PPDB 2.0: Supplementary Material

1 Full list of features distributed with PPDB 2.0

Features are listed alphabetically. Bold indicates that the feature is new in the 2.0 release of PPDB.

- Abstract – a binary feature that indicates whether the rule is composed exclusively of nonterminal symbols.
- Adjacent – a binary feature that indicates whether rule contains adjacent nonterminal symbols.
- AGigaSim – the distributional similarity of e_1 and e_2, computed according to contexts observed in the Annotated Gigaword corpus (Napoles et al., 2012).
- CharCountDiff – a feature that calculates the difference in the number of characters between the phrase and the paraphrase. This feature is used for the sentence compression experiments described in Napoles et al. (2011).
- CharLogCR – the log-compression ratio in characters, $\log \frac{chars(f_2)}{chars(f_1)}$, another feature used in sentence compression.
- **ComplexityDiff** – the difference in complexity between e_1 and e_2, according to the method described in Pavlick and Nenkova (2015). Not present for every pair. Positive value implies that e_1 is more complex than e_2, negative that e_1 is simpler than e_2.
- ContainsX – a binary feature that indicates whether the nonterminal symbol X is used in this rule. X is the symbol used in Hiero grammars (Chiang, 2007), and is sometimes used by our syntactic SCFGs when we are unable to assign a linguistically motivated nonterminal.
- **Equivalence** – predicted probability that the paraphrase pair represents semantic equivalence (e_1 entails e_2 and e_2 entails e_1), according to model used in Pavlick et al. (2015).
- **Exclusion** – predicted probability that the paraphrase pair represents semantic exclusion.
- FirstAppearsIn[S|M|L|XL|XXL|XXXL] – binary feature indicating the PPDB 1.0 size (S through XXXL) where the paraphrase pair first appears. Only one feature (e.g. one size) will be present for each pair.
- **FormalityDiff** – the difference in formality between e_1 and e_2. Not present for every pair. Positive value implies that e_1 is more formal than e_2, negative that e_1 is more casual than e_2.
- **ForwardEntailment** – predicted probability that the paraphrase pair represents forward entailment (e_1 entails e_2). Either this feature or the ReverseEntailment feature will be present, but not both.
- GlueRule – a binary feature that indicates whether this is a glue rule. Glue rules are treated specially by the Joshua decoder (Post et al., 2013). They are used when the decoder cannot produce a complete parse using the other grammar rules.
- GoogleNgramSim – the distributional similarity of e_1 and e_2, computed according to contexts observed in the Google Ngram corpus (Brants and Franz, 2006).
- Identity – a binary feature that indicates whether the phrase is identical to the paraphrase.

