Qualitative Evaluation of Word Embeddings: Investigating the Instability in Neural-Based Models

Bénédicte Pierrejean

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse - Jean Jaurès*

Présentée et soutenue le *8 Janvier 2020* **par :**

Bénédicte Pierrejean

Qualitative Evaluation of Word Embeddings: Investigating the Instability in Neural-Based Models

---

JURY

Ludovic Tanguy	Maître de Conférences, U. Toulouse – Jean Jaurès	Directeur
Olivier Ferret	Ingénieur Chercheur, CEA LIST	Rapporteur
Alessandro Lenci	Professor, University of Pisa	Rapporteur
Cécile Fabre	Professeure, U. Toulouse – Jean Jaurès	Examinatrice
Aurélie Herbelot	Assistant Professor, University of Trento	Examinatrice

---

École doctorale et spécialité :
CLESCO : Sciences du langage
Unité de Recherche :
CLLE-ERSS (UMR 5263)

Abstract

Distributional semantics has been revolutionized by neural-based word embeddings methods such as word2vec that made semantics models more accessible by providing fast, efficient and easy to use training methods. These dense representations of lexical units based on the unsupervised analysis of large corpora are more and more used in various types of applications. They are integrated as the input layer in deep learning models or they are used to draw qualitative conclusions in corpus linguistics. However, despite their popularity, there still exists no satisfying evaluation method for word embeddings that provides a global yet precise vision of the differences between models. In this PhD thesis, we propose a methodology to qualitatively evaluate word embeddings and provide a comprehensive study of models trained using word2vec. In the first part of this thesis, we give an overview of distributional semantics evolution and review the different methods that are currently used to evaluate word embeddings. We then identify the limits of the existing methods and propose to evaluate word embeddings using a different approach based on the variation of nearest neighbors. We experiment with the proposed method by evaluating models trained with different parameters or on different corpora. Because of the non-deterministic nature of neural-based methods, we acknowledge the limits of this approach and consider the problem of nearest neighbors instability in word embeddings models. Rather than avoiding this problem we embrace it and use it as a mean to better understand word embeddings. We show that the instability problem does not impact all words in the same way and that several linguistic features are correlated. This is a step towards a better understanding of vector-based semantic models.

Résumé

La sémantique distributionnelle a récemment connu de grandes avancées avec l'arrivée des plongements de mots (word embeddings) basés sur des méthodes neuronales qui ont rendu les modèles sémantiques plus accessibles en fournissant des méthodes d'entraînement rapides, efficaces et faciles à utiliser. Ces représentations denses d'unités lexicales basées sur l'analyse non supervisée de gros corpus sont de plus en plus utilisées dans diverses applications. Elles sont intégrées en tant que première couche dans les modèles d'apprentissage profond et sont également utilisées pour faire de l'observation qualitative en linguistique de corpus. Cependant, malgré leur popularité, il n'existe toujours pas de méthode d'évaluation des plongements de mots qui donne à la fois une vision globale et précise des différences existant entre plusieurs modèles. Dans cette thèse, nous proposons une méthodologie pour évaluer les plongements de mots. Nous fournissons également une étude détaillée des modèles entraînés avec la méthode word2vec. Dans la première partie de cette thèse, nous donnons un aperçu de l'évolution de la sémantique distributionnelle et passons en revue les différentes méthodes utilisées pour évaluer les plongements de mots. Par la suite, nous identifions les limites de ces méthodes et proposons de comparer les plongements de mots en utilisant une approche basée sur les voisins sémantiques. Nous expérimentons avec cette approche sur des modèles entrainés avec différents paramètres ou sur différents corpus. Étant donné la nature non déterministe des méthodes neuronales, nous reconnaissons les limites de cette approche et nous concentrons par la suite sur le problème de l'instabilité des voisins sémantiques dans les modèles de plongement de mots. Plutôt que d'éviter ce problème, nous choisissons de l'utiliser comme indice pour mieux comprendre les plongements de mots. Nous montrons que le problème d'instabilité n'affecte pas tous les mots de la même manière et que plus plusieurs traits linguistiques permettent d'expliquer une partie de ce phénomène. Ceci constitue un pas vers une meilleure compréhension du fonctionnement des modèles sémantiques vectoriels.

Acknowledgments

Les années de thèse ont été marquées par de nouvelles rencontres mais aussi par le soutien de nombreuses personnes que je souhaite remercier ici.

Tout d'abord je souhaite remercier mon directeur de thèse, Ludovic Tanguy, qui m'a fait confiance pour effectuer cette thèse. Je le remercie aussi pour sa disponibilité et son accompagnement tout au long de mon travail de recherche qui ont permis d'en assurer le bon déroulement. Son sens du détail et ses nombreuses relectures ont permis d'aboutir à ce manuscrit.

Je souhaite également remercier chaleureusement Olivier Ferret et Alessandro Lenci d'avoir accepté d'être les rapporteurs de cette thèse ainsi que Cécile Fabre et Aurélie Herbelot d'avoir accepté d'être examinatrices.

Je tiens à remercier plus particulièrement Aurélie pour ses retours sur mon travail et pour les collaborations que nous avons commencées à mener.

Je remercie également tous les membres du laboratoire CLLE-ERSS pour leur accueil. Je tiens plus particulièrement à remercier Anne Przewozny qui a été l'une des premières à me donner le goût de la recherche, Franck Sajous, Nabil Hathout, Bruno Gaume, Basilio Calderone et Josette Rebeyrolle. Je remercie plus particulièrement Cécile Fabre et Lydia-Mai Hodac qui m'ont beaucoup appris tant au niveau de l'enseignement qu'au niveau de la recherche. Je remercie plus particulièrement Mai pour son soutien et les discussions partagées, mais aussi d'avoir pris le temps de relire certaines parties de ma thèse.

Bien sûr je souhaite également remercier tous les doctorants avec qui j'ai vécu une partie de cette aventure que ce soit au travers du CEPEL, du volontariat à ESSLI ou au quotidien dans le laboratoire: Aleksandra, Camilla, Filip, Giusi, Julie H., Julie R., Lyanne, Marine, Maxime, Mélanie, Natalia, Nataly, Nour, Rogelio et Yizhe.

Et puis merci à Léa d'abord pour m'avoir si bien accueillie dans le bureau B503, ensuite d'avoir opéré le rôle de *debugging rubber duck* pendant les derniers mois de rédaction. Merci aussi à Pavel pour son soutien et ses petits interludes sémantiques ("Le-saviez-vous" ?). Les petits déj du B503 crew et les soirées à discuter (entre autres) de linguistique font partie des souvenirs de thèse que je chérirai. Je souhaite aussi remercier Marc-Philippe, Daniele et Efisio de m'avoir fait découvrir

les meilleures pizzas de Toulouse ainsi que la richesse musicale italienne. Enfin, je souhaite remercier Lison (mamen) pour tellement de choses: merci de m'avoir fait découvrir la joie de se déplacer à vélo (et de faire des dérapages), d'avoir partagé mon amour pour le café et la nourriture libanaise, d'avoir été là tant pour les fous-rires que pour les moments difficiles, pour les discussions philosophiques, pour ces moments privilégiés au café de psycho, à l'Autruche et bien d'autres encore !

Merci aussi à tous les copains du Gers et plus particulièrement à Hugo, Fanny, Simon, Loïc et Éléonore (le Toulouse crew) pour les petits rendez-vous qui faisaient du bien. Merci aussi à Manu, pour le soutien et les paroles rassurantes. Merci à Prim pour les intervalles musicaux qui réchauffent toujours mon coeur. Merci aussi aux copains rencontrés à Nomao, Jonathan et Oussama, qui m'ont donné le goût d'aller plus loin dans le TAL et qui sont toujours là malgré la distance. Et puis merci aux copains canadiens, en particulier Luiza pour le soutien et les rigolades.

Je souhaite aussi remercier ma famille. Merci à ma grand-mère de Metz, qui a été une source d'inspiration pour mener à bien cette thèse et qui était toujours de bon conseil. Merci aussi à ma grand-mère de Toulouse. Et puis bien sûr je souhaite aussi remercier mes parents, mon frère et mes soeurs, d'avoir cru en moi et de m'avoir soutenue et encouragée dans cette longue aventure. Surtout merci à ma petite soeur Marie (le poisson pilote), qui a su me rassurer et m'encourager dans les moments difficiles et partager avec moi les difficultés de la thèse (allez t'y es presque !).

Enfin, Mahmoud (ya 3omre) merci pour tout et tellement plus. Merci d'avoir su prendre soin de moi notamment en me concoctant de bon petits plats qui m'ont aidé à tenir le coup. Merci pour ton soutien, ta confiance, tes encouragements et ta patience sans faille surtout dans ces derniers mois difficiles.

Contents

List of Tables

List of Figures

Introduction

Motivation

The hypothesis behind distributional semantics stating that words sharing similar contexts have similar meanings (Harris, 1954) led to the modern distributional semantics where vector space is used to project meaning using observations automatically extracted from corpora. The past few years, distributional semantics has known a rapid growth mainly due to the introduction of distributional neural-based models, also known as word embeddings. These models rapidly became attractive because they are easy to train and to use, whether to compute semantic similarity between words or to be integrated in deep learning models. With the release of word2vec (Mikolov et al., 2013c), word embeddings rapidly became the most commonly used models to represent meaning. Several improvements of word2vec were proposed, as well as different ways to learn word embeddings (e.g. GloVe (Pennington et al., 2014)). More recently, contextual embeddings (e.g. ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019)) have become very popular, overshadowing earlier methods. While new techniques are released every year (or even every few months), there is still no evaluation method that provides a satisfying way to evaluate word embeddings from different perspectives and especially from a qualitative perspective.

One of the most common way to evaluate word embeddings is by using intrinsic methods that consist in comparing the similarity of selected pairs of words to human judgments. While this type of evaluation is convenient and easy to implement, it suffers from several biases. The ability of models to encode similarity is tested through human annotations often made without any context. This is a problem since the meaning of words is contextual, especially with the distributional hypothesis stating that meaning comes from the contexts of words. E.g. if we consider the word *tree*, it can be used to describe an element of nature or an algorithm depending on the context. As a consequence, the similarity evaluated in datasets might seems slightly artificial. Moreover, neural-based word embeddings are trained using non deterministic methods. As a consequence, training a model with the same hyperparameters and on the same data actually yields different se-

mantic representations. These variations are not detected by intrinsic evaluation datasets.

While using quantitative methods is useful to develop and prototype models, we are more interested in understanding how word representations are learned. As a consequence, we prefer to focus on the different hyperparameters the semantic representations are learned from, whether this means the corpus or the different mechanisms specific to word2vec.

When evaluating models by integrating a linguistic point a view, it is difficult to reduce observations within boundaries delimited by datasets. By integrating a linguistic point of view to the evaluation process, we do not restrict our observations to limits set by datasets. Rather, we propose to adopt an exploratory approach where we focus on comparing how models differ from a semantic point of view. To do so we chose to observe nearest neighbors of words.

Nearest neighbors of words are interesting because they provide immediate feedback on the meaning of a word as it was captured by a model from a given corpus. Let's look at a specific example to illustrate that. By evaluating using intrinsic datasets, we assess if two pairs of words are more similar than two other pairs of words. E.g. we expect the pair *magician* and *wizard* to be more similar than the pair *king* and *cabbage*[1]. By looking at nearest neighbors, we directly observe the words that are the most similar to a given word. Then we can imagine that in a generic corpus, *magician* would be more similar to words like *wizard*, *sorcerer* etc. As a consequence, we directly get an idea of the meaning captured in the corpus and we can determine if the fact that the model considered these words similar is appropriate or not. By observing nearest neighbors, we do not rely on any *a priori* construct of meaning.

Despite providing immediate and direct feedback on a word semantic representation, nearest neighbors also suffer biases. E.g. it is difficult to assert that a neighbor is of good quality. Moreover, it is sometimes challenging to explain why a nearest neighbor is considered similar to a target word. As a consequence, depending only on the direct observation of nearest neighbors in a model might not be appropriate. We propose instead to compare the variation in nearest neighbors across models, i.e. given two models, we compare what changes in the neighborhood of a given word across these two models. In this way, we are able to directly identify words that are highly impacted by variation. This also provides a starting point for detailed explorations. Moreover, by using this method to compare models trained with different hyperparameters, we might be able to relate nearest neighbors variation to the impact of different hyperparameters (whether this means using a different corpus or different settings specific to the system used for training).

[1]Examples selected from WordSim-353 Finkelstein et al. (2002).

We also propose to use this method to examine the instability phenomenon that is inherent to word embeddings trained using neural-based methods. We specifically focus on word2vec. While some works have focused on this problem, especially on measuring the amplitude of the phenomenon, it remains a poorly known and often disregarded phenomenon. Rather than trying to fix this problem, we propose here to embrace it and use it as a way to better understand word embeddings. If we consider the resistance to instability, i.e. the resistance to random processes, as an indicator of quality, we can identify stable and unstable zones in the lexical space. Moreover, we go further by questioning the linguistics aspects of the stability and instability of words. We are able to better understand word embeddings by being able to correlate linguistic features to words variation.

Publications

The research conducted during this PhD and the participation to research projects led to the following publications:

Pierrejean, B. and Tanguy, L. (2018a). Étude de la Reproductibilité des Word Embeddings : Repérage des Zones Stables et Instables dans le Lexique. In *TALN*, Rennes, France

Pierrejean, B. and Tanguy, L. (2018b). Predicting Word Embeddings Variability. In *Proceedings of the 7th Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 154–159, New Orleans

Pierrejean, B. and Tanguy, L. (2018c). Towards Qualitative Word Embeddings Evaluation: Measuring Neighbors Variation. In *Proceedings of NAACL-HLT 2018: Student Research Workshop*, pages 32–39, New Orleans

Gaume, B., Tanguy, L., Fabre, C., Ho-Dac, L.-M., Pierrejean, B., Hathout, N., Farinas, J., Pinquier, J., Danet, L., Péran, P., De Boissezon, X., and Jucla, M. (2018). Automatic Analysis of Word Association Data from the Evolex Psycholinguistic Tasks Using Computational Lexical Semantic Similarity Measures. In *13th International Workshop on Natural Language Processing and Cognitive Science (NLPCS)*, Krakow, Poland

Pierrejean, B. and Tanguy, L. (2019). Investigating the Stability of Concrete Nouns in Word Embeddings. In *Proceedings of the 13th International Conference on Computational Semantics*, pages 65–70, Gothenburg, Sweden

Gaume, B., Ho-Dac, L.-M., Tanguy, L., Fabre, C., Pierrejean, B., Hathout, N., Farinas, J., Pinquier, J., Danet, L., Péran, P., De Boissezon, X., and Jucla, M. (2019). Towards a Computational Multidimensional Lexical Similarity Measure

for Modeling Word Association Tasks in Psycholinguistics. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 71–76

Thesis Outline

We chose to organize this thesis in two parts.

In the first part we present Distributional Semantics. Chapter 1 focuses on the evolution of distributional semantics, from the use of vector space to represent meaning to the introduction of neural-based word embeddings models. We present hyperparameters that are important when training distributional semantics models. We then present in details word2vec and its different hyperparameters.

In chapter 2 we present the different existing methods to evaluate word embeddings. We particularly focus on intrinsic evaluation methods by discussing selected "traditional" evaluation datasets and the common problems they encounter. We also present alternatives to these datasets that were developed specifically for the evaluation of distributional semantics models.

Chapter 3 presents selected works that evaluate word embeddings using alternatives to intrinsic datasets. We also explain why we chose to evaluate word embeddings using qualitative methods.

In the second part, we focus on the method we used to evaluate word embeddings using a qualitative point of view.

Chapter 4 details the first experiment we conducted that focused on training word embeddings using different hyperparameters. We first give an overview of the different parameters we selected. We also present several studies that have investigated these parameters. Then, we present the method we use to compare variations existing between different models. Finally, we present the results of this experiment, first from a quantitative point of view and secondly by using the comparison of nearest neighbors.

In chapter 5 we discuss the internal instability phenomenon, which leads to internal variation when training word embeddings using the same hyperparameters. We investigate the amplitude of this phenomenon on three different corpora.

Finally in chapter 6 we chose to investigate neighbors instability as a linguistic phenomenon by looking at different features that could explain the variation. We are able to identify stable and unstable zones in the lexical space. We go further by being able to correlate these zones of stability and instability with selected linguistic features.

Part I

Distributional Semantics

Chapter 1

Distributional Semantics Evolution

Contents

1.1 Introduction

To better understand distributional semantics today, we wanted to go back to its foundations and origins. In this chapter, we propose to first look at the distributional hypothesis in the view of American structuralists. Then we show how the distributional hypothesis was used to model meanings using vector spaces. Along with modeling meaning in space, we focus on the notion of context, a key element in distributional semantics that has a considerable impact on the model trained independently of the type of method used to create models. Finally we present word embeddings and we focus more specifically on word2vec (Mikolov et al., 2013a), which is the main tool we use throughout this thesis to train word embeddings.

1.2 The distributional hypothesis

The distributional hypothesis is rooted in the American structuralist movement, with observations made by Harris (1954) and Firth (1957) that words appearing in similar contexts have similar meanings (Jurafsky and Martin, 2018; Turney and Pantel, 2010; Lenci, 2008). The following example illustrates the distributional hypothesis:

> If A and B have almost identical environments except chiefly for sentences which contain both, we say they are synonyms: *oculist* and *eye-doctor*. If A and B have some environments in common and some not (e.g. *oculist* and *lawyer*) we say that they have different meanings, the amount of meaning difference corresponding roughly to the amount of difference in their environments.
> (Harris, 1954, 157).

Thus *oculist* and *eye-doctor* are synonyms because they share almost the exact same contexts. However, words with different meanings can still share similar contexts (like *oculist* and *lawyer*) and the distribution of an element is defined by the sum of all the environments it appears in (Harris, 1970). It is also important to notice that the information we get about the meaning of a word is acquired relatively to another word. This means that the harissian hypothesis focuses on meaning not for a word by itself but by always comparing to another word.

Firth's work also contributed to shape the distributional hypothesis we know today and it is common to define the principle of distributional semantics with the following famous quote:

> You shall know a word by the company it keeps.
> (Firth, 1957)

Firth's hypothesis relied on the notion of collocation to show that distributional contexts can be used to explain the behavior of a word. By looking at the different contexts and environments a word appears in, it is possible to get information about its different senses (Pulman, 2013). We can differentiate this approach from Harris' because it focuses on a different aspect of meaning. Observing the contexts of a word gives an idea of its meaning but most importantly it provides information about its degree of polysemy. Moreover, Firth only considers contexts of a word by itself and not necessarily in contrast with other words. For Harris, the contexts that two words share give information about their degree of similarity.

We want to point out here the importance of the corpus in both approaches. The degree of similarity of words and the different meanings of a word are extracted from a corpus, through the different contexts observed. Those contexts are highly dependent on the corpus used and two words are similar or dissimilar based on their contexts in a given corpus. Their contexts might be completely different in another corpus. Let's take the example of the words *apple* and *pear*. If we search both words in a generic corpus, like the BNC[1], we find the following sentences:

(1) (...), it no longer makes cider but **apple** juice. (HMH)

(2) Yeah **pear** juice, I like (...). (KC5)

Both *apple* and *pear* are used with the word *juice* and refer to fruits. Their contexts are similar and, following the harissian hypothesis, we can deduce that *apple* and *pear* are very similar. They actually are co-hyponyms and are both hyponyms of *fruit*. However if we look up the same words in the Medical Web Corpus[2], we observe the following uses:

(3) At the front of the throat, a laryngeal prominence, known as the Adam's **apple** (...) (Web-med 3)

(4) (...) such as peach, pineapple, and **pear** (...) (Web-med 59)

While *pear* still refers to the fruit, *apple* has a different meaning in example (3) where it refers to a body part. It is thus very important to always consider that the similarity observed by distributional semantics is corpus-dependent.

Later on, Rubenstein and Goodenough (1965) investigated the distributional hypothesis by examining how synonymy is related to the similarity of contexts. According to them, it is obvious that words sharing very similar contexts have very similar meanings and words having very dissimilar contexts have very dissimilar meanings. However, they were particularly interested in words that do not fall in these two categories. The novelty of their approach was that rather than

[1] `https://www.english-corpora.org/bnc/`

[2] `https://www.sketchengine.eu/medical-web-corpus/`

only making observations of words in contexts, they integrated the judgments of the degree of synonymy made by human annotators. They confirmed that there is a positive relation between the degree of similarity of words and the number of contexts they shared. However, they could not prove that it was possible to quantify the degree of similarity using the amount of shared contexts.

We saw that the distributional hypothesis actually encompasses two dimensions. Both consider that the observation of words contexts provides information about their semantic properties. First, following Harris' hypothesis we can determine how similar two words are. Secondly, according to Firth, contexts provide information about the degree of polysemy of a word. In the next section we present how the distributional hypothesis has been used in computational linguistics.

1.3 Distributional Semantics

The distributional hypothesis proposes to observe word meanings by quantifying their contexts, i.e. by counting the different contexts a word appears with. As such, it provided a suitable framework for computational linguistics that makes it easy to automatically extract and quantify the contexts of a word. Moreover, multi-dimensional space can be used to model meaning. The use of vector spaces in semantics is attractive because vectors provide a natural mechanism for computing distance and similarity. The notion of distance allows to discriminate how the meanings of two words are different (Clark, 2015). In this section we present the representation of meaning in vector space. We also give an overview of the creation of distributional semantic models.

1.3.1 Meaning in vector space

The spatial representation of meaning is rooted in the cognitive theory identified by Lakoff and Johnson (1980), stating that because we are "embodied beings", we conceptualize and interpret abstract concepts through our spatio-temporal knowledge of the world (Sahlgren, 2006).

Osgood et al. (1957) were amongst the first to propose the representation of meaning in a multi-dimensional space with their work on the semantic differential approach. They used 3 dimensions to represent affective meaning: valence (that corresponds to the pleasantness of the stimulus), arousal (intensity of the emotion the stimulus provokes) and dominance (degree of control the stimulus exerts). Words are then represented with those 3 dimensions and can be projected in a 3-dimensional space (Sahlgren, 2006; Jurafsky and Martin, 2018).

Later on, vector space models based on co-occurrence matrices were implemented for SMART (Salton, 1971), an information retrieval system. The idea

was to represent documents as vectors in a vector space. Two documents that are supposedly similar have the same words thus their vectors are geometrically close. As a consequence, closer points represent documents that are semantically similar (Turney and Pantel, 2010; Jurafsky and Martin, 2018). The success of Vector Space Models in information retrieval motivated their use for other semantic tasks in Natural Language Processing. Deerwester et al. (1990) found that is was possible to measure word rather than document similarity (Turney and Pantel, 2010). In this case, word co-occurrences are used to compute similarity between words. Schütze (1998) used distributional techniques to learn word senses without any external resources. He proposed a corpus-based method, where all word representations are derived from a large corpus. Vectors are used to represent words, contexts and senses. Rapp (2003) also proposed an automatic method for word sense induction based on distributional similarity between words in a corpus (Pulman, 2013).

In the literature, several names are used to refer to models created from co-occurrences matrices. Sahlgren (2006) describes them as *word space models* in reference to Schütze (1993). Turney and Pantel (2010) refer to them as *vector space models*. Throughout this thesis we use *distributional semantics models* (henceforth DSMs) following Baroni and Lenci (2010) to refer to all types of models trained in distributional semantics. In the next section we present the main principles behind DSMs.

1.3.2 Distributional semantics models

Rather than manually encoding semantic relations between words, DSMs provide a way to automatically extract knowledge from corpora. In the most straightforward models, the semantic representations of words correspond to vectors where elements are the number of occurrences of the target word in different contexts (Turney and Pantel, 2010). Because the vector values are based on the number of co-occurrences, these models are created using the number of co-occurrences and are often referred to as count-based models. Figure 1.1 shows an example of the raw counts of words with different contexts in Wikipedia. Each row of the table corresponds to a word being described, the target word. The columns list all words appearing in the corpus. They constitute the contexts. For each target word, the number of co-occurrences[3] is reported. Thus each vector corresponds to the occurrences of the target word with contexts words. Using the number of co-occurrences, it is then possible to compare how similar words are. E.g. the vector of *digital* is circled in red and we can see that *digital* appears more with *computer* and *data* (tech-related words) than with *pie* and *sugar*. On the contrary,

[3]We will see in details what co-occurrences are in section 1.3.4.

the word *cherry* appears 8 times with *data* and 442 times with *pie*. The word *information* appears 5 times with *pie* and 3982 times with *data*. By just looking at these two words and their contexts, we already know that even though they share some contexts, *cherry* and *information* are not similar.

	aardvark	...	**computer**	**data**	**result**	**pie**	**sugar**	...
cherry	0	...	2	8	9	442	25	
strawberry	0	...	0	0	1	60	19	
digital	0	...	1670	1683	85	5	4	
information	0	...	3325	3982	378	5	13	

Figure 1.1: Example of co-occurrence vectors for words in the Wikipedia corpus, taken from Jurafsky and Martin (2018).

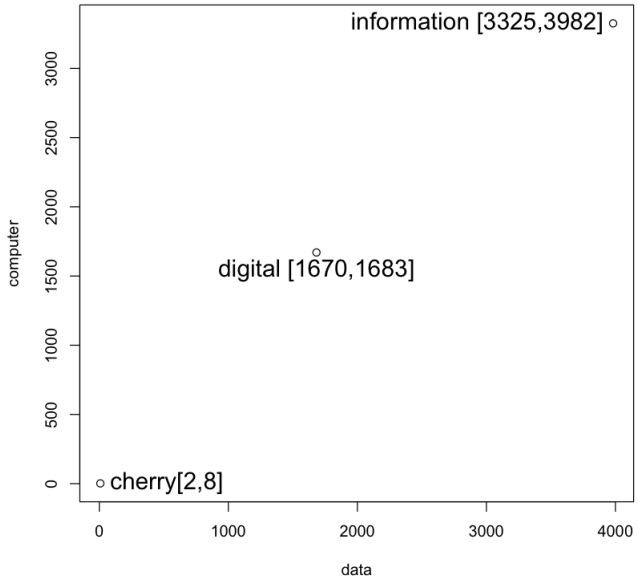


Figure 1.2: Example of the projection of the words *digital*, *information* and *cherry* using *computer* and *data* as dimensions.

Using the values from figure 1.1, we represented in figure 1.2 the words *digital*, *information* and *cherry* using the dimensions *computer* and *data*. The proximity in space represents the proximity in meaning and thus we can see that *information* is closer to *digital* than *cherry*. Because vectors are constituted of a large number of dimensions, it is not possible to visualize them all. Usually methods such as PCA

(Principal Component Analysis) are used to visualize the vectors by reducing the number of dimensions while keeping a maximum amount of information. When visualizing words in multiple dimensions, it is important to remember that only the proximity between words can be interpreted as an indicator of similarity. The position of words in space (e.g. the fact that words are on the left or on the right in the vector space) cannot be interpreted.

1.3.3 Measuring similarity

We just saw that words whose vectors are close in the semantic space are considered distributionally similar. To compute the similarity between vectors, it is very common to use cosine similarity which computes the similarity of two words based on the dot product of their vectors. The cosine similarity[4] is defined as follows:

$$cosine(v, w) = \frac{v \cdot w}{|v||w|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}} \tag{1.1}$$

where v and w are two vectors, and v_i and w_i correspond to the elements of the vectors. $|v|$ represents the norm of the vector and this normalization causes the measure to range from 0 (words that are very different) to 1 (words that are very similar).

Other measures exist to compute similarity, such as Euclidean distance, but cosine similarity is the most popular one. Using cosine similarity, given a target word it is then possible to compute its similarity with all the other words in the vector space. The words having the closest similarity with a word are called its nearest neighbors. For example, for a model built on a generic corpus we can imagine that the nearest neighbors of the word *cat* will be *dog*, *kitten*, *pet* etc. because they will appear in many similar contexts (e.g., *eat*, *drink*, *run* etc.). By retrieving nearest neighbors, we directly get an idea about the semantic representation of a word.

We gave an overview of distributional semantics and of the way multi-dimensional spaces are used to compute semantic similarity between words. We saw that the distributional hypothesis is based on the contexts of words and we showed exampled of vectors using co-occurrences. In the next section we propose to take some time to reflect on the notion of context in distributional semantics. This notion is extremely important when building distributional models since it highly influences the way semantic representations are learned.

[4]From Jurafsky and Martin (2018).

1.3.4 Contexts

Context is at the roots of the distributional hypothesis. As a consequence the meaning of a word is highly dependent on what we consider to be its context. In the previous section, we used simplified examples of DSMs built with raw co-occurrences. However, we did not go in details into what constitutes these co-occurrences. In this section we propose to examine the different types of contexts used in distributional semantics. Traditionally, we can distinguish between window-based contexts and dependency-based contexts.

1.3.4.1 Window-based contexts

Window contexts are based on co-occurrences of words extracted using a fixed size sliding window around the target word. The window can be oriented in only one direction (right or left) or it can be bi-directional and capture both words on the right and left of the target word. Usually the sentence is considered the maximum unit contexts are extracted from and the window stops at the end of a sentence.

Let's consider the following sentence:

(5) Montréal is a beautiful city in the fall season.

If we consider a bi-directional window of size 2, the contexts of the target word *city* are *beautiful*, *a*, *in*, *the*. If we consider a larger window of size 4 the contexts are *beautiful*, *a*, *is*, and *Montréal* on the left and *in*, *the*, *fall* and *season*. As a consequence, the size of the window is very important since it determines how much information is extracted for the target word. The amount of information then have a direct impact on the semantic representation of the word.

In the example above, words that might not be considered semantically interesting were also captured such as *a* or *the*. To avoid this, it is possible to filter contexts by removing highly frequent words or creating stop word lists (Curran, 2003).

Using window-based contexts presents several advantages. Their computational complexity is low and they are fast and easy to implement. They also present the advantage of being language independent (as long as we can rely on a tokenizer). However, window-based contexts can be considered slightly simplistic as they lack syntactic information about the words extracted. Similarly to a bag-of-word approach there is no consideration of word order and words appearing before or after the target word are considered the same (Curran, 2003; Padó and Lapata, 2007).

The window can take different shapes: it can be triangular or rectangular. A rectangular window is symmetrical and assigns the same weight to all contexts surrounding the target word while a triangular window weighs differently the words

surrounding the target word (Church and Hanks, 1990).

1.3.4.2 Dependency-based contexts

As an alternative to window-based contexts, it is possible to use contexts built using syntactic information. One of the first approaches dates back to Grefenstette (1994) who presented SEXTANT, a system using fine-grained syntactic contexts to compute similarity between words. Lin (1998) also proposed to redefine "local contexts" by using syntactic dependencies between words. Contrary to window-based contexts, words are extracted depending on the relation they share with the target word. As such, using syntactic analysis gives access to a wider range of contexts. The co-occurrences are not determined by a fixed size window. Moreover, contexts are more precise and word order can be mirrored in the semantic space (Padó and Lapata, 2007; Grefenstette, 1994). Let's consider the following example:

(6) Cats eat dry food.

Using window-based contexts (with a size of 3 or larger), *eat* and *food* are both extracted as two words co-occurring with *cat*. However using dependency-based contexts, the information about the syntactic relation can also be captured (*cat* is subject of *eat*, *food* is the direct object of *eat*).

Because they do not necessarily need to appear in a small, fixed window, dependency-based contexts can also provide a larger coverage of contexts (Padó and Lapata, 2007). Let's consider another example:

(7) I dislike my neighbor's new and noisy cat.

In example (7), using window-based contexts, a large window size would be needed to capture *dislike* as a context of *cat*. However, with dependency-based contexts, since *cat* is the direct object of *dislike*, it can be captured as one of its contexts.

Because the extracted contexts are different, investigations have been conducted on the type of information captured by both approaches. It was shown that dependency-based contexts are more likely to capture words that are taxonomically related, since words considered similar tend to have the same POS and the same dependency relations with other words. Window-based models tend to capture associative relations, since the extracted contexts consist of all words surrounding the target word with no specific distinction made between them (Fabre and Lenci, 2015).

Because it is possible to build a model by selecting different types of syntactic contexts, dependency-based contexts can offer more flexibility. However, because the syntactic information is obtained using a parsed corpus and might need ad-

ditional work to filter syntactic relations, dependency-based contexts can also be more expensive.

We presented the main differences between window-based and dependency-based contexts. The choice of the type of context when creating DSMs does not alter the representation method itself but reflects a choice made regarding the integration of information. As such, contexts are considered a central parameter when building DSMs. Context types are not the only parameters required when creating DSMs and a number of decisions needs to be made. In the next section we discuss weighting functions that can be applied to vectors.

1.3.5 Weighting functions

In section 1.3.2, we showed an example where co-occurrences were extracted using raw frequency. However, raw frequency is biased and is not a good way to discriminate between uses. Some words are highly frequent, e.g. the determiners *the* and *a*. Because they appear with many words they take too much importance compared to other words (Jurafsky and Martin, 2018). We saw in the previous section that it is possible to filter out some words (such as highly frequent words) using stop lists. However, vectors can still suffer sparcity problems, i.e. some of their elements being 0, because they are built using raw frequencies. To solve these problems it is possible to use alternatives rather than the raw co-occurrences. Several alternatives exist (tf-idf, log-likelihood etc.) but Positive Pointwise Mutual Information is a popular one. It compares the probability that two words appear together and the probability that they appear independently (Church and Hanks, 1990).

1.3.6 Dimensionality reduction

In the previous section, we saw that vectors often suffer sparcity problems. Several techniques exist to reduce the dimensionality of vectors, making the vectors dense. E.g. Singular Value Decomposition (SVD) was introduced by Deerwester et al. (1990) as a way to improve similarity measurements between documents. Landauer and Dumais (1997) applied it to word similarity (Latent Semantic Analysis) (Turney and Pantel, 2010). SVD consists in finding the most important dimensions, and limiting the number of dimensions helps improving the similarity measured between words (Jurafsky and Martin, 2018; Turney and Pantel, 2010). Following Deerwester et al. (1990), other dimensionality reductions techniques were introduced such as Non Negative Matrix Factorization (Lee and Seung, 1999) and Latent Dirichlet Allocation (Blei et al., 2003).

In the previous sections, we saw that building DSMs requires to make decisions concerning a number of parameters. Additional parameters exist. For example, using dependency-based contexts means that a parser is required. As a consequence, the choice of the parser can be considered as a choice of parameter. Another parameter which is not directly related to the models mechanics is the corpus used to create models as well as the pre-processing done on the corpus. We saw in section 1.2 that the meanings of words depend on the situation. In our case they depend on the corpus. As a consequence the selection of a corpus to create DSMs also constitutes a parameter choice that needs to be made. We will discuss this in more details in chapter 4.

We gave an overview of the main principles behind DSMs. We briefly discussed the fact that different information can be captured by models and we saw that many decisions need to be made when creating DSMs. Recently a new type of models has become increasingly popular: word embeddings. Because this thesis focuses on training models using word2vec, we present word2vec and its parameters in more details in the following section.

1.4 Word embeddings

Distributed representations of words, more commonly called word embeddings, are dense, low-dimensional and real valued vectors that are derived from neural language models (Turian et al., 2009). Neural Language Models were designed to overcome the high dimensionality problems when training language models for Natural Language Processing. By sharing statistical strength between similar words and their contexts, models learn to treat similarly words that have common features (Bengio and Ducharme, 2001; Goodfellow et al., 2016). Because they are dense, i.e. none of their elements is equal to 0, and have a low number of dimensions, word embeddings are often advertised as a more efficient alternative to count-based models.

