UNIVERSITY OF TRENTO

Inference with Distributional Semantic Models

by

Germán David Kruszewski Martel

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
CIMeC
Doctoral School in Cognitive and Brain Sciences

November 2016

UNIVERSITY OF TRENTO

Abstract

CIMeC
Doctoral School in Cognitive and Brain Sciences

Doctor of Philosophy

by Germán David Kruszewski Martel

Distributional Semantic Models have emerged as a strong theoretical and practical approach to model the meaning of words. Indeed, an increasing body of work has proved their value in accounting for a wide range of semantic phenomena. Yet, it is still unclear how we can use the semantic information contained in these representations to support the natural inferences that we produce in our every day usage of natural language. In this thesis, I explore a selection of challenging relations that exemplify these inferential processes. To this end, on one hand, I present new publicly available datasets to allow for their empirical treatment. On the other, I introduce computational models that can account for these relations using distributional representations as their conceptual knowledge repository. The performance of these models demonstrate the feasibility of this approach while leaving room for improvement in future work.

Acknowledgements

I would like to heartily thank my advisor, Marco Baroni, for all the wise and helpful guidance given during these years. I also want to thank the two other members of my Oversight Committee and CLIC Lab colleagues, Raffaella Bernardi and Roberto Zamparelli, for all their feedback and interesting discussions. Thanks also to Leah Mercanti and all the Doctoral School in Cognitive and Brain Sciences for their diligent work.

Thank you as well to my father for all of the support that has been fundamental at every stage.

I would also like to deeply thank Alessandra for being with me at every step of this journey, encouraging me and giving me strength whenever I needed it.

Another big shout-out to my fellows The Nghia Pham and Angeliki Lazaridou for all the discussions, adventures and *aperitivi* shared. Thanks to Denis Paperno as well for the many interesting conversations and research collaborations. Many thanks too to the current and former members of *Sgt. Pepper's Lonely Hearts CLIC Lab* for all the time we shared: Aurelie Herbelot, Gemma Boleda, Sandro Pezzelle, Rossella Varvara, Marco Marelli, Georgiana Dinu, Yuan Tao, Andrew Anderson and Eva Maria Vecchi.

Finally, I acknowledge and thank ERC 2011 Starting Independent Research Grant n. 283554 (COMPOSES) for bringing together such an amazing group of people.

Contents

List of Publications

The contents described in this thesis have appeared in the following publications:

- G. Kruszewski and M. Baroni. Dead parrots make bad pets: Exploring modifier effects in noun phrases. *Lexical and Computational Semantics (* SEM 2014)*, page 171, 2014
- G. Kruszewski, D. Paperno, and M. Baroni. Deriving boolean structures from distributional vectors. *Transactions of the Association for Computational Linguistics*, 3:375–388, 2015
- G. Kruszewski and M. Baroni. So similar and yet incompatible: Toward automated identification of semantically compatible words. In *Proceedings of NAACL*, pages 64–969, 2015

List of Figures

List of Tables

Chapter 1

Introduction

The study of the meaning of words has been one of the central problems in linguistic theory (Geeraerts, 2010). In the past few decades, the distributional approach has emerged as a strong theory of word meaning. This approach finds it motivation in the Distributional Hypothesis, which states that words that occur in similar contexts tend to have similar meanings (Harris, 1954). This idea is further summarized by Firth's principle: "You shall know a word by the company it keeps" (Firth, 1957).

Operationally, Distributional Semantic Models (DSMs) process large text corpora in order to extract statistics on the associations between words and their collocations. The degree of association with each context is then incorporated into feature **vectors** that will represent the words' semantic content (Clark, 2015; Erk, 2012; Turney and Pantel, 2010). In Chapter 2 I lay down a more detailed description of how these models are concretely built. For now, it will suffice to say that words that have similar meaning will receive similar feature representations thanks to sharing similar contexts in text. Consider, for example, Figure 1.1. Here, I have sorted the dimensions of 300-dimensional distributional vectors in an arbitrary but consistent order to highlight the similarity among vectors of concepts related to sports that exhibit very similar distributions. In contrast, consider the concepts related to political systems which show another different distribution of features.

These representations of meaning have been extremely successful at modeling graded semantic relations, such as similarity[1]. For example, it has been extensively shown that computing similarity scores between two distributional vectors using the cosine metric (see Section 2.1.2) highly correlates with human-annotated similarity ratings (Baroni et al., 2014b), predicts selectional preferences (Padó and Lapata, 2007), clusters words

[1]For simplicity, I don't make a distinction in this discussion between similarity and relatedness. Note, however, that these are two different concepts. See, for example, Hill et al. (2016).

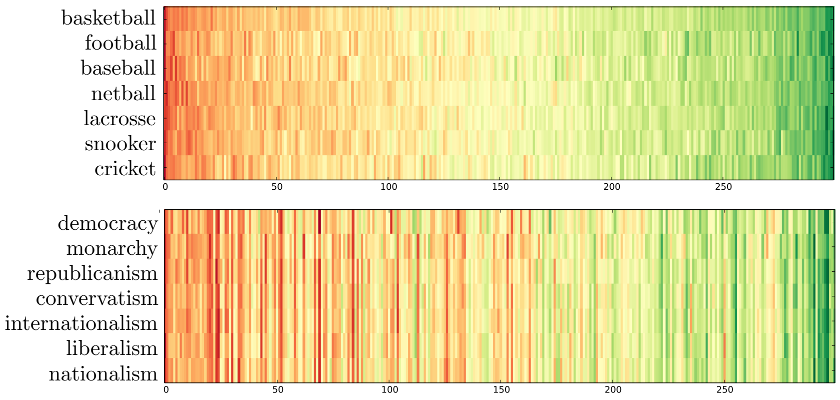


Figure 1.1: Distributional vectors associated with two clusters of concepts plotted with heat-maps. Dimensions are sorted in an arbitrary but consistent order to highlight the similarity of these vectors in the sports-related cluster.

that belong to the same category (Lund and Kevin, 1997) and correctly ranks words' synonyms (Landauer and Dumais, 1997).

However, for a semantic model to be complete, capturing these coarse-grained aspects of meaning is not enough. Rather, a good model should be able to account for the fine-grained inferences that our mental lexicon affords us. For instance, if I tell you that "all animals have an ulnar artery", you can safely infer that your dog also has an ulnar artery (Sloman, 1993). This inference is possible thanks to your conceptual knowledge telling you that dogs are a type of animal. Moreover, these inferences are moderated by typicality effects. Therefore, a semantic resource should not only be able to model these relations as sharp links between words, but rather as a fuzzy graded phenomenon.

In this thesis I will explore whether the semantic information contained in the distributional vectors can be used to do inference. To this end, I will study relations between concepts that depend on extracting fine-grained information from the conceptual knowledge that can support inferential processes. First, I will pay special attention to the hierarchical organization of concepts, which is ruled by the hyponymy relation, or more generally, by the **entailment** relation, defined as follows: A proposition P entails another proposition Q if the truth of Q is a logically necessary consequence of the truth of P (Cruse, 1986). Thus, *dog* is a hyponym of *animal* because *X is a dog* entails but is not in entailed by *X is an animal*. Conversely, *animal* is a hypernym of *dog*. Also, I will later explore the **compatibility** relation. Briefly, two concepts are said to be compatible if they can truthfully refer to the same object/animal/place; and they are incompatible if they cannot. For example, *criminal* and *lawyer* are compatible because a person being both things is entirely plausible, whereas *dog* and *cat* are clearly incompatible. These are two cases where non-trivial inference is required and where similarity is not sufficient to fully explain the phenomena.

As mentioned above, enabling these inferential processes is a necessary condition for a semantic model to be complete. Furthermore, computing them starting from distributional representations is advantageous. Consider the alternative of complementing a DSM with another semantic resource, such as WordNet (Fellbaum, 1998), where hyponymy and other phenomena are annotated for items within the lexicon's scope. Obviously, the lexicon should be very accurate with regard to the elements that have been stored in it. However, this approach faces many limitations. First, human conceptual knowledge is ever-expanding, so it is almost a sisyphic enterprise to try to hard-code every single relation between concepts, whereas DSMs can learn semantic representations from data alone, thus growing with experience. Even more crucially, mechanisms like conceptual combination enable us to creatively combine simple concepts into new ones that may afford very different relations to other words. For example, *bottle* is a hyponym (that is, a sub-class) of *drinkware*. However, *perfume bottle* does not belong to the class of *drinkware* anymore. Therefore, it is necessary to establish relations between an infinite number of linguistic units, leaving no other option than computing them at the time in which expressions are actually encountered.

The solution that I am going to explore in this thesis will be to use the distributional representations of arbitrary linguistic units (obtained via Compositional Distributional Semantic Models or CDSMs) to infer the relevant relations between them. These semantic representations will be then used to compute lexical or phrasal relations by means of an either pre-defined or learned-from-data mathematical function.

In the next chapter, I will introduce in detail DSMs and their compositional extension, CDSM. Next, I will start by exploring whether we can explain with DSMs the fact that some modifiers can shift the meaning of a noun so radically that they can change their natural categorization, like in the "*perfume bottle*/*drinkware*" example above (Chapter 3). Then, I will explore a novel method to establish an entailment relation between the vectors corresponding to two arbitrary linguistic units (Chapter 4). Finally, I will identify another paradigmatic relation, namely, that of compatibility, and also try to explain it following the same methodology outlined above (Chapter 5).

Chapter 2

Distributional Semantic Models

2.1 Distributional Semantic Models

Distributional Semantic Models (DSMs) find their underlying motivation in that words carrying similar meaning will occur in similar contexts (Harris, 1954). This is called the Distributional Hypothesis, which is captured in a nutshell by Firth's adage "You shall know a word by the company it keeps" (Firth, 1957).

The general operating principle is to use large (text) corpora in order to quantify the degree of association between each *target* term (that is, the word whose semantics we are trying to infer) and the *contexts* in which they occur. The definition of what constitutes a *context* is usually a design decision, but is often taken as the set of words occurring within a window around each occurrence of the target word.

Consider, for example, the small excerpt in Figure 2.1. Here we have three terms (`basketball`, `baseball` and `democracy`) for which we are gathering co-occurrence statistics. Notice that the sportive terms occur next to terms like `player`, `team` or `star`. These coincidences on highly associated contexts of occurrence can be harnessed to detect the similarity of these terms, in contrast to words like `democracy` whose associated collocates tend to be substantially different. Other contexts, like `bat`, `NBA`, `field` or `court` highlight the idiosyncratic differences between the two concepts. In contrast, articles such as `a` or `of` appear so frequently next to either of these concepts, that are hardly informative at all.

The computational model that builds this contextual information into feature vectors vary depending on the chosen approach by the system designer. For the purposes of this thesis, I will describe two important traditions called *count models* and *predict models* (Baroni et al., 2014b).

```
st one-armed professional <baseball> player . Hector Castro (
is is no one way to run a <baseball> team or a ballet company
 " invariably refers to a <baseball> field . Baseball has oft
ad out of the park with a <baseball> bat . Itchy and Scratchy
all " , to the sayings of <baseball> star Yogi Berra : " You

fortune in the 1980s as a <basketball> player , but it is his s
our favorite NBA and NCAA <basketball> team here . .. . Betting
e politicians is a former <basketball> star and another a forme
up was shown a video of a <basketball> game they were asked to
 with livestock or even a <basketball> court in your garage . A

tatorship to the Athenian <democracy> of Summerhill . Institut
ace in royal mail . It is <democracy> versus authoritarianism
 who murdered hundreds of <democracy> activists when they pour
 fully-fledged capitalist <democracy> with its own role to pla
tted to the rule of law , <democracy> and human rights that it
```

FIGURE 2.1: Distribution of words in contexts across a text corpus.

2.1.1 Count models

Count models derive their name from the fact that they first build a matrix counting the number of co-occurrences between target and context words in a sufficiently large text corpus. By context, we usually take words occurring no farther than a distance given by a parameter c from the target's instance. This co-occurrence counting procedure is shown in Algorithm 1. Typically, we only keep track of co-occurrence statistics for a manageable number of target and context words in order to alleviate memory usage.

Algorithm 1 Co-occurrence counting

```
procedure CooccurrenceCount($T$)                ▷ $T$ is a sequence of words
    $C \leftarrow \{\}$
    for $i = 1, \mathrm{len}(T)$ do
        for $k = -c, c; j \neq 0$ do
            $j \leftarrow i + k$
            if $j < \mathrm{len}(T)$ then
                $w_t, w_c \leftarrow T[i], T[j]$
                $C[w_t, w_c] \leftarrow C[w_t, w_c] + 1$
            end if
        end for
    end for
    return $C$
end procedure
```

This process produces a matrix of size $N \times M$ such as the one in Table 2.1, where N is the number of targets and M the number of contexts. In this example, the word `player` has occurred close to `baseball` 546 times.

	player	field	court	Athenian	king	a
baseball	546	350	5	1	35	975
basketball	485	10	410	1	45	1053
democracy	1	5	2	350	10	375
monarchy	2	1	4	7	276	330

Table 2.1: Sample counts of target/context co-occurrences in a text corpus.

In order to highlight infrequent but informative terms, and downplay the role of frequent but uninformative contexts these counts are further transformed with a non-linear operation. By far, the most commonly used transformation is Positive Pointwise Mutual Information or PPMI (Church and Hanks, 1990), defined as follows:

$$\text{PPMI}(i,j) = \max\left(\text{PMI}(i,j), 0\right) \tag{2.1}$$

$$\text{PMI}(i,j) = \log\left(\frac{ZC[i,j]}{\sum_{k=1}^{N} C[k,j] \sum_{l=1}^{M} C[i,l]}\right) \tag{2.2}$$

where $Z = \sum_{k=1}^{N} \sum_{l=1}^{M} C[k,l]$ is a normalizing constant.

PMI expresses how much more or less frequent a co-occurrence between two terms is in relation to what you would expect if these where two independent events. PPMI is the truncated version where negative values are discarded. Table 2.2 shows how the counts in the example above look when the PPMI transformation is applied. Notice how random noise is flattened down and non-informative all-present terms like the article a gets also discounted.

	player	field	court	Athenian	king	a
baseball	0.38	0.97	0	0	0	0
basketball	0.21	0	0.94	0	0	0.01
democracy	0	0	0	1.93	0	0
monarchy	0	0	0	0	1.86	0.03

Table 2.2: Sample pmi-transformed counts of target/context co-occurrences in a text corpus.

These resulting vectors (taken row-wise) can already be used as the semantic representations of the target words. However, it typically yields some improvement to apply a dimensionality reduction technique such as SVD (Landauer and Dumais, 1997; Strang, 2003), to obtain compact representations of much smaller dimension $k \ll M$.

SVD works by decomposing the original matrix (in this case, the PPMI-weigthed co-occurrence matrix) $X \in \mathbb{R}^{N \times M}$ as $X = USV^\top$, where $U \in \mathbb{R}^{N \times N}$ is an orthogonal matrix; $S \in \mathbb{R}^{N \times M}$, a diagonal matrix with sorted non-negative values (known as *singular values*) in the diagonal and another orthogonal matrix $V \in \mathbb{R}^{M \times M}$. Crucially, by computing the outer product between the first column vectors of U and V, weighted by the first singular value, one can recover as much information from the original matrix X as it is possible with a rank-1 matrix. Similarly, by adding the second column vectors weighted by the second singular value one can recover the maximal amount of information of X that is possible to express with a rank-2 matrix. In general, one can form the best rank-k approximation of the original matrix by summing over the first k *principal components*, that is, the outer product of the first k column vectors of U and V weighted by their singular values $S_{ii,1 \leq i \leq k}$. See Figure 2.2 for a schematic representation of this factorization and Figure 2.3, for an example of an image being approximated with the first k principal components.

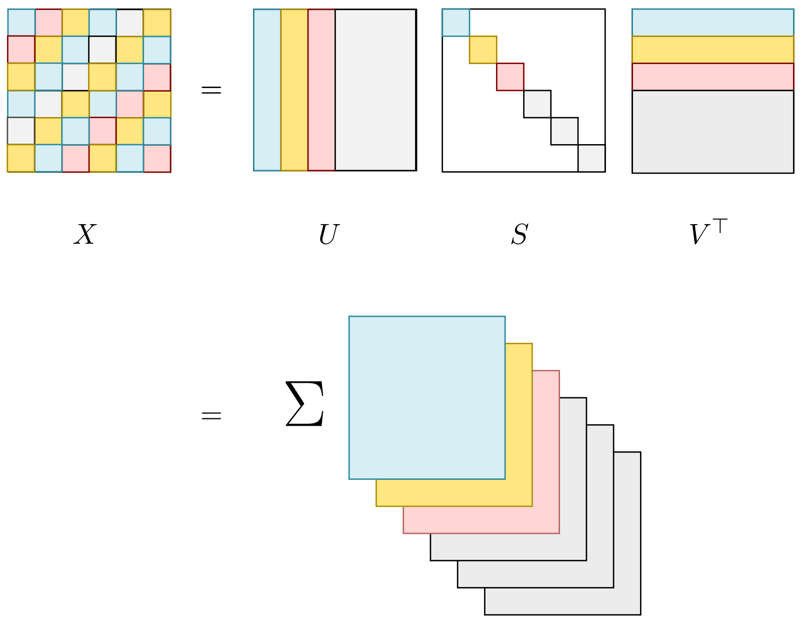
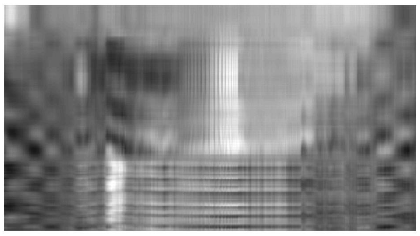


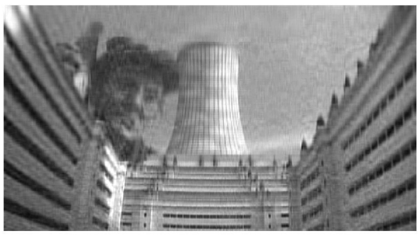
FIGURE 2.2: SVD decomposition as a sum of independent components that form the best rank-k approximation of a matrix.

