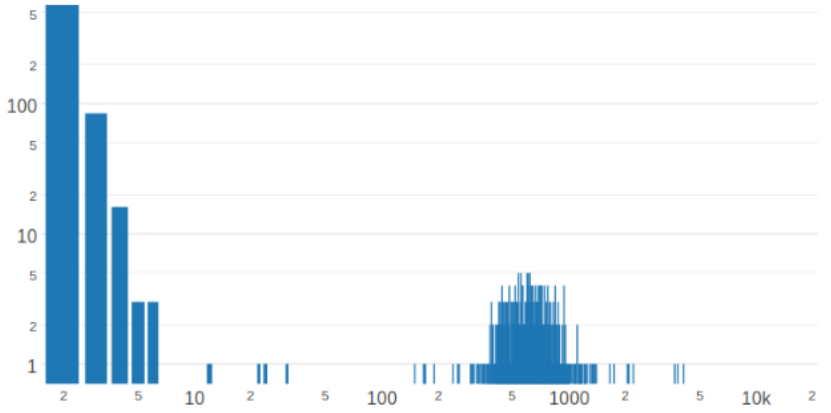


FIGURE 1 – Distribution de la taille des communautés (en log-log)

Par ailleurs, la taille des communautés a une influence similaire. L'algorithme de détection de communautés aboutit (sauf exception) à une partition dont les tailles des communautés sont hétérogènes. Si l'on ne tient pas compte de cet état de fait, les communautés les plus petites auront mécaniquement une composante plus faible que les grosses communautés dans les vecteurs.

Ainsi, si on note $e_n$ le plongement du nœud représentant le mot $n$, $e_n \in \mathbb{R}^{|C|}$ et $e_n^c$ la valeur de la composante correspondante à la communauté $c$ de $e_n$, cette valeur se calculera ainsi :

$$\hat{e}_n^c = \frac{1}{|N^c(e_n)|} \sum_{v \in N^c(e_n)} sppmi(n, v) \quad (2) \qquad e_n^c = \frac{\hat{e}_n^c - \mu(\hat{e}_*^c)}{\sigma(\hat{e}_*^c)} \quad (3)$$

Avec $N^c(e_n) = voisins(n) \cap C_c$, *i.e.* l'ensemble des voisins du noeud représentant le mot $n$ appartenant à la communauté $c$, $\mu(\hat{e}_*^c)$ et $\sigma(\hat{e}_*^c)$ respectivement la moyenne et l'écart-type des valeurs de $\hat{e}_n^c$, $\forall n \in V$ et $sppmi$ une version normalisée de la $ppmi$ (Éq. 1) à valeur dans $[0, 1]$.

L'utilisation de la $sppmi$ nous permet de contrebalancer l'influence du degré du nœud et de celui de ses voisins, celle du *z-score* (Éq. 3) l'influence de la taille des communautés. L'exemple de la Figure 2 illustre le résultat une fois toutes les étapes réalisées, en proposant une visualisation des vecteurs de *bush*, *putin* et *chirac* via les 30 dimensions les plus utiles pour la caractériser (10 par vecteur).

# 3 Résultats

Nous débuterons cette section avec quelques évaluations empiriques purement qualitatives concernant la pertinence des dimensions exploitées (les communautés) et l'espace appris (les voisinages). Nous donnerons enfin des résultats quantitatifs qui démontrent l'intérêt de l'approche. Sur notre corpus, après pré-traitement, nous aboutissons à une taille de vocabulaire qui est d'un peu plus de $135.000$ mots, ce qui correspond au nombre de nœuds du graphe. Après détection de communautés, nous obtenons environ $30.000$ communautés (voir la distribution de leurs tailles Figure 1), soit des vecteurs de taille $30.000$. En revanche, on constate qu'en moyenne, seulement 300 (environ) composantes du vecteur sont non-nulles, les vecteurs sont donc extrêmement creux.