Collobert and Weston (2008) were amongst the first ones to propose an efficient neural language model inspired by Bengio and Ducharme (2001). However, these models were computationally expensive. Several years later, Mikolov et al. (2013a) released word2vec, a toolkit that is based on neural networks to compute word embeddings in an efficient way.

When we started this work, other tools existed such as FastText (Bojanowski et al., 2016) but word2vec was the most popular by far and conference proceedings were flourishing with papers that proposed improved implementations of word2vec (Herbelot and Baroni (2017); Levy and Goldberg (2014a) to cite a few). Due to its high popularity it was thus obvious for us to use word2vec for our work. However, it is important to note that at the moment where we are writing this thesis, numerous

tools exist to train word embeddings and word2vec has mostly been replaced by contextual embeddings with BERT (Devlin et al., 2019) probably getting the most attention.

1.4.1 Word2vec

In this section, we present the main hyperparameters of word2vec. These hyperparameters correspond to decisions that need to be made when training word embeddings using word2vec,

1.4.1.1 Architecture

Word2vec is based on a neural network, a supervised learning method which is trained to correctly classify an item using a set of features. Neural networks consist of different layers of so-called artificial neurons. Each neuron is a simple calculation unit that provides a unique output based on several input values. The input corresponds to the set of features, the output correspond to the predicted classification. Building a neural language model consists in training a neural network to predict a word given its context. As a consequence, a process that was mainly non-supervised (building representations of words based on their distribution in a corpus) becomes a supervised learning method. The word embedding of a word is then the weights that were learnt in the hidden layer. As such, it is a dense vector of arbitrary dimensions that has "captured" the information gathered from the observation of the contexts of a word.

In word2vec, Mikolov et al. (2013a) proposed two different architectures to train word embeddings: Continuous Bag of Words (henceforth CBOW) and Skip-Gram (henceforth SG). Both architectures start learning from randomly initialized vectors. Then the vectors are updated depending on what is learnt from the training examples. The two architectures differ in the way they learn how to update vectors. The CBOW model is a bag of words architecture where word order is not considered. As shown in figure 1.3, training consists in correctly classifying the current word by using words surrounding the current word. The input and output are the values corresponding to words, represented by a sparse "one-hot" vector, where each dimension corresponds to a word in the corpus. As a consequence these vectors are mostly constituted of zeros except for the current word.

The number of context words that are used to learn from depend on the value of the window size parameter (we will describe this parameters in further details in section 1.4.1.4).

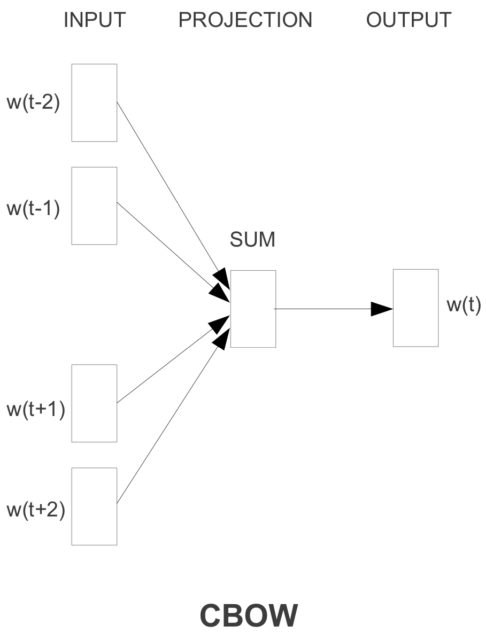


Figure 1.3: CBOW architecture, taken from Mikolov et al. (2013a).

The SG architecture is similar to CBOW except that instead of using surrounding words to predict the current word, as shown on figure 1.4, the words preceding and following the target word are predicted. The SG architecture is often preferred to CBOW because it gives better results on semantic and syntactic tasks (Mikolov et al., 2013b). It is mostly used with the Negative Sampling algorithm.

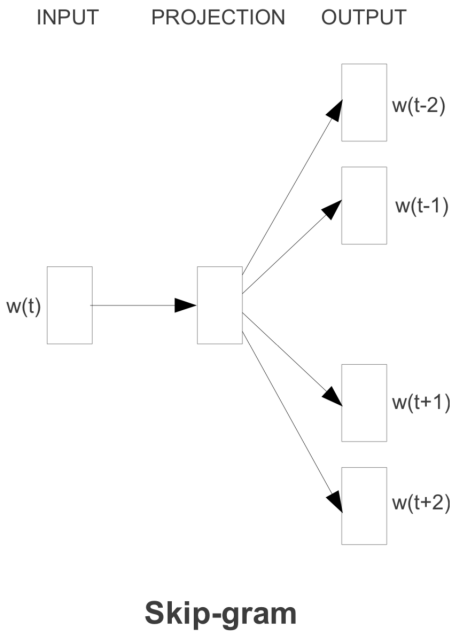


Figure 1.4: SG architecture, taken from Mikolov et al. (2013a).

1.4.1.2 Negative Sampling

Negative Sampling is an algorithm that was developed by Mikolov et al. (2013c) to improve the predictions of word2vec SG. The main idea behind this algorithm is that word2vec should learn both from positive and negative examples (contexts that the neural network is trained to reject as possible outputs). Positive examples correspond to contexts that were actually extracted from the corpus. Negative examples are a generated number k of negative examples for each positive training example (the default value of k is 5). To generate negative examples, the target word is associated to a noise word randomly selected from the corpus vocabulary according to its weighted unigram frequency (Mikolov et al., 2013c; Jurafsky and Martin, 2018). Figure 1.5 displays an example of a training sentence with the extracted positive examples for the target word t and the contexts c. The negative examples are given for k=2. The algorithm learns to maximize the similarity of positive examples and minimize the similarity of negative examples (Jurafsky and Martin, 2018).

```
... lemon,  a [tablespoon of apricot jam,      a] pinch ...
               c1         c2    t     c3       c4
```

positive examples +

t	c
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

negative examples -

t	c	t	c
apricot	aardvark	apricot	seven
apricot	my	apricot	forever
apricot	where	apricot	dear
apricot	coaxial	apricot	if

Figure 1.5: Example of positive and negative examples used to train word2vec SG (from Jurafsky and Martin (2018)).

1.4.1.3 Subsampling

In section 1.3.5, we mentioned that count-based models use different weighting schemes as alternatives to raw cooccurrence frequency values in order to balance the occurrences of frequent and infrequent words. Word2vec does something similar by *subsampling* very frequent words. The subsampling approach balances the occurrences of rare and frequent words by discarding words according to the following probability[5]:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{1.2}$$

$f(w_i)$ is the frequency of the word w_i and t corresponds to a chosen threshold (its default value is 10^{-3}). During training, some contexts words are not presented to the neural network. As a consequence, this approach is similar to removing high frequency words to avoid giving them too much importance when learning representations.

Using subsampling, the learning efficiency is improved since words with a frequency greater than t are not used as training examples. At the same time, the frequencies ranks are preserved (Mikolov et al., 2013c). However, in theory it means choosing to randomly remove certain occurrences.

1.4.1.4 Window size

The window size parameter is crucial when training DSMs because it determines the contexts provided to the model. We saw in section 1.3.4 that the window in DSM can be either bi-directional (symmetrical) or uni-directional (asymmetrical). In wor2vec the implementation of the window is symmetrical, i.e. the same number

[5]From Mikolov et al. (2013c).

of words are considered before and after the target word and it is only possible to control the size of the window (default value is 5). However, as noted by Levy et al. (2015), word2vec window parameter is not a simple symmetrical window. While the actual implementation is not mentioned in Mikolov et al. (2013a), the window is "dynamic". Contexts are weighted according to their distance from the target word. The intuition behind this implementation is that words that are closer to the target words are more important. As a consequence they are given more weight. Words that are further from the target word are considered less important and are thus assigned less weight. The weighting scheme is based on the distance of a context word from the target word. If the window has a size of 5, words surrounding the target word have a weight of $\frac{5}{5}$, $\frac{4}{5}$, $\frac{3}{5}$, $\frac{2}{5}$ and $\frac{1}{5}$ from closest to furthest (Levy et al., 2015). In other words, this corresponds to a triangular window.

1.4.1.5 Number of dimensions

The dimensions in word2vec models do not correspond to contexts and are not interpretable (Senel et al., 2018). The number of dimensions only corresponds to the size of the vectors. It might be challenging to set the optimal number of dimensions when training word embeddings. By default, in the C implementation of word2vec, the number of dimensions is set to 100. In chapter 4 we experiment with several different values.

1.5 Beyond word2vec

Word2vec makes it easy to train word embeddings efficiently and this partly explains the rapid success it encountered. Quite early on, many research papers proposed to improve word embeddings by injecting additional knowledge such as syntactic information (Levy and Goldberg, 2014a; Wang et al., 2015) or semantic knowledge (Yu and Dredze, 2014). At the same time, other dense representations models were introduced, such as GloVe, which is trained using the non-zeros elements of a word-by-word matrix (Pennington et al., 2014).

Because word2vec is a bit of a black box system, Levy and Goldberg (2014b) decided to investigate the way it learns semantic representations. They found that SG with negative sampling implicitly factorizes a word-context matrix. The cells of the matrix correspond to the PMI of a word with its contexts that is shifted using a global constant.

Choosing the appropriate parameters is also challenging with word2vec. As a consequence, several studies focused on investigating hyperparameters used when training word embeddings. E.g. Baroni et al. (2014) conducted a comparative

study of count-based and predictive models to see if the keen interest and the "triumphalist overtones surrounding predict models" were justified. They provided an extensive evaluation of DSMs on several semantic tasks across many parameters settings (window size, vectors dimensions, negative sampling rate). They found that predictive models yield very good results and recommended switching to predictive models. Chiu et al. (2016) investigated the influence of the corpus used as well as the impact of several hyper-parameters (negative sampling rate, subsampling rate, min-count threshold, learning rate, vector dimensions and window size) when training word embeddings for biomedical NLP. They found that the performance of word vectors changed with different corpora and that the model architecture and hyper-parameters settings had a huge impact on a model performance. Caselles-Dupré et al. (2018) investigated the importance of tuning hyper-parameters for recommendation systems. They found that optimizing the negative sampling rate, the number of epochs, the subsampling rate and the window size improves recommendations tasks performance.

Other studies compared the performance of models trained using word2vec with the performance of other distributional semantics models. Bernier-Colborne and Drouin (2016) provided an investigation of the impact of hyperparameters for DSMs in the context of specialized lexicography. Sahlgren and Lenci (2016) studied the impact of corpora size when training distributional semantics models. Among other results, they showed that word2vec did not perform well when trained on small corpora. Asr et al. (2016) examined several DSMs trained on child-directed speech. They showed that claiming one model is better than another is not as easy as it seems.

Finally, several studies investigated the influence of contexts used when training word embeddings. Melamud et al. (2016) provided an extensive evaluation of the influence of different context types when training skip-gram word embeddings. Li et al. (2017) provided a systematical investigation of different syntactic context types when training word embeddings.

Lots of works also focused on adapting word2vec to different types of input. Levy and Goldberg (2014a) presented a modified version of word2vec skip-gram that uses dependency-based context to train embeddings. Tissier et al. (2017) presented Dict2vec, a new approach to learn word embeddings from lexical dictionaries derived from word2vec skip-gram. Herbelot and Baroni (2017) implemented Nonce2Vec, a tool inspired from word2vec architecture to learn new words from tiny data.

Recently, we observed parallel research trends. On one hand, with the growing interest for deep learning, more and more people are not necessarily questioning the word embeddings they are using and mainly use pre-trained word embeddings. Most word embeddings implementations actually propose several pre-trained mod-

els (either trained on different corpora or for different languages). The main idea is that when training word embeddings, the more data the better. Since large corpora are not necessarily accessible to everyone, providing pre-trained models help making word embeddings "mainstream". Recently, the release of ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) even reinforced this impression that only pre-trained embeddings are needed since both models are not as straightforward to train as word2vec (and others) and are prohibitively expensive to train. On the other hand, we see people trying to make sense out of word embeddings and investigating the role and influence of linguistics in word embeddings models (see for example Andreas and Klein (2014); Tenney et al. (2019)).

Although giving access to better performing models seems necessary, one main drawback is that those models do not provide access to a lot of linguistic information that was previously accessible with count-based models. For example, count-based models permit lots of freedom in terms of the types of contexts selected and offer the possibility to observe the actual contexts that are used to build the model. Reconstructing contexts with word2vec is more challenging since word2vec algorithm transforms the input used for training, e.g. subsampling removes highly frequent words. Count-based models present the advantage of providing a controlled environment where understanding the impact of a parameter and investigating linguistic phenomena is easier.

In the next chapter we review selected evaluation methods for distributional semantics models. We also discuss some problems existing with those methods.

Chapter 2

Distributional Semantics Evaluation

Contents

2.1 Evaluating distributional semantics models

In the previous chapter, we gave an overview of distributional semantics and of different techniques available to build models. We saw that DSMs encode information about the similarity of words and that words appearing in similar contexts have similar vectors. However if DSMs are expected to encode semantic information about words, it is difficult to understand exactly what type of information

they capture and carry. One major issue that comes along with the use of DSMs is to find an appropriate way to evaluate them. Whether we are talking about neural-based models (word embeddings) or about traditional count-based models, evaluating DSMs is not a recent problem and it has been investigated for several decades. However, there is still no ideal way to evaluate DSMs and yearly calls for papers from workshops such as RepEval (Workshop on Evaluating Vector Space Representations for NLP) are a good illustration of that problem:

> Models that learn real-valued vector representations of words, phrases, sentences, and even document are ubiquitous in today's NLP landscape. These representations are usually obtained by training a model on large amounts of unlabeled data, and then employed in NLP tasks and downstream applications. While such representations should ideally be evaluated according to their value in these applications, doing so is laborious, and it can be hard to rigorously isolate the effects of different representations for comparison. There is therefore a need for evaluation via simple and generalizable proxy tasks. To date, these proxy tasks have been mainly focused on lexical similarity and relatedness, and do not capture the full spectrum of interesting linguistic properties that are useful for downstream applications. This workshop challenges its participants to propose methods and/or design benchmarks for evaluating the next generation of vector space representations, for presentation and detailed discussion at the event.
>
> *Call for Papers RepEval 2017*, `https://repeval2017.github.io/call/`

This call for the RepEval workshop proves that the evaluation of DSMs is a real and complex problem that impacts all types of DSMs. The current evaluation methods are not satisfying because they mainly focus on lexical similarity and while this notion is major in distributional semantics, other linguistic properties are omitted.

The difficulty of finding an appropriate evaluation task also lies in the fact that DSMs are used in many different ways and for different applications. Some studies focus on understanding the type of information encoded in vectors while other studies are rather more interested in using DSMs for a particular application. In both cases different aspects are investigated. As a consequence it is necessary to adapt the evaluation so that it is relevant to the purpose evaluated.

In deep learning, word embeddings are used as the first layer in neural network models to represent input words. As such, it is usual to use the best performing models. Deep learning models are expensive to train and it is thus common to test the performance of embeddings separately without necessarily evaluating the impact on the whole deep learning system. As a consequence, getting the

best performing models generally means tuning the various different parameters to find the combination that will train the best model. Sometimes different combinations are tested without even considering the various impacts of the different parameters. Other studies focusing on the best combinations of parameters take another approach by investigating the impact of those parameters (see for example Caselles-Dupré et al. (2018); Chiu et al. (2016); Melamud et al. (2016)). As we saw in the previous chapter, we can in fact state that DSMs have really become part of two worlds. The deep learning world focuses more on using DSMs without necessarily questioning the type of information they contain, while the linguistic world uses DSMs in a more empirical way. The division between the two approaches is actually identified as a problem.

DSMs provide great tools that can be used in linguistics. E.g. it is possible to retrieve words that are semantically similar in a corpus by observing a word's nearest neighbors. In that case, the focus is rather set on the ability of DSMs to properly reflect linguistic information and thus being able to use DSMs to observe linguistic phenomena. However, before being able to use DSMs to draw linguistic conclusions it is absolutely necessary to make sure that models are accurate and this remains a difficult challenge.

Another empirical use of DSMs consists in investigating the DSMs themselves to understand the type of information they encode. This type of research does not necessarily target the use of DSMs for a specific goal but is more focused on understanding the different linguistics processes encoded in DSMs (syntax, polysemy etc.). E.g., Gupta et al. (2015) investigated how DSMs encode referential attributes (e.g. *Italy* has *60 million* inhabitants) using a supervised regression model. Another example is Andreas and Klein (2014) who investigated the type of syntactic information encoded by word embeddings.

Evaluating DSMs consists in comparing the performances of several models and this comparison can sometimes aim at observing differences. E.g. in disciplines like *digital humanities* (i.e. the use of techniques like NLP for the humanities) DSMs can be used to investigate phenomena such as linguistic change (see for example Hamilton et al. (2016))[1].

This variety of usages and applications requires different ways to evaluate DSMs. It would be irrelevant but also a waste of time and resources to evaluate DSMs that are to be integrated into a deep learning system and DSMs that are used to investigate linguistic phenomena with the same methods. Usually DSMs are evaluated with two different types of methods: **extrinsic** and **intrinsic**.

Extrinsic methods do not directly evaluate the model but the benefits it brings when used as a component in a given task. This means that this type of evaluation focuses on the performance of the models integrated into the system rather than

[1]We will review this study in more details in chapter 3.

on the performance of the DSM itself. E.g. this is the case with part-of-speech tagging, Named Entity Recognition or sentiment classification. In those cases, evaluation is directly related to the task as a whole (texts used to evaluate are labeled with named entities, documents are classified according to the sentiment class they belong to etc.) and the evaluation measures used are specific to the task. This type of evaluation assumes that having good quality DSMs helps improving the performance of the tested model (Schnabel et al., 2015). Its primary focus is not set on understanding better the different linguistic aspects encoded in DSMs. We could expect that if a model helps get better results for a POS tagging task it means it encodes good quality information about syntax. However, Schnabel et al. (2015) specifically focused on the influence of embeddings quality on different tasks and they found that tasks were impacted differently by the quality of models. This means that by evaluating only with extrinsic methods it might be necessary to multiply evaluation tasks to get a global overview of the impact of a model on a task performances.

Intrinsic evaluation is the other type of method used to assess DSMs performances. In the evaluation of NLP systems, intrinsic evaluation consists in evaluating the output of a system according to predefined criteria that correspond to how the system is supposed to function. Compared to extrinsic evaluation, intrinsic evaluation is generally easy to setup automatically, since only the output of the system is assessed. The first part of intrinsic evaluation usually consists in getting data annotated by human judges. The annotated data is then used to compare against the output of the evaluated system (Resnik and Lin, 2010). Let's consider the evaluation of a POS-tagger as an example. Usually POS-taggers are used to process data and as such they are part of an intermediate phase. Then the extrinsic evaluation consists in evaluating the whole pipeline for a given task (e.g. pre-processing data by lemmatizing and POS-tagging words, and then using the tagged data as the input of a sentiment classification task). In order to improve the POS-tagger and be able to identify its limitations, it is necessary to use intrinsic evaluation methods. First, experts are asked to annotate sentences with the POS corresponding to each token. Then the same sentences are used as an input to the POS-tagger and the output given by the system is compared to the annotations made by the judges. The annotations made by judges are considered the gold standard. A well performing POS-tagging model is then able to get results as close as possible to the human annotations.

Contrary to extrinsic evaluation, intrinsic evaluation focuses more on the DSMs themselves by assessing the quality of word vectors through the comparison of their performance against datasets (or gold standards) that were created gathering human similarity judgments. If we considered intrinsic evaluation tasks literally, experts would be asked to create vectors from scratch. Then these vectors would

be compared to trained DSMs. However, this approach would be very naive and actually impossible to set up. There is a need for an intermediate task that remains close to the vectors. This is why the measure of similarity is used to evaluate DSMs. Because it is easy to use these datasets, they are usually the standard method to evaluate DSMs.

The work we present here focuses on the investigation of word embeddings from a linguistic point of view and we want to understand the role played by linguistics when training word embeddings. We are thus interested in investigating the model by itself. This is why we chose to focus on intrinsic evaluation. In the next section we present selected intrinsic evaluation datasets used to evaluate DSMs. We show that intrinsic methods can provide immediate feedback on the performance of DSMs. We also show that this type of evaluation is not satisfying to understand DSMs in a deeper way and that resources used in this type of evaluation present several biases.

2.2 Intrinsic evaluation

When evaluating DSMs, we are interested in the distance between vectors which reflects their similarity. The most direct way to test for this is by verifying if vectors properly encode the notion of similarity as it is conceptualized by humans. Several datasets were created by asking humans to rate the degree of semantic similarity between word pairs (Baroni and Lenci, 2011; Baroni et al., 2014; Schnabel et al., 2015). Pairs with high scores are considered highly similar and pairs with low score are considered highly dissimilar. The performance of the model is then assessed by computing the correlation between scores given by human annotators and a chosen similarity score (such as the cosine score) computed between vectors. This task is thus testing the extent to which a given model reproduces human judgments (Gladkova, 2016). When comparing different DSMs, the model getting similarity score the closest to the ones given by human annotators is considered the "best" model.

Because several datasets already exist, intrinsic evaluation does not require a lot of additional resources. Only a dataset annotated with human judgments as well as a chosen similarity measure are needed to evaluate the performance of a model. It is thus an efficient and easy way to evaluate DSMs with the guarantee of getting immediate and easy to interpret feedback on a model's performance. At the same time, it also allows for faster prototyping and development of DSMs (Faruqui et al., 2016). This explains why it is a very popular method to evaluate DSMs.

With the introduction of word embeddings, analogy has become a popular intrinsic evaluation task. Mikolov et al. (2013a) showed that word embeddings are

good at capturing semantic information that can be retrieved thanks to simple algebraic operations. This is illustrated by the famous following example:

$$vector(king) - vector(man) + vector(woman) = vector(queen)$$

where a model must be able to retrieve that *queen* is for *woman* what *king* is to *man*.

Mikolov et al. (2013a) designed the Semantic-Syntactic Word Relationship dataset to test how embeddings capture analogies. Table 2.1 displays some selected examples from this dataset. We can see then that various types of relations are expected to be captured, e.g. a capital to its country (*Beirut* is to *Lebanon* what *Berlin* is to *Germany*) or an adjective to its superlative (*bad* superlative is *worst* and *high* superlative is *highest*).

Relation	**Word Pair 1**	**Word Pair 2**
Capital - Country	Beirut - Lebanon	Berlin - Germany
City - State	Chicago - Illinois	Houston - Texas
Adjective - Adverb	complete -completely	infrequent - infrequently
Adjective - Superlative	bad - worst	high- highest

Table 2.1: Examples of relations and word pairs from the Semantic-Syntactic Word Relationship dataset (Mikolov et al., 2013a).

In the next sections we present selected datasets commonly used for intrinsic evaluation. We first present what we consider to be "traditional" datasets, in the sense that they are systematically used for DSMs evaluation. We then discuss the various problems encountered with these datasets as well as the alternative datasets developed to overcome these problems.

2.2.1 Traditional datasets

We consider traditional datasets, datasets of word similarity measures commonly used for the evaluation of DSMs.

Rubenstein and Goodenough (1965) This dataset is one of the first ones created to test semantic similarity. Rubenstein and Goodenough (1965) were concerned by the relationship between similarity of context and similarity of meaning and decided to investigate the hypothesis stating that words sharing many contexts are semantically related. The notion of similarity is difficult to evaluate. An inexpensive way to test for it is to ask human annotators to rate the similarity of

pairs of words. As a consequence, the authors gathered 65 pairs of common English nouns (48 different nouns) and asked 51 judges to rate those pairs between 0 and 4. The following instructions were given to the annotators:

> 1. After looking through the whole deck, order the pairs according to amount of "similarity of meaning" so that the slip containing the pair exhibiting the greatest amount, of "similarity of meaning" is at the top of the deck and the pair exhibiting the least amount is on bottom,
> 2. Assign a value from 4.0-0.0 to each pair–the greater the "similarity of meaning," the higher the number. You may assign the same value to more than one pair.
> (Rubenstein and Goodenough, 1965)

The annotated pairs were used to investigate the correlation between shared contexts and synonymy. The authors did find a positive relationship between the degree of similarity of words and the degree of similarity of their contexts.

Word1	**Word2**	**Judgment score**
fruit	furnace	0.05
magician	oracle	1.82
asylum	madhouse	3.04
magician	wizard	3.21
gem	jewel	3.94

Table 2.2: Examples of pairs of words along with judgment similarity scores selected from the Rubenstein and Goodenough (1965) dataset.

While this dataset was first designed to investigate a specific phenomenon, it is still used nowadays to evaluate the performances of DSMs. Table 2.2 displays selected examples of pairs of words from this dataset along with the average judgment score assigned to the pair. While it is not surprising that *fruit* and *furnace* have a very low similarity score, we can notice that *asylum* and *madhouse* have a lower similarity score than *magician* and *wizard* or than *gem* and *jewel*. We can wonder why this is the case and if it is really accurate. What makes *wizard* more similar to *magician* compared to *asylum* and *madhouse*? Since the judgments are given by human annotators without any explanations to justify their choices, we can question the scores accuracy. This type of phenomenon is a recurring problem with datasets used for intrinsic evaluation as we will observe later on.

WordSim-353 (Finkelstein et al., 2002) Later on, with the development of Natural Language Processing and Information Retrieval, there was an increased need to be able to test semantic similarity. WordSim-353 is a highly popular dataset that was created to test a system developed by Finkelstein et al. (2002). At the time there were "no accepted procedures for evaluating performance of semantic metrics" and the main idea was that "a good metric should approximate human judgments well" (Finkelstein et al., 2002). Some resources already existed such as the Rubenstein and Goodenough (1965) dataset, however they were a bit limited and consisted of only a few pairs of words. As a consequence, Finkelstein et al. (2002) decided to develop a test set containing 353 pairs of nouns (437 distinct nouns) with different degrees of similarity. Although we know that the task was carried out by several judges (16 subjects were asked to give a similarity score ranging from 0 to 10 to each pair of words), no details are explicitly given regarding the way the resource was designed and the choice of words constituting the 353 pairs.

Word1	**Word2**	**Judgment score**
king	cabbage	0.23
glass	metal	5.56
money	laundering	5.65
cup	coffee	6.58
tiger	cat	7.35
book	paper	7.46
computer	keyboard	7.62
king	queen	8.58
Maradona	football	8.62
magician	wizard	9.02

Table 2.3: Selected examples of pairs of words along with judgment similarity scores from WordSim-353.

Table 2.3 displays selected pairs of words along with their judgment scores. By looking at pairs with the highest scores, we notice that the semantic relations shared by words are heterogeneous. For example, *Maradona* and *football* are linked by a semantic relation of association, Maradona was a football player. *Magician* and *wizard* are synonyms. *Keyboard* is a meronym of *computer*. It is thus challenging to understand the meaning of a high similarity judgement. Moreover, pairs of co-hyponyms are not always evaluated in the same way. The pair *glass* and *metal* have a lower similarity score than *king* and *queen* while they are both co-hyponyms. Despite these anomalies, because there was a lack of resources to evaluate semantic

similarity systems and because WordSim-353 was one of the first large semantic similarity dataset, it rapidly became a standard for DSMs evaluation. It has been and is still used in many studies. E.g., Lapesa and Evert (2014) used it as one of the datasets to evaluate the impact of different parameters (corpus, window direction, transformation applied to the model etc.) when training DSMs. Similarly, Kiela and Clark (2014) used it to investigate several parameters used when training DSMs. More recently, Li et al. (2017) used it as part of the evaluation of several word embeddings models learned using different context types.

MEN (Bruni et al., 2013) While WordSim-353 contains more words than the resource developed by Rubinstein et al. (2015), its coverage is still rather limited especially considering that DSMs consist of thousands of words. MEN (Marco, Elia and Nam) is a larger resource made of 3000 pairs of words (751 distinct words) and was created specifically to evaluate multimodal distributional semantics models, i.e. models that mix text and image features. Because of its large coverage, MEN can be divided into development and test set to avoid overfitting on the dataset. The dataset was created using words randomly collected from text corpora (words had to appear at least 700 times in UKWaC and Wackypedia) combined with image collections (words had to appear as tags at least 50 times in ESP-Game and MIRFLICKR-1M). Each collected pair was randomly matched with another pair and a single Amazon Mechanical Turk worker was asked to rate if a given pair was more or less semantically related compared to the other pair provided. Each pair was rated against 50 pairs thus giving a 50-scale points final score[2]. This scoring method was used to avoid giving specific scores to pairs and to rather be able to compare judgments without bias due to similarity perception.

Word1	**Word2**	**Judgment score**
automobile	car	50
raspberry	sweet	34
bedroom	kitchen	30
beauty	orchid	23
fun	haircut	15
art	beer	9
bakery	zebra	0

Table 2.4: Selected pairs of words from MEN along with similarity score.

Table 2.4 gives some examples of word pairs in the MEN test set as well as

[2] `https://staff.fnwi.uva.nl/e.bruni/MEN`

the comparison score out of 50. Again, we notice the variety of semantic relations among pairs which is even made more challenging because of the variety of POS used in the resource. E.g. *sweet* is an adjective and is an attribute of *raspberry*. Its score is higher than the pair of co-hyponyms nouns *bedroom* and *kitchen* (34 and 30 respectively).

SimLex-999 (Hill et al., 2015) Recently, Hill et al. (2015) developed SimLex-999, a test set made of 999 pairs of words (1028 distinct words) that focuses mainly on similarity. This means that contrary to other datasets such as WordSim-353 that mixes the notion of similarity and of semantic relatedness[3][4] SimLex contains only pairs of words that are semantically similar. The pairs were selected using the University of South Florida Free Association Database. This database was constituted by presenting words to participants and asking them to write the first word that came to their mind and that was associated or "meaningfully related" to that word. Each concept was presented to at least 10 participants and the database contains more than 72000 pairs of words. To select the 999 pairs of words, some filtering was applied on the 72000 pairs of words. First, only pairs that were semantically similar were kept. Then pairs where a word POS was ambiguous were discarded. The pairs are made of words that have the same POS, resulting in pairs of adjectives (e.g. *wide-narrow*), pairs of nouns (e.g. *woman-man*) and pairs of verbs (e.g. *vanish-disappear*). The ratings of pairs was done by participants via Amazon Mechanical Turk (500 participants). Participants were asked to assign a rate ranging from 0 to 6 to 20 groups of pairs. Adjustments were made to ensure that the ratings were not biased. The mean rating of each participant was computed and when the absolute difference between the mean rating and the mean of the ratings of all participants was greater than 1, the ratings were increased or decreased to make sure the ratings were consistent. The average mean ratings for each pair were then transposed from the 0 to 6 scale to a 0 to 10 scale. Because Hill et al. (2015) wanted to represent all degrees of concreteness in the test set, pairs of words have different degrees of concreteness. Table 2.5 displays some pairs of words with the concreteness score[5] of each word as well as the similarity score assigned to the pair. Pairs such as *book* and *text* or *wood* and *paper* are constituted of highly concrete words while pairs such as *bizarre* and *strange* or *belief* and *impressions* are constituted of highly abstract words. We will investigate the role of the degree of concreteness on DSMs in chapter 6.

[3]We saw such an example before with the high score of 8.62 for the word pair *Maradona-football*.

[4]The distinction between semantic similarity and relatedness is discussed in section 2.2.2.6.

[5]We further discuss concreteness in section 2.2.2.4.

Word1	**Word2**	**Concr. Word1**	**Concr. Word2**	**Sim. score**
book	text	4.9	4.93	6.35
wood	paper	4.85	4.93	2.88
sad	funny	3.07	2.5	0.95
champion	winner	3.82	3.21	8.73
acquire	find	2.93	2.63	6.38
belief	flower	1.19	5	0.4
bizarre	strange	1.79	1.86	9.37
belief	impression	1.19	2.23	5.95

Table 2.5: Pairs of words selected from SimLex-999 along with similarity score and concreteness score for both words.

We presented four different datasets. We saw that each dataset was designed differently. As a consequence, when using these test sets to evaluate DSMs, it is not possible to compare the different results. Moreover, depending on the size of the test set, only a small portion of the model is evaluated. While numerous other datasets exist, we chose to present here a selection of popular ones. All the presented datasets were created using average human annotations for several pairs of words. We saw that these datasets were of different sizes. All datasets have a similar approach to similarity that is not always clear. Despite this heterogeneity and lack of clarity, there are all used in the same way, with automated measures that are easy to implement.

In this section, we presented selected popular datasets often used as a benchmark in DSMs evaluation. We also showed that effort is made to improve datasets over time with most recent ones trying to overcome known problems. However several problems remain unresolved, e.g. no information is given about the type of semantic relations governing the different pairs of words and no systematic difference made between semantic similarity and relatedness. Another problem is that is it common to use several datasets to evaluate one model. The idea then is to achieve the highest scores possible on the selected datasets. The better score represents the best final evaluation result. Does the variety of resources implies better results? Is a model considered more reliable because it performs well on several datasets? The next section reviews and questions the main issues existing with most of the datasets we presented.

2.2.2 Common problems with classic datasets

Recently, the use of word embeddings have raised questions regarding the use of intrinsic datasets as a main tool of evaluation. Faruqui et al. (2016) show that the test sets used suffer several problems. Schnabel et al. (2015) argue that pairwise similarity alone miss information encoded in word vectors. In this section, we explore several problems existing with traditional datasets.

2.2.2.1 Size of datasets

The first datasets created to evaluate semantic similarity were relatively small. While more recent datasets consisted of more pairs of words, datasets used for evaluation are still very limited. Moreover, words used in those pairs are re-used from one pair to the other. For example we saw that MEN is constituted of 3000 pairs of words made using 751 different words. This means that these datasets only cover a limited portion of the model's vocabulary, that might not always be relevant for all models. E.g., WordSim-353 was developed in a particular context, to evaluate the semantic core of an information retrieval system, IntelliZap. While this system was performing general knowledge search, we can wonder if it is really transferable to other systems and models without questioning the content of the dataset.

2.2.2.2 Subjectivity of the annotation task

Another major problem with datasets used for intrinsic evaluation is the subjectivity implied by the annotation process for the pairs of words. The similarity scores in datasets are human judgments, that are crowd-sourced most of the time. This means that there is no control of the environment in which the annotation task was conducted (Gladkova, 2016). The task is also not necessarily completed by experts and as a consequence these scores seem to correspond more to an intuition than to an exact way of measuring semantic similarity between words (Baroni and Lenci, 2011). Evaluating the similarity of pairs of words is not trivial. We can give as an example our own experience as annotator as part of the Evolex project, a multidisciplinary project combining psychological, neuropsychological and natural language processing methods. This project aims at proposing tools to be able to evaluate how lexical access differs for people with or without language deficits (Gaume et al., 2018). This project uses pairs of words that were gathered by asking participants to name the word that would come to their mind given a word. To perform analyses these pairs needed to be annotated with the semantic relation they shared. While both persons in charge of the annotation task were experienced linguists, they did not always agree on the type of semantic relation shared by words. E.g. for the pair *apricot* and *tree* one annotator considered

that *tree* was an holonym of *apricot* while the other annotator judged it was more relevant to say that the two words were associated.

The problem is also, as noted by Faruqui et al. (2016), that the annotations are not consistent between pairs of words. E.g. a pair such as *cup* and *coffee* was considered more similar by annotators in WordSim-353 than a pair like *car* and *train*. This could even be reinforced by the absence of context during the annotation task that makes it challenging to distinguish between meanings that are subtle most of the time[6]. Moreover, with no context available, it is challenging to annotate polysemous words. As a consequence, the inter-annotator agreement is low for a task that assumes that there exists a single similarity score for each pair of words (Batchkarov et al., 2016). Looking at a specific example, the word pair *tiger-cat* have scores ranging from 5 to 9 in WordSim-353.

