Adding Semantics to Data-Driven Paraphrasing: Supplementary Material

1 Data Annotation

We use Amazon Mechanical Turk (MTurk) to collect labels for our phrase pairs. We show each pair to 5 independent workers, and ask each worker to use their best judgement to label the relationship that holds between the words. The workers were asked to choose one of 7 relations, or to mark that "I cannot tell." The exact options given to the workers are shown in Figure 1. These options are based on the natural logic relations described, although with some simplifications. We omit the cover relation entirely, as its practicality is not obvious, and we replace "negation" with the weaker notion of "antonyms" or "opposites." Workers showed moderate agreement overall, with Fleiss's $\kappa = 0.56$ (Landis and Koch, 1977). Table 1 gives agreements for each relation individually. We take the majority label for each pair as the true label, breaking ties at random. Ties occurred in about 25% of cases.

Quality Control In order to measure worker reliability, we embedded gold-standard examples of synonyms ($\equiv$) and antonyms ($\hat{}$) from WordNet. We drew random pairs of words as gold standard examples of independent ($\#$) pairs. After inspecting the WordNet hypernym and hyponym pairs ourselves, we decided they were too unclear to be used as gold-standard examples. We considered any of $\equiv, \sqsupset, \sqsubset$ to be correct for the synonyms; this choice was made after we looked through the synonym controls and determined than many could be better labeled as hypernyms (*morning/sunrise*) or hyponyms (*fabric/material*). Each HIT consisted of two control questions, and workers who fell below 50% accuracy were rejected. Workers achieved 82% accuracies on our controls overall: 92% on the independent pairs, 70% on the synonyms, and 64% on the antonyms. Of the synonyms, 50% were labeled $\equiv$ and another 20% were labeled either $\sqsupset$ or $\sqsubset$.

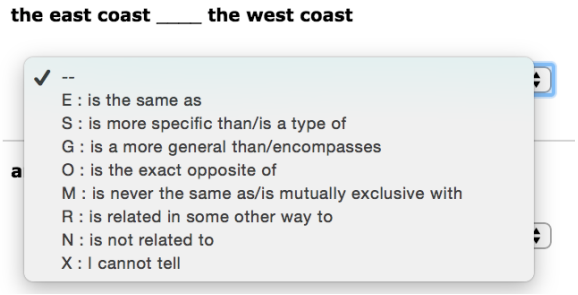

Figure 1: Turkers were asked to choose one of the above annotations to describe the relationship of the first word to the second. These options correspond to the 6 basic entailment relations plus "other".

		κ	N
$\equiv$	is the same as	0.46	811
$\sqsubset$	is more specific than	0.51	1,171
$\sqsupset$	is more general than	0.47	1,330
$\hat{}$	is the exact opposite of	0.75	280
$\mid$	is mutually exclusive with	0.41	442
$\sim$	is related in some other way to	0.43	1,825
$\#$	is not related to	0.66	3,722
	Overall	0.56	9,603

Table 1: Inter-annotator agreement and number of pairs for each of the entailment options given to Turkers.

2 Automatic Classification

2.1 Features

Lexical features We compute a variety of simple lexical features for each phrase pair. These include features about the words in each phrase, the length of each phrase, and the part of speech tags for each word in the phrase, as well as string similarity features, including Levenstein distance, Jaccard similarity, Hamming similarity, and common substrings. A full list is given in Table 2.

Distributional features We follow Lin and Pantel (2001) in computing context vectors for each word based on its dependency contexts in the Annotated Gigaword corpus (Napoles et al., 2012).

Binary	p_1 is a substring of p_2
Binary	p_2 is a substring of p_1
Binary	fine-POS(p_1) == fine-POS(p_2)
Binary	coarse-POS(p_1) == coarse-POS(p_2)
Binary	Both p_1 and p_2 are lexical
Binary	Either p_1 or p_2 is phrasal
Binary (sparse)	all words in p_1, position unspecified
Binary (sparse)	all words in p_2, position unspecified
Binary (sparse)	all words in p_1, noted as p_1
Binary (sparse)	all words in p_2, noted as p_2
Binary (sparse)	all POS tags in p_1, noted as p_1
Binary (sparse)	all POS tags in p_2, noted as p_2
Real-valued	Number of words in p_1
Real-valued	Number of words in p_2
Real-valued	Number of shared POS tags
Real-valued	levenstein(p_1, p_2)
Real-valued	jaccard(p_1, p_2)
Real-valued	hamming(p_1, p_2)

Table 2: Lexical features for a phrase pair $\langle p_1, p_2 \rangle$. "Position unspecified" means the feature reflected e.g that the word appeared at all in the phrase pair. "Noted as p_1" means that the feature was specific to the word having been observed in the first phrase of the pair.

