A dependency-based approach to word contextualization using compositional distributional semantics

Pablo Gamallo
Centro de Investigación en Tecnoloxías Intelixentes (CiTIUS)
University of Santiago de Compostela, Galiza
pablo.gamallo@usc.es

ABSTRACT

We propose a strategy to build the distributional meaning of sentences mainly based on two types of semantic objects: context vectors associated with content words and compositional operations driven by syntactic dependencies. The compositional operations of a syntactic dependency make use of two input vectors to build two new vectors representing the contextualized sense of the two related words. Given a sentence, the iterative application of dependencies results in as many contextualized vectors as content words the sentence contains. At the end of the contextualization process, we do not obtain a single compositional vector representing the semantic denotation of the whole sentence (or of the root word), but one contextualized vector for each constituent word of the sentence. Our method avoids the troublesome high-order tensor representations of approaches relying on category theory, by defining all words as first-order tensors (i.e. standard vectors). Some corpus-based experiments are performed to both evaluate the quality of the contextualized vectors built with our strategy, and to compare them to other approaches on distributional compositional semantics. The experiments show that our dependency-based method performs as (or even better than) the state-of-the-art.

Keywords:
distributional semantics,
compositionality,
dependency-based parsing

Journal of Language Modelling Vol 7, No 1 (2019), pp. 99–138

1 INTRODUCTION

Semantic compositionality is the crucial property of natural language according to which the meaning of a complex expression is a function of the meaning of its constituent parts and of the mode of their combination (Montague 1970). In the last decade, different distributional semantic models endowed with a compositional component have been proposed. The basic approach to composition (Mitchell and Lapata 2008, 2009, 2010) is to combine vectors of two syntactically related words with arithmetic operations: addition or component-wise multiplication. However, this approach is not fully compositional because the mode of combining the constituent parts is not considered. This way, two sentences with the same constituents but with different functions, e.g. *cats chase mice* and *mice chase cats*, are wrongly interepreted with the same flat vector combination.

To take into account the mode of combination, more recent distributional approaches (Coecke *et al.* 2010) follow a strategy aligned with the formal semantics perspective. Using the abstract mathematical framework of category theory, they provide the distributional models of meaning with the elegant mechanism expressed by the principle of compositionality, where words interact with each other according to their type-logical identities (Kartsaklis 2014; Baroni *et al.* 2014). The categorial-based approaches define arguments as vectors while functions taking arguments (e.g., verbs or adjectives that combine with nouns) are n-order tensors, with the number of arguments determining their order. Function application is the general composition operation. This is formalized as the tensor contraction which is nothing more than a generalization of matrix multiplication in higher dimensions.

Even if the type-logical compostional approach based on category theory is a very elegant proposal, it has, at least, four important drawbacks:

1. It results in an information scalability problem, since tensor representations grow exponentially (Kartsaklis *et al.* 2014). For instance, if noun meanings are encoded in vectors of 500 dimensions, adjectives, which are 2-order tensors, become matrices of 500^2 cells, while transitive verbs are described as tensors

with 500^3 dimensions. This situation leads to data sparseness problems, particularly for less common adjectives and verbs.

2. The use of tensor product for function application does not always perform as well as basic composition operations on vectors, such as component-wise multiplication (Mitchell and Lapata 2010).
3. The same word that occurs in different syntactic contexts is assigned different semantic types with incomparable representations (Paperno *et al.* 2014). For example, verbs like *eat* can be used in transitive or intransitive constructions (*children eat meat/children eat*), or in passive (*meat is eaten*). The different uses of the verb differ in the predicate arity and, then, are encoded in tensors of different orders. Since each of these tensors must be learned from examples individually, their obvious relation is missed. For each word, the creation of as many lexical entries as the number of its different syntactic uses is a drawback shared by all grammars based on the category theory.
4. The meaning of a sentence is a single representation and there is no access to the meaning of the constituents within the context of the whole sentence. For instance, let us observe the sense of the pronoun *They* in the sequence of sentences: *children eat meat. They are fat.* By co-reference, this pronoun is linked to *children* whose sense is contextualized by the fact that they are eaters of meat. However, there is no trivial mechanism to infer this specific sense of *children* from the meaning of the whole sentence.

Some approaches have tried to solve the issues described in the aforementioned four points. However, no strategy has been designed to deal with all of them together. For instance, the first issue has been addressed by the work reported in Paperno *et al.* (2014), where the representation size grows linearly, not exponentially, for higher semantic types, allowing for simpler and more efficient parameter estimation, storage, and computation. The third issue is at the center of the work described in Weir *et al.* (2016), where the meaning of a sentence is represented by the contextualized sense of its constituent words. The final point is addressed by Kruszewski and Baroni (2014), where the authors have observed that simpler and more economical models based on multiplication or addition yield better results than more complex ones.

These drawbacks have already been addressed by Socher *et al.* (2012) who proposed a strategy based on recursive neural networks, and by Paperno *et al.* (2014) whose proposal, *practical lexical function model*, represents each function word by a vector plus an ordered set of matrices enconding its arguments. We also address the four drawbacks by proposing a dependency-based framework with transparent vectors (and not embeddings as in Socher *et al.* (2012)). Moreover, the compositional model is different from that reported in Paperno *et al.* (2014), since we define all content words as unary-tensors (standard vectors), while syntactic dependencies are binary functions combining vectors in an iterative and incremental way. Take again the sentence *"children eat meat"*. The *subject* dependency builds two contextualized senses: the sense of *children* as nominal subject of *eat* and the sense of *eat* given *children* as subject. The two contextualized senses are vectors that can be involved in further dependencies. Then, the *direct object* dependency combines the previously contextualized sense of *eat* with the noun *meat* to build two new contextualized senses: a new contextualization of the sense of verb, on the one hand, and the sense of *meat* in the context of *"children eat"*, on the other. The intrepretation of the sentence is formalized as an incremental iteration giving rise to three contextualized senses. So, in this model, the meaning of a sentence is no more a single meaning, but one (contextualized) sense per content word, and each sense is represented by means of a word vector. In the previous example, dependencies have been applied iteratively from left-to-right: first the subject, and then the direct object. But they may also be applied from right-to-left: first the direct object and then the subject. The right-to-left iteration would result in slightly different contextualized senses. This way, the sense of *children* would be more specific since it would be built in the context of *"eat meat"*.

In our approach, syntactic dependencies are compositional functions that combine vectors to build the contextualized senses of words (still vectors) in an incremental way. While words are semantically represented as vectors, dependencies are compositional operations on them. It means that we operate with only two types of semantic objects: first-order tensors (or standard vectors) for content words, and binary functions for syntactic dependencies. This solves the scalability problem of high-order tensors (first drawback). In addition, it also prevents us from giving different categorical representations to verbs in different syntactic contexts. A verb is represented as a single vector which is contextualized as it is combined with its arguments (second drawback).

Concerning the compositional function, dependencies are operations that combine first-order vectors using simple arithmetic operations such as addition and multiplication, instead of more complex tensor products (third drawback). However, given that our vector space is enriched with syntactic information, the vectors built by composition cannot be a simple mixture of the input vectors as in the bag-of-words approaches (Mitchell and Lapata 2008). Our syntax-based vector representation of two related words encodes incompatible information and there is no direct way of combining the information encoded in their respective vectors. Vectors of content words (nouns, verbs, adjectives, and adverbs) are in different and incompatible spaces because they are constituted by different types of syntactic contexts. So, they cannot be merged. To combine them, on the basis of previous work (Thater *et al.* 2010; Erk and Padó 2008), we distinguish between direct denotation and selectional preferences (or indirect denotation) within a dependency relation.

The iterative application of the syntactic dependencies found in a sentence is actually the process of building the contextualized sense of all the content words constituting that sentence. So, the whole sentence is not assigned only one meaning – which could be the contextualized sense of the *root* word – but one sense per word, with the meaning of the root being only one such contentualized sense among many. This allows us to retrieve the contextualized sense of all constituent words within a sentence. The contextualized sense of any word might be required in further semantic processes, namely for dealing with coreference resolution involving anaphoric pronouns (fourth drawback).

The main contribution of our work is to propose a semantic space for Dependency Grammar, whose syntactic framework only consists of lexical units and dependencies (Kahane 2003; Hudson 2003). Our semantic model is wholly composed of binary operations (dependencies) and first-order vectors (words and selectional preferences). There is no room for semantic objects associated with composite expressions such as phrases or sentences. A sentence is interpreted as an iterative combination of word vectors with selectional preferences by using component-wise multiplication. This iterative and incremental compositional process may have two directions: from left-to-right and from right-to-left. These two directions result in slightly different contextualized words as we will show later in the experiments. Another important contribution of our work is that it should be seen as a continuation of Erk and Padó (2009) by allowing contextualized selectional preferences. Our approach was previously applied to other tasks: compositional translation (Gamallo and Pereira-Fariña 2017) and relational-based semantics (Gamallo 2017b). The current article is an extension of a previous conference work (Gamallo 2017c).

This article is organized as follows. In Section 2, our dependency-based compositional model is described. In Section 3, corpus-based experiments are performed to build and evaluate the quality of compositional/contextualized vectors. Then, in Section 4, several distributional compositional approaches are introduced and discussed. Finally, relevant conclusions are addressed in Section 5.

2 THE COMPOSITIONAL MODEL

We first give a quick overview of our vector space (Section 2.1), which is followed by a technical description of the compositional operations driven by syntactic dependencies (Section 2.2). We conclude by applying an incremental interpretation approach to our model (Section 2.3).

2.1 *Dependency-based vector representation*

Distributional Semantics associates the meaning of a word with the set of contexts in which it occurs (Firth 1957). Typically, in computational approaches, the distributional representation for a word is computed from the occurrences of that word in a given corpus (Grefenstette 1995). In distributional semantics models, each word is defined as a context vector, and each position in the vector represents a specific context of the word whose value is the frequency (or some statistical weight) of the word in that context. According to recent research, a vector space can be considered as a semantic model, since vector-based representations (i.e. distributional features) may be defined as extensions of logical expressions if they are seen as *ideal distributions* (Copestake and Herbelot 2012; Erk 2013).