When factorizing the semantic vectors matrix, the vectors in $U \in \mathbb{R}^{N \times N}$ still correspond to the target words: Multiplied by $SV^\top$ they would reconstruct the original M-dimensional vectors. Furthermore, US is a matrix having the same shape as the original $X \in \mathbb{R}^{N \times M}$, but expressed in a base where the first columns capture most of the original information. Therefore, the typical practice consist in truncating US keeping the first $k \ll M$ columns, which is interpreted as providing some sort of smoothing of the vectors.

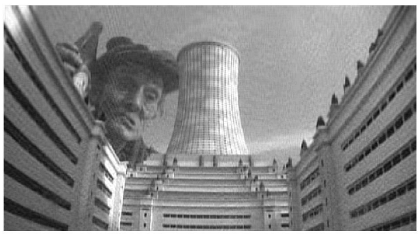
(A) Original image (rank 348)



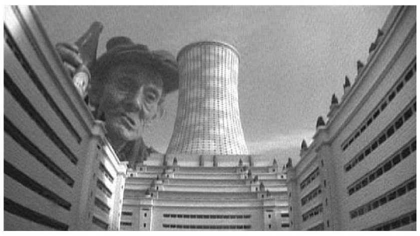
(B) Rank 5 approximation.



(C) Rank 30 approximation.



(D) Rank 55 approximation.



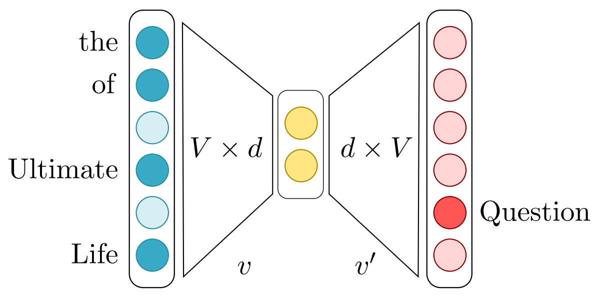
(E) Rank 80 approximation.

FIGURE 2.3: Reconstruction of an image using the rank-k approximation given by SVD.

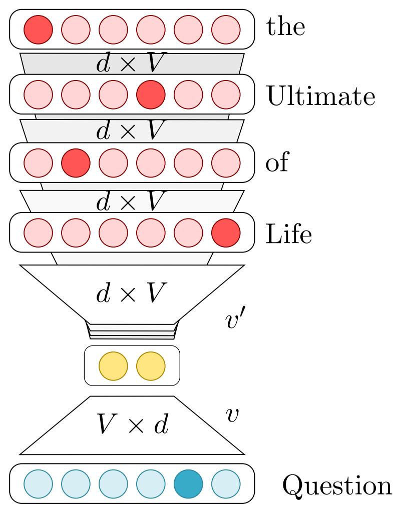
2.1.2 Predict models

Predict models where first introduced by Bengio et al. (2003), becoming particularly prominent with the word2vec model (Mikolov et al., 2013) thanks to its large-scale efficiency and strong empirical results. Word2vec finds its underlying idea in storing into each word vector the information that allows it to predict its most prototypical contexts or vice-versa – predict the contexts from the target. In spirit, this is very similar to count models, and indeed, Levy and Goldberg (2014) show that common implementations of the two optimize the same objective function.

The two main models introduced by Mikolov et al. can be construed as shallow neural networks with a single embedding layer and no non-linearity. Under this framework, words in the target and the context vocabularies get assigned dense vectors $v \in \mathbb{R}^d$ that are randomly initialized. Then, the contents of these vectors are adjusted with slightly different prediction objectives depending on the type of model. In the Continuous Bag-of-Words or CBOW model, the vectors are trained so that the sum of the context vectors best predict the target word (Figure 2.4a) from all the other words in the vocabulary. On the other hand, the Skip-Gram (SG) model, uses the target word as input and tries to predict each of the words in the context.



(A) CBOW model



(B) Skip-Gram model

FIGURE 2.4: Graphical illustration of the two predict models introduced by Mikolov et al. (2013).

More precisely, the SG objective function is to maximize the log-probability of the context word given the target, for each target-context words pair observed in the corpus:

$$J(\Theta) = \sum_{i=1}^{|T|} \sum_{i-c \leq j \leq i+c, j \neq i} \log P(w_j | w_i) \tag{2.3}$$

where the probability distribution is defined following the Boltzmann distribution [1]:

$$P(w_j | w_i) = \frac{\exp\left(v'_{w_j} \cdot v_{w_i}\right)}{\sum_{w=1}^{N} \exp\left(v'_w \cdot v_{w_i}\right)} \tag{2.4}$$

[1]In practice, a more efficient approximation called Negative Sampling is used, where instead of predicting the target among the whole vocabulary, only a small random sample is considered.

v_{w_i} is the vector corresponding to the given target word w_i, while v'_j is the vector corresponding to context word w_j that must be predicted. The vectors are updated after observing each word pair in the corpus according to the goal of optimizing the above mentioned objective function, using stochastic gradient descent (SGD).

Observe that what this objective function does is to enforce the dot product between vectors of co-occurring target-context word pairs to be larger than those of non-co-occurring ones. Also remember that the dot product between two vectors **a** and **b**, separated by an angle θ, is defined as:

$$\vec{a} \cdot \vec{b} = cos(\theta)\|\vec{a}\|\|\vec{b}\| \tag{2.5}$$

Therefore, the best way to modify the word vectors such that the dot product between co-occurring pairs (e.g. `basketball` and `player`) becomes larger, while lower for the others (e.g. `basketball` and `Athenian`) is to push the angle between the vectors corresponding to the first two terms smaller, and larger for the others. This is indeed what SG training updates achieve, by bringing word vectors' closer to the ones of their collocates, as illustrated in Figure 2.5.

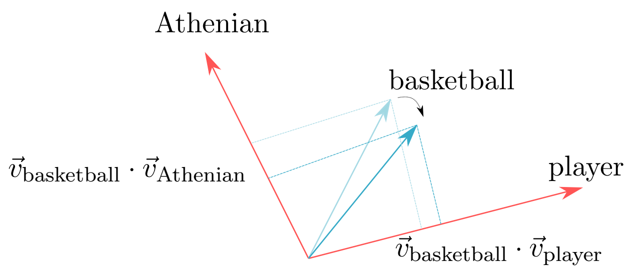

Athenian
basketball
player
$\vec{v}_{\text{basketball}} \cdot \vec{v}_{\text{Athenian}}$
$\vec{v}_{\text{basketball}} \cdot \vec{v}_{\text{player}}$

FIGURE 2.5: Sample vector update in a predict-style model when a `basketball`/`player` target-context pair is observed. `basketball` vector is "rotated" in the direction of `player`'s vector, thus enlarging the dot product between them, while reducing it with the non-observed context `Athenian`.

2.2 Compositional Distributional Semantic Models

In the previous section we have established a mechanism to produce semantic representations to individual words. How can we represent the meaning of larger expressions? One option would be to proceed exactly in the same way, collecting co-occurrences, this time not for words but for full phrases. However, this approach suffers the problem of data sparsity: With rare exceptions, such as frequent multi-word expressions in languages like English, the larger the phrase, the less it will be attested in corpora. Even more crucially, we can understand the meaning of phrases we have never heard before. How can we then assign a representation to them? This is the role of Compositional

Distributional Semantic Models (CDSM), which propose different algebraic operations to compose the meaning of the parts to obtain the meaning of the whole.

Mitchell and Lapata (2010) proposed a set of simple models in which each component of the phrase vector is a function of the corresponding components of the constituent vectors. Given vectors $\vec{a}$ and $\vec{b}$, the weighted additive model (**wadd**) returns their weighted sum: $\vec{p} = w_1\vec{a} + w_2\vec{b}$. In the dilation model (**dil**), the output vector is obtained by decomposing one of the input vectors, say $\vec{b}$, into a vector parallel to $\vec{a}$ and its orthogonal counterpart, and then dilating only the parallel vector by a factor λ before re-combining. The corresponding formula is: $(\vec{a} \cdot \vec{a})\vec{b} + (\lambda - 1)(\vec{a} \cdot \vec{b})\vec{a}$. In our experiments (Chapter 3 below), we stretch the head vector in the direction of the modifier (i.e., $\vec{a}$ is the modifier, $\vec{b}$ is the head). In the multiplicative model (**mult**), vectors are combined by component-wise multiplication, such that each phrase component p_i is given by: $p_i = a_i b_i$.

Guevara (2010) and Zanzotto et al. (2010) propose a full form of the additive model (**fulladd**), where the two constituent vectors are multiplied by weight matrices before being added, so that each phrase component is a weighted sum of *all* constituent components: $\vec{p} = W_1\vec{a} + W_2\vec{b}$.

Finally, the lexical function (**lexfunc**) model of Baroni and Zamparelli (2010) and Coecke et al. (2010) takes inspiration from formal semantics to characterize composition as function application. In particular, adjective-noun phrases, the adjective is treated as a linear function operating on the noun vector. Given that linear functions can be expressed by matrices and their application by matrix-by-vector multiplication, the adjective is represented by a matrix A to be multiplied with the noun vector $\vec{b}$, so that: $\vec{p} = A\vec{b}$.

This latter approach is further generalized by Baroni et al. (2014a) to cases in which words must take more than one argument. For example, verbs take the subject and the object as input and return the sentence representation as the output. To account for these cases, the authors propose to use high-order tensors, which can take as many arguments as allowed by their cardinality. The drawback of this approach is that it involves setting a very high number of parameters. For this reason, Paperno et al. (2014) propose to approximate these tensor operators by a linear combination of matrix-vector multiplication factors, denominated practical lexical function (**plf**).

In our experiments, the parameter settings for all of these models are learned by a procedure proposed by Baroni and Zamparelli (2010) for the lexical function model and later generalized to others by Dinu et al. (2013). The estimation is carried out by collecting representations for phrases that are frequent enough in the corpora and that involve the compositional operation that we are trying to estimate. The parameters are then learned through least-squares regression with the objective of approximating the phrases-observed vectors starting from the representations of their parts. For example, to approximate the lexical function corresponding to the adjective `red`, one would compute the corpus-extracted vectors for phrases like `red face`, `red wine`, `red car`, etc., and then estimate the weights of the mapping, such that, given the vector for `face`, produces `red face`; given `wine`, produces `red wine`, and so on.

Chapter 3

Modification effects in conceptual hierarchies

"This parrot is no more. It has ceased to be. It's expired and gone to meet its maker. This is a late parrot. It's a stiff. Bereft of life, it rests in peace. If you hadn't nailed it to the perch, it would be pushing up the daisies. It's rung down the curtain and joined the choir invisible. This is an ex-parrot!"

— John Cleese, Monty Python's Flying Circus

3.1 Introduction

Not all modifiers are created equal. *Green* parrots have all essential qualities of parrots, but *dead* parrots don't. For example, as vocally argued by the disgruntled costumer in Monty Python's famous Dead Parrot Sketch,[1] dead parrots make rather poor pet birds. In modifier-head constructions (that, for the purpose of this chapter, we restrict to right-headed adjective-noun and noun-noun constructions), modifiers are not simply picking a subset of the denotation of the head they modify, but they are often *distorting* the properties of the head in a radical manner.

These *modifier effects* on phrase meaning have been studied extensively by theoretical linguists, who have focused primarily on the extreme case of *intensional* modifiers such as *fake*, *alleged* and *toy*, where the phrase denotes something that is no longer (or is not necessarily) a *head* (a *toy gun* is not a *gun*). See McNally (2013) for a recent review of the linguistic literature. Cognitive scientists have looked at modification phenomena

[1] `http://en.wikipedia.org/wiki/Dead_Parrot_sketch`

within the general study of conceptual combination (see Chapter 12 of Murphy (2002) for an extensive review). The cognitive tradition has focused on how modification affects prototypicality: a *guppy* is the prototypical *pet fish*, but it is neither a typical *pet* nor a typical *fish* (Smith and Osherson, 1984). This line of research has highlighted how strong modification effects might be the rule, rather than the exception: Wisniewski (1997) reports that, when subjects were asked to provide the meaning for more than 200 novel modifier-head constructions, "70% [of the answers] involved the construal of a noun's referent as something other than the typical category named by the noun [head]." Indeed, recent research suggests that even the most stereotypical modifiers affect prototypicality, so that subjects are less willing to attribute to *quacking ducks* such obvious duck properties as *having webbed feet* (Connolly et al., 2007).

The impact of modification on phrase meaning is not only very interesting from a linguistic and cognitive perspective, but also important from a practical point of view, as it might affect expected entailment patterns: If *parrot* entails *pet*, then *lively parrot* also entails *pet*. However, as we saw above, *dead parrot* doesn't necessarily entail *pet* (at least not from the point of view of a disgruntled costumer who was just sold the corpse). Being able to track the impact that modifiers have on heads should thus have a positive effect on important tasks such as recognizing textual entailment, paraphrasing and anaphora resolution (Androutsopoulos and Malakasiotis, 2010; Dagan et al., 2009; Poesio et al., 2010).

Despite their theoretical and practical import, modification effects have been largely overlooked in computational linguistics, with the notable exception of Boleda et al. (Boleda et al., 2012, 2013), who only focused on the extreme case of intensional adjectives, studied a limited number of modifiers, and did not attempt to capture the graded nature of modification (a *dead parrot* is not a prototypical *animal*, but a *toy parrot* is not an *animal* at all).

In this Chapter, I will describe how we have built a large, publicly available data set of modifier-head phrases annotated with four kinds of modification-related subject ratings: whether the concept denoted by the phrase is an instance of the concept denoted by its head (is a *dead parrot* still a *parrot*?), to what extent it is a member of one of the larger categories the head belongs to (is it still a *pet*?), and typicality ratings for the same questions (how typical is a *dead parrot* as a *parrot*? and as a *pet*?).

Second, I will present our efforts to model the collected judgments computationally using DSM. In particular, we look at the *compositional* extension of distributional semantics, because we need representations not only for words, but also phrases, and we adopt the *asymmetric* similarity measures developed in the literature on lexical entailment (Kotlerman et al., 2010; Lenci and Benotto, 2012), because we are interested in an asymmetric relation (to what extent the concept denoted by the phrase is a good instance of the target class, and not *vice versa*). As far as we know, this is the first time these asymmetric measures are applied to composed representations (Baroni et al. (2012) experimented with entailment measures applied to phrase representations directly harvested from corpora, and not derived compositionally).

The setup of the task involves producing fine-grained inferences that make intensive use of human conceptual knowledge, both from the compositional side and the hierarchical organization side. We are thus evaluating distributional representations on a challenging setting where they could also potentially be very useful.

3.2 The Norwegian Blue Parrot data set

We introduce *Norwegian Blue Parrot* (NBP),[2] a new, large data set to explore modification effects. Given a **h**ead noun h and a **m**odifier adjective or noun m, NBP contains average membership and typicality ratings for the phrase mh both as an instance of h and as an instance of c (a broader **c**ategory h belongs to). As a control, we also present ratings for unmodified h as an instance of c (we will use them below to test similarity measures on their ability to capture the direction of the membership relation, and to zero in on the effect of modification vs. more general membership/typicality effects). We include, and indeed focus on, relations with broader categories because they are more prone to modification effects: Intuitively, a *dead parrot* is still a *parrot*, but it is, at the very least, an atypical *pet*. The statistics in Table 3.1, discussed below, confirm our intuition that subjects are more likely to assign lower scores with respect to a broader category than to the head category itself (although this is, no doubt, in part by construction, since we started constructing the dataset by mining examples where mh is atypical of c, not h). We collect both membership and typicality ratings because we expect them to have different implications for sound entailment. If x is not a member of class y, then x obviously does not entail y. However, if x is an atypical y, entailment still holds, but some typical properties of y might not carry over (e.g., in an anaphora resolution setting, we might still consider co-indexing *dead parrot* with *animal*, but not with *breathing creature*, despite the fact that *breathing* is a highly characteristic property of *animals*).

