Paraphrase assessment in structured vector space: Exploring parameters and datasets

Katrin Erk
Department of Linguistics
University of Texas at Austin
katrin.erk@mail.utexas.edu

Sebastian Padó
Department of Linguistics
Stanford University
pado@stanford.edu

Abstract

The appropriateness of paraphrases for words depends often on context: "grab" can replace "catch" in "catch a ball", but not in "catch a cold". Structured Vector Space (SVS) (Erk and Padó, 2008) is a model that computes *word meaning in context* in order to assess the appropriateness of such paraphrases. This paper investigates "best-practice" parameter settings for SVS, and it presents a method to obtain large datasets for paraphrase assessment from corpora with WSD annotation.

1 Introduction

The meaning of individual occurrences or *tokens* of a word can change vastly according to its context. A central challenge for computational lexical semantics is describe these *token meanings* and how they can be computed for new occurrences.

One prominent approach to this question is the *dictionary-based model* of token meaning: The different meanings of a word are a set of distinct, disjoint senses enumerated in a lexicon or ontology, such as WordNet. For each new occurrence, determining token meaning means choosing one of the senses, a classification task known as Word Sense Disambiguation (WSD). Unfortunately, this task has turned out to be very hard both for human annotators and for machines (Kilgarriff and Rosenzweig, 2000), not at least due to granularity problems with available resources (Palmer et al., 2007; McCarthy, 2006). Some researchers have gone so far as to suggest fundamental problems with the concept of categorical word senses (Kilgarriff, 1997; Hanks, 2000).

An interesting alternative is offered by *vector space models* of word meaning (Lund and Burgess, 1996; McDonald and Brew, 2004) which characterize the meaning of a word entirely without reference to word senses. Word meaning is described in terms of a vector in a high-dimensional vector space that is constructed with distributional methods. Semantic similarity is then simply distance to vectors of other words. Vector space models have been most successful in modeling the meaning of word types (i.e. in constructing *type vectors*). The characterization of token meaning by corresponding *token vectors* would represent a very interesting alternative to dictionary-based methods by providing a direct, graded, unsupervised measure of (dis-)similarity between words in context that completely avoids reference to dictionary senses. However, there are still considerable theoretical and practical problems, even though there is a substantial body of work (Landauer and Dumais, 1997; Schütze, 1998; Kintsch, 2001; Mitchell and Lapata, 2008).

In a recent paper (Erk and Padó, 2008), we have introduced the *structured vector space (SVS)* model which addresses this challenge. It yields one token vector per input word. Token vectors are not computed by combining the lexical meaning of the surrounding words – which risks resulting in a "topicality" vector – but by modifying the type meaning of a word with the semantic expectations of syntactically related words, which can be thought of as selectional preferences. For example, in *catch a ball*, the token vector for *ball* is computed by combining the type vector of *ball* with a vector for the *selectional preferences* of *catch* for its object. The token vector for *catch*, conversely, is constructed from the type vector of *catch* and the *inverse object preference vector* of *ball*. The resulting token vectors describe the meaning of a word in a particular sentence not through a sense label, but through the distance of the token vector to other vectors.

A natural question that arises is how vector-based models of token meaning can be evaluated. It is of course possible to apply them to a traditional WSD task. However, this strategy remains vulnerable to all criticism concerning the annotation of categorical word senses, and also does not take advantage of the vector models' central asset, namely gradedness. Thus, *paraphrase-based assessment for models of token meaning* was proposed as a representation-neutral disambiguation task that can replace WSD (McCarthy and Navigli, 2007; Mitchell and Lapata, 2008). Given a word token in context and a set of potential paraphrases, the task consists of identifying the *subset* of valid paraphrases. For example, in the following example, the first paraphrase is appropriate, but the second is not:

(1) Google *acquired* YouTube ⇒
Google *bought* YouTube

(2) How children *acquire* skills ⇏
How children *buy* skills

This task is graded in the sense that there is no disjoint set of labels from which exactly one is picked for each token; rather, the paraphrases form a set of labels of which a subset is appropriate for each word token,

Proceedings of the EACL 2009 Workshop on GEMS: GEometical Models of Natural Language Semantics, pages 57–65,
Athens, Greece, 31 March 2009.

and the appropriate sets for two tokens may overlap to varying degrees. In an ideal vector-based model, valid paraphrases such as (1) should possess similar vectors, and invalid ones such as (2) dissimilar ones.

In Erk and Padó (2008), we evaluated SVS on two variants of the paraphrase assessment test: first, the prediction of human judgments on a seven-point scale for paraphrases for verb-subject pairs (Mitchell and Lapata, 2008); and second, the original Lexical Substitution task by McCarthy and Navigli (2007). To avoid overfitting, we optimized our parameters on the first dataset and evaluated only the best model on the second dataset. However, given evidence for substantial inter-task differences, it is unclear to what extent these parameters are optimal beyond the Mitchell and Lapata dataset. This paper addresses this question with two experiments:

Impact of parameters. We re-examine three central parameters of SVS. The first one is the choice of *vector combination function*. Following Mitchell and Lapata (2008), we previously used componentwise multiplication, whose interpretation in vector space is not straightforward. The second one is *reweighting*. We obtained the best performance when the context expectations were reweighted by taking each component to a (high) n-th power, which is counterintuitive. Finally, we found subjects to be more informative in judging the appropriateness of paraphrases than objects. This appears to contradict work in theoretical syntax (Levin and Rappaport Hovav, 2005).