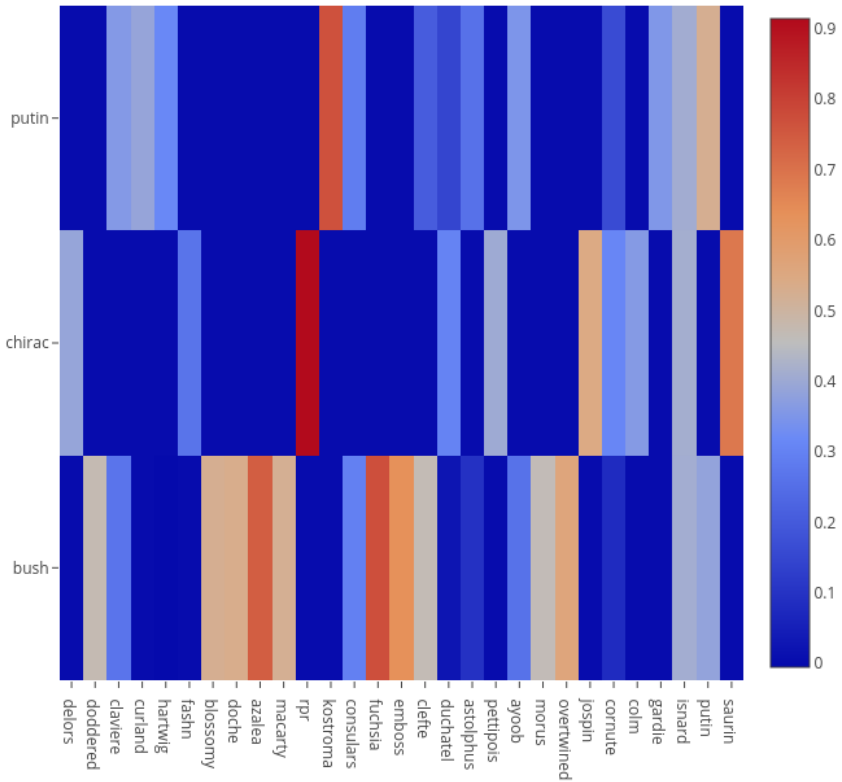


FIGURE 2 – En abscisse, les étiquettes des 30 dimensions les plus caractéristiques des vecteurs de putin, chirac et bush (10 par vecteur). La couleur représente la valeur de la composante pour chacun des vecteurs.

**Communautés et interprétabilité.** En utilisant des méthodes d'étiquetage des communautés, il est possible d'évaluer empiriquement la pertinence de l'approche. Les communautés extraites constituent les dimensions des vecteurs qui semblent ainsi interprétables et cohérentes. Nous donnons ici en exemple trois communautés et leurs étiquettes caractéristiques :

— ('officiel', 'republique', 'parisienne', 'couture', 'senat')
— ('copper', 'iron', 'stand', 'metal', 'upon')
— ('volleyball', 'handball', 'softball', 'badminton', 'basketball')

La première communauté regroupe les mots français du corpus. La seconde concentre du vocabulaire lié aux métaux même si l'on peut constater qu'on y trouve des intrus (*stand* et *upon*). Enfin, la dernière communauté regroupe du vocabulaire lié au sport. Ce sont les mêmes méthodes d'étiquetage qui sont utilisées dans la Figure 2 pour étiqueter chaque dimension des vecteurs visualisés. On reconnaît un bon nombre de ces étiquettes comme par exemple *Delors* (Jacques), *rpr* (Rassemblement pour la république), *Jospin* (Lionel) pour caractériser le vecteur de Jacques Chirac. Il est particulièrement intéressant de s'intéresser au vecteur *bush*, où l'on trouve *Mcarthy* (Joseph), ou encore *Putin* (Vladimir) mais également des noms d'arbustes puisque c'est l'un des sens de *bush* (*azalea*, *fuschia*). Grâce à l'interprétabilité du modèle, nous pouvons ainsi observer la façon dont celui-ci intègre la polysémie/l'homonymie.