2.2.2.3 Identification of words

We saw that datasets sizes remain limited and that the provided scores cannot be fully relied on. Another major problem is that the same amount of information is not provided across datasets making it difficult to compare results from one dataset to another. For example, some datasets do not provide any information about the POS of words. This is a problem for several reasons. First, it questions the viability of the annotation task, since some words could be ambiguous and be either nouns or verbs. Secondly, it also means that when using those datasets one does not exactly know what is evaluated. While some datasets have tried to overcome this problem, such as SimLex-999 which provides the POS of words, it questions the use of multiple datasets and the ability to understand and interpret results from one dataset to the other.

2.2.2.4 Concreteness of words

We saw that SimLex-999 incorporated information about the concreteness of pairs of words. We were curious about the different concreteness degrees of words in different datasets and decided to investigate it. We chose to focus on WordSim-353, MEN and SimLex-999.

To investigate the average concreteness of words in evaluation test sets we used crowdsourced concreteness ratings for 39954 words gathered by Brysbaert et al. (2014). Each word has a concreteness score ranging from 1 (for abstract words) to 5 (for concrete words). The average concreteness score over the whole test set is of 3.04 with a standard deviation of 1.04. The following instructions were given to participants:

[6]See also Muller et al. (2014).

> Some words refer to things or actions in reality, which you can *experience directly through one of the five senses*. We call these words concrete words. Other words refer to meanings that cannot be experienced directly but which we know because the meanings can be defined by other words. These are abstract words. Still other words fall in-between the two extremes, because we can experience them to some extent and in addition we rely on language to understand them. We want you to indicate how concrete the meaning of each word is for you by using a 5-point rating scale going from abstract to concrete. A concrete word comes with a higher rating and refers to something that exists in reality; you can have immediate experience of it through your senses (smelling, tasting, touching, hearing, seeing) and the actions you do. The easiest way to explain a word is by pointing to it or by demonstrating it (e.g. To explain 'sweet' you could have someone eat sugar; To explain 'jump' you could simply jump up and down or show people a movie clip about someone jumping up and down; To explain 'couch', you could point to a couch or show a picture of a couch). An abstract word comes with a lower rating and refers to something you cannot experience directly through your senses or actions. Its meaning depends on language. The easiest way to explain it is by using other words (e.g. There is no simple way to demonstrate 'justice'; but we can explain the meaning of the word by using other words that capture parts of its meaning).
> Brysbaert et al. (2014)

According to Brysbaert et al. (2014), concrete words correspond to things you can experience through your different senses while abstract words are things you cannot experience directly and that are more difficult to demonstrate. Words were presented to annotators without their POS which was added a posteriori according to the dominant usage in corpus. Table 2.6 gives examples of abstract words (*although* and *belief*), somewhat concrete words (*synaptic* and *tonight*) and very concrete words (*human* and *elk*). We notice that the POS attributed to words do not correspond to the reality of the annotation. The high concreteness score of *human* is probably a result of the fact that when it was presented to annotators they thought about the noun. Moreover, *elk* is annotated as a verb while it actually is a noun and it does not exist in any dictionary as a verb. Despite these aspects, we decided to experiment with this resource because it is, to our knowledge, the most comprehensive resource on concreteness.

Word	Concr. score	Dominant POS
although	1.07	Conjunction
belief	1.19	Noun
synaptic	2.54	Adjective
tonight	2.93	Adverb
human	4.53	Adjective
elk	4.93	Verb

Table 2.6: Selected examples of words with their concreteness ratings and POS from Brysbaert et al. (2014) resource.

For each word in each test set we retrieved its concreteness score and discarded words that were not found in Brysbaert's database. We then computed the average degree of concreteness for each test set. This resulted in concreteness scores for 1028 words for SimLex-999, 743 words for MEN and 418 words for WordSim-353. For the latter, words that did not exist in the resource were mainly proper nouns.

We found that both SimLex-999 and WordSim-353 contains a mix of abstract and concrete words with an average concreteness score of respectively 3.63 and 3.73 and a standard deviation of respectively 1.13 and 1. Regarding SimLex-999 we were expecting words with different degrees of concreteness since it was designed this way. MEN contains mainly highly concrete words with an average concreteness score of 4.42 and a standard deviation of 0.61. This bias towards concrete words is most certainly due to the way the resource was constituted, selecting words that appeared as tags in image collections.

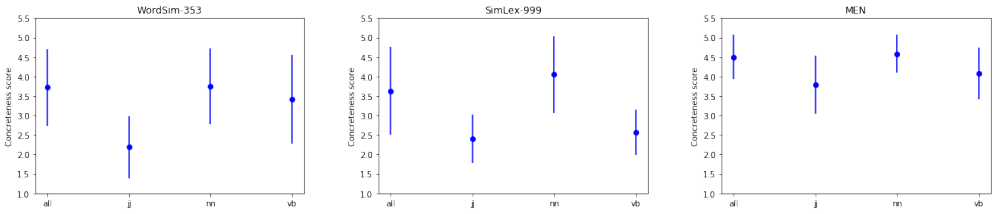


Figure 2.1: Average concreteness score with standard deviation by POS and for all POS together for words in WordSim-353, SimLex-999 and MEN.

Figure 2.1 displays the concreteness score grouped by POS as well as the overall concreteness score for all POS for WordSim-353, SimLex-999 and MEN. We can see that although SimLex-999 and WordSim-353 are on average a mix of abstract and concrete words, the repartition among POS is not equally distributed. In WordSim-353 adjectives are mainly abstract while it seems that nouns are mostly concrete. A similar pattern can be observed in SimLex-999 with adjectives and

verbs being mostly abstract and nouns being mostly concrete. We also found that some POS are more present than others in SimLex-999 and that the majority of the test set consists of nouns (1332 nouns, 444 verbs and 222 adjectives). MEN mostly consists of concrete words, regardless of their POS.

2.2.2.5 Polysemy

Because of the nature of the annotation task, we also wanted to investigate the polysemy of words in different test sets. Again, we chose to focus on WordSim-353, SimLex-999 and MEN.

We computed the degree of polysemy of each word using Princeton Wordnet (Miller, 1995). For each word in each evaluation test set we retrieved the number of synsets it has. A word with a high number of synsets is considered to be highly polysemous.

In order to get an idea of the average degree of polysemy of nouns, verbs and adjectives, we computed the average number of synsets per POS in Wordnet. We found that nouns have 1 synset on average (± 1), verbs have 3 synsets on average (± 3) and adjectives have 2 synsets on average (± 1).

When computing the average number of synsets per POS for words in the selected benchmarks we found that in SimLex-999 and MEN verbs were highly polysemous with an average number of synsets of respectively 7 (± 8) and 12 (± 10). Among those highly polysemous verbs we observe generic verbs such as *make*, *carry*, *hold* or *draw*. Verbs in WordSim-353 have 4 synsets on average (± 1) which is similar to what was observed for all verbs in WordNet. Adjectives also have more synsets on average than all the adjectives in WordNet. However we observed that adjectives have about the same number of synsets in the 3 benchmarks: 5 in WordSim-353 (± 4), 5 in SimLex-999 (± 4) and 5 in MEN (± 5). We found that nouns in all the three benchmarks have 4 synsets on average (± 3). It is interesting to notice that unlike what could have been expected given their high degree of concreteness, words in MEN are as polysemous as the one in SimLex-999 and WordSim-353.

We found that on average words in benchmarks are more polysemous than the average English nouns, verbs and adjectives. We also found that verbs and adjectives are more polysemous than nouns. Since there is only one representation per word, polysemy is directly related to the distributional semantics problem. As a consequence, it is important to consider the bias towards polysemous words in the datasets when using them.

2.2.2.6 Semantic similarity and relatedness

The last issue we want to point at concerns how similarity is represented in datasets. It is generally not clear what type of semantic relations are being assessed. What does it mean that two words are considered similar? Do they need to be synonyms? Are hyponyms considered similar and if so to what degree? Are words bound by different semantic relations considered to hold different types of similarity?

We usually distinguish between two different concepts: **semantic similarity** and **relatedness**. For Budanitsky and Hirst (2006), relatedness is a more general concept than semantic similarity and it refers to words that are related by the semantic relation of meronymy (*body-arm*), antonymy (*tall-small*), by functional association (*cup-coffee*) or by "non-classical relations". Non-classical relations are a reference to Lakoff (1987)'s terminology of "classical" relations that described "categories whose members are related by shared properties" (Morris and Hirst, 2004). Non-classical relations then do not depend on shared properties. Such relations are often referred to using a "general case relation", e.g. instrument/activity for the pair of word *ball-cricket* (Morris and Hirst, 2004). Concerning semantic similarity, Budanitsky and Hirst (2006) consider that pairs of words that are semantically similar are pairs linked by a relation of synonymy (*magician-wizard*) or hypernymy (*animal-cat*).

In their study on similarity and relatedness, Agirre et al. (2009) consider that words that are similar are words that share a semantic relation of synonymy, antonymy, hyponymy and hypernymy. According to them, meronyms and holonyms are considered related just like words that are linked by a relation of association. We can point out some differences with Budanitsky and Hirst (2006)'s theory who considered that antonyms are semantically related and not semantically similar. As a consequence, we already get an idea of how complex and challenging it can be to determine if two words are semantically similar or semantically related.

Most of the datasets used for intrinsic evaluation mix both pairs of words that are semantically related and semantically similar. As pointed by Agirre et al. (2009), when rating word pairs in WordSim-353, annotators were not clearly instructed to differentiate between semantic similarity and relatedness and as a consequence it contains a variety of semantic relations that are not equally distributed between the different types of relations (Baroni and Lenci, 2011). Sometimes, semantically related pairs are also rated differently and it is not clear how to explain this rating difference. E.g. in table 2.3 we saw that *cup* and *coffee* have a similarity score of 6.58 while *Maradona* and *football* have a similarity score of 8.62. Does it mean than the second pair is more semantically related than the first one? What is a good way to explain this difference apart from intuition? Moreover, some pairs

which are semantically similar such as *car-automobile* have a rating very similar to pairs that are semantically related (8.94 compared to 8.62 for *Maradona-football*).

To avoid the confusion between semantically related and semantically similar, Agirre et al. (2009) decided to create two different dataset from WordSim-353, one focusing on similarity and one focusing on relatedness[7]. Similarly, datasets like SimLex-999 focus only on words that are similar to avoid ambiguity. While it is interesting to distinguish between semantically similar and semantically related, we do not really know which type of similarity should be captured by the model (Faruqui et al., 2016) and as a consequence avoiding one type of similarity or another might not be the appropriate solution.

As a response to the various problems we presented here, effort has been made to propose datasets specifically designed to evaluate DSMs. In the next section we present BLESS and Evalution 1.0, two alternative datasets addressing common problems encountered with traditional datasets, that attempt to answer specific questions about the way similarity is encoded in DSMs.

2.2.3 Alternative datasets

BLESS (Baroni and Lenci Evaluation of Semantic Spaces) (Baroni and Lenci, 2011) Because most datasets existing to evaluate DSMs were not designed specifically for distributional semantics and were also suffering caveats, Baroni and Lenci (2011) decided to create a dataset specifically designed to evaluate DSMs. The main goal was to provide a benchmark that would overcome problems and biases existing with current datasets as well as give information about the extent to which words that are close in the semantic space are actually semantically related. This means that instead of getting only a similarity score for a pair of words, the dataset provides information about the type of semantic relations governing words.

BLESS is made of 200 words that are distinct English concrete nouns that are equally divided between living entities (e.g. *frog*, *turtle*, *tuna*) and non-living entities (e.g. *freezer*, *van*, *onion*). Words are divided into 17 classes that are identified in table 2.7. The selection of words was made to ensure a good coverage of semantic relations and making sure at the same time that they presented no ambiguity. This is why concrete nouns were chosen since they are the most studied and they also present more agreement about the semantic relation that characterize them. Words were mainly selected using the McRae Norms (McRae et al., 2005). McRae Norms are semantic feature norms that were collected from about 725 participants for 541 living and non-living basic concepts. Participants were asked

[7]See Agirre et al. (2009); Baroni and Lenci (2011) for a detailed annotation of WordSim-353 test set.

Semantic class	Examples
AMPHIBIAN REPTILE	alligator, frog
APPLIANCE	dishwasher, fridge
BIRD	dove, falcon
BUILDING	castle, cottage
CLOTHING	blouse, coat
CONTAINER	bottle, mug
FRUIT	banana, lemon
FURNITURE	chair, sofa
GROUND MAMMAL	beaver, cat
INSECT	butterfly, moth
MUSICAL INSTRUMENT	clarinet, guitar
TOOL	fork, saw
TREE	cedar, oak
VEGETABLE	cucumber, radish
VEHICLE	ambulance, bus
WATER ANIMAL	goldfish, salmon
WEAPON	dagger, revolver

Table 2.7: Semantic classes in the BLESS dataset.

to give different types of features to each concept such as physical properties, functional properties or the category the concept belongs to. For example *alligator* was assigned features such as *a_reptile*, *an_animal*, *is_dangerous* etc. *Banana* was assigned the features *is_yellow*, *a_fruit*, *grows_on_trees* etc. Out of the 200 words used in BLESS, 175 were taken from the McRae norms and the 25 remaining were chosen by the authors following the same criteria.

Words were also chosen to be reasonably high-frequency. To check frequency, the authors used ukWaC and Wackypedia[8]. All words are single nouns in their singular form and ambiguous and polysemous words were avoided.

To constitute pairs, each word was associated with several relata with which it shares one of the following semantic relation: co-hyponymy (*turtle-frog*), hypernymy (*turtle-amphibian*), meronymy (*turtle-head*), adjective that is an attribute of the concept (*turtle-aquatic*) or verb referring to an action that the concept is involved in (*turtle-eat*). Additionally a random adjective (*turtle-complete*), verb (*turtle-offer*) and noun (*turtle-salary*) were selected to ensure the sanity of the model. All relata were selected using a variety of semantic resources (McRae Norms, WordNet and ConceptNet) as well as corpora (Wikipedia and ukWaC).

[8] `http://wacky.sslmit.unibo.it/`

BLESS is a comprehensive dataset that contains various information about the pairs of words and effort was made to represent equally different semantic relations that are clearly identified. As a consequence, this test set allows the user to be in control of the semantic properties tested in DSMs.

EVALution1.0 (Santus et al., 2015) Following Baroni and Lenci (2011) work, Santus et al. (2015) created EVALution 1.0, a dataset designed to support the training and evaluation of DSMs. EVALution contains 7429 pairs of words related by different types of semantic relations such as hypernymy (e.g. *sport* and *basketball*), synonymy (e.g. *college* and *university*), antonymy (e.g. *abstract* and *concrete*) and meronymy (e.g. *foot* and *body*). It also contains additional information for each pair of words such as the frequency of each word,the POS or the semantic field it belongs to (e.g. nature, object etc.). Pairs were selected using ConceptNet 5.0 (Liu and Singh, 2004) and WordNet (Fellbaum, 1998). Additionally, judgments were collected through CrowdFlower. Corpus data was used to collect information about the frequency of a word and its POS. Each word in word pairs has to occur in more than one semantic relation. Moreover, all pairs are different from the ones in BLESS. To collect information about relation between words, sentences such as "Dog is *a kind of* animal" constructed using the pairs of words were presented to subjects that were asked to rate the truth of the sentence on a scale of 1 (strongly disagree) to 5 (strongly agree). For each sentence, 5 judgments were collected and only pairs getting at least 3 positive judgments were kept. Subjects were also asked to provide information about the abstractness of a term or its generality.

The two datasets we just described present several advantages. They were designed and organized in a way that gives more freedom to evaluate DSMs. Since semantic relations are provided, it is possible to focus only on a certain type of semantic relation when evaluating a model. It is also possible to focus only on a certain type of POS[9]. However, these datasets requires more effort to interpret results compared to the previous ones such as WordSim-353. This might explain why despite efforts put into their creation they are less used than traditional intrinsic evaluation datasets.

2.3 Discussing intrinsic datasets

We presented several datasets commonly used for the evaluation of DSMs. Because most of these datasets were not designed specifically for distributional semantics, the way we use them to perform intrinsic evaluation tasks does not provide deep

[9]Only for EVALution since BLESS concepts are only nouns.

information about evaluated DSMs. They do not provide any information on the type of semantic relations captured by DSMs nor do they allow to observe if results are different for one part-of-speech or another. This type of evaluation only scratches at the surface some of the essential questions we might have about DSMs. Even using datasets specifically designed to evaluate DSMs such as SimLex-999 does not provide enough satisfying insight.

Despite efforts made to propose alternatives, such as BLESS or EVALution 1.0, to improve upon existing datasets, their use is still limited and it is still much more common to use classic datasets. We can think of several reasons that would explain this phenomenon. First, classic datasets have been used for a long time to evaluate DSMs. As a consequence, they really constitute a benchmark in the evaluation of DSMs and it is common to use a specific dataset when evaluating to be able to compare results to similar work that has been done before. Moreover, alternative datasets often require to run more in depth analysis and were not necessarily designed to make prototyping faster. If one wants to train the best performing model to integrate it as a component of an NLP system, it is easier to run an evaluation task that output a single score and to check if this score is improving rather than conduct analyses on the type of semantic relations captured by models.

Truth is, one might not even know what type of specific information they try to capture and represent. Does it matter when trying to represent generic knowledge that the models captures better co-hyponyms than synonyms or meronyms? This leads to an actual question about the type of information encoded in DSMs. What do we actually expect them to be about? When we know that common sense is also a big part of semantic interpretation in human interactions, how do we expect models to encode this type of information. Although this question is not the focus of this thesis, it helps us understand why it is so difficult to find an appropriate way to evaluate DSMs. With the recent arrival of word embeddings and their massive use in the deep learning world, it is also not surprising to see that people fall back on intrinsic evaluation tasks driven by the goal to get the highest score possible. Deep learning models are complex and take time to train. Testing DSMs as part of the entire system in downstream tasks could prove very time consuming and it is common-sense to run intrinsic evaluations to get the best performing models. Sometimes, models used in these systems are not even customed. This is even more true with contextual word embeddings. Rather than re-train models it is more common to fine tune pre-trained models because these models are considered the best performing ones.

Our work focuses specifically on the evaluation of word embeddings. Our first concern was to understand what changes when training word embeddings with different configurations. As a consequence, we were curious about the type of

information missing to be able to evaluate word embeddings in a way that allows us to understand the real impact of parameters. In the next chapter we present several works that adopt different methodologies for the evaluation of DSMs. We will see how the evaluation of DSMs parameters has been investigated until now and we will then present the approach we propose to adopt through the rest of this work.

Chapter 3

Towards Qualitative Evaluation

Contents

3.1 Introduction

In the previous chapter we presented traditional quantitative methods commonly used to evaluate DSMs. These methods are convenient, easy to run and fast to interpret which makes them one of the preferred methods when evaluating DSMs. However, by only computing the correlation with human judgements on a limited set of items, these methods fail to provide interesting material that allows to understand what changes across different models. While efforts are made to provide

more comprehensive and suitable datasets, we still lack a method that supplies both a global and detailed view of what is changing between models. Quantitative methods do allow to rapidly detect anomalies. They also help detecting differences between two models. However they do not really give any insight about the different problems encountered in models and the difference of accuracy is not necessarily questioned as an indicator of a phenomenon to be examined. When DSMs constitute a component in an NLP system, it is logical that the model integrated has to perform optimally. The easiest way to measure this performance and fine tune models if necessary, is by using intrinsic datasets. However, when models are used to investigate linguistic phenomena, any fluctuation in the model can disrupt results. As a consequence, it is essential to measure what changes from one model to another. This means that the evaluation method should focus on understanding the various impacts of different training configurations.

In social sciences, using only quantitative or qualitative methods is not satisfying because favoring one approach might end up omitting important information. The quantitative methods measure the performance of the model, which is a way to insure the sanity of the model, while qualitative methods provide more holistic analyses based on observations. As a consequence the combination of quantitative and qualitative methods is interesting (Galliers and Sparck Jones, 1993).

Adopting a qualitative evaluation methodology can be expensive in terms of time and setup. Moreover, it takes time to investigate the data and interpret results. However, it also presents several significant advantages. First, it is possible to control the extent of the investigation by being able to choose exactly what is analyzed (focus on a single phenomenon, target specific words etc.). Secondly, qualitative evaluation allows to stay close to the explored data. It makes it possible to adopt an approach targeting a specific goal or to rather conduct experiments in an exploratory way.

In this chapter we present several studies that implemented custom evaluation methods in order to explore specific phenomena. These methods can be either quantitative or qualitative or include aspects of both. We chose to present the different works grouped by research interest.

3.2 Alternative evaluation setups

We saw in chapter 1 that DSMs are used in many different ways and for various applications, from their integration in deep learning systems where they act as a representation of semantic knowledge to their use as a tool to investigate linguistic phenomena such as textual entailment or the type of relations captured by DSMs. DSMs are also used to explore phenomena such as diachronic linguistic change. More recently, research studies have also focused on different aspects of DSMs

such as the sociological biases they suffer from (Bolukbasi et al., 2016). With these varieties of applications, different evaluation processes are required.

We present here selected studies that integrate a qualitative point of view to evaluate word embeddings for a specific goal. The first set of studies investigate parameters used when training DSMs. The second set focuses on the use of word embeddings to explore linguistic change. Finally, we present a last group of studies using DSMs in the context of specialized corpora.

3.2.1 Evaluating DSMs parameters

Building DSMs requires to tune a certain number of parameters whether the goal is getting a model with optimal performances or finding appropriate parameters in the context of a specific task. Because of the number of parameters to consider, choosing the right configuration proves to be challenging. Moreover, understanding those parameters is not always straightforward. We saw that in the case of count-based models parameters are most of the time directly related to the data used to build semantic representations. These parameters usually consist of the window size or the direction of the window. Word embeddings are slightly different since they involve parameters affecting the data used to learn (window used, subsampling rate, negative sampling rate) and parameters impacting the learning algorithm itself (architecture). Moreover, when adjusting parameters, changes might not directly be visible without investigating the implementation in details. This is especially true for the tool we focus on in this thesis, word2vec. For example, it is not possible to directly visualize the effects of subsampling without modifying word2vec code. As a consequence, tuning word2vec parameters is an opaque process making it difficult to understand the various effects of parameters. As a consequence using only intrinsic evaluation datasets is not satisfying to fully understand the impact of the various parameters. Using alternative evaluation methods, that can be based on both quantitative and qualitative aspects, might help shedding light on the relations between a model's performance and the organization of its lexical space by providing insights about the behavior of specific words or classes of words in regards to parameters changes. We present two different works evaluating the impact of parameters when training DSMs. The first one focuses on the interaction of different parameters and the impact of these interactions on performances. The second one evaluates DSMs trained for specialized lexicography.

3.2.1.1 Evaluating parameters interaction

Building DSMs requires to tune numerous parameters. One big challenge is that the combination of all the available parameters can quickly expand and increase

the number of models to be built and evaluated. Lapesa and Evert (2014) conducted a major investigation of count-based models parameters and their interaction. They chose to examine various parameters: corpus, window size, window direction, context filtering, distance metric, dimension reduction algorithm etc. This work aimed at shedding light on unanswered questions about the different parameters and their influence on models. They trained 537600 models total that were evaluated on traditional datasets simlar to the ones we presented in chapter 2. They also performed a noun clustering task which consisted in assigning nouns to pre-defined semantic classes. The performances of models are evaluated in terms of cluster purity.

Because of the large number of models evaluated, using qualitative evaluation methods seems ambitious and this explains the use of tasks that are all very (or exactly) similar to the intrinsic evaluation methods presented in chapter 2. The clustering task is very similar to a word similarity dataset task. In fact, the clusters are only evaluated quantitatively without further analysis of the words that were assigned to the wrong clusters. However, looking at errors could provide interesting insights about the semantic properties of models. The work conducted by Lapesa and Evert (2014) allowed to detect general tendencies when building DSMs. E.g. the transformation applied to vectors and the distance metric used play an important role on the performances of DSMs. Moreover, they showed that there is not one configuration better than other when building DSMs.

3.2.1.2 Evaluation of parameters applied to specialized lexicography

If finding optimal parameters when training DSMs is challenging for general semantic knowledge, it can be even more challenging when training DSMs for specific tasks and domains. Bernier-Colborne and Drouin (2016) conducted a "holistic" study to find optimal parameters for DSMs in the context of specialized lexicography. It is interesting to notice here the use of the word *holistic* in the name of their paper, since it reminds us of what Galliers and Sparck Jones (1993) who pointed out that qualitative evaluations are more holistic. As a consequence, from the title we expect an analysis more based on qualitative observations.

The interaction of model-related and application-related factors was investigated. Model-related factors refer to parameters used when building models: direction of the context window, size and shape of the context window, weighting scheme. In the case of specialized lexicography, application-related factors refer to the kind of terms included in the lexical resource that is built as well as the kind of relations described (paradigmatic relations such as synonymy and syntactic relations which correspond to words with the same meaning but different POS). As pointed by the authors, it is rare to find studies that combine both the analysis of model-related factors and application-related factors. Most studies tend to focus

on the importance of model-related factors only by focusing on the influence of parameters on a model's performance. Moreover, as we already showed before, specific tasks and specific applications of DSMs lack resources that would be designed for those needs. This is especially the case for specialized morphology since there is no datasets that cover both morphological and syntactic derivations.

To evaluate the trained models, Bernier-Colborne and Drouin (2016) used a thesaurus, i.e. a specialized resource, constituted of a subset of words specific to the domain. Each word was associated to other words they share different semantic relations with. Then, given a query, its nearest neighbors are retrieved using cosine similarity. The ranked list of neighbors is then compared to the gold standard and Mean Average Precision (MAP) is computed. When the related terms are closer to the top of the list on average, the MAP is high.

The use of nearest neighbors in the evaluation process is interesting. Most of the time and especially with traditional intrinsic evaluation methods, only the similarity score or the ranks of neighbors rank is used to assess the quality of a model. Nearest neighbors are usually not considered as part of the evaluation process and metric and are rather used as a proof that a system works properly. While the use of nearest neighbors along with their semantic relations is interesting, the presented approach is very similar to intrinsic evaluation datasets since a specific gold set is used to measure the performance of models.

3.2.2 Investigating linguistic change

DSMs are also used to explore specific linguistic phenomena. In particular, many recent works have relied on word embeddings to investigate the semantic shift of words, i.e. the observation of how words meanings change in different situations (over time or in corpus of different nature). Different methods exist to explore this type of linguistic change and evaluating their efficiency can be challenging. Semantic shift is a phenomenon where the use of existing datasets is not possible since part of the task is exploratory. Moreover, the analyzed shifts are corpus-dependent and are either shifts that happen over time or shifts that happen because of specialized meaning (e.g. the word *tree* has a different meaning in the BNC corpus or in a specialized NLP corpus like ACL). In that sense the use of DSMs for that particular task is pretty different from the traditional use of DSMs integrated in NLP systems where the most important thing is to reflect a generic representation of the world. In this section, we give an overview of selected works leveraging DSMs to investigate semantic shift and of the different evaluation methods implemented.

3.2.2.1 Semantic shift in the context of medias

Because language evolves over time and especially nowadays with the use of internet and social media tools, Kulkarni et al. (2015) proposed a methodology to detect semantic shift. They particularly focus on medias such as micro-blogs posts, reviews left for products and books. Different methods were tested based on frequency, syntactic and distributional aspects. We focus here on the method based on distributional semantics. To measure the evolution of a word's meaning, time series were constructed for each word. Word embeddings were trained with `gensim` on different parts of the corpus corresponding to specific periods of time. Aligning models in the same space is a challenging process when the dimensions of vectors do not correspond to identified traits. As a consequence, the different embeddings learned were aligned in the same space using words' nearest neighbors as a landmark. Once the embeddings were aligned, it was possible to measure how words shifted across different time spans. The authors also identified the moment where the semantic change occurred. To test the developed methods, experiments were run on multiple corpora: Google Books N-Gram, Amazon Movies Reviews and Twitter data. Several words were identified as undergoing semantic shift. E.g. *tape* was used to designate adhesive paper before 1960 (*red tape*, *tape from her mouth*) and then shifted to designate cassettes (*a copy of the tape*). Another example is the word *diet* that was used to designate food consumed by people (*diet of bread and butter*) while after 1970 it was used to describe a different way of eating (*go on a diet*).

3.2.2.2 Diachronic semantic change

Hamilton et al. (2016) also focused on investigating linguistic diachronic change over several time periods using different types of DSMs (count-based and word embeddings). The authors showed that with enough data available, DSMs are a good tool to detect semantic shifts. To be able to track diachronic change, first the synchronic validity of models was tested, i.e. the authors made sure that the trained models were able to capture similarity for words in the same time-period. To do so, they decided to run an intrinsic evaluation using the MEN test set[1]. Once synchronic validity was confirmed, Hamilton et al. (2016) were able to measure semantic shifts. First, word embeddings were aligned in the same space. Then, the aligned vectors were used to measure the displacement of words across different periods of time. The cosine similarity between the different word

[1]We saw in chapter 2 that intrinsic evaluation datasets evaluate only a subset of a semantic space. As a consequence we could argue that while they prove that the model does not suffer any major problem, they cannot guarantee that the model properly captures semantic relations between words.

representations over time was used to quantify the amplitude of semantic change.

The ability of models to detect semantic shifts was evaluated using a set of words known to undergo semantic shift (e.g. *gay*, *fatal*, *nice*, *broadcast* etc.) gathered from different research works done on semantic shift. The change of nearest neighbors across different periods of time allows to detect and visualize the undergone semantic shifts. E.g. the neighbors of *nice* shifted from *pleasant* and *lovely* to *refined* and *dainty*. Another part of the evaluation focused on the model's aptitude to detect unknown semantic shifts. To do so the authors examined words having the highest displacement score. This was computed using cosine similarity between the representations of a given word across several time periods. Examples of words that changed the most between 1900 and 1990 are *wanting*, *check*, *major* etc.

3.2.3 DSMs applied to specialized corpora

DSMs are often used to model general knowledge with models trained on generic corpora, i.e. corpora that do not account for specialized domains but that reflect the world's general knowledge. However, before training DSMs using the largest corpora possible became the norm, DSMs were used in the context of specialized corpora to build distributional thesauri in the 90s (Fabre and Lenci, 2015; Grefenstette, 1994). Nowadays, a growing number of works is focusing on biomedical NLP, with several conferences workshops particularly dedicated to this topic. Using DSMs for the biomedical domain cannot be efficiently done using models trained on generic corpora since the meaning of several words will be completely different in those particular contexts[2]. As a consequence, common tasks designed to evaluate word similarity in generic corpora are not applicable in the context of specialized DSMs. As such, there is a need for different types of evaluation. In this section we present two different works that focus on training and evaluating models for specialized domains (child-directed speech and NLP scientific papers).

3.2.3.1 DSMs in the context of child-directed speech

Asr et al. (2016) focus on the ability of DSMs to learn semantic categories from child-directed speech. To do so, the authors trained DSMs on the CHILDES corpus (MacWhinney, 2000), a corpus gathering conversations between children and their parents and care-givers. The corpus is relatively small with about 8 million words. Models were built using count-based and neural-based models. Because of the specificity of the corpus as well as its small size, it is not possible to use traditional datasets to evaluate models. In that case, contrary to what we

[2]We saw an example of such a phenomenon in chapter 1 with the word *apple* in the BNC and in the Medical Web Corpus.

observed with diachronic studies, it is not even possible to use those datasets to attest the validity of models. As a consequence, the authors designed a specific word categorization task. They used 1244 nouns with a high frequency in the corpus. Each word belonged to different categories that were non-ambiguous (e.g. *mammal*, *clothing* etc.). Using those words, they constituted pairs of words that either belonged to the same category (e.g. *dog* and *cat* are both *mammal*) or not (e.g. *dog* is a *mammal* while *shoe* belongs to the *clothing* category). Models were used to predict if both words belonged to the same category.

This word categorization task was used in two different contexts: to find optimal parameters for models trained with word2vec and to determine if count-based or neural-based models perform better for this type of task. The authors were able to determine that different architectures in word2vec capture semantic categories differently. E.g., the model trained using CBOW was better at capturing similarity of *furniture* items than *fruit* items. This type of evaluation is interesting because it provides insight about the properties of DSMs that are often ignored.

3.2.3.2 DSMs trained on specialized corpora

Tanguy et al. (2015) focused on models trained on the TALN corpus (Boudin, 2013), a corpus gathering french NLP papers that is made of about 2 million words. The goal of this study was to compare count-based models trained using different types of contexts (window-based and dependency-based contexts) and parameters resulting in 2592 models. To evaluate models, the authors improved a pre-existing dataset designed by Fabre et al. (2014) based on words extracted from the TALN corpus. The dataset comprised 15 words (5 nouns, 5 verbs and 5 adverbs). To improve the dataset the authors added 5 more nouns, 5 more verbs and 5 more adverbs of high and low frequency to ensure that all frequency ranges are represented in the dataset. The target words in the datasets represent both specialized (e.g. *performance*, *fréquence*) and non-specialized terms (e.g. *élément*, *correct*). For each target word, the 3 nearest neighbors were retrieved in each trained model, resulting in 64 to 445 nearest neighbors for each target word. To ensure the validity of these neighbors, 4 annotators that were NLP experts were asked to select accurate nearest neighbors. As a consequence, each selected nearest neighbor (i.e. nearest neighbors that were at least chosen by one annotator) were assigned a score between 1 and 4 corresponding to the number of annotators that chose this neighbor. After the annotation process, the dataset was constituted of 1328 pairs of words (target word and one of its closest neighbor) representing different types of semantic relations (e.g. synonyms such as *complexe* and *compliqué* or antonyms such as *complexe* and *simple*).

Models were evaluated by retrieving the 50 nearest neighbors of each target word in each of the 2592 models. These neighbors were then compared to the

selected neighbors constituting the gold standard using a metric which helped determine the best configurations. Because the dataset was created using POS and frequency information, it makes it possible to analyze and interpret results based on their POS or their frequency range allowing better understanding of the information captured by models.

They also compared the differences in ranking of nearest neighbors across the best 106 models by computing the Rank Biased Overlap (Webber et al., 2010), a measure comparing two ranked lists that considers that differences at the top of the list are more important than differences at the bottom of the list. This method showed that different models capture different type of semantic information.

The two methods presented here offer the advantage to perform advanced analyses of the type of semantic information captured by different models. However, the construction of a gold standard specific to the corpus is expensive in terms of time and resources. Moreover, the gold standard is not applicable to another corpus and experts are required to validate nearest neighbors.

3.3 Moving towards qualitative evaluation

We presented several studies using DSMs to analyze different phenomena: finding the appropriate combinations of parameters, detecting semantic shifts and using DSMs in the context of specialized corpora. The evaluation methods used in these research works tried to go beyond intrinsic evaluation using traditional datasets with tasks ranging from semantic clustering to the construction of a corpus-specific gold standard. The choice of these alternative evaluation methods was often motivated by the lack of resources (this was the case especially for specialized corpora). It is also a proof that existing methods are not satisfying for the various types of applications of DSMS. Despite their convenience, traditional intrinsic evaluation datasets often provide insufficient insights regarding the semantic information captured by DSMs.