Our model employs vector representations for words (or lemmas) based on syntactic contexts. Syntactic contexts are derived from binary dependencies, which can be found in a corpus analyzed with a dependency-based parser. Let's suppose the composite expression *a horse is running* was found in a corpus and is analyzed as the following syntactic dependency:

$$(nsubj, run, horse)$$

It states that the noun *horse* (dependent word) is related to the head verb *run* by means of the relation *nsubj* (nominal subject). A dependency is then a triple consisting of a relation, a head, and a dependent word. From this dependency, we can identify two complementary word contexts:

$$<nsubj_{\uparrow}, run>, <nsubj_{\downarrow}, horse>$$

Then, we count co-occurrences between words and contexts. In this case, the context $<nsubj_{\uparrow}, run>$ is assigned frequency 1 within the vector of *horse,* while we add a new occurrence to $<nsubj_{\downarrow}, horse>$ within the vector of *run*. The up arrow in $nsubj_{\uparrow}$ means that the head word *run* in the subject relation is expecting a dependent word, while the down arrow in $nsubj_{\downarrow}$ means that the dependent noun *horse* is searching for the head verb. This representation is inspired by Gamallo *et al.* (2005) and is similar to that used for distinguishing traditional selectional preferences from *inverse* selectional preferences (Erk and Padó 2008). To reduce the number of contexts, we apply a technique to filter out contexts by relevance. The filtering strategy to select the most relevant contexts consists in selecting, for each word, the R (relevant) contexts with highest log-likelihood measure. The top R contexts are considered to be the most *relevant* and informative for each word. R is a global, arbitrarily defined constant whose usual values range from 10 to 1000 (Biemann and Riedl 2013; Padró *et al.* 2014). In short, we keep at most the R most relevant contexts for each target word (where $R = 500$ in our experiments). This is an explicit and transparent representation giving rise to a non-zero matrix.

2.2 Vector composition

In our approach, composition is modeled by two semantic functions, *head* and *dependent,* that take three arguments each:

(1) $$head_{\uparrow}(r, \vec{x}, \vec{y}^{\circ})$$

(2) $$dep_{\downarrow}(r, \vec{x}^{\circ}, \vec{y})$$

where *head*$_{\uparrow}$ and *dep*$_{\downarrow}$ represent the head and dependent functions, respectively, r is the name of the relation (*nsubj*, *dobj*, *nmod*, etc.), and $\vec{x}$, $\vec{x}^{\circ}$, $\vec{y}$, and $\vec{y}^{\circ}$ stand for vector variables. On the one hand, $\vec{x}$ and $\vec{y}$ represent the denotation of the head and dependent words, respectively. They represent standard context distributions which we call *direct vectors*. On the other hand, $\vec{x}^{\circ}$ represents the selectional preferences imposed by the head, while $\vec{y}^{\circ}$ stands for the selectional preferences imposed by the dependent word. Selectional preferences are also called *indirect vectors* and the way we build them is described below.

Consider now a specific dependency relation, nominal subject (*nsubj*), and two specific words: *horse* and *run*. The application of the two functions consists of multiplying the direct and indirect vectors by taking into account the *nsubj* relation:

(3) $$head_{\uparrow}(nsubj, \vec{run}, \vec{horse}^{\circ}) = \vec{run} \odot \vec{horse}^{\circ} = \vec{run}_{nsubj\uparrow}$$

(4) $$dep_{\downarrow}(nsubj, \vec{run}^{\circ}, \vec{horse}) = \vec{horse} \odot \vec{run}^{\circ} = \vec{horse}_{nsubj\downarrow}$$

Each multiplicative operation results in a compositional vector which represents the contextualized sense of one of the two words (either the head or the dependent). Component-wise multiplication has an intersective effect: the selectional preferences restricts the direct vector by assigning frequency 0 to those contexts that are not shared by both vectors. Here, $\vec{horse}^{\circ}$ and $\vec{run}^{\circ}$ are indirect vectors resulting from the following vector additions:

(5) $$\vec{horse}^{\circ} = \sum_{\vec{w} \in \boldsymbol{H}} \vec{w}$$

(6) $$\vec{run}^{\circ} = \sum_{\vec{w} \in \boldsymbol{R}} \vec{w}$$

where $\boldsymbol{H}$ is the vector set of those verbs having *horse* as subject (except the verb *run*). More precisely, given the linguistic context <*nsubj*$_{\downarrow}$, *horse*>, the indirect vector $\vec{horse}^{\circ}$ is obtained by adding the vectors $\{\vec{w} | \vec{w} \in \boldsymbol{H}\}$ of those verbs (*eat*, *jump*, etc.) that are combined with the noun *horse* in that syntactic context. Component-wise addition of vectors has an union effect. In more intuitive terms, $\vec{horse}^{\circ}$ stands for the inverse selectional preferences imposed by *horse* on any verb at the subject position. As this new vector consists of verbal contexts, it lives in the same vector space as verbs and, therefore, it can be combined with the direct vector of *run*.

	$\vec{red}$	$\vec{white}$	$\vec{vague}$	$\vec{car}^{\circ}$	$\vec{red} \odot \vec{car}^{\circ}$
$<amod_{\uparrow}, car>$	5	2	0	2	10
$<amod_{\uparrow}, pencil>$	2	0	0	0	0
$<amod_{\uparrow}, idea>$	1	0	7	0	0
$<amod_{\uparrow}, book>$	2	1	2	1	2

Table 1:
Deriving the vector of *red* in *red car* by dependency-based compositionality (dependent function)

	$\vec{run}$	$\vec{eat}$	$\vec{sleep}$	$\vec{horse}^{\circ}$	$\vec{horse}^{\circ} \odot \vec{run}$
$<nsubj_{\downarrow}, horse>$	3	5	1	6	18
$<dobj_{\downarrow}, program>$	5	0	0	0	0
$<prep\_in_{\downarrow}, prairie>$	2	1	1	2	4
$<prep\_with_{\downarrow}, gas>$	3	0	0	0	0

Table 2:
Deriving the vector of *run* in *horses run* by dependency-based compositionality (head function)

On the other hand, $\boldsymbol{R}$ in Equation 6 represents the vector set of nouns occurring as subjects of *run* (except the noun *horse*). Given the lexico-syntactic context $<nsubj_{\uparrow}, run>$, the vector $\vec{run}^{\circ}$ is obtained by adding the vectors $\{\vec{w} | \vec{w} \in \boldsymbol{R}\}$ of those nouns (e.g. *dog*, *car*, *computer*, etc.) that might be at the subject position of the verb *run*. Indirect vector $\vec{run}^{\circ}$ stands for the selectional preferences imposed by the verb on any noun at the subject position. It is constituted by nominal contexts and, therefore, is compatible with the direct vector of *horse*.

Tables 1 and 2 are toy examples showing how to construct the compositional vectors of two contextualized words: *red* in *red car* (Table 1) and *run* in *horses run* (Table 2). Vectors are in columns and rows are dependency-based contexts. Each vector position is filled with the frequency of the word in the corresponding context. In the two tables, we represent three direct vectors, one indirect vector (derived from the direct vectors) and the compositional vector (last column). In this toy example, words are hypothetical four-dimensional vectors; whereas in real scenarios extracted from large corpora, vectors may have hundreds of thousands of dimensions.

In Table 1, the indirect vector $\vec{car}^{\circ}$, associated to the noun *car* given *red* as modifier, is obtained by adding the vectors of those adjectives that are also modifiers of *car* (except *red*). In this toy example, only the direct vector of *white* fulfills such conditions. In Table 2, the indirect vector $\vec{horse}^{\circ}$ is the result of adding the direct vectors of *eat* and *sleep*, since *horse* also occurs as subject of these verbs.

It is worth noticing that the contextualized vector of *red* within *red car* (last column in Table 1) has fewer contexts with positive values than the direct vector of the polysemous adjective *red* (out of context). The (inverse) selectional preferences imposed by *car* are able to select a more compact and less ambiguous vector of the adjective. This way, the context activating the ideological sense of *red* ($<\mathit{amod}_{\uparrow},\mathit{idea}>$) is filtered out as it is multiplied by 0. Similarly, the resulting vector of *run* within *horses run* has fewer positive contexts and then tends to be less ambiguous than the direct vector of the polysemous verb *run* out of context. In Table 2, the contexts ($<\mathit{prep\_with}_{\uparrow},\mathit{gas}>$, $<\mathit{dobj}_{\uparrow},\mathit{program}>$), which hardly appears with words denoting animals, are removed (frequency 0) from the new contextualized vector of *run*. So, the inverse selectional preferences imposed by *horse* activate one specific sense of the verb: physical movement. Notice that we do not consider prepositions as content words, but as syntactic dependencies.

In approaches to computational semantics inspired by Combinatory Categorial Grammar (Steedman 1996) and Montagovian semantics (Montague 1970), the interpretation process activated by composite expressions such as *dogs chase cats*, *horses run* or *red car* relies on rigid function-argument structures. Relational expressions like verbs and adjectives are used as predicates while nouns and nominals are their arguments. In the composition process, each word is supposed to play a rigid and fixed role: the relational word is semantically represented as a selective function imposing constraints on the denotations of the words it combines with, while non-relational words are in turn seen as arguments filling the constraints imposed by the function. For instance, *run* and *red* denote functions while *horses* and *car* are their respective arguments.

By contrast, we do not define verbs and adjectives as functional artifacts driving the compositional process. In our compositional approach, dependencies are the active functions that control and rule the selectional requirements imposed by the two related words. Dependencies, instead of relational words, are then conceived of as the main functional operations taking part in composition. This way, two syntactically dependent expressions are no longer interpreted as a rigid "predicate-argument" structure, where the predicate is the active function imposing the semantic preferences on a passive argument, which matches such preferences. On the contrary, each constituent word imposes its selectional preferences on the other. This is in accordance with non-standard linguistic research which assumes that the words involved in a composite expression impose semantic restrictions on each other (Pustejovsky 1995; Gamallo 2008; Gamallo *et al.* 2005). Not only verbs or adjectives are taken as predicates selecting different types of nouns, but so too do nouns select for different types of verbs and adjectives. Following this idea, we propose a co-compositional approach: in the head function, the dependent element imposes its restrictions on the head denotation, and the output is a more specific and less ambiguous denotation of the head. By contrast, in the dependent function, it is the head that imposes its selectional restrictions on the dependent denotation to produce a more elaborate and less ambiguous denotation of the dependent expression.

