Representing words as regions in vector space

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

Abstract

Vector space models of word meaning typically represent the meaning of a word as a vector computed by summing over all its corpus occurrences. Words close to this point in space can be assumed to be similar to it in meaning. But how far around this point does the region of similar meaning extend? In this paper we discuss two models that represent word meaning as *regions* in vector space. Both representations can be computed from traditional point representations in vector space. We find that both models perform at over 95% F-score on a token classification task.

1 Introduction

Vector space models of word meaning (Lund and Burgess, 1996; Landauer and Dumais, 1997; Lowe, 2001; Jones and Mewhort, 2007; Sahlgren and Karlgren, 2005) represent words as points in a high-dimensional semantic space. The dimensions of the space represent the contexts in which each target word has been observed. Distance between vectors in semantic space predicts the degree of semantic similarity between the corresponding words, as words with similar meaning tend to occur in similar contexts. Because of this property, vector space models have been used successfully both in computational linguistics (Manning et al., 2008; Snow et al., 2006; Gorman and Curran, 2006; Schütze, 1998) and in cognitive science (Landauer and Dumais, 1997; Lowe and McDonald, 2000; McDonald and Ramscar, 2001). Given the known problems with defining globally appropriate senses (Kilgarriff, 1997; Hanks, 2000), vector space models are especially interesting for their ability to represent word meaning without relying on dictionary senses.

Vector space models typically compute one vector per target word (what we will call *word type* vectors), summing co-occurrence counts over all corpus tokens of the target. If the target word is polysemous, the representation will constitute a union over the uses or senses of the word. Such a model does not provide information on the amount of *variance in each dimension*: Do values on each dimension vary a lot across occurrences of the target? Also, it does not provide information on *co-occurrences of feature values* in occurrences of the target. To encode these two types of information, we study richer models of word meaning in vector space beyond single point representations.

Many models of categorization in psychology represent a concept as a region, characterized by feature vectors with dimension weights (Smith et al., 1988; Hampton, 1991; Nosofsky, 1986). Taking our cue from these approaches, we study two models that represent a word as a *region* in vector space rather than a point. The first model is one that we have recently introduced for representing hyponymy in vector space (Erk, 2009). We now test its suitability as a general region model for word meaning. This model can be viewed as a prototype-style model that induces a region surrounding a central vector. As it does not record co-occurrences of feature values, we contrast it with a second model, an exemplar-style model using a k-nearest neighbor analysis, which can represent both degree of variance in each dimension and value co-occurrences.

Both models induce regions representations without labeled data. The idea on which both models are based is to use *word token* vectors to estimate a

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL), pages 57–65,
Boulder, Colorado, June 2009. ©2009 Association for Computational Linguistics

region representation. We evaluate the two region models on a task of token classification: Given a point in vector space, the task is predict the word of which it is a token vector.

By representing the meaning of words as regions in vector space, we can describe areas in which points encode similar meanings. This description is flexible, depending on the target word in question, rather than uniform for all words through a fixed distance threshold from the target's type vector. One possible application of region models of word meaning is in the task of determining the appropriateness of a paraphrase in a given context (Connor and Roth, 2007). This task is highly relevant for textual entailment (Szpektor et al., 2008). Current vector space approaches typically compare the target word's token vector to the type vector of the potential paraphrase (Mitchell and Lapata, 2008; Erk and Pado, 2008). A region model could instead test the target's token vector for inclusion in the potential paraphrase's region.

2 Related work

This section discusses existing vector space models and compares vector space models in computational linguistics to feature-based models of human concept representation in psychology.

Vector space models. Vector space models represent the meaning of a target word as a vector in a high-dimensional space (Lund and Burgess, 1996; Landauer and Dumais, 1997; Sahlgren and Karlgren, 2005; Padó and Lapata, 2007; Jones and Mewhort, 2007). Dimensions stand for context items which which the target word has been observed to co-occur, for example other words (Lund and Burgess, 1996) or syntactic paths (Padó and Lapata, 2007). In the simplest case, the value on a dimension is the raw co-occurrence count between the target word and the context item for which the dimension stands. Raw counts are often transformed, for example using a log-likelihood transformation (Lowe, 2001). Sometimes the vector space as a whole is transformed using dimensionality reduction (Landauer and Dumais, 1997).