While the work done for this PhD does not specifically focus on specialized corpora, the approach we adopt is similar in many points to the different studies presented here. We are interested in investigating as well as understanding the type of linguistic information encoded in DSMs. As a consequence, we want to go further than running evaluations limited to measure the performance of models on traditional datasets. We want to go further than simply evaluating the semantic similarity of selected pairs of words and want to be able to understand, or at least get intuitions about, what is happening in a model's semantic space as a whole. This means that we want to adopt an evaluation approach that is corpus independent (i.e. that can be applied to any corpus) and that is not relying on pre-existing resources. We want to get both a global overview of the model, by

being able to describe what is happening in general (is the model sane? Or can we already detect that there is something wrong?), and a local overview, i.e. we want to be able to zoom in selected local areas to observe the semantic representations and semantic behaviors of selected words.

To be able to get such an overview of a model, we need to approach the evaluation from two different points of view, similarly to what Galliers and Sparck Jones (1993) advised. First, we adopt a quantitative point of view to be able to operate a quick sanity check to detect any major issue with the trained models. Then we use qualitative methods to investigate semantic behavior of words using linguistic knowledge and methods. To do so, we choose to evaluate models using words nearest neighbors, since they provide immediate and interpretable feedback on the quality of semantic representations. Observing the behavior of the nearest neighbors of a word from one model to another can give us accurate insights about what is really happening when training DSMs. In the second part of this thesis, we present several experiments conducted to get a better understanding of the type of information captured by word embeddings.

Part II

Qualitative evaluation

Chapter 4

Investigating word embeddings hyperparameters

Contents

4.1 Introduction

In the previous chapters, we saw that there is still no perfect way to evaluate word embeddings. While several works focused on appropriate ways to evaluate DSMs, we still lack a methodology evaluating word embeddings both globally, i.e. by giving a general idea of the performance of a model, and locally, i.e. a methodology that allows to explore selected local areas.

Several studies have investigated the choice of parameters to build good DSMs. E.g. in chapter 1 we presented the work of Lapesa and Evert (2014) who performed a large-scale evaluation of DSMs. Another example was Bernier-Colborne and Drouin (2016) who focused on the importance of the choice of parameters for specialized lexicography. Often, the evaluation of parameters is done using intrinsic evaluation datasets without necessarily questioning the impact of the different configurations. We propose here to investigate the qualitative impact of hyperparameters choices when training word embeddings models. As a consequence, we decided to change only one hyperparameter at a time. First we explain what hyperparameters are and we present the ones selected for this study. Then we use nearest neighbors to compare models trained with one different hyperparameter. Using nearest neighbors present several advantages. They give direct feedback on the semantic representation of a word. They also are easy to manipulate and can be easily integrated in an evaluation metric. While this type of method has already been used in works investigating DSMs (e.g. (Sahlgren, 2006; Bernier-Colborne and Drouin, 2016; Tanguy et al., 2015)), the novelty of our approach lies in the fact that we propose to use nearest neighbors to evaluate models as a whole and not only selected words.

4.2 Hyperparameters

4.2.1 What are hyperparameters?

Training "good" word embeddings requires to tune a certain number of hyperparameters, such as the size of the window, vectors dimensions etc. We can distinguish between internal and external hyperparameters.

Internal hyperparameters are directly related to the algorithm used for training, e.g. this corresponds to the number of dimensions, the negative sampling rate or the subsampling rate that we described in chapter 1.

External hyperparameters are related to the input given to the system. They are also crucial to train word embeddings. These external hyperparameters can be the corpus or the type of context used for training. Another example would be the tagger used to label the training data.

In the following sections, we present the different hyperparameters we chose to investigate when training models using word2vec. Some hyperparameters in word2vec are more opaque than others. As a consequence, we decided to select the ones that are easy to manipulate and for which we can have an intuition that helps us linguistically interpreting the different observed phenomena.

4.2.2 Selected hyperparameters

We selected hyperparameters that are often investigated. For each hyperparameter presented here, we selected related works that focused on investigating those specific parameters. Table 4.1 displays all of the selected hyperparameters along with the different works that examined them with the different examined values. In this work, we do not only consider these different values and try to find the best combination of parameters. We are rather interested in approaching hyperparameters investigation by adopting a linguistic point of view and trying to understand the impact of different hyperparameters on the lexical space when training word embeddings.

4.2.2.1 Architecture

In chapter 1 we presented the two different architectures of word2vec (Skip-Gram and Continuous Bag of Words). Most of the studies focusing on training good word embeddings try both architectures but SG is often preferred to CBOW because as we already mentioned in section 1.4.1.1 it gives better results on semantic and syntactic tasks. However it is slower to train compared to the CBOW architecture and it might be preferable to use it when training on smaller corpora.

Parameters	**Architecture**	**Vectors Dim.**	**Window**	**Context**	**Corpus**
Bernier-Colborne and Drouin (2016)	CBOW, SG	100, 300	1 to 10	window	PANACEA[a]
Levy and Goldberg (2014a)	SG	300, 600	2, 5	window, dependencies	Wikipedia
Levy et al. (2015)	SG	500	2, 5, 10	window	Wikipedia
Baroni et al. (2014)	CBOW	200, 300, 400, 500	2, 5	window	ukWaC[b], Wikipedia, BNC
Melamud et al. (2016)	SG	25, 50, 100, 200, 300, 600	1, 5, 10	window, dependencies	Wikipedia, UMBC, Gigaword
Sahlgren and Lenci (2016)	CBOW, SG	200, 300, 400, 500	2	window	ukWaC, (different sizes bins)
Roberts (2016)	CBOW	100	5	window	
Asr et al. (2016)	CBOW, SG	30, 50, 100, 200, 300	2, 12	window	CHILDES[c]
Chiu et al. (2016)	CBOW, SG	25, 50, 100, 200, 500, 800	1, 2, 4, 5, 8, 16, 20, 25, 30	window	PubMed
Li et al. (2017)	SG, CBOW	25, 50, 100, 250, 500	2, 12	window, dependencies	Wikipedia

Table 4.1: Selected hyperparameters used in different studies along with values used for experiments.

[a] `http://www.panacea-lr.eu`
[b] `https://wacky.sslmit.unibo.it/doku.php?id=corpora`
[c] `https://childes.talkbank.org/access/`

In their work investigating the hyperparameters of DSMs in the context of specialized lexicography, Bernier-Colborne and Drouin (2016) tested both the CBOW and SG architectures. They found that both architectures captured different semantic relations. CBOW gave better results for synonyms while SG is better at capturing syntactic derivatives.

Sahlgren and Lenci (2016) also tested different hyperparameters for both count-based models and predictive models. For word2vec, they found that the CBOW architecture was giving better results for words in the high and low frequency range.

4.2.2.2 Number of dimensions

The number of dimensions is another hyperparameter often investigated when training word embeddings. While Mikolov et al. (2013a) advises to use more dimensions to get better accuracy results, several studies have been conducted to investigate the impact of the number of dimensions when training word embeddings. Baroni et al. (2014) compared the performances of count-based models and predictive models on different semantic tasks. They considered dimensions values of 200, 300, 400, and 500 and found that the best results were achieved with word embeddings trained with 400 dimensions. Chiu et al. (2016) tested a wider range of values (25, 50, 100, 200, 300, 400, 500, 800) for word embeddings trained for biomedical NLP. They tested the models on various extrinsic and intrinsic tasks and observed results improvements with higher dimensions. Asr et al. (2016) experimented with 30, 50, 100, 200 and 300 dimensions to train word embeddings in the context of the acquisition of semantic categories. Following Mikolov et al. (2013a), they expected the model with the highest number of dimensions to perform the best. However, they achieved better results on the word categorization task they developed when training models with 200 dimensions. Bernier-Colborne and Drouin (2016) also tested different numbers of dimensions (100 and 300). They found that models trained with more dimensions were performing better on the gold set they designed.

The different studies we presented do not give consistent results regarding the number of dimensions. Sometimes models perform better when they have the larger number of dimensions, sometimes less dimensions will give better results. This is also dependent on the task used to test models. It thus seems essential to try to understand what is impacted in the lexical space when the number of dimensions is changed.

4.2.2.3 Window size

Numerous studies have investigated the impact of the window size when training word embeddings. It has been shown that different window sizes capture different type of semantic information. E.g., Bansal et al. (2014) showed that when training models with a small window size, nearest neighbors tend to have the same POS while training with larger windows retrieve nearest neighbors that are more topically related. This is logical since with a small window the only context captured for an adjective for example, is the noun it modifies. With a bigger size window, the contexts of adjectives are more varied and include verbs, non-modified nouns, other adjectives etc. Melamud et al. (2016) trained word embeddings with different window size values (1, 5 and 10) and found that larger windows group words that are both functionally and topically similar while smaller windows group words that share the same POS and that are less topically related. Chiu et al. (2016) also tested window size values ranging from 1 to 30 and found that results are different depending on the type of the evaluation task (intrinsic tasks and extrinsic tasks). Asr et al. (2016) noticed that changing the window size from 2 to 12 when training using the CBOW architecture did not improve results on a word categorization task. Bernier-Colborne and Drouin (2016) experimented with a window size ranging from 1 to 10 and found that smaller windows are good at capturing paradigmatic relations while larger windows are good at capturing syntactic relations.

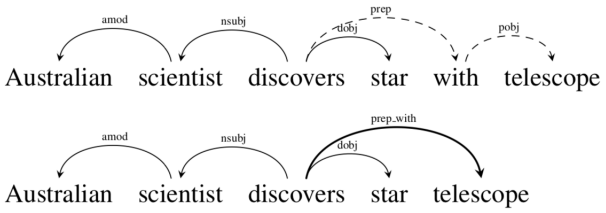
From all the examples presented above, it is clear that the window size is a highly important hyperparameter that can yield very different semantic representations. Because of the different type of information smaller and larger windows capture, it is one of the hyperparameter that we expect to highly impact the lexical space.

4.2.2.4 Context type

In chapter 1, we saw that several studies have investigated the contribution of syntactic information to DSMs. Padó and Lapata (2007) found that dependency-based models performed better at word sense disambiguation tasks than window-based models. Tanguy et al. (2015) found that using complex syntactic information when training DSMs yield better results than using raw co-occurrences on the gold set they designed for the TALN corpus. Lapesa and Evert (2017) found that dependency-based models performed better only for a noun clustering task.

Training count-based models requires to prepare contexts beforehand. While this might appear as a restriction, it actually comes with a lot of flexibility making it possible to choose to only select certain POS or certain syntactic relations. The original implementation of word2vec does not allow to simply modify the type of

contexts used for training and only the size of the window can be chosen. To be able to experiment with different types of contexts, Levy and Goldberg (2014a) modified the SG with negative sampling algorithm of word2vec making it possible to train using different types of contexts. Their modified version, *word2vecf*, does not rely on the system to create the contexts but requires to create contexts before training. This means that the contexts can be easily modified and manipulated. Levy and Goldberg (2014a) provided the code and scripts to extract contexts that are dependency-based, i.e. that integrate syntactic information[1]. To be able to extract dependency-based contexts, a dependency parsed corpus is required. Using this corpus it is possible to extract syntactic triples constituted by a dependent, a governor and the dependency relation that they share. Levy and Goldberg (2014a) suggest to collapse preposition by connecting the head and the object of the preposition.



WORD	CONTEXTS
australian	scientist/amod^{-1}
scientist	australian/amod, discovers/nsubj^{-1}
discovers	scientist/nsubj, star/dobj, telescope/prep_with
star	discovers/dobj^{-1}
telescope	discovers/prep_with^{-1}

Figure 4.1: Example of a dependency annotated sentence taken from Levy and Goldberg (2014a). The second sentence illustrates the idea of collapsing relations that include prepositions by connecting the head and the object of the preposition. The table below the sentences display the resulting extracted dependencies.

There are different ways to use dependencies for DSMs. Figure 4.1 displays an example of a dependency annotated sentence and the resulting extracted triples from Levy and Goldberg (2014a). As we can see, the contexts extracted to train are selected depending on the relations shared by words. In a window-based model the target word *scientist* would have *star* associated as one of its contexts. However, this is not the case with dependency-based contexts since these two words are not related by any dependency relation. While dependencies are interesting to build

[1] `https://bitbucket.org/yoavgo/word2vecf/src/default/`

DSMs, it is important to be aware that they can sometimes be problematic because they are highly dependent on the parser used.

Levy and Goldberg (2014a) showed that training word embeddings using different type of contexts has an impact on the type of nearest neighbors retrieved by models. Raw co-occurrence models capture domain similar words while models trained using dependency-based contexts capture words sharing the same semantic type.

Figure 4.2 displays selected examples from Levy and Goldberg (2014a) of words and their nearest neighbors retrieved from models using window contexts of size 5 and 2 (BoW5 and BoW2) and models using dependency contexts (DEPS). We see that for the word *batman* all models retrieved the same type of nearest neighbors, i.e. other super heroes names (e.g. *aquaman*, *catwoman* etc.). For the word *hogwarts* window-based models retrieved words related to the Harry Potter universe (*dumbledore*, *hallows* etc.) while the dependency-based model retrieved names of schools (e.g. *sunnydale*, *collinwood* etc.).

Target Word	BoW5	BoW2	Deps
batman	nightwing	superman	superman
	aquaman	superboy	superboy
	catwoman	aquaman	supergirl
	superman	catwoman	catwoman
	manhunter	batgirl	aquaman
hogwarts	dumbledore	evernight	sunnydale
	hallows	sunnydale	collinwood
	half-blood	garderobe	calarts
	malfoy	blandings	greendale
	snape	collinwood	millfield

Figure 4.2: Examples of nearest neighbors retrieved for the words *batman* and *hogwarts* from models trained using bag of words contexts (BOW5 and BOW2) and model trained using dependency-based contexts. Example taken from Levy and Goldberg (2014a).

Melamud et al. (2016) used word2vecf to experiment with the impact of different contexts types on extrinsic and intrinsic evaluation tasks. They found similar results to Levy and Goldberg (2014a) regarding the differences in topical and functional similarity. They also found significant differences of results on evaluation tasks. E.g. they observed low performances for the dependency-based model on the WordSim-353 relatedness test set. However the dependency-based model performed better on the WordSim-353 Similarity test set and on SimLex-999. They also observed that dependency-based models yielded better results on parsing tasks. Similar results were observed by Li et al. (2017).

4.2.2.5 Corpus

The last hyperparameter presented here is the corpus used to train word embeddings. The corpus constitutes the source, the raw material from which the semantic representations are learned. With recent neural-based word embeddings, it is popular to attest that the more data used when training the better the trained model and Mikolov et al. (2013a) recommends to use a larger corpus to get better accuracy results. It is also important to take into account the type of corpus used when training. To go back to the example presented in chapter 1, *apple* can mean different things in a general corpus and in a medical corpus.

We are aware of only a very limited number of studies investigating the impact of the corpus used for training and most of these studies solely focus on the size of the data used. Asr et al. (2016) experimented with training models using a small corpus (child directed speech data in the CHILDES corpus). They found that word2vec models are outperformed by count-based models where Principal Component Analysis was applied (dimension reduction method) when training with a small amount of data. Sahlgren and Lenci (2016) investigated the amount of data needed to train good word embeddings. They performed a comparative study with count-based models to determine which type of models to use when the quantities of available data are limited. The models were evaluated using different benchmarks: the TOEFL and ESL multiple choice vocabulary test set as well as MEN, SimLex-999 and Stanford Rare Words. Their results showed that word embeddings were not performing the best with small data. However, they also found that no model was displaying good performance on small amounts of data. Chiu et al. (2016) also experimented with different corpus sizes in the context of biomedical NLP. They were surprised to find that the corpus size impact was not what they initially expected with more data being detrimental to the performances of their models.

We presented several hyperparameters used to train word embeddings as well as selected studies that investigated the impact of those hyperparameters on the quality of word embeddings. Results were sometimes different from what was expected, e.g. adding more dimensions is not always helping to improve results or adding more data is not necessarily beneficial when training word embeddings for biomedical NLP. Most of the studies research done on hyperparameters are carried out in the context of specific tasks. To understand better the impact of each hyperparameter, we propose in the following section an experiment that helps us analyzing the impact of each hyperparameter used to train word embeddings.

4.3 Method

4.3.1 Models and hyperparameters

In this section, we present the different models trained for our experiment. One model (default model) serves as the basis of comparison. All the other models are trained using one different parameter from the default model.

4.3.1.1 Default model

The purpose of the DEFAULT model is not to achieve good performances in any way. It is simply a model that constitutes the basis to which all other models are compared to. To train this model we used word2vec original source code with the default values for all hyperparameters. However, we set the architecture to SG since most people prefer this architecture when training models with word2vec. This is explained by the fact that SG, while slower to train yields better results. We also used window-based contexts by default. The default hyperparameters are the following:

- window size: 5,
- vector size: 100,
- negative sampling rate: 5,
- subsampling rate: 1e-3,
- number of iterations: 5.

We did not experiment with the negative sampling rate, the subsampling rate and the number of iterations (which corresponds to the number of times the training data is seen). As a consequence for each model trained, their values remain the ones indicated here. We also decided to set the min-count hyperparameter to a value that remains the same across trainings. Min-count corresponds to a threshold that determines the minimum frequency words must have to be taken into account for training, To set the frequency threshold value, we followed literature recommendation and set a threshold of 100 occurrences. This means that we only considered words appearing at least 100 times to be included in our models.

Corpus used To train the default model we used the British National Corpus[2]. The BNC is made of samples of written and spoken texts that were assembled to give an overview of British English at the end of the 20th century. Overall, the BNC contains about 100 million words and around 10 percent of the BNC correspond to transcripts of spoken texts. Because it is too different from the

[2] `http://www.natcorp.ox.ac.uk/`

rest of the corpus, we decided to remove the spoken texts. The corpus we used is thus made of around 90 million words. Given the size of corpora used today, the BNC is considered a small size corpus. However, it is regularly used in studies investigating DSMs (see for example Lapesa and Evert (2014) or Padó and Lapata (2007)). As a consequence we decided to used it as well to train our models.

We wanted to keep POS information when training word embeddings to be able to detect how hypeparameters impact different POS. To do so, we lemmatized and POS-tagged the BNC corpus with Talismane (Urieli, 2013).

The POS-tagset used (Penn Treebank tagset[3]) distinguishes between verbs in their base form, in their past participle form etc. Because we wanted to maximize the number of contexts for each word, we decided to group together selected POS. We grouped POS as shown in table 4.2.

Original POS	**Mapped POS**
VBG, VBD, VBN, VBP, VBZ, VB	VB
NNS, NN	NN
NNPS, NNP	NNP
JJS, JJR, JJ	JJ
RBS, RBR, RB	RB

Table 4.2: Mappings for POS.

Example (1) shows a tagged and lemmatized sentence before grouping POS. The simplified version is shown in example (2).

(1) VBZ#do DT#the NN#lead VB#cause DT#the JJR#lower P#' NNP#iq #' P#?

(2) VB#do DT#the NN#lead VB#cause DT#the JJ#lower P#' NNP#iq P#' P#?

4.3.1.2 Variant models: hyperparameters variation

The DEFAULT model serves as a basis to make hyperparameters vary. Models were trained with one different hyperparameter value at a time to isolate as much as possible the effect of the given hyperparameter[4]. The evaluated hyperparameters are the one we presented before:

[3] `https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html`

[4] As we mentioned before, word2vec implies some random processes such as the starting seed for training. We chose to not fix the seed to the same value every time and controlled only the hyperparameters presented before.

- architecture,
- number of dimensions,
- window size,
- context type,
- corpus.

To choose the different values to be examined, we relied on the different values tested in the literature. Those values are reported in table 4.3.

Hyperparameters	**Default**	**Alternate values tested**
Architecture	SG	CBOW
Vectors dimensions	100	50, 200, 300, 400, 500, 600
Window size	5	1 to 10
Context type	window	deps, deps+
Corpus	BNC	ACL

Table 4.3: Hyperparameters values used to train word embeddings that are compared.

Architecture We decided to test both architectures: CBOW and SG.

Number of dimensions The number of dimensions is another important factor when training word embeddings. The default value in word2vec is 100. We wanted to test a smaller value as well as larger values, following the different studies we presented in section 4.2. We thus chose to test the following values: 50, 200, 300, 400, 500 and 600.

Window size In word2vec C implementation the default window size is 5. In order to get a good overview of the impact of the window size we decided to test values ranging from 1 to 10.

Context type In section 4.2.2.4, we presented the modified version of word2vec developed by Levy and Goldberg (2014a). We decided to use it to experiment with dependency-based contexts. Table 4.4 below shows a sentence from the BNC corpus that was lemmatized, POS-tagged and parsed (CoNLL format) using Talismane[5].

[5]Recently, Tanguy et al. (2019) investigated several parsers for specialized corpora. They found that Talismane yielded the best results.

1	The	the	DT	NMOD	2
2	loss	loss	NN	SBJ	5
3	of	of	IN	NMOD	2
4	communication	communication	NN	PMOD	3
5	did	do	VBD	ROOT	0
6	not	not	RB	ADV	5
7	affect	affect	VB	VC	5
8	the	the	DT	NMOD	9
9	mission	mission	NN	OBJ	7
10	.	.	P	P	5

Table 4.4: Example of a lemmatized, POS-tagged and parsed sentence from the BNC.

The dependency relations used are the ones from the Penn Tree Bank converted to the Stanford Dependencies[6]. Using the annotated corpus, we extracted triples with the structure *head modifier#reltype*. These triples constitute the input to train word2vecf. We wanted to assess the impact of the dependencies on the models. To do so we trained embeddings using two different types of triples. For the DEPS triples, we used the scripts provided by Levy and Goldberg (2014a)[7]. All dependencies relations are extracted and only prepositions are "collapsed" so that the head and object of the preposition are connected[8].

For the DEPS+ triples we only selected a set of syntactic relations based on Padó and Lapata (2007) work. The following dependency relations were selected: subject, noun modifier, object, adjectival modifier, coordination, apposition, prepositional modifier, predicate, verb complement. Prepositions were still collapsed à la Levy and Goldberg and the same was done for conjunctions.

Table 4.5 and 4.6 display the triples extracted for DEPS and DEPS+ for the sentence in table 4.4

For DEPS, the inverse relation is annotated with `I` while in DEPS+ it is annotated with `-1`. We see that there are fewer triples for DEPS+ since we only selected a specific set of dependency relations.

[6] `https://nlp.stanford.edu/software/stanford-dependencies.shtml`

[7] `https://bitbucket.org/yoavgo/word2vecf/src/default/scripts/`

[8] An example of collapsed preposition is displayed in figure 4.1

Word	**Context**
nn#loss	nmod_dt#the
dt#the	nmodI_nn#loss
vb#do	sbj_nn#loss
nn#loss	sbjI_vb#do
nn#loss	nmod_in#of
in#of	nmodI_nn#loss
in#of	pmod_nn#communication
nn#communication	pmodI_in#of
vb#do	adv_rb#not
rb#not	advI_vb#do
vb#do	vc_vb#affect
vb#affect	vcI_vb#do
nn#mission	nmod_dt#the
dt#the	nmodI_nn#mission
vb#affect	obj_nn#mission
nn#mission	objI_vb#affect
vb#do	p_p#.
p#.	pI_vb#do

Table 4.5: Triples extracted for DEPS using Levy and Goldberg (2014a)'s scripts for the sentence in table 4.4.

Word	**Context**
vb#do	nn#loss#sbj
nn#loss	vb#do#sbj-1
nn#loss	nn#communication#pmod:of
nn#communication	nn#loss#pmod:of-1
vb#affect	nn#mission#obj
nn#mission	vb#affect#obj-1

Table 4.6: Triples extracted for DEPS+ for the sentence in table 4.4.

Corpus We saw that most studies considering the corpus as one of the parameter used for training focused only on the size of the corpus. We wanted to test other types of texts and domains to train word embeddings. Because the method we propose is not relying on benchmarks, we wanted to choose a corpus that is specialized in a domain we are familiar with. As a consequence, we chose the ACL

corpus (Bird et al., 2008) as an alternative corpus. It is constituted of NLP scientific papers from the ACL Anthology. Its size is similar to the BNC with about 100 million words. Because the BNC and ACL are two very different corpora, we expect a lot of variation in the semantic representations of words. However, we might observe interesting phenomena that are independent of the corpus nature.

Starting from the default configuration, we trained one model per possible hyperparameter value listed in table 4.3. We thus trained 20 models. To compare these models to the DEFAULT model, we needed to use an evaluation metric that will allow us to approach word embeddings evaluation differently. We present the metric we chose in the next section.

4.3.2 Variation metric

In the next section we explain why we decided to use nearest neighbors to compare DSMs and how we integrated them in a metric that is easy to understand and manipulate.

4.3.2.1 Nearest neighbors for embeddings evaluation

There are numerous works on the evaluation of DSMs that aim at finding the best set of hyperparameters to train word embeddings that will perform optimally and improve state of the art results. The work presented throughout this thesis does not pretend to *improve* word embeddings models but rather aims at providing clues and insights that helps understanding those models. We want to examine how information is encoded and organized in the semantic space of a model. To do so we need to access information in the model that reflects what is happening in the lexical space. Using only intrinsic evaluation datasets, it is not possible to get enough information and material to investigate the lexical structure of word embeddings models. Moreover those types of approach do not allow to really compare what is changing from one model to another. If we were training word embeddings for a specific task, we could design a custom evaluation task. However, because our work is exploratory, we want to use a method that gives us both a global overview of how a model compares to another by quantifying the amount of difference as well as a method that allows us to explore selected areas more thoroughly.

By using a metric that involves nearest neighbors, we can get information about the degree of difference of a word neighborhood between two models but we can also retrieve concrete linguistic material that can provide an entry point to investigate what varies from one model to another.

Using a metric based on nearest neighbors also presents several advantages. First, it is not dependent on a specific resource. Then, by just looking at the nearest neighbors of a word without performing any computation we can already get an idea of the quality of the semantic representation as well as observe which semantic senses were captured by the model. Finally, using nearest neighbors provide landmarks that are easy to assess. Because neighbors are retrieved as an ordered list, sorted from most to least similar, a rank can be associated with every nearest neighbor. It is thus easy to compare if two neighbors are the same or different for a given rank or to check if a neighbors rank has changed from one model to another.

The type of approach we present here was already proposed by Sahlgren (2006) who used it to compare syntagmatic and paradigmatic word space models by measuring their overlap. It also corresponds to usage of DSMs in corpus linguistics where the quality of a semantic representation is evaluated through its nearest neighbors. Nearest neighbors are also often used as a proof that a model works properly. We saw it for word2vecf with the nearest neighbors of *hogwarts* (see figure 4.2). The problem of simply looking at the nearest neighbors without using a metric is that it might lead to over interpretation. It is necessary to measure what is changing between word spaces and it is possible to do so by comparing the nearest neighbors of a target word in two models. To select the nearest neighbors to be compared, it is possible to either set a threshold corresponding to the minimum similarity score to be considered or to select a given N number of neighbors. The overlap between the two models is then computed on those selected neighbors. A large overlap means that the two semantic spaces are very similar while a low overlap means that the two spaces are very different (Sahlgren, 2006).

In the following section, we present in more details the variation metric we used to compare models trained using different hyperparameters.

4.3.2.2 Nearest neighbors variation score

While inspired by Sahlgren (2006), our approach brings novelty by comparing a new type of models (word embeddings) and by using the global overlap score to identify local zones of words that we then investigate individually.

To compute the variation between models, we propose to compute the degree of nearest neighbors variation between two models. For two models M_1 and M_2, we first get the common vocabulary since it might change from one model to another depending on the corpus or the types of contexts used for training. More precisely the variation score of a word *varnn* across two models M_1 and M_2 is measured as:

$$varnn^{N}_{M_1,M_2}(w) = 1 - \frac{|nn^{N}_{M_1}(w) \cap nn^{N}_{M_2}(w)|}{N} \tag{4.1}$$

$nn_M^N(w)$ represents the N words having the closest cosine similarity score with word w in a distributional model M.

For example, let's consider the word *cat*. Its 3 nearest neighbors in a model trained on ACL are *dog*, *bark* and *fish*. In a model trained on the BNC its nearest neighbors are *monkey*, *dog* and *fox*. As a consequence the variation score of *cat* across these two models is:

$$varnn^3_{M_1,M_2}(cat) = 1 - \frac{1}{3} = 0.66$$

Using a cosine score threshold to limit the number of nearest neighbors does not appear to be the best solution. The main problem would be to find a good way to set that threshold. Cosine scores can vary greatly from one word to another and some words' top neighbor might have a low cosine score. Would that mean no neighbor should be selected if it does not reach the fixed threshold? Moreover, retrieving neighbors according to a given threshold might retrieve a different number of neighbors for two models that are compared. Is it then possible to compare models by considering a different number of neighbors? To avoid this type of situations it seems more reliable to select a fixed number N of nearest neighbors.

Even though we decided to select a fixed number of nearest neighbors to compare models, we still need to set the number of neighbors. To do so, we need to consider several factors. Choosing a too small number could lead to lack of material to be examined. On the other hand choosing a too high number could provide unnecessary information that could lead to overlook important information. We thus decided to experiment with different values to be able to select an appropriate and representative value.

We retrieved neighbors according to the following candidate values of N: 1, 5, 10, 15, 25, 50 and 100. For each value we computed the variation score for each word of the vocabulary for two randomly selected models (DEFAULT and WIN10 models). We then computed the correlation matrix between all different values of N. The matrix is reported in table 4.7. We can observe an important decrease of the correlation for the values 1 and 5. We need to use the value of N where the correlation coefficients are maximized. This corresponded to the value of 25. We thus chose to always compute the variation score with $N = 25$.

Values of N	N1	N5	N10	N15	N25	N50	N100
N1	1	0.42	0.35	0.33	0.30	0.28	0.26
N5	0.42	1	0.78	0.70	0.65	0.60	0.56
N10	0.5	0.78	1	0.89	0.80	0.74	0.69
N15	0.33	0.71	0.89	1	0.89	0.81	0.76
N25	0.30	0.65	0.80	0.89	1	0.90	0.84
N50	0.28	0.60	0.73	0.81	0.90	1	0.93
N100	0.26	0.56	0.69	0.76	0.84	0.93	1

Table 4.7: Correlation matrix between the different values of N for the DEFAULT and WIN10 models.

We also decided to compute the variation only for open-class parts of speech i.e. nouns, verbs, adjectives and adverbs since these open classes are more likely to be impacted by variation. However, to preserve the geometry of the lexical space we did not filter out nearest neighbors according to their POS meaning that all POS can be found among nearest neighbors.

We are aware that this measure present the disadvantage of not accounting for the rank of neighbors and only the presence or absence of a nearest neighbor is considered. However, the variation measure we presented has several advantages. First, it can easily be interpreted since we directly get an idea of the number of neighbors that vary for a given word between two models. Moreover, it is not computationally expensive. This measure also allows to get both a global overview, by being able to compute the average variation score for a model, and a local overview, by computing the variation score for every word. This means that it is easy to examine words that vary more or less.

4.4 Results

In this section we examine the variation score between the 20 models trained. Before comparing models, we first run an intrinsic evaluation task. Although this task is not going to satisfy our linguistic curiosity, we saw before that many research works were first performing a sanity check to make sure that the models do not present any major problem.

4.4.1 Quantitative Evaluation

To perform the quantitative part of our evaluation, we selected two datasets that are commonly used to evaluate DSMs in the literature: SimLex-999 (Hill et al.,

2015) and WordSim-353 (Finkelstein et al., 2002). Because our models are POS-tagged, we automatically added POS in the datasets using the available information (SimLex-999 contains POS information and WordSim-353 is constituted of pairs of nouns).

Figures 4.3, 4.4 and 4.5 display the performances of the different models on both WordSim-353 and SimLex-999 as well as the confidence interval for the DEFAULT model. We can see that depending on the hyperparameters we changed, there are more or less differences in models' performance. However these differences are generally not significant.

By taking a closer look at figure 4.3 we see that the performances of the different models are not consistent between the two benchmarks. On WordSim-353 most models perform similarly independently of the number of dimensions. However, on SimLex-999 we observe more notable performance differences. Models with fewer dimensions have less good performance results than models with a higher number of dimensions.

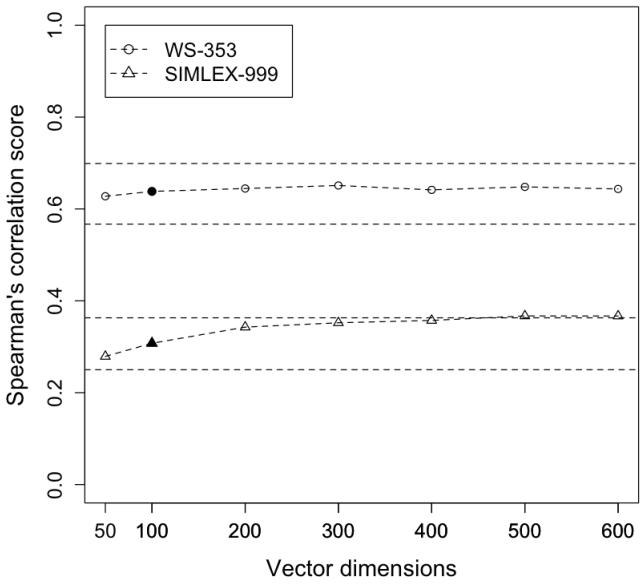


Figure 4.3: Evaluation results for models trained with different vector size (DIM models) on WordSim-353 and SimLex-999 with 95% confidence interval span computed from DEFAULT model. DEFAULT model is shown in bold.

We observe a similar tendency on figure 4.4 with different results on the two benchmarks. Increasing the window size gives better results for WordSim-353 but not necessarily for SimLex-999. However, results are less good when increasing

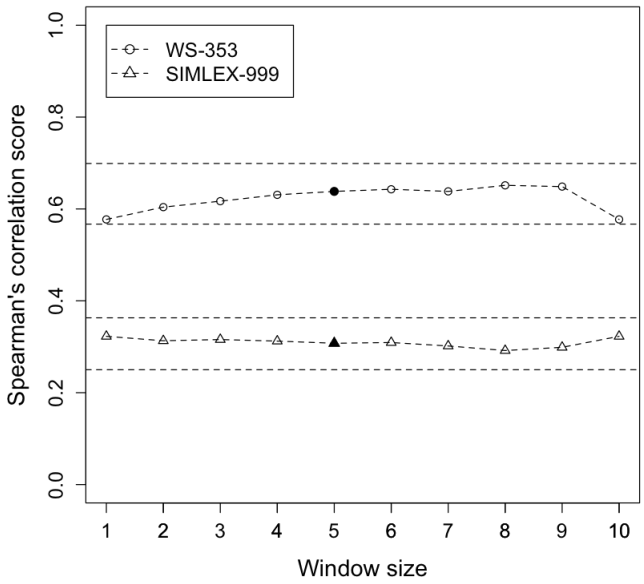


Figure 4.4: Evaluation results for models trained with different size windows (WIN models) on WordSim-353 and SimLex-999 with 95% confidence interval span computed from DEFAULT model. DEFAULT model is shown in bold.

the window size to the maximum value of 10. The performance score is actually similar to the model trained with a window size of 1. On SimLex-999 results are pretty similar for the different window sizes even if increasing the window size actually seems to give slightly worse results. We also notice that models trained with a window size of 1 and 10 have similar performances on SimLex-999. This is surprising since this corresponds to the most extreme values tested and we would expect both models to perform differently.

Figure 4.5 displays the performance for models trained with a different architecture (SG and CBOW), a different corpus (BNC and ACL) and a different type of contexts (window and dependency-based contexts). Differences in performances for those models are more noticeable on both datasets. This is not surprising because the hyperparameters changed in those models are expected to drastically change the semantic space since in one case completely different information is provided to the algorithm (the semantic world is completely different in ACL compared to BNC) and another type of information is provided to the model to learn from (dependency-based contexts can capture information that is further away than information captured by a window of size 5).