It means that the semantic space consists of just two types of entities: word vectors and dependency-based functions. Vectors represent both word senses (direct vectors) and selectional preferences (indirect vectors), while head/dependent functions represent compositional operations. A dependency-based function takes as arguments a relation and a pair of vectors (direct + indirect), and returns a more elaborate direct vector.

2.3 *Dependencies and incremental interpretation*

Frameworks such as Discourse Representation Theory (Kamp and Reyle 1993) and Situation Semantics (Barwise 1987) make two basic assumptions about interpretation: that the meaning of a sentence is dependent of the meaning of the previous sentence in the discourse; and that a sentence modifies in turn the meaning of the following sentence. Sentence meaning is not isolated from discursive unfolding; rather, meaning is incrementally constructed at the same time as discourse information is processed.

We assume that incrementality is true not only at the inter-sentence level but also at the inter-word level, i.e., between dependent words. In order for a sentence-level interpretation to be attained, dependencies must be established between individual constituents as soon as possible. This claim is assumed by a great variety of research (Kempson *et al.* 2001, 1997; Milward 1992; Costa *et al.* 2001; Schlesewsky and Bornkessel 2004). The incremental hypothesis states that information is built up on a left-to-right word-by-word basis in the interpretation process (Kempson *et al.* 2001). The meaning of an utterance is progressively built up as the words come in. The sense of a word is provided as part of the context for processing each subsequent word. Incremental processing assumes that humans interpret language without reaching the end of the input sentence; that is, they are able to assign a sense to the initial left fragment of an utterance. This hypothesis has received a large experimental support in the psycholinguistic community over the years (McRae *et al.* 1997; Tanenhaus and Carlson 1989; Truswell *et al.* 1994).

For instance, to interpret *the cat chased a mouse*, it is required to interpret *cat chased* as a fragment that restricts the type of nouns that can appear at the direct object position: *mouse*, *rat*, *bird*, etc.[1] In the same way *police chases* restricts the entities that can be chased by police officers: *thieves, robbers,* and so on. However, a left-to-right interpretation process cannot be easily assumed by a standard compositional approach. In a Montagovian model, *chase* is a transitive verb denoting the binary function $\lambda x \lambda y\ chase(x, y)$, *chased a mouse* is an intransitive verb denoting a unary predicate, while *the cat chased a mouse* is a sentence denoting a truth value. The standard compositional model does not provide any interpretation for *the cat chased* within the sentence *the cat chased a mouse*; consequently, it is unable to predict how the expression *the cat chased* restricts the type of nouns appearing at the direct object position.

By contrast, in our left-to-right incremental compositional strategy, *the cat chased* is a grammatical expression referring to two semantic objects: the compositional vectors of the two related lexical units.

In our approach, the iterative application of the syntactic dependencies found in a sentence is actually the recursive process of building the contextualized sense of all the content words which constitute the sentence. Thus, the whole sentence is not assigned only one meaning (which could be the contextualized sense of the *root* word), but one

[1] We do not consider the compositional meaning of determiners, auxiliary verbs, or tense affixes. Quantificational issues associated with them are beyond the scope of this work. An interesting work on determiners in compositional distributional semantics is reported by Bernardi *et al.* (2013).

sense per lemma, where the sense of the root is only one such sense considered.

This recursive and incremental process may have two directions: from left-to-right and from right-to-left.

The incremental left-to-right interpretation of *the cat chased a mouse* is illustrated in Equation 7 (without considering the meaning of determiners nor verbal tense):

$$\begin{aligned} head_{\uparrow}(nsubj, \vec{chase}, \vec{cat}^{\circ}) &= \vec{chase}_{nsubj\uparrow} \\ dep_{\downarrow}(nsubj, \vec{chase}^{\circ}, \vec{cat}) &= \vec{cat}_{nsubj\downarrow} \\ head_{\uparrow}(dobj, \vec{chase}_{nsubj\uparrow}, \vec{mouse}^{\circ}) &= \vec{chase}_{nsubj\uparrow+dobj\uparrow} \\ dep_{\downarrow}(dobj, \vec{chase}^{\circ}_{nsubj\downarrow}, \vec{mouse}) &= \vec{mouse}_{nsubj\downarrow+dobj\downarrow} \end{aligned} \tag{7}$$

First, the head and dependent functions are applied on the subject dependency *nsubj* to build the compositional vectors $\vec{chase}_{nsubj\uparrow}$ and $\vec{cat}_{nsubj\downarrow}$. Then, the head function is applied *dobj* to produce a more elaborate chasing event, $\vec{chase}_{nsubj\uparrow+dobj\uparrow}$, which stands for the full contextualized sense of the root verb. In addition, the dependent function takes *dobj* to yield a new nominal vector, $\vec{mouse}_{nsubj\downarrow+dobj\downarrow}$, whose internal information only can refer to a specific animal: *mouse chased by the cat.* In the context of a chasing event, *mouse* does not refer to a computer's device.

The contextualized selectional preferences, $\vec{chase}^{\circ}_{nsubj\downarrow}$, represent an indirect vector obtained as follows:

$$\vec{chase}^{\circ}_{nsubj\downarrow} = \vec{cat}_{nsubj\downarrow} \odot \sum_{\vec{w} \in \boldsymbol{C}} \vec{w} \tag{8}$$

where $\boldsymbol{C}$ is the vector set of those nouns that are in the direct object role of *chase* (except the noun *mouse*). The new vector resulting by adding the vectors of $\boldsymbol{C}$ is combined by multiplication (intersection) with the contextualized dependent vector, $\vec{cat}_{nsubj\downarrow}$, to build the contextualized selectional preferences. In more intuitive terms, the selectional preferences built in Equation 8 are constituted by selecting the contexts of the nouns appearing as direct object of *chase*, which are also part of *cat* after having been contextualized by the verb at the subject position.

The dependency-by-dependency functional application results in three contextualized word senses: $\vec{cat}_{nsubj\downarrow}$, $\vec{chase}_{nsubj\uparrow+dobj\uparrow}$ and

$\vec{mouse}_{nsubj\downarrow+dobj\downarrow}$. They all together represent the meaning of the sentence in the left-to-right direction. Notice that $\vec{cat}_{nsubj\downarrow}$ is not a fully contextualized vector: it was only contextualized by the verb, but not by the direct object noun. In order to fully contextualize the subject, we need to initialize the composition process in the other way around.

In the opposite direction, from right-to-left, the incremental process starts with the direct object dependency:

$$\begin{aligned}
head_{\uparrow}(dobj, \vec{chase}, \vec{mouse}^{\circ}) &= \vec{chase}_{dobj\uparrow} \\
dep_{\downarrow}(dobj, \vec{chase}^{\circ}, \vec{mouse}) &= \vec{mouse}_{dobj\downarrow} \\
head_{\uparrow}(nsubj, \vec{chase}_{dobj\uparrow}, \vec{cat}^{\circ}) &= \vec{chase}_{dobj\uparrow+nsubj\uparrow} \\
dep_{\downarrow}(nsubj, \vec{chase}^{\circ}_{dobj\downarrow}, \vec{cat}) &= \vec{cat}_{dobj\downarrow+nsubj\downarrow}
\end{aligned} \tag{9}$$

In Equation 9, the verb *chase* is first restricted by *mouse* at the direct object position, and then by its subject *cat*. In addition, this noun is restricted by the vector $\vec{chase}^{\circ}_{dobj\downarrow}$, which represents the contextualized selectional preferences built by combining $\vec{mouse}_{dobj\downarrow}$ with the vectors of the nouns that are in the subject position of *chase* (except *cat*). This new compositional vector represents a very contextualized nominal concept: *the cat that chased a mouse*. The word *cat* and its specific sense can be related to anaphorical expressions by making use of co-referential relationships at the discourse level: e.g., pronoun *it*, other definite expressions (*that cat, the cat*, ...), and so on. Notice that this compositional vector might also be used to represent the contextualized sense of a nominal restricted by a relative clause. For this type of construction, it is worth paying special attention to the work reported in Sadrzadeh *et al.* (2013), where the authors describe a tensor-based method to represent the compositional meaning of relative pronouns.

The meaning of a sentence is ideally represented by the full contextualization of its constituent words. Yet, as has been said, not all words in a sentence can be fully contextualized using left-to-right combination. For instance, to fully contextualize the noun subject $\vec{cat}_{dobj\downarrow+nsubj\downarrow}$ within the subject-verb-object sentence *the cat chased a mouse*, the iterative process must follow the right-to-left direction: first, the noun vector $\vec{mouse}$ is combined with chasing preferences on the object ($\vec{chase}^{\circ}$). Then, the resulting vector of the previous combination is used to generate the restricted verb preferences on the subject ($\vec{chase}^{\circ}_{dobj\downarrow}$), which are combined with the noun vector $\vec{cat}$ to eventually return the fully contextualized vector of the subject noun. As in standard compositional approaches, vectors are combined with pointwise multiplication. The main difference with regard to standard vector combination is that our compositional strategy also relies on vectors representing selection preferences. Both selection preferences and compositional (contextualized) vectors are generated dynamically during word combination.

The order of function application is flexible since it is not constrained by the type of dependencies or by the arity of function words (mainly verbs). A particular order may be applied by principles or constraints that are independent of the syntactic structure. The constraints that specify a particular order may be defined by external factors. For instance, if the objective is to simulate a psycholinguistic notion of incrementality, where the meaning of words is gradually elaborated as they are syntactically integrated into new dependencies, then the best option is to implement the left-to-right algorithm. However, nothing prevent us implementing the complementary right-to-left direction in order to compare the contextualized senses generated by using both directions (as we will show later in the experimental section). Instead of applying all possible orders, which has high computational cost, it would be possible to apply external constraints and principles to impose a very restricted order. One of these contraints might be, for instance, to consider the degree of ambiguity of lexical units: we could apply first those dependencies containing less ambiguous words with more semantically homogenous vectors; and then use these in a subsequent step to disambiguate more heteregenous word vectors (i.e., more ambiguous ones) (Gamallo 2008).