To reassess the role of these parameters, we construct a controlled dataset of transitive instances from the Lexical Substitution corpus to reexamine and investigate these issues, with the aim of providing "best practice" settings for SVS. This turns out to be more difficult than expected, leading us to suspect that a globally optimal parameter setting across tasks may simply not exist. We also test a simple extension of SVS that uses a richer context (both subject and object) to construct the token vector, with first positive results.

Dataset creation. The Lexical Substitution dataset used in Erk and Padó (2008) was very small, which limits the conclusions that can be drawn from it. This points towards a more general problem of paraphrase-based assessment for models of token meaning: Until now, all datasets for this task were specifically created by hand. It would provide a strong boost for paraphrase assessment if the large annotated corpora that are available for WSD could be reused.

We present an experiment on converting the WordNet-annotated SemCor corpus into a set of "pseudo-paraphrases" for paraphrase-based assessment. We use the synonyms and direct hypernyms of an annotated synset as these "pseudo-paraphrases". While the synonyms and hypernyms are not guaranteed to work as direct replacements of the target word in the given context, they are semantically similar to the target word. The result is a dataset ten times larger than the Lex-Sub dataset. As we describe in this paper, we find that this method is nevertheless problematic: The resulting dataset is considerably more difficult to model than the existing hand-built paraphrase corpora, and its properties differ considerably from the manually constructed Lexical Substitution dataset.

2 The structured vector space model

The main intuition behind the SVS model is to treat the interpretation of a word in context as guided by *expectations about typical events*. This move to include typical arguments and predicates into a model of word meaning is motivated both on cognitive and linguistic grounds. In cognitive science, the central role of expectations about typical events on almost all aspects of human language processing is well-established (McRae et al., 1998; Narayanan and Jurafsky, 2002). In linguistics, expectations have long been used in semantic theories in the form of *selectional restrictions* and *selectional preferences* (Wilks, 1975), and more recently induced from corpora (Resnik, 1996). Attention has mostly been limited to selectional preferences of verbs, which have been used for for a variety of tasks (Hindle and Rooth, 1993; Gildea and Jurafsky, 2002). A recent result that the SVS model builds on is that selectional preferences can be represented as *prototype* vectors constructed from seen arguments (Erk, 2007; Padó et al., 2007).

Representing lemma meaning. To accommodate information about semantic expectations, the SVS model extends the traditional representation of word meaning as a single vector by a set of vectors, each of which represents the word's *selectional preferences* for each relation that the word can assume in its linguistic context. While we ultimately think of these relations as "properly semantic" in the sense of semantic roles, the instantiation of SVS we consider in this paper makes use of dependency relations as a level of representation that generalizes over a substantial amount of surface variation but that can be obtained automatically with high accuracy using current NLP tools.

The idea is illustrated in Figure 1. In the representation of the verb *catch*, the central square stands for the lexical vector of *catch* itself. The three arrows link it to *catch*'s preferences for dependency relations it can participate in, such as for its $subj$ects, its objects, and for verbs for which it appears as a complement ($comp^{-1}$). The figure shows the head words that enter into the computation of the selectional preference vector. Likewise, *ball* is represented by one vector for *ball* itself, one for *ball*'s preferences for its modifiers (mod), and two for the verbs of which it can occur as a subject ($subj^{-1}$) and an object (obj^{-1}), respectively.

This representation includes selectional preferences (like $subj$, obj, mod) exactly parallel to *inverse* selectional preferences ($subj^{-1}$, obj^{-1}, $comp^{-1}$). The SVS model is then formalized as follows. Let D be a vector space, and let $\mathcal{R}$ be some set of relation labels. We then

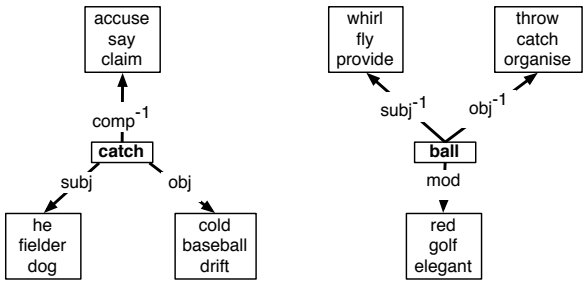

Figure 1: Structured Vector Space representations for noun *ball* and verb *catch*: Each box represents one vector (lexical information or expectations)

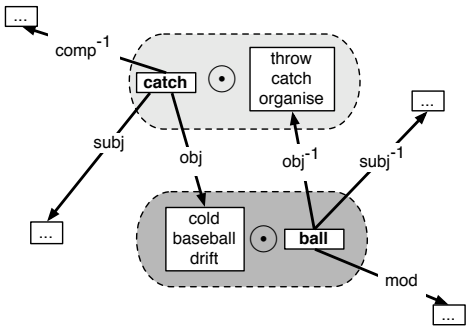

Figure 2: Combining predicate and argument via relation-specific semantic expectations

represent the meaning of a lemma w as a triple

$$m(w) = (v_w, R, R^{-1})$$

where $v_w \in D$ is the type vector of the word w itself, $R : \mathcal{R} \rightarrow D$ maps each relation label onto a vector that describes w's selectional preferences, and $R^{-1} : \mathcal{R} \rightarrow D$ maps from role labels to vectors describing inverse selectional preferences of w. Both R and R^{-1} are partial functions. For example, the direct object preference is undefined for intransitive verbs.[1]