In NLP, vector space models have featured most prominently in information retrieval (Manning et al., 2008), but have also been used for ontology learning (Lin, 1998; Snow et al., 2006; Gorman and Curran, 2006) and word sense-related tasks (McCarthy et al., 2004; Schütze, 1998). In psychology, vector space models have been used to model synonymy (Landauer and Dumais, 1997; Padó and Lapata, 2007), lexical priming phenomena (Lowe and McDonald, 2000), and similarity judgments (McDonald and Ramscar, 2001). There have also been studies on inducing hyponymy information from vector space representations. Geffet and Dagan (2005) use a dimension re-weighting scheme, then predict entailment when the most highly weighted dimensions of two verbs stand in a subset relation. However, they find that while recall of this method is good (whenever some senses of two words stand in an entailment relation, top-weighted dimensions of their vectors stand in a subset relation), precision is problematic. Weeds, Weir and McCarthy (2004) introduce the notion of distributional generality (x is more distributionally general than y if x occurs in more contexts than y) and find that for hyponym-hypernym pairs from WordNet, hyponyms are typically more distributionally general. (As they study only word pairs that are known to be related by hyponymy, they test for recall but not precision.) Erk (2009) suggests that while it may not be possible to *induce* hyponymy information from a vector space representation, it is possible to *encode* it in a vector space representation after it has been obtained through some other means.

Vector space models of word tokens. Vector space models have mostly been used to represent the meaning of a word *type* by summing its co-occurrence counts over a complete corpus. There are several approaches to computing vectors for individual word *tokens*. All of them compute word type vectors first, then combine them into token vectors. Kintsch (2001) and Mitchell and Lapata (2008) combine the target's type vector with that of a single word in the target's syntactic context. Landauer and Dumais (Landauer and Dumais, 1997) and Schütze (1998) combine the type vectors of all the words surrounding the target token. Erk and Padó (2008) combine the target's type vector with a vector representing the selectional preference of a single word in the target's syntactic context. Smolensky (1990) focuses on integrating syntactic information in the vector representation rather than on representing the lexical meaning of the target.

Feature-based models of human concept representation. Many models of human concept representation in psychology are based on vectors of features (e.g. (Smith et al., 1988; Hampton, 1991; Nosofsky, 1986)). Features in these models are typically weighted to represent their importance to the concept in question. Similarity to a given feature vector is usually taken to decrease *exponentially* with distance from that vector, following Shepard's law (Shepard, 1987). Categorization involves competition between categories. Feature-based models of human concept representation can be broadly categorized into *prototype models*, which represent a concept by a single summary representation, and *exemplar models*, which assume that categorization is by comparison to remembered exemplars. As an example of a feature-based model of concept representation, we show the definition of Nosofsky's (1986) Generalized Context Model (GCM). This exemplar model estimates the probability of categorizing an exemplar $\vec{e}$ as a member of a concept C as

$$P(C|\vec{e}) = \frac{\sum_{\vec{\eta} \in C} w_{\vec{\eta}} sim(\vec{\eta}, \vec{e})}{\sum_{\text{concept } C'} \sum_{\vec{\eta} \in C'} w_{\vec{\eta}} sim(\vec{\eta}, \vec{e})} \quad (1)$$

where the concept C is a set of remembered exemplars, $w_{\vec{\eta}}$ is an exemplar weight, and the similarity $sim(\vec{\eta}, \vec{e})$ between $\vec{\eta}$ and $\vec{e}$ is defined as

$$sim(\vec{\eta}, \vec{e}) = exp(z \cdot \sum_{\text{dimension } i} w_i(\eta_i - e_i)^2) \quad (2)$$

Here, z is a general sensitivity parameter, w_i is a weight for dimension i, and η_i, e_i are the values of $\vec{\eta}$ and $\vec{e}$ on dimension i. This model shows all the properties listed above: It has weighted dimensions through the w_i. It incorporates Shepard's law through the exponential relation between sim and the sum of squared value distances $w_i(\eta_i - e_i)^2$. Competition between categories arises through the normalization of $\vec{e}$'s similarity to C by the similarity to all other categories in Eq. (1). While feature-based models of concept representation talk about concepts rather than word meaning, Murphy (2002) argues that there is "overwhelming empirical evidence for the conceptual basis of word meaning" through experimental results on conceptual phenomena that have also been shown to hold for words.