In the sentence *the coach drives the team*, this contraint should lead us to interpret *drives the team* before *the coach drives*, since *team* is less ambiguous than *coach*. By contrast, in *the team hired a coach*, the order should be the other way around following the same principle. In a more complex sentence such as *I lost my computer mouse*, the same principle would force us to interpret first the less ambiguous noun-noun dependency between *computer* and *mouse* before the more ambiguous relation between *lost* and *mouse*. This ambiguity-based constraint may be seen as a general procedure to word sense discrimination. Yet, the definition and implementation of specific contraints and principles restricting function application is beyond the scope of the current work.

Finally, it is worth noticing that the compositional objects we build using dependencies are not flat representations such as those derived from typical dependency-based analysis. The order of functional application is meaningful and allows us to build vectors at different constituency levels in terms of immediate constituent analysis. A criticism of dependency analysis is that it is not able to deal with the different interpretations obtained from expressions like *fastest American runner* and *American fastest runner*. As both expressions are analyzed with the same flat dependency-based structure (*fastest* and *American* are dependent of *runner*), it would not be possible to derive different semantic entailments from the same syntactic analysis. However, in our dependency-based model, the order in which the functions are applied allows us to build several compositional entities, which simulate the construction of different constituent units.

3 EXPERIMENTS

We have designed and developed a system, *DepFunc*, based on the method described in the previous section. Although the method can potentially be applied to any sentence, regardless of its syntactic structure, the limitations of the implementation and the complexity of the task have led us to apply it only to expressions with a predetermined and fixed structure: adjective-noun, noun-verb, and noun-verb-noun.

Two different types of experiments were carried out to evaluate the performance of our system. The specific objective of the first experiment (Section 3.1) is to compare the distributional similarity between compositional vectors of composite expressions and corpus-observed vectors of the same composite expressions. If they are similar, our vectors predicted by compositionality can be considered correct because they are close to standard vectors built with observed data. We compared our strategy with the one defined in Baroni and Zamparelli (2010), which also carried out a similar evaluation. Experiments were made with ADJ-NOUN (to abbreviate: AN) and NOUN-VERB expressions (to abbreviate: NV).

In the second type of experiments (Section 3.2), we use test datasets to measure the correlation between human similarity judgements and similarity coefficients computed with our compositional expressions. In Subsection 3.2.2, we measure the quality of compositional vectors built from NV composite expressions, using as gold standard the test dataset provided by Mitchell and Lapata and described in (Mitchell and Lapata 2008). In Subsection 3.2.3, we check the quality of more complex composite expressions, namely NOUN-VERB-NOUN constructions (NVN) incrementally composed with *nsubj* and *dobj* dependencies.

3.1 *Compositional and corpus-observed vectors*

As in Baroni and Zamparelli (2010), the experiment consists in comparing the distributional similarity between two different types of vectors associated with composite expressions: *compositional vectors* and *corpus-observed vectors*. Compositional vectors are those built following the compositional method described in the previous section. They are thus model-generated vectors constructed according to the corpus-based observed frequencies of their constituents. Corpus-observed vectors of composite expressions are constructed with the frequencies associated with the whole expression. They are called *holistic* vectors by Turney (2013). We should expect that the compositional and the holistic vectors built for the same composite expression should be similar (Baroni and Zamparelli 2010). More precisely, we expect that the predicted distribution computed by our compositional approach should yield similar vectors to those built with real distributions calculated from real-world corpora. For instance, if we build a compositional vector for *red car* according to the frequency of its parts in a compositional way, the resulting vector should be similar to the vector constructed by just observing the co-occurrences of the composite expression as a whole. Notice that there are exceptions to that, namely those cases where the meaning of the compound expression is not compositional (e.g., collocations, frozen expressions, idioms, and so on).

3.1.1 Corpus and distributional models

In order to build the compositional and holistic (corpus-observed) vectors, we made two partitions from the English Wikipedia (dump file of November 2015), with 100M tokens each. The first partition was used to build the compositional vectors (and to train learning models) while the second partition was used for extracting corpus-observed vectors as well as for testing and evaluation. Word vectors were built by computing their co-occurrences in lexico-syntactic contexts. We used the dependency parser DepPattern (Gamallo and Garcia 2018; Gamallo 2015) to perform syntactic analysis. Three different types of vectors were built from the corpus: nominal, verbal, and adjectival vectors. Then, for each word we filtered out irrelevant contexts using simple count-based techniques inspired by those described in Gamallo (2017a), where matrices are stored in hash tables with only non-zero values. More precisely, the association between words and their contexts were weighted with the Dunning's likelihood ratio (Dunning 1993) and then, for each word, only the N contexts with highest likelihood scores were stored in the hash table (where $N = 500$). So, the remaining contexts were removed from the hash (whereas in standard vector/matrix representations, instead of removing contexts we should assign them zero values). This filtering-based approach turned out to be more efficient than other strategies based on dimensionality reduction such as Singular Value Decomposition (Gamallo and Bordag 2011). In addition, our approach requires explicit vector spaces, which are more linguistically transparent than dense representations such as neural-based word embeddings.

Not all words were selected for computing similarity; in particular, we selected those nouns, verbs, and adjectives occurring in more than 100 different contexts. The experiments were made with lemmas.

Experiments were performed with two types of composite units: AN expressions in the nominal space and NV in the verbal space. Our specific task consists of selecting a list of both AN and NV composites, building their compositional and corpus-observed vectors, and checking whether each particular compositional vector is similar to its corresponding corpus-observed vector. To avoid possible bias between predicted and observed occurrences, corpus-observed vectors were derived from the second partition of the corpus, while compositional vectors were built from the first partition. To build compositional vectors, the strategy defined in the previous section was implemented in Perl giving rise to the software *DepFunc*.

3.1.2 Evaluation

The list of target composites for evaluation was created as follows. In the second partition with 100M tokens, we selected the composites with more than 50 different contexts: 6676 ANs and 3004 NVs. Then, we filtered out those composites with at least one constituent word which does not appear in the matrices created from the first corpus partition (since these constituent words had fewer than 100 lexico-syntactic contexts in the first partition). Finally, we obtained a test list of 1,841 ANs and another of 767 NVs, which were subsequently manually revised in order to filter out ill-formed expressions. We obtained more AN composites than those of NVs because the nominal space has a higher number of entities and lexico-syntactic contexts than the verbal space.

Then, we built, on the one hand, the compositional vectors of the selected 1,841 ANs and 767 NVs using the first corpus partition and, on the other hand, the corpus-observed vectors of the same composites using the quantitative information of the second partition. The new vectors are added to both the nominal and verbal matrices. In total, the nominal matrix contains 22,025 single nouns and $1{,}841 \times 2$ AN composites, while the verbal matrix consists of 5,131 single verbs and 767×2 NV composites. Next, all possible pairs were generated and cosine similarity was computed in each matrix. For each corpus-observed composite, we created a ranked list of the N most similar expressions, and finally, we verified whether its corresponding compositional composite is found in the list and recorded its ranking.

We define *hit* to mean an instance of finding the compositional vector of a composite expression in the ranked list of its corresponding corpus-observed vector. For instance, if the compositional vector of *"red car"* is in the top N list of similar candidates of the corpus-observed vector associated to the same expression, we count one *hit*.

To compare our model with a state-of-the-art system, we used the software DISSECT (Dinu *et al.* 2013a)[2] The software was used to train and apply the compositional functions described in Baroni and Zamparelli (2010), taking as input the first (part-of-speech tagged) corpus partition and the lists of test composites introduced above. The training process was performed by selecting all the adjectives and verbs of the test list and all their occurrences with those nominal arguments

[2] `http://clic.cimec.unitn.it/composes/toolkit/introduction.html`

that are not in the test list. To compute word-context co-occurrences, we defined the contexts of a word as the bag-of-lemmas extracted from a window of size 5 (two context words both to the left and to the right of the target word). Co-occurrence matrices were reduced to 300 dimensions by making use of Singular Value Decomposition. Similarity between vectors was computed with the cosine measure. The function we have used for training the model is *LexFunc* (Lexical Function), which gave rise to the best results in the experiments described in Baroni and Zamparelli (2010) and Dinu *et al.* (2013b).

The final results are shown in Tables 3 and 4. Each system is evaluated with regard to different values of K: 10, 50 and 100. For each value, we count the proportion of hits to compute precision at K, noted P@K. For instance P@10 is the number of hits within the 10 most similar expressions divided by the total number of evaluated expressions. The other measure, *ranking_average*, stands for the average of the ranking positions of all hits within the ranked list with the 100 most similar expressions. For instance, if 3 hits were found in rankings 25, 50, and 75, the *ranking_average* is 50. This evaluation is inspired by standard information retrieval metrics.