In order to make sure that NBP would contain a fair number of examples affected by strong modification effects, we first came up with a set of $\langle m, h, c\rangle$ tuples where, according to our own intuition, m makes h fairly atypical as an instance of c. For example, a *bottle* is a piece of *drinkware*. If we add the modifier *perfume*, we expect that,

[2]Available from `http://clic.cimec.unitn.it/composes/`

while subjects might still agree that a *perfume bottle* is a *bottle*, they should generally disagree on the statement that a *perfume bottle* belongs to the *drinkware* category. We refer to tuples of this sort (e.g., $\langle perfume, bottle, drinkware \rangle$) as *distorted* tuples in what follows.[3]

We then constructed a number of tuples that should not display a strong modification effect. In particular, in order to insure that any atypical rating we obtained on the distorted tuples could not be explained away by characteristics of m or h alone (rather than by their combination), for each distorted tuple we constructed a few more tuples with the same h and c but a different m, that we did not expect to be strongly distorting (e.g., $\langle plastic, bottle, drinkware \rangle$). Similarly, for each distorted tuple we generated a few more with the same m, but combined with (the same or different) h and c on which the m should not exert a strong effect ($\langle perfume, bottle, container \rangle$). In total, NBP is based on 489 distorted tuples and 1938 more matching tuples.

We constructed NBP to insure that it would contain many tuples displaying strong modification effects, and highly comparable tuples that do not feature such effects. An alternative approach would have been to rate phrases that were randomly selected from a corpus. This would have led to a dataset reflecting a more realistic distribution of modification effects, but it would not have guaranteed, for the same number of pairs, a fair amount of distorted tuples and comparable controls. We leave the study of the natural distribution of modification strength in text to further work.

To find inspiration for the tuples, we looked into various databases containing concepts organized by category, namely BLESS (Baroni and Lenci, 2011), ConceptNet (Speer and Havasi, 2013) and WordNet (Fellbaum, 1998). We insured that all words in our tuples occurred at least 200 times in the large corpus we describe below (phrases were not filtered by frequency, due to data sparseness). Finally, when looking for tuples matching the distorted ones, we made sure that the mh phrases in the new tuples have similar Pointwise Mutual Information to the corresponding phrases in the distorted tuple (or, where the latter were not attested in the corpus, similar m and h frequencies). Finding meaningful combinations among unattested or infrequent phrases was not an easy task and there was not always a perfect candidate. However, the phrases selected in this way yielded challenging items for which there is little or no direct corpus evidence, so that compositional models are required to account for them.

From each source tuple (e.g., $\langle plastic, bottle, drinkware \rangle$), we generated 3 instance-class combinations to be rated: $mh \rightarrow c$ (*plastic bottle* $\rightarrow$ *drinkware*), $mh \rightarrow h$ (*plastic bottle*

[3] When creating the tuples, we also used some adjectives that have been traditionally labeled as intensional by semanticists: *artificial*, *toy*, *former*.

$\rightarrow$ *bottle*), $h \rightarrow c$ (*bottle* $\rightarrow$ *drinkware*), for a total of 5,849 pairs, that constitute the final NBP data set (2,417 $mh \rightarrow c$ pairs, 2,115 $mh \rightarrow h$ pairs and 1,317 $h \rightarrow c$ pairs).[4]

For each of these pairs, we collected both membership and typicality ratings through two surveys on the CrowdFlower platform.[5] Subjects came exclusively from English speaking countries and no special qualifications were required from them. Membership ratings were collected by asking subjects whether the instance is a member of the class (formulated as a yes/no question). In a separate study, we asked subjects to rate how typical the instance is as member of the class on a 7-point scale. For both questions, we collected 10 judgments per pair and report their averages in NBP. For both surveys, we added 48 control pairs with an expected answer (yes/no for membership, high/low range for typicality), that the subjects had to provide in order for their ratings to be included in the final set ("gold standard" items in crowd-sourcing parlance). These controls included highly prototypical pairs (*dog* $\rightarrow$ *animal*), possibly with stereotypical modifiers (*beautiful rose* $\rightarrow$ *flower*), and unrelated pairs (*biology* $\rightarrow$ *dance*), also possibly under modification (*popular magazine* $\rightarrow$ *animal*).

We asked for binary rather than graded membership judgments because these are more in line with commonsense intuitions about category membership (we might naturally speak of *sparrows* being more typical birds than *penguins*, but it is strange to say that they are "more birds"). The standard view in the psychology of concepts (Hampton, 1991) is that membership judgments are the product of a hard threshold we impose on the typicality scale (x is not y if the typicality of x as y is below a certain, subject-dependent threshold), although under certain experimental conditions subjects can also conceptualize membership as a graded property (Kalish, 1995).

Membership and typicality ratings, especially in borderline cases such as those we constructed, are the output of complex cognitive processes where large inter-subject differences are expected, so it doesn't make sense to worry about "inter-annotator agreement" in this context. Still, several sanity checks indicate that, overall, our subjects understood our questions as we meant them, and behaved in a reasonably coherent manner. First, both average membership and typicality, ratings are significantly lower ($p < 0.001$) for the $mh \rightarrow c$ pairs deriving from those tuples that we manually labeled as distorted than for the non-distorted ones. Moreover, for membership, in 86% of the cases at least 8 over 10 subjects gave the same response. For typicality, the observed average rating standard deviation across pairs (1.2) is significantly below what expected by chance ($p < 0.05$), based on a simulated random rating distribution. Membership and typicality ratings are highly correlated, but not identical ($r = 0.76$)

[4]There is a larger number of $mh \rightarrow c$ pairs because different tuples can lead to the same $mh \rightarrow h$ or $h \rightarrow c$ combinations.

[5]http://crowdflower.com/

Table 3.1 reports mean membership and typicality scores in NBP. Both ratings are negatively skewed, that is, subjects had the tendency to respond assertively to the membership question and to give high typicality scores. This is not surprising: Because of the way NBP was constructed, there are about 4 tuples with no expected strong modification effect for each distorted tuple. Furthermore, except for the negative control items (not entered in NBP), our questions did not feature cases where a negative/low response would be entirely straightforward (of the "is a cat a building?" kind). We observe moreover that, in accordance with the intuition we discussed at the beginning of this section, the ratings are extremely high when the class is identical to the phrase head. On the other hand, the $mh \rightarrow c$ condition displays, as expected, the lowest averages, suggesting that this will be the most interesting type to model experimentally.

measure	$mh \rightarrow c$	$mh \rightarrow h$	$h \rightarrow c$	tot.
memb.	0.84 (0.2)	0.97 (0.1)	0.88 (0.2)	0.89 (0.2)
typ.	5.45 (1.1)	6.29 (0.6)	5.81 (1.0)	5.84 (1.0)

Table 3.1: NBP summary statistics: Mean average ratings and their standard deviations across pairs, itemized by instance-class type and in total. Membership values range from 0 to 1, typicality values from 1 to 7.

Table 3.2 presents a few example entries from NBP. The first block of the table illustrates cases with the highest possible membership and typicality scores. At the other extreme, the second block contains examples with very low membership and typicality. Interestingly, there are also cases, such as the ones in the third block of the table, where all subjects agreed on class membership, but the typicality scores are relatively low (we did not find clear cases of the opposite pattern, and indeed we would have been surprised to find highly typical instances of a class not being treated as members of the class).

Some examples in Table 3.2 illustrate an important design choice we made in constructing NBP, namely, to ignore the issue of whether potential modification effects are actually due to the modifier and the category pertaining to different *word senses* of the head term. One might argue, for example, that *egg* has a *food* sense and a *reproductive vessel* sense. The *human* modifier picks the second sense, and so, obviously, *human eggs* are judged as bad instances of *food*. While we see the point of this objection, we think it's impossible to draw a clear-cut distinction between discrete word senses (even in the rather extreme egg case, the eggs we eat are reproductive vessels from a chicken point of view!). This has been long recognized in the linguistic and cognitive literature (Kilgarriff, 1997; Murphy, 2002), and even by the computational word sense disambiguation community, that is currently addressing the continuous nature of polysemy by shifting to the lexical-substitution-in-context task (McCarthy and Navigli, 2009). Context

instance	*class*	*memb.*	*typ.*
top membership, top typicality			
gourmet soup	food	1.00	7.00
huge tiger	predator	1.00	7.00
sugared soda	drink	1.00	7.00
live fish	animal	1.00	7.00
Thai rice	rice	1.00	7.00
silver spoon	spoon	1.00	7.00
low membership, low typicality			
fatal shooting	sport	0.20	1.40
human egg	food	0.40	1.50
perfume bottle	drinkware	0.10	1.30
explosive vest	commodity	0.30	1.90
lemon water	chemical	0.20	1.60
creamy rice	bean	0.20	1.30
top membership, (relatively) low typicality			
sick tuna	tuna	1.00	3.20
explosive vest	vest	1.00	3.50
perforated sieve	tool	1.00	4.20
bottled oxygen	substance	1.00	4.30
grilled trout	creature	1.00	4.40
educational toy	amusement	1.00	4.50

Table 3.2: Instance-class pairs illustrating various combinations of membership and typicality ratings in NBP.

provides fundamental cues to disambiguating polysemous words, and noun modifiers typically act as important disambiguating contexts for the nouns. Thus, we think that it is more productive for computational systems to handle modifier-triggered disambiguation as a special case of the more general class of modification effects, than to engage in the quixotic pursuit to determine, *a priori*, what's the boundary between a word-sense and a "pure" modification effect. Note in Table 3.2 that *grilled trout* was unanimously rated by subjects as an instance of the *creature* category, despite the fact that the cooking-related *grilled* modifier cues a classic shift from an *animal* (and thus *creature*) sense to *food* (Copestake and Briscoe, 1995). Examples like this suggest that our agnosticism is warranted.

3.3 Methods

3.3.1 Composition models

We experiment with many ways to derive a phrase vector by combining the vectors of its constituents. In particular, we explore the **wadd**, **dil**, **mult**, **fulladd** and **lexfunc** compositional models (see Section 2.2 for a description).

We use the DISSECT toolkit[6] to estimate the parameters of the composition methods and derive phrase vectors. In particular, DISSECT finds optimal parameter settings by learning to approximate corpus-extracted phrase vector examples with least-squares methods (Dinu et al., 2013). We use as training examples all the modifier-head phrases that contain a modifier of interest and occur at least 50 times in our source corpus (see Section 3.3.3 below).

3.3.2 Asymmetric similarity measures

Several measures to identify word pairs that stand in an instance-class relationship by comparing their vectors have been proposed in the recent distributional semantics literature (Kotlerman et al., 2010; Lenci and Benotto, 2012; Weeds et al., 2004).[7] While the task of deciding if u is in class v is typically framed (also by distributional semanticists) in binary, yes-or-no terms, all proposed measures return a continuous numerical score.[8] Consequently, we conjecture that they might be well-suited to capture the graded notions of class membership and typicality we recorded in NBP.[9]

In what follows, we use $w_x(f)$ to denote the weight (value) of feature (dimension) f in the distributional vector of term x. F_x denotes the set of features (dimensions) in the vector of x such that $w_x(f) > t$, where t is a predefined threshold to decide whether a feature is active.[10] Importantly, all measures assume non-negative values.

Most asymmetric measures proposed in the literature build upon the *distributional inclusion hypothesis*, stating that "if u is a semantically narrower term than v, then a significant number of salient distributional features of u is included in the feature vector of v as well" (Lenci and Benotto, 2012). In our terminology, u is the potential instance, and v is the class. We re-implement all the measures adopted by Lenci and Benotto, namely **weedsprec**, **cosweeds**, **clarkede** and **invcl** (see their paper for the original references):

[6] `http://clic.cimec.unitn.it/composes/toolkit/`

[7] We speak of "instance-class relations" in a very broad and loose sense, to encompass classic relations such as hyponymy but also the fuzzier notion of lexical entailment.

[8] SVM classifiers have also been shown by Baroni et al. (2012) to be well-suited for entailment detection, but they do not naturally return continuous scores.

[9] Subjects had to answer a yes/no question concerning class membership, but by averaging their response we derive continuous membership scores.

[10] The obvious choice for t is 0. However, when working with the low-rank spaces described in Section 3.3.3 below, we set t to 0.1, since after SVD/NMF smoothing we observe widespread low-frequency noise.

$$weedsprec(u,v) = \frac{\sum_{f \in F_u \cap F_v} w_u(f)}{\sum_{f \in F_u} w_u(f)} \tag{3.1}$$

$$cosweeds(u,v) = \sqrt{weedsprec(u,v) \times cosine(u,v)} \tag{3.2}$$

$$clarkede(u,v) = \frac{\sum_{f \in F_u \cap F_v} \min(w_u(f), w_v(f))}{\sum_{f \in F_u} w_u(f)} \tag{3.3}$$

$$invcl(u,v) = \sqrt{clarkede(u,v) \times (1 - clarkede(u,v))} \tag{3.4}$$

The cosweeds formula combines weedsprec with the widely used symmetric *cosine* measure:

$$cosine(u,v) = \frac{\sum_{f \in F_u \cap F_v} w_u(f) \times w_v(f)}{\sqrt{\sum_{f \in F_u} w_u(f)^2} \times \sqrt{\sum_{f \in F_v} w_v(f)^2}} \tag{3.5}$$

Finally, we experiment with the carefully crafted **balapinc** measure of Kotlerman et al. (2010):

$$balapinc(u,v) = \sqrt{lin(u,v) \cdot apinc(u,v)} \tag{3.6}$$

where the *lin* term is computed as follows:

$$lin(u,v) = \frac{\sum_{f \in F_u \cap F_v} w_u(f) + w_v(f)}{\sum_{f \in F_u} w_u(f) + \sum_{f \in F_v} w_v(f)} \tag{3.7}$$

The balapinc score is the geometric average of a symmetric similarity measure (*lin*) and the strongly asymmetric *apinc* measure, that takes large values when dimensions with high values in the vector of the more specific term are also high in the vector of the more general term (refer to Kotlerman et al. (2010) for the apinc formula).

3.3.3 Distributional semantic spaces

We use count models to produce our distributional vectors because their deterministic training procedure makes it easier to train compositional models. We extract co-occurrence information from a corpus of about 2.8 billion words obtained by concatenating ukWaC,[11] Wikipedia[12] and the British National Corpus.[13] With DISSECT, we build co-occurrence vectors for the top 20K most frequent lemmas in the source corpus (plus any NBP term missing from this list). We treat the top 10K most frequent lemmas as context elements. We consider context windows of 2 and 20 words on the two sides of the targets. We weight the vectors by non-negative Pointwise Mutual Information and Local Mutual Information (Evert, 2005). We experiment with vectors in the resulting full-rank (10K-dimensional) semantic spaces as well as with vectors in spaces of ranks 100 and 300. Rank reduction is performed by applying the Singular Value Decomposition (Golub and Van Loan, 1996) or Non-negative Matrix Factorization (Lee and Seung, 2000). It is customary to represent the output of these operations directly in a dense low-dimensional space. However, the asymmetric similarity measures we use assume sparse vectors (or the "inclusion" criterion would be meaningless), so we project back the outcome of SVD and NMF to sparse 10K-dimensional but low-rank spaces. In total, we explore 20 distinct semantic spaces.

We also collect co-occurrence vectors for the phrases needed to estimate the composition method parameters (see Section 3.3.1 above). We use DISSECT's "peripheral space" option to project the phrase raw count vectors into the various spaces without affecting their structure.

Due to memory constraints, we restrict evaluation in the full-rank spaces to the *wadd* and *mult* models.

3.4 Experiments

Given the methods described above, the main question we want to answer is: Which combination of compositional model and asymmetric similarity measure yields a better fit for the data in the NBP dataset?

We start however with a sanity check on the ability of the measures to capture the *direction* of the instance-class membership relation. Even a measure that is good at

[11]http://wacky.sslmit.unibo.it
[12]http://en.wikipedia.org
[13]http://www.natcorp.ox.ac.uk

capturing degrees of membership/typicality won't be of much practical use if it is not able to tell us which item in a pair is the instance and which is the class.

Detecting membership direction As described in Section 3.2 above, NBP also contains single-word $h \rightarrow c$ pairs (*parrot*$\rightarrow$*pet*). We extracted the subset of those that all judges considered to be in the category membership relation, and we checked them manually to make sure that the direction was one-way only. This resulted in a set of 639 pairs where the membership relation holds unidirectionally. We tested all combination of semantic spaces (Section 3.3.3) and asymmetric similarity measures (Section 3.3.2) on the task of assigning a higher score to the pairs in the $h \rightarrow c$ (vs. $c \rightarrow h$) direction (e.g., ($score(parrot \rightarrow pet) > score(pet \rightarrow parrot)$). Table 3.3 reports, for each measure, the number of spaces in which the measure was able to predict membership direction significantly better than chance (binomial test, $p < 0.05$). We report results on full- and low-rank (SVD, NMF) spaces separately since, as discussed above, for most composition models we can only use the latter. We observe that all measures are able to significantly detect directionality in at least some spaces. For all the analyses below, we exclude from further testing the space-measure combinations that failed to pass this sanity check, since they are clearly failing to capture properties pertaining to the instance-class relation (if a combination is not able to tell that it is a *parrot* that is a *pet*, and not *vice versa*, there is no point in asking if the same combination is able to model how typical a *dead parrot* is as a *pet*).

clarkede	weedsprec	balapinc	cosweeds	invcl
		Low-rank spaces		
10	8	11	8	7
		Full-rank spaces		
2	4	4	4	2

TABLE 3.3: Number of spaces (over totals of 16 low-rank and 4 full-rank spaces) in which each measure was able to predict class membership direction significantly above chance.

Modeling typicality ratings of $mh \rightarrow c$ pairs Next, for each of the remaining spaces, we first performed composition as described in Section 3.3.1 above to build the representations for the nominal phrases in the NBP dataset, and then computed asymmetric similarity scores for pairs made of a phrase and the corresponding potential class.