**Base canonique et interprétabilité.** Avec une approche telle que celle de *Word2Vec*, il est difficile d'interpréter les dimensions. Tout d'abord, supposer qu'il est possible d'interpréter les dimensions revient à faire l'hypothèse que chaque dimension peut être considérée indépendamment des autres, et que chaque dimension a un sens cohérent, i.e. qu'explorer les vecteurs colinéaires aux vecteurs de la base canonique de l'espace appris permettrait d'extraire ces *sens*. Empiriquement, il est pourtant difficile d'affirmer cela. Considérons un modèle Word2vec à 300 dimensions appris sur le corpus Google News (Mikolov *et al.*, 2013), et prenons des contre-exemples simples. Soit $C$ la base canonique de l'espace de dimension 300 de notre expérimentation telle que $C = \{e_1 = (1, 0, 0, \cdots, 0), e_2 = (0, 1, 0, \cdots, 0), \cdots, e_{300} = (0, 0, 0, \cdots, 1)\}$.

— Considérons ainsi les 10 termes les plus proches de $e_4$ dans l'espace : Ginsburgs, Dinty Moore, jelly sandwiches, cheartier appetites, they'd, Fabens fliers, banana republics, isn, Chipotle burritos, payroll deduction.
— Pour $e_9$, nous obtenons les résultats suivants : costliest natural disasters, counterparty defaults, mute button, closely scrutinized, Bernankes, damage Minsch, student Tyler Clementi, degraded Kenneth Merten, historian Bob Kreipke, Nishu Sood.

Il semble très difficile de tirer une quelconque cohérence dans les termes qui sont retournés, contrairement aux communautés précédemment citées. Les communautés sont en effet des objets concrets, des ensembles de mots du corpus qui sont particulièrement connectés ensemble, et elles peuvent de plus être considérées indépendamment les unes des autres.

Considérons maintenant notre modèle et les vecteurs canoniques de l'espace constitué via l'extraction des communautés. Dans notre modèle comme dans les autres, il est possible d'extraire les plus proches voisins d'un mot ou d'un vecteur en utilisant la similarité *cosine* pour évaluer la distance entre deux vecteurs. Considérons dans les cas des deux exemples suivants les 10 vecteurs les plus proches du vecteur canonique dont la composante non-nulle correspond à la communauté qui contient le mot **alcohol**, puis **petal** :

— mannite, dinitro, polyhydric, benzole, benzol, lactose, fermenter, disaccharide, bisulphide, reconverted.
— sepal, papilionaceous, floret, stamen, blotch, dewdrops, petals, bracts, corolla.

Dans le premier exemple, de manière générale, on obtient des termes liés à l'alcool directement : *mannite* pour le manniotal qui est un alcool, *polyhydric* parce que les alcools de sucre sont dits polyhydriques ; des termes liés au sucre qui est l'un des éléments de base pour la création d'alcool (*lactose*, *disaccharise*), au processus de création d'alcool (*fermenter*), ou à la chimie (*dinitro*, *bisulphide*). Dans le second exemple, tout ou presque est lié à la fleur : les sépales (*sepal*), les étamines (*stamen*), *papilionaceous* qui est une fleur, *drewdrop* qui signifie goutte de rosée, *bract* qui est une petite feuille, *corolla* qui est un synonyme de pétale. Dans ces deux cas, il existe un fort recouvrement entre les 10 plus proches voisins mentionnés ci-dessus et les représentants les plus caractéristiques de la communauté (étiquettes).

Un autre exemple parlant est celui du vecteur $e_{746}$ de notre modèle dont les 10 plus proches voisins sont : sifteen, fiftyfour, fortyseven, fiftyseven, sixtyfive, fortysix, fiftyfive, sixtyseven, twentyeight, twentyseven.