Four strategies are compared: what we call *lower-bound* is just the by chance probability of finding hits at each K level. The hits found at $K = 100$ tend to occur at position 50 on average. The *baseline* strategy consists in associating the compositional vector to the head vector. For instance, the baseline compositional vector of *"red car"* would be the vector of the head noun *car*, while the baseline compositional vector of *"horses run"* would be the vector of the head verb *run*. This is a very reliable and sound baseline because there is a straight semantic relationship between any composite expression and its head: the concept designated by the head tends to be the direct hypernym of the concept designated by the composite expression. So, *"red car"* (hyponym) must be closely related to *car* (hypernym). In the experiments described by Baroni and Zamparelli (2010), this baseline was the third best strategy out of six evaluated systems, outperforming the approaches introduced by Mitchell and Lapata (2009) and Guevara (2010). The system denoted *LexFunc* represents the best compositional system, known as *alm* and based on the *Lexfunc* model, described in

Baroni and Zamparelli (2010).[3] These two systems are compared with our compositional approach: *DepFunc (head)*. It is worth noting that, in these experiments, the evaluation is just focused on the contextualized heads of compositional vectors. The reason for this is that the syntactic contexts of holistic expressions are found in the space of the heads: AN expressions are nouns and NVs are verbs. So, in order to compare compositional with holistic expressions, we have to consider that compositional ANs are contextualized nouns and compositional NVs are contextualized verbs.

system	*P@10*	*P@50*	*P@100*	*ranking_average*
lower-bound	0%	0.2%	0.4%	50
baseline (noun)	11.74%	31.36%	42.95%	33.90
DepFunc (head)	**36.39**%	**53.01**%	**60.32**%	**17.69**
LexFunc	21.36%	35.79%	42.87%	22.43

Table 3:
Percentages of hits (precision at 10, 50 and 100) and ranking average in the ranked lists of AN expressions

system	*P@10*	*P@50*	*P@100*	*ranking_average*
random	0.1%	0.7%	1.5%	50
baseline (verb)	6.21%	23.16%	35.02%	37.74
DepFunc (head)	17.51%	**37.85**%	**45.76**%	25.64
LexFunc	**24.54**%	35.24%	39.81%	**24.23**

Table 4:
Percentages of hits (precision at 10, 50 and 100) and ranking average in the ranked lists of NV expressions

Tables 3 and 4 show the results of AN and NV expressions. Our compositional approach, DepFunc (head), clearly outperforms the baseline strategies for both AN and NV. In addition, it also outperforms LexFunc for AN. However, the differences between LexFunc and DepFunc are not so sharp for NV. In fact, DepFunc finds more hits within larger ranked lists (50 and 100), but those found by LexFunc are in better ranks, being even more precise at $K = 10$.

The main problem of this evaluation is that it does not allow us to take advantage of the contextualization of the dependent word. This will be solved in the following experiments.

[3] Additive and multiplicative models implemented in DISSECT were also evaluated, but the results obtained were below the baseline.

3.2 *Correlation with human judgements*

In the following experiments, we compare the similarity between pairs of compositional vectors representing composite expressions with the similarity given by annotators to those expressions. In this case, we will compare all contextualized words of the expressions instead of only considering the word heads.

3.2.1 Corpus and distributional models

In these experiments, our working corpus consists of both the English Wikipedia (dump file of November 2015[4]) and the British National Corpus (BNC)[5]. In total, the corpus contains about 2.5 billion word tokens, which were analysed with DepPattern.

Word vectors were built by computing their co-occurrences in syntactic contexts. Two different types of vectors were built from the corpus: nominal and verbal vectors. Distributional matrices were built using the same strategy as the one defined for the previous experiment.

This process of matrix reduction resulted in the selection of 330 953 nouns (most of them proper names) with 236,708 different nominal contexts; and 6,618 verbs with 140,695 different verbal contexts. As the contexts of nouns and verbs are not compatible, we created two different vector spaces. Words and their contexts were stored in two hashes, one per vector space, which represent matrices containing only non-zero values. Cosine similarity was calculated for pairs of composite expressions.

3.2.2 NV composite expressions

The test dataset by Mitchell and Lapata (2008) comprises a total of 3600 human similarity judgements. Each item consists of an intransitive verb and a subject noun, which are compared to another NV pair combining the same noun with a synonym of the verb that is chosen to be either similar or dissimilar to the verb in the context of the given subject. For instance, *child stray* is related to *child roam*, *roam* being a synonym of *stray*. The dataset was constructed by extracting NV composite expressions from the British National Corpus (BNC) and verb synonyms from WordNet (Miller *et al.* 1990). To evaluate the results

[4] https://dumps.wikimedia.org/enwiki/

[5] http://www.natcorp.ox.ac.uk

of our systems, Spearman correlation is computed between individual human similarity scores and the systems' predictions.

As the objective of the experiment is to compute the similarity between pairs of NV composite expressions, we are able to compare the similarity not only between the contextualized heads of two NV composite expressions, but also between their contextualized dependent expressions. So, we built compositional vectors using not only the head function, but also the dependent one. For instance, we compute the similarity between *eye flare* vs. *eye flame* by comparing first the verbs *flare* and *flame* when combined with *eye* in the subject position (head function), and by comparing how (dis)similar the noun *eye* is when combined with both the verbs *flare* and *flame* (dependent function). In addition, as we are provided with two similarities (*head* and *dep*) for each pair of compared expressions, it is possible to compute a new similarity measure by averaging *head* and *dep*, and what we call *head* + *dep* system.

Table 5 shows the Spearman's correlation values (ρ) obtained by our compositional strategy (*DepFunc*). We compare our results to the *Lexfunc* algorithm (Baroni and Zamparelli 2010), which is the state-of-the-art method for this dataset according to the ρ score reported in Dinu *et al.* (2013b) using a corpus consisting of approximately 2.8 billion tokens compiled from Wikipedia, BNC and ukWaC (Baroni *et al.* 2009). In the first row of *DepFunc*, we show the ρ value obtained by our combinatorial similarity measure (*head* + *dep*). The ρ score reaches 0.32, which is higher than using only head similarity (*head*) or dep similarity (*dep*). This shows that the similarity obtained by combining the head and dependent functions is more accurate than that obtained by using only one type of compositional function. The *head* + *dep* similarity strategy based on *DepFunc* outperforms the *Lexfunc* system (0.26). The baseline method we have implemented (first

system	ρ	***size of training corpus***
non-compositional (V)	0.11	2.5B tokens: Wiki & BNC
DepFunc (head + dep)	**0.32**	2.5B tokens: Wiki & BNC
DepFunc (head)	0.27	2.5B tokens: Wiki & BNC
DepFunc (dep)	0.31	2.5B tokens: Wiki & BNC
Lexfunc (Dinu et al. 2013)	0.26	2.8B tokens: Wiki, BNC & ukWaC

Table 5: Spearman's correlation for intransitive expressions using the benchmark by Mitchell and Lapata (2008)

row in Table 5) is a non-compositional strategy just based on the similarity between the head verbs within the NV pairs. In this case, all the compositional methods clearly outperform this basic strategy. Finally, the non-compositional similarity between the noun subjects has not been computed because the nouns of each NV pair are identical in the current dataset.

3.2.3 NVN composite expressions

The last experiment consists of evaluating the quality of compositional vectors built by means of the consecutive application of head and dependency functions associated with nominal subject and direct object. The experiment is performed on the dataset developed in Grefenstette and Sadrzadeh (2011a). The dataset was built using the same guidelines as Mitchell and Lapata (2008), using transitive verbs paired with subjects and direct objects: NVN composites.

Given our compositional strategy, we are able to compositionally build several vectors that somehow represent the meaning of the whole NVN composite expression. Take the expression *the coach runs the team*. If we follow the left-to-right strategy (noted *nv-n*), at the end of the compositional process, we obtain two fully contextualized senses:

nv-n_head The sense of the head *run*, as a result of being contextualized first by the preferences imposed by the subject and then by the preferences required by the direct object. We note *nv-n_head* the final sense of the head in a NVN composite expression following the left-to-right strategy.

nv-n_dep The sense of the object *team*, as a result of being contextualized by the preferences imposed by *run* previously combined with the subject *coach*. We note *nv-n_dep* the final sense of the direct object in a NVN composite expression following the left-to-right strategy.

If we follow the right-to-left strategy (noted *n-vn*), at the end of the compositional process, we obtain two fully contextualized senses:

n-vn_head The sense of the head *run* as a result of being contextualized first by the preferences imposed by the object and then by the subject.

n-vn_dep The sense of the subject *coach*, as a result of being contextualized by the preferences imposed by *run* previously combined with the object *team*.

Table 6 shows the Spearman's correlation values (ρ) obtained by all the different variations built by our system *DepFunc*. The best score was achieved by averaging the head and dependent similarity values derived from the *n-vn* (right-to-left) strategy. Let us note that, for NVN composite expressions, the left-to-right strategy seems to build less reliable compositional vectors than the right-to-left counterpart. Note too that the broader model (*n-vn* + *nv-n*) resulting from combining the two strategies does not improve the results of the best one (*n-vn*). This model, *n-vn* + *nv-n*, is computed by averaging the similarities of both *n-vn_head* + *dep* and *nv-n_head* + *dep*. More precisely, it is the result of averaging the four fully contextualized vectors:

- *nv-n_head*: left-to-right full contextualization of the verb,
- *nv-n_dep*: left-to-right full contextualization of the object noun,
- *n-vn_head*: right-to-left full contextualization of the verb,
- *n-vn_dep*: right-to-left full contextualization of the subject noun.

system	ρ
non-compositional (V)	0.27
DepFunc (nv_head)	0.33
DepFunc (nv_dep)	0.19
DepFunc (vn_head)	0.36
DepFunc (vn_dep)	0.38
DepFunc (nv-n_head + dep)	0.35
DepFunc (nv-n_head)	0.33
DepFunc (nv-n_dep)	0.20
DepFunc (n-vn_head + dep)	**0.46**
DepFunc (n-vn_head)	0.36
DepFunc (n-vn_dep)	0.42
DepFunc (n-vn + nv-n)	0.44
Grefenstette and Sadrzadeh (2011)	0.28
Hashimoto and Tsuruoka (2014)	0.43
Polajnar et al. (2015)	0.35

Table 6:
Spearman's correlation
for transitive expressions
using the benchmark
by Grefenstette and Sadrzadeh (2011)

It is worth mentioning that the best fully contextualized vector is the subject noun generated with the right-to-left algorithm (*n-vn_dep* = 0.42), which outperforms the two contextualized verb senses: *n-vn_head* and *nv-n_head*. This result was not expected since the sense of the verb represents the meaning of the syntactic root of the sentence, which is the best connected word in the syntactic tree and, by extension, the best positioned word to represent the core meaning of the sentence. However, the fact that the subject noun works so well is conceptually possible since any fully contextualized vector may represent the meaning of the whole sentence from a specific point of view.

