Contextualizing Semantic Representations Using Syntactically Enriched Vector Models

Stefan Thater and **Hagen Fürstenau** and **Manfred Pinkal**
Department of Computational Linguistics
Saarland University
{stth, hagenf, pinkal}@coli.uni-saarland.de

Abstract

We present a syntactically enriched vector model that supports the computation of contextualized semantic representations in a quasi compositional fashion. It employs a systematic combination of first- and second-order context vectors. We apply our model to two different tasks and show that (i) it substantially outperforms previous work on a paraphrase ranking task, and (ii) achieves promising results on a word-sense similarity task; to our knowledge, it is the first time that an unsupervised method has been applied to this task.

1 Introduction

In the logical paradigm of natural-language semantics originating from Montague (1973), semantic structure, composition and entailment have been modelled to an impressive degree of detail and formal consistency. These approaches, however, lack coverage and robustness, and their impact on realistic natural-language applications is limited: The logical framework suffers from over-specificity, and is inappropriate to model the pervasive vagueness, ambivalence, and uncertainty of natural-language semantics. Also, the handcrafting of resources covering the huge amounts of content which are required for deep semantic processing is highly inefficient and expensive.

Co-occurrence-based semantic vector models offer an attractive alternative. In the standard approach, word meaning is represented by feature vectors, with large sets of context words as dimensions, and their co-occurrence frequencies as values. Semantic similarity information can be acquired using unsupervised methods at virtually no cost, and the information gained is soft and gradual. Many NLP tasks have been modelled successfully using vector-based models. Examples include information retrieval (Manning et al., 2008), word-sense discrimination (Schütze, 1998) and disambiguation (McCarthy and Carroll, 2003), to name but a few.

Standard vector-space models have serious limitations, however: While semantic information is typically encoded in phrases and sentences, distributional semantics, in sharp contrast to logic-based semantics, does not offer any natural concept of compositionality that would allow the semantics of a complex expression to be computed from the meaning of its parts. A different, but related problem is caused by word-sense ambiguity and contextual variation of usage. Frequency counts of context words for a given target word provide invariant representations averaging over all different usages of the target word. There is no obvious way to distinguish the different senses of e.g. *acquire* in different contexts, such as *acquire knowledge* or *acquire shares*.

Several approaches for word-sense disambiguation in the framework of distributional semantics have been proposed in the literature (Schütze, 1998; McCarthy and Carroll, 2003). In contrast to these approaches, we present a method to model the mutual contextualization of words in a phrase in a compositional way, guided by syntactic structure. To some extent, our method resembles the approaches proposed by Mitchell and Lapata (2008) and Erk and Padó (2008). We go one step further, however, in that we employ *syntactically enriched vector models* as the basic meaning representations, assuming a vector space spanned by combinations of dependency relations and words (Lin, 1998). This allows us to model the semantic interaction between the meaning of a head word and its dependent at the micro-level of relation-specific co-occurrence frequencies. It turns out that the benefit to precision is considerable.

Using syntactically enriched vector models raises problems of different kinds: First, the use

of syntax increases dimensionality and thus may cause data sparseness (Padó and Lapata, 2007). Second, the vectors of two syntactically related words, e.g., a target verb *acquire* and its direct object *knowledge*, typically have different syntactic environments, which implies that their vector representations encode complementary information and there is no direct way of combining the information encoded in the respective vectors.

To solve these problems, we build upon previous work (Thater et al., 2009) and propose to use *syntactic second-order vector representations*. Second-order vector representations in a bag-of-words setting were first used by Schütze (1998); in a syntactic setting, they also feature in Dligach and Palmer (2008). For the problem at hand, the use of second-order vectors alleviates the sparseness problem, and enables the definition of vector space transformations that make the distributional information attached to words in different syntactic positions compatible. Thus, it allows vectors for a predicate and its arguments to be combined in a compositional way.

We conduct two experiments to assess the suitability of our method. Our first experiment is carried out on the SemEval 2007 lexical substitution task dataset (McCarthy and Navigli, 2007). It will show that our method significantly outperforms other unsupervised methods that have been proposed in the literature to rank words with respect to their semantic similarity in a given linguistic context. In a second experiment, we apply our model to the "word sense similarity task" recently proposed by Erk and McCarthy (2009), which is a refined variant of a word-sense disambiguation task. The results show a substantial positive effect.

