Proceedings of the Society for Computation in Linguistics

Volume 2 Article 13

2019

Jabberwocky Parsing: Dependency Parsing with Lexical Noise

Jungo Kasai
University of Washington, jkasai@cs.washington.edu

Robert Frank
Yale University, robert.frank@yale.edu

Jabberwocky Parsing: Dependency Parsing with Lexical Noise

Jungo Kasai *
University of Washington
jkasai@cs.washington.edu

Robert Frank
Yale University
robert.frank@yale.edu

Abstract

Parsing models have long benefited from the use of lexical information, and indeed current state-of-the art neural network models for dependency parsing achieve substantial improvements by benefiting from distributed representations of lexical information. At the same time, humans can easily parse sentences with unknown or even novel words, as in Lewis Carroll's poem *Jabberwocky*. In this paper, we carry out *jabberwocky* parsing experiments, exploring how robust a state-of-the-art neural network parser is to the absence of lexical information. We find that current parsing models, at least under usual training regimens, are in fact overly dependent on lexical information, and perform badly in the jabberwocky context. We also demonstrate that the technique of word dropout drastically improves parsing robustness in this setting, and also leads to significant improvements in out-of-domain parsing.

1 Introduction

Since the earliest days of statistical parsing, lexical information has played a major role (Collins, 1996, 1999; Charniak, 2000). While some of the performance gains that had been derived from lexicalization can be gotten in other ways (Klein and Manning, 2003), thereby avoiding increases in model complexity and problems in data sparsity (Fong and Berwick, 2008), recent neural network models of parsing across a range of formalisms continue to use lexical information to guide parsing decisions (constituent parsing Dyer et al. (2016)); dependency parsing: Chen and Manning (2014); Kiperwasser and Goldberg (2016); Dozat and Manning (2017); CCG parsing: Ambati et al. (2016); TAG parsing: Kasai et al. (2018); Shi and Lee (2018)). These models exploit lexical information in a way that avoids some of the data sparsity issues, by making use of distributed representations (i.e., word embeddings) that support generalization across different words.

While humans certainly make use of lexical information in sentence processing (MacDonald et al., 1994; Trueswell and Tanenhaus, 1994), it is also clear that we are able to analyze sentences in the absence of known words. This can be seen most readily by our ability to understand Lewis Carroll's poem, *Jabberwocky* (Carroll, 1883), in which open class items are replaced by non-words.

> Twas brillig, and the slithy toves
> Did gyre and gimble in the wabe;
> All mimsy were the borogoves,
> And the mome raths outgrabe

Work in neurolinguistics and psycholinguistics has demonstrated the human capacity for unlexicalized parsing experimentally, showing that humans can analyze syntactic structure even in presence of pseudo-words (Stromswold et al., 1996; Friederici et al., 2000; Kharkwal, 2014).

The word embeddings used by current lexicalized parsers are of no help in sentences with nonce words. Yet, it is at present unknown the degree to which these parsers are dependent on the information contained in these embeddings. Parsing evaluation on such nonce sentences is, therefore, critical to bridge the gap between cognitive models and data-driven machine learning models in sentence processing. Moreover, understanding the degree to which parsers are dependent upon lexical information is also of practical importance. It is advantageous for a syntactic parser to generalize well across different domains. Yet, heavy reliance upon lexical information could have detrimental effects on out-of-domain parsing because

*Work done at Yale University.

Proceedings of the Society for Computation in Linguistics (SCiL) 2019, pages 113-123.
New York City, New York, January 3-6, 2019

lexical input will carry genre-specific information (Gildea, 2001).

In this paper, we investigate the contribution of lexical information (via distributed lexical representations) by focusing on a state-of-the-art graph-based dependency parsing model (Dozat and Manning, 2017) in a series of controlled experiments. Concretely, we simulate jabberwocky parsing by adding noise to the representation of words in the input and observe how parsing performance varies. We test two types of noise: one in which words are replaced with an out-of-vocabulary word without a lexical representation, and a second in which words are replaced with others (with associated lexical representations) that match in their Penn TreeBank (PTB)-style fine-grained part of speech. The second approach is similar to the method that Gulordava et al. (2018) propose to assess syntactic generalization in LSTM language models.

In both cases, we find that the performance of the state-of-the-art graph parser dramatically suffers from the noise. In fact, we show that the performance of a lexicalized graph-based parser is substantialy worse than an unlexicalized graph-based parser in the presence of lexical noise, even when the lexical content of frequent or function words is preserved. This dependence on lexical information presents a severe challenge when applying the parser to a different domain or heterogeneous data, and we will demonstrate that indeed parsers trained on the PTB WSJ corpus achieve much lower performance on the Brown corpus.

On the positive side, we find that word dropout (Iyyer et al., 2015), applied more aggressively than is commonly done (Kiperwasser and Goldberg, 2016; de Lhoneux et al., 2017; Nguyen et al., 2017; Ji et al., 2017; Dozat and Manning, 2017; Bhat et al., 2017; Peng et al., 2017, 2018), remedies the susceptibility to lexical noise. Furthermore, our results show that models trained on the PTB WSJ corpus with word dropout significantly outperform those trained without word dropout in parsing the out-of-domain Brown corpus, confirming the practical significance of jabberwocky parsing experiments.