We computed the correlations between mean human membership or typicality ratings and the scores produced with each combination of composition model, similarity measure and space. The resulting performance profiles for membership and typicality are very highly correlated ($r = .99$), and we thus report only the latter. We leave it to further work to devise measures that are more specifically tuned to capture membership or typicality.

Table 3.4 reports the top correlation coefficients between typicality judgments and scores of each $mh \to c$ pair (*dead parrot→pet*) across spaces, organized by measures and composition methods. The best correlation is achieved with the weedsprec measure using the mult composition model in a full-rank space (precisely that of context window size 2 and ppmi weighting). Recall that mult returns the component-wise product of the vectors it combines. Thus, modification under mult is carried out by picking only those features of the head that are also present in the modifier, and enhancing them by a factor given by the modifier's feature value. The weedsprec measure is then given by the weighted proportion of active features in mh that are also active in c. Therefore, the more the modifier shares features with the parent category, the higher weedsprec will be. This might explain why weedsprec is a good fit for the mult model in measuring degrees of category typicality.

Looking at composition methods, there is no evidence that the more complex, matrix-based fulladd and lexfunc approaches are performing any better than the simple multiplicative and additive methods. Indeed, mult shows the most consistent overall performance, confirming the conclusion of Blacoe and Lapata (2012) that, at the present time, when it comes to composition, "simpler is better". A related point emerges from the comparison of the low- and full-rank results for mult and wadd. The smoothing process due to dimensionality reduction is quite disruptive for the current asymmetric measures, that are based on feature inclusion. This is a further reason to stick to simpler composition methods, that can be applied directly in the full-rank spaces.

Regarding the measures themselves, we see that cosweeds, that balances weedsprec with the classic cosine score, is the most robust, returning good results across all composition methods. On the other hand, the related clarkede and invcl measures turn out to be quite brittle.

The highly significant correlations show that the measures do capture to some extent the patterns of variance in the data. However, when considering potential practical applications, even the highest reported correlation (.39) is certainly not impressive, indicating that there is plenty of room for further research into developing better composition methods and/or membership/typicality measures.

Focusing on the modifier effect for $mh \to c$ pairs The typicality judgment for *dead parrot* as a *pet* is influenced by two factors: how typical *parrots* are as *pets*, and

	clarkede	weedsprec	balapinc	cosweeds	invcl
		Low-rank spaces			
dil	9*	15*	16*	19*	8*
fulladd	17*	16*	12*	24*	−3
lexfunc	17*	12*	12*	27*	−2
mult	13*	19*	19*	29*	12*
wadd	14*	14*	16*	27*	−2
		Full-rank spaces			
mult	9*	**39***	**33***	**36***	**15***
wadd	**30***	34*	31*	35*	14*

TABLE 3.4: Percentage Pearson r between asymmetric similarity measures and $mh \rightarrow c$ typicality ratings. $*p < 0.001$

	clarkede	weedsprec	balapinc	cosweeds	invcl
		Low-rank spaces			
dil	5	−1	−1	−2	7*
fulladd	10*	7*	5+	7+	−2
lexfunc	**15***	9*	10*	18*	−2
mult	4+	14*	13*	15*	**9***
wadd	7+	7*	9*	12+	−2
		Full-rank spaces			
mult	1	**25***	**21***	**24***	5+
wadd	11*	18*	13*	20*	2

TABLE 3.5: Percentage Pearson r between asymmetric similarity measures and $mh \rightarrow c$ typicality ratings where $h \rightarrow c$ scores have been partialed out. $*p < 0.001$, $+p < 0.05$

how much more or less typical *dead parrots* are as *pets*, as opposed to *parrots* in general. A good model must be able to capture both factors (and this is what we tested above). However, we are also interested in assessing to what extent the models are capturing the modification effectproper, as opposed to the overall degree of typicality of the h concept as member of the c category. To focus on the modification factor, we partialed out the $h \rightarrow c$ (*parrot→pet*) ratings from the $mh \rightarrow c$ (*dead parrot→pet*) ratings and from the corresponding model scores (that is, we correlated the residuals of $mh \rightarrow c$ ratings and model-produced scores after regressing the $h \rightarrow c$ ratings on both). The results are shown in Table 3.5. Correlations are lower overall, but the general picture from the previous analysis still holds, confirming that the computational models are (also) capturing modifier effects. Interestingly, wadd, dil and fulladd generally undergo larger performance drops than mult and lexfunc. Evidently, models like the latter, in which the modifier selects the relevant features from the head, are better suited to explain modification than the former, in which the modifier features are just added to those of the head by means of a linear combination.

Modeling typicality ratings of $mh \rightarrow h$ pairs We repeated the first analysis for pairs of the type $mh \rightarrow h$ (*dead parrot→parrot*). The results, shown in Table 3.6, are lower than in the previous analysis. This is probably due to the fact that, as discussed in

	clarkede	weedsprec	balapinc	cosweeds	invcl
			Low-rank spaces		
dil	2	-1	-2	-3	4
fulladd	5+	5+	2	1	-1
lexfunc	**14***	8*	14*	17*	-1
mult	3	-	13*	15*	**5+**
wadd	6+	8*	7+	6	-3
			Full-rank spaces		
mult	-2	-	**18***	**19***	-2
wadd	7*	**13***	7*	12*	-2

TABLE 3.6: Percentage Pearson r between asymmetric similarity measures and $mh \rightarrow h$ typicality ratings. $*p < 0.001$, $+p < 0.05$

Section 3.2, when the very same concept is used as phrase head and category, judgments are subject to a strong ceiling effect, and none of our measures is designed to flatten out above a certain threshold. Indeed, if we measure the skewness of the typicality ratings,[14] we obtain that, while for $h \rightarrow c$ and $mh \rightarrow c$ the skewness is of -1.9 and -1.5, respectively, for $mh \rightarrow h$ it gets to -3.9.

In any case, the results confirm the brittleness of the clarkede and invcl measures. The linguistically motivated lexfunc model emerges here as a competitive alternative to the simpler models. Still, the best results are obtained with mult and cosweeds (on the full-rank, context window size 20, ppmi weighted space). Notably, weedsprec applied to a pair of the type $mh \rightarrow h$, where the phrase is constructed using the mult model, results in a constant value of 1, whatever the modifier and the head noun is. This is due to the fact that the features of a phrase composed using mult are a subset of the features of the head,[15] and in this case the head is the same as the category. Therefore, by definition, weedsprec yields a score of 1 for every pair, the variance is null and hence the correlation is undefined. As a consequence, in this case cosweeds, which is the geometric mean between weedsprec and cosine, reduces to cosine similarity! The latter might be effective in capturing the degree of similarity between the phrase and its potential category but, as a symmetric measure, it cannot, alone, provide a full account of category typicality effects.

[14]A skewness factor of 0 means that the distribution is balanced around the mean, while the more negative the coefficient is, the more the left tail is longer and the distribution is concentrated to the right (toward high typicality values in our case).

[15]In set notation: $F_u \cap F_v = F_u$ since $F_u \subseteq F_v$

3.5 Conclusion

We introduced the challenge of quantifying the impact of modification on the meaning of noun phrases, presenting a new dataset that collects membership and typicality ratings for modifier-head phrases with respect to the category represented by the head as well as a broader category. Since accounting for modifier distortion requires semantic representations of phrases and modeling graded judgments, we consider this an ideal testbed for compositional distributional semantics.

In the interaction between compositional models and directional similarity measures, we have observed that simpler models yield better results. Specifically, mult and wadd are economical composition models than can be applied on full-rank spaces, which in turn work best with our similarity measures.

Psychologists studying modification effects in concept combination have proposed models that are usually quite complex, relying on hand-crafted feature definitions and making very strong assumptions about the combination process (see for example Cohen and Murphy (1984), Smith et al. (1988)). Some of these assumptions have led other researchers to argue that prototypes do not compose at all (Connolly et al., 2007). In contrast, the approach we borrow from distributional semantics, while only mildly successful for now, has the advantage of being very simple both in its construction and application, and in the assumptions that it makes.

Also notable is that we are putting under the same umbrella tasks that have been traditionally tackled separately. For example, among the effects present in the dataset, we can find both word sense disambiguation (see discussion at the end of Section 3.2) and what Murphy (2002) calls "knowledge effects" (e.g., a *plane* makes a very good *machine*, but a *paper plane* doesn't). Moreover, these effects can also interact (people know that a *human egg* is actually a single, small cell, and hence not even cannibals would consider it satisfactory food). We can thus explore the empirical question of whether all these related phenomena can be tackled together, with a single model accounting for all of them.

In conclusion, the challenge that we introduced brings together concept combination and non-subsective modification phenomena studied in psychology and theoretical linguistics, and tries to handle them with the standard machinery of computational linguistics. This challenge has proved quite difficult for current tools, but this is exactly what we expected in the first place. Our goal, from the outset, was to create a task that could help us delimiting the boundaries of computational methods for characterizing human concepts, while delimiting, at the same time, the notion of human concepts itself.

Chapter 4

Boolean Distributional Semantic Models

"You need Boolean structure."

— Denis Paperno

4.1 Introduction

You might have never heard of a *jabuticaba* before. Nevertheless, if I tell you that it is a type of *fruit*, then you will probably be able to infer many of its properties. This simple example demonstrates the value of a hierarchical organization of concepts. Yet, regardless of some proposals such as the asymmetric measures we have relied upon in the last chapter, it is still unclear how can we account for such organization in DSMs. More generally, we would like to be able to detect an entailment relation between two distributed representations of arbitrary linguistic expressions.

In formal semantics, entailment is well-characterized as an *inclusion* relation between the sets (of the relevant type) denoted by words or other linguistic expressions, e.g., sets of possible worlds that two propositions hold of (Chierchia and McConnell-Ginet, 2000, 299). In finite models, a mathematically convenient way to represent these denotations is to encode them in *Boolean* vectors, i.e., vectors of 0s and 1s (Sasao, 1999, 21). Given all elements e_i in the domain in which linguistic expressions of a certain type denote, the Boolean vector associated to a linguistic expression of that type has 1 in position i if $e_i \in S$ for S the set denoted by the expression, 0 otherwise. An expression a entailing b will have a Boolean vector including the one of b, in the sense that all positions occupied by 1s in the b vector are also set to 1 in the a vector. Very general expressions (entailing nearly everything else) will have very dense vectors, whereas very specific expressions will have very sparse vectors. The negation of an expression a will denote a "flipped" version of the a Boolean vector. *Vice versa*, two expressions with at least partially compatible meanings will have some overlap of the 1s in their vectors; conjunction and disjunction are carried through with the obvious bit-wise operations, etc.

Despite all these theoretical advantages, formal models lack the large-scale inductive power of distributional semantic models. To narrow the gap between the two, we create Boolean meaning representations that build on the wealth of information inherent in distributional vectors of words (and sentences). More precisely, we use word (or sentence) pairs labeled as entailing or not entailing to train a mapping from their distributional representations to Boolean vectors, enforcing feature inclusion in Boolean space for the entailing pairs. By focusing on inducing Boolean representations that respect the inclusion relation, our method is radically different from recent supervised approaches that learn an entailment classifier directly on distributional vectors, without enforcing inclusion or other representational constraints. We show, experimentally, that the method is competitive against state-of-the-art techniques in lexical entailment, improving on them in sentential entailment, while learning more effectively from less training data. This is crucial for practical applications that involve bigger and more diverse data than the focused test sets we used for testing. Moreover, extensive qualitative analysis reveals several interesting properties of the Boolean vectors we induce, suggesting that they are representations of greater generality beyond entailment, that might be exploited in further work for other logic-related semantic tasks.

4.2 Related work

Entailment in distributional semantics Due to the lack of methods to induce the relevant representations on the large scale needed for practical tasks, the Boolean structure defined by the entailment relation is typically not considered in efforts to automatically recognize entailment between words or sentences (Dagan et al., 2009). On the other hand, some researchers relying on distributional representations of meaning have attempted to apply various versions of the notion of feature inclusion to entailment detection. This is based on the intuitive idea – the so-called *distributional inclusion hypothesis* – that the features (vector dimensions) of a hypernym and a hyponym should be in a superset-subset relation, analogously to what we are trying to achieve in the Boolean space we induce, but directly applied to distributional vectors (Geffet and Dagan, 2005; Kotlerman et al., 2010; Lenci and Benotto, 2012; Weeds et al., 2004). Indeed, this is the approach we have taken in the previous chapter to recognize and entailment relation between nominal phrases and their super-ordinates (refer to Section 3.3.2 for a more in-detail description of these measures). On the other hand, it has been noticed that distributional context inclusion defines a Boolean structure on vectors just as entailment defines a Boolean structure on formal semantic representations (Clarke, 2012). However, the match between context inclusion and entailment is far from perfect.

First, distributional vectors are real-valued and contain way more nuanced information than simply inclusion or exclusion of certain features. Second, and more fundamentally, the information encoded in distributional vectors is simply not of the right kind since "feature inclusion" for distributional vectors boils down to contextual inclusion, and there is no reason to think that a hypernym should occur in all the contexts in which its hyponyms appear. For example, *bark* can be a typical context for *dog*, but we don't expect to find it a significant number of times with *mammal* even in a very large corpus. In practice distributional inclusion turns out to be a weak tool for recognizing the entailment relation (Erk, 2009; Santus et al., 2014) because denotational and distributional inclusion are independent properties.

More recently, several authors have explored supervised methods. In particular, Baroni et al. (2012), Roller et al. (2014) and Weeds et al. (2014) show that a Support Vector Machine trained on the distributional vectors of entailing or non-entailing pairs outperform the distributional inclusion measures. In our experiments, we will use this method as the main comparison point. The similarly supervised approach of Turney and Mohammad (2014) assumes the representational framework of Turney (2012), and we do not attempt to re-implement it here.

Very recently, other properties of distributional vectors, such as entropy (Santus et al., 2014) and topical coherence (Rimell, 2014), have been proposed as entailment cues. Since they are not based on feature inclusion, we see them as complementary, rather than alternative to our proposal.

Formal and distributional semantic models We try to derive a structured representation inspired by formal semantic theories from data-driven distributional semantic models. Combining the two approaches has proven a hard task. Some systems adopt logic-based representations but use distributional evidence for predicate disambiguation (Lewis and Steedman, 2013) or to weight probabilistic inference rules (Beltagy et al., 2013; Garrette et al., 2013). Other authors propose ways to encode aspects of logic-based representations such as logical connectives and truth values (Grefenstette, 2013) or predicate-argument structure (Clark and Pulman, 2007) in a vector-based framework. These studies are, however, entirely theoretical. Rocktäschel et al. (2015) expand on the first, allowing for some generalization to unseen knowledge, by introducing some degree of fuzziness into the representations of predicates and terms. Still, this work does not attempt to map concepts to a logic-based representation nor tries to exploit the wealth of information contained in distributional vectors.

Socher et al. (2013), Bordes et al. (2012) and Jenatton et al. (2012) try to discover unseen facts from a knowledge base, which can be seen as a form of inference based on a restricted predicate logic. To do so, they build vector representations for entities, while relations are represented through classifiers. Only Socher et al. (2013) harness distributional vectors, and just as initialization values. The others, unlike us, do not build on independently-motivated word representations. Moreover, since the representations are learned from entities present in their knowledge base, one cannot infer the properties of unseen concepts.

In the spirit of inducing a variety of logical relations and operators (including entailment), Bowman (2013) applies a softmax classifier to the combined distributional representation of two given statements, which are in turn learned compositionally in a supervised fashion in order to guess the relation between them. The paper, however, only evaluates the model on a small restricted dataset, and it is unclear whether the method would scale to real-world challenges.

None of the papers with concrete implementations reviewed above tries, like us, to learn a Boolean structure where entailment corresponds to inclusion. A paper that does attempt to exploit a similar idea is Young et al. (2014), which also uses the notion of *model* from Formal Semantics to recognize entailment based on denotations of words and phrases. However, since the denotations in their approach are ultimately derived from human-generated captions of images, the method does not generalize to concepts that are not exemplified in the training database.

Finally, a number of studies, both theoretical (Baroni et al., 2014a; Coecke et al., 2010) and empirical (Paperno et al., 2014; Polajnar et al., 2014), adapt compositional methods from formal semantics to distributional vectors, in order to derive representations of phrases and sentences. This line of research applies formal operations to distributional representations, whereas we derive formal-semantics-like *representations* from distributional ones. Below, we apply our method to input sentence vectors constructed with the composition algorithm of Paperno et al. (2014).

4.3 The Boolean Distributional Semantic Model

We build the Boolean Distributional Semantic Model (BDSM) by mapping real-valued vectors from a distributional semantic model into Boolean-valued vectors, so that feature

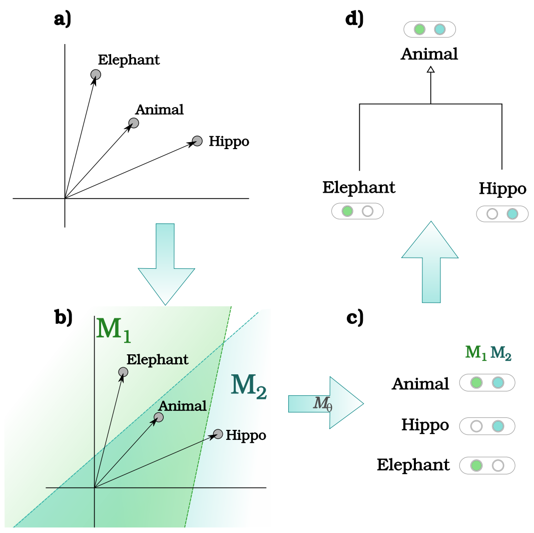


FIGURE 4.1: The BDSM architecture. a) Input distributional space b) Training of a Mapping M where each output dimension M_i can be seen as a (linear) cut in the original distributional space. c) Output representations after mapping. d) Fragment of the Boolean structure with example output representations.

inclusion in Boolean space corresponds to entailment between words (or sentences). That is, we optimize the mapping function so that, if two words (or sentences) entail each other, then the more specific one will get a Boolean vector included in the Boolean vector of the more general one. The is illustrated in Figure 4.1.