Plan of the paper. We will first review related work in Section 2, before presenting our model in Section 3. In Sections 4 and 5 we evaluate our model on the two different tasks. Section 6 concludes.

2 Related Work

Several approaches to contextualize vector representations of word meaning have been proposed. One common approach is to represent the meaning of a word a in context b simply as the sum, or centroid of a and b (Landauer and Dumais, 1997).

Kintsch (2001) considers a variant of this simple model. By using vector representations of a predicate p and an argument a, Kintsch identifies words that are similar to p and a, and takes the centroid of these words' vectors to be the representation of the complex expression $p(a)$.

Mitchell and Lapata (2008), henceforth M&L, propose a general framework in which meaning representations for complex expressions are computed compositionally by combining the vector representations of the individual words of the complex expression. They focus on the assessment of different operations combining the vectors of the subexpressions. An important finding is that component-wise multiplication outperforms the more common addition method. Although their composition method is guided by syntactic structure, the actual instantiations of M&L's framework are insensitive to syntactic relations and word-order, assigning identical representation to *dog bites man* and *man bites dog* (see Erk and Padó (2008) for a discussion). Also, they use syntax-free bag-of-words-based vectors as basic representations of word meaning.

Erk and Padó (2008), henceforth E&P, represent the meaning of a word w through a collection of vectors instead of a single vector: They assume selectional preferences and inverse selectional preferences to be constitutive parts of the meaning in addition to the meaning proper. The interpretation of a word p in context a is a combination of p's meaning with the (inverse) selectional preference of a. Thus, a verb meaning does not combine directly with the meaning of its object noun, as on the M&L account, but with the centroid of the vectors of the verbs to which the noun can stand in an object relation. Clearly, their approach is sensitive to syntactic structure. Their evaluation shows that their model outperforms the one proposed by M&L on a lexical substitution task (see Section 4). The basic vectors, however, are constructed in a word space similar to the one of the M&L approach.

In Thater et al. (2009), henceforth TDP, we took up the basic idea from E&P of exploiting selectional preference information for contextualization. Instead of using collections of different vectors, we incorporated syntactic information by assuming a richer internal structure of the vector representations. In a small case study, moderate improvements over E&P on a lexical substitution task could be shown. In the present paper, we formulate a general model of syntactically informed contextualization and show how to apply it to a number a of representative lexical substitution tasks. Evaluation shows significant improvements over TDP

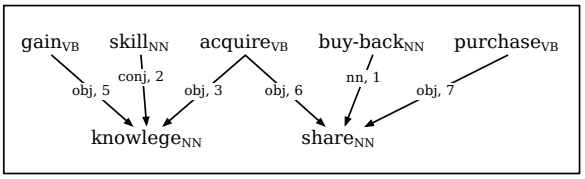

Figure 1: Co-occurrence graph of a small sample corpus of dependency trees.

and E&P.

3 The model

In this section, we present our method of contextualizing semantic vector representations. We first give an overview of the main ideas, which is followed by a technical description of first-order and second-order vectors (Section 3.2) and the contextualization operation (Section 3.3).

3.1 Overview

Our model employs vector representations for words and expressions containing syntax-specific first and second order co-occurrences information.

The basis for the construction of both kinds of vector representations are co-occurrence graphs. Figure 1 shows the co-occurrence graph of a small sample corpus of dependency trees: Words are represented as nodes in the graph, possible dependency relations between them are drawn as labeled edges, with weights corresponding to the observed frequencies. From this graph, we can directly read off the first-order vector for every word w: the vector's dimensions correspond to pairs (r, w') of a grammatical relation and a neighboring word, and are assigned the frequency count of (w, r, w').

The noun *knowledge*, for instance, would be represented by the following vector:

$$\langle 5_{(\text{OBJ}^{-1},\text{gain})}, 2_{(\text{CONJ}^{-1},\text{skill})}, 3_{(\text{OBJ}^{-1},\text{acquire})}, \ldots \rangle$$

This vector talks about the possible dependency heads of *knowledge* and thus can be seen as the (inverse) selectional preference of *knowledge* (see Erk and Padó (2008)).

