Do Supervised Distributional Methods Really Learn Lexical Inference Relations?

Omer Levy[†] **Steffen Remus**[§] **Chris Biemann**[§] **Ido Dagan**[†]

† Natural Language Processing Lab
Bar-Ilan University
Ramat-Gan, Israel
{omerlevy,dagan}@cs.biu.ac.il

§ Language Technology Lab
Technische Universität Darmstadt
Darmstadt, Germany
{remus,biem}@cs.tu-darmstadt.de

Abstract

Distributional representations of words have been recently used in supervised settings for recognizing lexical inference relations between word pairs, such as hypernymy and entailment. We investigate a collection of these state-of-the-art methods, and show that they do not actually learn a relation between two words. Instead, they learn an independent property of a single word in the pair: whether that word is a "prototypical hypernym".

1 Introduction

Inference in language involves recognizing inference relations between two words (x and y), such as causality ($flu \rightarrow fever$), hypernymy ($cat \rightarrow animal$), and other notions of lexical entailment. The distributional approach to automatically recognize these relations relies on representing each word x as a vector $\vec{x}$ of *contextual features*: other words that tend to appear in its vicinity. Such features are typically used in word similarity tasks, where cosine similarity is a standard similarity measure between two word vectors: $sim(x, y) = cos(\vec{x}, \vec{y})$.

Many unsupervised distributional methods of recognizing lexical inference replace cosine similarity with an asymmetric similarity function (Weeds and Weir, 2003; Clarke, 2009; Kotlerman et al., 2010; Santus et al., 2014). Supervised methods, reported to perform better, try to learn the asymmetric operator from a training set. The various supervised methods differ by the way they represent each candidate pair of words (x, y): Baroni et al. (2012) use concatenation $\vec{x} \oplus \vec{y}$, others (Roller et al., 2014; Weeds et al., 2014; Fu et al., 2014) take the vectors' difference $\vec{y} - \vec{x}$, and more sophisticated representations, based on contextual features, have also been tested (Turney and Mohammad, 2014; Rimell, 2014).

In this paper, we argue that these supervised methods do not, in fact, learn to recognize lexical inference. Our experiments reveal that much of their previously perceived success stems from lexical memorizing. Further experiments show that these supervised methods learn whether y is a "prototypical hypernym" (i.e. a category), regardless of x, rather than learning a concrete relation between x and y.

Our mathematical analysis reveals that said methods ignore the interaction between x and y, explaining our empirical findings. We modify them accordingly by incorporating the similarity between x and y. Unfortunately, the improvement in performance is incremental. We suspect that methods based solely on contextual features of single words are not learning lexical inference relations because contextual features might lack the necessary information to deduce how one word relates to another.

2 Experiment Setup

Due to various differences (e.g. corpora, train/test splits), we do not list previously reported results, but apply a large space of state-of-the-art supervised methods and review them comparatively. We observe similar trends to previously published results, and make the dataset splits available for replication.[1]

[1] http://u.cs.biu.ac.il/~nlp/resources/downloads/

Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL, pages 970–976,
Denver, Colorado, May 31 – June 5, 2015.

Dataset	#Instances	#Positive	#Negative
Kotlerman 2010	2,940	880	2,060
Bless 2011	14,547	1,337	13,210
Baroni 2012	2,770	1,385	1,385
Turney 2014	1,692	920	772
Levy 2014	12,602	945	11,657

Table 1: Datasets evaluated in this work.

2.1 Word Representations

We built 9 word representations over Wikipedia (1.5 billion tokens) using the cross-product of 3 types of contexts and 3 representation models.

2.1.1 Context Types

Bag-of-Words Uses 5 tokens to each side of the target word (10 context words in total). It also employs subsampling (Mikolov et al., 2013a) to increase the impact of content words.

Positional Uses only 2 tokens to each side of the target word, and decorates them with their position (relative to the target word); e.g. the_{-1} is a common positional context of cat (Schütze, 1993).

Dependency Takes all words that share a syntactic connection with the target word (Lin, 1998; Padó and Lapata, 2007; Baroni and Lenci, 2010). We used the same parsing apparatus as in (Levy and Goldberg, 2014).

2.1.2 Representation Models

PPMI A word-context positive pointwise mutual information matrix M (Niwa and Nitta, 1994).