Gärdenfors (2004) proposes a model that represents concepts as convex regions in a *conceptual space*. Feature structures play no central role in this model, but Gärdenfors suggests that concepts may be represented by a central point, such that categorization could simply be determining the nearest central point (without positing an exponential relation between distance and similarity).

3 Models

In this section, we present two models for representing word meaning as regions in vector space.

The centered model. The first model that we define, which we call the *centered model*, is prototype-like. As the representation for a target word, it induces a region surrounding the target's type vector (Erk, 2009). Let w be the target word and $\vec{w}$ its type vector. Let $\vec{x}$ be a point in the same vector space. To predict whether $\vec{x}$ represents the same meaning as $\vec{w}$, we estimate the probability $P(\text{IN}(\vec{x}, \vec{w}))$ that $\vec{x}$ is in the region around $\vec{w}$, using a log-linear model:

$$P(\text{IN}(\vec{x}, \vec{w})) = \frac{1}{Z} \exp(\sum_i \beta_i^{\text{IN}} f_i(\vec{x}, \vec{w})) \quad (3)$$

where the f_i are features that characterize the point $\vec{x}$, and the β_i^{IN} are weights identifying the importance of the different features for the class IN. Z is a normalizing factor that ensures that P is a probability distribution: If $P(\text{OUT}(\vec{x}, \vec{w})) = 1 - P(\text{IN}(\vec{x}, \vec{w}))$ is the probability that $\vec{x}$ is not in the region around $\vec{w}$, with associated weights β_i^{OUT} for the same features f_i, then $Z = \sum_{\ell=\text{IN,OUT}} exp(\sum_i \beta_i^\ell f_i(\vec{x}, \vec{w}))$.

We define the features f_i as follows: If $\vec{w} = \langle w_1, \ldots, w_n \rangle$, we define the feature $f_i(\vec{x}, \vec{w})$, for $1 \leq i \leq n$, as the squared distance between $\vec{w}$ and $\vec{x}$ on dimension i:

$$f_i(\vec{x}) = (w_i - x_i)^2 \quad (4)$$

This model, like feature-based models of categorization from psychology, has weighted dimensions through the β_i. It follows Shepard's law – the exponential relation between similarity and distance – through the exponential function in Eq. (3). Competition between categories is implicit in the estimation of $P(\text{OUT}(\vec{x}, \vec{w}))$.

Most of the weights β_i^{IN} can reasonably be expected to be negative, since a negative β_i^{IN} indicates that membership of a point $\vec{x}$ in the w-region gets less likely as the distance $(w_i - x_i)^2$ increases. If β_i^{IN} has a large negative value, categorization is highly sensitive to changes in the ith dimension. If on the other hand, β_i^{IN} is negative but close to zero, this means that vector entries in dimension i can vary greatly without much influence on categorization.

The parameters β_i^{IN} and β_i^{OUT} need to be estimated from training data. Although the log-likelihood model is a supervised learning scheme, we do not need to take recourse to labeled data. Instead, we use token vectors: Token vectors of w will serve as positive training data for estimating $P(\text{IN}(\vec{x}, \vec{w}))$, and token vectors of other words than w will constitute negative training data. The amount of preprocessing needed depends on the approach to computing token vectors that we use. We will use an approach that combines w's type vector with that of a single word in its syntactic context. This presupposes a syntactic parse of the corpus. Note that we could just as well have used a Schütze-style approach, which does not rely on parsing.

