Mapping the Paraphrase Database to WordNet

Anne Cocos[*], **Marianna Apidianaki**[*♠] and **Chris Callison-Burch**[*]
[*] Computer and Information Science Department, University of Pennsylvania
[♠] LIMSI, CNRS, Université Paris-Saclay, 91403 Orsay
`{acocos,marapi,ccb}@seas.upenn.edu`

Abstract

WordNet has facilitated important research in natural language processing but its usefulness is somewhat limited by its relatively small lexical coverage. The Paraphrase Database (PPDB) covers 650 times more words, but lacks the semantic structure of WordNet that would make it more directly useful for downstream tasks. We present a method for mapping words from PPDB to WordNet synsets with 89% accuracy. The mapping also lays important groundwork for incorporating WordNet's relations into PPDB so as to increase its utility for semantic reasoning in applications.

1 Introduction

WordNet (Miller, 1995; Fellbaum, 1998) is one of the most important resources for natural language processing research. Despite its utility, WordNet[1] is manually compiled and therefore relatively small. It contains roughly 155k words, which does not approach web scale, and very few informal or colloquial words, domain-specific terms, new word uses, or named entities. Researchers have compiled several larger, automatically-generated thesaurus-like resources (Lin and Pantel, 2001; Dolan and Brockett, 2005; Navigli and Ponzetto, 2012; Vila et al., 2015). One of these is the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013; Pavlick et al., 2015b). With over 100 million paraphrase pairs, PPDB dwarfs WordNet in size but it lacks WordNet's semantic structure. Paraphrases for a given word are indistinguishable by sense, and PPDB's only inherent semantic relational information is predicted entailment relations between word types (Pavlick et al., 2015a). Several earlier studies attempted to incorporate se-

RULE-PRESCRIPT: **imperative*, demand*, duty*, request**, gun, **decree**, ranking
RULE-REGULATION: **constraint*, limit*,** derogation*, notion
RULE-FORMULA: **method*,** standard*, **plan*,** proceeding
RULE-LINGUISTIC RULE: notion

Table 1: Example of our model's top-ranked paraphrases for three WordNet synsets for *rule (n)*. Starred paraphrases have a predicted likelihood of attachment of at least 95%; others have predicted likelihood of at least 50%. Bold text indicates paraphrases that match the correct sense of *rule*.

mantic awareness into PPDB, either by clustering its paraphrases by word sense (Apidianaki et al., 2014; Cocos and Callison-Burch, 2016) or choosing appropriate PPDB paraphrases for a given context (Apidianaki, 2016; Cocos et al., 2017). In this work, we aim to marry the rich semantic knowledge in WordNet with the massive scale of PPDB by predicting WordNet synset membership for PPDB paraphrases that do not appear in WordNet. Our goal is to increase the lexical coverage of WordNet and incorporate some of the rich relational information from WordNet into PPDB. Table 1 shows our model's top-ranked outputs mapping PPDB paraphrases for the word *rule* onto their corresponding WordNet synsets.

Our overall objective in this work is to map PPDB paraphrases for a target word to the WordNet synsets of the target. This work has two parts. In the first part (Section 4), we train and evaluate a binary lemma-synset membership classifier. The training and evaluation data comes from lemma-synset pairs with known class (member/non-member) from WordNet. In the second part (Section 5), we predict membership for lemma-synset pairs where the lemma appears in PPDB, but not in WordNet, using the model trained in part one.

---

[1] In this work we refer specifically to WordNet version 3.0

2 Related Work

There has been considerable research directed at expanding WordNet's coverage either by integrating WordNet with additional semantic resources, as in Navigli and Ponzetto (2012), or by automatically adding new words and senses. In the second case, there have been several efforts specifically focused on hyponym/hypernym detection and attachment (Snow et al., 2006; Shwartz et al., 2016). There is also previous work aimed at adding semantic structure to PPDB. Cocos and Callison-Burch (2016) clustered paraphrases by word sense, effectively forming synsets within PPDB. By mapping individual paraphrases to WordNet synsets, our work could be used in coordination with these previous results in order to extend WordNet relations to the automatically-induced PPDB sense clusters.

3 WordNet and PPDB Structure

The core concept in WordNet is the synonym set, or *synset* – a set of words meaning the same thing. Since words can be polysemous, a given lemma may belong to multiple synsets corresponding to its different senses. WordNet also defines relationships between synsets, such as hypernymy, hyponymy, and meronymy. In the rest of the paper, we will use $S(w_p)$ to denote the set of WordNet synsets containing word w_p, where the subscript p denotes the part of speech. Each synset $s_p^i \in S(w_p)$ is a set containing w_p as well as its synonyms for the corresponding sense. PPDB also has a graph structure, where nodes are words, and edges connect mutual paraphrases. We will use $PPDB(w_p)$ to denote the set of PPDB paraphrases connected to target word w_p.