SVD We reduced M's dimensionality to $k = 500$ using Singular Value Decomposition (SVD).[2]

SGNS Skip-grams with negative sampling (Mikolov et al., 2013b) with 500 dimensions and 5 negative samples. SGNS was trained using a modified version of `word2vec` that allows different context types (Levy and Goldberg, 2014).[3]

2.2 Labeled Datasets

We used 5 labeled datasets for evaluation. Each dataset entry contains two words (x, y) and a label whether x entails y. Note that each dataset was created with a slightly different goal in mind, affecting word-pair generation and annotation. For example, both of Baroni's datasets are designed to capture hypernyms, while other datasets try to capture broader notions of lexical inference (e.g. causality). Table 1 provides metadata on each dataset, and the description below explains how each one was created.

(Kotlerman et al., 2010) Manually annotated lexical entailment of distributionally similar nouns.

(Baroni and Lenci, 2011) a.k.a. BLESS. Created by selecting unambiguous word pairs and their semantic relations from WordNet. Following Roller et al. (2014), we labeled noun hypernyms as positive examples and used meronyms, noun cohyponyms, and random noun pairs as negative.

(Baroni et al., 2012) Created in a similar fashion to BLESS. Hypernym pairs were selected as positive examples from WordNet, and then permutated to generate negative examples.

(Turney and Mohammad, 2014) Based on a crowdsourced dataset of 79 semantic relations (Jurgens et al., 2012). Each semantic relation was linguistically annotated as entailing or not.

(Levy et al., 2014) Based on manually annotated entailment graphs of subject-verb-object tuples (propositions). Noun entailments were extracted from entailing tuples that were identical except for one of the arguments, thus propagating the existence/absence of proposition-level entailment to the noun level. This dataset is the most realistic dataset, since the original entailment annotations were made in the context of a complete proposition.

2.3 Supervised Methods

We tested 4 compositions for representing (x, y) as a feature vector: **concat** ($\vec{x} \oplus \vec{y}$) (Baroni et al., 2012), **diff** ($\vec{y} - \vec{x}$) (Roller et al., 2014; Weeds et al., 2014; Fu et al., 2014), **only x** ($\vec{x}$), and **only y** ($\vec{y}$). For each composition, we trained two types of classifiers, tuning hyperparameters with a validation set: logistic regression with L_1 or L_2 regularization, and SVM with a linear kernel or quadratic kernel.

3 Negative Results

Based on the above setup, we present three negative empirical results, which challenge the claim that the methods presented in §2.3 are learning a relation between x and y. In addition to our setup, these results were also reproduced in preliminary exper-

[2] Following Caron (2001), we used the square root of the eigenvalue matrix Σ_k for representing words: $M_k = U_k\sqrt{\Sigma_k}$.

[3] `http://bitbucket.org/yoavgo/word2vecf`

Dataset	Lexical	+Contextual	Δ
Kotlerman 2010	.346	.437	.091
Bless 2011	.960	.960	.000
Baroni 2012	.638	.802	.164
Turney 2014	.644	.747	.103
Levy 2014	.302	.370	.068

Table 2: The performance (F_1) of lexical versus contextual feature classifiers on a random train/test split with lexical overlap.

Dataset	Best Supervised	Only $\vec{y}$	Unsupervised
Kotlerman 2010	.408	.375	**.461**
Bless 2011	**.665**	.637	.197
Baroni 2012	.774	.663	**.788**
Turney 2014	**.696**	.649	.642
Levy 2014	**.324**	**.324**	.231

Table 3: A comparison of each dataset's best supervised method with: (a) the best result using **only y** composition; (b) unsupervised cosine similarity $cos(\vec{x}, \vec{y})$. Performance is measured by F_1. Uses lexical train/test splits.

iments by applying the JoBimText framework[4] for scalable distributional thesauri (Biemann and Riedl, 2013) using Google's syntactic N-grams (Goldberg and Orwant, 2013) as a corpus.

Lexical Memorization is the phenomenon in which the classifier learns that a specific word in a specific slot is a strong indicator of the label. For example, if a classifier sees many positive examples where $y = animal$, it may learn that anything that appears with $y = animal$ is likely to be positive, effectively *memorizing* the word $animal$.

The following experiment shows that supervised methods with contextual features are indeed memorizing words from the training set. We randomly split each dataset into 70% train, 5% validation, and 25% test, and train lexical-feature classifiers, using a one-hot vector representation of y as input features. By definition, these classifiers memorize words from the training set. We then add contextual-features (as described in §2.1), on top of the lexical features, and train classifiers analogously. Table 2 compares the best lexical- and contextual-feature classifiers on each dataset. The performance difference is under 10 points in the larger datasets, showing that much of the contextual-feature classifiers' success is due to lexical memorization. Similar findings were also reported by Roller et al. (2014) and Weeds et al. (2014), supporting our memorization argument.