The score value obtained by our *n-vn_head* + *dep* right-to-left strategy outperforms the three other systems tested using this dataset: Grefenstette and Sadrzadeh (2011b) and Polajnar *et al.* (2015), which are two works based on the categorical compositional distributional model of meaning of Coecke *et al.* (2010), and the neural network strategy described in Hashimoto and Tsuruoka (2015).

At the top of Table 6, we show the non-contextual baseline we have created for this dataset: similarity between single verbs. No test has been made for subject and object nouns since they are identical in each pair of transitive clauses, as was the case with the subject nouns in the dataset of intransitive expressions. In the current experiment, the correlation ρ of the non-compositional baseline is much higher than in Table 5. This might explain why the best correlation value of the compositional strategy is also much higher for this dataset (0.46 vs. 0.32). The table also shows four intermediate values resulting from comparing partial compositional constructions: the noun-verb (*nv_head* and *nv_dep*) and the verb-noun (*vn_head* and *vn_dep*) combinations. Two interesting remarks can be made from these values when they are compared with the full compositional constructions.

First, there is no clear improvement of performance if we compare the full compositional information of the two transitive constructions with the partial combinations. On the one hand, the full *nv-n* construction does not improve the scores obtained by the partial intransitive *nv*. On the other hand, *n-vn* performs slightly better than *vn* but only in the case of the dependent function which makes use of contextualized selectional preferences: *n-vn_dep* = 0.42 / *vn_dep* = 0.38. The low performance at the second level of composition might call into question the use of contextualized vectors to build still more contextualized senses. The scarcity problem derived from the recursive combination of contextualized vectors is an important issue which could be resolved by incorporating more text/additional corpora, and which we should analyze with more complex evaluation tests.

The second remark is about the difference between the two algorithms: left-to-right and right-to-left. The scores achieved by the left-to-right algorithm (*nv, nv-n*) are clearly below those achieved by right-to-left (*vn, n-vn*) . This might be due to the weak semantic motivation of the selectional preferences involved in the subject dependency of transitive constructions in comparison to the direct object. In fact, right-to-left and left-to-right function application produces substantially different vectors because each algorithm corresponds to a particular hierarchy of constituents. Change of constituency implies different semantic entailments; for example, consider the different levels of constituency of noun modifiers (e.g. *fastest American runner* ≠ *American fastest runner*). Finally, the poor results of *nv* in this dataset compared with those obtained in Table 5 is explained because the subject role is less meaningful in transitive clauses than in intransitive ones. The subject of intransitive clauses is assigned a complex semantic role that tends to merge the notions of agent and patient. By contrast, the subject of transitive constructions tends to be just the agent of an action with an external patient.

4 RELATED WORK

Several models for compositionality in vector spaces have been proposed in the last decade, and most of them use bag-of-words as basic representations of word contexts. As has been said in the introduction, the basic approach to composition, explored by Mitchell and Lapata (2008, 2009, 2010), is to combine vectors of two syntactically related words with arithmetic operations: addition and component-wise multiplication. The additive model produces a sort of union of word contexts, whereas multiplication has an intersective effect. According to Mitchell and Lapata (2008), component-wise multiplication performs better than the additive model. However, in Mitchell and Lapata (2009, 2010), these authors explore weighted additive models giving more weight to some constituents in specific word combinations. For instance, in a noun-subject-verb combination, the verb is assigned a higher weight because the whole construction is closer to the verb than to the noun. Other weighted additive models are described in Guevara (2010) and Zanzotto *et al.* (2010). Another work using these basic composition operations is reported in Reddy *et al.* (2011). In this work, the compositional model is enriched with a notion close to our concept of contextualization, which the authors call *dynamic prototype*, but only applied to noun-noun compounds. The model represents each constituent by a prototype vector which is built dynamically by activating only the contexts considered to be relevant with regard to the other constituent. All these models share a common trait: they define composition operations solely for pairs of words. Their main drawback is that they do not propose a more systematic model accounting for all types of semantic composition. They do not focus on the logical aspects of the functional approach underlying compositionality.

As has been said before, other distributional approaches develop sound compositional models of meaning where functional words are represented as high-dimensional tensors (Coecke *et al.* 2010; Baroni and Zamparelli 2010; Grefenstette *et al.* 2011; Krishnamurthy and Mitchell 2013; Kartsaklis and Sadrzadeh 2013; Baroni 2013; Baroni *et al.* 2014). This idea is mostly based on Combinatory Categorial Grammar and typed functional application inspired by Montagovian semantics. The functional approaches relying on Categorial Grammar distinguish the words denoting atomic types, which are represented as vectors, from those that denote compositional functions applied to vectors. By contrast, in our compositional approach, we show that function application is not associated with predicate words such as adjectives or verbs, but rather with binary dependencies. Our semantic space does not map the syntactic structure of Combinatory Categorial Grammar but that of Dependency Grammar. This way, we avoid the troublesome high-order tensor representations of verbs with n-arity arguments.

Some of the approaches cited above induce the compositional meaning of the functional words from examples adopting regression techniques commonly used in machine learning (Baroni and Zamparelli 2010; Krishnamurthy and Mitchell 2013; Baroni 2013; Baroni *et al.* 2014). In our approach, by contrast, functions associated with dependencies are just basic arithmetic operations on vectors, as in the case of the arithmetic approaches to composition described above (Mitchell and Lapata 2008). Arithmetic approaches are easy to implement and produce high-quality compositional vectors, which makes them a good choice for practical applications (Baroni *et al.* 2014).

The other compositional approaches based on Categorial Grammar use tensor products for composition (Coecke *et al.* 2010; Grefenstette *et al.* 2011). As has been said in the introduction, at least two problems arise with tensor products. First, they result in an information scalability problem, since tensor representations grow exponentially as the phrases grow longer (Turney 2013). And second, tensor products did not perform as well as component-wise multiplication in the experiments made by Mitchell and Lapata (2010). To improve the performance of the composition process, the tensor-based approach reported in Kartsaklis *et al.* (2014) is provided with an explicit disambiguation step prior to composition. In Paperno *et al.* (2014), the authors try to partially overcome the scalability problem of tensors by representing a functional word as a vector plus an ordered set of matrices, with one matrix for each argument the function takes.

There are a few works using vector spaces structured with syntactic information which, as in our approach, are not based on *n*-order tensors. Thater *et al.* (2010) distinguish between *first-order* and *second-order* vectors in order to allow two syntactically incompatible vectors to be combined. Similarly, in Melamud *et al.* (2015) the second-order vectors are called “substitute vectors”. The notion of second-order (or substitute) vector is close to our concept of *indirect vector*, while their first-order vector corresponds to our *direct vector*. However, there are important differences with regard to our approach. In (Thater *et al.* 2010), the combination of a first-order with a second-order vector returns a second-order vector, which can be combined with other second-order vectors. This could require the resort to third-order (or *n*-order) vectors at further levels of vector composition. By contrast, in our approach, any vector combination always returns a first-order (i.e. *direct*) vector, and we only permit compositional combinations between a direct vector and an indirect one. This simplifies the compositional process at any level of analysis.

The work by Thater *et al.* (2010) is inspired by that described in Erk and Padó (2008) and Erk and Padó (2009). Erk and Padó (2008) proposes a method in which the combination of two words, *a* and *b*, returns two vectors: a vector *a'* representing the sense of *a* given the selectional preferences imposed by *b*, and a vector *b'* standing for the sense of *b* given the (inverse) selectional preferences imposed by *a*. The main problem is that this approach does not propose any compositional model for sentences. Its objective is to simulate word sense disambiguation, but not to model semantic composition at any level of analysis. In Erk and Padó (2009), the authors briefly describe an extension of their model by proposing a recursive application of the compositional function. However, they only formalize the recursive application when the composite expression consists of two dependents linked to the same head. So, they explain how the head is contextualized by its dependents, but not the other way around. They do not model the influence of context on the selectional preferences. In other terms, their recursive model does not make use of contextualized selectional preferences. By contrast, in our approach, selectional preference are contextualized recursively. This is formalized in Equation 8 (Section 2.3).

Thater *et al.* (2010) took up the basic idea from Erk and Padó (2008) which consists in exploiting selectional preference information for contextualization and disambiguation. However, they did not borrow the idea of splitting the output of a word combination into two different vectors (one per word). As far as we know, no fully and coherent compositional approach has been proposed on the basis of the interesting idea of returning two contextualized vectors per combination. Our approach is an attempt to join the main ideas of these syntax-based models (namely, selectional preferences as indirect vectors and two returning senses per word combination) into an entirely compositional model. In sum, we generalize the model introduced by Erk and Padó (2008) to include dependencies as compositional operations allowing us to interpret any composite expression with any number of word constituents. Finally, it is important to point out that there is another relevant difference between our work and that reported in Erk and Padó (2008), Thater *et al.* (2010), and Melamud *et al.* (2015). While they tested their systems on a task of determining word meaning in context by lexical substitution, to evaluate our system we performed experiments in the task of measuring phrase similarity.

A very similar work to our compositional approach has been reported in Weir *et al.* (2016). The authors also state that distributional composition is a matter of integrating the meaning of each of the words in the phrase. The main difference is the type of context they use to build word vectors. Each word occurrence is modelled by what they call "anchored packed dependency tree", which is a dependency-based graph that captures the full sentential context of the word. The main drawback of this context approach is its critical tendency to build very sparse word representations.

Finally, recent works make use of deep learning strategies to build compositional vectors, such as recursive neural network models (Socher *et al.* 2012; Hashimoto and Tsuruoka 2015), which share with our model the idea that in the composition of two words both words modify each other's meaning. Similarly, the bidirectional recursive neural network reported in Irsoy and Cardie (2014) computes a context vector for each word. It is also worth noting the deep learning syntax-based compositional version of the C-BOW algorithm (Pham *et al.* 2015).

5 CONCLUSIONS

In this paper, we described a distributional model to contextualize word meaning in composite expressions based on a syntactically structured vector space. To avoid the different syntactic environments associated with two syntactically dependent words, we proposed to combine direct with indirect vectors, which are compatible and can be merged into a new direct vector. An indirect vector represents the selectional preferences that one word uses to contextualize the direct vector of the other word. The combination of two related words gives rise to two vectors which represent the senses of the two contextualized words. This process can be repeated until no syntactic dependency is found in the analyzed composite expression. The compositional interpretation of a composite expression builds the sense of each constituent word in a recursive and incremental way.