The distributed model. The second model that we consider is an exemplar-style, instance-based model. The simplest instance-based models are k-nearest neighbor classifiers, which assign to a test item the majority label of its k nearest neighbors among the training items. We will here use a very simple model, doing k nearest neighbor classification where the distance between two vectors $\vec{w}$ and $\vec{x}$ is the sum of dimension distances δ_i with

$$\delta_i = \frac{\beta_i |w_i - x_i|}{\max_i - \min_i}$$

$\max_i$ and $\min_i$ are the maximum and minimum values observed for dimension i, and β_i is a feature weight. We use a standard feature weighting method, *gain ratio*, which is information gain normalized by the entropy of feature values. Information gain on its own has a bias towards features with many values, which gain ratio attenuates in favor of features with lower entropy:

$$\beta_i = \frac{H(C) - \sum_{y \in val(i)} P(y) H(C|y)}{- \sum_{y \in val(i)} P(y) \log_2 P(y)} \quad (5)$$

for the set $C = \{\text{IN}, \text{OUT}\}$ of classes and sets $val(i)$ of values seen for dimension i. We call this the *distributed model*. As with the centered model, we compare it to models of concept representation: It has weighted dimensions (Eq. (5)), and it incorporates competition between categories by storing both positive and negative exemplars and categorizing according to the majority among the k nearest neighbors. However, it does not implement Shepard's law. It additionally differs from the GCM (Eq. (1)) in basing categorization on the k nearest neighbors rather than summed similarity to all neighbors.

Like the centered model, the distributed model needs both positive and negative training data. Again, labeled data is not necessary as we can use word token vectors. Positive training data consists of tokens of the target word, and tokens of other words are negative training data. This model does not make use of the target's type vector.

Above we have discussed two pieces of information that region models can encode and that are hard to encode in single-point models of word meaning: *variance in each dimension* and *co-occurrence of feature values*. The centered model encodes the variance in the values of each dimension through the weights β_i^{IN}, but it does not retain information on feature values of different dimensions that tend to co-occur. The distributed model encodes both variance in each dimension and co-occurrence of feature values through the remembered exemplars. So the centered model should do well for monosemous words, since it seems reasonable that their token vectors should form a single region around the type vector. For polysemous words, token vectors could be more scattered in semantic space, in which case the distributed model should do better.

Note that neither the centered nor the distributed model is a clustering model: Both are supervised models learning the distinctions between tokens of the target word and other vectors. Neither of them groups vectors in an unsupervised fashion.

Hard versus soft region boundaries. In the current paper, we consider only regions with sharp boundaries. In the centered model, a point $\vec{x}$ will be considered a member of the w-region if $P(\text{IN}(\vec{x}, \vec{w})) \geq 0.5$. In the distributed model, $\vec{x}$ will be considered a member if the majority of its k nearest neighbors are members. However, it is important that both models can also be used to represent regions with soft boundaries. In the centered model, we can use $P(\text{IN}(\vec{x}, \vec{w}))$ without a threshold. In the distributed model, we can use the fraction of k that are positive instances, or we can compute summed similarity to the positive instances like the GCM does. So both models can be used to estimate degrees of membership in a target word's region.

4 Task, Data, and Implementation

This section describes the task used for evaluation, the data, and the implementation of the models.

Task. The main task will be for a model trained on a target word w to predict, for a given point $\vec{x}$ in semantic space, whether $\vec{x}$ is a token vector of w or not. This task is a direct test of whether the region induced for w succeeds in characterizing the region in semantic space in which tokens of w will occur.

As an example, consider the target word *supersede*: Region models of *supersede* will be trained on tokens of *supersede* in a training dataset. One such token is *supersede knowledge* (i.e., *knowledge* as the direct object of *supersede*). We compute a token vector for this occurrence by combining the type vectors of *supersede* and *knowledge*. After training a model, we test it on tokens occurring in a test dataset. Positive test items are tokens of *supersede*, and negative test items are tokens of other words, for example *guard*. An example of a positive test item is *supersede collection*. The test items will consist solely of tokens that do not occur in the training data.