Computing meaning in context. SVS computes the meaning of a word a in the context of another word b via their selectional preferences as follows: Let $m(a) = (v_a, R_a, R_a^{-1})$ and $m(b) = (v_b, R_b, R_b^{-1})$ be the representations of the two words, and let $r \in \mathcal{R}$ be the relation linking a to b. Then, the meaning of a and b in this context is defined as a pair of structured vector triples: $m(a \xrightarrow{r} b)$ is the meaning of a with b as its r-argument, and $m(b \xrightarrow{r^{-1}} a)$ the meaning of b as the r-argument of a:

$$\begin{aligned} m(a \xrightarrow{r} b) &= \left(v_a \odot R_b^{-1}(r), R_a - \{r\}, R_a^{-1}\right) \\ m(b \xrightarrow{r^{-1}} a) &= \left(v_b \odot R_a(r), R_b, R_b^{-1} - \{r\}\right) \end{aligned} \tag{3}$$

where $v_1 \odot v_2$ is a direct vector combination function as in traditional models, e.g. addition or component-wise multiplication. If either $R_a(r)$ or $R_b^{-1}(r)$ are not defined, the combination fails. Afterward, the filled argument position r is deleted from R_a and R_b^{-1}.

Figure 2 illustrates the procedure on the representations from Figure 1. The dotted lines indicate that the lexical vector for *catch* is combined with the inverse object preference of *ball*. Likewise, the lexical vector for *ball* combines with the object preference vector of *catch*.

Recursive application. In Erk and Padó (2008), we considered only one combination step; however, the syntactic context of a word in a dependency tree often consists of more than one word. It seems intuitively plausible that disambiguation should profit from more context information. Thus, we extend SVS with recursive application. Let a stand in relation r to b. As defined above, the result of combining $m(a)$ and $m(b)$ by relation r are two structured vector triples $m(a \xrightarrow{r} b)$ and $m(b \xrightarrow{r^{-1}} a)$. If a also stands in relation $s \neq r$ to a word c with $m(c) = (v_a, R_a, R_a^{-1})$, we define the meaning of a in the context of b and c canonically as

$$\begin{aligned} m(m(a \xrightarrow{r} b) \xrightarrow{s} c) &= \big((v_a \odot R_b^{-1}(r)) \odot R_c^{-1}(s), \\ &\quad R_a - \{r, s\}, R_a^{-1}\big) \end{aligned} \tag{4}$$

If $\odot$ is associative and commutative, then $m(m(a \xrightarrow{r} b) \xrightarrow{s} c) = m(m(a \xrightarrow{s} c) \xrightarrow{r} b)$. This will be the case for all the combination functions we use in this paper.

Note that this is a simplistic model of the influence of multiple context words: it computes only lexical meaning recursively, but does not model the influence of context on the *selectional preferences*. For example, the subject selectional preferences of *catch* are identical to those of *catch the ball*, even though one would expect that the *outfielder* corresponds much better to the expectations of *catch the ball* than of just *catch*.

3 Experimental Setup

The task that we are considering is *paraphrase assessment in context*. Given a predicate-argument pair and a paraphrase candidate, the models have to decide how appropriate the paraphrase is for the predicate-argument combination. This is the main task against which token vector models have been evaluated in the past (Mitchell and Lapata, 2008; Erk and Padó, 2008). In Experiment 1, we use manually created paraphrases. In Experiment 2, we replaces human-generated paraphrases with "pseudo-paraphrases", contextually similar words that may not be completely appropriate as paraphrases in the given context, but can be collected automatically. Our parameter choices for SVS are as similar as possible to the second experiment of our earlier paper.

Vector space. We use a dependency-based vector space that counts a target word and a context word

[1] We use separate functions R, R^{-1} rather than a joint syntactic context preference function because (a) this separation models the conceptual difference between predicates and arguments, and (b) it allows for a simpler, more elegant formulation of the computation of meaning in context in Eq. 3.

as co-occurring in a sentence if they are connected by an "informative" path in the dependency graph for the sentence.[2] We build the space from a Minipar-parsed version of the British National Corpus with dependency parses obtained from Minipar (Lin, 1993). It uses raw co-occurrence counts and 2000 dimensions.

Selectional preferences and reweighting. We use a prototype-based selectional preference model (Erk, 2007). It models the selectional preferences of a predicate for an argument position as the weighted centroid of the vectors for all head words seen for this position in a large corpus. Let $f(a, r, b)$ denote the frequency of a occurring in relation r to b in the parsed BNC. Then, we compute the selectional preferences as:

$$R'_b(r) = \frac{1}{N} \sum_{a: f(a,r,b)>0} f(a, r, b) \cdot \vec{v}_a \qquad (5)$$

where N is the number of fillers a with $f(a, r, b) > 0$.