4 Predicting Synset Membership

Our objective is to map paraphrases for a target word, t, to the WordNet synsets of the target. For a given target word in a vocabulary, we make a binary synset-attachment prediction between each of t's paraphrases, $w_p \in PPDB(t)$, and each of t's synsets, $s_p^i \in S(t)$. We predict the likelihood of a word w_p belonging to synset s_p^i on the basis of multiple features describing their relationship. We construct features from four primary types of information.

PPDB 2.0 Score The PPDB 2.0 Score is a supervised metric trained to estimate the strength of the paraphrase relationship between pairs of words connected in PPDB (Pavlick et al., 2015b). Scores range roughly from 0 to 5, with 5 indicating a strong paraphrase relationship. We compute several features for predicting whether a word w_p belongs to synset s_p^i as follows. We call the set of all lemmas belonging to s_p^i and any of its hypernym or hyponym synsets the *extended* synset s_p^{+i}. We calculate features that correspond to the average and maximum PPDB scores between w_p and lemmas in s_p^{+i}:

$$x_{ppdb.max} = \max_{w' \in s_p^{+i}} PPDBScore(w_p, w')$$

$$x_{ppdb.avg} = \frac{\sum_{w' \in s_p^{+i}} PPDBScore(w_p, w')}{|s_p^{+i}|}$$

Distributional Similarity Our distributional similarity feature encodes the extent to which the word and lemmas from the synset tend to appear within similar contexts. Word embeddings are real-valued vector representations of words that capture contextual information from a large corpus. Comparing the embeddings of two words is a common method for estimating their semantic similarity and relatedness. Embeddings can also be constructed to represent word *senses* (Iacobacci et al., 2015; Flekova and Gurevych, 2016; Jauhar et al., 2015; Ettinger et al., 2016). Camacho-Collados et al. (2016) developed compositional vector representations of WordNet noun senses – called *NASARI* embedded vectors – that are computed as the weighted average of the embeddings for words in each synset. They share the same embedding space as a publicly available[2] set of 300-dimensional word2vec embeddings covering 300 million words (hereafter referred to as the *word2vec* embeddings) (Mikolov et al., 2013a,b). We calculate a distributional similarity feature for each word-synset pair by simply taking the cosine similarity between the word's *word2vec* vector and the synset's *NASARI* vector:

$$x_{distrib} = cos(v_{NASARI}(s_p^i), v_{word2vec}(w_p))$$

where v_{NASARI} and $v_{word2vec}$ denote the target word and synset embeddings respectively. Since NASARI covers only nouns, and only 80% of the noun synsets for our target vocabulary are in NASARI, we construct weighted vector representations for the remaining 20% of noun synsets and

---

[2]https://code.google.com/archive/p/word2vec/

all non-noun synsets as follows. We take the vector representation for each synset *not* in NASARI to be the weighted average of the *word2vec* embeddings of the synset's lemmas, where weights are determined by the PPDB2.0 Score between the lemma and the target word, if it exists, or 1.0 if it does not:

$$v(s_p^i) = \frac{\sum_{l \in s_p^i} PPDBScore(t, l) \cdot v_{word2vec}(l)}{\sum_{l \in s_p^i} PPDBScore(t, l)}$$

Lesk Similarity Among the information contained in WordNet for each synset is its definition, or gloss. The simplified Lesk algorithm (Vasilescu et al., 2004) identifies the most likely sense of a target word in context by measuring the overlap between the given context and the definition of each target sense. We use a slightly modified version of the algorithm to compute features that measure the overlap between the PPDB paraphrases for the target and the gloss of a synset. For calculating these Lesk-based features, we find synset glosses from WordNet 3.0 and from Babel-Net v3.0 (Navigli and Ponzetto, 2012). First, we find D, the set of *content words* of the gloss for synset s_p^i, by taking all nouns, verbs, adjectives, and adverbs that appear within the gloss. In cases where more than one gloss is available, we take D to be the set of all content words in all glosses. We also calculate an extended version of each feature, in which we take D to be the set of content words, plus the PPDB paraphrases *for each content word*. Next, we calculate features that measure the relationship between the paraphrase w_p and the words in D in terms of PPDB2.0 Scores. These features include the maximum PPDB score between the paraphrase and any word in D, the average score over all words in D, the percent of words in D that are connected to the paraphrase in PPDB, and the count of words in D that are connected to the paraphrase in PPDB:

$$x_{lesk.max} = \max_{d \in D} PPDBScore(w_p, d)$$

$$x_{lesk.avg} = \frac{\sum_{d \in D} PPDBScore(w_p, d)}{|D|}$$

$$x_{lesk.cnt} = |\{d \in D : PPDBScore(w_p, d) > 0\}|$$

$$x_{lesk.pct} = \frac{|\{d \in D : PPDBScore(w_p, d) > 0\}|}{|D|}$$

Lexical Substitutability The fourth feature type that we compute to predict whether word w_p belongs to synset s_p^i is based on the substitutability of w_p for instances of s_p^i in context. To compute this feature we measure lexical substitutability using a simple but high-performing vector space model, AddCos (Melamud et al., 2015). The AddCos method quantifies the fit of substitute word s for target word t in context C by measuring the semantic similarity of the substitute to the target, and the similarity of the substitute to the context:

$$AddCos(s, t, C) = \frac{|C| \cdot cos(s, t) + \sum_{c \in C} cos(s, c)}{2 \cdot |C|}$$

The vectors s and t are word embeddings of the substitute and target generated by the *skip-gram with negative sampling* model (Mikolov et al., 2013b,a). The context C is the set of words appearing within a fixed-width window of the target t in a sentence (we use a window of 2), and the embeddings c are context embeddings generated by *skip-gram*. In our implementation, we train 300-dimensional word and context embeddings over the 4B words in the Annotated Gigaword (AGiga) corpus (Napoles et al., 2012) using the gensim word2vec package (Mikolov et al., 2013b,a; Řehůřek and Sojka, 2010). [3]

To compute the lexical substitutability score between a word w_p and synset s_p^i, we first retrieve example sentences $e \in E$ containing t in sense s_p^i from BabelNet v3.0 (Navigli and Ponzetto, 2012). Then, for each example e, we compute the AddCos lexical substitutability between w_p and the target word in context C_e. We compute two types of this feature: The average AddCos score over all synset examples, and the maximum AddCos score over all synset examples.

$$x_{addcos.max} = max_{e \in E} AddCos(w_p, t, C_e)$$

$$x_{addcos.avg} = avg_{e \in E} AddCos(w_p, t, C_e)$$

Derived Features For each paraphrase, we also compute a set of derived features using the softmax and logodds functions over all synsets with which that paraphrase is paired. This is to encode the relative strength of association with each synset as compared to the others.

For a given feature x_* calculated between lemma w_p and synset s_p^i, the derived versions of the feature are calculated as:

$$x_*^{softmax}(w_p, s_p^i) = \frac{e^{x_*(w_p, s_p^i)}}{\sum_{s_p^j \in S(w_p)} e^{x_*(w_p, s_p^j)}}$$

---

[3]The `word2vec` training parameters we use are a context window of size 3, learning rate *alpha* from 0.025 to 0.0001, minimum word count 100, sampling parameter $1e^{-4}$, 10 negative samples per target word, and 5 training epochs.

	Cross-Validation				Cross-Validation-LexSplit				Test			
	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1	Acc.
Baseline: All negative attachments	0	0.854	0	0.854	0	0.854	0	0.854	0	0.858	0	0.858
Baseline: PPDB2.0 Score Match	0.536	0.419	0.471	0.862	0.536	0.419	0.471	0.862	0.268	0.718	0.390	0.681
Gaussian Naive Bayes (All Feat.)	0.528	0.369	0.372	0.800	0.527	0.310	0.352	0.825	0.496	0.282	0.359	0.858
Gaussian Naive Bayes (Sel. Feat.)	0.605	0.600	**0.600**	**0.883**	0.606	0.572	**0.581**	**0.882**	0.622	0.558	**0.588**	**0.889**

Table 2: Precision, recall, F1, and accuracy results over the training set (normal 10-fold Cross-Validation, and lexical split 20-fold Cross-Validation-LexSplit) and test set for predicting paraphrase-synset attachment.

$$x_*^{logodds}(w_p, s_p^i) = \ln \frac{x_*(w_p, s_p^i) + \alpha}{\sum_{s_p^j \in S(w_p), i \neq j}(x_*(w_p, s_p^j) + \alpha)}$$

Model Training We train a binary classification model that takes lemma-synset pairs as input, and predicts whether the lemma belongs in the synset. We train the model by generating features for a set of lemma-synset pairs from WordNet for which we know the correct classification. We evaluate whether the resulting model correctly finds lemma-synset pairs that belong together.