Syntactic dependencies are endowed with a combinatorial meaning. Characterizing dependencies as compositional devices has important consequences on the way in which the process of semantic interpretation is considered. First, dependencies are binary functions on vectors while all content words are vectors. Vectors of content words (as well as collocations and idioms) can be constructed from a corpus directly, while vectors of composite expressions are the result of composition operations driven by dependencies. Second, the contextualization process is performed in an incremental way dependency by dependency. It starts with very ambiguous vectors associated with the constituent words before composition and results in more compact and less ambiguous vectors associated with the contextualized words. And third, as syntactic dependencies are conceived here as semantic operations, syntax becomes a semantic participant involved in the interpretation process (Langacker 1991).

Our compositional model tackles the problem of *information scalability*. This problem states that the size of semantic representations should grow in proportion to the amount of information that they are representing. If the size of the contextualized vectors is fixed, eventually there will be information loss. Besides, the size of vector representations should not grow exponentially. In our approach, even if the size of the contextualized vectors is fixed, there is no information loss since each word of the composite expression is associated to a compositional vector representing its context-sensitive sense. In addition, the contextualized vectors do not grow exponentially since their size is fixed by the vector space: they are all first-order tensors.

Substantial problems still remain unsolved. For instance there is no clear boundary between compositional and non-compositional expressions (collocations, compounds, or idioms). It seems to be obvious that vectors of full compositional units should be built by means of compositional operations and predictions based on their constituent vectors. It is also evident that vectors of entirely frozen expressions should be totally derived from corpus co-occurrences of the whole expressions without considering internal constituency. However, there are many expressions, in particular collocations (such as *save time*, *go mad*, *heavy rain*, etc.) which can be considered as both compositional and non-compositional. In those cases, it is not clear which is the best method to build their distributional representation: predicted vectors by compositionality; or corpus-observed vectors of the whole expression.

Another problem that has not been considered is how to represent the semantics of some grammatical words, namely determiners and auxiliary verbs (i.e., noun and verb specifiers). This might require a different functional approach, probably closer to the work described by Baroni *et al.* (2014), which defines functions as linear transformations on vector spaces. A solution might be similarly inspired by Gupta *et al.* (2015), where the authors analyze the distributional features associated with referential expressions.

An obvious drawback of the recursive strategy is the scarcity that results from the iterative application of several contextualizations to the same word vector. The more complex the dependency structure, the fewer occurrences there will be to compute the in-context selectional preferences. This problem also underlies other similar approaches based on transparent and interpretable distributional models, such as that reported in Weir *et al.* (2016). Kober *et al.* (2016) proposed a solution to this problem. Their proposal involves explicitly inferring un-observed co-occurrences using the distributional neighborhood. More precisely, in order to transform a sparse word vector w into a new enriched vector w', the algorithm iterates over all word vectors w in a given distributional model M, and adds the vector representations of the nearest neighbors, determined by cosine similarity, to the representation of the new enriched word vector w'. In future work, we will carry out new experiments by using this strategy on similarity datasets containing phrases or sentences with more complex syntactic structures.

Among the most fundamental applications of compositional models are paraphrasing and textual entailment. For instance, by making use of sentence similarity, we should be able to infer that the sentence *A stadium craze is sweeping the country* entails *A craze is covering the nation*, but not *A craze is brushing the nation* (Garrette *et al.* 2014). These applications build compositional vectors from co-occurrences observed in monolingual corpora. However, if the same methodology is applied to acquire phrase and sentence similarity from comparable corpora, it could be possible to learn translation equivalents of composite units. This could lead to new machine translation techniques.

In future work, we will try to define more complex semantic word models by combining relation-based features (from WordNet or other lexical resources) with distributional-based representations. We will also explore the link between distributional representations and model-theoretical objects (entities, events, and so on), by considering bridging concepts such as *ideal distributions*.

The code for DepFunc and the distributional models used in the experiments are made freely available.[6]

ACKNOWLEDGMENTS

This work received financial support from project DOMINO (PGC2018-102041-B-I00, MCIU/AEI/FEDER, UE), eRisk (RTI2018-093336-B-C21), the Consellería de Cultura, Educación e Ordenación Universitaria (accreditation 2016-2019, ED431G/08), and the European Regional Development Fund (ERDF).

REFERENCES

Marco BARONI (2013), Composition in Distributional Semantics, *Language and Linguistics Compass*, 7:511–522.

Marco BARONI, Raffaella BERNARDI, and Roberto ZAMPARELLI (2014), Frege in Space: A Program for Compositional Distributional Semantics, *Linguistic Issues in Language Technology (LiLT)*, 9:241–346.

Marco BARONI, Silvia BERNARDINI, Adriano FERRARESI, and Eros ZANCHETTA (2009), The WaCky Wide Web: A Collection of Very Large Linguistically Processed Webcrawled Corpora, *Language Resources and Evaluation*, 43(3):209–226.

Marco BARONI and Roberto ZAMPARELLI (2010), Nouns Are Vectors, Adjectives Are Matrices: Representing Adjective-noun Constructions in Semantic Space, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP'10, pp. 1183–1193, Stroudsburg, PA, USA.

Jon BARWISE (1987), Recent Developments in Situation Semantics, in M. NAGAO, editor, *Language and Artificial Intelligence*, pp. 387–399, North Holla.

Raffaella BERNARDI, Georgiana DINU, Marco MARELLI, and Marco BARONI (2013), A Relatedness Benchmark to Test the Role of Determiners in Compositional Distributional Semantics, in *The 51st Annual Meeting of the Association for Computational Linguistics ACL-2013*, pp. 53–57, The Association for Computational Linguistics.

Chris BIEMANN and Martin RIEDL (2013), Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity, *Journal of Language Modelling*, 1(1):55–95.

[6] `http://fegalaz.usc.es/DepFunc.tgz`

B. COECKE, M. SADRZADEH, and S. CLARK (2010), Mathematical Foundations for a Compositional Distributional Model of Meaning, *Linguistic Analysis*, 36(1–4):345–384.

Ann COPESTAKE and Aurelie HERBELOT (2012), Lexicalised Compositionality, in *http://www.cl.cam.ac.uk/˜ah433/lc-semprag.pdf*, Unpublished article.

Fabrizio COSTA, Vincenzo LOMBARDO, Paolo FRASCONI, and Giovanni SODA (2001), Wide Coverage Incremental Parsing by Learning Attachment Preferences, in *Conference of the Italian Association for Artificial Intelligence (AIIA)*, pp. 297–307.

Georgiana DINU, Nghia PHAM, and Marco BARONI (2013a), DISSECT: DIStributional SEmantics Composition Toolkit, in *ACL 2013 Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pp. 31–36, East Stroudsburg PA.

Georgiana DINU, Nghia PHAM, and Marco BARONI (2013b), General Estimation and Evaluation of Compositional Distributional Semantic Models, in *ACL 2013 Workshop on Continuous Vector Space Models and their Compositionality (CVSC 2013)*, pp. 50–58, East Stroudsburg PA.

Ted DUNNING (1993), Accurate Methods for the Statistics of Surprise and Coincidence, *Computational Linguistics*, 19(1):61–74.

Katrin ERK (2013), Towards a Semantics for Distributional Representations, in *10th International Conference on Computational Semantics (IWCS 2013)*, pp. 95–106.

Katrin ERK and Sebastian PADÓ (2008), A Structured Vector Space Model for Word Meaning in Context, in *2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-2008*, pp. 897–906, Honolulu, HI.

Katrin ERK and Sebastian PADÓ (2009), Paraphrase Assessment in Structured Vector Space: Exploring Parameters and Datasets, in *Proceedings of the EACL Workshop on Geometrical Methods for Natural Language Semantics*, pp. 57–65, Athens, Greece.

John Rupert FIRTH (1957), A synopsis of linguistic theory 1930-1955, *Studies in Linguistic Analysis*, pp. 1–32.

Pablo GAMALLO (2008), The Meaning of Syntactic Dependencies, *Linguistik OnLine*, 35(3):33–53.

Pablo GAMALLO (2015), Dependency Parsing with Compression Rules, in *Proceedings of the 14th International Workshop on Parsing Technology (IWPT 2015)*, pp. 107–117, Association for Computational Linguistics, Bilbao, Spain.

Pablo GAMALLO (2017a), Comparing Explicit and Predictive Distributional Semantic Models Endowed with Syntactic Contexts, *Language Resources and Evaluation*, 51(3):727–743.

Pablo GAMALLO (2017b), The Role of Syntactic Dependencies in Compositional Distributional Semantics, *Corpus Linguistics and Linguistic Theory*, 13(2):261–289.

Pablo GAMALLO (2017c), Sense Contextualization in a Dependency-Based Compositional Distributional Model, in *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 1–9, Association for Computational Linguistics, doi:10.18653/v1/W17-2601, `http://aclweb.org/anthology/W17-2601`.

Pablo GAMALLO, Alexandre AGUSTINI, and Gabriel LOPES (2005), Clustering Syntactic Positions with Similar Semantic Requirements, *Computational Linguistics*, 31(1):107–146.

Pablo GAMALLO and Stefan BORDAG (2011), Is Singular Value Decomposition Useful for Word Simalirity Extraction, *Language Resources and Evaluation*, 45(2):95–119.

Pablo GAMALLO and Marcos GARCIA (2018), Dependency Parsing with Finite State Transducers and Compression Rules, *Information Processing & Management*, 54(6):1244–1261.

Pablo GAMALLO and Martín PEREIRA-FARIÑA (2017), Compositional Semantics using Feature-Based Models from WordNet, in *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*, pp. 1–11, Association for Computational Linguistics, `http://aclweb.org/anthology/W17-1901`.

Dan GARRETTE, Katrin ERK, and Raymond MOONEY (2014), A Formal Approach to Linking Logical Form and Vector-Space Lexical Semantics, in H. BUNT, J. BOS, and S. PULMAN, editors, *Text, Speech and Language Technology: Computing Meaning*, pp. 27–48, Springer.