The DEFAULT model trained with SG performs slightly better than all the

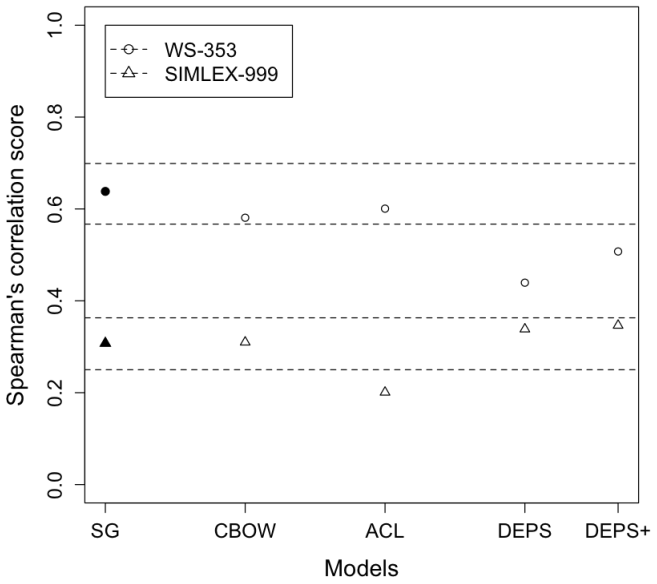


Figure 4.5: Evaluation results for models trained with a different architecture (CBOW), different contexts (DEPS and DEPS+) and a different corpus (ACL) on WordSim-353 and SimLex-999 with 95% confidence interval span computed from DEFAULT model. DEFAULT model is shown in bold.

other models on WordSim-353. However its performance on SimLex-999 is very similar to the CBOW model and it is outperformed by DEPS and DEPS+ on SimLex-999. DEPS and DEPS+ get the worst results on WordSim-353 but the best results on SimLex-999. While trained on a very different corpus, the ACL-based model performs better than the CBOW model (trained with the BNC) on WordSim-353. However it gets the worst results on SimLex-999.

Generally speaking, we can say that there are some differences between the models trained according to the benchmarks. First, all models have lower performance scores on SimLex-999 compared with WordSim-353. The number of pairs is different so it could influence the results but this could also be a reflection of the fact that the two datasets are evaluating different types of information. Secondly, it is important to be aware that while models perform differently on the same dataset when we change only one hyperparameter, those differences are generally not significant (the differences in Spearman's correlation score are very low).

Because evaluating using this type of datasets only assesses the performance and quality of a few hundreds of pairs of words, it is difficult to know what changes from one model to another. Even if the evaluation scores remain similar from one

model to another, it does not guarantee that the same pairs of words have the same cosine values.

Using the evaluation measure we presented in section 4.3.2.2, we want to go further than just investigating selected pairs of words. In the next section we show that by evaluating the variation of nearest neighbors for each word in the model, we are able to get some insights that help us understand the impact of each hyperparameter when training word embeddings.

4.4.2 Qualitative Evaluation

To perform a qualitative comparison of our models, we computed the nearest neighbors variation score between the DEFAULT model and every other model trained with a different hyperparameter.

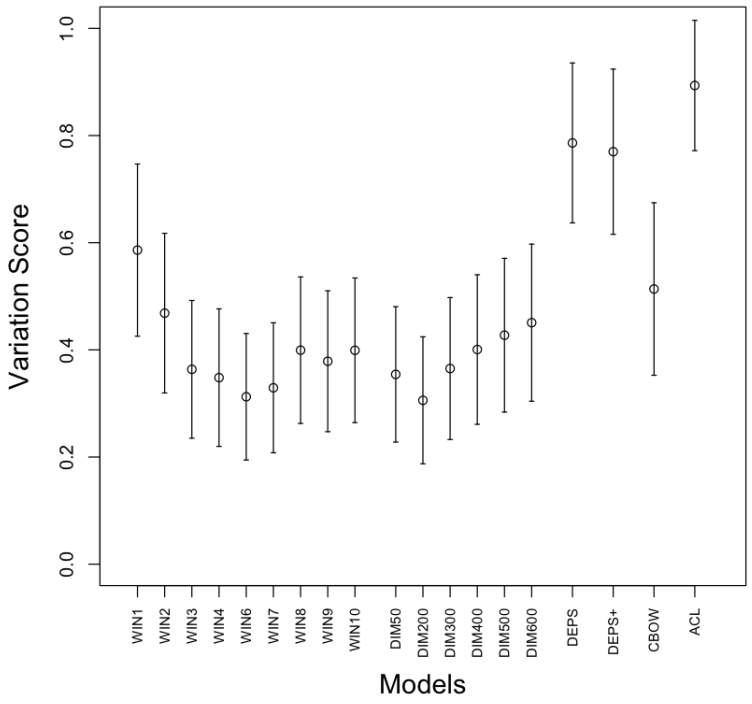


Figure 4.6: Mean variation value with standard deviation interval for all trained models compared to DEFAULT model.

4.4.2.1 Variation score

Figure 4.6 shows the mean variation score (computed on all common words between models) with the standard deviation span between the DEFAULT model and the 19 other models. The size of the vocabulary is different for models trained on the BNC and the model trained on the ACL corpus. When comparing models trained on the BNC, the neighbors of 27437 words were evaluated. When comparing the DEFAULT model to the ACL model the vocabulary size was much smaller with the evaluation of 10274 words. As we can see, it is obvious that training using different hyperparameters triggers a high variation between nearest neighbors from one model to another with a variation score of at least 0.3. This means that by changing only one hyperparameter, among the 25 nearest neighbors of a given word about 1 neighbor out of 3 is missing from one model to another.

We computed paired t-tests between models presenting the same sample size (e.g. we computed paired t-tests between each WIN model and every other WIN model as well as every DIM model since they are constituted of the same words). We found that the differences in variation are generally significant with $p < 0.05$. However the differences between WIN8 and WIN10 and WIN8 and DIM400 were not significant ($p = 0.66$ and $p = 0.10$ respectively) meaning that training a model with a window of size 8 or 10 or training a model with a window of size 8 or 400 dimensions had the same impact on the model.

The ACL model is the one showing the highest variation with an average score of 0.8. This variation was expected since both models were trained on different corpora and the ACL corpus is a specialized corpus.

The high variation displayed by the DEPS and DEPS+ models is slightly more surprising. We saw in section 4.2.2.4 that when trained using different contexts, word embeddings capture different type of semantic information, this could explain the high variation observed. Models showing the lowest variation are models with less drastic differences with the DEFAULT model. E.g. when the vector size was changed from 100 to 200 or when the window size was changed from 5 to 6. A general tendency is that models trained with minimum and maximum values for a given hyperparameter show more variation. For example the WIN1 and WIN10 models display more variation than the WIN6 and WIN7 models.

The performance scores on WordSim-353 and SimLex-999 that we presented in section 4.4.1 were not displaying such a high variation. Scores were sometimes different but those differences in scores were generally not significant. It is interesting to notice here that models with a performance score close to the DEFAULT model can still display a high variation in their neighbors. E.g. the DIM600 model had a performance score very similar to the DEFAULT model. However its variation score is higher than 0.4.

Figure 4.6 displays a high standard variation for each model. This means that

there are different behaviors across the lexicon and some words vary more than others. In the next section we propose to manually observe some words displaying high variation and some words displaying lower variation scores as well as factors that could explain variation.

4.4.3 Factors explaining the variation

Variation does not impact all words in the same way. To continue our investigation at a global level, we decided to explore two factors, the POS of a word and its frequency, that could explain the difference of variation among words.

4.4.3.1 POS

We first investigated the interaction between the POS of a word and its variation score. Figure 4.7 displays the relation between the POS of words and their variation scores for two models presenting high variation, CBOW and WIN1. The labels correspond to the tagset used by Talismane, *jj* stands for adjectives, *nn* for nouns, *nnp* for proper nouns, *rb* for adverbs and *vb* for verbs. We observe a few outliers in both graphs mainly for nouns and adjectives. However, we see that the boxplots are mostly aligned, meaning that the repartition of the variation is similar independently of the POS. Only proper nouns displayed a mean variation score slightly higher than all the other POS in both models. Similar tendencies were observed for all models meaning that the POS does not explain the measured variation.

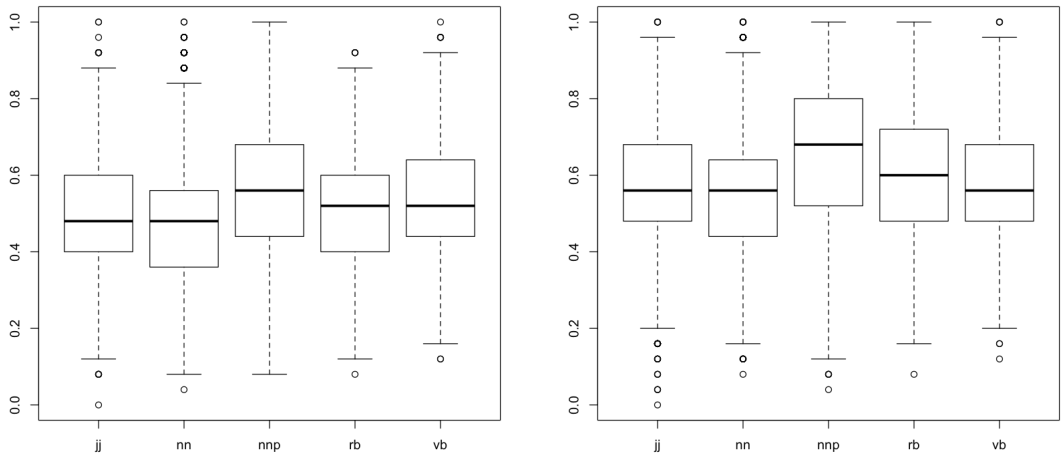


Figure 4.7: Variation score per POS for the CBOW (on the left) and WIN1 (on the right) models.

4.4.3.2 Frequency

The second factor we investigated is the role played by frequency on the variation score. Schnabel et al. (2015) showed that word embeddings encode information about frequency even after length normalization and that this could be the cause of variability across embeddings and evaluation methods. As a consequence, we decided to examine the role of the frequency in the variation of nearest neighbors. Figure 4.8 and 4.9 display the mean variation score given the frequency of a word. We see that the effect of frequency over variation is not linear. For all window-based models, we observe a clear pattern: words in the mid-frequency range (1000 to 10000) display less variation than words in lower and higher frequency ranges. This is in line with Sahlgren and Lenci (2016) who showed that DSMs perform the best for medium to high-frequency ranges items. Models trained with different dimensions seem less affected by frequency. The variation is quite constant across all frequency ranges. CBOW, DEPS and DEPS+ follow the same pattern than the window models, with a variation less high for medium frequency words. ACL[9] displays a very high variation for low frequency words but the variation decreases with frequency.

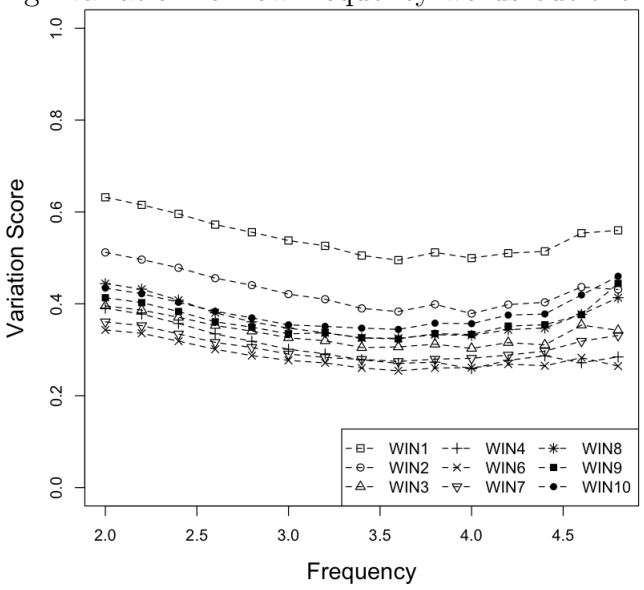


Figure 4.8: Effect of frequency on words variation for models trained with different window sizes. The frequency is indicated in log base 10.

[9]The variation for ACL was measured on a smaller vocabulary set. The frequency used in Figure 4.8 is the one from the BNC.

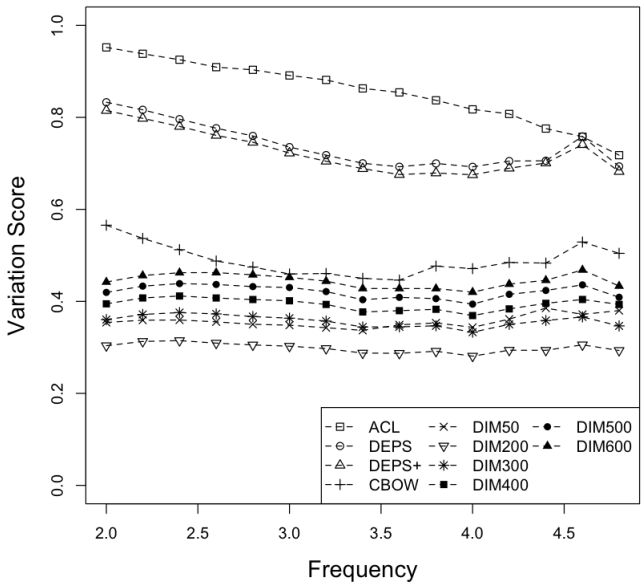


Figure 4.9: Effect of frequency on words variation for models trained with different dimensions, architecture, contexts and corpus. The frequency is indicated in log base 10.

4.4.3.3 Exploring the variation

We used the variation measure to examine more local differences. For given pairs of models we can easily identify which words show the most extreme variation values. Table 4.8 displays the 38 words varying the most in the ACL model. We see that all the words presented have a variation score of 1. This means that they do not share any of their 25 nearest neighbors with the DEFAULT model. This is not surprising since the ACL and BNC corpora are very different. We notice some tagging errors with some nouns tagged as proper nouns (e.g. *mission* and *flight*). However, we do notice that there are many proper nouns among the words that vary the most. We also observe words that have a specialized meaning in ACL.

Word	Variation score
jj#proximal	1
nn#casing	1
nn#algorithm	1
vb#slash	1
nn#forest	1
nn#grass	1
nnp#doc	1
nn#token	1
nn#multiplier	1
nnp#salim	1
nnp#mission	1
nnp#flight	1
nnp#monde	1
nn#voice	1
nnp#kuhn	1
nnp#grove	1
nn#contradiction	1
nn#exponent	1
nnp#gerhard	1
rb#qualitatively	1
nnp#ben	1
nnp#dd	1
nnp#hercules	1
nnp#hubert	1
nnp#viktor	1
nnp#or	1
jj#intractable	1
nn#gloss	1
nnp#lucas	1
nn#presenter	1
nnp#live	1
nnp#district	1
nn#boss	1
nnp#workshop	1
nnp#jay	1
nn#development	1
nn#tracing	1
nn#independence	1

Table 4.8: Words displaying the most variation when comparing the DEFAULT and ACL models.

Table 4.9 displays the nearest neighbors of the word *forest* in the ACL and DEFAULT models. We see that in ACL, its nearest neighbors refer to the algorithm (*hypergraph*, *parse*) while in the DEFAULT models, the nearest neighbors are related to nature (*woodland*, *meadow* etc.).

ACL model	**DEFAULT model**
packed	woodland
hypergraph	marsh
tree	grassland
parse	meadow

Table 4.9: Nearest neighbors of the word *forest* in the ACL and DEFAULT models.

We also investigated words varying the least for the ACL model. Table 4.10 displays the 38 most stable words. Just by looking at that table, we are able to identify some regularities for words varying the least. We observe numerous adverbs (*consequently*, *sufficiently*, *furthermore* etc.) and we also see numerals (*3rd*, *2nd*, *sixth* etc.).

Word	Variation score
rb#moreover	0.28
nnp#e.	0.28
nnp#c.	0.28
nnp#o.	0.32
nnp#n.	0.32
nnp#w	0.32
nnp#b.	0.32
jj#hungarian	0.32
nn#afternoon	0.32
rb#consequently	0.32
rb#sufficiently	0.32
nnp#l.	0.32
rb#usually	0.32
rb#furthermore	0.32
rb#however	0.32
nnp#t.	0.32
nnp#march	0.32
jj#less	0.32
nnp#april	0.32
nnp#f.	0.36
nnp#a.	0.36
jj#greater	0.36
rb#nevertheless	0.36
rb#mostly	0.36
jj#several	0.36
rb#thirdly	0.36
rb#dramatically	0.36
nnp#d	0.36
rb#secondly	0.36
rb#unfortunately	0.36
jj#2nd	0.36
jj#sixth	0.36
nnp#2nd	0.4
nnp#d.	0.4
nnp#r.	0.4
jj#easier	0.4
jj#3rd	0.4
rb#mainly	0.4

Table 4.10: Words displaying the least variation when comparing the DEFAULT and ACL models.

We wanted to see if it was possible to identify regularities among words that vary the most and the least in the different models. To do so, we selected two models, ACL and DIM200. We based our observations on the first hundreds of words displaying the most and the least variation compared to the DEFAULT model. Table 4.11 displays selected examples of words that vary the most and the least for both models. We chose to ignore phenomena that seemed to be the result of errors (e.g. tagging errors) and classes of words that were less interesting (such as *e.*, *c.* etc.). We were able to identify different semantic classes that emerge in each case. For stable words, these classes seem to correspond to dense clusters (e.g. nationalities), each word having all others as close neighbor. For example among the nearest neighbors of *hungarian* in the ACL model we observe other nationalities such as *german*, *polish*, *russian*, *french* etc.

We found that some of these clusters remain the same across the two pairs of models (e.g. nationality adjectives) while other clusters are different. In the ACL model, we find a cluster of time nouns while in the DIM200 model we find family nouns. We see that words varying the most for the specialized corpus are words carrying a specific meaning (e.g. *nominal*, *graph*). We also observe that words with a high variation score are highly polysemic or generic in the DIM200 model (e.g. *field*, *marker*).

Model	**Var.**	**Identified semantic classes**
ACL	Low	numerals (*2nd*, *14th*, *10th*...) nationalities (*hungarian*, *french*, *danish*, *spanish*...) time nouns (*afternoon*, *week*, *evening*...)
	High	specialized lexicon (*embedded*, *differential*, *nominal*, *probabilistic*, *patch*, *spell*, *string*, *graph*...)
DIM200	Low	numerals (*40th*, *15th*...) nationalities (*hungarian*, *dutch*, *french*, *spanish*...) family nouns (*grandparent*, *sister*, *son*, *father*...)
	High	generic adjectives (*all*, *near*, *very*, *real*...) polysemic nouns (*field*, *marker*, *turn*, *position*...)

Table 4.11: Words showing lowest and highest variation for ACL and DIM200 compared to the DEFAULT model.

4.5 Conclusion

We experimented with selected hyperparameters when training word embeddings and saw that evaluating on intrinsic evaluation datasets was not identifying major

differences between the trained models. Using a metric that compares the nearest neighbors between models, we were able to detect an important variation among nearest neighbors when training models with one different parameter. This variation shows that changing hyperparameters has an impact on the lexical space, and that while these changes are not detected by evaluation datasets, at least one third of the nearest 25 neighbors of a word are different from one model to another.

We saw that the variation is not affecting all words of the semantic space equally and we found features which help identify some areas of (in)stability in the semantic space. Words having a low and high frequency range have a tendency to display more variation. Words in the medium frequency range show more stability. By observing words that vary the most and the least in two models, we were able to identify regularities and found that numerals and nationalities were less likely to be impacted by variation than polysemic nouns.

While the variation we observed with the DEFAULT model was fairly high, it is difficult to draw conclusions about neighbors variation if we ignore the inherent instability of word2vec that is a result of several random processes (initialization of the vectors, negative sampling etc.). To be able to properly understand parameters variation, it is necessary to understand this inherent variation. As a consequence, we decided to investigate this phenomenon. We present our method and results in the next chapter.

Chapter 5

Internal instability - a poorly known phenomenon

Contents

5.1 Introduction

In the previous chapter, we investigated the impact of different hyperparameters on word embeddings by comparing nearest neighbors across models. We observed an important variation between models with a minimum mean variation of 0.3. However, to be able to interpret this variation, it is important to take into account the instability that is inherent to word embeddings. Because it is a neural-based method, word2vec implies random processes when training and models trained with the same hyperparameters are different across runs. As a consequence, how can we be sure that the variation observed in chapter 4 is not a result of the inherent instability of the system?

In this chapter, we propose to investigate the internal stability caused by random processes in word2vec. We first explain what causes internal instability. Then, we propose to measure the amplitude of this phenomenon for word embeddings trained using word2vec by measuring nearest neighbors variation for models trained with the same hyperparameters. The different experiments we made are conducted on different corpora to ensure the phenomena observed are not dependent on the corpus.

5.2 Word embeddings instability

5.2.1 What is instability?

The question of reproducibility, reliability and stability has been recently brought up in the machine learning community and has been more and more of a concern. It is sometimes referred to as a "reproducibility crisis" and workshops have been specifically dedicated to this phenomenon. This crisis is due to the lack of sharing of resources or code used when training models but it also comes from the lack of clarity about the conditions in which experiments have been conducted (Hutson, 2018). Neural-based word embeddings methods imply random processes that introduce inherent instability even if models are trained using the same hyperparameters. In the case of word2vec, the instability is a result of several processes such as the random initialization of vectors that happens prior training, the use of negative examples for training (negative sampling) and the removal of highly frequent words (subsampling)[1]. Because of this, training word embeddings with the same hyperparameters yields different vectors. The initialization of vectors is done differently depending on the version of word2vec used for training. In the original C implementation developed by Mikolov et al. (2013c) the pseudo-random number generator used to initialize vectors depends on the multi-threading option. As a consequence, it is possible to force word2vec to be deterministic by setting the number of threads to 1. Word2vec was also implemented as a Python library named `gensim` (Rehurek and Sojka, 2010). In this library, the training process can also be forced to be deterministic by setting the seed for the random number generator. It is also required to set the thread option to 1 to avoid randomness caused by arbitrary thread scheduling[2].

Training without setting the seed may have a direct impact on the nearest neighbors of words because vectors vary across runs. Table 5.1 displays the 5 nearest neighbors for the words *paper* and *white* for 3 models trained on the BNC using the same hyperparameters (SG with negative sampling, negative sampling

[1] See chapter 1.

[2] `https://radimrehurek.com/gensim/models/word2vec.html`

rate of 5, vectors of 100 dimensions, subsampling rate set to 10^{-3}, 5 iterations and mincount set to 100). We see that the rank of some neighbors change across models (e.g. *magazine*, *pamphlet* and *book* for *paper* and *yellow* and *blue* for *white*). Other words disappear from the top 5 nearest neighbor. E.g. for the target word *paper*, the neighbor *newspaper* only appears in one model. Similarly, for the word *white*, the neighbor *pink* only appears once.

Target word	paper (n)			white (adj)		
Rank	**Model 1**	**Model 2**	**Model 3**	**Model 1**	**Model 2**	**Model 3**
1	magazine	book	book	black	black	black
2	book	magazine	sheaf	grey	grey	yellow
3	pamphlet	newspaper	magazine	blue	yellow	grey
4	journal	pamphlet	folder	yellow	red	blue
5	article	folder	parchment	pink	blue	red

Table 5.1: Example of nearest neighbors variation for the words *paper* and *white* in models trained using word2vec with the same hyperparameters on the BNC corpus.

Although it is possible to control for instability by setting the seed, doing so seems artificial and defeats the purposes of the algorithm (Hellrich and Hahn, 2016). It is important to note here that this instability is not unique to neural-based models. Some techniques used to reduce the number of dimensions in count-based models, such as random indexing where the word representations are randomly initialized (Sahlgren, 2005), also suffer from these effects. However, in all these methods, algorithms are designed to converge towards the same representation and the effects of random processes are expected to be minimal.

When using word embeddings as a component of a complex deep learning system, not paying too much attention to instability might not necessarily be a problem since a number of random processes are already involved. Word embeddings only constitute an additional source of randomness. However, when word embeddings are used to explore different phenomena with a qualitative point of view, it is important to be aware of the inherent instability and account for it.

Word embeddings have rapidly become popular to qualitatively explore data in various domains such as digital humanities and corpus linguistics. E.g. Kim et al. (2014) used word2vec embeddings to automatically detect changes in word usage. We also saw in chapter 3 that Kulkarni et al. (2015) used word embeddings to detect and track linguistic shifts. Hamilton et al. (2016) compared words meanings and their evolution by aligning word embeddings trained on corpora

representing different eras. Bolukbasi et al. (2016) focused on the study of bias in word embeddings. However, these studies disregard instability.

A few studies have specifically focused on the instability problem. Hellrich and Hahn (2016) investigated the reliability[3] of word embeddings in the context of diachronic linguistic, i.e. they assessed if word embeddings can be trusted to conduct analyses in diachronic linguistics. They identified different factors influencing neighborhood reliability. Some factors are directly related to the algorithm, e.g. the number of epochs used for training. Other factors are related to the corpus used for training, e.g. word frequency, or to the intrinsic features of a word, e.g. its degree of polysemy. They also showed that the identified factors vary depending on the corpus used for training. As a consequence, it is important to consider the corpus used to train word embeddings and more data is not necessarily better. The final conclusion of this work is that word embeddings are not reliable with top neighbors varying a lot over different training. As a consequence the authors advise against using word embeddings for digital humanities. Antoniak and Mimno (2018) also investigated the different factors influencing neighbors instability but they chose to focus on models trained on small corpora. Similarly to Hellrich and Hahn (2016) they insist on the importance of acknowledging neighbors instability as ignoring it could completely invalidate the presented results. They also found that the ranks of some nearest neighbors change across models and sometimes some words even disappear from the nearest neighbors, similarly to what we illustrated with table 5.1. They concluded that word embeddings "are not even a single objective view of a corpus, much less an objective view of language". As such to be able to use word embeddings for qualitative analyses, they advise to present results taking into account the instability observed. While this practice is common in deep learning, it might be more complex to apply for qualitative studies. Should a neighbor be considered valid only if it appears across all models?

Rather than rejecting completely word embeddings or trying to fix the instability problem, we propose to embrace the instability phenomenon and use it to better understand word embeddings.

5.2.2 A note on terminology

Different words are used to describe the internal instability phenomenon that exists with word embeddings: reliability, reproducibility and stability. While similar, these three words refer to different aspects of the phenomenon. We propose to explain them here.

Galliers and Sparck Jones (1993) define **reliability** when evaluating NLP systems as follows:

[3]See section 5.2.2.

> Reliability refers to the extent to which variation in measures is attributable to the nature of measurements themselves as opposed to the nature of the phenomena being measured: a reliable measure can be depended on to give consistent and stable results.
> Galliers and Sparck Jones (1993, 50)

As a consequence, reliability really encompasses the ability of models to be trusted. Moreover reliability is associated with random errors and a model that suffers from a lot of random errors cannot be trusted (Galliers and Sparck Jones, 1993). This is exactly how Hellrich and Hahn (2016) used reliability, as a measure assessing if word embeddings can be used or not.

Reproducibility, is an important aspect of scientific research (Jimenez et al., 2017). Results have to be reproducible by someone else and by applying the same procedure on the same data, results are expected to be the same.

Stability refers to the idea that neighbors stay the same from one model to another without any judgment on the performance of the model (such as the investigation done by (Antoniak and Mimno, 2018)).

In this thesis, we propose to embrace the instability problem. We are not trying to improve word embeddings performances and we are not using them to draw qualitative conclusions in a specific context. We consider that the variation in nearest neighbors across models trained with the same hyperparameters is a phenomenon worth to be investigated. We aim at understanding better word embeddings by comparing nearest neighbors across models. As a consequence, we mainly use the words (in)stability and variation to describe the observed changes of nearest neighbors across models.

In the following section we present the amplitude of the stability phenomenon.

5.3 Amplitude of the instability phenomenon

In section 5.2.1 we saw that the nearest neighbors of the words *paper* and *white* vary across two models trained with the same hyperparameters. Some neighbors disappeared from the top nearest neighbors while other neighbors' ranks changed. We also saw that is it possible to control for instability by setting the seed to a fixed number and train using only one thread. However, this process seems artificial and would raise other questions (should the choice of the seed be arbitrary?). In chapter 4 we observed different variations scores for words across models. Words are differently impacted by variation with some words having a low variation score (e.g. ordinals and nationalities) and some words having a high variation score (e.g. polysemous nouns and generic adjectives). As a consequence we wondered if a similar phenomenon could be observed for models trained with the same hyperparameters. If we are able to identify words that are always impacted in the

same way, we might be able to gather useful information about nearest neighbor variation and that might help us work better with word embeddings. We might be able to identify some types of words for which they perform well and other types of words for which we should avoid to use word embeddings. This means that we will be partly able to control this instability phenomenon in a way that will not hurt the linguist's satisfaction. Because we wanted to know if the observations made depend on the corpus used for training, we decided to experiment with three different corpora. We present the corpora used and the trained models in the following section.

5.3.1 Models

The models used for the following experiments were trained on 3 different corpora. We already presented two of them in chapter 4: ACL, a specialized corpus made of NLP scientific papers, and the BNC (British National Corpus). The third corpus we selected is PLOS, a specialized corpus we created using biology scientific papers from *All of PLOS*[4]. The size of PLOS is similar to ACL and the BNC with about 100 million words. Similarly to what we presented in chapter 4, we lemmatized and POS-tagged the corpora using Talismane (Urieli, 2013).

Our experiments focus on models trained with the same hyperparameters. We decided to train 5 models per corpus to get a good overview of the variation phenomenon. This allows us to compare 10 pairs of models per corpus. We decided to use the same hyperparameters used for the DEFAULT model in chapter 4. As a consequence, for each corpus we trained 5 models using the original word2vec C code without deactivating multi-threading and with the following hyperparameters:

- Skip-Gram with negative sampling,
- negative sampling rate of 5,
- window size of 5,
- vectors of 100 dimensions,
- subsampling rate set to 10^{-3},
- number of iterations set to 5,
- mincount set to 100.

5.3.2 Quantitative evaluation

In chapter 4, before measuring the variation of nearest neighbors across models, we first ran a quantitative evaluation. We decided to do the same here, to see if any change between models was detected by intrinsic evaluation datasets. We

[4] `https://www.plos.org/text-and-data-mining`

checked the performance of our models on WordSim-353, SimLex-999 and MEN. The minimum and maximum performance scores for each corpus are reported in table 5.2. Since ACL and PLOS are specialized corpora, it is not surprising that their performance scores on all 3 datasets are lower than the BNC. However, we note that ACL performs better on WordSim-353 than PLOS but the contrary is observed on SimLex-999 and MEN. Similarly to what we observed in 4, models trained on the BNC perform better on WordSim-353 and MEN than on SimLex-999. Most importantly, the differences in variation are very low for all 3 corpora on the 2 datasets with minimum and maximum values that are very similar.

Corpus	**WS353 (min–max)**	**Simlex-999 (min–max)**	**MEN (min–max)**
ACL	0.592 – 0.601	0.192 – 0.201	0.505 – 0.510
BNC	0.631 – 0.639	0.306 – 0.312	0.727 – 0.730
PLOS	0.392 – 0.403	0.273 – 0.279	0.573 – 0.575

Table 5.2: Minimum and maximum performance scores on WordSim-353 and SimLex-999 for the 5 models trained for each corpus.

We observe that the differences in performances between models are relatively low. This means that the inherent instability of word embeddings is not captured by intrinsic datasets and this could be the reason explaining that it is often disregarded. In the next section we measure variation across models by comparing nearest neighbors.

5.3.3 Measuring variation

As we discussed in chapter 4, nearest neighbors are a good way to get immediate feedback on what changes across models. Moreover, they allowed us to discover that not all words are similarly impacted by variation, with the nearest neighbors of some words varying more than others. As a consequence, we decided to use the same measure of nearest neighbor variation presented in section 4.3.2.2. We compare the 25 nearest neighbors of nouns, proper nouns, adjectives, adverbs and verbs across the 10 pairs of models, i.e. we computed 10 variation scores for each word per corpus.

Table 5.3 displays the number of words evaluated for each model as well as the mean variation score for each corpus. The mean score was computed across the 10 comparisons and across all words in the model. We also indicated the mean standard deviation for models and the mean standard deviation for words.

We computed paired t-tests for pairs of comparisons, i.e. BNC1_BNC2 vs. BNC1_BNC3, BNC1_BNC2 vs. BNC1_BNC4, BNC2_BNC3 vs. BNC2_BNC4

etc. such as both comparisons had one model in common. This resulted in 20 paired t-tests per corpus. We found that the differences were generally significant[5] with $p < 0.05$ meaning that models are different.

Corpus	**Vocabulary**	**Mean variation**	**Std dev. (models)**	**Std dev. (words)**
ACL	22292	0.16	0.04	0.08
BNC	27434	0.17	0.04	0.07
PLOS	31529	0.18	0.05	0.07

Table 5.3: Size of evaluated vocabulary, mean variation score and standard deviations for models trained using the same parameters on ACL, BNC and PLOS.

In table 5.3, we see that the mean variation score is similar for all three corpora meaning that the instability phenomenon does not seem to be dependent on the corpus used for training. The average variation score is about 0.17. As a consequence, when looking at the 25 nearest neighbors of a word, 4 or 5 of these 25 nearest neighbors change from one model to another. If we only relied on evaluating models using intrinsic datasets, we would not have noticed the importance of this variation (as seen in section 5.3.2).

We also notice that the variation score is stable across pairs of models with a standard deviation of about 0.04 for the 3 corpora. This means that from one pair of models to another, the variation measured for a given word is very similar. However we observe a slightly higher standard deviation between words, with a score of about 0.07 for the 3 corpora. This means that variation across words is more heterogeneous with some words varying more than others.

Before looking into the differences of variation between words, it seems important to go back to the experiments we conducted with models trained with different hyperparameters. Since we used the BNC as the main corpus in that experiment, we reported the mean variation score of the BNC on figure 5.1. While this score is lower than the mean variation scores of models trained with different hyperparameters, considering the standard deviation score of 0.08 across words, some words have the same variation score in models that are trained using the same hyperparameters and models trained using different hyperparameters. E.g. in the WIN6 model, words displaying the lowest variation have a score similar to words displaying a high variation score in the models trained with the same hyperparameters. As a consequence it seems highly important to understand internal variation before being able to investigate variation related to change of hyperparameters.

[5]Except for ACL1_ACL4 vs. ACL1_ACL5 ($p = 0.13$), ACL2_ACL3 vs. ACL1_ACL2 ($p = 0.57$) and PLOS2_PLOS4 vs. PLOS2_PLOS5 ($p = 0.19$).

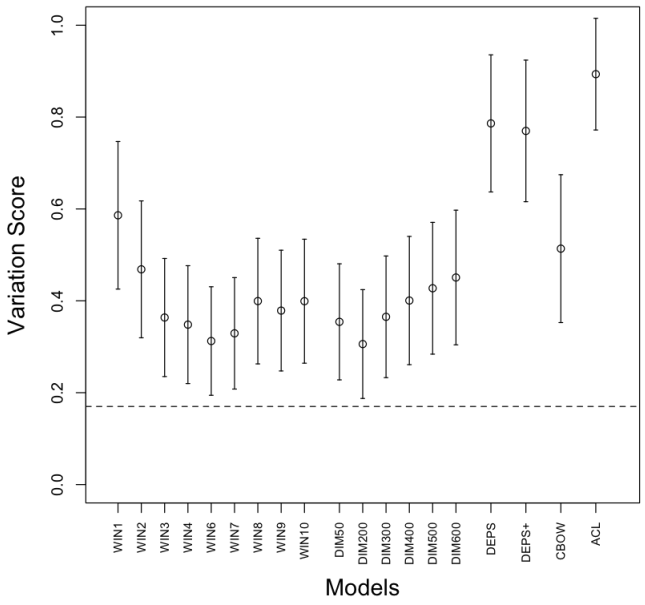


Figure 5.1: Variation score for models trained with different parameters. We added the mean variation score for 5 models trained using the same hyperparameters. The dotted line indicates the mean variation score for 5 models trained using the same hyperparameters as the DEFAULT model.