In Erk and Padó (2008), we found that applying a *reweighting* step to the selectional preference vector by taking each component of the centroid vector $R'_b(r)$ to the n-th power lead to substantial improvements. The motivation for this technique is to alleviate noise arising from the use of unfiltered head words for the construction. The reweighted selectional preference vector $R_b(r)$ is defined as:

$$R_b(r) = \langle v_1^n, \ldots, v_m^n \rangle \text{ for } R'_b(r) = \langle v_1, \ldots, v_m \rangle \quad (6)$$

where we write $\langle v_1, \ldots, v_m \rangle$ for the sequence of values that make up a vector $R'_b(r)$. Inverse selectional preferences $R_b^{-1}(r)$ of nouns are defined analogously, by computing the centroid of the verbs seen as governors of the noun in relation r.

In this paper, we test reweighting parameters of n between 0.5 and 30. Generally, small ns will decrease the influence of the selectional preference vector. The result can be thought of as a "word type vector modified by context expectations", while large ns increase the role of context, until we arrive at a "contextual expectation vector modified by the word type vector". [3]

Vector combination. We test three vector combination functions $\odot$, which have different interpretations in vector space. The simplest one is componentwise addition, abbreviated as **add**, i.e., simple vector addition.[4] With addition, context dimensions receive a high count whenever either of the two vectors has a high co-occurrence count for the context.

[2] We used the minimal context specification and plain weight of the DependencyVectors software package.

[3] For the component-wise minimum combination (see below), where we normalize the vectors before the combination, the reweighting has a different effect. It shifts most of the mass onto the largest-value dimensions and sets smaller dimensions to values close to zero.

[4] Since we subsequently focus on cosine similarity, which is length-invariant, vector addition can also be interpreted as centroid computation.

Next, we test component-wise multiplication (**mult**). This operation is more difficult to interpret in terms of vector space, since it does not correspond to the standard inner or outer vector products. The most straightforward interpretation is to reinterpret the second vector as a diagonal matrix, i.e., as a linear transformation of the first vector. Large entries in the second vector increase the weight of the corresponding contexts; small entries decrease it. Mitchell and Lapata (2008) found this method to yield the best results.

The third vector combination function we consider is component-wise minimum (**min**). This combination function results in a vector with high counts only for contexts which co-occur frequently with both input vectors and can thus be understood as an intersection between the two context sets. Since the entries of two vectors need to be on the same order to magnitude for this method to yield meaningful results, we normalize vectors before the combination for **min**.

Assessing models of token meaning. Given a transitive verb v with subject a and direct object b, we test three variants of computing a token vector for v. The first two involve only one combination step. In the **subj** condition, v's type vector is combined with the inverse subject preference vector of a. In the **obj** condition, v's type vector is combined with the inverse object preference vector of b. The third variant is the recursive application of the SVS combination procedure described in Section 2 (condition **both**). Specifically, we combine v's type vector with both a's inverse subject preference and with b's inverse object preference to obtain a "richer" token vector.

In all three cases, the resulting token vector is compared to the *type* vector of the paraphrase (in Experiment 1) or the semantically related word (in Experiment 2). We use Cosine Similarity, a standard choice as vector space similarity measure.

4 Experiments

4.1 Experiment 1: The impact of parameters

In our 2008 paper, we tested the LexSub data only with the parameters that showed best results on the Mitchell and Lapata data: vector combination using componentwise multiplication (*mult*), and the computation of (inverse) selectional preference vectors with high powers of $n = 20$ or $n = 30$. However, there were indications that the two datasets showed fundamental differences. In particular, the Mitchell and Lapata data could only be modeled using a PMI-transformed vector space, while the LexSub data could only be modeled using raw co-occurrence count vectors.

Another one of our findings that warrants further inquiry stems from our comparison of different context choices (verb plus subject, verb plus object, noun plus embedding verb). We found that subjects are better disambiguators than objects. This seems counterintuitive both on theoretical and empirical grounds. Theoretically,

Sentence	Substitutes
By asking people who **work** there, I have since determined that he didn't. (# 2002)	be employed 4; labour 1
Remember how hard your ancestors **worked**. (# 2005)	toil 4; labour 3; task 1

Figure 3: Lexical substitution example items for "work"

		0.5	1	2	5	10	20
add	obj	**61.5**	59.7	58.9	56.1	56.0	55.7
add	subj	**61.7**	**61.7**	59.5	58.4	57.3	57.0
add	both	**61.3**	60.0	60.2	57.7	57.1	56.7
mult	obj	**59.8**	59.7	57.8	55.7	55.7	55.4
mult	subj	**60.3**	59.7	59.3	57.3	57.7	56.7
mult	both	**59.9**	58.8	57.1	55.8	55.3	$<1^{\text{Pr}}$
min	obj	**60.2**	60.0	59.5	57.3	55.7	55.8
min	subj	**62.2**	60.5	59.1	58.5	57.8	57.0
min	both	**62.3**	60.2	59.8	57.3	55.8	55.1

Table 1: OOT accuracy on the LEXSUB-PARA dataset across models and reweighting values (best results for each model boldfaced). Random baseline: 53.7. Target type vector baseline: 57.1. $^{\text{Pr}}$: Numerical problem.

the notion of verb phrase has been motivated, among other things, with the claim that direct objects contribute more to a verb's disambiguation than subjects (Levin and Rappaport Hovav, 2005). Empirically, subjects are known to be realized more often as pronouns than objects, which makes their vector representations less semantically specific. However, we used *two different datasets* – the subject results on a set of intransitive verbs, and the object results on a set of transitive verbs, so the results are not comparable.