La distance *cosine* peut comme dans les autres modèles être exploitée pour étudier la similarité entre les termes du vocabulaire, pas seulement avec les vecteurs canoniques. Ainsi, la liste de voisins suivante fournit quelques exemples de résultats illustrant le bon fonctionnement de la méthode :

— metal : (metals, metallic, iron, copper, steel, alloy, aluminium, oxides, chromium)
— picture : (pictures, portrait, image, painting, view, images, depiction, portrayal, painted)
— salad : (mayonnaise, ketchup, lettuce, tomato, vegetables, sauce, celery, mashed, cheese)
— mars : (altimeter, orbiter, venus, saturn, jupiter, orbit, pioneer, planets, planet)
— news : (television, cnn, bbc, pathe, nbc, tidings, newspapers, cbs, gaumont)

**Comparaison à l'état de l'art.** Pour obtenir des résultats quantitatifs, nous comparons notre approche à celles de l'état de l'art en considérant deux tâches d'évaluation (Schnabel *et al.*, 2015) :

**Similarité** La tâche de similarité se présente comme une base de données de paires de mots, avec pour chaque paire un score associé. Le score de similarité entre deux mots est issu d'une évaluation humaine. La qualité du modèle peut donc être évaluée en calculant la corrélation entre le vecteur de score humain et le vecteur de distances entre les vecteurs appris. Une corrélation linéaire (coefficient de *Spearman* proche de 1) correspond à un modèle complètement en accord avec l'évaluation humaine.

**Catégorisation** La tâche de catégorisation se présente comme une base de données de paires (mot, catégorie). Le but est de réussir à regrouper des mots en différentes catégories en utilisant les vecteurs appris. Pour faire cela, on opère une analyse de regroupement sur les vecteurs appris. On évalue ensuite le modèle en calculant la pureté entre les regroupements et la catégorisation humaine.

Nous utilisons la librairie *word-embeddings-benchmarks* [2] pour réaliser nos évaluations (Jastrzebski *et al.*, 2017). Nous comparons nos résultats à ceux obtenus avec des plongements pré-entraînés accessibles en ligne en utilisant cette librairie. Les plongements utilisés sont ceux obtenus via les méthodes Glove (Pennington *et al.*, 2014), NMT (Hill *et al.*, 2014), HDC et PDC (Sun *et al.*, 2015), Skip-gram (Mikolov *et al.*, 2013) et Lexvec (Salle *et al.*, 2016).

Les résultats Table 1 sont encourageants, ils montrent que notre approche est pertinente. Pour chaque tableau, nous comparons les résultats de notre méthode aux meilleurs résultats des méthodes de l'état de l'art citées, pour 50 et 300 dimensions. Sur deux corpus (en gras dans la Table), l'un exploité

2. https ://github.com/kudkudak/word-embeddings-benchmarks

| Benchmark | Notre modèle | État de l'art dim=300 | État de l'art dim=50 |
|---|---|---|---|
| | | Similarité | |
| MEN | 0.650 | 0.809 | 0.720 |
| SimLex | 0.364 | 0.427 | 0.309 |
| RG65 | **0.803** | 0.790 | 0.763 |
| | | Catégorisation | |
| ESSLI1a | 0.75 | 0.818 | 0.773 |
| ESSLI2b | **0.775** | 0.750 | 0.775 |
| ESSLI2c | 0.6 | 0.667 | 0.556 |

TABLE 1 – Résultats sur les tâches de catégorisation et de similarité en comparaison de l'état de l'art.

pour la tâche de similarité, l'autre pour la tâche de catégorisation, notre méthode obtient des résultats comparables à celles de l'état de l'art auxquelles nous nous comparons. Dans le reste des cas, notre méthode obtient des résultats supérieurs aux performances des approches de l'état de l'art paramétrés pour retourner des vecteurs en dimension $50$, mais inférieurs lorsque ces vecteurs sont en dimension $300$. En accord avec l'état de l'art nos meilleurs résultats sont obtenus pour les plus grandes tailles de fenêtre ($f = 5$ dans notre cas).