Our model differs crucially from a neural network with a softmax objective in imposing a strong bias on the hypothesis space that it explores. In contrast to the latter, it only learns the weights corresponding to the mapping, while all other operations in the network (in particular, the inference step) are fixed in advance. The goal of such a bias is to improve learning efficiency and generalization using prior knowledge of the relation that the model must capture.

I will now discuss how the model is formalized in an incremental manner. The goal of the model is to find a function M_Θ (with parameters Θ) that maps the distributional representations into the Boolean vector space. To facilitate optimization, we relax the image of this mapping to be the full $[0, 1]$ interval, thus defining $M_\Theta : \mathbb{R}^N \mapsto [0, 1]^H$. This mapping has to respect the following condition as closely as possible: For two given words (or other linguistic expressions) p and q, and their distributional vectors v_p and v_q, all the active features (i.e., those having value close to 1) of $M_\Theta(v_p)$ must also be active in $M_\Theta(v_q)$ if and only if $p \Rightarrow q$.

To find such a mapping, we assume training data in the form of a sequence $[(p_k, q_k), y_k]_{k=1}^{m}$ containing both positive ($p_k \Rightarrow q_k$ and $y_k = 1$) and negative pairs ($p_k \not\Rightarrow q_k$ and $y_k = 0$).

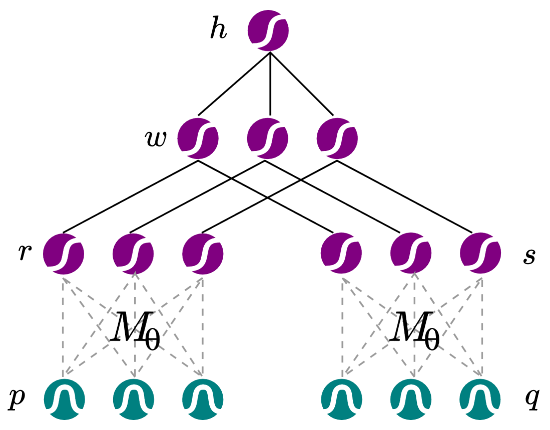


FIGURE 4.2: Schematic view of the entailment hypothesis function h_Θ. Solid links represent calculations that are fixed during learning, while dashed links represent the parameters Θ, which are being learned. The p and q input distributional vectors corresponding to each data point are fixed, r and s are their respective mapped Boolean representations. The w layer is a feature-inclusion detector and h is the final entailment judgment produced by the network.

More concretely, the mapping M_Θ is defined as a sigmoid function applied to a linear transformation: $M_\Theta(x) = g(Wx + b)$ and $g(x) = \frac{1}{1+e^{\frac{-x}{t}}}$, where t stands for an extra "temperature" parameter. We represent the $W \in \mathbb{R}^{H\times N}$, $b \in \mathbb{R}^H$ parameters succinctly by $\Theta = [W, b]$. We learn the mapping by minimizing the difference between the model's entailment predictions (given by a function h_Θ) and the training targets, as measured by the MSE:

$$J(\Theta) = \frac{1}{2}\sum_{k=1}^{m}\left(h_\Theta(p_k, q_k) - y_k\right)^2 \tag{4.1}$$

The calculation of $h_\Theta(p, q)$ involves a series of steps that can be construed as the architecture of a neural network, schematically represented in Figure 4.2. Recall that the output value of this function represents the model's prediction of the truth value for $p_k \Rightarrow q_k$. Here is an outline of how it is calculated. For each pair of words (or sentences) (p, q) in the training set, we map them onto their (soft) boolean correlates (r, s) by applying M_Θ to their corresponding distributional vectors. Next, we measure whether features that are active in r are also active in s (analogously to how Boolean implication works), obtaining a soft Boolean vector w. Finally, the output of h can be close to 1 only if all values in w are also close to 1. Thus, we compute the output value of h as the conjunction across all dimensions in w.

More precisely, $h_\Theta(p, q)$ is obtained as follows. The passage from the first to the second layer is computed as $r_\Theta = M_\Theta(v_p)$ and $s_\Theta = M_\Theta(v_q)$. Next, we compute whether the features that are active in r_Θ are also active in s_Θ. Given that we are working in the

$[0, 1]$ range, we approximate this operation as $w_{\Theta i} = \max(1 - r_{\Theta i}, s_{\Theta i})$[1]. It is easy to see that if $r_{\Theta i} = 0$, then $w_{\Theta i} = 1$. Otherwise, $s_{\Theta i}$ must also be equal to 1 for $w_{\Theta i}$ to be 1. Finally, we compute $h_\Theta = \min_i w_{\Theta i}$. This is a way to compute the conjunction over the whole previous layer, thus checking whether all the features of r_Θ are included in those of s_Θ[2].

Finally, to allow for better generalization, the cost function is extended with two more components. The fist one is a L2 regularization term weighted by a parameter λ. The second one is a term that enforces sparsity of the resulting representations based on some desired level ρ.

4.3.1 Assessing entailment with BDSM

During training, positive pairs $p \Rightarrow q$ are required to satisfy full feature inclusion in their mapped representations (all the active features of $M_\Theta(v_p)$ must also be in $M_\Theta(v_q)$). At test time, we relax this condition to grant the model some flexibility. Concretely, entailment is quantified by the BI ("Boolean Inclusion") function, counting the proportion of features in the antecedent that are also present in the consequent after binarizing the outputs:

$$BI(u, v) = \frac{\sum_i \mathrm{rnd}(M_\Theta(u)_i)\,\mathrm{rnd}(M_\Theta(v)_i)}{\sum_i \mathrm{rnd}(M_\Theta(u)_i)}$$

where $\mathrm{rnd}(x) = \mathbb{1}[x > 0.5]$. The 0.5 threshold comes from construing each of the features in the output of M as probabilities. Of course, other formulas could be used to quantify entailment through BDSM, but we leave this to further research.

Since BI returns continuous values, we use development data to calculate a threshold e above which an entailment response is returned.

4.4 Evaluation setup

4.4.1 Distributional semantic spaces

Our approach is agnostic to the kind of distributional representation used, since it doesn't modify the input vectors, but builds on top of them. Still, it is interesting to test whether

[1]In practice, we use a differentiable approximation given by $\max(x, y) \approx \frac{\log(e^{Lx}+e^{Ly})}{L}$, where L is a sufficiently large number. We set $L = 100$, which yields results accurate enough for our purposes.

[2]Analogously, we use the differentiable approximation given by $min(w_\theta) = -\log(\frac{\sum_i e^{-Lw_{\theta i}}}{L})$

specific kinds of distributional vectors are better suited to act as input to BDSM. For our experiments, we use both the **count** and **predict** distributional semantic vectors of Baroni et al. (2014b).[3] These vectors were shown by their creators to reach the best average performance (among comparable alternatives) on a variety of semantic relatedness/similarity tasks, such as synonymy detection, concept categorization and analogy solving. If the same vectors turn out to also serve as good inputs for constructing Boolean representations, we are thus getting the best of both worlds: distributional vectors with proven high performance on relatedness/similarity tasks which can be mapped into a Boolean space to tackle logic-related tasks. We also experiment with the pre-trained vectors from **TypeDM** (Baroni and Lenci, 2010),[4] which are built by exploiting syntactic information, and should have different qualitative properties from the window-based approaches.

The count vectors of Baroni and colleagues are built from a 2-word-window co-occurrence matrix of 300k lower-cased words extracted from a 2.8 billion tokens corpus. The matrix is weighted using positive Pointwise Mutual Information (Church and Hanks, 1990). We use the full 300k×300k positive PMI matrix to compute the asymmetric similarity measures discussed in the next section, since the latter are designed for non-negative, sparse, full-rank representations. Due to efficiency constraints, for BDSM and SVM (also presented next), the matrix is reduced to 300 dimensions by Singular Value Decomposition (Schütze, 1997). The experiments of Baroni et al. (2014b) with these very same vectors suggest that SVD is *lowering* performance somewhat. So we are, if anything, giving an advantage to the simple asymmetric measures.

The predict vectors are built with the word2vec tool (Mikolov et al., 2013) on the same corpus and for the same vocabulary as the count vectors, using the CBOW method. They are constructed by associating 400-dimensional vectors to each word in the vocabulary and optimizing a single-layer neural network that, while traversing the training corpus, tries to predict the word in the center of a 5-word window from the vectors of those surrounding it. The word2vec subsampling parameter (that downweights the impact of frequent words) is set to $1e^{-5}$.

Finally, TypeDM vectors were induced from the same corpus by taking into account the dependency links of a word with its sentential collocates. See Baroni and Lenci (2010) for details.

Composition methods For sentence entailment (Section 4.6), we need vectors for sentences, rather than words. We derive them from the count vectors compositionally

[3]`http://clic.cimec.unitn.it/composes/semantic-vectors.html`
[4]`http://clic.cimec.unitn.it/dm`

(see Section 2.2) in two different ways. First, we use the additive model (**add**), under which we sum the vectors of the words they contain to obtain sentence representations (Mitchell and Lapata, 2010). This approach, however, does not take into account word order, which is of obvious relevance to determining entailment between phrases. For example, *a dog chases a cat* does not entail *a cat chases a dog*, whereas each sentence entails itself. Therefore, we also used sentence vectors derived with the linguistically-motivated "practical lexical function" model (**plf**), that takes syntactic structure and word order into account (Paperno et al., 2014). In short, words acting as argument-taking functions (such as verbs) are not only associated to vectors, but also to one matrix for each argument they take (e.g., each transitive verb comes with a subject and an object matrix). Vector representations of arguments are recursively multiplied by function matrices, following the syntactic structure of a sentence. The final sentence representation is obtained by summing all the resulting vectors. We used pre-trained vector and matrix representations provided by Paperno and colleagues. Their setup is very comparable to the one of our count vectors: same source corpus, similar window size (3-word-window), positive PMI, and SVD reduction to 300 dimensions. The only notable differences are a vocabulary cut-off to the top 30K most frequent words in the corpus, and the use of content words only as windows.

4.4.2 Alternative entailment measures

As reviewed in Section 4.2, the literature on entailment with distributional methods has been dominated by the idea of feature inclusion. We thus compare BDSM to a variety of state-of-the art asymmetric similarity measures based on the distributional inclusion hypothesis (the dimensions of hyponym/antecedent vectors are included in those of their hypernyms/consequents). We consider the measures described in Lenci and Benotto (2012) (**clarkeDE**, **weedsPrec**, **cosWeeds**, and **invCL**), as well as **balAPinc**, which was shown to achieve optimal performance by Kotlerman et al. (2010). All these measures provide a score that is higher when a significant part of the candidate antecedent features (=dimensions) are included in those of the consequent. The measures are only meaningful when computed on a non-negative sparse space. Therefore, we evaluate them using the full count space. As an example, weedsPrec is computed as follows:

$$weedsPrec(u, v) = \frac{\sum_i \mathbb{1}[v_i > 0] \cdot u_i}{\sum_i u_i}$$

where u is the distributional vector of the antecedent, v that of the consequent.[5] Please refer to Section 3.3.2 for the definition of the rest of these measures.

[5]BI is equivalent to weedsPrec in Boolean space.

Finally, we implement a full-fledged supervised machine learning approach directly operating on distributional representations. Following the recent literature reviewed in Section 4.2 above, we train a Support Vector Machine (**SVM**) (Cristianini and Shawe-Taylor, 2000) on the concatenated distributional vectors of the training pairs, and judge the presence of entailment for a test pair based on the same concatenated representation (the results of Weeds et al. (2014) and Roller et al. (2014) suggest that concatenation is the most reliable way to construct SVM input representations that take both the antecedent and the consequent into account).

4.4.3 Data sets

Lexical entailment We test the models on benchmarks derived from two existing resources. We used the Lexical Entailment Data Set (LEDS) from Baroni et al. (2012) that contains both entailing (obtained by extracting hyponym-hypernym links from WordNet) and non-entailing pairs of words (constructed by reversing a third of the pairs and randomly shuffling the rest). We edited this resource by removing dubious data from the entailing pairs (e.g., *logo/signal*, *mankind/mammal*, *geek/performer*) and adding more negative cases (non-entailing pairs), obtained by shuffling words in the positive examples. We derived two balanced subsets: a development set (**LEDS-dev**) with 236 pairs in each class and a core set with 911 pairs in each class (**LEDS-core**), such that there is no lexical overlap between the positive classes of each set, and negative class overlap is minimized. Since a fair amount of negative cases were obtained by randomly shuffling words from the positive examples, leading to many unrelated couples, just pair similarity might be a very strong baseline here. We thus explore a more challenging setup, **LEDS-dir**, where we replace the negative examples of LEDS-core by positive pairs in reverse order, thus focusing on entailment *direction*.

We derive two more benchmarks from BLESS (Baroni and Lenci, 2011). BLESS lists pairs of concepts linked by one of 5 possible relations: coordinates, hypernymy, meronymy, attributes and events. We employed this resource to construct **BLESS-coord**, which –unlike LEDS, where entailing pairs have to be distinguished from pairs of words that, mostly, bear no relation– is composed of 1,236 super-subordinate pairs (which we treat as positive examples) to be distinguished from 3,526 coordinate pairs. **BLESS-mero** has the same positive examples, but 2,943 holo-meronyms pairs as negatives. Examples of all lexical benchmarks are given in Table 4.1.

Sentence entailment To evaluate the models on recognizing entailment between sentences, we use a benchmark derived from SICK (Marelli et al., 2014b). The original data set contains pairs of sentences in entailment, contradiction and neutral relations.

	Positive	*Negative*
LEDS	elephant → animal	ape ↛ book
LEDS-dir		animal ↛ elephant
BLESS-coord	elephant → herbivore	elephant ↛ hippo
BLESS-mero		elephant ↛ trunk

Table 4.1: Lexical entailment examples.

Positive	*Negative*
A man is slowly trekking in the woods → The man is hiking in the woods	A group of scouts are camping in the grass ↛ A group of scouts are hiking through the grass

Table 4.2: SICK sentence entailment examples.

We focus on recognizing entailment, treating both contradictory and neutral pairs as negative examples (as in the classic RTE shared tasks up to 2008).[6] Data are divided into a development set (**SICK-dev**) with 500 sentence pairs (144 positive, 356 negative), a training set (**SICK-train**) with 4,500 pairs (1,299 positive, 3,201 negative) and a test set (**SICK-test**) with 4,927 pairs (1,414 positive, 3,513 negative). Examples from SICK are given in Table 4.2.

4.4.4 Training regime

We tune once and for all the hyperparameters of the models by maximizing accuracy on the small LEDS-dev set. For SVM, we tune the kernel type, picking a 2nd degree polynomial kernel for the count and TypeDM spaces, and a linear one for the predict space (alternatives: RBF and 1st, 2nd or 3rd degree polynomials). The choice for the count space is consistent with Turney and Mohammad (2014). For BDSM, we tune H (dimensionality of Boolean vectors), setting it to 100 for count, 1,000 for predict and 500 for TypeDM (alternatives: 10, 100, 500, 1,000 and 1,500) and the sparsity parameter ρ, picking 0.5 for count, 0.75 for predict, and 0.25 for TypeDM (alternatives: 0.01, 0.05, 0.1, 0.25, 0.5, 0.75). For BDSM and the asymmetric similarity measures, we also tune the e threshold above which a pair is treated as entailing for each dataset.

[6]This prevents a direct comparison with the results of the SICK shared task at SemEval (Marelli et al., 2014a). However, all competitive SemEval systems were highly engineered for the task, and made extensive use of a variety of pre-processing tools, features and external resources (cf. Table 8 of Marelli et al. (2014a)), so that a fair comparison with our simpler methods would not be possible in any case.

model	LEDS *core*	LEDS *dir*	BLESS *coord*	BLESS *mero*
count				
clarkeDE	77	63	27	36
weedsPrec	79	75	27	33
cosWeeds	79	63	26	35
invCL	77	63	27	36
balAPinc	79	66	26	36
SVM (count)	**84**	**90**	55	57
BDSM (count)	83	87	53	55
predict				
SVM (predict)	71	85	70	55
BDSM (predict)	80	79	**76**	**68**
TypeDM				
SVM (TypeDM)	78	83	56	60
BDSM (TypeDM)	83	71	31	59

Table 4.3: Percentage accuracy (LEDS) and F1 (BLESS) on the lexical entailment benchmarks.