In this experiment, we construct a new, more controlled dataset from the Lexical Substitution corpus to systematically assess the importance of the three main parameters: the relation used for disambiguation, the combination function, and the reweighting parameter.

Construction of the LEXSUB-PARA dataset. The original Lexical Substitution corpus, constructed for the SemEval-1 lexical substitution task (McCarthy and Navigli, 2007), consists of 10 instances each of 200 target words in sentential contexts, drawn from a large internet corpus (Sharoff, 2006). Contextually appropriate paraphrases for each instance of each target word were elicited from up to 6 participants. Figure 3 shows two instances for the verb *to work*. The frequency distribution over paraphrases can be understood as a characterization of the target word's meaning in each context.

For the current paper, we constructed a new subset of LexSub we call LEXSUB-PARA by parsing LexSub with Minipar (Lin, 1993) and extracting all 177 sentences with transitive verbs that had overtly realized subjects and objects, regardless of voice. We did not manually verify the correctness of the parses, but discarded 17 sentences where we were not able to compute inverse selectional preferences for the subject or object head word (these were mostly rare proper names). This left 160 transitive instances of 42 verbs.

Evaluation For evaluation, we use a variant of the SemEval "out of ten" (OOT) evaluation metrics defined by McCarthy and Navigli (2007). They developed two metrics, OOT Precision and Recall, which compare where a predicted set of appropriate paraphrases must be evaluated against a gold standard set. Their metrics are called "out of ten" because they are measure the accuracy of the first ten paraphrases predicted by the system. Since they allow systems to abstain from predictions for any number of tokens, their two variants average this accuracy (a), over the tokens with a prediction (OOT Precision), and (b), over all tokens (OOT Recall). Since our system produces predictions for all tokens, OOT Precision and Recall become identical.

Formally, let G_i be the gold paraphrases for occurrence i, and let $f(s,i)$ be the frequency with which s has been named as paraphrase for i. Let M_i be the ten paraphrase candidates top-ranked by the SVS model for i. We write out-of-ten accuracy (OOT) as:

$$\text{OOT} = 1/|I| \sum_i \frac{\sum_{s \in M_i \cap G_i} f(s,i)}{\sum_{s \in G_i} f(s,i)} \qquad (7)$$

We compute two baselines. The first one is random baseline that guesses whether paraphrases are appropriate. The second baseline uses the original type vector of the target verb without any combination, i.e., its "out of context meaning", as representation for the token.

Results. Table 1 shows the results on the LEXSUB-PARA dataset. Recall that the task is to decide the appropriateness of paraphrases for verb instances, disambiguated by the inverse selectional preferences of their subjects (*subj*), their objects (*obj*), and *both*. The random baseline attains an OOT accuracy of 53.7, and the type vector of the target vector performs at 57.1.

SVS is able to outperform both baselines for all values of the reweighting parameter n <2, and we find the best results for the lowest value, $n = 0.5$. As for the influence of the vector combination function, the best result is yielded by **min** (OOT=62.3), followed by **add** (OOT=61.7), while **mult** shows generally worse results (OOT=60.3). For both **add** and **mult**, using only the subject as context only is optimal. The overall best result, using **min**, is seen for *both*; however, the improvement over *subj* is very small.

In the model **mult**-*both*-20, where target vectors were multiplied with two very large expectation vectors, almost all instances failed due to overflow errors.

Discussion. Our results indicate that our parameter optimization strategy in Erk and Padó (2008) was in fact flawed. The parameters that were best for the Mitchell and Lapata (2008) data (**mult**, $n = 20$) are suboptimal for LEXSUB-PARA data.[5] The good results for low val-

[5]We assume that our results hold for the Padó & Erk (2008) lexical substitution dataset as well, due to its similar nature.

ues of n indicate that good discrimination between valid and invalid paraphrases can be obtained by relatively small modifications of the target vector in the direction indicated by the context. Surprisingly, we still find that the results in the *subj* condition are almost always better than those in the *obj* condition, even though the dataset consists only of transitive verbs, where we would have expected the inverse result. We have two partial explanations. First, we find that pronouns, which occur frequently in subject position (*I*, *he*), are still informative enough to distinguish "animate" from "inanimate" paraphrases of verbs such as *touch*. Second, we see a higher number of Minipar errors in for object positions than for subject positions, and consequently more data both for object fillers and for object selectional preferences.

The overall best result was yielded by a condition that used *both* (subject plus object) for disambiguation, using the recursive modification from Eq. (4). While we see this as a promising result, the difference to the second-best result is very small, in almost all other conditions the performance of *both* is close to the average of *obj* and *subj* and thus a suboptimal choice.

4.2 Experiment 2: Creating larger datasets with pseudo-paraphrases

With a size of 2,000 sentences, even the complete LexSub dataset is tiny in comparison to many other resources in NLP. Limiting attention to successfully parsed transitive instances results in an even smaller dataset on which it is difficult to distinguish noise from genuine differences between models. This is a large problem for the use of paraphrase appropriateness as evaluation task for models of word meaning in context.