As soon as we want to compute a meaning representation for a phrase like *acquire knowledge* from the verb *acquire* together with its direct object *knowledge*, we are facing the problem that verbs have different syntactic neighbors than nouns, hence their first-order vectors are not easily comparable. To solve this problem we additionally introduce another kind of vectors capturing informations about all words that can be reached with two steps in the co-occurrence graph. Such a path is characterized by two dependency relations and two words, i.e., a quadruple (r, w', r', w''), whose weight is the product of the weights of the two edges used in the path. To avoid overly sparse vectors we generalize over the "middle word" w' and build our second-order vectors on the dimensions corresponding to triples (r, r', w'') of two dependency relations and one word at the end of the two-step path. For instance, the second-order vector for *acquire* is

$$\langle 15_{(\text{OBJ},\text{OBJ}^{-1},\text{gain})}, 6_{(\text{OBJ},\text{CONJ}^{-1},\text{skill})}, 6_{(\text{OBJ},\text{OBJ}^{-1},\text{buy-back})}, 42_{(\text{OBJ},\text{OBJ}^{-1},\text{purchase})}, \ldots \rangle$$

In this simple example, the values are the products of the edge weights on each of the paths. The method of computation is detailed in Section 3.2. Note that second order vectors in particular contain paths of the form (r, r^{-1}, w'), relating a verb w to other verbs w' which are possible substitution candidates.

With first- and second-order vectors we can now model the interaction of semantic information within complex expressions. Given a pair of words in a particular grammatical relation like *acquire knowledge*, we contextualize the second-order vector of *acquire* with the first-order vector of *knowledge*. We let the first-order vector with its selectional preference information act as a kind of weighting filter on the second-order vector, and thus refine the meaning representation of the verb. The actual operation we will use is pointwise multiplication, which turned out to be the best-performing one for our purpose. Interestingly, Mitchell and Lapata (2008) came to the same result in a different setting.

In our example, we obtain a new second-order vector for *acquire* in the context of *knowledge*:

$$\langle 75_{(\text{OBJ},\text{OBJ}^{-1},\text{gain})}, 12_{(\text{OBJ},\text{CONJ}^{-1},\text{skill})}, 0_{(\text{OBJ},\text{OBJ}^{-1},\text{buy-back})}, 0_{(\text{OBJ},\text{OBJ}^{-1},\text{purchase})}, \ldots \rangle$$

Note that all dimensions that are not "licensed" by the argument *knowledge* are filtered out as they are multiplied with 0. Also, contextualisation of *acquire* with the argument *share* instead of *knowledge*

would have led to a very different vector, which reflects the fact that the two argument nouns induce different readings of the inherently ambiguous *acquire*.

3.2 First and second-order vectors

Assuming a set W of words and a set R of dependency relation labels, we consider a Euclidean vector space V_1 spanned by the set of orthonormal basis vectors $\{\vec{e}_{r,w'} \mid r \in R, w' \in W\}$, i.e., a vector space whose dimensions correspond to pairs of a relation and a word. Recall that any vector of V_1 can be represented as a finite sum of the form $\sum a_i \vec{e}_{r,w'}$ with appropriate scalar factors a_i. In this vector space we define the *first-order vector* $[w]$ of a word w as follows:

$$[w] = \sum_{\substack{r \in R \\ w' \in W}} \omega(w, r, w') \cdot \vec{e}_{r,w'}$$

where ω is a function that assigns the dependency triple (w, r, w') a corresponding weight. In the simplest case, ω would denote the frequency in a corpus of dependency trees of w occurring together with w' in relation r. In the experiments reported below, we use *pointwise mutual information* (Church and Hanks, 1990) instead as it proved superior to raw frequency counts:

$$pmi(w, r, w') = \log \frac{p(w, w' \mid r)}{p(w \mid r)p(w' \mid r)}$$

We further consider a similarly defined vector space V_2, spanned by an orthonormal basis $\{\vec{e}_{r,r',w'} \mid r, r' \in R, w' \in W\}$. Its dimensions therefore correspond to triples of two relations and a word. Evidently this is a higher dimensional space than V_1, which therefore can be embedded into V_2 by the "lifting maps" $L_r : V_1 \hookrightarrow V_2$ defined by $L_r(\vec{e}_{r',w'}) := \vec{e}_{r,r',w'}$ (and by linear extension therefore on all vectors of V_1). Using these lifting maps we define the *second-order vector* $[\![w]\!]$ of a word w as

$$[\![w]\!] = \sum_{\substack{r \in R \\ w' \in W}} \omega(w, r, w') \cdot L_r\left([w']\right)$$