Our target vocabulary comes from the SensEval3 English Lexical Sample Task data (Mihalcea et al., 2004) which contains sentences corresponding to 57 noun, verb, and adjective lemmas. Each sentence may contain a different form of the lemma (i.e. different in number or tense), and PPDB paraphrases vary depending on the form. So we take the set of all forms of all lemmas (251 word types in total) as our target vocabulary. To generate pairs for training and evaluation, for each of the 251 targets w_p, we find the lemmas in the intersection of w_p's synsets – $S(w_p)$ – and its paraphrases – $PPDB(w_p)$. We call the set of lemmas in the intersection $L(w_p)$. Then, we take the lemma-synset pairs in $L(w_p) \times S(w_p)$ as instances for training and evaluation. The total number of resulting lemma-synset pairs is 7459. We randomly divide these into 80% training and 20% test pairs.

We then generate all variations of each of the four feature types for the lemma-synset pairs in our training and test sets. In the case of positive synset-lemma pairs – i.e. those pairs for which the lemma actually belongs to the WordNet synset – we exclude the lemma from the synset before calculating the PPDB Score and distributional features.

Finally, we train a Gaussian Naive Bayes (GNB) classification model over the training data.

GNB is advantageous for our setting, as 10% percent of our instances have missing data (e.g. in the case where a synset does not have an example). For feature selection, we use two versions of cross-validation. The first is standard 10-fold *Cross-Validation*. In order to estimate how well our model will generalize to unseen lemmas, we also experiment with a lexical split technique described in Roller and Erk (2016) (*Cross-Validation-LexSplit*). This method ensures that for each cross-validation fold, none of the lemmas in that fold's validation lemma-synset pairs are seen in the training split. Specifically, for each split we randomly select 5% of training pairs for validation and take the remainder of the training set that does not share a lemma with the validation set as that fold's training instances. As a result, the validation set size remains constant for each fold, but the training set sizes may vary between folds.

We train two versions of the model. The first (*All Features*) uses all computed features. The second (*Selected Features*) includes features selected using cross validation (the selected features were the same using standard and lexical split cross-validation). We select one feature of each type (PPDB Score, distributional, Lesk, and lexical substitutability) whose combination maximizes cross-validation F1 score. The selected features are $x_{lesk.cnt}$ (non-extended), $x_{distrib}$, $x_{addcos.max}$, and $softmax(x_{ppdb.max})$.

Model Evaluation We report results of the model using all features, and the results of the best model achieved after feature selection (Table 2). In each case we give both the Cross-Validation and Cross-Validation-LexSplit performances, and performance on the held-out test set. We compare our model to two simple baselines. The first predicts all negative attachments, which yields an accuracy of 85.8% on the test set (with F1 of 0). The second baseline maps each paraphrase to the synset of t with which it has the highest-scoring
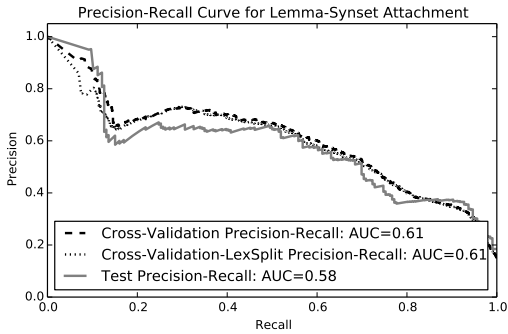
Figure 1: Precision-Recall curve for our paraphrase-synset attachment classifier.

PPDB feature ($x_{ppdb.max}$) and yields an accuracy of 68.1% on the test set. In comparison, our GNB model with selected features yields an accuracy of over 88% on the cross-validation and test sets. Both cross-validation and test accuracies are significantly higher than baselines (based on McNemar's test, $p < .001$).

Ablated Feature Type	Change in Cross-Val. F1
PPDB	0.042
Lexical Substitution	0.031
Distributional	0.009
Lesk	-0.092

Table 3: Absolute decrease in mean cross-validation F1 with different feature types ablated. Higher numbers indicate greater feature importance.

In order to interpret the importance of each feature type, we also run an ablation experiment where we train our GNB model with all features except those from a particular type (Table 3). We find that removing the PPDB features leads to the greatest drop in cross-validation F1, indicating that these are the most important for our classifier. Ablating all Lesk features improved F1, but on further analysis we found that ablating only the derived Lesk log-odds features led to a decrease in F1. This suggests that the Lesk features in general are useful for classification, but the derived Lesk log-odds features are not.