In consequence, the automatic creation of larger datasets is an important task. While unsupervised methods for paraphrase induction are becoming available (e.g., Callison-Burch (2008)), they are still so noisy that the created datasets cannot serve as gold standards. However, there is an alternative strategy: there is a considerable amount of data in different languages annotated with *categorical* word sense, created (e.g.) for Word Sense Disambiguation exercises such as Senseval. We suggest to convert these data for use in a task similar to paraphrase assessment, interpreting available information about the word sense as *pseudo-paraphrases*. Of course, the caveat is that these pseudo-paraphrases may behave differently than genuine paraphrases. To investigate this issue, we repeat Experiment 1 on this dataset.

Construction of the SEMCOR-PARA dataset The SemCor corpus is a subset of the Brown corpus that contains 23,346 lemmas annotated with senses according to WordNet 1.6. Fortunately, WordNet provides a rich characterization of word senses. This allows us to use the WordNet *synonyms* of a given word sense as pseudo-paraphrases. Since it can be the case that the target word is the only word in a synset, we also need to add *direct hypernyms*. Direct hypernyms have been used in annotation tasks to characterize WordNet senses (Mihalcea and Chklovski, 2003), an indicator that they are usually close enough in meaning to function as pseudo-paraphrases.

		0.5	1	2	5	10	20
add	obj	21.7	20.7	23.2	**24.3**	24.2	21.8
add	subj	20.6	20.1	22.9	**24.4**	23.3	19.7
add	both	21.1	20.3	23.2	**24.4**	23.3	18.9
mult	obj	22.6	24.8	**25.0**	24.4	24.2	21.4
mult	subj	21.1	23.9	**24.4**	**24.4**	23.5	19.8
mult	both	24.5	24.5	**25.6**	24.3	20.0	17.4
min	obj	20.9	19.5	23.6	**24.4**	24.3	21.9
min	subj	20.1	19.6	22.5	**24.2**	23.9	19.6
min	both	20.1	19.8	**25.2**	24.5	24.3	19.0

Table 2: OOT accuracy on the SEMCOR-PARA dataset across models and reweighting values (best results for each line boldfaced). Random baseline: 19.6. Target type vector baseline: 20.8

Again, we parsed the corpus with Minipar and identified all sense-tagged instances of the verbs from LEXSUB-PARA, to keep the two corpora as comparable as possible. For each instance w_i of word w, we collected all synonyms and direct hypernyms of the synset as the set of appropriate paraphrases. The list of synonyms and direct hypernyms of all other senses of w, whether they occur in SemCor or not, were considered inappropriate paraphrases for the instance w_i. This method does not provide us with frequencies for the pseudo-paraphrases; we thus assumed a uniform frequency of 1. This does not do away with the gradedness of the meaning representation, though, since each token is still associated with a set of appropriate paraphrases.

Out of 2242 transitive verb instances, we further removed 153 since we could not compute selectional preferences for at least one of the fillers. 484 instances were removed because WordNet did not list any verbal paraphrases for the annotated synset or its direct hypernym. This resulted in 1605 instances for 40 verbs, a dataset an order of magnitude larger than LEXSUB-PARA. (See Section 4.3 for an example verb with paraphrases.)

Results and Discussion. We again use the OOT accuracy measure. The results for paraphrase assessment on SEMCOR-PARA are shown in Table 2. The numbers are substantially lower than for LEXSUB-PARA. This is first and foremost a consequence of the higher "polysemy" of the pseudo-paraphrases. In LEXSUB-PARA, the average numbers of possible paraphrases per target word is 20; in SEMCOR-PARA, 54. This is to be expected and also reflected in the much lower random baseline (19.6% OOT). However, we also observe that the reduction in error rate over the baseline is considerably lower for SEMCOR-PARA than for LEXSUB-PARA (10% vs. 20% reduction).

Among the parameters of the model, we find the largest impact for the reweighting parameter. The best results occur in the middle range($n = 2$ and $n = 5$),

with both lower and higher weights yielding considerably lower scores. Apparently, it is more difficult to strike the right balance between the target and the expectations on this dataset. This is also mirrored in the smaller improvement of the target type vector baseline over the random baseline. As for vector combination functions, we find the best results for the more "intersection"-like **mult** and **min** combinations, with somewhat lower results for **add**; however, the differences are rather small. Finally, combination with *obj* works better than combination with *subj*. At least among the best results, *both* is able to improve over the use of either individual relation. The best result uses **mult**-*both*, with an OOT accuracy of 25.6.

4.3 Further analysis

In our two experiments, we have found systematic relationships between the SVS model parameters and their performance within the LEXSUB-PARA and SEMCOR-PARA datasets. Unfortunately, few of the parameter settings we found to work well appear to generalize across the two datasets; neither do they correspond to the optimal parameter values we established for the Mitchell and Lapata dataset in our 2008 paper. Variables that vary particularly strikingly are the reweighting parameter and the performance of different relations. To better understand these differences, we perform a further validation analysis that attempts to link model performance to a variable that (a) behaves consistently across the two datasets used in this paper and (b) sheds light onto the patterns we have observed for the parameters.

The quantity we will use for this purpose is the average *discriminativity* of the model. We define discriminativity as the degree to which the token vector computed by the model is on average more similar to the valid than to the invalid paraphrases. For a paraphrase ordering task such as the one we are considering, we want this quantity to be as large as possible; very small quantities indicate that the model is basically "guessing" an order.