Substituting the definitions of L_r and $[w']$, this yields

$$[\![w]\!] = \sum_{\substack{r, r' \in R \\ w'' \in W}} \left(\sum_{w' \in W} \omega(w, r, w')\omega(w', r', w'') \right) \vec{e}_{r,r',w''}$$

which shows the generalization over w' in form of the inner sum.

For example, if w is a verb, $r = \textsc{Obj}$ and $r' = \textsc{Obj}^{-1}$ (i.e., the inverse object relation), then the coefficients of $\vec{e}_{r,r',w''}$ in $[\![w]\!]$ would characterize the distribution of verbs w'' which share objects with w.

3.3 Composition

Both first and second-order vectors are defined for lexical expressions only. In order to represent the meaning of complex expressions we need to combine the vectors for grammatically related words in a given sentence. Given two words w and w' in relation r we contextualize the second-order vector of w with the r-lifted first-order vector of w':

$$[\![w_{r:w'}]\!] = [\![w]\!] \times L_r([w'])$$

Here $\times$ may denote any operator on V_2. The objective is to incorporate (inverse) selectional preference information from the context (r, w') in such a way as to identify the correct word sense of w. This suggests that the dimensions of $[\![w]\!]$ should be filtered so that only those compatible with the context remain. A more flexible approach than simple filtering, however, is to re-weight those dimensions with context information. This can be expressed by pointwise vector multiplication (in terms of the given basis of V_2). We therefore take $\times$ to be pointwise multiplication.

To contextualize (the vector of) a word w with multiple words $w_1, \ldots, w_n$ and corresponding relations $r_1, \ldots, r_n$, we compute the sum of the results of the pairwise contextualizations of the target vector with the vectors of the respective dependents:

$$[\![w_{r_1:w_1,\ldots,r_n:w_n}]\!] = \sum_{k=1}^{n} [\![w_{r_k:w_k}]\!]$$

4 Experiments: Ranking Paraphrases

In this section, we evaluate our model on a paraphrase ranking task. We consider sentences with an occurrence of some target word w and a list of paraphrase candidates $w_1, \ldots, w_k$ such that each of the w_i is a paraphrase of w for some sense of w. The task is to decide for each of the paraphrase candidates w_i how appropriate it is as a paraphrase of w in the given context. For instance, *buy*, *purchase* and *obtain* are all paraphrases of *acquire*, in the sense that they can be substituted for *acquire* in some contexts, but *purchase* and *buy* are not paraphrases of *acquire* in the first sentence of Table 1.

Sentence	Paraphrases
Teacher education students will *acquire* the knowledge and skills required to [...]	gain 4; amass 1; receive 1; obtain 1
Ontario Inc. will [...] *acquire* the remaining IXOS shares [...]	buy 3; purchase 1; gain 1; get 1; procure 2; obtain 1

Table 1: Two examples from the lexical substitution task data set

4.1 Resources

We use a vector model based on dependency trees obtained from parsing the English Gigaword corpus (LDC2003T05). The corpus consists of news from several newswire services, and contains over four million documents. We parse the corpus using the Stanford parser[1] (de Marneffe et al., 2006) and a non-lexicalized parser model, and extract over 1.4 billion dependency triples for about 3.9 million words (lemmas) from the parsed corpus.

To evaluate the performance of our model, we use various subsets of the SemEval 2007 lexical substitution task (McCarthy and Navigli, 2007) dataset. The complete dataset contains 10 instances for each of 200 target words—nouns, verbs, adjectives and adverbs—in different sentential contexts. Systems that participated in the task had to generate paraphrases for every instance, and were evaluated against a gold standard containing up to 10 possible paraphrases for each of the individual instances.