Data. We focus on verbs in this paper since paraphrase appropriateness for verbs is an important task in the context of textual entailment. Since we suspect that the centered model will be better suited to modeling monosemous words while the distributed model should do equally well on monosemous and polysemous words, we first test a group of monosemous verbs, then a mixed group. We use WordNet 3.0 to form the two groups. The first group consists of all verbs listed in WordNet 3.0 as being monosemous. We refer to this set as ***Mon***. Since we also want to compare the two region models on the task of hyponymy encoding (Erk, 2009), we use as our set of mixed monosemous and polysemous verbs the verbs used there to test hyponymy encoding: the set of all verbs that are hypernyms of the *Mon* verbs according to WordNet 3.0. We call this set ***Hyp***.

We use the British National Corpus (BNC) to compute the vector space and as our source of target word tokens. We need token vectors for training the two region models, and we need separate, previously unseen token vectors as test data. So we split the written portion of the BNC in half at random, leaving files intact. This yielded a training and a test set. We computed word type vectors from the training half of the BNC, using a syntax-based vector space (Padó and Lapata, 2007) of 500 dimensions, with raw co-occurrence counts as dimension values. We used the `dv` package[1] to compute type vectors from a Minipar (Lin, 1993) parse of the BNC.

We computed token vectors by combining the target verb's type vector with the type vector of the word occurring as the target's direct object. We test three methods for combining type vectors: First, component-wise multiplication (below called ***mult***), which showed best results in Mitchell and Lapata's (2008) analysis. Second, component-wise averaging (below called ***avg***), a variant of type vector addition, a method often used for computing token vectors. Third, we consider component-wise minimum (***min***), which can be viewed as a kind of intersection of the contexts with which the two words have been observed. We used the training half of the BNC to extract training tokens of the target verbs, and the test half for extracting test tokens. We used only those verb/object pairs as test tokens that did not also occur in the training data.

We restricted the set of verbs to avoid data sparseness issues, using only verbs that occurred with at least 50 different direct objects in the training part of the BNC. The direct objects, in turn, were restricted to exclude overly rare and overly frequent (and thus potentially uninformative) items. We restricted the direct objects to those with no more than 6,500 and no less than 270 occurrences in $Mon \cup Hyp$. The resulting set *Mon* consisted of 120 verbs, and *Hyp* consisted of 430 verbs.

Model implementation. We implemented the centered model using the OpenNLP maxent package[2], and the distributed model using TiMBL[3] in the IB1 setting with $k = 5$ nearest neighbors. We use bi-

[1] `http://www.nlpado.de/~sebastian/dv.html`

[2] `http://maxent.sourceforge.net/`

[3] `http://ilk.uvt.nl/timbl/`

	centered			distributed		
	Prec	Rec	F	Prec	Rec	F
mult	**100**	73.2	84.5	29.4	47.5	36.3
avg	99.6	91.3	**95.3**	71.1	**99.9**	83.1
min	97.9	85.4	91.2	21.0	90.3	34.1

Table 1: Results: token classification for monosemous verbs. Random baseline: Prec 0.8, Rec 49.8, F 1.6.

nary models throughout, such that the classification task is always between IN and OUT. In training and testing, each token vector was presented to a model only once, ignoring the frequency of direct objects.

5 Experiments

This section reports on experiments that test the performance of the two region models of word meaning in vector space that we have presented in Sec. 3, the centered and the distributed model.

Experiment 1: Token classification for monosemous verbs

In the first experiment, we test whether the two region models can identify novel tokens of the monosemous verbs in *Mon*. The task is the one described in Sec. 4. We focus on monosemous verbs first because we suspect that the centered model should do better here than on polysemous verbs. Both models were trained using token vectors computed from the training half of the BNC. Token vectors of the target verb were treated as positive data, and token vectors of other verbs as negative data.[4] We used resampling to restrict the number of negative items used during training, using 3% of the negative items, randomly sampled.[5] We use for testing only those direct objects that do not also appear in the training data, yielding 6,339 positive and 1,396,552 negative test items summed over all target verbs. The case of *supersede* discussed in Sec. 4 is an example of a monosemous verb according to WordNet 3.0.