Edward GREFENSTETTE and Mehrnoosh SADRZADEH (2011a), Experimental Support for a Categorical Compositional Distributional Model of Meaning, in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pp. 1394–1404.

Edward GREFENSTETTE and Mehrnoosh SADRZADEH (2011b), Experimenting with Transitive Verbs in a DisCoCat, in *Workshop on Geometrical Models of Natural Language Semantics (EMNLP 2011)*.

Edward GREFENSTETTE, Mehrnoosh SADRZADEH, Stephen CLARK, Bob COECKE, and Stephen PULMAN (2011), Concrete Sentence Spaces for Compositional Distributional Models of Meaning, in *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, pp. 125–134.

Gregory GREFENSTETTE (1995), Evaluation Techniques for Automatic Semantic Extraction: Comparing Syntatic and Window Based Approaches, in Branimir BOGURAEV and James PUSTEJOVSKY, editors, *Corpus processing for Lexical Acquisition*, pp. 205–216, The MIT Press.

Emiliano GUEVARA (2010), A Regression Model of Adjective-Noun Compositionality in Distributional Semantics, in *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '10, p. 33–37.

Abhijeet GUPTA, Gemma BOLEDA, Marco BARONI, and Sebastian PADÓ (2015), Distributional Vectors Encode Referential Attributes, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 12–21, Association for Computational Linguistics, Lisbon, Portugal, `http://aclweb.org/anthology/D15-1002`.

Kazuma HASHIMOTO and Yoshimasa TSURUOKA (2015), Learning Embeddings for Transitive Verb Disambiguation by Implicit Tensor Factorization, in *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pp. 1–11, Association for Computational Linguistics, Beijing, China, `http://www.aclweb.org/anthology/W15-4001`.

Richard HUDSON (2003), The Psychological Reality of Syntactic Dependency Relations, in *Proceedings of the First International Conference on Meaning-Text Theory*, pp. 181–192, Paris.

Ozan IRSOY and Claire CARDIE (2014), Deep Recursive Neural Networks for Compositionality in Language, in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2096–2104, `http://papers.nips.cc/paper/5551-deep-recursive-neural-networks-for-compositionality-in-language`.

Sylvain KAHANE (2003), Meaning-Text Theory, in *Dependency and Valency: An International Handbook of Contemporary Reseach*, Berlin: De Gruyter.

Hans KAMP and Uwe REYLE (1993), *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Languge. Formal Logic and Discourse Representation Theory*, Kluwer Academic Publisher.

Dimitri KARTSAKLIS (2014), Compositional Operators in Distributional Semantics, *Springer Science Reviews*, 2(1–2):161–177.

Dimitri KARTSAKLIS, Nal KALCHBRENNER, and Mehrnoosh SADRZADEH (2014), Resolving Lexical Ambiguity in Tensor Regression Models of Meaning, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Vol. 2: Short Papers)*, pp. 212–217, Association for Computational Linguistics, Baltimore, USA.

Dimitri KARTSAKLIS and Mehrnoosh SADRZADEH (2013), Prior Disambiguation of Word Tensors for Constructing Sentence Vectors, in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pp. 1590–1601.

Ruth M. KEMPSON, Wilfried MEYER-VIOL, and Dov GABBAY (1997), Language Understanding: A Procedural Perspective, in C. RETORE, editor, *First*

International Conference on Logical Aspects of Computational Linguistics, pp. 228–247, Lecture Notes in Artificial Intelligence Vol. 1328. Springer Verlag.

Ruth M. KEMPSON, Wilfried MEYER-VIOL, and Dov GABBAY (2001), *Dynamic Syntax: The Flow of Language Understanding*, Blackwell, Oxford.

Thomas KOBER, Julie WEEDS, Jeremy REFFIN, and David J. WEIR (2016), Improving Sparse Word Representations with Distributional Inference for Semantic Composition, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1–4, 2016*, pp. 1691–1702, http://aclweb.org/anthology/D/D16/D16-1175.pdf.

Jayant KRISHNAMURTHY and Tom MITCHELL (2013), Vector Space Semantic Parsing: A Framework for Compositional Vector Space Models, in *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pp. 1–10, Association for Computational Linguistics.

Germán KRUSZEWSKI and Marco BARONI (2014), Dead Parrots Make Bad Pets: Exploring Modifier Effects in Noun Phrases, in *Proceedings of the Third Joint Conference on Lexical and Computational Semantics, *SEM@COLING 2014, August 23-24, 2014, Dublin, Ireland.*, pp. 171–181, http://aclweb.org/anthology/S/S14/S14-1021.pdf.

Ronald W. LANGACKER (1991), *Foundations of Cognitive Grammar: Descriptive Applications*, volume 2, Stanford University Press, Stanford.

Ken MCRAE, Todd R. FERRETI, and Liane AMYOTE (1997), Thematic Roles as Verb-specific Concepts, in M.C. MACDONALD, editor, *Lexical Representations and Sentence Processing*, pp. 137–176, Psychology Press.

Oren MELAMUD, Ido DAGAN, and Jacob GOLDBERGER (2015), Modeling Word Meaning in Context with Substitute Vectors, in *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pp. 472–482, http://aclweb.org/anthology/N/N15/N15-1050.pdf.

George A. MILLER, Richard BECKWITH, Christiane FELLBAUM, Derek GROSS, and Katherine J. MILLER (1990), Introduction to Wordnet: an On-Line Lexical Database, *International Journal of Lexicography*, 3(4):235–244.

David MILWARD (1992), Dynamics, Dependency Grammar and Incremental Interpretation, in *14th Conference on Computational Linguistics (COLING 1992)*, pp. 1095–1099, Nantes.

Jeff MITCHELL and Mirella LAPATA (2008), Vector-Based Models of Semantic Composition, in *Proceedings of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, pp. 236–244, Columbus, Ohio.

Jeff MITCHELL and Mirella LAPATA (2009), Language Models Based on Semantic Composition, in *Proceedings of Empirical Methods in Natural Language Processing (EMNLP-2009)*, pp. 430–439.

Jeff MITCHELL and Mirella LAPATA (2010), Composition in Distributional Models of Semantics, *Cognitive Science*, 34(8):1388–1439.

Richard MONTAGUE (1970), Universal Grammar, *Theoria*, 36(3):373–398.

Muntsa PADRÓ, Marco IDIART, Aline VILLAVICENCIO, and Carlos RAMISCH (2014), Nothing like Good Old Frequency: Studying Context Filters for Distributional Thesauri, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 419–424.

Denis PAPERNO, Nghia The PHAM, and Marco BARONI (2014), A Practical and Linguistically-Motivated Approach to Compositional Distributional Semantics, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 90–99, Association for Computational Linguistics, Baltimore, Maryland, `http://www.aclweb.org/anthology/P/P14/P14-1009`.

Nghia The PHAM, Germán KRUSZEWSKI, Angeliki LAZARIDOU, and Marco BARONI (2015), Jointly Optimizing Word Representations for Lexical and Sentential Tasks with the C-PHRASE Model, in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pp. 971–981, `http://aclweb.org/anthology/P/P15/P15-1094.pdf`.

Tamara POLAJNAR, Laura RIMELL, and Stephen CLARK (2015), An Exploration of Discourse-Based Sentence Spaces for Compositional Distributional Semantics, in *Proceedings of the First Workshop on Linking Computational Models of Lexical, Sentential and Discourse-level Semantics*, pp. 1–11, Association for Computational Linguistics, Lisbon, Portugal, `http://aclweb.org/anthology/W15-2701`.

James PUSTEJOVSKY (1995), *The Generative Lexicon*, MIT Press, Cambridge.

Siva REDDY, Ioannis P. KLAPAFTIS, Diana MCCARTHY, and Suresh MANANDHAR (2011), Dynamic and Static Prototype Vectors for Semantic Composition, in *Fifth International Joint Conference on Natural Language Processing, IJCNLP 2011, Chiang Mai, Thailand, November 8–13, 2011*, pp. 705–713, `http://aclweb.org/anthology/I/I11/I11-1079.pdf`.

Mehrnoosh SADRZADEH, Stephen CLARK, and Bob COECKE (2013), The Frobenius Anatomy of Word Meanings I: Subject and Object Relative Pronouns, *Journal of Logic and Computation*, 23(6):1293–1317, doi:10.1093/logcom/ext044, `http://dx.doi.org/10.1093/logcom/ext044`.

Matthias Schlesewsky and Ina Bornkessel (2004), On Incremental Interpretation: Degrees of Meaning Accessed During Sentence Comprehension, *Lingua*, 114:1213–1234.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng (2012), Semantic Compositionality Through Recursive Matrix-vector Spaces, in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pp. 1201–1211, Association for Computational Linguistics, Stroudsburg, PA, USA, http://dl.acm.org/citation.cfm?id=2390948.2391084.

Mark Steedman (1996), *Surface Structure and Interpretation*, The MIT Press.

Michael K. Tanenhaus and Greg Carlson (1989), Lexical Structure and Language Comprehension, in William Marslen-Wilson, editor, *Lexical Representation and Process*, pp. 530–561, The MIT Press.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal (2010), Contextualizing Semantic Representations Using Syntactically Enriched Vector Models, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 948–957, Stroudsburg, PA, USA.

John Truswell, Michael K. Tanenhaus, and Susan M. Garnsey (1994), Semantic Influences on Parsing: Use of Thematic Role Information in Syntactic Ambiguity Resolution, *Journal of Memory and Language*, 33:285–318.

Peter D. Turney (2013), Domain and Function: A Dual-Space Model of Semantic Relations and Compositions, *Journal of Artificial Intelligence Research (JAIR)*, 44:533–585.

David J. Weir, Julie Weeds, Jeremy Reffin, and Thomas Kober (2016), Aligning Packed Dependency Trees: A Theory of Composition for Distributional Semantics, *Computational Linguistics*, 42(4):727–761.

Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar (2010), Estimating Linear Models for Compositional Distributional Semantics, in *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pp. 1263–1271.