- **Independent** – predicted probability that the paraphrase pair represents semantic independence.
- Lex($e_2|e_1$) – the "lexical translation" probability of the paraphrase given the original phrase. This feature is estimated as defined by Koehn et al. (2003)
- Lex($e_1|e_2$) – the lexical translation probability of phrase given the paraphrase.
- Lexical – a binary feature that says whether this is a single word paraphrase.
- LogCount – the log of the frequency estimate for this paraphrase pair.
- Monotonic – a binary feature that indicates whether multiple nonterminal symbols occur in the same order (are monotonic) or if they are re-ordered.
- **MVLSASim** – Cosine similarity according to the Multiview Latent Semantic Analysis embeddings described by Rastogi et al. (2015).
- **OtherRelated** – predicted probability that the paraphrase pair represents topical relatedness but not entailment. In terms of strict entailment, this can be treated the same as Independent, but pairs in the OtherRelated class are predicted to be more semantically similar than pairs in the Independent class.
- PhrasePenalty – this feature is used by the decoder to count how many rules it uses in a derivation. Turning helps it to learn to prefer fewer longer phrases, or more shorter phrases. The value of this feature is always 1.
- **PPDB1.0Score** – the score used to rank paraphrases in the original release of PPDB, computed according to the heurisitic weighting given in the paper.
- RarityPenalty – this feature marks rules that have only been seen a handful of times. It is calculated as $\exp(1 - c(e_1, e_2))$, where $c(e_1, e_2)$ is the estimate of the frequency of this paraphrase pair.
- **ReverseEntailment** – predicted probability that the paraphrase pair represents reverse entailment (e_2 entails e_1). Either this feature or the ForwardEntailment feature will be present, but not both.
- SourceTerminalsButNoTarget – a binary feature that fires when the phrase contains terminal symbols, but the paraphrase contains no terminal symbols.
- **SourceComplexity** – the complexity score for e_1 according to the method described in Pavlick and Nenkova (2015). Not present for every pair. Higher numbers indicate more complex phrases, lower indicate simpler phrases.
- **SourceFormality** – the formality score for e_1 according to the same method. Not present for every pair. Higher numbers indicate more formal phrases, lower indicate more casual phrases.
- SourceWords – the number of words in the original phrase.
- TargetTerminalsButNoSource – a binary feature that fires when the paraphrase contains terminal symbols but the original phrase only contains nonterminal symbols.
- TargetWords – the number of words in the paraphrase.
- **TargetComplexity** – the complexity score for e_2. Not present for every pair.
- **TargetFormality** – the formality score for e_2. Not present for every pair.
- UnalignedSource – a binary feature that fires if there are any words in the original phrase that are not aligned to any words in the paraphrase.
- UnalignedTarget – a binary feature that fires if there are any words in the paraphrase that are not aligned to any words in the original phrase.
- WordCountDiff – the difference in the number of words in the original phrase and the paraphrase. This feature is used for our sentence compression experiments.
- WordLenDiff – the difference in average word length between the original phrase and

the paraphrase. This feature is useful for text compression and simplification experiments.

- WordLogCR – the log-compression ratio in words, estimated as log words(e) words(f). This feature is used for our sentence compression experiments.
- $p(LHS|e_2)$ – the (negative log) probability of the lefthand side nonterminal symbol given the paraphrase.
- $p(LHS|e_1)$ – the (negative log) probability of the lefthand side nonterminal symbol given the original phrase.
- $p(e_2|LHS)$ – the (negative log) probability of the paraphrase given the lefthand side nonterminal symbol (this is typically a very low probability).
- $p(e_2|e_1)$ – the paraphrase probability of the paraphrase given the original phrase, as defined by Bannard and Callison-Burch (2005). This is given as a negative log value.
- $p(e_2|e_1, LHS)$ – the (negative log) probability of paraphrase given the the lefthand side nonterminal symbol and the original phrase.
- $p(e_1|LHS)$ – the (negative log) probability of original phrase given the the lefthand side nonterminal (this is typically a very low probability).
- $p(e_1|e_2)$ – the paraphrase probability of the original phrase given the paraphrase, as defined by Bannard and Callison-Burch (2005). This is given as a negative log value.
- $p(e_1|e_2, LHS)$ – the (negative log) probability of original phrase given the the lefthand side nonterminal symbol and the paraphrase.

2 Data Annotation

This section describes the method we used for sampling and labeling training data for our scoring model.

Sampling We take a random sample of paraphrase pairs from PPDB to be scored manually. We attempt to sample evenly across syntactic categories and across paraphrase qualities. Specifically, we collect a sample of pairs $\langle e_1, e_2 \rangle$ as follows. First, we take a random 50 e_1s from each syntactic category from PPDB-XXXL (or as many as are in the database, if there are fewer than 50), ignoring e_1s for which there are fewer than 3 paraphrases (e_2s). Then, for each e_1, we sort the list of e_2s by the paraphrase probability $p(e_1|e_2)$, and divide the list into 10 buckets. We use the $p(e_1|e_2)$ score rather than the PPDB 1.0 of the paper so as not to bias our sample based on the current scoring. From each bucket, we sample 3 e_2s (or as many as are available), resulting in a list of up to 30 e_2s per chosen e_1. In total, this gives us a sample of 26,455 $\langle e_1, e_2 \rangle$ pairs to be labeled.