There are two natural subtasks in generating paraphrases: identifying paraphrase candidates and ranking them according to the context. We follow E&P and evaluate it only on the second subtask: we extract paraphrase candidates from the gold standard by pooling all annotated gold-standard paraphrases for all instances of a verb in all contexts, and use our model to rank these paraphrase candidates in specific contexts. Table 1 shows two instances of the target verb *acquire* together with its paraphrases in the gold standard as an example. The paraphrases are attached with weights, which correspond to the number of times they have been given by different annotators.

4.2 Evaluation metrics

To evaluate the performance of our method we use *generalized average precision* (Kishida, 2005), a variant of *average precision.*

Average precision (Buckley and Voorhees, 2000) is a measure commonly used to evaluate systems that return ranked lists of results. Generalized average precision (GAP) additionally rewards the correct order of positive cases w.r.t. their gold standard weight. We define average precision first:

$$AP = \frac{\Sigma_{i=1}^{n} x_i \; p_i}{R} \qquad p_i = \frac{\Sigma_{k=1}^{i} x_k}{i}$$

where x_i is a binary variable indicating whether the ith item as ranked by the model is in the gold standard or not, R is the size of the gold standard, and n is the number of paraphrase candidates to be ranked. If we take x_i to be the gold standard weight of the ith item or zero if it is not in the gold standard, we can define *generalized average precision* as follows:

$$GAP = \frac{\sum_{i=1}^{n} I(x_i) \; p_i}{\sum_{i=1}^{R} I(y_i)\bar{y}_i}$$

where $I(x_i) = 1$ if x_i is larger than zero, zero otherwise, and $\bar{y}_i$ is the average weight of the ideal ranked list $y_1, \ldots, y_i$ of gold standard paraphrases.

As a second scoring method, we use *precision out of ten* (P_{10}). The measure is less discriminative than GAP. We use it because we want to compare our model with E&P. P_{10} measures the percentage of gold-standard paraphrases in the top-ten list of paraphrases as ranked by the system, and can be defined as follows (McCarthy and Navigli, 2007):

$$P_{10} = \frac{\Sigma_{s \in M \cap G} \; f(s)}{\Sigma_{s \in G} \; f(s)},$$

where M is the list of 10 paraphrase candidates top-ranked by the model, G is the corresponding annotated gold-standard data, and $f(s)$ is the weight of the individual paraphrases.

4.3 Experiment 1: Verb paraphrases

In our first experiment, we consider verb paraphrases using the same controlled subset of the

[1] We use version 1.6 of the parser. We modify the dependency trees by "folding" prepositions into the edge labels to make the relation between a head word and the head noun of a prepositional phrase explicit.

lexical substitution task data that had been used by TDP in an earlier study. We compare our model to various baselines and the models of TDP and E&P, and show that our new model substantially outperforms previous work.

Dataset. The dataset is identical to the one used by TDP and has been constructed in the same way as the dataset used by E&P: it contains those gold-standard instances of verbs that have—according to the analyses produced by the MiniPar parser (Lin, 1993)—an overtly realized subject and object. Gold-standard paraphrases that do not occur in the parsed British National Corpus are removed.[2] In total, the dataset contains 162 instances for 34 different verbs. On average, target verbs have 20.5 substitution candidates; for individual instances of a target verb, an average of 3.9 of the substitution candidates are annotated as correct paraphrases. Below, we will refer to this dataset as "LST/SO."

Experimental procedure. To compute the vector space, we consider only a subset of the complete set of dependency triples extracted from the parsed Gigaword corpus. We experimented with various strategies, and found that models which consider all dependency triples exceeding certain *pmi*- and frequency thresholds perform best.

Since the dataset is rather small, we use a four-fold cross-validation method for parameter tuning: We divide the dataset into four subsets, test various parameter settings on one subset and use the parameters that perform best (in terms of GAP) to evaluate the model on the three other subsets. We consider the following parameters: *pmi*-thresholds for the dependency triples used in the computation of the first- and second-order vectors, and frequency thresholds. The parameters differ only slightly between the four subsets, and the general tendency is that good results are obtained if a low *pmi*-threshold (≤ 2) is applied to filter dependency triples used in the computation of the second-order vectors, and a relatively high *pmi*-threshold (≥ 4) to filter dependency triples in the computation of the first-order vectors. Good performing frequency thresholds are 10 or 15. The threshold values for context vectors are slightly different: a medium *pmi*-threshold between 2 and 4 and a low frequency threshold of 3.