For a single word w, we compute the "dependency context" vector by simply considering every dependency relation in which the w participates. When w is the governor of a relation r and *v* is the dependent, we record the context as *r:gov:v*; when w is the dependent of a relation r and *v* is the governor, we record the relation as *r:dep:v*. For multiword phrases $p = w_1 \dots w_k$, we consider the dependency context of p to be the combined dependency contexts *r:*:v* of the words $w_1 \dots w_k$, subject to the constraint that *v* is not one of $w_1 \dots w_k$.

Given the phrase pair $\langle p_1, p_2 \rangle$, let P_1 be the set of contexts of p_1 and P_2 the set of contexts of p_2. We compute the following features:

- The number of contexts for each phrase: $|P_1|$,$|P_2|$
- The difference in the number of contexts: $|P_1| - |P_2|$
- The number of shared contexts: $|P_1 \cap P_2|$
- The Jaccard similarity of the contexts: $|P_1 \cap P_2| / |P_1 \cup P_2|$

Let $w_1(c)$ be the number of times p_1 was observed in context c, and $w_2(c)$ be the number of times p_2 was observed in context c. We compute the various symmetric and asymmetric similarity measure (Lin, 1998; Weeds et al., 2004; Szpektor and Dagan, 2008; Clarke, 2009) using definitions given in Kotlerman et al. (2010):

$$lin = \frac{\sum_{c \in P_1 \cap P_2} w_1(c) + w_2(c)}{\sum_{c \in P_1} w_1(c) + \sum_{c \in P_2} w_2(c)},$$

$$weeds = \frac{\sum_{c \in P_1 \cap P_2} w_1(c)}{\sum_{c \in P_1} w_1(c)},$$

$$clark = \frac{\sum_{c \in P_1 \cap P_2} \min(w_1(c), w_2(c))}{\sum_{c \in P_1} w_1(c)},$$

$$balprec = \sqrt{lin \times weeds}.$$

Paraphrase features There are a variety of features distributed with PPDB, which we include in our classifier. These include 33 different measures used to sort the goodness of the paraphrases, including distributional similarity, bilingual alignment probabilities, and lexical similarity. Among those we found to have the best signal were *p(f | e)* and *p(e | f)*, the paraphrase probabilities for phrase pair calculated according to Bannard and Callison-Burch (2005), and *AGigaSim*, the distributional similarity of the two words computed over the Annotated Gigaword corpus. A complete list is given in Ganitkevitch and Callison-Burch (2014).

Translation features PPDB is based on the "bilingual pivoting"' method, in which two phrases are considered paraphrases if they share a foreign translation. The English PPDB was built by pivoting through 24 foreign languages. We use these pivot words as features. For each pair of phrases $\langle p_1, p_2 \rangle$ in our data and each language l, we compute two asymmetric similarity scores sim_{l_1} and sim_{l_2} capturing the number of shared translations as a fraction of the total translations of each phrase:

$$sim_{l_1} = \frac{| \, t_l(p_1) \cap t_l(p_2) \, |}{| \, t_l(p_1) \, |}$$

and

$$sim_{l_2} = \frac{| \, t_l(p_1) \cap t_l(p_2) \, |}{| \, t_l(p_2) \, |}$$

where $t_l(p)$ is set of observed translations of the phrase p in language l. We compute these ratios by looking at each language l separately as well as by pooling the translations from all languages, e.g.

$$sim_{*_1} = \frac{\mid t_*(p_1) \cap t_*(p_2) \mid}{\mid t_*(p_1) \mid}$$

where $t_*(p)$ is the pooled set of observed translations of the phrase p across all languages:

$$t_*(p) = \bigcup_l t_l(p).$$

We also compute the mean, minimum, and maximum of the ratios across languages, e.g.

$$\text{mean}_1 = \frac{1}{\text{\# languages}} \sum_l sim_{l_1}.$$

Path features We use the Annotated Gigaword corpus to compute path features as in Snow et al. (2004). For each pair $\langle p_1, p_2 \rangle$, we find all sentences in the corpus in which the phrases co-occur, and find all paths through the dependency tree which connect the pair, ignoring paths longer than 5 nodes. If p_1 or p_2 is a multiword phrase, we collapse the entire phrase into a single node, so that we consider all paths which originate from any word in p_1 and end at any word in p_2, subject to the constraint that none of the intermediate nodes on the path belong to p_1 or p_2.

We build a path lexicon consisting of all paths which occurred between at least 5 unique pairs in our data set. Then, the feature vector for $\langle p_1, p_2 \rangle$ is a binary vector indicating whether or not the pair was observed with each path in our path lexicon. We use three separate features to indicate the special cases which p_1 was not observed anywhere in Gigaword, p_2 was not observed anywhere in Gigaword, or p_1 never co-occured in a sentence with p_2.

WordNet Features We include features to capture the WordNet relation for each pair $\langle p_1, p_2 \rangle$. We consider WordNet's defined synonym, hypernym, hyponym, and antonym relations, as well as the *holonym*, *meronym*, *cause*, *entailment*, *derivationally-related*, *similar-to*, *also-see*, and *attribute* links. We define the relation *alternation* in WordNet as holding when p_1 and p_2 share a common parent, but are not themselves in a hypernymy relationship. We define the relation *independence* in WordNet as holding when both p_1 and p_2 appear in WordNet but none of the previously defined relationships hold.