Before exploring the variation of words for models trained with the same hyperparameters in details, we decided to examine words varying the most and the least in those models in the 3 corpora.

Table 5.4 displays the first 35 words with the lowest mean variation score across the 5 models trained using the same hyperparameters for the BNC corpus. First, we notice some tagging errors such as *finance* which should be tagged as a noun. Then, we observe some words with a variation score of 0 (*immense*, *70s*, textitrumble etc.), meaning that their nearest neighbors remained the same across the different models. However, it is important to remember here that the way we compare neighbors across models does not include the ranks of neighbors. As a consequence, while nearest neighbors remain the same, their ranks might have changed. We also observe words that remained stable across models trained with different hyperparameters in chapter 4 such as ordinals (e.g. *sixteenth*, *25th*) and family members (e.g. *wife*, *daughter* and *grandmother*). It thus seems that these words are not sensitive to the different hyperparameters used when training nor to the inherent instability in word embeddings. We observed similar behaviors

for models trained on ACL and PLOS[6]. For ACL, words that vary the least are ordinals (e.g. *51st*, *50th* etc.) and words that are specific to the NLP domain (e.g. *precision*, *recall*, *n-grams* etc.). For PLOS, words that vary the least are also ordinals (e.g. *8th*, *ninth* etc.), conjunctive adverbs (e.g. *furthermore*, *firstly*, *moreover*) and words that described molecules (e.g. *polyacrylamide*, *hygromycin* etc.).

Table 5.5 displays the 35 words with the highest mean variation score across the 5 models trained for the BNC corpus. The variation scores are quite high reaching at least 0.4. This means that almost half of their nearest neighbors are different from one model to another. Among words that vary the most we observe a lot of proper nouns (e.g. *Hatton*, *Jacobsen* etc.) similarly to what we observed in chapter 4. We also observe generic verbs such as *make* that can appear in a variety of contexts. For ACL and PLOS[7] we also observe a lot of proper nouns among words that vary the most that seem to correspond to acronyms (e.g. *cd*, *cbs* for ACL and *hsr* and *pcb* for PLOS). It is thus clear that not all words are impacted similarly by variation and that the observed variation is a phenomenon related to the words (or at least to their use in a given corpus) and not to the models.

5.4 What to do next?

In this chapter we presented the instability phenomenon that impacts word embeddings trained with word2vec. This instability is a result of random processes inherent to neural-based methods. We decided to investigate this instability by training 5 models with the same hyperparameters for 3 different corpora. We saw that the variation was not captured by intrinsic datasets, with very similar performance scores across models. By observing the 35 words varying the most and the least in all 3 corpora we were able to identify regularities (e.g. ordinals tend to vary less). We also found that words varying the most and the least were similar to those observed when training models with different hyperparameters (ordinals, family nouns, polysemic nouns). As a consequence we want to explore in details the variation across models. We propose to do so in the next chapter where we investigate variation through different linguistic features.

[6]See Appendix A for ACL and Appendix B for PLOS
[7]See Appendix C for ACL and appendix D for PLOS

Word	**Mean var. across models**
jj#immense	0
nn#70s	0
nn#rumble	0
nn#vice-president	0
nnp#w.l.r.	0
vb#analyse	0
jj#1/4	0.016
jj#100g	0.016
jj#sixteenth	0.016
jj#windy	0.016
jj#worried	0.016
nn#melody	0.016
nn#sitting-room	0.016
nn#unemployment	0.016
nn#whisky	0.016
nn#writer	0.016
nn#1890s	0.016
nn#kitchen	0.016
nn#magma	0.016
nn#pm	0.016
nn#scotch	0.016
nn#seventy	0.016
nn#wife	0.016
nnp#£	0.016
nnp#a.c.	0.016
nnp#finance	0.016
nnp#m.r.	0.016
jj#tory	0.024
jj#few	0.024
nn#1960s	0.024
nn#cottage	0.024
nn#daughter	0.024
nn#grandmother	0.024
nn#television	0.024
nnp#25th	0.024

Table 5.4: Words displaying the lowest mean variation score for 5 models trained with the same hyperparameters on the BNC corpus.

Word	**Mean var. across models**
nnp#hvk	0.772
nnp#jacobsen	0.532
nn#rink	0.528
nnp#hatton	0.512
nnp#boyd	0.5
nnp#harlequin	0.492
nn#reckoning	0.48
nnp#page	0.476
nnp#hook	0.472
nnp#wolff	0.472
nnp#bean	0.468
nnp#met	0.468
nnp#bart	0.468
nnp#priestley	0.464
nnp#banks	0.464
jj#finishing	0.46
nn#natural	0.46
nnp#vince	0.46
nn#con	0.456
nnp#moira	0.456
nnp#lewis	0.452
nnp#stapleton	0.452
nnp#beattie	0.448
nnp#sandison	0.448
nnp#mallender	0.448
nnp#cochrane	0.448
vb#make	0.448
nnp#wilcox	0.444
nnp#horne	0.444
nnp#shepherd	0.444
nnp#draper	0.44
nnp#mowbray	0.44
nnp#sec	0.44
nnp#burgess	0.44
nnp#clifford	0.44

Table 5.5: Words displaying the highest mean variation for 5 models trained using the same hyperparameters on the BNC corpus.

Chapter 6

Neighbors instability as a linguistic phenomenon

Contents

6.1 Introduction

In the previous chapter we presented the instability phenomenon and its amplitude by experimenting with 5 models trained with the same hyperparameters. We used corpora of different genres to investigate if the observed effects were caused by the corpus used for training. We found that words are impacted differently by instability with some words displaying more instability in their nearest neighbors. We were able to identify some regularities for words that remained stable across trainings. In this chapter we propose to investigate several factors that could explain why some words are more impacted by variation than others. We selected internal and external factors. Internal factors corresponds to features directly related to the model and the data used for training, such as words frequency. External factors are directly related to a word, such as its POS, its degree of polysemy or its degree of concreteness.

6.2 Factors explaining variation

In chapter 4, section 4.4.3, we used frequency and POS to explore the variation of words when training models using different hyperparameters. As a consequence, before exploring any other factors, we wanted to know if the observed effects were similar for models trained using the same hyperparameters.

6.2.1 Frequency

When training models using different hyperparameters, we observed a non-linear effect of frequency on nearest neighbors' variation. Words in the mid-frequency range were more stable than words in the low and high frequency range. We wanted to know if the same effect was observed for models trained with the same hyperparameters. As a consequence, we investigated the relation between frequency and nearest neighbors variation. For each corpus, we compared the mean variation score (that was computed across 5 models trained with the same hyperparameters presented in section 5.3.1) to the frequency of words.

Figure 6.1 displays this relation for all 3 corpora. The frequency is indicated using its log (base 10). We observe that for words in the low and mid-frequency range, variation decreases with frequency. For words in the high frequency range,

we observe different behaviors depending on the corpus. For ACL, variation decreases with frequency even for high frequency words (e.g. *model* occurs 285127 time and has a variation score of 0.1). For PLOS, high frequency words display more variation (e.g. *cell* appears 629937 times and has a mean variation score of 0.436). Because of the difference of behavior, we decided to look into the nearest neighbors of *model* in ACL and *cell* in PLOS. The nearest neighbors of models all refer to types of models and measures (e.g. *log-linear*, *rnn*, *bigram* etc.) and the neighbors that change across models are accurate. For *cell* nearest neighbors mostly refer to types of cells (e.g. *d4+cd25+*, *HSCs* etc.). However, despite the high variation score of *cell*, its nearest neighbors are accurate across models. Finally for the BNC, the variation score between mid-frequency and high frequency range words is very similar. In any case it is recommended to interpret carefully high-frequency words since there are fewer high-frequency words in all 3 corpora and it might be more difficult to identify patterns to explain their (in)stability.

Globally, we observe a similar effect than for models trained with different hyperparameters. Words in the mid-frequency range (1000 to 10000 occurrences) are the most stable. Low frequency words tend to vary the most independently of the corpus.

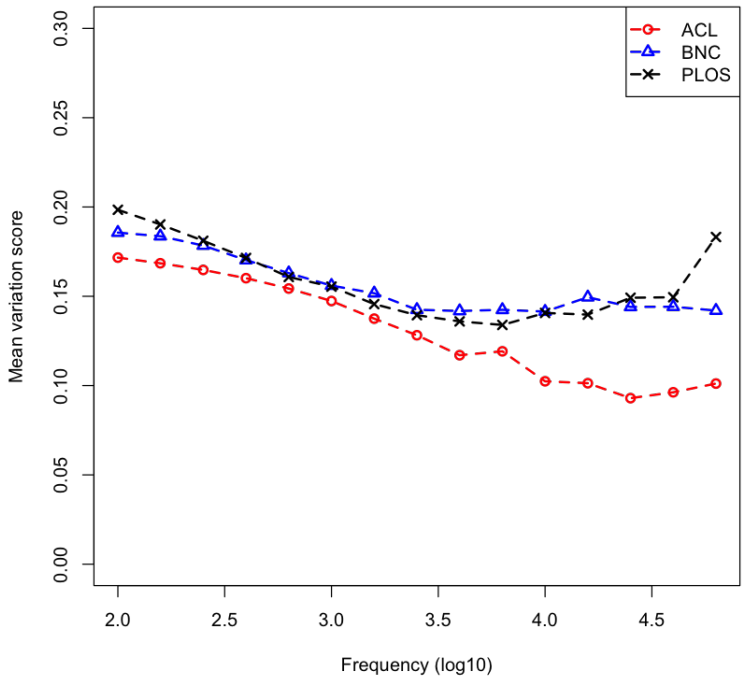


Figure 6.1: Relation between variation score and frequency for ACL, BNC and PLOS.

6.2.2 POS

When training models using different parameters we saw that variation was affecting all POS similarly except for proper nouns that were displaying a slightly higher variation score. Because we are experimenting with 3 different corpora, we decided to examine the effect of POS when training models using the same parameters. Figures 6.2, 6.3 and 6.4 show the repartition of variation per POS in ACL, the BNC and PLOS. We observe similar patterns for all 3 corpora. The mean variation score is similar across POS except for proper nouns. This is actually in line with the first observations made in section 5.3.3 where we looked at words varying the most and noticed a considerable amount of proper nouns (names and acronyms). In all 3 corpora we also observe quite a few outliers that are highly impacted by variation. They correspond to low frequency words in ACL and PLOS, e.g. the verb *gloss* and the noun *umbrella* in ACL, the adjective *inclusive* and the verb *browse* in PLOS. In the BNC these outliers are both low frequency words (e.g. the

adverb *narrowly*) and high frequency words (the verbs *make* and *might*). Some of the outliers also correspond to tagging errors in all 3 corpora.

Examining the influence of frequency and POS provided interesting insights about the way words are impacted by instability. Frequency has an impact on the instability of words but it does not explain it all. Moreover, while proper nouns have a tendency to vary more, POS by itself does not allow to completely understand why some words vary more than others.

Similarly to what we have done before, it seems crucial to explore words that display the least and most variation across models for all corpora. We propose to do so in the next section.

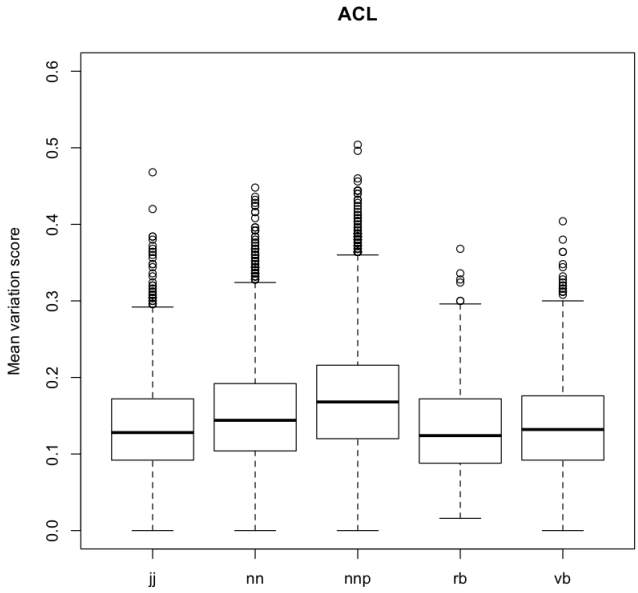


Figure 6.2: Mean variation score per POS for ACL.

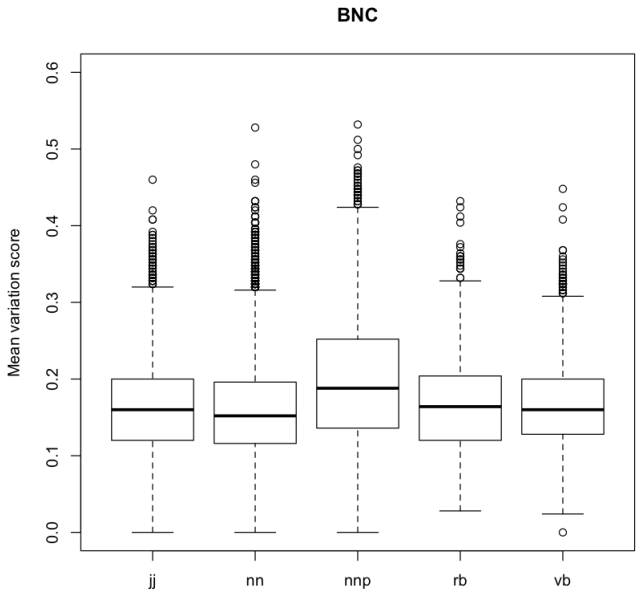


Figure 6.3: Mean variation score per POS for the BNC.

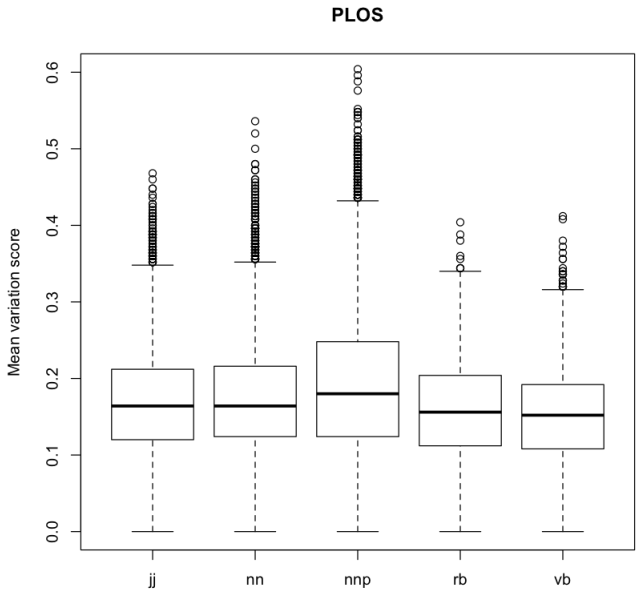


Figure 6.4: Mean variation score per POS for PLOS.

6.3 Exploring stable and unstable words

In chapter 5, we already took a closer look at words varying the least and the most in all 3 corpora. We observed some words with a variation score of 0 across runs. We also were able to roughly identify some clusters of words for all 3 corpora: family nouns in the BNC, words that are specific to the NLP domain for ACL and conjunctive adverbs for PLOS. We also managed to identify regularities for words varying the most in all 3 corpora with proper nouns being more likely to vary in all 3 corpora. This was even confirmed in the previous section when we examined the relation between variation and POS. As a consequence, we wondered if it could be possible to identify semantic classes that remain stable across runs. We present this experiment in the following sections.

6.3.1 Stability of nearest neighbor

Before trying to identify clusters of words that remain stable, we were curious about the clusters we observed. We wanted to first investigate if the cosine score of the first nearest neighbor could be an indicator of stability, with the idea that a high cosine score would guarantee a stable nearest neighbor. To check this, we computed the correlation between the cosine score of the nearest neighbor of each word and the variation score. We found a partial correlation of about -0.4. This means that the cosine score of the nearest neighbor of a word partly explains the stability of the word.

We also decided to investigate the stability of nearest neighbors for stable words. For example, *grandmother*, *daughter* and *wife* were identified as stable words. We decided to look at their nearest neighbors. In the BNC, the nearest neighbors of *grandmother* are also family members (e.g. *aunt*, *mother*, *sister*, *uncle* etc.). We observed the same phenomenon for *daughter* with nearest neighbors like *sister*, *son* and *niece*. It thus seems that family members have other family members as nearest neighbors, and they correspond to family members that we were able to identify as stable words.

We tested this hypothesis on all our data for the BNC. For each word with a computed variation score (nouns, proper nouns, adjectives, adverbs and verbs), we retrieved the union of the 25 nearest neighbors across models. Then we computed the mean variation score for the union of the nearest neighbors. We computed the correlation between the mean variation score of the neighbors and the variation score of the word. We found that the variation score of stable words is correlated with the variation score of its nearest neighbors, with a correlation of 0.5 on average. This means that we can explain the stability of some words because of the stability of their neighbors. Some words remain stable across runs because they belong to stable zones in the semantic space (e.g. family nouns). As a consequence

Word	NN1	Rank1	Rank2	NN2	Rank1	Rank2
nn#head	nn#shoulder	1	1	nn#shoulder	1	1
nn#head	nn#ear	2	2	nn#ear	2	2
nn#head	vb#bend	3	3	vb#bend	3	3
nn#head	nn#nose	4	6	nn#arm	6	4

Table 6.1: Nearest neighbors of the target word *head* along with their ranks in 2 models trained on the BNC using the same hyperparameters.

it means that we should be able to identify these stable semantic zones.

During this PhD, we also co-advised the research's work of a master's student. For this work we selected a subset of nouns, adjectives and verbs belonging to different frequency ranges (low, mid and high) in the BNC. For each word, we retrieved its 25 nearest neighbors in two models trained with the same hyperparameters on the BNC (SG with negative sampling, negative sampling rate of 5, vectors of 100 dimensions, subsampling rate set to 10^{-3}, 5 iterations and mincount set to 100). Table 6.1 gives an example of a word with its nearest neighbors in two models (NN1 and NN2). The rank of each neighbor is indicated in both models (Rank1 and Rank2). Using this data, for each neighbor Bravo Candel (2019) computed the variation of the rank. He also identified semantic relations between the target word and each neighbor. His analyses showed that nearest neighbors that are hypernyms, hyponyms or synonyms of the target word significantly remain stable across the 2 models. However, neighbors that are only related to the target word (or when associating a semantic relation was more challenging) are mostly unstable. As a consequence, we can consider that the stability is an indicator of the good quality of the semantic relation shared by a target word and its neighbor.

6.3.2 Identifying semantic stable zones

We saw that some words are less sensitive to variation across runs. To identify semantic zones of stable words, we decided to cluster words using their semantic similarity computed with cosine score. E.g. in the BNC, we selected the 300 words with the lowest mean variation scores across the 10 pairs of models. Then, we computed the cosine similarity between all those words for each of the five models, so that for two given words we have the cosine similarity in the first model, the second model etc. Using these cosine similarities, we performed hierarchical agglomerative clustering using the `R`[1] function `hclust`. We used the Ward's agglomeration method and arbitrarily set the number of clusters to 10. We performed clustering

[1] R Core Team (2017).

for the 5 models trained. As a consequence we ended up with 5 different clusterings, one for each model, each containing 10 clusters. We wanted to make sure that the identified clusters remained the same across models. To do so, we evaluated the agreement between clusters using Rand Index (Rand, 1971) which is defined as follows:

$$R(Y, Y') = \frac{\sum_{i<j}^{N} \gamma(ij)}{\binom{N}{2}} \tag{6.1}$$

Y and Y' are two clusterings the agreement is computed for. N is the number of data points in the clusters, so $\binom{N}{2}$ is the number of different pairs of words. For each possible combination of pairs of points, $\gamma(ij)$ is 1 if the two points appear in the same cluster in Y and Y' or if the two points belong to different clusters in Y and Y'. Otherwise, $\gamma(ij)$ is 0. As such the Rand Index is the ratio of word pairs that are treated the same way in both clusterings. We measured the Rand index for all 10 pairs of models and obtained an average value of 0.93 ($\pm$ 0.01, IC 95%). The same values were obtained for the other two corpora. As a consequence, the clusters identified are very similar across models.

Figure 6.5 shows the 10 identified clusters for models trained on ACL[2]. First, we notice that the clusters are of various sizes. Then, we are able to manually identify the following semantic clusters:

- foreign words (e.g. *cada*, *politecnica*, *habe* etc.),
- ordinals (*40th*, *52nd* etc.),
- evaluation measures and processes in NLP (e.g. *crossvalidation*, *non-expert*, *reliability*, *precision*, *f-score* etc.),
- conjunctive adverbs (e.g. *nevertheless*, etc.),
- verbs describing scientific processes (e.g. *observe*, *describe* etc.),
- words referring to pre-processing in NLP (e.g. *tokenization*, *n-grams*, *post-processing* etc.),
- internal references to figures and tables (e.g. *6b*, *7a* etc.).

We were not able to identify a semantic class for all clusters, some being constituted of a variety of proper nouns that are mostly tagging or tokenization errors (e.g. *r.a*, *1995a*, *g.* etc.).

For the BNC, among the 10 clusters we were able to identify the following classes:

- cooking measurements (e.g. *4oz*, *100g*, *tbsp* etc.),

[2]Visualization was done on a randomly selected ACL model. We used `https://projector.tensorflow.org` to project points with a PCA on the vectors of the 300 most stable words.

- family members (e.g. *uncle*, *son*, *spouse*, *nephew* etc.),
- countries (e.g. *sweden*, *france*, *poland* etc.),
- rooms and objects of the house (e.g. *bathroom*, *bedroom*, *furniture* etc.),
- references to date and time (e.g. *noon*, *april*, *pm* etc.),
- ordinals (e.g. *sixteenth*, *fifteenth* etc.),
- words referring to sadness and pain (e.g. *regret*, *rueful*, *misgiving*, *grief* etc.).

Finally among the 10 detected clusters for PLOS, we were able to identify the following semantic clusters:

- enzymes (e.g. *saci*, *bglii*, *clai* etc.),
- internal references to figures (e.g. *7a*, *1b*, *figs*, *table* etc.),
- dates (e.g. *october*, *december* etc.),
- serial dilution[3] (e.g. *2-fold*, *ten-fold* etc.),
- conjunctive adverbs (e.g. *thirdly*, *moreover* etc.),
- scientific processes (e.g. *implicate*, *reason* etc.),
- injection type (e.g. *intranasal*, *intraperitoneally* etc.),
- antibiotics (e.g. *puromycin*, *colistin*, *blacsticidin* etc.).

As we noticed earlier, the identified clusters are of different sizes ranging from a few elements (e.g. performance measures) to several dozens (e.g. ordinals). This means that the identified clusters are not a result of the bias of the number of nearest neighbors selected.

By looking at the identified clusters across the 3 corpora, we are able to classify clusters into broader semantic classes. For example for both specialized corpora we identified conjunctive adverbs and scientific processes that correspond to transdisciplinary scientific lexicon (Tutin, 2007). Table 6.2 lists the different semantic classes of clusters we analyzed. In addition to the transdisciplinary scientific lexicon, we were able to identify broader semantic clusters for all 3 corpora:

- specific localized contexts: foreign words used in examples in ACL, internal references in PLOS, measures used in recipes in the BNC;
- closed classes of co-hyponyms: performance measures in ACL, antibiotics in PLOS and family members in the BNC.

Overall, these semantic classes correspond to words which appear with restricted, specific and regular contexts for which we expect distributional semantics to perform well.

[3]Dilution of a substance in solution (`https://en.wikipedia.org/wiki/Serial_dilution`).

Figure 6.5: Clusters identified for the 300 most stable words for models trained on ACL. Visualization done using `https://projector.tensorflow.org`.

Cluster type	**Corpus**	**Examples**
Specific localized contexts	ACL	foreign words used in examples (*para, com, sobre...*), (*der, das, nicht, die...*) ordinals (*12th, eleventh, 41st...*)
	PLOS	internal references (*figures, table, 6b, 1a...*) figures description (*dot, triangle, filled, orange...*)
	BNC	temporal expressions (*am, pm, 31st, noon...*) measures in recipes (*tsp, tbsp, oz...*)
Closed classes of co-hyponyms	ACL	performance measures (*precision, recall, f-score...*) linguistic pre-processing (*parsing, lemmatization, tokenizing...*)
	PLOS	antibiotics (*puromycin, blasticidin, cefotaxime...*) injection type (*intraperitoneally, intranasal, intramuscular...*)
	BNC	family members (*wife, grandmother, son, sister...*) rooms and objects of the house (*kitchen, sitting-room, bathroom, furniture...*)
Transdisciplinary scientific lexicon	ACL	conjunctive adverbs (*nevertheless, relatively, secondly, additionally...*) scientific processes (*discuss, describe, observe...*)
	PLOS	conjunctive adverbs (*moreover, furthermore, conversely...*) scientific processes (*hypothetize, reason, elucidate...*)

Table 6.2: Selected stable clusters identified for each corpus.

6.3.3 Investigating unstable words

We also decided to try to find common features that would help us group unstable words into classes. We already examined a sample of the most unstable words from the BNC in section 5.3.3. E.g. we saw that among words varying the most there were a lot of proper nouns. We decided to extend our observations by looking at the 300 most unstable words in all 3 corpora. Given the instability of these words,

it is not possible to identify regularities using clustering. As a consequence, we decided to directly observe the most unstable words, by considering words with the highest mean variation scores across models.

We reported the unstable classes we identified in table 6.3. As we did before, we noticed that some of the unstable words corresponded to POS-tagging mistakes (e.g. *smile* was tagged as a proper noun in PLOS, *course* was also tagged as a proper noun in ACL). We also observed that real proper nouns were highly impacted by variation. They correspond to names in the BNC (e.g. *Bart*, *Vince* etc.) and ACL (e.g. *Steve*, *Joyce* etc.) and acronyms in PLOS (e.g. *PCB*, *DMC* etc.). We also noticed an important number of words that correspond to central concepts in the specialized corpora (e.g. *gene*, *cell* and *protein* in PLOS, *language* and *sign* in ACL). These words vary across corpora and while very frequent, their semantic content is generic.

Corpus	**Series**	**Examples**
ACL	proper nouns	*Steve*, *Joyce*, *Ivan*...
	generic words	*language*, *sign*
	generic adjectives	*free*, *mix*, *special*
	polysemous words	*account*, *card*, *zone*, *sign*
PLOS	proper nouns	*PCB*, *DMC*, *TLP*, *ACD* ...
	generic words	*gene*, *cell*, *protein*
	generic adjectives	*free*, *current*, *near*, *double*
BNC	proper nouns	*Bart*, *Vince*, *Lewis*...
	generic adjectives	*whole*, *general*
	polysemous words	*make*, *close*, *cast*

Table 6.3: Selected examples of classes of unstable words for each corpus.

We also observed that generic adjectives (e.g. *whole*, *super* and *general* in the BNC, *free* and *current* in PLOS) and highly polysemous words (e.g *account* and *zone* in ACL, *make* and *close* in the BNC) are also challenging for the distributional semantics mechanism. This is because they appear in a variety of contexts but also because few words are similar to those words. Even though those words are highly impacted by variation, it does not mean that their neighbors are irrelevant. E.g. the adjective *exclusive* in the BNC has a mean variation score of 0.408. Some of its neighbors reflect the idea of privilege (*prestigious*, *complimentary* or *unique*) conveyed by this word. However, we also observe irrelevant neighbors that are more likely to change from one model to another (e.g. *Granada*, *Disney* etc.).

We were able to identify features possibly correlated to the stability and instability of some words. Using clustering on the most stable words across trainings,

we identified classes of words sharing semantic similarities. We also managed to identify linguistic regularities (polysemous words, proper nouns etc.). We now propose to further explore those features by using them to predict the stability of a word.

6.4 Predicting variation

We want to know if the stable and unstable words we identified in the previous section are associated with linguistic features that determine their stability. To do so, we decided to use a predictive model. In this way we can investigate the impact of several features and observe which ones are the most significant. Subsequently, the predictive model can be used to detect words that are more impacted by variation. In the following section, we present the features we decided to experiment with and the observations made.

6.4.1 Selected features

We selected three different types of features to predict the variation: features intrinsic to a word, to the corpus used for training and to the model. We present them in details below.

6.4.1.1 Features intrinsic to the word: POS and polysemy

Features that are intrinsic to a word represent inherent characteristics of words such as its POS and its degree of polysemy. We chose to select the POS as a feature because we observed in section 6.2.2 that proper nouns had a tendency to be more unstable. Regarding the degree of polysemy, we located highly polysemous words (e.g. *make* and *cast* in the BNC, *zone* and *sign* in ACL) that were highly impacted by variation.

We could argue that polysemy is actually a feature intrinsic to the corpus since the polysemy of a word is also dependent on the corpus and the meanings it has in the corpus. However, because we computed the degree of polysemy of words using an external resource, it does not necessarily reflect the usage made in the corpus.

To get the POS of a word, we simply retrieved the information from the tagger. To compute the degree of polysemy, we used ENGLAWI (Sajous and Hathout, 2015), an XML-encoded machine-readable dictionary extracted from the Wiktionary. It consists of 752770 articles and contains 108747 adjectives, 18062 adverbs, 283183 nouns, 46272 proper nouns and 36919 verbs[4]. We computed the degree of polysemy of a word by counting the number of entries it has in the

[4]It also contains entries for all other POS but we only selected the ones we are interested in.

resource. When a word did not exist in the resource, we made the decision to assign a degree of polysemy of value 1 to the word. We are conscious that the way we computed polysemy is not perfect and might be perceived as a bit simplistic. Because the Wiktionary is a crowd-sourced resource, it contains errors and we did notice some problems for polysemous words that had only one definition in the resource. However, this method was computationally inexpensive and was satisfying for our investigation.

When looking at the most unstable words, we noticed that many polysemous words were impacted by nearest neighbors variation. Moreover, because word embeddings encompasses several definitions in one vector, we expect polysemous words to display more variation.

6.4.1.2 Features relative to the corpus: frequency and entropy

Unlike the POS of a word and its degree of polysemy, features such as frequency and entropy are dependent on the corpus. In section 6.2.1, we were able to identify a relation between frequency and stability. We observed that mid-frequency words are more stable than low and high frequency words. Moreover, frequency is an essential feature in distributional semantics. Frequency is also part of the different hyperparameters that need to be set when training word embeddings with word2vec (we set the mincount threshold to 100 occurrences and high frequency words are subsampled). As a consequence, it was logical to choose frequency as one of the selected features.

We also noticed in section 6.3.3 that generic words specific to the corpus domain tend to be more impacted by variation (e.g. *gene*, *cell*, *language* etc.). We wondered if this was related to the fact that these words appear with a lot of different contexts making it difficult for the distributional method to detect regularities. As a consequence, we decided to investigate the role played by the variety of contexts a word appears in on its stability. To do so, we computed the entropy of a word with its contexts. This corresponds to the dispersion of the contexts of a word in the corpus. We computed the normalized entropy of a word as follows[5]:

$$H(x) = \frac{-\sum_{i=1}^{n} p_i log_2(p_i)}{log_2(n)} \tag{6.2}$$

where p_i is the probability of context i to appear with word x, i.e. the relative frequency of context i, and n is the number of context types. The denominator allows to normalize the entropy to be able to compare words with different numbers of occurrences and contexts. A higher value indicates a high variability in the contexts. We expect the entropy to be correlated to variation.

[5]Adapted from `https://en.wikipedia.org/wiki/Entropy_(information_theory)`

We computed the normalized entropy of a word by considering its collocates on a symmetrical window of size 5 for open classes of words only. However we did not filter the POS of contexts.

6.4.1.3 Features intrinsic to the model: L2-norm and nearest neighbor similarity

We considered two last features that are intrinsic to the model: the norm of the vector and the nearest neighbor cosine similarity score. We considered the L2-norm as a feature because Trost and Klakow (2017) showed that the L2-norm of common words do not follow the general distribution of models. We thus wondered if the L2-norm would be a good indicator of a word's stability.

The L2-norm is the square root of the sum of the squared vector values. It measures the distance of a vector from the arbitrary center of the vector space. Vectors with a high norm are separated from other vectors. When computing the similarity between two vectors, the norm is not taken into account since the cosine score only measure the angle between vectors. This would be completely different if the euclidean distance was used instead.

We also chose to investigate the impact of the nearest neighbor cosine score because we saw in section 6.3.1 that words having a nearest neighbors with a high cosine score have a tendency to be more stable. We expect this measure to be correlated to the stability of a word.

6.4.2 Models and results

We chose to study the impact of these 6 features on the variation score using multiple linear regression models. While we could have tried to use models that give the best prediction results, we are mostly interested in examining which features have the most impact on the stability of a word. As such, linear regression is the most straightforward to interpret results and understand the impact of each feature used for training.

We decided to check if the features used for training were suffering from multicollinearity effects. We computed the Variance Inflation Factor (VIF) using the `R` package `usdm`[6]. While the VIF scores for the entropy and frequency were higher than for other variables for all corpora, there were still under the value of 10 which would indicate a multicollinearity problem.

We used the word embeddings models presented in section 5.3.1. For each corpus, we trained 5 regression models, one per word embeddings model, using pairwise interaction of the features. This means that features were both used by

[6] `https://cran.r-project.org/web/packages/usdm/usdm.pdf`

themselves and combined. The 5 regression models all had the same target to be predicted, the mean variation score of a word that was computed across the 10 comparisons made for each corpus.

To evaluate the accuracy of our models, we used the *adjusted* R^2 value which measures the amount of variance of the target variable which is explained by the statistical model. The adjusted R^2 value is interesting because the R^2 value increases with added features, even though these features are not relevant. Table 6.4 displays the mean values and standard deviations across the 5 regression models for the 3 corpora. We were able to explain 39% of the variance for ACL, 43% for BNC and 48% for PLOS. The low standard deviations indicate that these scores are reliable. While far from an efficient prediction, these values prove that we nevertheless captured important features that can explain the stability of embeddings.

Corpus	**Mean adjusted R^2 (std. dev.)**
ACL	0.39 (0.0007)
BNC	0.43 (0.0102)
PLOS	0.48 (0.0006)

Table 6.4: Mean adjusted R^2 score for predicting the variation of a word on ACL, BNC and PLOS.

To understand the contribution of each feature on the variation prediction, we decided to use a feature ablation approach similar to the one used by Lapesa and Evert (2017) which consists in training several regression models by removing one feature at a time and compute the loss in R^2 from not using each feature. If the removed feature is an important predictor, the R^2 score will decrease. On the contrary, a superfluous feature will not impact the R^2 score. As a consequence, it is possible to classify features according to their importance. This method is simple to use and to interpret, especially since we are training models with pairwise interactions where the interpretation of coefficients could be challenging.

For each word embedding model, we trained one multiple linear regression model using all features and we then trained 6 other models by removing one feature at a time. We computed the loss of the adjusted R^2 for each of this model compared to models trained with all 6 features. The loss is seen as the relative importance of the ablated feature.