To rank paraphrases in context, we compute contextualized vectors for the verb in the input sentence, i.e., a second order vector for the verb that is contextually constrained by the first order vectors of all its arguments, and compare them to the unconstrained (second-order) vectors of each paraphrase candidate, using cosine similarity.[3] For the first sentence in Table 1, for example, we compute $[\![\text{acquire}_{\text{SUBJ:student,OBJ:knowledge}}]\!]$ and compare it to $[\![\text{gain}]\!]$, $[\![\text{amass}]\!]$, $[\![\text{buy}]\!]$, $[\![\text{purchase}]\!]$ and so on.

Baselines. We evaluate our model against a random baseline and two variants of our model: One variant ("2^{nd} order uncontexualized") simply uses contextually unconstrained second-order vectors to rank paraphrase candidates. Comparing the full model to this variant will show how effective our method of contextualizing vectors is. The second variant ("1^{st} order contextualized") represents verbs in context by their first order vectors that specify how often the verb co-occurs with its arguments in the parsed Gigaword corpus. We compare our model to this baseline to demonstrate the benefit of (contextualized) second-order vectors. As for the full model, we use *pmi* values rather than raw frequency counts as co-occurrence statistics.

Results. For the LST/SO dataset, the generalized average precision, averaged over all instances in the dataset, is 45.94%, and the average P_{10} is 73.11%.

Table 2 compares our model to the random baseline, the two variants of our model, and previous work. As can be seen, our model improves about 8% in terms of GAP and almost 7% in terms of P_{10} upon the two variants of our model, which in turn perform 10% above the random baseline. We conclude that both the use of second-order vectors, as well as the method used to contextualize them, are very effective for the task under consideration.

The table also compares our model to the model of TDP and two different instantiations of E&P's model. The results for these three models are cited from Thater et al. (2009). We can observe that our model improves about 9% in terms of GAP and about 7% in terms of P_{10} upon previous work. Note that the results for the E&P models are based

[2] Both TDP and E&P use the British National Corpus.

[3] Note that the context information is the same for both words. With our choice of pointwise multiplication for the composition operator $\times$ we have $(\vec{v}_1 \times \vec{w}) \cdot \vec{v}_2 = \vec{v}_1 \cdot (\vec{v}_2 \times \vec{w})$. Therefore the choice of which word is contextualized does not strongly influence their cosine similarity, and contextualizing both should not add any useful information. On the contrary we found that it even lowers performance. Although this could be repaired by appropriately modifying the operator $\times$, for this experiment we stick with the easier solution of only contextualizing one of the words.

Model	GAP	P_{10}
Random baseline	26.03	54.25
E&P (add, object)	29.93	66.20
E&P (min, subject & object)	32.22	64.86
TDP	36.54	63.32
1^{st} order contextualized	36.09	59.35
2^{nd} order uncontextualized	37.65	66.32
Full model	**45.94**	**73.11**

Table 2: Results of Experiment 1

on a reimplementation of E&P's original model—the P_{10}-scores reported by Erk and Padó (2009) range between 60.2 and 62.3, over a slightly lower random baseline.

According to a paired t-test the differences are statistically significant at $p < 0.01$.

Performance on the complete dataset. To find out how our model performs on less controlled datasets, we extracted all instances from the lexical substitution task dataset with a verb target, excluding only instances which could not be parsed by the Stanford parser, or in which the target was mistagged as a non-verb by the parser. The resulting dataset contains 496 instances. As for the LST/SO dataset, we ignore all gold-standard paraphrases that do not occur in the parsed (Gigaword) corpus.

If we use the best-performing parameters from the first experiment, we obtain a GAP score of 45.17% and a P_{10}-score of 75.43%, compared to random baselines of 27.42% (GAP) and 58.83% (P_{10}). The performance on this larger dataset is thus almost the same compared to our results for the more controlled dataset. We take this as evidence that our model is quite robust w.r.t. different realizations of a verb's subcategorization frame.

4.4 Experiment 2: Non-verb paraphrases

We now apply our model to parts of speech (POS) other than verbs. The main difference between verbs on the one hand, and nouns, adjectives, and adverbs on the other hand, is that verbs typically come with a rich context—subject, object, and so on—while non-verbs often have either no dependents at all or only closed class dependents such as determiners which provide only limited contextual informations, if any at all. While we can apply the same method as before also to non-verbs, we might expect it to work less well due to limited contextual information.