To prevent lexical memorization in our following experiments, we split each dataset into train and test sets with zero lexical overlap. We do this by randomly splitting the vocabulary into "train" and "test" words, and extract train-only and test-only subsets of each dataset accordingly. About half of each original dataset contains "mixed" examples (one train-word and one test-word); these are discarded.

[4] http://jobimtext.org

Supervised vs Unsupervised While supervised methods were reported to perform better than unsupervised ones, this is not always the case. As a baseline, we measured the "vanilla" cosine similarity of x and y, tuning a threshold with the validation set. This unsupervised symmetric method outperforms all supervised methods in 2 out of 5 datasets (Table 3).

Ignoring x's Information We compared the performance of **only y** to that of the best configuration in each dataset (Table 3). In 4 out of 5 datasets, the difference in performance is less than 5 points. This means that the classifiers are *ignoring* most of the information in x. Furthermore, they might be overlooking the compatibility (or incompatibility) of x to y. Weeds et al. (2014) reported a similar result, but did not address the fundamental question it beckons: if the classifier *cannot* capture a relation between x and y, then what is it learning?

4 Prototypical Hypernyms

We hypothesize that the supervised methods examined in this paper are learning whether y is a likely "category" word – a *prototypical hypernym* – and, to a lesser extent, whether x is a likely "instance" word. This hypothesis is consistent with our previous observations (§3).

Though the terms "instance" and "category" pertain to hypernymy, we use them here in the broader sense of entailment, i.e. as "tends to entail" and "tends to be entailed", respectively. We later show (§4.2) that this phenomenon indeed extends to other inference relations, such as meronymy.

4.1 Testing the Hypothesis

To test our hypothesis, we measure the performance of a trained classifier on *mismatched* instance-

Dataset	Top Positional Contexts of y
Kotlerman 2010	grave$_{-1}$, substances$_{+2}$, lend-lease$_{-1}$, poor$_{-2}$, bureaucratic$_{-1}$, physical$_{-1}$, dry$_{-1}$, air$_{-1}$, civil$_{-1}$
Bless 2011	other$_{-1}$, resembling$_{+1}$, such$_{+1}$, assemblages$_{+1}$, magical$_{-1}$, species$_{+1}$, any$_{-2}$, invertebrate$_{-1}$
Baroni 2012	any$_{-1}$, any$_{-2}$, social$_{-1}$, every$_{-1}$, this$_{-1}$, kinds$_{-2}$, exotic$_{-1}$, magical$_{-1}$, institute$_{-2}$, important$_{-1}$
Turney 2014	of$_{+1}$, inner$_{-1}$, including$_{+1}$, such$_{+1}$, considerable$_{-1}$, their$_{-1}$, extra$_{-1}$, types$_{-2}$, different$_{-1}$, other$_{-1}$
Levy 2014	psychosomatic$_{-1}$, unidentified$_{-1}$, auto-immune$_{+2}$, specific$_{-1}$, unspecified$_{-1}$, treatable$_{-2}$, any$_{-1}$

Table 4: Top positional features learned with logistic regression over **concat**. Displaying positive features of y.

category pairs, e.g. $(banana, animal)$. For each dataset, we generate a set of such synthetic examples S, by taking the positive examples from the test portion T^+, and extracting all of its instance words T_x^+ and category words T_y^+.

$$T_x^+ = \{x|(x,y) \in T^+\} \quad T_y^+ = \{y|(x,y) \in T^+\}$$

We then define S as all the in-place combinations of instance-category word pairs that did not appear in T^+, and are therefore likely to be false.

$$S = \left(T_x^+ \times T_y^+\right) \setminus T^+$$

Finally, we test the classifier on a sample of S (due to its size). Since all examples are assumed to be false, we measure the false positive rate as *match error* – the error of classifying a mismatching instance-category pair as positive.

According to our hypothesis, the classifier cannot differentiate between matched and mismatched examples (T^+ and S, respectively). We therefore expect it to classify a similar proportion of T^+ and S as positive. We validate this by comparing *recall* (proportion of T^+ classified as positive) to *match error* (proportion of S classified as positive). Figure 1 plots these two measures across all configurations and datasets, and finds them to be extremely close (regression curve: $match\ error = 0.935 \cdot recall$), thus confirming our hypothesis.