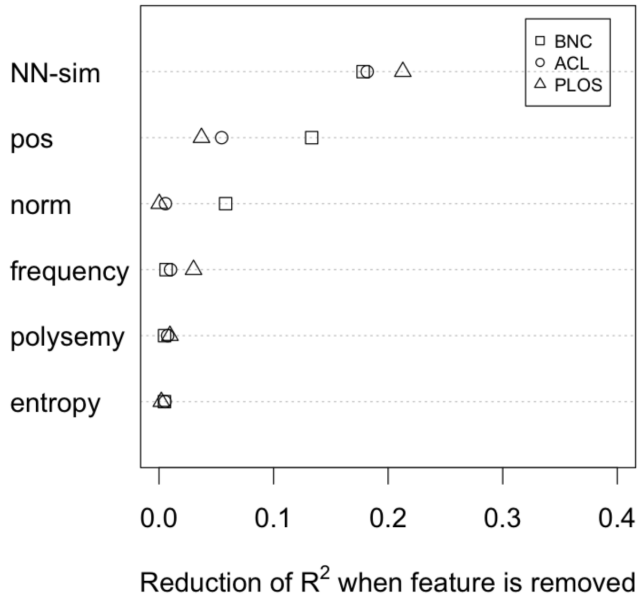


Figure 6.6: Feature ablation for multilinear regression models trained for ACL, BNC and PLOS.

Figure 6.6 display the impact of each feature. We observe a similar pattern for the 3 corpora. The most important feature is the cosine similarity score of the first nearest neighbor (*NN-sim*). It explains around 20% of the variance by itself. However, this feature does not explain all the variance. We expected this result since we had already observed the impact of the nearest neighbor cosine score on the variation score in section 6.3.1

The second most important feature is the POS (*pos*) of a word. This is a bit surprising since when we observed the impact of the POS on the variation score, we only noticed that proper nouns were more impacted by variation. We thus did not expect it to be the second most important feature.

The entropy of a word's contexts and the degree of polysemy are the features displaying the less (or even none) effect on all 3 corpora. Because in section 6.3.3 we observed that some of the most unstable words were highly polysemous, we expected the degree of polysemy to play an important role in predicting the variation. The lack of impact could be due to the resource we used or to the fact that we computed polysemy outside the corpus while it would be more accurate to compute it inside a given corpus. We were also surprised that entropy was not playing an important role in predicting the variation. Because we observed that generic words specific to the domain of the corpus (e.g. *protein*, *language* etc.) were higly impacted by variation, we expected words with a variety of contexts to be more sensitive to variation.

The norm and frequency features have different impact depending on the corpus. The norm helps to predict the variation for models trained on the BNC but not for models trained on PLOS and ACL. Frequency has slighty more importance for PLOS than for the BNC and ACL. While we saw before that mid-frequency words tend to be more stable, the lack of impact here can be explained by the fact that the frequency effect on variation is not linear as we saw in section 6.2.2.

To get a better understanding of the impact of each feature on the variation of a word, we analyzed the effects of features using partial effects[7]. In the following plots the blue line represents the prediction made by the model. The magenta line represents the LOESS (locally estimated scatterplot smoothing) non parametric-regression smooth of the points, that displays the relationship between the variables[8][9]. Finally, the magenta points represent residuals, i.e. the difference between the observed value and the value predicted by the model (Fox and Hong, 2003; Gries, 2013).

We observed similar effects of the features for all 3 corpora. Figure 6.7 display the partial effects of the cosine similarity score of the nearest neighbor of a word for a randomly sampled multi-linear regression model trained on ACL[10]. We see that words having a higher nearest neighbor similarity score display less variation. On the contrary, when the similarity score is lower, the variation is higher. It seems logical that a very close neighbor remains stable from one model to another. However this is not a systematic behavior since some words having very close neighbors display a high variability.

[7]Using the `R` package `effect` (Fox and Hong, 2003).

[8]`https://en.wikipedia.org/wiki/Local_regression`

[9]`https://en.wikipedia.org/wiki/Scatterplot_smoothing`

[10]Partial effects for PLOS and the BNC models can be found Appendices E and F.

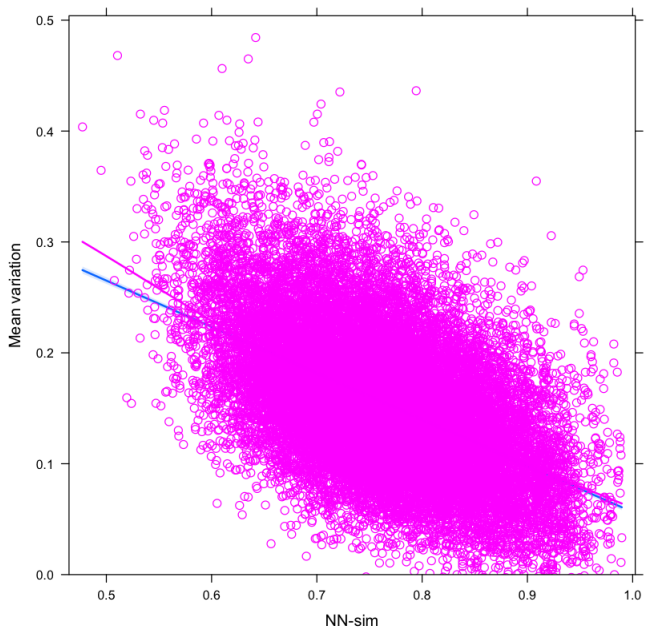


Figure 6.7: Partial effects of the nearest neighbor cosine similarity for a randomly sampled multi-linear regression model trained on ACL.

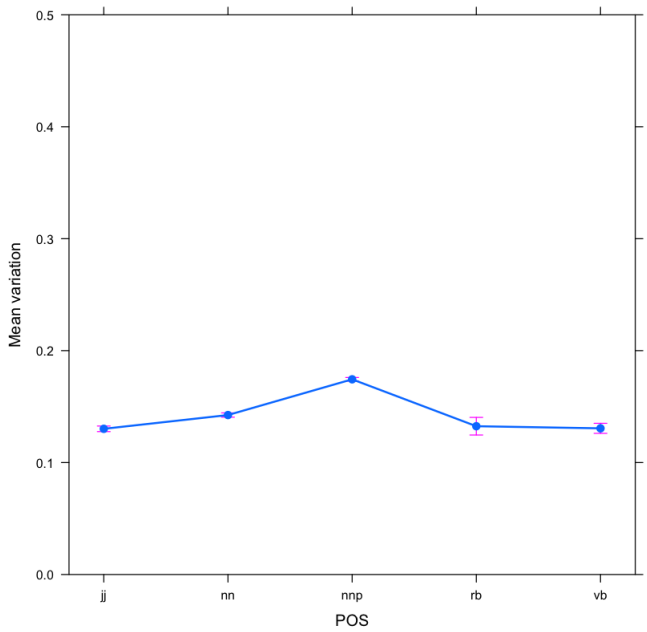


Figure 6.8: Partial effects of POS for a randomly sampled multi-linear regression model trained on ACL.

For POS, figure 6.8 confirms that proper nouns (*nnp*) have a higher variation than other categories, along with nouns on a smaller scale. No differences could be found among other categories.

As we can see on figure 6.9, the norm of the vector is negatively correlated to variation: word with vectors distant from the origin show less variation. This effect was confirmed but less clear for the ACL models. This phenomenon has to be further examined as is the overall geometry of word embeddings vector space. E.g., Mimno and Thompson (2017) have shown that embeddings trained using word2vec Skip-Gram are not evenly dispersed through the semantic space.

The effect of frequency is visible figure 6.10. The predictability of the variation is actually not linear as we saw in section 6.2.1. Words having very low or very high frequency are more affected by variation than words in the mid-frequency range. This partly infirms the common knowledge that embeddings of more frequent words are of better quality. We actually found a number of frequent words displaying instability in each corpus (e.g. *gene* and *protein* in PLOS, *language* in ACL and *make* in BNC etc.).

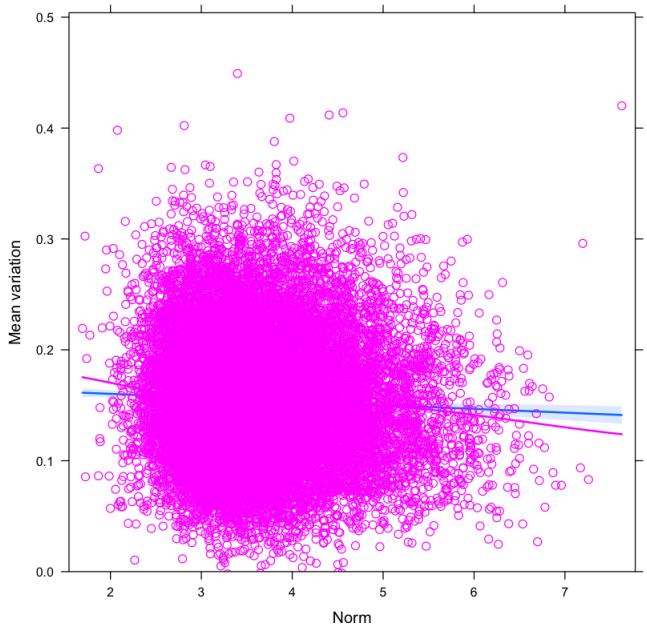


Figure 6.9: Partial effects of the L2-norm for a randomly sampled multi-linear regression model trained on ACL.

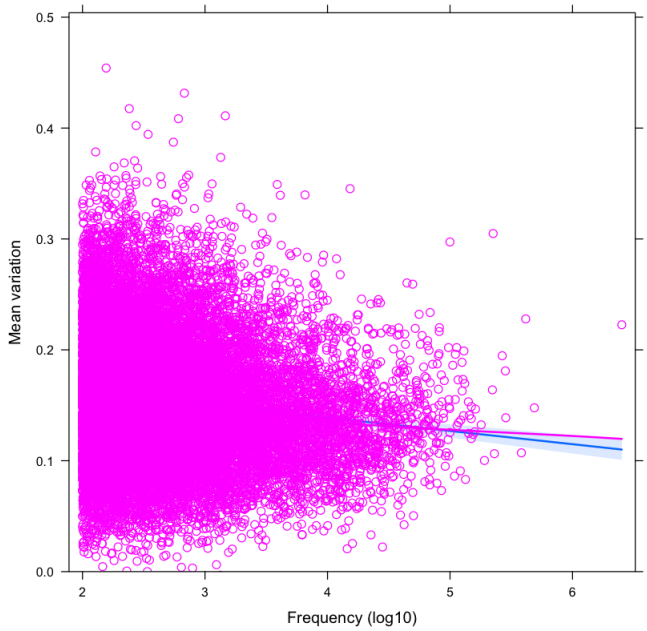


Figure 6.10: Partial effects of frequency for a randomly sampled multi-linear regression model trained on ACL.

The degree of polysemy of a word also has a slight effect on the predictability of the variation of a word as we can see on figure 6.11. The more polysemic a word is, the more likely its variation score is to be high. We expected this effect to be more obvious and we only observed a slight effect. This could be due to the resource we used but it could also be because we computed polysemy as something external to the corpus when in reality we should have computed it according to the different senses used in the corpus to reflect real usage. This would be especially important for specialized corpora.

As for the entropy, we observed for ACL and the BNC, that words having higher entropy with their contexts display more variation. Concerning these two last features (polysemy and entropy) experiments confirm that distributional semantics has more difficulty in representing the meaning of words that appear in scattered contexts.

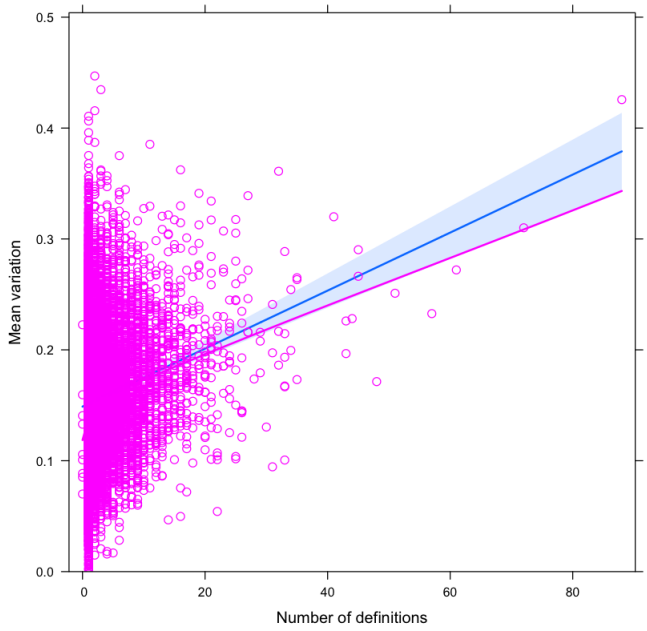


Figure 6.11: Partial effects of polysemy for a randomly sampled multi-linear regression model trained on ACL.

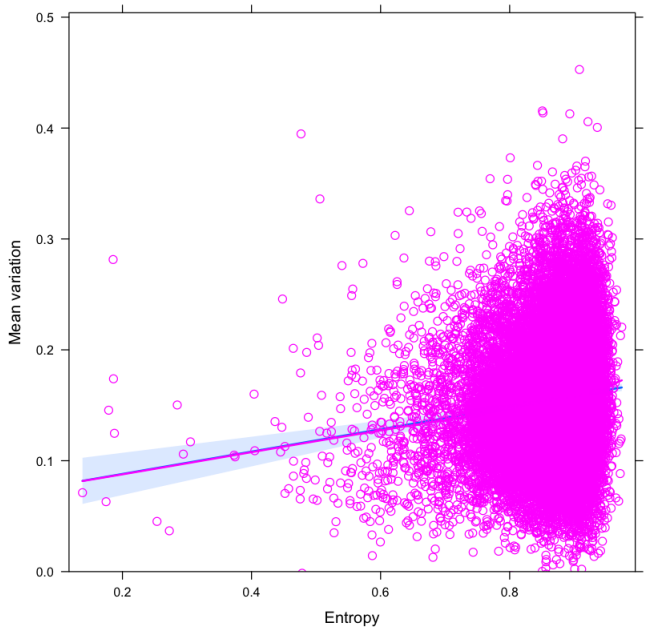


Figure 6.12: Partial effects of entropy for a randomly sampled multi-linear regression model trained on ACL.

6.4.3 Other approaches to the prediction of stability

Concurrently to the research we conducted on the prediction of the variation, another study was done by Wendlandt et al. (2018) to investigate factors influencing the instability of word embeddings trained using different techniques: models created using PPMI, word embeddings trained using word2vec and word embeddings trained using GloVe. The authors define stability as the overlap of the 10 nearest neighbors in a DSM space. First, they examined the relation between stability and frequency. They observed that high frequency words are the most stable and that mid-frequency words are more impacted by instability. This is different from what we observed in section 6.2.1. However this difference could be the result of the different number of nearest neighbors considered[11]. Moreover, our observations were limited to a subset of POS while Wendlandt et al. (2018) examined all POS.

To predict stability the authors chose several properties related to a word (POS, polysemy, number of syllables), to the corpus (2 different corpora, raw frequency, vocabulary size, overlap of vocabulary between corpora) and to the algorithm used to train embeddings (window size and min-count). They trained a ridge regression (Hoerl and Kennard, 1970) to predict the stability of a word across 2 models. These 2 models were not necessarily trained using the same techniques, parameters and corpus. As a consequence their results are different from the observations we made in the previous sections.

Similarly to what we observed, they found that the POS of a word is one of the factors playing an important role on stability. They observed that numerals, verbs and determiners are the most stable while punctuation marks, adpositions and particles are the most unstable. They also observed that frequency is not a major factor in the prediction of stability.

In this section, we were able to further investigate the stability of words through several features related to a word, the corpus it appears in and the model trained. We observed that the cosine score of the nearest neighbor of a word explains partly the stability of a word with top nearest neighbors having a high cosine score being more likely to remain stable across runs. Confirming observations we made before, the POS of a word also has an impact on the word stability, with proper nouns being more unstable.

Naumann et al. (2018) and Frassinelli et al. (2017) have investigated the contexts of concrete and abstract words. We got curious about the influence of the degree of concreteness of a word on its stability. As a consequence we decided to investigate this additional feature. However, we solely focused on nouns because they are easier to qualify (Frassinelli et al., 2017). As a consequence, to avoid interactions with other features and get direct insights about the influence of con-

[11] We observed the 25 nearest neighbors of each word

creteness, we decided to examine this feature by itself rather than integrating it in a prediction model.

6.5 Confronting concreteness to noun stability

Words' concreteness has been studied in cognitive science and psycholinguistics and numerous studies have focused on the way concrete nouns are processed (Naumann et al., 2018). In the Context Availability Theory (Schwanenflugel and Shoben, 1983), meaning comes from the possibility to associate a context to a concept and it has been shown that this is more challenging for abstract than concrete concepts (Naumann et al., 2018). Quantitative investigations of similarities and differences between concrete and abstract words contexts have been conducted in the context of distributional semantics (Frassinelli et al., 2017). It was found that distributionally similar words have similar concreteness ratings, i.e. concrete words have concrete nearest neighbors. It was also shown that the contexts of concrete words tend to have various concreteness scores while abstract words clearly prefer contexts words that are abstract. We wanted to use this information to investigate the role of concreteness on neighbor variation and examine if concrete nouns are less affected by instability than abstract nouns.

6.5.1 Concreteness and neighbors variation

To measure the degree of concreteness of words, we used the resource developed by Brysbaert et al. (2014) presented in section 2.2.2.4. In the following experiments, we only considered nouns since they are easier to qualify in terms of concreteness and abstractness (Frassinelli et al., 2017). The resource contains 14592 nouns that have an average concreteness score of 3.53 ($\pm$ 1.02). For each corpus we only considered the words that existed in the resource, i.e. 8796 nouns for the BNC, 5288 nouns for PLOS and 3899 nouns for ACL. We computed the average concreteness score of nouns for each corpus. This resulted in an average concreteness score of 3.48 for the BNC ($\pm$ 1.02), 3.28 for ACL ($\pm$ 1.01) and 3.47 for PLOS ($\pm$ 0.98). For the BNC and PLOS, nouns have a mean concreteness score close to the mean concreteness score of the resource. However, we notice that nouns in ACL are more abstract. This can be explained by the nature of the corpus which is made of scientific papers mostly dealing with training algorithms and NLP systems.

Our first analysis was made to confirm Frassinelli et al. (2017) result stating that distributionally similar words have similar degrees of concreteness. To test this hypothesis we selected the 1000 most concrete (mean concreteness of 4.88, $\pm$ 0.07) and the 1000 most abstract nouns (mean concreteness of 1.88, $\pm$ 0.21) in the BNC. For each noun, we computed the average concreteness score of nearest

neighbors that were nouns amongst its 25 nearest neighbors. The neighbors of the most concrete nouns have an average concreteness score of 4.6 and the nearest neighbors of abstract nouns have an average concreteness score of 2.37 meaning that distributionally similar words do have similar concreteness scores as stated by Frassinelli et al. (2017). For example, the noun *lemon* (concreteness score of 5, which is the maximum score) has very concrete nearest neighbors that are other fruits, vegetables or food item: *onion* (4.86), *clove* (4.42), *almond* (4.71) etc. On the contrary the very abstract noun *spirituality* (1.07) has abstract neighbors: *theology* (1.93), *mysticism* (1.86), *religion* (1.71) etc.

Because we saw that instability was correlated to frequency, before testing the relation between concreteness and variation we decided to check if concreteness and frequency were correlated. Table 6.5 displays the number of nouns for which we could retrieve a concreteness score in each corpus as well as the correlation between frequency and variation, frequency and concreteness score and concreteness score and variation.

Corpus	**Number of nouns**	**Nouns with concr. score**	**Correl. freq-var**	**Correl. freq-concr.**	**Correl. var-concr.**
ACL	5534	3899	-0.42	-0.12	+0.10
BNC	10266	8796	-0.15	+0.03	-0.16
PLOS	9751	5288	-0.26	-0.07	+0.01 (ns)

Table 6.5: Spearman correlation scores between frequency and variation, frequency and degree of concreteness and variation and degree of concreteness. All correlation scores are significant at the 0.05 level except for the one where *ns* is indicated.

We do not observe the same behaviors for all corpora. In the BNC, a generic corpus, abstract words have a clear tendency to vary more with a Spearman correlation of -0.16. In the BNC, words such as *kitchen*, *wife*, *sitting-room* or *grandmother* are concrete and have a low variation score. These words correspond to the clusters we identified in section 6.3.2. We also observed nouns like *legacy*, *realization*, *succession* or *coverage* that are abstract and whose neighbors vary significantly. They correspond to words that appear in many different contexts when we observed variation in section 6.3.3.

Our observations were rather different for the specialized corpora. No effect was visible in PLOS and the opposite effect was observed in ACL with a positive correlation. Concrete words such as *carrot*, *turtle*, *umbrella* or *horse* vary a lot. These words have a low frequency in the corpus (around 100 occurrences) and correspond to words that are used in examples. This also explains the higher negative correlation between concreteness and frequency in ACL. Abstract words

in ACL correspond to words that are very stable across the different models, e.g. *recall* and *precision*.

This difference in behavior observed between specialized and non-specialized corpora is not surprising since we use a resource where concreteness was defined as something you can experience through your senses. This raises questions concerning the notion of concreteness and what it means for a word to be concrete in a specialized corpus. It seems very important to consider the nature of the corpus when performing this type of experiments and to take into consideration that changing corpus equals to changing world. This question is especially crucial when working in specialized domains where the quantity of available data might be limited.

Regarding frequency, we had already observed a globally negative correlation between frequency and variation. We also saw that the relation between variation and frequency is not linear with words in very low or high frequency range having a tendency to vary more than words in the mid-frequency range. As we can see in table 6.5, we notice that the correlation between frequency and concreteness is almost null for the BNC. However we observe a weak negative correlation for ACL and PLOS with less frequent words being more concrete. This means that frequency is not an indirect factor of the observed correlation between concreteness and variation.

6.5.2 Concreteness as an indicator of neighbors stability

In section 6.3.1, we confirmed that stable words have stable neighbors. We then wanted to see if stable neighbors are more concrete than unstable neighbors. To do so, we only experimented with the BNC. For each noun, we retrieved the union of its 25 nearest neighbors in the 5 models. Because we only retrieved concreteness ratings for nouns, we only kept nearest neighbors that were nouns. We associated the concreteness score from the resource to each of those nouns. For each neighbor, we computed its cosine similarity with the target word as well as the absolute difference between its degree of concreteness and the degree of concreteness of the target word. We also computed the standard deviation of cosine scores across the 5 models.

We selected highly concrete nouns (nouns with a concreteness score above 4.2) and computed the Spearman correlation between the absolute difference of concreteness and the standard deviation of cosines. We found that in 65% of the cases where the correlation is significant the correlation is positive. This means that when the concreteness score of a target word with one of its nearest neighbor is very high, this neighbor is more likely to change from one model to another. E.g., we observed this phenomenon with the word *telescope* which has a concreteness score of 5. Let's look at two of its closest neighbors. *Wavelength* has a concreteness

score of 3.35 and has an average cosine score of 0.67 with *telescope*. *Lens* has a concreteness score of 4.64 and has an average cosine scores of 0.62. We notice that both cosine scores are very similar. However the similarity score of *telescope* with *wavelength*, which is more abstract, displays much more variation across the 5 models (0.013 against 0.005 for *lens*). This effect is less visible for more abstract nouns.

6.6 Conclusion

The instability of nearest neighbors is a phenomenon that goes hand in hand with the use of word embeddings trained using neural-based methods. In this chapter, rather than trying to find solutions to avoid the instability phenomenon, we proposed to embrace it and investigated it from a linguistic point of view on 3 different corpora (ACL, BNC and PLOS).

While we could not explain all the observed instability through the different factors we examined, we were able to identify patterns in the variation that help us better understanding this phenomenon. We saw that mid-frequency words are less impacted by variation and that words varying the most were low and high frequency words. We also found that proper nouns had a tendency to vary more in all three corpora, whether they correspond to names or acronyms.

We were also able to identify semantic zones that remain stable across trainings. We found that these semantic clusters are very similar across corpora (specific localized contexts, closed classes of co-hyponyms and transdisciplinary scientific lexicon). Because we observed patterns for words varying the most for all 3 corpora, we decided to investigate the instability by training multiple linear regressions to investigate the impact of each features on the prediction of the (in)stability. The features we examined were intrinsic to the word (POS and polysemy), to the corpus (frequency and entropy) and to the model (L2-norm and nearest neighbor similarity score). We found that the nearest neighbor similarity score was the most important feature when predicting the variation of a word's nearest neighbors. We were also surprised to observe that the POS of a word has an important impact when predicting the instability of a word.

Finally, we investigated the stability using a more subtle feature, the degree of concreteness of words. We specifically focused on nouns and confirmed that concrete words tend to have concrete neighbors. We also observed that for concrete nouns, concrete neighbors are less likely to change across runs.

Conclusion and perspectives

Contributions

Despite advances in the techniques proposed to train word embeddings, the evaluation of these dense semantic representations of words remains a challenging problem, and it is difficult to find and apply the appropriate evaluation method. In this PhD thesis, we proposed to evaluate word embeddings differently by incorporating qualitative aspects to the evaluation process. We proposed a method that allows to compare what changes in the neighborhoods of words across pairs of models, without relying on any external resource. Because the proposed method is resource independent, our method suffers less bias than traditional methods used to evaluate models, such as intrinsic evaluation datasets.

The methodology process we presented provides both a global overview of what changes across models, by computing an average variation score of all words for a pair of models, as well as a local overview by providing an average variation score per word. The variation scores are easy to compute, to use and to interpret, giving immediate feedback on what changes across models. A high variation score for a given word indicates an important change in the neighborhood of that word, while a low variation score indicates that the neighbors remained the same across the compared models. While intrinsic datasets present the advantage of rapidly detecting if a model is sane (scores that are low would indicate a problem with the current model), their coverage is rather limited with the evaluation of only a few hundreds of different words. Contrary to intrinsic datasets, the proposed method is not limited to a subset of the vocabulary. We must acknowledge that using it by itself does not allow to detect if a model has a problem nor does it gives information about which model is "better" when there are important differences. However, the method we proposed is directed towards exploring what changes across models rather than judging if these changes are good or bad. Another interesting aspect of this method is that it guides towards elements that can lead to further exploration.

We used this method to conduct different investigations on models trained using word2vec. In chapter 4, we compared nearest neighbors of words across models

trained with different hyperparameters. We were able to detect variations across models that were not captured by intrinsic datasets. Moreover, this comparison allowed us to observe that some hyperparameters have more impact than others when training models with word2vec by completely disrupting the neighborhoods in the lexical space. We decided to explore these changes through two straightforward factors, the frequency of a word and its POS. These two factors gave us interesting insights about the variation across models. E.g. we found that words with a low or high frequency are more impacted by neighbors' variation, their neighbors are more likely to change across models. The observed effect of POS was less important but we did notice that proper nouns have a tendency to vary more. We were also able to manually identify regularities across words presenting the most and the least variation.

Because word2vec is a neural-based method, it is impacted by instability problems and semantic representations change across trainings done with the same hyperparameters. As a consequence, we could not completely understand the variation observed in chapter 4 without assessing the inherent instability of word embeddings. In chapter 5, we explained the instability phenomenon and measured its amplitude on three different corpora, to ensure that the measured variation was not an indirect effect of the corpus used for training. We observed that the variation of nearest neighbors across models trained with the same hyperparameters was lower than what we had measured for models trained with different hyperparameters. However, it was not negligible and it seemed highly important to examine this phenomenon. As a consequence, in chapter 6, we decided to try to shed light on this phenomenon by investigating it through different factors that could help understanding it better. We were able to detect that not all words are impacted the same by instability, with some words varying more than others. We also identified clusters corresponding to semantic zones that remain stable across training for all three corpora. We also observed recurring patterns for words highly impacted by variation (polysemous words, generic adjectives etc.). Finally, we explored the variation with using several features intrinsic to a word, the corpus used for training and the models, to get a better understanding of the instability phenomenon. We found that these features help us explaining the (in)stability of some words.

Perspectives

While we were not able to completely explain the instability of word embeddings trained using word2vec, we provided a qualitative analysis of the phenomenon, using linguistic features to understand word embeddings better. Several directions could be followed to continue this work. First, it would be interesting to confirm

that the experiments made in this thesis are not biased by the size of the corpus used to train models since we used 3 corpora of about the same size (100 million words). We started to conduct experiments with a larger corpus, UMBC, a web-based corpus constituted of about 3 billion words (Han et al., 2013). Preliminary results showed that the same amount of nearest neighbors variation was observed when training models using the same hyperparameters, proving that the amount of variation is not related to the size of the corpora we used. However, it would be interesting to see if we are able to identify similar semantic clusters for this corpus than the one observed for the BNC, ACL and PLOS.

The measure we used to quantify nearest neighbors variation does not take the rank of neighbors into account. While this made the presented measure computationally efficient, it would be interesting to investigate how the rank is related to the variation of nearest neighbors. Similarly, it would be interesting to get a closer look at the density of neighborhoods and see if neighborhoods that are more compact correspond to the semantic stable zones we observed in chapter 6.

Finally, we started to investigate more sophisticated linguistic features such as concreteness which correspond to more subtle characteristics of words. We would like to observe the impact of the degree of concreteness of words differently, by grouping words using several features (e.g. very concrete words that are very frequent, very concrete words that are rare etc.) to see if the observed behaviors are different.

While using neural-based word embeddings to perform qualitative studies is at your own risk, we do not necessarily advise against it. However, in view of the observations made about the instability phenomenon, we think it is necessary to acknowledge it. While the analyses can be validated by making observations across models trained with the same hyperparameters, being aware of the phenomenon and referring to it is already a big step. We find it reassuring that we were able to identify features that help defining semantic zones across models and corpora that are more or less impacted by instability. Understanding the tools used for linguistic analyses is crucial because it helps validating the observed phenomena. Moreover, a better comprehension of these models means being able to identify if a phenomenon is the result of a random process or if it is the result of a linguistic phenomenon that is worth investigating. Finally, it seems important to keep investigating and questioning models, even the best performing ones, by adopting a linguistic point of view as it leads to a better understanding and use of these models and to better science.

References

Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M., and Soroa, A. (2009). A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches. In *Human Language Technologies: The 2015 Conference of the North American Chapter of the ACL*, pages 19–27, Boulder, Colorado.

Andreas, J. and Klein, D. (2014). How Much Do Word Embeddings Encode About Syntax? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 822–827.

Antoniak, M. and Mimno, D. (2018). Evaluating the Stability of Embedding-based Word Similarities. *Transactions of the Association for Computational Linguistics*, 6:107–119.

Asr, F. T., Willits, J. A., and Jones, M. N. (2016). Comparing Predictive and Co-occurrence Based Models of Lexical Semantics Trained on Child-directed Speech. In *Proceedings of the 37th Meeting of the Cognitive Science Society*.

Bansal, M., Gimpel, K., and Livescu, K. (2014). Tailoring Continuous Word Representations for Dependency Parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pages 809–815, Baltimore, Maryland, USA.

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't Count, Predict! A Systematic Comparison of Context-Counting vs. Context-Predicting Semantic Vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 238–247, Baltimore, Maryland, USA.

Baroni, M. and Lenci, A. (2010). Distributional Memory: A General Framework for Corpus-Based Semantics. *Computational Linguistics*, 36(4):673–721.

Baroni, M. and Lenci, A. (2011). How we BLESSed Distributional Semantic Evaluation. In *Proceedings of the GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics, EMNLP 2011*, pages 1–10, Edinburgh, Scotland, UK.

Batchkarov, M., Kober, T., Reffin, J., Weeds, J., and Weir, D. (2016). A Critique of Word Similarity as a Method for Evaluating Distributional Semantic Models. In *The First Workshop on Evaluating Vector Space Representations for NLP*, Berlin.

Bengio, Y. and Ducharme, R. (2001). A Neural Probabilistic Language Model. In *NIPS*.

Bernier-Colborne, G. and Drouin, P. (2016). Evaluation of Distributional Semantic Models: A Holistic Approach. In *Proceedings of the 5th International Workshop on Computational Terminology*, pages 52–61, Osaka, Japan.

Bird, S., Dale, R., Dorr, B., Gibson, B., Joseph, M., Kan, M.-Y., Lee, D., Powley, B., Radev, D., and Fan Tan, Y. (2008). The ACL Anthology Reference Corpus: A Reference Dataset for Bibliographic Research in Computational Linguistics. In *Proceedings of Language Resources and Evaluation Conference (LREC 08)*.

Blei, D. B., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching Word Vectors with Subword Information. In *EMNLP*, volume 91, pages 28–29.

Bolukbasi, T., Chang, K.-W., Zou, J., Saligrama, V., and Kalai, A. (2016). Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *30th Congerence on Neural Information Processing Systems (NIPS 2016)*, Barcelona, Spain.

Boudin, F. (2013). TALN Archives: Une Archive Numérique Francophone des Articles de Recherche en Traitement Automatique de la Langue. In *Actes de TALN'2013*, pages 507–514, Les Sables d'Olonne.

Bravo Candel, M. (2019). Évaluation des Similarités Lexicales Instables Identifiées par Analyse Distributionnelle. Technical report, Université Toulouse II.

Bruni, E., Tran, N.-K., and Baroni, M. (2013). Multimodal Distributional Semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

Brysbaert, M., Warriner, A. B., and Kuperman, V. (2014). Concreteness Ratings for 40 Thousand Generally Known English Word Lemmas. *Behavior Research Methods*, 46:904–911.

Budanitsky, A. and Hirst, G. (2006). Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13–47.

Caselles-Dupré, H., Lesaint, F., and Royo-Letelier, J. (2018). Word2vec Applied to Recommendation: Hyperparameters Matter. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys'18*, 352-356.

Chiu, B., Crichton, G., Korhonen, A., and Pyysalo, S. (2016). How to Train Good Word Embeddings for Biomedical NLP. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 166–174, Berlin, Germany.

Church, K. W. and Hanks, P. (1990). Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, 16(1):22–29.

Clark, S. (2015). Vector Space Models of Lexical Meaning. In Lappin, S. and Fox, C., editors, *The Handbook of Contemporary Semantic Theory*, pages 493–522. John Wiley & Sons, Ltd.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning.

Curran, J. R. (2003). *From Distributional to Semantic Similarity.* PhD Thesis, University of Edinburgh, Edinburgh, Scotland, UK.

Deerwester, S., Dumais, S. T., Laundaer, T. K., Furnas, G. W., and Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference on the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Fabre, C., Hathout, N., Sajous, F., and Tanguy, L. (2014). Ajuster l'Analyse Distributionnelle à un Corpus Spécialisé de Petite Taille. In *21ème Traitement Automatique des Langues Naturelles*, pages 266–279.

Fabre, C. and Lenci, A. (2015). Distributional Semantics Today Introduction to the special issue. *TAL*, 56(2):7–20.

Faruqui, M., Tsvetkov, Y., Rastogi, P., and Dyer, C. (2016). Problems with Evaluation of Word Embeddings Using Word Similarity Tasks. In *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*, pages 30–35, Berlin, Germany.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, MA.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2002). Placing Search in Context: The Concept Revisited. In *ACM Transactions on Information Systems*, volume 20, page 116:131.

Firth, J. (1957). *A Synopsis of Linguistic Theory. Studies in Linguistic Analysis.* Blackwell, Oxford.

Fox, J. and Hong, J. (2003). Effect Displays in R for Generalised Linear Models. *Journal of Statistival Software*, 8(15):1–27.