POS	Instances	M1	M2	Baseline
Noun	535	46.38	42.54	30.01
Adj	508	39.41	43.21	28.32
Adv	284	48.19	51.43	37.25

Table 3: GAP-scores for non-verb paraphrases using two different methods.

We therefore propose an alternative method to rank non-verb paraphrases: We take the second-order vector of the target's head and contextually constrain it by the first order vector of the target. For instance, if we want to rank the paraphrase candidates *hint* and *star* for the noun *lead* in the sentence

(1) Meet for coffee early, swap *leads* and get permission to contact if possible.

we compute $[\![\text{swap}_{\text{OBJ:lead}}]\!]$ and compare it to the lifted first-order vectors of all paraphrase candidates, $L_{\text{OBJ}}([\text{hint}])$ and $L_{\text{OBJ}}([\text{star}])$, using cosine similarity.

To evaluate the performance of the two methods, we extract all instances from the lexical substitution task dataset with a nominal, adjectival, or adverbial target, excluding instances with incorrect parse or no parse at all. As before, we ignore gold-standard paraphrases that do not occur in the parsed Gigaword corpus.

The results are shown in Table 3, where "M1" refers to the method we used before on verbs, and "M2" refers to the alternative method described above. As one can see, M1 achieves better results than M2 if applied to nouns, while M2 is better than M1 if applied to adjectives and adverbs. The second result is unsurprising, as adjectives and adverbs often have no dependents at all.

We can observe that the performance of our model is similarly strong on non-verbs. GAP scores on nouns (using M1) and adverbs are even higher than those on verbs. We take these results to show that our model can be successfully applied to all open word classes.

5 Experiment: Ranking Word Senses

In this section, we apply our model to a different word sense ranking task: Given a word w in context, the task is to decide to what extent the different

WordNet (Fellbaum, 1998) senses of *w* apply to this occurrence of *w*.

Dataset. We use the dataset provided by Erk and McCarthy (2009). The dataset contains ordinal judgments of the applicability of WordNet senses on a 5 point scale, ranging from *completely different* to *identical* for eight different lemmas in 50 different sentential contexts. In this experiment, we concentrate on the three verbs in the dataset: *ask*, *add* and *win*.

Experimental procedure. Similar to Pennacchiotti et al. (2008), we represent different word senses by the words in the corresponding synsets. For each word sense, we compute the centroid of the second-order vectors of its synset members. Since synsets tend to be small (they even may contain only the target word itself), we additionally add the centroid of the sense's hypernyms, scaled down by the factor 10 (chosen as a rough heuristic without any attempt at optimization).

We apply the same method as in Section 4.3: For each instance in the dataset, we compute the second-order vector of the target verb, contextually constrain it by the first-order vectors of the verb's arguments, and compare the resulting vector to the vectors that represent the different WordNet senses of the verb. The WordNet senses are then ranked according to the cosine similarity between their sense vector and the contextually constrained target verb vector.

To compare the predicted ranking to the gold-standard ranking, we use Spearman's ρ, a standard method to compare ranked lists to each other. We compute ρ between the similarity scores averaged over all three annotators and our model's predictions. Based on agreement between human judges, Erk and McCarthy (2009) estimate an upper bound ρ of 0.544 for the dataset.

Results. Table 4 shows the results of our experiment. The first column shows the correlation of our model's predictions with the human judgments from the gold-standard, averaged over all instances. All correlations are significant ($p < 0.001$) as tested by approximate randomization (Noreen, 1989).

The second column shows the results of a frequency-informed baseline, which predicts the ranking based on the order of the senses in WordNet. This (weakly supervised) baseline outperforms our unsupervised model for two of the three verbs. As a final step, we explored the effect of combining our rankings with those of the frequency baseline, by simply computing the average ranks of those two models. The results are shown in the third column. Performance is significantly higher than for both the original model and the frequency-informed baseline. This shows that our model captures an additional kind of information, and thus can be used to improve the frequency-based model.