Table 1 summarizes precision, recall and F-score results. Both models easily beat the random baseline. The centered model shows better performance overall than the distributed one, and the *avg* method of computing token vector worked best for both models. The centered model has extremely high precision throughout, while the distributed model has better recall for conditions *avg* and *min*. Table 2 breaks down the results by the frequency of the target verb, measured in the number of different verb/object tokens in the training data.

		centered			distributed		
	freq.	Prec	Rec	F	Prec	Rec	F
mult	50-100	100	59.3	74.5	20.8	47.2	28.9
	100-200	100	89.4	94.4	57.4	49.7	53.2
	200-500	100	97.4	98.7	92.1	41.1	56.9
avg	50 - 100	99.5	86.6	92.6	61.6	99.8	76.2
	100-200	99.7	96.6	98.1	86.3	100	92.6
	200-500	100	100	100	99.1	100	99.6
min	50-100	100	82.9	90.6	17.9	92.6	30.1
	100-200	98.2	88.2	93.0	25.4	89.2	39.6
	200-500	86.4	90.3	88.3	42.9	80.0	55.9

Table 2: Results: token classification for monosemous verbs, by target frequency

	centered			distributed		
# senses	Prec	Rec	F	Prec	Rec	F
all	**100**	92.9	96.3	99.6	**99.8**	**99.7**
1	100	86.1	92.5	99.0	99.5	99.2
2-5	100	90.8	95.2	99.4	99.6	99.5
6-10	100	93.5	96.7	99.9	99.9	99.9
11-20	100	96.6	98.3	100	100	100
≥ 21	100	99.5	99.7	100	100	100

Table 3: Results: Token classification for polysemous verbs, *avg* token computation. Random baseline: Prec 8.2, Rec 50.4, F 14.0.

Experiment 2: Token classification for polysemous verbs

We now test how the centered and distributed models fare on the same task, but with a mixture of monosemous and polysemous verbs. We use the verbs in *Hyp*, which in WordNet 3.0 have on average 6.79 senses. For example, *follow* is a WordNet hypernym of the monosemous *supersede*. It has 24 senses, among them *comply* and *postdate*. Among its training tokens are *follow instruction* and *follow dinner*. The first is probably the *comply* sense of *follow*, the second the *postdate* sense. An example of a test token (i.e., occurring in the test but not the train-

[4] This simplification breaks down for 6 of the 120 verbs (5%), which are in fact synonyms. We consider this an acceptable level of noise.

[5] The number of 3% was determined on a development set constructed by further splitting the training set into training and development portion.

ing data) is *follow tea*. (If *tea* is *tea time*, this is also the *postdate* sense.)

We computed type vectors for the *Hyp* verbs and their objects from the training half of the BNC, and computed token vectors using the best method from Exp. 1, *avg*. Again, we use for testing only those tokens that do not also appear in the training data. Due to the larger amount of data, we used resampling in the training as well as the test data, using only a random 3% of negative tokens for testing. This yielded 25,736 positive and 670,630 negative test items.

Table 3 shows the results: The first line has the overall results, and the following lines break down the results by the number of senses each lemma has in WordNet 3.0.[6] Both models, centered and distributed, easily beat the random baseline. The centered model has comparable results for the *Hyp* as for the *Mon* verbs (cf. Table 1), while the distributed model has better results for this dataset, and better results than the centered model. The centered model shows a marked improvement in recall as the number of senses increases.

Experiment 3: Encoding hyponymy

We first proposed the centered model as a method for encoding hyponymy information in a vector space representation (Erk, 2009). Hyponymy information from another source, in this case WordNet, was encoded in a centered region representation of a target verb by using tokens of the verb itself as well as tokens from its direct hyponyms in training the model. Negative data consisted of training data tokens that were not occurrences of the target verb or its direct hyponyms. In the example of the verb *follow*, the positive training data would contain tokens of *follow* along with tokens of *supersede* and *guard*, another direct hyponym of *follow*. Negative training tokens would include, for example, tokens of the word *destroy*. The resulting centered model, in this case of *follow*, was then tested on previously unseen tokens, for example *guard purpose* (a token of a hyponym) and *destroy lawn* (a token of a non-hyponym), with the task of predicting whether they were tokens of direct hyponyms of *follow* or not.

centered			distributed		
Prec	Rec	F	Prec	Rec	F
95.2	43.4	59.6	68.3	**58.6**	**63.1**