Frassinelli, D., Naumann, D., Utt, J., and Schulte im Walde, S. (2017). Contextual Characteristics of Concrete and Abstract Words. In *IWCS 2017 - 12th International Conference on Computational Semantics - Short papers.*

Galliers, J. R. and Sparck Jones, K. (1993). The Framework: Scope and Concepts. In *Evaluating Natural Language Processing Systems.* Cambridge, UK.

Gaume, B., Ho-Dac, L.-M., Tanguy, L., Fabre, C., Pierrejean, B., Hathout, N., Farinas, J., Pinquier, J., Danet, L., Péran, P., De Boissezon, X., and Jucla, M. (2019). Towards a Computational Multidimensional Lexical Similarity Measure for Modeling Word Association Tasks in Psycholinguistics. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 71–76.

Gaume, B., Tanguy, L., Fabre, C., Ho-Dac, L.-M., Pierrejean, B., Hathout, N., Farinas, J., Pinquier, J., Danet, L., Péran, P., De Boissezon, X., and Jucla, M. (2018). Automatic Analysis of Word Association Data from the Evolex Psycholinguistic Tasks Using Computational Lexical Semantic Similarity Measures. In *13th International Workshop on Natural Language Processing and Cognitive Science (NLPCS)*, Krakow, Poland.

Gladkova, A. (2016). Intrinsic Evaluations of Word Embeddings: What Can We Do Better? In *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*, pages 36–42.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning.* Cambridge, MA, the mit press edition.

Grefenstette, G. (1994). *Explorations in Automatic Thesaurus Discovery.* Kluwer Academic Publishers, Norwell, MA, USA.

Gries, S. T. (2013). *Statistics for Linguistics with R: A Practical Introduction.* De Gruyter Mouton, 2nd revised edition edition.

Gupta, A., Boleda, G., Baroni, M., and Pado, S. (2015). Distributional Vectors Encode Referential Attributes. In *Proceedings of EMNLP 2015 (Conference on Empirical Methods in Natural Language Processing)*.

Hamilton, W. L., Leskovec, J., and Jurafsky, D. (2016). Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1489–1501, Berlin, Germany.

Han, L., Kashyap, A. L., Finin, T., Mayfield, J., and Weese, J. (2013). UMBC EBIQUITY-CORE: Semantic Textual Similarity Systems. In *Proceeding of the 2nd Joint Conference on Lexical and Computational Semantics, Association for Computational Linguistics*.

Harris, Z. (1970). Distributional Structure. *Papers in Structural and Transformational Linguistics*, pages 775–794.

Harris, Z. S. (1954). Distributional Structure. *Word*, 10(2-3):146–162.

Hellrich, J. and Hahn, U. (2016). Bad Company - Neighborhoods in Neural Embedding Spaces Considered Harmful. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2785–2796, Osaka, Japan.

Herbelot, A. and Baroni, M. (2017). High-Risk Learning: Acquiring New Word Vectors from Tiny Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 304–309, Copenhagen, Denmark.

Hill, F., Reichart, R., and Korhonen, A. (2015). SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. *Computational Linguistics*, 41:665–695.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge Regression: Biased Estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67.

Hutson, M. (2018). Artificial Intelligence Faces Reproducibility Crisis. *Science*, 359(6377):725–726.

Jimenez, I., Sevilla, M., Watkins, N., Maltzann, C., Lofstead, J., Mohror, K., Arpaci-Dusseau, A., and Arpaci-Dusseau, R. (2017). The Popper Convention: Making Reproducible Systems Evaluation Practical. pages 1561–1570.

Jurafsky, D. and Martin, J. H. (2018). Vector Semantics. In *Speech and Language Processing*, volume Draft of October 16, 2019.

Kiela, D. and Clark, S. (2014). A Systematic Study of Semantic Vector Space Model Parameters. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality CVSC EACL , Gothenburg, Sweden, April 2014*, volume 353, pages 21–30.

Kim, Y., Chiu, Y.-I., Hanaki, K., Hegde, D., and Petrov, S. (2014). Temporal Analysis of Language through Neural Language Models. In *Proceedings of the ACL Workshop on Language Technologies and Computational Social Science*, pages 61–65, Baltimore, Maryland, USA.

Kulkarni, V., Al-Rfou', R., Perozzi, B., and Skiena, S. (2015). Statistically Significant Detection of Linguistic Change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635.

Lakoff, G. (1987). *Women, Fire and Dangerous Things*. University of Chicago Press, Chicago.

Lakoff, G. and Johnson, M. (1980). *Metaphors we Live by*. University of Chicago Press, Chicago.

Landauer, T. K. and Dumais, 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.

Lapesa, G. and Evert, S. (2014). A Large Scale Evaluation of Distributional Semantic Models: Parameters, Interactions and Model Selection. *Transactions of the Association for Computational Linguistics*, 2:531–545.

Lapesa, G. and Evert, S. (2017). Large-Scale Evaluation of Dependency-Based DSMs: Are they Worth the Effort? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 2, Short papers, pages 394–400, Valencia, Spain.

Lee, D. D. and Seung, H. S. (1999). Learning the Parts of Objects by Non-Negative Matric Factorization. *Nature*, 401:788–791.

Lenci, A. (2008). Distributional Semantics in Linguistic and Cognitive Research. *Rivista di Linguistica*, 20(1):1–31.

Levy, O. and Goldberg, Y. (2014a). Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 302–308, Baltimore, Maryland, USA.

Levy, O. and Goldberg, Y. (2014b). Neural Word Embedding as Implicit Matrix Factorization. In *Advances in neural information processing systems*, pages 2177–2185.

Levy, O., Goldberg, Y., and Dagan, I. (2015). Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Li, B., Liu, T., Zhao, Z., Tang, B., Drozd, A., Rogers, A., and Du, X. (2017). Investigating Different Syntactic Context Types and Context Representations for Learning Word Embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2411–2421.

Lin, D. (1998). Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity. In *Proceedings of the 35th Annuak Meeting of the Association for Computational Linguistics and Eight Conference on the European Chapter of the Association for Computational Linguistics*, pages 64–71, Madrid, Spain.

Liu, H. and Singh, P. (2004). ConceptNet-a Practical Commonsense Reasoning Toolkit. *BT Technology Journal*.

MacWhinney, B. (2000). The CHILDES Project: The Database (Vol.2).

McRae, K., Cree, G., Seidenberg, M., and McNorgan, C. (2005). Semantic Feature Production Norms for a Large Set of Living and Nonliving things. *Behavior Research Methods*, 37(4):547–559.

Melamud, O., McClosky, D., Patwardhan, S., and Bansal, M. (2016). The Role of Context Types and Dimensionality in Learning Word Embeddings. In *Proceedings of NAACL-HLT 2016*, San Diego, California.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. In *ICLR Workshop*.

Mikolov, T., Le, Q. V., and Sutskever, I. (2013b). Exploiting Similarities among Languages for Machine Translation. *CoRR, abs/1309.4168*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013c). Distributed Representations of Words and Phrases and their Compositionality. In *NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems*, volume 2, pages 3111–3119, Lake Tahoe, Nevada.

Miller, G. A. (1995). WordNet: A Lexical Database for English. In *Communicatoins of the ACM*, volume 38 of *11*, pages 39–41.

Mimno, D. and Thompson, L. (2017). The Strange Geometry of Skip-Gram with Negative Sampling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2873–2878, Copenhagen, Denmark.

Morris, J. and Hirst, G. (2004). Non-classical Lexical Semantic Relations. In *Workshop on Computational Lexical Semantics*, Boston, MA.

Muller, P., Fabre, C., and Adam, C. (2014). Predicting the Relevance of Distributional Semantic Similarity with Contextual Information. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 479–488, Baltimore, Maryland, USA.

Naumann, D., Frassinelli, D., and Schulte im Walde, S. (2018). Quantitative Semantic Variation in the Contexts of Concrete and Abstract Words. In *Proceedings of the 7th Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 76–85, New Orleans.

Osgood, C. E., Suci, G. J., and Tannenbaum, P. (1957). *The Measurement of Meaning*. University of Illinois Press.

Padó, S. and Lapata, M. (2007). Dependency-Based Construction of Semantic Space Models. *Computational Linguistics*, 33(2):161–199.

Pennington, J., Socher, R., and Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Christopher, C., Kenton, L., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations. In *NAACL*.

Pierrejean, B. and Tanguy, L. (2018a). Étude de la Reproductibilité des Word Embeddings : Repérage des Zones Stables et Instables dans le Lexique. In *TALN*, Rennes, France.

Pierrejean, B. and Tanguy, L. (2018b). Predicting Word Embeddings Variability. In *Proceedings of the 7th Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 154–159, New Orleans.

Pierrejean, B. and Tanguy, L. (2018c). Towards Qualitative Word Embeddings Evaluation: Measuring Neighbors Variation. In *Proceedings of NAACL-HLT 2018: Student Research Workshop*, pages 32–39, New Orleans.

Pierrejean, B. and Tanguy, L. (2019). Investigating the Stability of Concrete Nouns in Word Embeddings. In *Proceedings of the 13th International Conference on Computational Semantics*, pages 65–70, Gothenburg, Sweden.

Pulman, S. (2013). Distributional Semantics. In Heunen, C., Sadrzadeh, M., and Grefenstette, E., editors, *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*, pages 333–358. Oxford University Press, Oxford.

R Core Team (2017). R: A Language and Environment for Statistical Computing.

Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850.

Rapp, R. (2003). Word Sense Discovery Based on Sense Descriptor Similarity. In *Proceedings of the Ninth Machine Translation Summit*, pages 315–322.

Rehurek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA.

Resnik, P. and Lin, J. (2010). Evaluation of NLP Systems. In *The Handbook of Computational Linguistics and Natural Language Processing*, pages 271–295. Wiley-Blackwell.

Roberts, K. (2016). Assessing the Corpus Size vs. Similarity Trade-off for Word Embeddings in Clinical NLP. In *Proceedings of the Clinical Natural Language Processing Workshop*, pages 54–63, Osaka, Japan.

Rubenstein, H. and Goodenough, J. (1965). Contextual Correlates of Synonymy. In *Communicatoins of the ACM*.

Rubinstein, D., Levi, E., Schwartz, R., and Rappoport, A. (2015). How Well Do Distributional Models Capture Different Types of Semantic Knowledge? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers)*, pages 726–730.

Sahlgren, M. (2005). An Introduction to Random Indexing. In *Proceedings of the Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering (TKE)*.

Sahlgren, M. (2006). *The Word-Space Model*. PhD Thesis, Stockholm University, Sweden.

Sahlgren, M. and Lenci, A. (2016). The Effects of Data Size and Frequency Range on Distributional Semantic Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 975–980, Austin, Texas.

Sajous, F. and Hathout, N. (2015). GLAWI, a Free XML-Encoded Machine-Readable Dictionary Built from the French Wikitionary. In *Proceedings of the eLex 2015 conference*, pages 405–426, Herstmonceux, England.

Salton, G. (1971). *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice Hall.

Santus, E., Yung, F., Lenci, A., and Huang, C.-R. (2015). EVALution1.0: an Evolving Semantic Dataset for Training and Evaluation of Distributional Semantic Models. In *Proceedings of the 4th Workshop on Linked Data in Linguistics (LDL-2015)*, pages 64–69, Beijing, China.

Schnabel, T., Labutov, I., Mimno, D., and Joachims, T. (2015). Evaluation Methods for Unsupervised Word Embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307.

Schütze, H. (1993). Word Space. In *Proceedings of the 1993 Conference on Advances in Neural Information Processing Systems*, pages 895–902, San Franciso, CA, USA.

Schütze, H. (1998). Automatic Word Sense Discrimination. *Computational Linguistics*, 24(1):97–123.

Schwanenflugel, P. J. and Shoben, E. J. (1983). Differential Context Effects in the Comprehension of Abstract and Concrete Verbal Materials. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 9(1):82–102.

Senel, L. K., Utlu, I., Yucesoy, V., Koc, A., and Cukur, T. (2018). Semantic Structure and Interpretability of Word Embeddings. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 26(10):1769–1779.

Tanguy, L., Brunet, P., and Ferret, O. (2019). Comparaison Qualitative et Extrinsèque d'Analyseurs Syntaxiques du Français : Confrontation de Modèles Distributionnels sur un Corpus Spécialisé. In *TALN*, Toulouse.

Tanguy, L., Sajous, F., and Hathout, N. (2015). Evaluation sur Mesure de Modèles Distributionnels sur un Corpus Spécialisé : Comparaison des Approches par Contextes Syntaxiques et par Fenêtres Graphiques. *TAL*, 56(2):103–127.

Tenney, I., Das, D., and Pavlick, E. (2019). BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy.

Tissier, J., Gravier, C., and Habrard, A. (2017). Dict2vec: Learning Word Embeddings using Lexical Dictionaries. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Trost, T. A. and Klakow, D. (2017). Parameter Free Hierarchical Graph-Based Clustering for Analyzing Continuous Word Embeddings. In *Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing*, Vancouver, Canada.

Turian, J., Ratinov, L., Bengio, Y., and Roth, D. (2009). A Preliminary Evaluation of Word Representations for Named-Entity Recognition. In *NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*.

Turney, P. D. and Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Tutin, A. (2007). Traitement Sémantique par Analyse Distributionnelle des Noms Transdisciplinaires des Ecrits Scientifiques. In *Actes de la 14e conférence sur le Traitement Automatique des Langues Naturelles (TALN'2007)*, pages 283–292, Toulouse, France.

Urieli, A. (2013). *Robust French syntax analysis: reconciling statistical methods and linguistic knowledge in the Talismane toolkit*. PhD Thesis, Université Toulouse-II Le Mirail.

Wang, T., Mohamed, A.-r., and Hirst, G. (2015). Learning Lexical Embeddings with Syntactic and Lexicographic Knowledge. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers)*, pages 458–463.

Webber, W., Moffat, A., and Zobel, J. (2010). A Similarity Measure for Indefinite Rankings. In *ACM Transactions on Information Systems*, volume 28.

Wendlandt, L., Kummerfeld, J. K., and Mihalcea, R. (2018). Factors Influencing the Surprising Instability of Word Embeddings. In *Proceedings of NAACL-HLT 2018*, pages 2092–2102, New Orleans.

Yu, M. and Dredze, M. (2014). Improving Lexical Embeddings with Semantic Knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pages 545–550.

APPENDICES

Appendix A

Words varying the least - ACL

Word	Mean var. across models
jj#51st	0
nn#decline	0
nnp#buch	0
nnp#g.	0
jj#50th	0
nn#boldface	0
nn#recall	0
vb#observe	0
nnp#1a	0
nnp#arda	0
nnp#10a	0
jj#53rd	0
nnp#5c	0
nnp#6b	0
nn#precision	0
jj#49th	0
vb#describe	0
jj#tree-to-string	0
nnp#12a	0
nnp#6a	0
vb#try	0
nnp#fmeasure	0
nnp#52nd	0
nnp#1b	0
nnp#1d	0

Word	**Mean var. across models**
nnp#uso	0
nnp#f-score	0
nn#t-test	0
nn#reliability	0
nn#foram	0
nnp#4-9	0
nnp#2c	0
nnp#f1	0.016
nnp#9a	0.016
nnp#n-grams	0.016

Appendix B

Words varying the least - PLOS

Word	Mean var. across models
nnp #bis-tris	0
jj #100-fold	0
nn #rt-pcr	0
nnp #maldi	0
nn #2-fold	0
nnp #june	0
nnp #october	0
jj #one-way	0
nn #two-fold	0
rb #firstly	0
nnp #colistin	0
nn #np-40	0
nnp #puromycin	0
nn #polyacrylamide	0
jj #8th	0
jj #2-fold	0
rb #moreover	0
nnp #table	0
nnp #modified	0
rb #tenfold	0
nn #mean-sd	0
nnp #berthold	0
nnp #tris-buffered	0
jj #three-fold	0
nnp #g418	0

Word	**Mean var. across models**
vb #abolish	0
nn #sem	0
rb #furthermore	0
jj #ninth	0
rb #overall	0.016
jj #10-fold	0.016
jj #2-tailed	0.016
nnp #96-well	0.016
nnp #difco	0.016
nnp #hygromycin	0.016

Appendix C

Words varying the most - ACL

Word	Mean var. across models
nnp #ment	0.504
nnp #lowe	0.496
jj #inclusive	0.468
nnp #steve	0.46
nnp #finn	0.456
nn #gem	0.448
nnp #foo	0.444
nnp #hand	0.444
nnp #joyce	0.444
nnp #meet	0.444
nnp #lang	0.44
nnp #cbs	0.44
nn #cd	0.436
nnp #irvine	0.432
nnp #pd	0.432
nn #ment	0.432
nnp #scs	0.432
nn #tor	0.428
nnp #jay	0.428
nn #genotype	0.424
nn #may	0.424
nnp #dps	0.424
jj #to	0.42
nnp #ivan	0.42
nn #pack	0.416

Word	**Mean var. across models**
nnp #alt	0.416
nn #pre	0.416
nnp #generalize	0.412
nnp #organ	0.412
nnp #just	0.412
nnp #general	0.408
nnp #lines	0.408
nn #tions	0.408
nnp #equivalence	0.408
nnp #tex	0.404

Appendix D

Words varying the most - PLOS

Word	Mean var. across models
nnp #era	0.604
nnp #hsr	0.596
nnp #ae	0.588
nnp #phi	0.576
nnp #pcb	0.552
nnp #dmc	0.548
nnp #cnm	0.548
nnp #pra	0.548
nnp #cvm	0.544
nnp #dot	0.54
nnp #car	0.54
nn #pt	0.536
nnp #arp	0.532
nnp #qm	0.524
nnp #ppf	0.524
nn #ip	0.52
nnp #gra	0.516
nnp #rcs	0.516
nnp #sh	0.512
nnp #mgd	0.512
nnp #zp	0.512
nnp #sps	0.512
nnp #pic	0.512
nnp #tlp	0.508

Word	**Mean var. across models**
nnp #acd	0.508
nnp #pmps	0.508
nnp #ahr	0.508
nnp #gml	0.504
nnp #aip	0.504
nnp #sls	0.504
nn #si	0.5
nnp #an	0.5
nnp #tg2	0.5
nnp #px	0.5

Appendix E

Partial effects for a randomly sampled model trained on PLOS

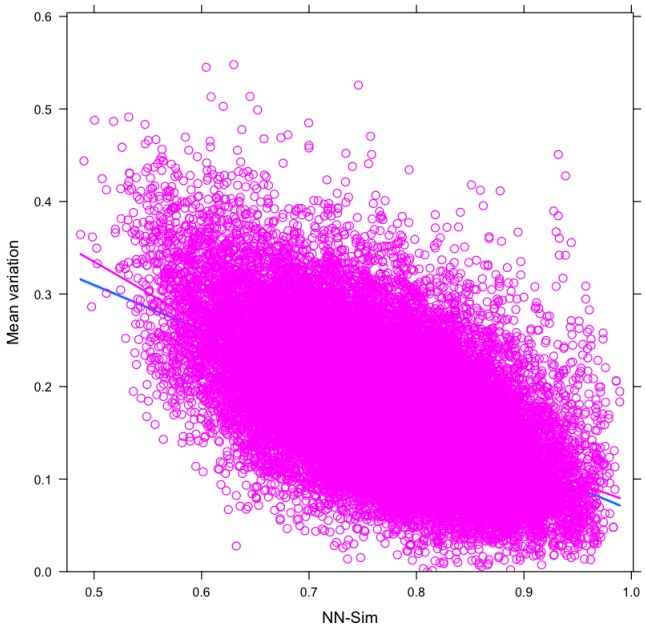


Figure E.1: Partial effects of nearest neighbor cosine similarity for a randomly sampled multi-linear regression model trained on PLOS.

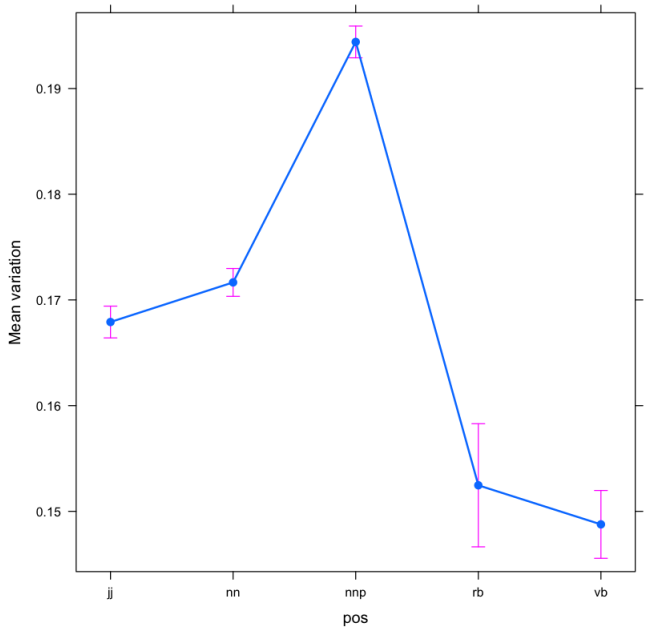


Figure E.2: Partial effects of the POS for a randomly sampled multi-linear regression model trained on PLOS.

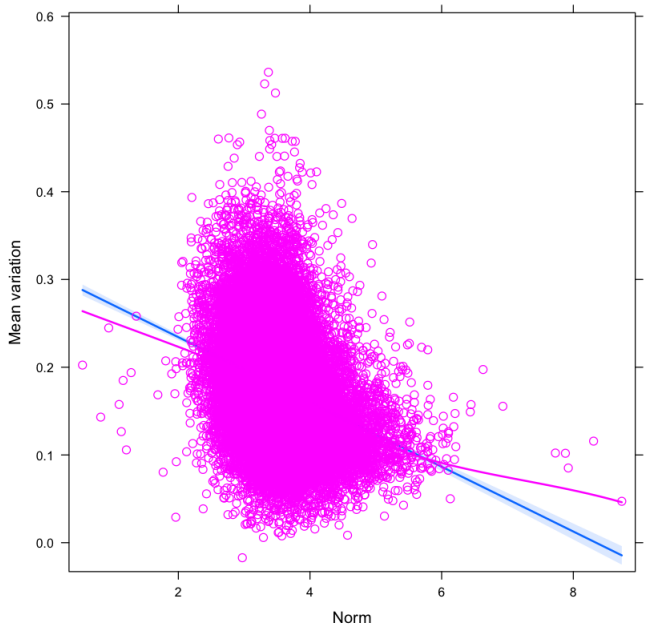


Figure E.3: Partial effects of the L2-norm for a randomly sampled multi-linear regression model trained on PLOS.

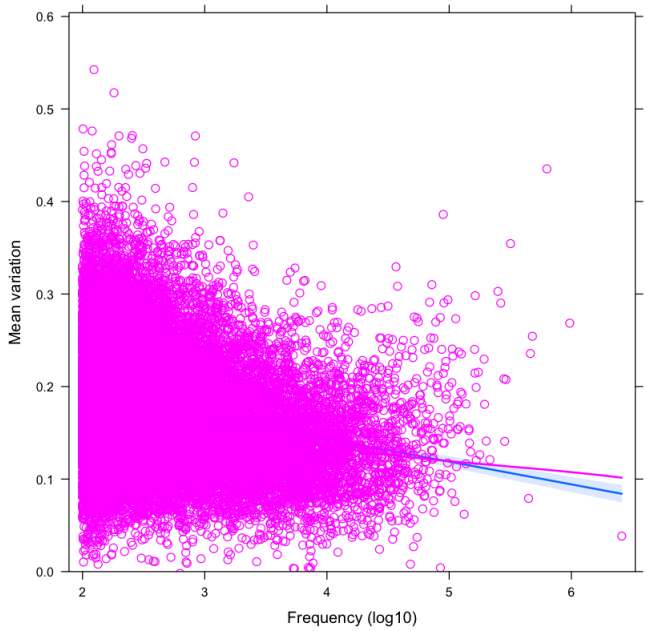


Figure E.4: Partial effects of frequency for a randomly sampled multi-linear regression model trained on PLOS.

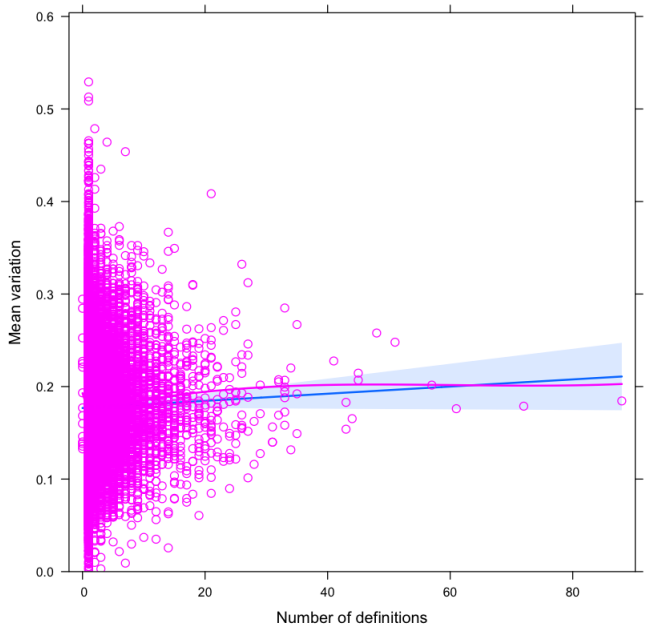


Figure E.5: Partial effects of polysemy for a randomly sampled multi-linear regression model trained on PLOS.

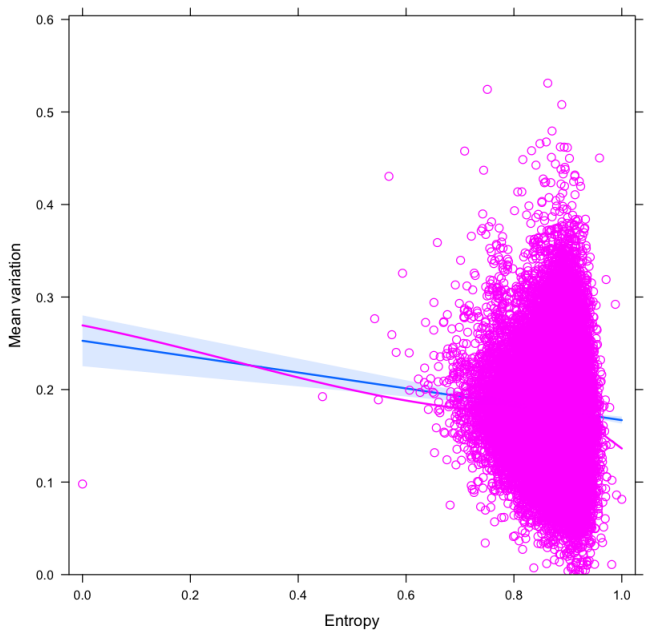


Figure E.6: Partial effects of the entropy for a randomly sampled multi-linear regression model trained on PLOS.

Appendix F

Partial effects for a randomly sampled model trained on the BNC

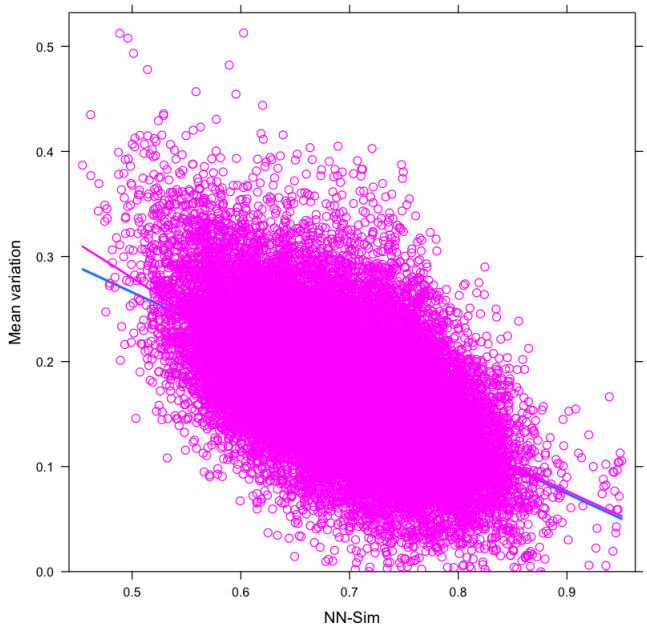


Figure F.1: Partial effects of nearest neighbor cosine similarity for a randomly sampled multi-linear regression model trained on the BNC.

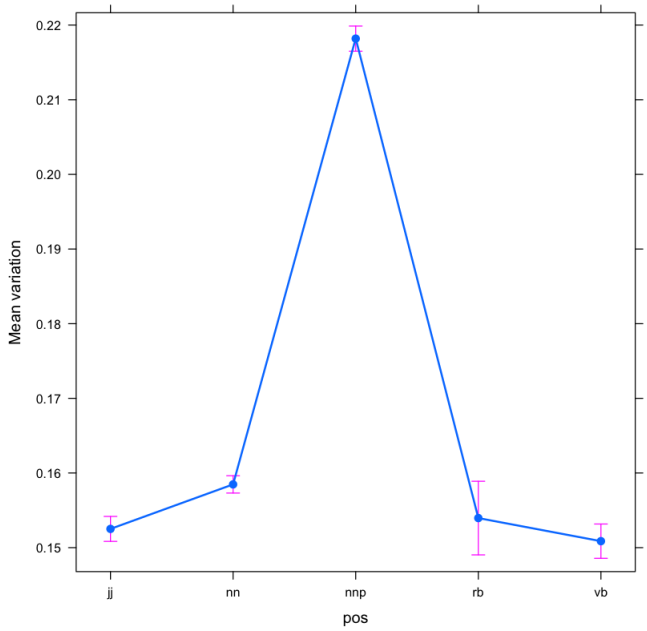


Figure F.2: Partial effects of the POS for a randomly sampled multi-linear regression model trained on the BNC.

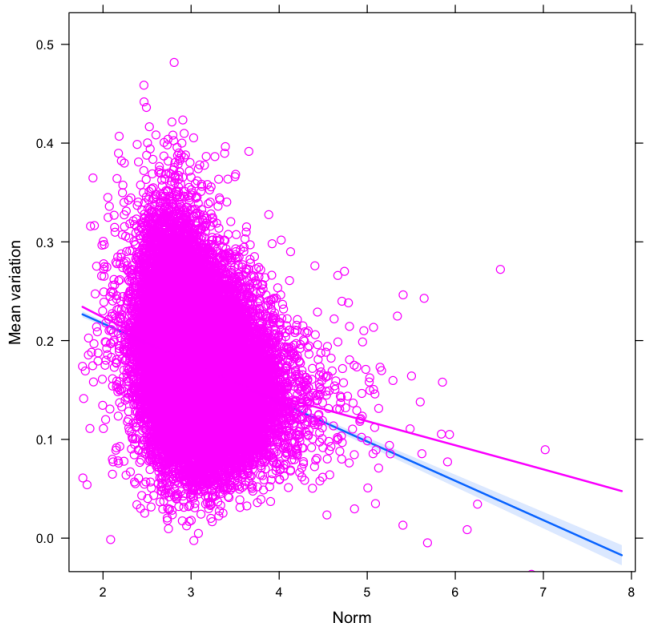


Figure F.3: Partial effects of the L2-norm for a randomly sampled multi-linear regression model trained on the BNC.

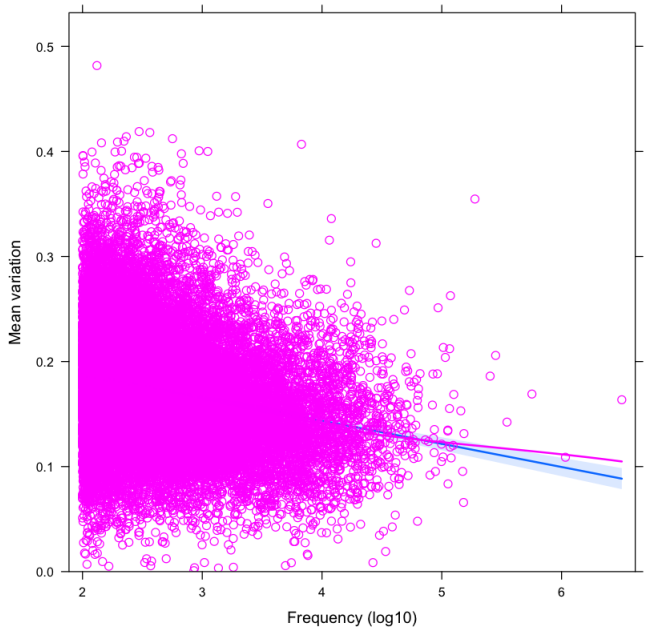


Figure F.4: Partial effects of frequency for a randomly sampled multi-linear regression model trained on the BNC.

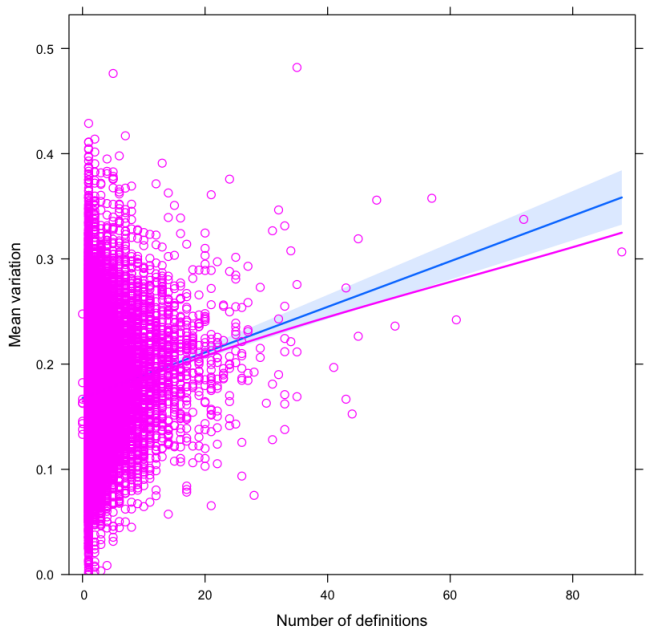


Figure F.5: Partial effects of polysemy for a randomly sampled multi-linear regression model trained on the BNC.

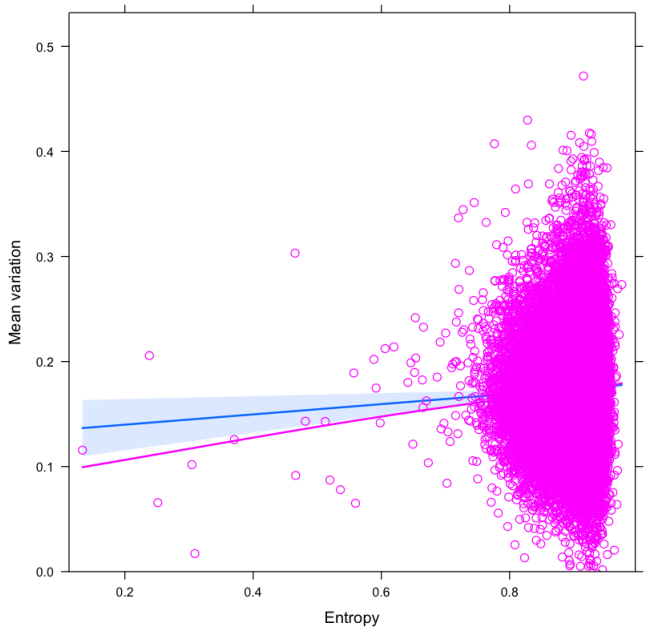


Figure F.6: Partial effects of the entropy for a randomly sampled multi-linear regression model trained on the BNC.