Word	Present paper	WN-Freq	Combined
ask	0.344	0.369	0.431
add	0.256	0.164	0.270
win	0.236	0.343	0.381
average	*0.279*	*0.291*	*0.361*

Table 4: Correlation of model predictions and human judgments

6 Conclusion

We have presented a novel method for adapting the vector representations of words according to their context. In contrast to earlier approaches, our model incorporates detailed syntactic information. We solved the problems of data sparseness and incompatibility of dimensions which are inherent in this approach by modeling contextualization as an interplay between first- and second-order vectors.

Evaluating on the SemEval 2007 lexical substitution task dataset, our model performs substantially better than all earlier approaches, exceeding the state of the art by around 9% in terms of generalized average precision and around 7% in terms of precision out of ten. Also, our system is the first unsupervised method that has been applied to Erk and McCarthy's (2009) graded word sense assignment task, showing a substantial positive correlation with the gold standard. We further showed that a weakly supervised heuristic, making use of WordNet sense ranks, can be significantly improved by incorporating information from our system.

We studied the effect that context has on target words in a series of experiments, which vary the target word and keep the context constant. A natural objective for further research is the influence of varying contexts on the meaning of target expressions. This extension might also shed light on the status of the modelled semantic process, which we have been referring to in this paper as "contextualization". This process can be considered one of

mutual disambiguation, which is basically the view of E&P. Alternatively, one can conceptualize it as semantic composition: in particular, the head of a phrase incorporates semantic information from its dependents, and the final result may to some extent reflect the meaning of the whole phrase.

Another direction for further study will be the generalization of our model to larger syntactic contexts, including more than only the direct neighbors in the dependency graph, ultimately incorporating context information from the whole sentence in a recursive fashion.

Acknowledgments. We would like to thank Eduard Hovy and Georgiana Dinu for inspiring discussions and helpful comments. This work was supported by the Cluster of Excellence "Multimodal Computing and Interaction", funded by the German Excellence Initiative, and the project SALSA, funded by DFG (German Science Foundation).

References

Chris Buckley and Ellen M. Voorhees. 2000. Evaluating evaluation measure stability. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 33–40, Athens, Greece.

Kenneth W. Church and Patrick Hanks. 1990. Word association, mutual information and lexicography. *Computational Linguistics*, 16(1):22–29.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the fifth international conference on Language Resources and Evaluation (LREC 2006)*, pages 449–454, Genoa, Italy.

Dmitriy Dligach and Martha Palmer. 2008. Novel semantic features for verb sense disambiguation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 29–32, Columbus, OH, USA.

Katrin Erk and Diana McCarthy. 2009. Graded word sense assignment. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 440–449, Singapore.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Honolulu, HI, USA.

Katrin Erk and Sebastian Padó. 2009. Paraphrase assessment in structured vector space: Exploring parameters and datasets. In *Proc. of the Workshop on Geometrical Models of Natural Language Semantics*, Athens, Greece.

Christiane Fellbaum, editor. 1998. *Wordnet: An Electronic Lexical Database*. Bradford Book.

Walter Kintsch. 2001. Predication. *Cognitive Science*, 25:173–202.

Kazuaki Kishida. 2005. Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments. *NII Technical Report*.

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

Dekang Lin. 1993. Principle-based parsing without overgeneration. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 112–120, Columbus, OH, USA.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 768–774.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Diana McCarthy and John Carroll. 2003. Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences. *Computational Linguistics*, 29(4):639–654.

Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 Task 10: English Lexical Substitution Task. In *Proc. of SemEval*, Prague, Czech Republic.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, OH, USA.

Richard Montague. 1973. The proper treatment of quantification in ordinary English. In Jaakko Hintikka, Julius Moravcsik, and Patrick Suppes, editors, *Approaches to Natural Language*, pages 221–242. Dordrecht.

Eric W. Noreen. 1989. *Computer-intensive Methods for Testing Hypotheses: An Introduction*. John Wiley and Sons Inc.

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

Marco Pennacchiotti, Diego De Cao, Roberto Basili, Danilo Croce, and Michael Roth. 2008. Automatic induction of framenet lexical units. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 457–465, Honolulu, HI, USA.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124.

Stefan Thater, Georgiana Dinu, and Manfred Pinkal. 2009. Ranking paraphrases in context. In *Proceedings of the 2009 Workshop on Applied Textual Inference*, pages 44–47, Singapore.