Table 4: Results: Identifying hyponyms based on extended hypernym representations, *avg* token computation. Random baseline: Prec 11.0, Rec 50.2, F 18.0

We now repeat this experiment with the distributed model. We use the *direct* hypernyms of the verbs in *Mon*, with the same frequency restrictions as above. We refer to this set of 273 verbs as ***DHyp***. We train one centered and one distributed model for each verb w in *DHyp*. Positive training tokens for training a model for a verb $w \in$ *DHyp* are tokens of w and of all sufficiently frequent children of w in WordNet 3.0. Negative training tokens are tokens of other verbs in *DHyp* and their children. We again sample a random 3% of the negative data during both training and testing.

Table 4 shows the results. Both models again beat the baseline. The distributed model shows slightly better results overall, while the centered model has by far the highest precision.

Discussion

Performance on monosemous verbs. For the monosemous verbs in Exp. 1, both models succeed in inducing regions that characterize tokens of a target word with high precision as well as high recall. The extremely high precision of the centered model shows that in general the region surrounding the type vector does not contain any tokens of other verbs than the target. Concerning the distributed model, it is to be expected that in *min*, and even more so in *mult*, dimension values will vary more than in *avg*; this could explain the huge difference between *avg* and the other two conditions for this model. It is interesting to note that the centered model achieves better precision, while the distributed model reaches higher recall. Maybe it will be possible in later models to combine their strengths. The breakdown by frequency bands in Table 2 shows that in *mult* and *avg*, the models get strictly better with more data, while *min* has a precision/recall tradeoff.

Performance on polysemous verbs. For the polysemous verbs in Exp. 2, like for the monosemous verbs in Exp. 1, both models show excellent per-

[6] The one-sense items in Table 3 are a 43 verb subset of *Mon*. The reason for the difference in performance in comparison to Table 1 is unclear, as the two sets have similar distributions of lemma frequencies.

formance in distinguishing tokens of the target verb from tokens of other verbs.[7] The distributed model surpasses the centered one on this dataset. However, it is not clear that this is because the contiguous region that the centered model infers is inappropriate for polysemous verbs. After all the centered model, too, achieves better performance on this dataset than on *Mon*. The fact that results get better with the degree of polysemy, at first surprising, may indicate that the centered model draws an overly tight boundary around the type vector and that this boundary improves when token vectors differ more, and are at greater distance from the type vector, as should be the case for more polysemous lemmas. Another possible reason for the better performance of both models is that this dataset is larger and in particular provides a larger set of negative data.

Encoding external information in a region model. In the hyponymy encoding task in Exp. 3, both models successfully encode hyponymy information in vector space representations. The centered model manages to derive a high-precision region around the type vector, while the distributed model makes use of outliers in the training data to achieve higher recall.

Comparing region representations to point representations. We now compare the two region models to existing variants of point-based vector space models. Both region models have dimension weights, whose function is somewhat similar to that of log-likelihood or mutual information transformations of raw co-occurrence counts: to estimate the importance of each dimension for characterizing the target word in question. However, dimension weights in region models are computed based on token vectors, while all co-occurrence count transformations work on type vectors.

The distributed model additionally has the ability to represent typical co-occurrences of feature values because the training tokens are remembered in their entirety. The most similar mechanism in point-based vector space models is probably dimensionality reduction, which strives to find latent dimensions that explain most of the variance in the data. But again, dimensionality reduction uses type vectors while the distributed model stores token vectors, which can show more variance than the type vectors alone.

Applications of region models. Region models of word meaning are interesting for the task of testing the appropriateness of paraphrases in context. Previous models either used competition between paraphrase candidates or a global similarity threshold to decide whether to accept a paraphrase candidate (Mitchell and Lapata, 2008; Szpektor et al., 2008). A region model of word meaning used for the same task would still require a threshold, in this case a threshold on membership probability, but the regions for which membership is tested could differ in their size, and the extent of each region would be learned individually from the data. To use the model, for example to test whether *trickle* is a good paraphrase for *run* in *the color ran*, we would test whether the sentence-specific token vector for *run* falls into the region of *trickle*.