# 4 Discussion et perspectives

Nous avons décrit une méthode originale d'apprentissage de plongements lexicaux basée sur une approche réseaux complexes. Nous proposons d'utiliser les communautés détectées sur le réseau de co-occurrences représentant le corpus comme dimensions de nos plongements. Les vecteurs sont ensuite directement extraits de la distribution des liens de chaque nœud à travers la structure communautaire. Les résultats qualitatifs et quantitatifs montrent la pertinence de l'approche qui obtient des scores comparables à l'état de l'art. Néanmoins, une étude avec les mêmes méta-paramètres (corpus, taille de fenêtre) semblent nécessaire pour se situer exactement par rapport à l'état de l'art.

Cette approche a pour avantage de fournir des dimensions qui sont des objets concrets, physiquement existants : les communautés. Ces dimensions semblent donc interprétables : il est possible de consulter le contenu de ces communautés, de les étiqueter avec des éléments caractéristiques. Néanmoins, cela ne garantit pas l'interprétabilité des vecteurs appris. Pour que ces vecteurs soient interprétables, il s'agit à notre sens de réunir deux conditions. La première, est de disposer d'un étiquetage suffisamment précis pour qu'il soit tout à fait compréhensible. La seconde nécessite d'avoir un vecteur de taille raisonnable, ou du moins un vecteur creux afin de ne pas avoir trop de communautés à inspecter. Ces questions sont en lien direct avec le paramétrage des algorithmes de détection de communautés et constituent des perspectives directes de notre travail. Nous souhaitons en effet travailler à évaluer l'interprétabilité des vecteurs extraits par notre méthode par des humains.

De plus, notre méthode permet l'extraction rapide du plongement d'un mot ou d'une expression. Le calcul de ce vecteur découle en effet directement de la connectivité du nœud qui représente le mot, de la façon dont ses liens se dispersent au sein de la structure communautaire. Ainsi, le calcul du vecteur d'un nouveau mot ou d'une expression composée ne nécessite pas de réapprendre un modèle, mais

simplement d'ajouter le terme au réseau pour extraire le vecteur.

Enfin, les langues évoluent avec le temps : le sens des mots change ou de nouveaux sens apparaissent. Ces évolutions de la langue ont été décrites, notamment par Bloomfield 1983; Mitra *et al.* 2015. Des travaux considèrent des méthodes automatiques basées sur les plongements lexicaux et de grands corpus temporels pour la détection de ces néologismes sémantiques (Tang, 2018). Ces méthodes peuvent être séparées en deux classes. La première classe est celle des méthodes *diachroniques* : elles discrétisent le temps et séparent ainsi le corpus en plusieurs sous-corpus. Sur chacun de ces sous-corpus, les auteurs proposent d'apprendre des plongements lexicaux puis d'aligner les espaces appris entre les sous-corpus deux à deux (Hamilton *et al.*, 2016). Ces approches sont basées sur l'hypothèse très forte qu'il est possible d'aligner des espaces différents issus d'algorithmes non déterministes aboutissant à des résultats sous-optimaux. La seconde classe, celle des méthodes *dynamiques*, propose une optimisation globale de tous les plongements du vocabulaire à travers le temps, aboutissant à un problème gourmand en calcul et très difficile (Bamler & Mandt, 2017). Notre approche peut permettre d'ouvrir le champ à de nouveaux travaux basés sur les algorithmes de détection de communautés incrémentaux (Xie *et al.*, 2013). Cela permettrait ainsi de s'abstraire d'une optimisation globale coûteuse, et de contourner l'hypothèse d'alignement diachronique des espaces.

# Références


BAMLER R. & MANDT S. (2017). Dynamic word embeddings. In *International Conference on Machine Learning*, p. 380–389.