Manual judgements We collect judgements on Mechanical Turk using the 1 to 5 rating described in Callison-Burch (2008) and summarized in Table 1. Each pair was judged out of context and Turkers were asked to consider the meaning only, not the grammaticality of the paraphrase in relation to the phrase. Each pair was judged by 5 independent workers and we take the average of their scores as the true score for each pair. Table 2 shows examples of pairs with varying quality levels according to the average MTurk rating.

We embedded quality control questions in each HIT, using WordNet synonyms as a gold standard example of good paraphrases (for which we expected a rating of 4 or 5) and random word pairs as gold standard examples of bad paraphrases (for which we expected a rating of 1 or 2). We rejected workers who fell below 40% accuracy on our controls after 10 or more HITs. Overall, workers averaged 71% accuracy on our controls. We use Spearman correlation to measure agreement: the average ρ between two workers was 0.57 and the average correlation of each worker with the mean of the other 4 was 0.65.

3 Features used to train PPDB 2.0 ranking model

Below we list the features in the supervised model used to predict the PPDB 2.0 paraphrase scores. The model used was a ridge regression, with parameter settings tuned using cross validation on the 26k training pairs described above.

Lexical features We compute the following string similarity features: Levenstein distance, Jaccard distance, Hamming distance, whether e_1 is a substring of e_2, whether e_2 is a substring of e_1, the number of words in e_1, the number of words in e_2, the number of words shared by e_1 and e_2.

5	Perfect	All of the meaning of the first phrase is retained in the second, and nothing is added.
4	Minor differences	The meaning of the first phrase is retained in the second, although some minor information may be added, or deleted.
3	Moderate differences	Some of the meaning of the first phrase is retained in the second, although a non-trivial amount of additional information was added or deleted.
2	Substantially different	Substantial amount of the meaning is different.
1	Completely different	The second phrase doesn't mean anything close to the first phrase.

Table 1: Paraphrase scoring metric shown to annotators on MTurk.

5	4	3	2	1
about 10/roughly 10	afflicts/effects	uncivilized/dirty	advise/guess	thank you, yes/match?
gladness/joy	what then/what now	drafting/preprocessing	preferably/ever	should/protect
5,000.00/5000	redifining/restating	telescope/binoculars	just now/doing what	sweet/uh
65 and older/65 or older	prepare/create	smithereens/parts	how hard/how angry	disqualified/engulfed
km/kilometers	of vacation/off work	our age/this time	8,600/11,600	$/march

Table 2: Example pairs at each quality level, according to the average of 5 ratings assigned by annotators on MTurk.

We compute all of these features for five versions of the strings: 1) e_1 and e_2 as is, 2) e_1 and e_2 with punctuation removed and written numbers replaced by numerals (e.g. *two* $\rightarrow$ *2*) 3) e_1 and e_2 with spaces, punctuation, and numbers removed, 4) e_1 and e_2 with words replaced by POS tags and 5) e_1 and e_2 with words replaced by their sounded codes. (Soundex is an algorithm for mapping words to codes so that words can be compared based on pronunciation rather than spelling. These features were shown to be useful for paraphrase ranking in Malakasiotis and Androutsopoulos (2011)). We also include an explicit indicator feature for when there are numbers in e_1 and e_2 and the numbers are not equivalent.

MVLSA Embeddings features We include the cosine similarity of e_1 and e_2 according to Rastogi et al. (2015)'s vector embeddings based on Multiview LSA. In cases when e_1 or e_2 is longer than a single word, we approximate the vector for the phrase with the vector of the rarest word in the phrase, using the word counts in the Google Ngram corpus.