6 Conclusion and outlook

In this paper, we have proposed using region models for word meaning in vector space, predicting regions in space in which points can be assumed to carry the same meaning. We have studied two models, the prototype-like *centered* models and the exemplar-like *distributed* model, both of which are learned without labeled data by making use of *token vectors* of the target word in question. Both models show excellent performance, with F-scores of 83%-99%, on the task of identifying previously unseen occurrences of the target word.

Our aim is to to test the usability of region models for predicting paraphrase appropriateness in context. The next step towards that will be to test region models on the task of identifying synonym tokens.

Acknowledgements. Many thanks to Jason Baldridge, David Beaver, Graham Katz, Alexander Koller, Ray Mooney, and Manfred Pinkal for helpful discussions. This work was supported by the Morris Memorial Grant from the New York Community Trust.

References

M. Connor and D. Roth. 2007. Context sensitive paraphrasing with a single unsupervised classifier. In *Proceedings of ECML-07*, Warsaw, Poland.

[7] The near-perfect performance in particular of the distributed model has been confirmed on a separate noun dataset.

K. Erk and S. Pado. 2008. A structured vector space model for word meaning in context. In *Proceedings of EMNLP-08*, Hawaii.

K. Erk. 2009. Supporting inferences in semantic space: representing words as regions. In *Proceedings of IWCS-8*, Tilburg, Netherlands.

P. Gärdenfors. 2004. *Conceptual spaces*. MIT press, Cambridge, MA.

M. Geffet and I. Dagan. 2005. The distributional inclusion hypotheses and lexical entailment. In *Proceedings of ACL-05*, Ann Arbor, MI.

J. Gorman and J. R. Curran. 2006. Scaling distributional similarity to large corpora. In *Proceedings of ACL '06*, Sydney.

J. A. Hampton. 1991. The combination of prototype concepts. In P. Schwanenflugel, editor, *The psychology of word meanings*. Lawrence Erlbaum Associates.

P. Hanks. 2000. Do word meanings exist? *Computers and the Humanities*, 34(1-2):205–215(11).

M. Jones and D. Mewhort. 2007. Representing word menaing and order information in a composite holographic lexicon. *Psychological Review*, 114:1–37.

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.

D. Lin. 1993. Principle-based parsing without overgeneration. In *Proceedings of ACL'93*, Columbus, Ohio.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *COLING-ACL98*, Montreal, Canada.

W. Lowe and S. McDonald. 2000. The direct route: Mediated priming in semantic space. In *Proceedings of the Cognitive Science Society*.

W. Lowe. 2001. Towards a theory of semantic space. In *Proceedings of the Cognitive Science Society*.

K. Lund and C. Burgess. 1996. Producing high-dimensional semantic spaces from lexical cooccurrence. *Behavior Research Methods, Instruments, and Computers*, 28:203—208.

C. D. Manning, P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant senses in untagged text. In *Proceedings of ACL'04*, Barcelona, Spain.

S. McDonald and M. Ramscar. 2001. Testing the distributional hypothesis: The influence of context on judgements of semantic similarity. In *Proceedings of the Cognitive Science Society*.

J. Mitchell and M. Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08*, Columbus, OH.

G. L. Murphy. 2002. *The Big Book of Concepts*. MIT Press.

R. M. Nosofsky. 1986. Attention, similarity, and the identification-categorization relationship. *Journal of Experimental Psychology: General*, 115:39–57.

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

M. Sahlgren and J. Karlgren. 2005. Automatic bilingual lexicon acquisition using random indexing of parallel corpora. *Journal of Natural Language Engineering, Special Issue on Parallel Texts*, 11(3).

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

R. Shepard. 1987. Towards a universal law of generalization for psychological science. *Science*, 237(4820):1317–1323.

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

P. Smolensky. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46:159–216.

R. Snow, D. Jurafsky, and A. Y. Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of COLING/ACL'06*.

I. Szpektor, I. Dagan, R. Bar-Haim, and J. Goldberger. 2008. Contextual preferences. In *Proceedings of ACL-08*, Columbus, OH.

J. Weeds, D. Weir, and D. McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of COLING-04*, Geneva, Switzerland.