The γ (RBF kernel radius) and C (margin slackness) parameters of SVM and the λ, β and t parameters of BDSM (see Section 4.3) are set by maximizing accuracy on LEDS-dev for all lexical entailment experiments. For sentence entailment, we tune the same parameters on SICK-dev. In this case, given the imbalance between positive and negative pairs, we maximize weighted accuracy (that is, we count each true negative as $(|pos| + |neg|)/2|neg|$, and each true positive as $(|pos| + |neg|)/2|pos|$, where $|class|$ is the cardinality of the relevant class in the tuning data).

Finally, for lexical entailment, we train the SVM and BDSM weights by maximizing accuracy on LEDS-core. For LEDS-core and LEDS-dir evaluation, we use 10-fold validation. When evaluating on the BLESS benchmarks, we train on full LEDS-core, excluding any pairs also present in BLESS. For sentential entailment, the models are trained by maximizing weighted accuracy on SICK-train.

4.5 Lexical entailment

Table 4.3 reports lexical entailment results (percentage accuracies for the LEDS benchmarks, F1 scores for the unbalanced BLESS sets). We observe, first of all, that SVM and BDSM are clearly outperforming the asymmetric similarity measures in all tasks. In only one case the lowest performance attained by a supervised model drops below the level of the best asymmetric measure performance (BDSM using TypeDM on LEDS-dir).[7] The performance of the unsupervised measures, which rely most directly on the original distributional space, confirms that the latter is more suited to capture similarity than entailment. This is shown by the drop in performance from LEDS-core (where many negative examples are semantically unrelated) to LEDS-dir (where items in positive and negative pairs are equally similar), as well as by the increase from BLESS-coord to BLESS-mero (as coordinate negative examples are more tightly related than holo-meronym pairs).

In the count input space, SVM and BDSM perform similarly across all 4 tasks, with SVM having a small edge. In the next sections, we will thus focus on count vectors, for the fairest comparison between the two models. BDSM reaches the most consistent results with predict vectors, where it performs particularly well on BLESS, and not dramatically worse than with count vectors on LEDS. On the other hand, predict vectors have a negative overall impact on SVM in 3 over 4 tasks. Concerning the interaction of input representations and tasks, we observe that count vectors work best with LEDS, whereas for BLESS predict vectors are the best choice, regardless of the supervised method employed.

Confirming the results of Baroni et al. (2014b), the TypeDM vectors are not a particularly good choice for either model. BDSM is specifically negatively affected by this choice in the LEDS-dir and BLESS-coord tasks. The tight taxonomic information captured by a dependency-based model such as TypeDM might actually be detrimental in tasks that require distinguishing between closely related forms, such as coordinates and hypernyms in BLESS-coord.

In terms of relative performance of the supervised entailment models, if one was to weigh each task equally, the best average performance would be reached by BDSM trained on predict vectors, with an average score of 75.75, followed by SVM on count vectors, with an average score of 71.5. We assess the significance of the difference between supervised models trained on the input vectors that give the best performance for each task by means paired t-tests on LEDS and McNemar tests on BLESS. SVM with count vectors is better than BDSM on LEDS-core (*not significant*) and LEDS-dir ($p<0.05$). On the other hand, BDSM with predict vectors is better than SVM on BLESS-coord ($p<0.001$) and BLESS-mero ($p<0.001$). We conclude that, overall, the two models perform similarly on lexical entailment tasks.

[7] We also inspected ROC curves for BDSM (count) and the asymmetric measures, to check that the better performance of BDSM was not due to a brittle e (entailment threshold). The curves confirmed that, for all tasks, BDSM is clearly dominating all asymmetric measures across the whole e range.

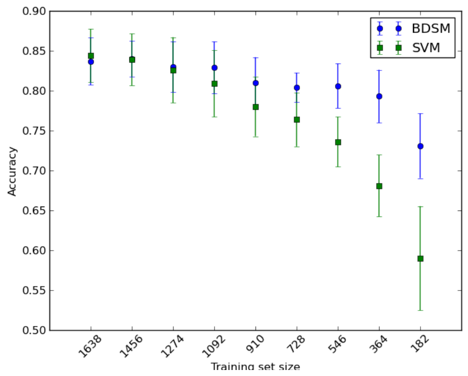


FIGURE 4.3: Average LEDS-core accuracy using count vectors in function of training set size.

4.5.1 Learning efficiency

We just observed that SVM and BDSM have similar lexical entailment performance, especially in count space. However, the two models are radically different in their structure. SVM fits a 2nd order polynomial separating entailing from non-entailing pairs in a space formed by the concatenation of their distributional representations. BDSM, on the other hand, finds a linear transformation into a space where features of the antecedent are included in those of the consequent. We conjecture that the latter has much larger *bias*, imposed by this strict subsective constraint.[8] We expect this bias to help learning, by limiting the search space and allowing the algorithm to harness training data in a more efficient way. Thus, BDSM should be better at learning with less data, where SVM will be prone to overfitting. To test this claim, we measured the cross-validated LEDS-core accuracy obtained from using vectors in count space when reducing the training items in steps of 182 pairs. The results can be seen in Figure 4.3. As expected, BDSM scales down much more gracefully, with accuracy well above 70% with as little as 182 training pairs.

4.6 Sentence entailment

Having shown in the previous experiments that the asymmetric measures are not competitive, we focus here on SVM and BDSM. As mentioned above in Section 4.5, we use count vectors for a fair comparison between the two models, based on their similar performance on the lexical benchmarks.

[8] Mitchell (1980) defines bias as any basis for choosing one generalization over another, other than strict consistency with the observed training instances.

Recall that for sentence entailment we use the same hyperparameters as for the lexical tasks, that the model constants were tuned on SICK-dev, and the model weights on SICK-train (details in Section 4.4.4 above). Sentence representations are derived either with the plf approach, that returns sentence vectors built according to syntactic structure, or the additive (add) method, where constituent word vectors are simply summed to derive a sentence vector (see Section 4.4.1 above).

We compare SVM and BDSM to the **Sycophantic** baseline classifying all pairs as entailing and to a **Majority** baseline classifying everything as non-entailing. The Word Overlap method (**WO**) calculates the number of words in common between two sentences and classifies them as entailing whenever the ratio is above a certain threshold (calibrated on SICK-train).

Results are given in Table 4.4. Because of class unbalance, F1 is more informative than accuracy (the Majority baseline reaches the best accuracy with 0 precision and recall), so we focus on the former for analysis. We observe first that sentence vectors obtained with the additive model are consistently outperforming the more sophisticated plf approach. As we have observed in Section 3.4, we again confirm the results of Blacoe and Lapata (2012) on the effectiveness of simple composition methods. We leave it to further studies to determine to what extent this can be attributed to specific characteristics of SICK that make word order information redundant, and to what extent it indicates that plf is not exploiting syntactic information adequately (note that Paperno et al. (2014) report minimal performance differences between additive and plf for their *msrvid* benchmark, that is the closest to SICK).

Coming now to the crucial comparison of BDSM against SVM (focusing on the results obtained with the additive method), BDSM emerges as the best classifier when evaluated alone, improving over SVM, although the difference is not significant. Since the Word Overlap method is performing quite well (better than SVM) and the surface information used by WO should be complementary to the semantic cues exploited by the vector-based models, we built combined classifiers by training SVMs (on SICK-dev) with linear kernels and WO value plus each method's score (BI for BDSM and distance to the margin for SVM) as features. The combinations improve performance for both models and BDSM+WO attains the best overall F1 score, being statistically superior to both SVM+WO (p<0.001) and WO alone (p<0.001) (statistical significance values obtained through McNemar tests).

We repeated the training data reduction experiment from Section 4.5.1 by measuring cross-validated F1 scores for SICK (with additive composition). We confirmed that BDSM is robust to decreasing the amount of training data, maintaining an F1 score of

model	P	R	F1	A
Sycophantic	29	100	45	29
Majority	0	0	0	**71**
WO	40	86	55	60
SVM (add)	47	54	51	70
BDSM (add)	**48**	74	58	69
SVM (plf)	39	45	42	64
BDSM (plf)	44	71	55	66
SVM(add) + WO	44	**82**	58	65
BDSM(add) + WO	**48**	80	**60**	69
SVM(plf) + WO	42	76	54	63
BDSM(plf) + WO	42	77	54	63

Table 4.4: SICK results (percentages).

56 with only 942 training items, whereas, with the same amount of training data, SVM drops to a F1 of 42.

4.7 Understanding Boolean vectors

BDSM produces representations that are meant to respect inclusion and be interpretable. We turn now to an extended analysis of the learned representations (focusing on those derived from count vectors), showing first how BDSM activation correlates with generality and abstractness, and then how similarity in BDSM space points in the direction of an extensional interpretation of Boolean units.

4.7.1 Boolean dimensions and generality

The BDSM layer is trained to assign more activation to a hypernym than its hyponyms (the hypernym units should include the hyponyms' ones), so the more general (that is, higher on the hypernymy scale) a concept is, the higher the proportion of activated units in its BDSM vector. The words that activate all nodes should be implied by all other terms. Indeed, very general words such as *thing(s)*, *everything*, and *anything* have Boolean vectors with all 1s. But there are also other words (a total of 768) mapping to the top element of the Boolean algebra (a vector of all 1s), including *reduction*, *excluded*, *results*, *benefit*, *global*, *extent*, *achieve*. The collapsing of these latter terms must be due to a combination of two factors: low dimensionality of Boolean space,[9] and the fact that the model was trained on a limited vocabulary, mostly consisting of concrete nouns, so there was simply no training evidence to characterize abstract words such as *benefit* in a more nuanced way.

[9]With count input representations, our tuning favoured relatively dense 100-dimensional vectors (see Section 4.4.4).

Still, we predict that the proportion of Boolean dimensions that a word activates (i.e., dimensions with value 1) should correspond, as a trend, to its degree of semantic generality. More general concepts also tend to be more abstract, so we also expect a correlation between Boolean activation and the word rating on the concrete-abstract scale.[10] To evaluate these claims quantitatively, we rely on WordNet (Fellbaum, 1998), which provides an `is-a` hierarchy of word senses ('synsets') that can be used to measure semantic generality. We compute the average length of a path from the root of the hierarchy to the WordNet synsets of a word (shortest is most general, so that a higher depth score corresponds to a more *specific* concept). We further use the Ghent database (Brysbaert et al., 2013), that contains 40K English words rated on a 1-5 scale from least to most concrete (as expected, depth and concreteness are correlated, $\rho = .54$).

Boolean vector activation significantly correlates with both variables (ρ=-18 with depth, ρ=-30 with concreteness; these and all correlations below significant at $p < 0.005$). Moreover, the BDSM activations are much higher than those achieved by distributional vector L1 norm (which, surprisingly, has positive correlations: ρ=13 with depth, ρ=21 with concreteness) and word frequency (ρ=-2 with depth, ρ=4 with concreteness).

We visualize how Boolean activation correlates with generality in Figure 4.4. We plot the two example words *car* and *newspaper* together with their 30 nearest nominal neighbours in distributional space,[11] sorting them from most to least activated. More general words do indeed cluster towards the top, while more specific words are pushed to the bottom. Interestingly, while *vehicle* and *organization* were present in the training data, that was not the case for *media* or *press*. Moreover, the training data did not contain any specific type of car (like *volvo* or *suv*) or newspaper (*tribune* or *tabloid*).

4.7.2 Similarity in Boolean space

From the model-theoretical point of view, word-to-BDSM mapping provides an *interpretation function* in the logical sense, mapping linguistic expressions to elements of the model domain (Boolean dimensions). If distributional vectors relate to concepts in a (hyper)intensional construal (Erk, 2013), Boolean vectors could encode their (possible) extensions along the lines suggested by Grefenstette (2013), with vector dimensions corresponding to entities in the domain of discourse.[12] Under the extensional interpretation, the Boolean vector of a word encodes the set of objects in the word extension.

[10] Automatically determining the degree of abstractness of concepts is a lively topic of research (Kiela et al., 2014; Turney et al., 2011).

[11] Due to tagging errors, the neighbors also include some verbs like *parked* or adjectives like *weekly*.

[12] In fact everything we say here applies equally well to certain *intensional* interpretations of Boolean vectors. For example, the atoms of the Boolean algebra could correspond not to entities in the actual world but to classes of individuals across possible worlds. Alternatively, one can think of the atoms as "typical cases" rather than actual individuals, or even as typical properties of the relevant individuals.

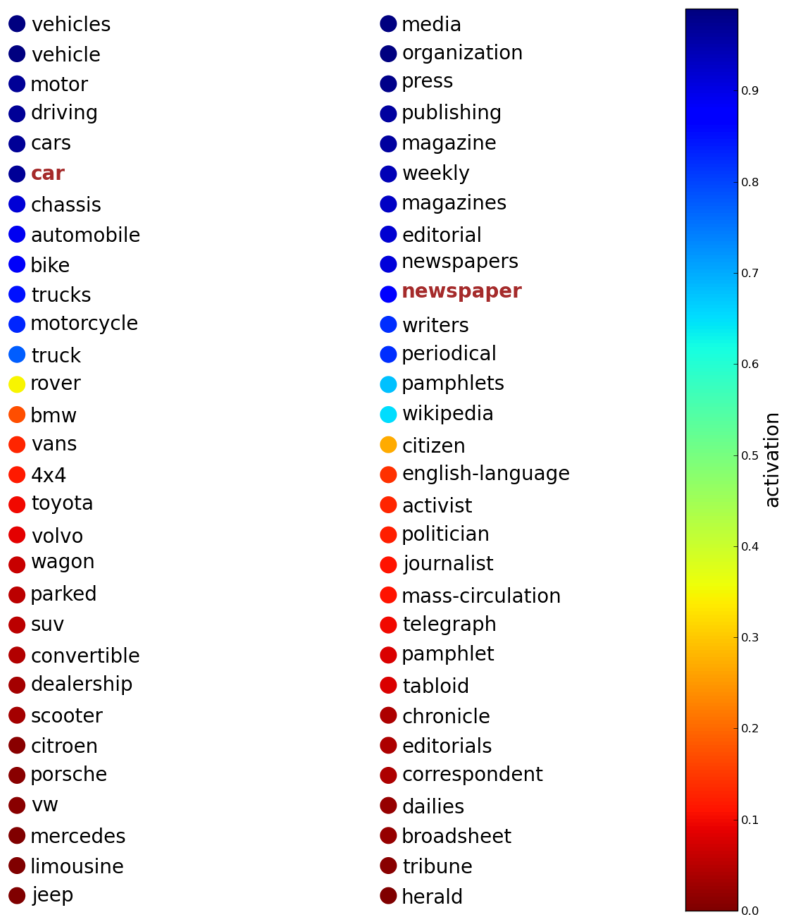


Figure 4.4: Boolean activation (percentage of positive dimensions) of the 30 nearest distributional neighbours of *car* and *newspaper*.

But of course, given that our BDSM implementation operates with only 100 dimensions, one cannot expect such an extensional interpretation of the model to be realistic. Still, the extensional interpretation of the Boolean model, while being highly idealized, makes some testable predictions. Under this view, synonyms should have identical Boolean vectors, antonyms should have disjoint vectors. Compatible terms (including hyponym-hypernym pairs) should overlap in their 1s. Cohyponyms, while high on the relatedness scale, should have low "extensional similarity"; *singer* and *drummer* are very related notions but the intersection of their extensions is small, and that between *alligator* and *crocodile* is empty (in real life, no entity is simultaneously a crocodile and an alligator).

As expected, the straightforward interpretation of dimensions as individuals in a possible world close to ours is contradicted by many counterexamples in the present BDSM implementation. For example, the nouns *man* and *woman* have a considerable overlap in activated Boolean dimensions, while in any plausible world hermaphrodite humans are rare. Still, compared to distributional space, BDSM goes in the direction of an extensional model as discussed above. To quantify this difference, we compared the similarity scores (cosines) produced by the two models. Specifically, we first created a list of pairs of semantically related words using the following procedure. We took the 10K most frequent words paired with their 10 closest neighbors in the count distributional space. We then filtered them to be of "medium frequency" (both words must lie within the 60K-90K frequency range in our 2.8B token corpus). One of the authors annotated the resulting 624 pairs as belonging to one of the following types: cohyponyms (137, e.g., *AIDS* vs. *diabetes*); derivationally related words (10, e.g., *depend* vs. *dependent*); hypernym-hyponym pairs (37, e.g., *arena* vs. *theater*); personal names (97, e.g., *Adams* vs. *Harris*); synonyms (including contextual ones; 49, e.g., *abilities* vs. *skill*); or "other" (294, e.g., *actress* vs. *starring*), if the pair does not fit any of the above types (some relations of interest, such as antonymy, were excluded from further analysis as they were instantiated by very few pairs). Since cosines have different distributions in distributional (**DS**) and Boolean space (**BS**), we z-normalized them before comparing those of pairs of the same type across the two spaces.

Under the extensional interpretation, we expect co-hyponyms to go apart after Boolean mapping, as they should in general have little extensional overlap. Indeed they have significantly lower cosines in BS than DS ($p < 0.001$; paired t-test). As expected under the extensional interpretation, personal names are very significantly less similar in BS than DS ($p<0.001$). Synonyms and hypo/hypernyms have significant denotational overlap, and they move closer to each other after mapping. Specifically, synonyms significantly gain in similarity between BS and DS ($p<0.01$), whereas hyponym-hypernym pairs, while not differing significantly in average similarity across the spaces, change from being weakly significantly lower in cosine than all other pairs in DS ($p<0.05$) to being indistinguishable from the other pairs in BS. Derivationally related words gain in similarity ($p<0.01$) collapsing to almost identical vectors after Boolean mapping. This deserves a special comment. Although words in these pairs typically belong to different parts of speech and are not synonyms in the usual sense, one could interpret them as denotational synonyms in the sense that they get reference in the same situations. Taking two word pairs from our data as examples, the existence of *experiments* entails the presence of something *experimental*, anything *Islamic* entails the presence of *Islam* in the situation, etc. If so, the fact that derivationally related words collapse under Boolean mapping makes perfect sense from the viewpoint of denotational overlap.