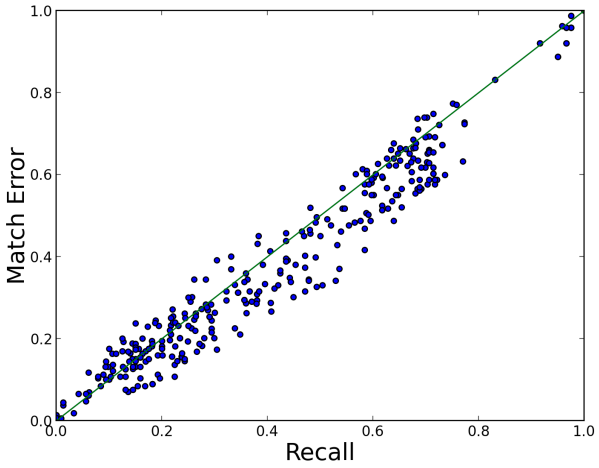

Figure 1: The correlation of recall (positive rate on T^+) with match error (positive rate on S) compared to perfect correlation (green line).

4.2 Prototypical Hypernym Features

A qualitative way of analyzing our hypothesis is to look at which features the classifiers tend to consider. Since SVD and SGNS features are not easily interpretable, we used PPMI with positional contexts as our representation, and trained a logistic regression model with L_1 regularization using **concat** over the entire dataset (no splits). We then observed the features with the highest weights (Table 4).

Many of these features describe dataset-specific category words. For example, in Levy's medical-domain dataset, many words entail "symptom", which is captured by the discriminative feature $psychosomatic_{-1}$. Other features are domain-independent indicators of category, e.g. any_{-1}, $every_{-1}$, and $kinds_{-2}$. The most striking features, though, are those that occur in Hearst (1992) patterns: $other_{-1}$, $such_{+1}$, $including_{+1}$, etc. These features appear in all datasets, and their analogues are often observed for x (e.g. $such_{-2}$). Even qualitatively, many of the dominant features capture prototypical or dataset-specific hypernyms.

As mentioned, the datasets examined in this work also contain inference relations other than hypernymy. In Turney's dataset, for example, 77 % of positive pairs are non-hypernyms, and y is often a quality ($coat \rightarrow warmth$) or a component ($chair \rightarrow legs$) of x. Qualities and components can often be detected via possessives, e.g. of_{+1} and $their_{-1}$. Other prominent features, such as $extra_{-1}$

and $exotic_{-1}$, may also indicate qualities. These examples suggest that our hypothesis extends beyond hypernymy to other inference relations as well.

5 Analysis of Vector Composition

Our empirical findings show that **concat** and **diff** are clearly ignoring the relation between x and y. To understand why, we analyze these compositions in the setting of a linear SVM. Given a test example, (x, y) and a training example that is part of the SVM's support (x_s, y_s), the linear kernel function yields Equations (1) for **concat** and (2) for **diff**.

$$K\left(\vec{x} \oplus \vec{y}, \vec{x_s} \oplus \vec{y_s}\right) = \vec{x} \cdot \vec{x_s} + \vec{y} \cdot \vec{y_s} \quad (1)$$

$$K\left(\vec{y} - \vec{x}, \vec{y_s} - \vec{x_s}\right) = \vec{x} \cdot \vec{x_s} + \vec{y} \cdot \vec{y_s} - \vec{x} \cdot \vec{y_s} - \vec{y} \cdot \vec{x_s} \quad (2)$$

Assuming all vectors are normalized (as in our experiments), the kernel function of **concat** is actually the similarity of the x-words plus the similarity of the y-words. Two dis-similarity terms are added to **diff**'s kernel, preventing the x of one pair from being too similar to the other pair's y (and vice versa).

Notice the absence of the term $\vec{x} \cdot \vec{y}$. This means that the classifier has no way of knowing if x and y are even related, let alone entailing. This flaw makes the classifier believe that any instance-category pair (x, y) is in an entailment relation, even if they are unrelated, as seen in §4. Polynomial kernels also lack $\vec{x} \cdot \vec{y}$, and thus suffer from the same flaw.

6 Adding Intra-Pair Similarity

Using an RBF kernel with **diff** slightly mitigates this issue, as it factors in $\vec{x} \cdot \vec{y}$, among other similarities:

$$K_{RBF}\left(\vec{y} - \vec{x}, \vec{y_s} - \vec{x_s}\right) = e^{-\frac{1}{\sigma^2}|(\vec{y}-\vec{x})-(\vec{y_s}-\vec{x_s})|^2}$$
$$= e^{-\frac{1}{\sigma^2}(\vec{x}\vec{y}+\vec{x_s}\vec{y_s}+\vec{x}\vec{x_s}+\vec{y}\vec{y_s}-\vec{x}\vec{y_s}-\vec{y}\vec{x_s}-2)} \quad (3)$$