Figure 4 plots disciminativity against model performance. As can be expected, it is indeed a very strong correlation between discriminativity and OOT accuracy across all models. A Pearson's correlation test confirms that the correlation is highly significant for both datasets (LEXSUB-PARA: r=0.65, $p < 0.0001$; SEMCOR-PARA: r=0.76, $p < 0.0001$).

Next, we considered the relationship between the mean discriminativity for different combinations and reweighting values n. Figure 5 shows the resulting plots, which reveal two main differences between the datasets. The first one is the influence of the reweighting parameter. For LEXSUB-PARA, the highest discriminativity is found for small values of n, with decreasing values for higher parameter values. In contrast, SEMCOR-PARA shows the highest discriminativity for middle values of n (on the order of 5–10), with lowest values on either side. The second difference is the relative discriminativity of *obj* and *subj*. On LEXSUB-PARA, the *subj* predictions are more discriminative than *obj* predictions for all values of n. On SEMCOR-PARA, this picture is reversed, with more discriminative *obj* predictions for the best (and thus relevant) values of n.

We interpret these patterns, which fit the observed OOT accuracy numbers well, as additional evidence that the variations we see between the datasets are not noise or artifacts of the setup, but arise due to the different makeup of the two datasets. This ties in with our intuitions about the differences between human-generated paraphrases and WordNet "pseudo-paraphrases". Compare the following paraphrase lists:

dismiss (LexSub): banish, deride, discard, discharge, dispatch, excuse, fire, ignore, reject, release, remove, sack

dismiss (SemCor/WordNet): alter, axe, brush, can, change, discount, displace, disregard, dissolve, drop, farewell, fire, force, ignore, modify, notice, packing, push, reject, remove, sack, send, terminate, throw, usher

The SEMCOR-PARA list contains a larger number of unspecific pseudo-paraphrases such as *change, push, send*, which stem from direct WordNet hypernyms of the more specific *dismiss* senses. Presumably, these terms are assigned rather general vectors which the SVS finds difficult to rule out as paraphrases. This lowers the discriminativity of the models, in particular for *subj*, and results in the smaller relative improvement over the baseline we observe for SEMCOR-PARA. This suggests that the usability of word sense-derived datasets in evaluations could be improved by taking depth in the WordNet hierarchy into account when including direct hypernyms among the pseudo-paraphrases.

5 Conclusions

In this paper, we have explored the parameter space for the computation of vector-based representations of token meaning with the SVS model.

Our evaluation scenario was paraphrase assessment. To systematically assess the impact of parameter choice, we created two new controlled datasets. The first one, the LEXSUB-PARA dataset, is a small subset of the Lexical Substitution corpus (McCarthy and Navigli, 2007) that was specifically created for this task. The second dataset, SEMCOR-PARA, which is considerably larger, consists in instances from the SemCor corpus whose WordNet annotation was automatically converted into "pseudo-paraphrase" annotation.[6]

We found a small number of regularities that hold for both datasets: namely, that the reweighting parameter is the most important choice for a SVS model, followed by the relation used as context, while the influence of the vector combination function is comparatively small. Unfortunately, the actual settings of these parameters appeared not to generalize well from one dataset to the other. We have collected evidence that these divergences are not due to noise, but to genuine differences

[6] Both datasets can be obtained from the authors.

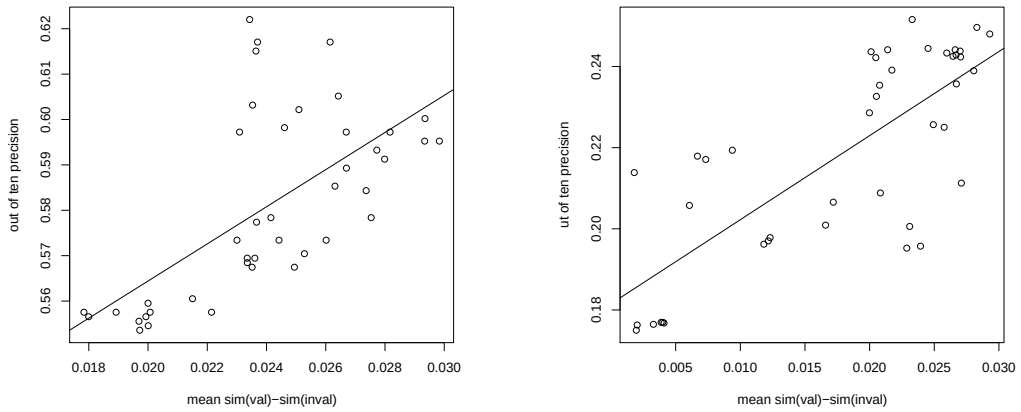

Figure 4: Scatterplot of "out of ten" accuracy against model discriminativity between valid and invalid paraphrases. Left: LEXSUB-PARA, right: SEMCOR-PARA.