5 Mapping PPDB to WordNet

Using our trained lemma-synset attachment classifier, we can now augment the lexical coverage of WordNet with PPDB paraphrases. For the 251 targets in our original dataset, we retrieve the PPDB paraphrases (with PPDB score greater than 2.5, to ensure high-quality paraphrases) that do not be-

long to any synset of the target or any of their direct hypernyms or hyponyms. We then make an attachment prediction between each remaining paraphrase and each of the target's WordNet synsets. In total, we make 160,813 unique paraphrase-synset attachment predictions for the 4821 unique paraphrase lemmas and 458 unique synsets associated with the targets in our dataset.

When we map PPDB to WordNet we can estimate the expected precision and recall of attachment decisions based on the results of our model evaluation on the test set. If we would like to emphasize precision over recall in the predicted attachments, we can adjust a threshold for attachment corresponding to the predicted likelihood of our model (as shown in Figure 1). At a threshold of 50% predicted likelihood, our classifier predicts attachment for 7032 (4.4%) of the paraphrase-synset pairs with an estimated precision of 62.2%. If we increase the threshold to 95% predicted likelihood, the number of predicted attachments is 3690 (2.3%) with an estimated precision of 66.3%. With the publication of this paper we release our PPDB to WordNet mapping results.

6 Conclusion

We have proposed a method for mapping PPDB paraphrases to WordNet synsets. Our classifier makes accurate paraphrase-synset attachment predictions using features that capture paraphrase and distributional similarity, and the substitutability of paraphrases and synsets in context. The results show that the classifier can successfully add new PPDB paraphrases to WordNet synsets and increase their coverage.

Acknowledgments

References

Marianna Apidianaki. 2016. Vector-space models for ppdb paraphrase ranking in context. In *Proceedings of EMNLP*. Austin, Texas, pages 2028–2034.

Marianna Apidianaki, Emilia Verzeni, and Diana McCarthy. 2014. Semantic Clustering of Pivot Paraphrases. In *Proceedings of LREC*. Reykjavik, Iceland, pages 4270–4275.

José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. *Artificial Intelligence* 240:36–64.

Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2017. Word sense filtering improves embedding-based lexical substitution. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*. pages 110–119.

Anne Cocos and Chris Callison-Burch. 2016. Clustering Paraphrases by Word Sense. In *Proceedings of NAACL/HLT*. San Diego, California, pages 1463–1472.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing*. Jeju Island, Korea, pages 9–16.

Allyson Ettinger, Philip Resnik, and Marine Carpuat. 2016. Retrofitting sense-specific word vectors using parallel text. In *Proceedings of NAACL-HLT*. San Diego, California, pages 1378–1383.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Lucie Flekova and Iryna Gurevych. 2016. Supersense Embeddings: A Unified Model for Supersense Interpretation, Prediction, and Utilization. In *Proceedings of ACL*. Berlin, Germany, pages 2029–2041.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of NAACL/HLT*. Atlanta, Georgia, pages 758–764.

Ignacio Iacobacci, Mohammed Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: Learning Sense Embeddings for Word and Relational Similarity. In *Proceedings of ACL/IJCNL*. Beijing, China, pages 95–105.

Sujay Kumar Jauhar, Chris Dyer, and Eduard Hovy. 2015. Ontologically Grounded Multi-sense Representation Learning for Semantic Vector Space Models. In *Proceedings of NAACL*. Denver, Colorado, pages 683–693.

Dekang Lin and Patrick Pantel. 2001. Dirt@ sbt@ discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 323–328.

Oren Melamud, Omer Levy, and Ido Dagan. 2015. A Simple Word Embedding Model for Lexical Substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. Denver, Colorado, pages 1–7.

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. 2004. The Senseval-3 English lexical sample task. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*. Barcelona, Spain, pages 25–28.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*. pages 3111–3119.

George A. Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 31(11):39–41.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*. Montréal, Canada, pages 95–100.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* 193:217–250.

Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Benjamin Van, and Durme Chris Callison-burch. 2015a. Adding semantics to data-driven paraphrasing. In *Proceedings of ACL*. Beijing, China, pages 425–430.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevich, and Chris Callison-Burch Ben Van Durme. 2015b. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of ACL*. Beijing, China, pages 425–430.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, pages 45–50.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *Proceedings of EMNLP*. pages 2163–2172.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of ACL*. Berlin, Germany, pages 2389–2398.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of ACL*. pages 801–808.

Florentina Vasilescu, Philippe Langlais, and Guy Lapalme. 2004. Evaluating Variants of the Lesk Approach for Disambiguating Words. In *Proceedings of LREC*. Lisbonne, Portugal, pages 633–636.

Marta Vila, Horacio Rodríguez, and M Antònia Martí. 2015. Relational paraphrase acquisition from wikipedia: The WRPA method and corpus. *Natural Language Engineering* 21(03):355–389.