BLOOMFIELD L. (1983). *An introduction to the study of language*, volume 3. John Benjamins Publishing.

DAO V.-L., BOTHOREL C. & LENCA P. (2018). Estimating the similarity of community detection methods based on cluster size distribution. In *International Workshop on Complex Networks and their Applications*, p. 183–194 : Springer.

DOROGOVTSEV S. N. & MENDES J. F. F. (2001). Language as an evolving word web. *Proceedings of the Royal Society B : Biological Sciences*, **268**(1485), 2603–2606.

GAO Y., LIANG W., SHI Y. & HUANG Q. (2014). Comparison of directed and weighted co-occurrence networks of six languages. *Physica A : Statistical Mechanics and its Applications*, **393**, 579–589.

HAMILTON W. L., LESKOVEC J. & JURAFSKY D. (2016). Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv :1605.09096*.

HILL F., CHO K., JEAN S., DEVIN C. & BENGIO Y. (2014). Embedding word similarity with neural machine translation. *arXiv preprint arXiv :1412.6448*.

I CANCHO R. F. & SOLÉ R. V. (2001). The small world of human language. *Proceedings of the Royal Society of London B*, **268**(1482), 2261–2265.

JASTRZEBSKI S., LEŚNIAK D. & CZARNECKI W. M. (2017). How to evaluate word embeddings ? on importance of data efficiency and simple supervised tasks. *arXiv preprint arXiv :1702.02170*.

LEVY O., GOLDBERG Y. & DAGAN I. (2015). Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, **3**, 211–225.

MASUCCI A. P. & RODGERS G. J. (2006). Network properties of written human language. *Physical Review E*, **74**(2), 026102.

MATULA D. W. & BECK L. L. (1983). Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM*, **30**(3), 417–427.

MIKOLOV T., SUTSKEVER I., CHEN K., CORRADO G. S. & DEAN J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, p. 3111–3119.

MITRA S., MITRA R., MAITY S. K., RIEDL M., BIEMANN C., GOYAL P. & MUKHERJEE A. (2015). An automatic approach to identify word sense changes in text media across timescales. *Natural Language Engineering*, **21**(5), 773–798.

NEWMAN M. E. (2004). Analysis of weighted networks. *Physical review E*, **70**(5), 056131.

NEWMAN M. E. & GIRVAN M. (2004). Finding and evaluating community structure in networks. *Physical review E*, **69**(2), 026113.

PALLA G., DERÉNYI I., FARKAS I. & VICSEK T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, **435**(7043), 814–818.

PENNINGTON J., SOCHER R. & MANNING C. (2014). Glove : Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, p. 1532–1543.

RAGHAVAN U. N., ALBERT R. & KUMARA S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, **76**(3).

SALLE A., IDIART M. & VILLAVICENCIO A. (2016). Enhancing the lexvec distributed word representation model using positional contexts and external memory. *arXiv preprint arXiv :1606.01283*.

SARKAR S. & DONG A. (2011). Community detection in graphs using singular value decomposition. *Physical Review E*, **83**(4).

SCHNABEL T., LABUTOV I., MIMNO D. & JOACHIMS T. (2015). Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, p. 298–307.

SUN F., GUO J., LAN Y., XU J. & CHENG X. (2015). Learning word representations by jointly modeling syntagmatic and paradigmatic relations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, volume 1, p. 136–145.

TANG X. (2018). A State-of-the-Art of Semantic Change Computation. *arXiv preprint arXiv :1801.09872*.

XIE J., CHEN M. & SZYMANSKI B. K. (2013). Labelrankt : Incremental community detection in dynamic networks via label propagation. In *Proceedings of the Workshop on Dynamic Networks Management and Mining*, p. 25–32 : ACM.

YAN X., JEUB L. G., FLAMMINI A., RADICCHI F. & FORTUNATO S. (2018). Weight thresholding on complex networks. *arXiv preprint arXiv :1806.07479*.