For each relation r and each part of speech pos (noun, verb, adjective, and adverb), we include a binary feature r_{pos} indicating whether WordNet contains any senses for p_1 and p_2 with POS pos such that that r holds. We use special OOV_{pos} features to signify that either p_1 or p_2 did not appear in WN with the given POS tag pos.

2.2 Training

We use the scikit-learn[1] toolkit to train a logistic regression classifier. In order to overcome the imbalanced distribution of our dataset, we subsample training examples from each class inversely proportionally to the class's frequency in the training data; this is corresponds to the `class_weight='auto'` parameter setting.

3 Nutcracker Configuration

We run NC without the paraphrasing preprocessing step which was used to achieve the results reported in Marelli et al. (2014). Our reason for doing so is that the paraphrasing step uses PPDB and interferes with our ability to isolate the effect of our entailment annotations on the end-to-end performance of the system. As a result, our numbers differ slightly from the state-of-the-art performance reported for Nutcracker.

4 Evaluation of Predicted Entailment Relations in Full PPDB

The main evaluation in the paper focuses on the pairs in PPDB which also appear in RTE data. We also evaluate the quality of the entailment relations for randomly chosen paraphrase pairs from the database. We expect performance on these paraphrase pairs to be much lower, since the pairs cover more complex syntactic categories (e.g. *which have resulted/and that have led*) and more abstract expressions (e.g. *go back/start all over again*).

To evaluate these relations, we take a random sample of 1,000 pairs for each of the predicted relation types ($\#$, $\equiv$, $\sqsubset$, $\neg$, $\sim$). We take a stratified sample across confidence levels: i.e. for each relation, we take all the pairs that the classifier predicted as having that relation, divide the list into 5 buckets based on the classifier's confidence in the prediction, and sample evenly from each bucket. For the $\neg$ relation, there are 430 pairs, so we take all of them. We gather labels on MTurk the same way we did for the training data. Table 3 shows the precisions for each relation at varying levels of

[1] `http://scikit-learn.org`

confidence. Note that when the classifier predicts a directed entailment, we fix the direction to the forward entailment ($\sqsubset$) direction.

The classifiers results are very good for the $\equiv$ and $\neg$ classes. The performance is lower for the $\sqsubset$ relation, but most of these errors come from misclassifying $\equiv$ as $\sqsubset$, an error that will still result in correct behavior for most entailment tasks. For example, mistakenly assuming that *couch* $\sqsubset$ *sofa* instead of *couch* $\equiv$ *sofa* will still lead to correct predictions.

		N	Top 10%	Top 25%	Top 50%	All Pairs
Predicted	#	13.M	0.42	0.44	0.40	0.34
	$\equiv$	3.1M	0.89	0.74	0.73	0.67
	$\sqsubset$	6.4M	0.40	0.32	0.30	0.17
	$\neg$	430	1.00	0.90	0.84	0.82
	$\sim$	1.2M	0.29	0.24	0.24	0.20

Table 3: Precision for predicted pairs, at varying confidence cutoffs. N is the total number of unique pairs in the database predicted for each relation. Top 10% refers to the 10% of pairs for which the classifier predicted the given relation with the highest confidence. Note that these precisions reflect only lexical and phrasal relations, not syntactic paraphrase rules.

Classifying syntactic paraphrase rules In addition to lexical and phrasal paraphrase rules, PPDB contains millions of syntactic paraphrase rules which contain nonterminal symbols, e.g. *the* NP_1 *of the* NP_2 / *the* NP_2*'s* NP_1. While we do predict entailment relations for each of these rules, we do so naively by applying the same process that we apply to phrasal paraphrases, i.e. treating the nonterminal symbols as though they are simply words. We acknowledge that these paraphrase rules require special treatment and we leave this for future work. We make no claims about the quality of the entailment relations predicted for the syntactic paraphrase rules, but release the predictions anyway with the warning to use at your own risk.

References

[Bannard and Callison-Burch2005] Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604.

[Clarke2009] 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.

[Ganitkevitch and Callison-Burch2014] Juri Ganitkevitch and Chris Callison-Burch. 2014. The multilingual paraphrase database. In *The 9th edition of the Language Resources and Evaluation Conference*, Reykjavik, Iceland, May. European Language Resources Association.

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

[Landis and Koch1977] J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

[Lin and Pantel2001] Dekang Lin and Patrick Pantel. 2001. DIRT – Discovery of Inference Rules from Text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328. ACM.

[Lin1998] Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774.

[Marelli et al.2014] Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. *SemEval-2014*.

[Napoles et al.2012] 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*, pages 95–100.

[Snow et al.2004] Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*, volume 17, pages 1297–1304.

[Szpektor and Dagan2008] Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 849–856.

[Weeds et al.2004] Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04.