A more direct approach of incorporating $\vec{x} \cdot \vec{y}$ is to create a new kernel, which balances intra-pair similarities with inter-pair ones:

$$K_{SIM}\left((\vec{x}, \vec{y}), (\vec{x_s}, \vec{y_s})\right) = \left(\vec{x}\vec{y} \cdot \vec{x_s}\vec{y_s}\right)^{\frac{\alpha}{2}} \left(\vec{x}\vec{x_s} \cdot \vec{y}\vec{y_s}\right)^{\frac{1-\alpha}{2}} \quad (4)$$

While these methods reduce match error – $match\ error = 0.618 \cdot recall$ versus the previous regression curve of $match\ error = 0.935 \cdot recall$ – their overall performance is only incrementally better than that of linear methods (Table 5). This improvement is also, partially, a result of the non-linearity introduced in these kernels.

Dataset	LIN(concat)	LIN(diff)	RBF(diff)	SIM
Kotlerman 2010	.367	.187	**.407**	.332
Bless 2011	.634	.665	.636	**.687**
Baroni 2012	.745	.769	.848	**.859**
Turney 2014	**.696**	.694	.691	.641
Levy 2014	.229	.219	**.252**	.244

Table 5: Performance (F_1) of SVM across kernels. **LIN** refers to the linear kernel (equations (1) and (2)), **RBF** to the Gaussian kernel (equation (3)), and **SIM** to our new kernel (equation (4)). Uses lexical train/test splits.

7 The Limitations of Contextual Features

In this work, we showed that state-of-the-art supervised methods for recognizing lexical inference appear to be learning whether y is a prototypical hypernym, regardless of its relation with x. We tried to factor in the similarity between x and y, yet observed only marginal improvements. While more sophisticated methods might be able to extract the necessary relational information from contextual features alone, it is also possible that this information simply *does not exist* in those features.

A (de)motivating example can be seen in §4.2. A typical y often has $such_{+1}$ as a dominant feature, whereas x tends to appear with $such_{-2}$. These features are relics of the Hearst (1992) pattern "y such as x". However, contextual features of single words cannot capture the *joint* occurrence of x and y in that pattern; instead, they record only this observation as two independent features of different words. In that sense, contextual features are inherently handicapped in capturing relational information, requiring supervised methods to harness complementary information from more sophisticated features, such as textual patterns that connect x with y (Snow et al., 2005; Turney, 2006).

Acknowledgements

This work was supported by the Adolf Messer Foundation, the Google Research Award Program, and the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1). We thank Stephen Roller for his valuable insights.

References

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

Marco Baroni and Alessandro Lenci. 2011. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10, Edinburgh, UK.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, Avignon, France.

Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! A framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.

John Caron. 2001. Experiments with LSA scoring: Optimal rank and basis. In *Proceedings of the SIAM Computational Information Retrieval Workshop*, pages 157–169.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 112–119, Athens, Greece.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1199–1209, Baltimore, Maryland.

Yoav Goldberg and Jon Orwant. 2013. A dataset of syntactic-ngrams over time from a very large corpus of english books. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 241–247, Atlanta, Georgia, USA.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*, pages 529–545, Nantes, France.

David A Jurgens, Peter D Turney, Saif M Mohammad, and Keith J Holyoak. 2012. Semeval-2012 task 2: Measuring degrees of relational similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 356–364, Montréal, Quebec, Canada.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 4(16):359–389.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland.

Omer Levy, Ido Dagan, and Jacob Goldberger. 2014. Focused entailment graphs for open ie propositions. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 87–97, Baltimore, Maryland.

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, Montréal, Quebec, Canada.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Yoshiki Niwa and Yoshihiko Nitta. 1994. Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *COLING 1994 Volume 1: The 15th International Conference on Computational Linguistics*, Kyoto, Japan.

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

Laura Rimell. 2014. Distributional lexical entailment by topic coherence. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 511–519, Gothenburg, Sweden.

Stephen Roller, Katrin Erk, and Gemma Boleda. 2014. Inclusive yet selective: Supervised distributional hypernymy detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1025–1036, Dublin, Ireland.

Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *Proceedings of the 14th*

Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers, pages 38–42, Gothenburg, Sweden.

Hinrich Schütze. 1993. Part-of-speech induction from scratch. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 251–258, Columbus, Ohio, USA.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing*.

Peter D Turney and Saif M Mohammad. 2014. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering*, pages 1–40.

Peter D Turney. 2006. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 81–88, Sapporo, Japan.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259, Dublin, Ireland.