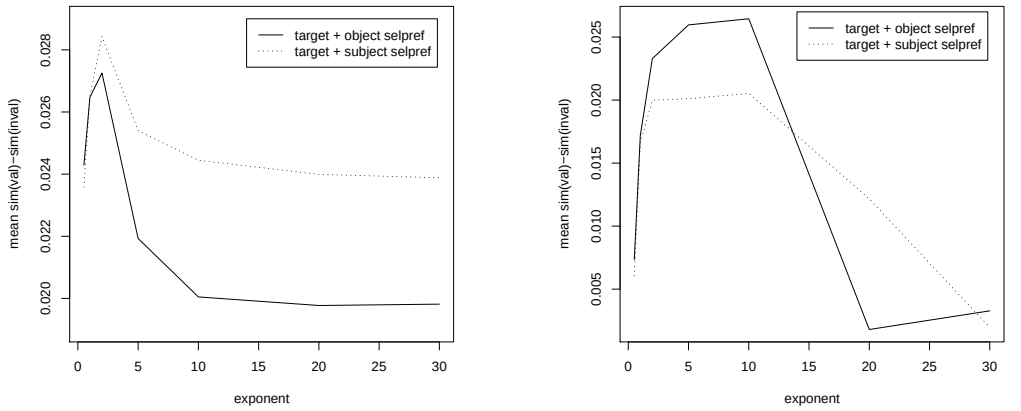

Figure 5: Average amount to which predictions are more similar to valid than to invalid paraphrases, for different reweighting values. Left: LEXSUB-PARA, right: SEMCOR-PARA.

in the datasets. We describe an auxiliary quantity, discriminativity, that measures the ability of the model's predictions to distinguish between valid and invalid paraphrases.

The consequence we draw from this study is that it is surprisingly difficult to establish generalizable "best practice" parameter setting for SVS. Good parameter values appear to be sensitive to the properties of datasets. For example, we have attributed the observation that subjects are more informative on LEXSUB-PARA, while objects work better on SEMCOR-PARA, to differences in the set of paraphrase competitors. In this regard, the conversion of the WSD corpus can be considered a partial success. We have constructed the largest existing paraphrase assessment corpus. However, the use of WordNet information to create paraphrases results in a very difficult corpus. We will investigate methods that exclude overly general hypernyms of the target words as paraphrases to alleviate the problems we see currently.

Discriminativity further suggests that paraphrase assessment can be improved by selectional preference representations that are trained to maximize the distance between valid and invalid paraphrases. Such a representation could be provided by discriminative formulations (Bergsma et al., 2008), or by exemplar-based models that are able to deal better with the ambiguity present in the preferences of very general words.

Another important topic for further research is the computation of token vectors that incorporate more than one context word. The current results we obtain for "*both*" are promising but limited; it appears that the successful integration of multiple context words requires strategies that go beyond simplistic addition or intersection of observed contexts.

References

S. Bergsma, D. Lin, and R. Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *Proceedings of EMNLP*, pages 59–68.

C. Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of EMNLP*, pages 196–205.

K. Erk and S. Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of EMNLP*.

K. Erk. 2007. A simple, similarity-based model for selectional preferences. In *Proceedings of ACL*, pages 216–223.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

P. Hanks. 2000. Do word meanings exist? *Computers and the Humanities*, 34(1-2):205–215.

D. Hindle and M. Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120.

A. Kilgarriff and J. Rosenzweig. 2000. Framework and results for English Senseval. *Computers and the Humanities*, 34(1-2).

A. Kilgarriff. 1997. I don't believe in word senses. *Computers and the Humanities*, 31(2):91–113.

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

T. Landauer and S. Dumais. 1997. A solution to Platos problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.

B. Levin and M. Rappaport Hovav. 2005. *Argument Realization*. Research Surveys in Linguistics Series. CUP.

D. Lin. 1993. Principle-based parsing without overgeneration. In *Proceedings of ACL*, pages 112–120.

K. Lund and C. Burgess. 1996. Producing high-dimensional semantic spaces from lexical cooccurrence. *Behavior Research Methods, Instruments, and Computers*, 28.

D. McCarthy and R. Navigli. 2007. SemEval-2007 Task 10: English Lexical Substitution Task. In *Proceedings of SemEval*, pages 48–53.

D. McCarthy. 2006. Relating WordNet senses for word sense disambiguation. In *Proceedings of the ACL Workshop on Making Sense of Sense*, pages 17–24.

S. McDonald and C. Brew. 2004. A distributional model of semantic context effects in lexical processing. In *Proceedings of ACL*, pages 17–24.

K. McRae, M. Spivey-Knowlton, and M. Tanenhaus. 1998. Modeling the influence of thematic fit (and other constraints) in on-line sentence comprehension. *Journal of Memory and Language*, 38:283–312.

R. Mihalcea and T. Chklovski. 2003. Open Mind Word Expert: Creating large annotated data collections with web users' help. In *Proceedings of the EACL 2003 Workshop on Linguistically Annotated Corpora (LINC 2003)*, Budapest, Hungary.

J. Mitchell and M. Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL*, pages 236–244.

S. Narayanan and D. Jurafsky. 2002. A Bayesian model predicts human parse preference and reading time in sentence processing. In *Proceedings of NIPS*, pages 59–65.

S. Padó, U. Padó, and K. Erk. 2007. Flexible, corpus-based modelling of human plausibility judgements. In *Proceedings of EMNLP/CoNLL*, pages 400–409.

M. Palmer, H. Dang, and C. Fellbaum. 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Journal of Natural Language Engineering*. To appear.

P. Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, 61:127–159.

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

Serge Sharoff. 2006. Open-source corpora: Using the net to fish for linguistic data. *International Journal of Corpus Linguistics*, 11(4):435–462.

Y. Wilks. 1975. Preference semantics. In *Formal Semantics of Natural Language*. CUP.