PPDB features We use all of the features included with PPDB 1.0. This includes all of the non-bold features listed in Section 1. We also include the PPDB 1.0 score a feature. We include the following "meta-PPDB" features:

- For each paraphrase pair $\langle e_1, e_2 \rangle$, we include the log count of the number of phrase pairs $\langle E_1, E_2 \rangle$ that exist in PPDB such that e_1 is a substring of E_1 and e_2 is a substring of E_2.
- For each paraphrase pair $\langle e_1, e_2 \rangle$ and each pair of lengths l_1 and l_2 such that $l_1 \leq len(e_1)$ and $l_2 \leq len(e_2)$, we include a binary indicator for whether there exists a phrase pair $\langle \epsilon_1, \epsilon_2 \rangle$ in PPDB such that ϵ_1 is a substring of e_1 and ϵ_2 is a substring of e_2 and $len(\epsilon_1) = l_1$ and $len(\epsilon_2) = l_2$. We also include the total number of such $\langle \epsilon_1, \epsilon_2 \rangle$ pairs, as well as the length of the longest such ϵ.

Polarity features We include explicit indicators for the following words, prefixes and suffixes which are indicative of negation or of comparitives/superlatives: *-est, -er, im-, no, non-, not, -nt, -n't, un-*. We also include indicators when there is a word pair in e_1 that exhibits a strong positive/negative polarity according to Feng et al. (2013) and an equivalently strong positive/negative does not also appear in e_2.

Syntax features We include the syntactic category (LHS) assigned to the pair in PPDB. We also include a coarse-grained syntactic category, in which we use only the first letter of the true syntactic category. We include the POS tags of the unigrams in e_1 and in e_2 as a bag of words.

Translation features We derive features from the foreign pivot words (f) used to extract each $\langle e_1, e_2 \rangle$ pair. For each $\langle e_1, e_2 \rangle$ and each of the 24 languages l in our bitext, 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.

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

Unigram features We include the unigrams present in both e_1 and e_2 as a bag of words. We also include lemmatized unigrams and normalized unigrams in which punctuation is removed and written numbers are replaced by the corresponding numerals (e.g. *two* $\rightarrow$ *2*).

WordNet features For pairs in which both e_1 and e_2 appear in WordNet, we include binary indicator features for the relation assigned to the pair by WordNet. These features do not apply to most of the phrasal paraphrases.

References

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.

Thorsten Brants and Alex Franz. 2006. The google web 1t 5-gram corpus version 1.1. *LDC2006T13*.

Chris Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *EMNLP*, pages 196–205. Association for Computational Linguistics.

David Chiang. 2007. Hierarchical phrase-based translation. *computational linguistics*, 33(2):201–228.

Song Feng, Jun Sak Kang, Polina Kuznetsova, and Yejin Choi. 2013. Connotation lexicon: A dash of sentiment beneath the surface meaning. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Sofia, Bulgaria, Angust. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

Prodromos Malakasiotis and Ion Androutsopoulos. 2011. A generate and rank approach to sentence paraphrasing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 96–106. Association for Computational Linguistics.

Courtney Napoles, Chris Callison-Burch, Juri Ganitkevitch, and Benjamin Van Durme. 2011. Paraphrastic sentence compression with a character-based metric: Tightening without deletion. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 84–90. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Ellie Pavlick and Ani Nenkova. 2015. Inducing lexical style properties for paraphrase and genre differentiation. In *Proceedings of NAACL*.

Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Benjamin Van Durme, and Chris Callison-Burch. 2015. Adding semantics to data-driven paraphrasing. In *ACL*, pages 891–896.

Matt Post, Juri Ganitkevitch, Luke Orland, Jonathan Weese, Yuan Cao, and Chris Callison-Burch. 2013. Joshua 5.0: Sparser, better, faster, server. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 206–212.

Pushpendre Rastogi, Benjamin Van Durme, and Raman Arora. 2015. Multiview LSA: Representation learning via generalized CCA. In *NAACL*.