4.8 Conclusion

We introduced BDSM, a method that extracts representations encoding semantic properties relevant for making inferences from distributional semantic vectors. When applied to the task of detecting entailment between words or sentences, BDSM dramatically improves of asymmetric unsupervised measures and is competitive against a state-of-the-art SVM classifier, and needs less learning data to generalize. In contrast to SVM, BDSM is transparent: we are able not only to classify a pair of words (or sentences) with respect to entailment, but we also produce a compact Boolean vector for each word, that can be used alone for recognizing its entailment relations. Besides the analogy with the structures postulated in formal semantics, this can be important for practical applications that involve entailment recognition, where Boolean vectors can reduce memory and computing power requirements.

The Boolean vectors also allow for a certain degree of interpretability, with the number of active dimensions correlating with semantic generality and abstractness. Qualitative analysis suggests that Boolean mapping moves the semantic space from one organized around word relatedness towards a different criterion, where vectors of two words are closer to each other whenever their denotations have greater overlap. This is, however, just a tendency. Ideally, the overlap between dimensions of two vectors should be a measure of compatibility of concepts. In future research, we would like to explore to what extent one can reach this ideal, explicitly teaching the network to also capture other types of relations (e.g., no overlap between cohyponym representations), and using alternative learning methods.

Chapter 5

Compatibility

"When the going gets tough, you don't want a criminal lawyer: You want a ***criminal*** *lawyer."*

— Jesse Pinkman, Breaking Bad

5.1 Introduction

In the previous two chapters I have discussed how can we compute entailment relations between words and phrases starting from their distributional representations. Yet, despite that recognizing entailment is one fundamental part of conceptual reasoning, there are other relations that are also important to capture. For example, even though *iguana* does not entail *pet*, the relation between these two concepts is very different to that between *iguana* and *turtle*: Whereas Rudolph the iguana may or may not be a pet, he is definitely not a turtle.

The previous examples hint at a fundamental semantic property, namely *compatibility*, that we define, for our current purposes, as follows: *Linguistic expressions* w_1 *and* w_2 *are compatible iff, in a reasonably normal state of affairs, they can both truthfully refer to the same thing. If they cannot, then they are incompatible.* We realize that the notion of a "reasonably normal sate of affairs" is dangerously vague, but we want to exclude science-fiction scenarios in which dogs mutate into cats. And we use *thing* as a catch-all term for anything words (or other linguistic expressions) can refer to (entities, events, collections, etc.).

The notions of compatibility and incompatibility have been linguistics and cognitive science before (Cruse, 1986; Murphy, 2010). The definition that we give here for compatibility is related, but different from the one by Cruse. For example, subsuming pairs are out of the scope of compatibility under his definition, whereas we include them. Murphy defines incompatibility similarly to us, but she does not define compatibility. We are not aware, on the other hand, of any earlier systematic attempt to study the phenomenon empirically, nor to model it computationally.

In general, compatible terms will be semantically related (*dog* and *animal*). However, relatedness does not suffice: many semantically related, even very similar terms are not compatible (*dog* and *cat*). Relatedness is not even a necessary condition: A *husband* can be a *hindrance* in an all-too-normal state of affairs, but the concepts of husband and hindrance are not semantically close. Moreover, compatibility does not reduce to (a set of) more commonly studied semantic relations. While it relates to hypernymy, synonymy and co-hyponymy, there are cases, such as *husband/hindrance*, that do not naturally map to any of these relations. Also, although many incompatibles among closely related pairs are co-hyponyms, this is not necessarily the case: You cannot be both a *dog* and a *cat*, but you can be a *violinist* and a *drummer*.

We argue that, since knowing what's compatible plays a central role in human semantic reasoning, algorithms that determine compatibility automatically will help in many domains that require human-like semantic knowledge. Most obviously, compatibility is a necessary (although not sufficient) prerequisite for coreference. *Dog* and *puppy* could belong to the same coreference chain, whereas *dog* and *cat* do not. We conjecture that the relatively disappointing performance of DSMs in support of coreference resolution (Poesio et al., 2010) is at least partially due to the inability of standard DSMs to distinguish compatible and incompatible terms. Compatibility is also central to recognizing entailment (and contradiction): Standard DSMs are of relatively little use in recognizing entailment as they treat antonymous, contradictory words such as *dead* and *alive* as highly related (Adel and Schütze, 2014; Mohammad et al., 2013), with catastrophic results for the inferences that can be drawn (antonyms are just the tip of the incompatibility iceberg: *dog* and *cat* are not antonyms, but one still contradicts the other). Knowing what's compatible might also help in tasks that require recognizing (distant) paraphrases, such as question answering, document summarization or even machine translation (*the violinist also played the drum* might corefer with *the drummer also played the violin*, whereas *the dog was killed* and *the cat was killed* must refer to different events). Other applications could include modeling semantic plausibility of a nominal phrase (Lynott and Connell, 2009; Vecchi et al., 2011), where the goal is to accept expressions like *coastal mosquito*, but reject *parlamentary tomato*. Finally, the notion of incompatibility relates to (certain kinds of) negation. Negation is notoriously difficult to model with DSMs (Hermann et al., 2013), and compatibility might offer a new angle into it.

In this chapter, we introduce a new, large benchmark to evaluate computational models on compatibility detection. We then present a supervised neural-network based model that takes distributional semantic vectors as input and embeds them into a space that is optimized for compatibility detection. The model performs significantly better than direct DSM relatedness, and achieves high scores in absolute terms.

5.2 The compatibility benchmark

We started the benchmark construction by manually assembling a list of 299 words including mostly concrete, basic-level concepts picked from categories where taxonomically close terms tend to be incompatible (e.g., biological classes such as animals and vegetables), as well as from categories that are more compatibility-prone (kinship terms, professions), or somewhere in the middle (tools, places). The list also included category names at different levels of abstraction (*creature*, *animal*, *carnivore*. . .), as well as some terms that were expected to be of high general compatibility (*hindrance*, *expert*, *companion*. . .). By randomly coupling words from this list, we generated pairs that should reflect a wide range of compatibility patterns (compatible and incompatible coordinate terms, words in an entailment relation, dissimilar but compatible, dissimilar and incompatible, etc.).[1] We generated about 18K such random pairs.

We used a subset of about 3K pairs in a pilot study on the CrowdFlower[2] crowd-sourcing platforms, in which we asked participants to annotate them for compatibility either as a yes/no judgment accompanied by a confidence rating, or on a 7-point scale. Correlation between mean binary and ordinal ratings was extremely high (>0.95), so we decided to adopt the potentially more precise, albeit more noisy, 7-point scale. Confidence judgments (median: 6.6/7), participant agreement and sanity checks on obvious cases confirmed that the raters understood the task well and produced the expected judgments consistently.

We thus launched a larger CrowdFlower survey, asking participants to rate pairs on a 7-point scale by answering the following question: "How much do you agree with the statement that <*word1*> and <*word2*> can refer to the same thing, animal or person?" We asked the judges to consider real-life scenarios and fairly ordinary circumstances; in case of ambiguity, they were asked to choose the sense that would make the pair compatible, as long as it was sufficiently common. 20 control items with obvious choices

[1]We realize that the resulting pairs might not resemble the natural distribution of compatibility decisions that an average person might encounter in daily life. However, the fact that (as we show below) subjects were highly consistent in judging the items proves that the data reflect genuine shared semantic knowledge a computational model should be able to capture.

[2]http://www.crowdflower.com

(e.g. *drummer/ant* - *writer/father*) were inserted to exclude raters that did not perform the task seriously. We paid close attention to contributors' feedback, correcting dubious controls. For example, we removed *bucket/chair*, since one contributor pointed out that you could turn a bucket upside down and use it as a chair.[3] In this way, we obtained usable annotation for 17973 pairs, each rated by 10 participants[4]. The average standard deviation was as as low as 0.70, compared to the standard deviation of a uniformly distributed multinomial distribution, which amounts to 1.8. As expected, ratings were highly skewed as most random pairs are incompatible: the median is 1.10 (with a standard deviation of 1.81). Yet, the overall distribution is bimodal, peaking at the two ends of the scale.

In order to be able to phrase (in)compatibility detection not only in continuous terms, but also as dichotomous tasks, we further produced a list of unambiguously (in)compatible pairs from the ends of the rating scale. Specifically, we manually inspected a subset of the list (before any computational simulation was run), and picked a mean 3.7 rating (exclusive) as minimum value for compatible pairs, and 1.6 (inclusive) as maximum score for incompatible ones. The number of problematic cases above/below these thresholds was absolutely negligible. We thus coded the data set by classifying the 2,933 pairs above the first threshold as compatible (e.g., *expert/criminal*, *hill/obstacle*, *snake/vermin*), the 12,669 pairs below the second as incompatible (e.g., *bottle/plate*, *cheetah/queen*), and the remainder as neither.

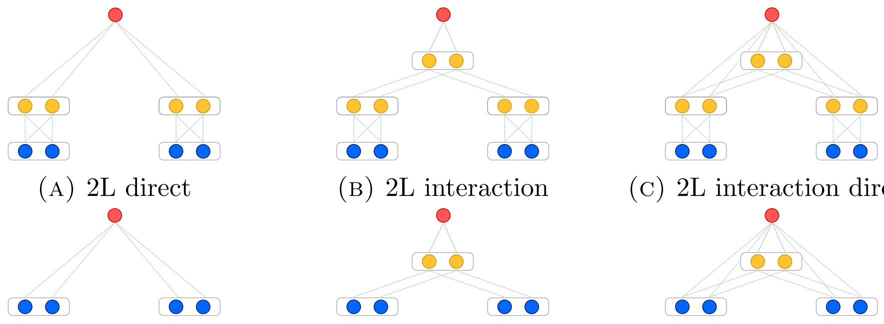
5.3 Models

As we have done before with entailment (Chapter 4), we start from distributional representations of concepts, and seek to induce a compatibility measure by learning the parameters of a model in a supervised manner. In particular, we used the word vectors publicly available at `http://clic.cimec.unitn.it/composes/semantic-vectors.html`. These vectors, extracted with the word2vec toolkit (Mikolov et al., 2013) from a 3B token corpus, were shown by Baroni et al. (2014b) to produce near-state-of-the-art performance on a variety of semantic tasks.

We hypothesized that the interaction between a simple set of features (induced from the distributional ones) should account for a large portion of compatibility patterns. For example, human roles would typically be compatible (*classmate/friend*), whereas two

[3]We also were surprised to learn that drummer ants actually exist. Yet, in that case we decided to keep the control item since, under the most common sense of *drummer*, and in ordinary circumstances, ants cannot be drummers.

[4]The guidlienes provided to the participants and the collected data set are available at: `http://clic.cimec.unitn.it/composes/`



(A) 2L direct (B) 2L interaction (C) 2L interaction direct

(D) 1L direct (E) 1L interaction (F) 1L interaction direct

FIGURE 5.1: Schematic representation of the models

animals would probably be incompatible (*iguana/zebra*). The model should thus be able to learn features associated to such classes, and compatibility rules associated to their interaction (e.g., if both w_1 and w_2 have large values for a *human* feature, compatibility is more likely). We incorporated this insight into the **2L interaction** neural network illustrated in Figure 5.1b. This network takes the distributional representations of the words in a pair, transforms them into new feature vectors by means of a mapping that is shared by both inputs, constructs the vector of pairwise interactions between the induced features, and finally uses the weighted combination of the latter to produce a real-number score.

We considered then some variations of the 2L interaction model, to investigate the importance of each of its components. In **2L direct** (Figure 5.1a), we removed the interaction layer, making the model score a weighted combination of the mapped vectors. The **2L interaction direct** model (Figure 5.1c) computes the final score through a weighted combination of both the mapped representations and their interaction vector. The **1L** models (Figures 5.1d, 5.1e and 5.1f) are analogous to the corresponding 2L models, but removing the feature mapping layer, thus operating directly on the distributional vectors.

5.4 Experiments

Since compatibility is a symmetric relation, we first duplicated each pair in the benchmark by swapping the two words. We then split it into training, testing and development sections. To make the task more challenging, we enforced disjoint vocabularies in each of them. For example, *drummer* only occurs in the training set, while *ant*, only in the test set. We use about 1/10th of the vocabulary (29 words) on the development set and the rest was split equally between train and test (135 words each). The resulting partitions contain 7,228 (train), 7,336 (test) and 312 (development) pairs, respectively.

Model	corr.	comp.			incomp.		
	r	P	R	F1	P	R	F1
1L direct	50	59	55	57	80	83	72
1L interaction	51	50	61	55	80	77	79
1L int. direct	49	52	57	54	80	79	80
2L direct	49	51	58	54	81	79	80
2L interaction	**72**	76	58	**66**	84	90	**87**
2L int. direct	67	71	58	64	82	85	84
1L mono	35	31	57	41	79	77	78
2L mono	35	32	64	43	80	72	76
Cosine	36	29	58	38	78	71	74

TABLE 5.1: Experimental results. Correlation with human ratings measured by Pearson r. (In)compatibility detection scored by the F1 measure.

To train the models, we used the scores they generate in three sub-tasks: approximation of average ratings, classification of compatibles and classification of incompatibles. We used mean square error as cost function for the first sub-task, cross-entropy for the latter two.

We implemented the models in Torch7 (Collobert et al., 2011).[5] We trained them for 120 epochs with adagrad, with a batch size of 150 items and adopting an emphasizing scheme (LeCun et al., 2012), where compatibles, incompatibles and middle-ground items appear in equal proportions. We fixed hidden-layer size to 100 dimensions, while we tuned a coefficient for a L2-norm regularization term on the development data.

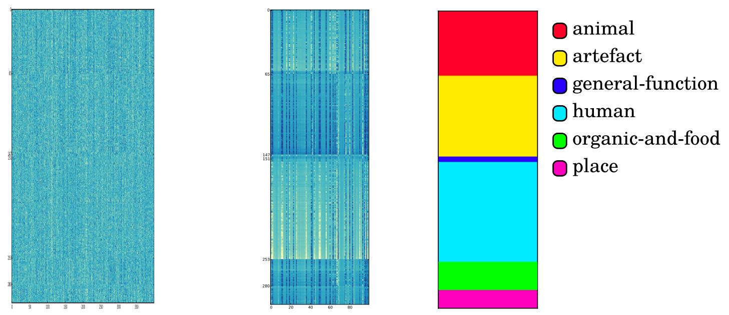
We evaluated the models ability to predict human compatibility ratings as well as to detect compatible and incompatible items.

We compared the supervised measures to the cosine of pairs directly represented by their DSM vectors (with thresholds tuned on the training set). We expected this baseline to fare relatively well on incompatibility detection, since many of our randomly generated pairs were both incompatible and dissimilar (e.g., *bag/bus*).

Also, we controlled for the portion of the data that can be accounted just by looking at one of the words of the relation (for example, the presence of a word might indicate that the relation is incompatible). To this end, we included two models that look at only one of the words in the pair. **1L mono** is a logistic regression model that only looks at the first word of the pair while **2L mono** is an analogous neural network with one hidden layer.

Results are reported in Table 5.1. As it can be seen, all the supervised models from Figure 5.1 strongly outperform the cosine (that, as expected, is nevertheless quite good at detecting incompatibles). Also, they outperform the mono models (with the only

[5]We make the code available at `https://github.com/germank/compatibility-naacl2015`



(A) Input vectors (B) Mapped vectors (C) Categories

FIGURE 5.2: Heatmap visualization of original DSM features and features learned by the mapping function of the 2L interaction model.

exception of 1L direct on incompatibility), showing that the data they account for cannot be reduced to properties of individual lexical items. Importantly, the 2L interaction model is way ahead of all other models, confirming our expectations.

To gain some insight into the features learned by the best model, we labeled the words of our input vocabulary with one of the following general category tags: *animal*, *artefact*, *general-function*, *human*, *organic-and-food* and *place*. The distribution of the vocabulary across the labels is shown in Figure 5.2c. If we plot the input distributional vectors so that words tagged with the same category are adjacent to each other, and categories arranged as in Figure 5.2c, we obtain the heatmap in Figure 5.2a, where no obvious pattern emerges. If instead we plot the output vectors of 2L interaction mapping in the same way, we obtain the heatmap in Figure 5.2b. It is evident that the mapping produces vectors that are similar within most categories, and very different across them. Thus, the 2L interaction model clearly learned the relevance of general categories in capturing compatibility judgments. The fact that this model produced the best results hints at the importance of exploiting this source of information, confirming the intuition we used in designing it, that compatibility can be characterized by a combination of general relatedness and category-specific cues.

Finally, we explored to what extent the data can be accounted by co-hyponymy, an idea briefly introduced in the introductory discussion of Section 5.1. For simplicity purposes, we take the same category tags we just introduced as a word's hypernym. Classifying co-hyponyms as incompatibles and non-cohyponyms as compatibles performs very poorly (7 and 18 F1-scores for compatibility and incompatibility, respectively). On the other hand, the opposite strategy – co-hyponyms as compatibles and non-cohyponyms as incompatibles – works much better (62 and 84 F1), even outperforming many supervised models. Yet, this strategy does not suffice. For example, all animal pairs would be treated as compatibles, whereas 54% of them are actually incompatible. By contrast the L2 interaction model gets 78% of these incompatible pairs right.

5.5 Conclusion

We have introduced the challenge of modeling semantic compatibility. To this end, we collected a data set, and produced a model that satisfactorily captures a large portion of the data, that cannot be accounted for by simple semantic relatedness. Finally, we have explored the features learned by the model, confirming that high-order category information extracted from the distributional representations is relevant for producing compatibility judgements.

Computational models of compatibility could help in many semantic tasks, such as coreference resolution, question answering, modeling plausibility and negation. Future lines of research will explore the contributions that accounting for compatibility can make to these tasks.

Chapter 6

Conclusions

Human conceptual knowledge is vast. In the course of this thesis I have described how we can model some of the inferences that it can afford us, including the detection of entailment relations (Chapter 4), how modification effects can alter these entailment relations (Chapter 3) and accounting for the fact that some concepts can be semantically compatible, whereas other are not (Chapter 5). Crucially, the models tackling these phenomena make use of the information contained in distributional representations, confirming that we can extract from them the semantic features required to draw the necessary inferences.

The results that I have presented are thus encouraging, even though performances are still not sufficiently high to consider any of the previous tasks to be solved. In particular, the Norwegian Blue Parrot (NBP) dataset has proved to be quite challenging to model (see Chapter 3). There, we wanted to capture the fact that, for example, people judge a *cannon* a highly typical *weapon*, in contrast to, for example, a *confetti cannon*, which is judged to be rather atypical. These types of inferences form part of every day language understanding. For example, in the phrase *"the table is served"* we are not referring to a piece of furniture but to the stuff sitting on it (Murphy, 2002). However, it is difficult to draw a line between these particular reasoning instances and many others.

Understanding natural language involves solving a myriad of different inferential problems, where linguistic and conceptual knowledge interact. One such example is given by the Winograd Schema Challenge (Levesque et al., 2011). In this competition, the model is prompted with a phrase such as *"The trophy would not fit in the brown suitcase because it was too big (small). What was too big (small)?"* and it has to decide between the trophy and the suitcase depending on which of the two variants was given. Trophies or suitcases can be either big or small in general, but it is the particular context indicating that one must fit in the other that gives away the answer. However, these phrases are also hard to construct and involve a lot of expert work.

Yet another perspective is that brought in by Paperno et al. (2016) with the LAMBADA dataset, where the goal of the proposed task is to guess the missing word in a passage extracted from fiction novels. Interestingly, the chosen passages have proved to be easily guessable by human annotators, while challenging for computational models. Here, not only subtle conceptual knowledge gets into play but also sophisticated discourse understanding is needed in order to process the contextual cues that hint at the missing word.

Finally, Mikolov et al. (2015) propose a dataset where the goal is no longer to solve any natural language task in particular, but rather focusing on fostering the learning skills that may be needed in order to efficiently assign meaning to linguistic content. In the proposed framework, the learning algorithm is exposed to natural language descriptions of goals that it needs to accomplish also by issuing commands in natural language. Given that the learning agent will only be sparsely rewarded for its accomplishments it will have to be very efficient at capturing the patterns that trigger positive rewards.

Distributional Semantic Models of meaning have shown to be extremely powerful at capturing human knowledge. In this thesis I have again presented evidence confirming this view, but when it comes to doing inference, it seems that we are still far from matching the power of the human mind. In order to close this gap, we may need to focus on new learning strategies. To this end, finding plausible interpretations for sentences, as in the Winograd Schema Challenge; completing passages with exactly the missing word, as in the LAMBADA dataset; and solving linguistic puzzles may all be possible benchmarks to measure our progress in developing algorithms that can be more data-efficient, fast and accurate when making inferences.

Bibliography

H. Adel and H. Schütze. Using mined coreference chains as a resource for a semantic task. In *Proceedings of EMNLP*, pages 1447–1452, Doha, Qatar, 2014.

I. Androutsopoulos and P. Malakasiotis. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38:135–187, 2010.

M. Baroni and A. Lenci. Distributional Memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721, 2010.

M. Baroni and A. Lenci. How we BLESSed distributional semantic evaluation. In *Proceedings of the EMNLP GEMS Workshop*, pages 1–10, Edinburgh, UK, 2011.

M. Baroni and R. Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of EMNLP*, pages 1183–1193, Boston, MA, 2010.

M. Baroni, R. Bernardi, N.-Q. Do, and C.-c. Shan. Entailment above the word level in distributional semantics. In *Proceedings of the 13th conference of the European chapter of the association for computational linguistics*, pages 23–32. Association for Computational Linguistics, 2012.

M. Baroni, R. Bernardi, and R. Zamparelli. Frege in space: A program of compositional distributional semantics. *LiLT (Linguistic Issues in Language Technology)*, 9, 2014a.

M. Baroni, G. Dinu, and G. Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of ACL*, pages 238–247, Baltimore, MD, 2014b.

I. Beltagy, C. Chau, G. Boleda, D. Garrette, K. Erk, and R. Mooney. Montague meets Markov: Deep semantics with probabilistic logical form. In *Proceedings of *SEM*, pages 11–21, Atlanta, GA, 2013.

Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

W. Blacoe and M. Lapata. A comparison of vector-based representations for semantic composition. In *Proceedings of EMNLP*, pages 546–556, Jeju Island, Korea, 2012.

G. Boleda, E. M. Vecchi, M. Cornudella, and L. McNally. First order vs. higher order modification in distributional semantics. In *Proceedings of EMNLP*, pages 1223–1233, Jeju Island, Korea, 2012.

G. Boleda, M. Baroni, L. McNally, and N. Pham. Intensionality was only alleged: On adjective-noun composition in distributional semantics. In *Proceedings of IWCS*, pages 35–46, Potsdam, Germany, 2013.

A. Bordes, X. Glorot, J. Weston, and Y. Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *Proceedings of AISTATS*, pages 127–135, La Palma, Canary Islands, 2012.

S. R. Bowman. Can recursive neural tensor networks learn logical reasoning? *CoRR*, abs/1312.6192, 2013.

M. Brysbaert, A. B. Warriner, and V. Kuperman. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, pages 1–8, 2013.

G. Chierchia and S. McConnell-Ginet. *Meaning and Grammar: An Introduction to Semantics*. MIT Press, Cambridge, MA, 2000.

K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29, 1990.

S. Clark. Vector space models of lexical meaning. In S. Lappin and C. Fox, editors, *Handbook of Contemporary Semantics, 2nd ed.*, pages 493–522. Blackwell, Malden, MA, 2015.

S. Clark and S. Pulman. Combining symbolic and distributional models of meaning. In *Proceedings of the First Symposium on Quantum Interaction*, pages 52–55, Stanford, CA, 2007.

D. Clarke. A context-theoretic framework for compositionality in distributional semantics. *Computational Linguistics*, 38(1):41–71, 2012.

B. Coecke, M. Sadrzadeh, and S. Clark. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36:345–384, 2010.

B. Cohen and G. L. Murphy. Models of concepts. *Cognitive Science*, 8(1):27–58, 1984.

R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.

A. Connolly, J. Fodor, L. Gleitman, and H. Gleitman. Why stereotypes don't even make good defaults. *Cognition*, 103(1):1–22, 2007.

A. Copestake and T. Briscoe. Semi-productive polysemy and sense extension. *Journal of Semantics*, 12:15–67, 1995.

N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, Cambridge, UK, 2000.

D. A. Cruse. *Lexical semantics.* Cambridge University Press, 1986.

I. Dagan, B. Dolan, B. Magnini, and D. Roth. Recognizing textual entailment: rationale, evaluation and approaches. *Natural Language Engineering*, 15:459–476, 2009.

G. Dinu, N. Pham, and M. Baroni. General estimation and evaluation of compositional distributional semantic models. In *Proceedings of ACL Workshop on Continuous Vector Space Models and their Compositionality*, pages 50–58, Sofia, Bulgaria, 2013.

K. Erk. Supporting inferences in semantic space: Representing words as regions. In *Proceedings of IWCS*, pages 104–115, Tilburg, Netherlands, 2009.

K. Erk. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653, 2012.

K. Erk. Towards a semantics for distributional representations. In *Proceedings of IWCS*, pages 95–106, Potsdam, Germany, 2013. Association for Computational Linguistics.

S. Evert. *The Statistics of Word Cooccurrences.* Ph.D dissertation, Stuttgart University, 2005.

C. Fellbaum. *WordNet.* Wiley Online Library, 1998.

J. R. Firth. A synopsis of linguistic theory, 1930-1955. 1957.

D. Garrette, K. Erk, and R. Mooney. A formal approach to linking logical form and vector-space lexical semantics. In H. Bunt, J. Bos, and S. Pulman, editors, *Computing Meaning, Vol. 4*, pages 27–48. Springer, Berlin, 2013.

D. Geeraerts. *Theories of lexical semantics.* Oxford University Press, 2010.

M. Geffet and I. Dagan. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of ACL*, pages 107–114, Ann Arbor, MI, 2005.

G. Golub and C. Van Loan. *Matrix Computations (3rd ed.).* JHU Press, Baltimore, MD, 1996.

E. Grefenstette. Towards a formal distributional semantics: Simulating logical calculi with tensors. *Proceedings of *SEM*, pages 1–10, 2013.

E. Guevara. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the EMNLP GEMS Workshop*, pages 33–37, Uppsala, Sweden, 2010.

J. Hampton. The combination of prototype concepts. In P. Schwanenflugel, editor, *The psychology of word meanings*, pages 91–116. Erlbaum, Hillsdale, NJ, 1991.

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

K. M. Hermann, E. Grefenstette, and P. Blunsom. "Not not bad" is not "bad": A distributional account of negation. In *Proceedings of ACL Workshop on Continuous Vector Space Models and their Compositionality*, pages 74–82, Sofia, Bulgaria, 2013.

F. Hill, R. Reichart, and A. Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 2016.

R. Jenatton, N. L. Roux, A. Bordes, and G. Obozinski. A latent factor model for highly multi-relational data. In *Proceedings of NIPS*, pages 3176–3184, La Palma, Canary Islands, 2012.

C. Kalish. Essentialism and graded membership in animal and artifact categories. *Memory and Cognition*, 23(3):335–353, 1995.

D. Kiela, F. Hill, A. Korhonen, and S. Clark. Improving multi-modal representations using image dispersion: Why less is sometimes more. In *Proceedings of ACL*, pages 835–841, Baltimore, MD, 2014.

A. Kilgarriff. I don't believe in word senses. *Computers and the Humanities*, 31:91–113, 1997.

L. Kotlerman, I. Dagan, I. Szpektor, and M. Zhitomirsky-Geffet. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(04):359–389, 2010.

G. Kruszewski and M. Baroni. Dead parrots make bad pets: Exploring modifier effects in noun phrases. *Lexical and Computational Semantics (* SEM 2014)*, page 171, 2014.

G. Kruszewski and M. Baroni. So similar and yet incompatible: Toward automated identification of semantically compatible words. In *Proceedings of NAACL*, pages 64–969, 2015.

G. Kruszewski, D. Paperno, and M. Baroni. Deriving boolean structures from distributional vectors. *Transactions of the Association for Computational Linguistics*, 3:375–388, 2015.

T. Landauer and S. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.

Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, Berlin, 2012.

D. Lee and S. Seung. Algorithms for Non-negative Matrix Factorization. In *Proceedings of NIPS*, pages 556–562, 2000.

A. Lenci and G. Benotto. Identifying hypernyms in distributional semantic spaces. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 75–79. Association for Computational Linguistics, 2012.

H. J. Levesque, E. Davis, and L. Morgenstern. The winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47, 2011.

O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

M. Lewis and M. Steedman. Combined distributional and logical semantics. *Transactions of the Association for Computational Linguistics*, 1:179–192, 2013.

C. B. Lund and Kevin. Modelling parsing constraints with high-dimensional context space. *Language and cognitive processes*, 12(2-3):177–210, 1997.

D. Lynott and L. Connell. Embodied conceptual combination. *Frontiers in Psychology*, 1:212, 2009.

M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli. SemEval-2014 Task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of SemEval*, pages 1–8, Dublin, Ireland, 2014a.

M. Marelli, S. Menini, M. Baroni, L. Bentivogli, R. bernardi, and R. Zamparelli. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of LREC*, pages 216–223, Rekjavik, Iceland, 2014b.

D. McCarthy and R. Navigli. The English lexical substitution task. *Language Resources and Evaluation*, 43(2):139–159, 2009.

L. McNally. Modification. In M. Aloni and P. Dekker, editors, *Cambridge Handbook of Semantics*. Cambridge University Press, Cambridge, UK, 2013. In press.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. http://arxiv.org/abs/1301.3781/, 2013.

T. Mikolov, A. Joulin, and M. Baroni. A roadmap towards machine intelligence. *arXiv preprint arXiv:1511.08130*, 2015.

J. Mitchell and M. Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010.

T. M. Mitchell. The need for biases in learning generalizations. Technical report, New Brunswick, NJ, 1980.

S. Mohammad, B. Dorr, G. Hirst, and P. Turney. Computing lexical contrast. *Computational Linguistics*, 39(3):555–590, 2013.

G. L. Murphy. *The big book of concepts*. MIT press, 2002.

M. L. Murphy. Antonymy and incompatibility. In K. Allan, editor, *Concise Encyclopedia of Semantics*. Elsevier, Amsterdam, 2010.

S. Padó and M. Lapata. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199, 2007.

D. Paperno, N. T. Pham, and M. Baroni. A practical and linguistically-motivated approach to compositional distributional semantics. In *Proceedings of ACL*, pages 90–99, Baltimore, MD, 2014.

D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of ACL*, pages 1525–1534, Berlin, Germany, 2016.

M. Poesio, S. Ponzetto, and Y. Versley. Computational models of anaphora resolution: A survey. http://clic.cimec.unitn.it/massimo/Publications/lilt.pdf, 2010.

T. Polajnar, L. Fagarasan, and S. Clark. Reducing dimensions of tensors in type-driven distributional semantics. In *Proceedings of EMNLP*, pages 1036–1046, Doha, Qatar, 2014.

L. Rimell. Distributional lexical entailment by topic coherence. *EACL 2014*, 2014.

T. Rocktäschel, S. Singh, and S. Riedel. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of HLT-NAACL*, pages 1119–1129, Denver, CO, 2015.

S. Roller, K. Erk, and G. Boleda. Inclusive yet selective: Supervised distributional hypernymy detection. In *Proceedings of COLING*, pages 1025–1036, Dublin, Ireland, 2014.

E. Santus, A. Lenci, Q. Lu, and S. S. im Walde. Chasing hypernyms in vector spaces with entropy. *EACL 2014*, page 38, 2014.

T. Sasao. *Switching Theory for Logic Synthesis.* Springer, London, 1999. ISBN 9780792384564.

H. Schütze. *Ambiguity Resolution in Natural Language Learning.* CSLI, Stanford, CA, 1997.

S. A. Sloman. Feature-based induction. *Cognitive psychology*, 25(2):231–280, 1993.

E. Smith and D. Osherson. Conceptual combination with prototype concepts. *Cognitive Science*, 8(4):337–361, 1984.

E. E. Smith, D. N. Osherson, L. J. Rips, and M. Keane. Combining prototypes: A selective modification model. *Cognitive Science*, 12(4):485–527, 1988.

R. Socher, D. Chen, C. D. Manning, and A. Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS*, pages 926–934, Lake Tahoe, NV, 2013.

R. Speer and C. Havasi. ConceptNet 5: A large semantic network for relational knowledge. In I. Gurevych and J. Kim, editors, *The People's Web Meets NLP*, pages 161–176. Springer, Berlin, 2013.

G. Strang. *Introduction to linear algebra, 3d edition.* Wellesley-Cambridge Press, Wellesley, MA, 2003.

P. Turney. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585, 2012.

P. Turney and S. Mohammad. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, 2014. In press.

P. Turney and P. Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188, 2010.

P. Turney, Y. Neuman, D. Assaf, and Y. Cohen. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of EMNLP*, pages 680–690, Edinburgh, UK., 2011.

E. M. Vecchi, M. Baroni, and R. Zamparelli. (Linear) maps of the impossible: Capturing semantic anomalies in distributional space. In *Proceedings of the ACL Workshop on Distributional Semantics and Compositionality*, pages 1–9, Portland, OR, 2011.

J. Weeds, D. Weir, and D. McCarthy. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1015. Association for Computational Linguistics, 2004.

J. Weeds, D. Clarke, J. Reffin, D. Weir, and B. Keller. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING*, pages 2249–2259, Dublin, Ireland, 2014.

E. Wisniewski. When concepts combine. *Psychonomic Bulletin & Review*, 4(2):167–183, 1997. ISSN 1069-9384.

P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

F. Zanzotto, I. Korkontzelos, F. Falucchi, and S. Manandhar. Estimating linear models for compositional distributional semantics. In *Proceedings of COLING*, pages 1263–1271, Beijing, China, 2010.