2 Parsing Models

Here we focus ourselves on a graph-based parser with deep biaffine attention (Dozat and Manning, 2017), a state-of-the-art graph-based parsing model, and assess its ability to generalize over lexical noise.

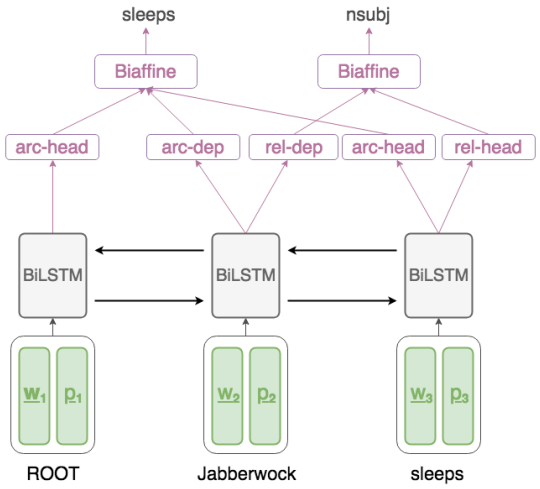

Figure 1: Biaffine parsing architecture. W and p denote the word and POS embeddings.

Input Representations The input for each word is the concatenation of a 100-dimensional embedding of the word and a 25-dimensional embedding of the PTB part of speech (POS). We initialize all word embeddings to be a zero vector and the out-of-vocabulary word is also mapped to a zero vector in testing. The POS embeddings are randomly initialized. We do not use any pretrained word embeddings throughout our experiments in order to encourage the model to find abstractions over the POS embeddings. Importantly, PTB POS categories also encode morphological features that should be accessible in jabberwocky situations. We also conducted experiments by taking as input words, universal POS tags, and character CNNs (Ma and Hovy, 2016). We observed similar patterns throughout the experiments. While those approaches can more easily scale to other languages, one concern is that the character CNNs can encode the identity of short words along side their morphological properties, and therefore would not achieve a pure jabberwocky situation. For this reason, we only present results using fine-grained POS.

Biaffine Parser Figure 1 shows our biaffine parsing architecture. Following Dozat and Manning (2017) and Kiperwasser and Goldberg (2016), we use BiLSTMs to obtain features for each word in a sentence. We first perform unla-

beled arc-factored scoring using the final output vectors from the BiLSTMs, and then label the resulting arcs. Specifically, suppose that we score edges coming into the ith word in a sentence i.e. assigning scores to the potential parents of the ith word. Denote the final output vector from the BiLSTM for the kth word by h_k and suppose that h_k is d-dimensional. Then, we produce two vectors from two separate multilayer perceptrons (MLPs) with the ReLU activation:

$$h_k^{\text{arc-dep}} = \text{MLP}^{(\text{arc-dep})}(h_k)$$
$$h_k^{\text{arc-head}} = \text{MLP}^{(\text{arc-head})}(h_k)$$

where $h_k^{\text{arc-dep}}$ and $h_k^{\text{arc-head}}$ are d_{arc}-dimensional vectors that represent the kth word as a dependent and a head respectively. Now, suppose the kth row of matrix $H^{(\text{arc-head})}$ is $h_k^{\text{arc-head}}$. Then, the probability distribution s_i over the potential heads of the ith word is computed by

$$s_i = \text{softmax}(H^{(\text{arc-head})} W^{(\text{arc})} h_i^{\text{arc-dep}} + H^{(\text{arc-head})} b^{(\text{arc})}) \quad (1)$$

where $W^{(\text{arc})} \in \mathbb{R}^{d_{arc} \times d_{arc}}$ and $b^{(arc)} \in \mathbb{R}^{d_{arc}}$. In training, we simply take the greedy maximum probability to predict the parent of each word. In the testing phase, we use the heuristics formulated by Dozat and Manning (2017) to ensure that the resulting parse is single-rooted and acyclic.

Given the head prediction of each word in the sentence, we assign labeling scores using vectors obtained from two additional MLP with ReLU. For the kth word, we obtain:

$$h_k^{\text{rel-dep}} = \text{MLP}^{(\text{rel-dep})}(h_k)$$
$$h_k^{\text{rel-head}} = \text{MLP}^{(\text{rel-head})}(h_k)$$

where $h_k^{\text{rel-dep}}, h_k^{\text{rel-head}} \in \mathbb{R}^{d_{rel}}$. Let p_i be the index of the predicted head of the ith word, and r be the number of dependency relations in the dataset. Then, the probability distribution ℓ_i over the possible dependency relations of the arc pointing from the p_ith word to the ith word is calculated by:

$$\ell_i = \text{softmax}(h_{p_i}^{T(\text{rel-head})} U^{(\text{rel})} h_i^{(\text{rel-dep})} + W^{(\text{rel})}(h_i^{(\text{rel-head})} + h_{p_i}^{(\text{rel-head})}) + b^{(\text{rel})}) \quad (2)$$

where $U^{(\text{rel})} \in \mathbb{R}^{d_{rel} \times d_{rel} \times r}$, $W^{(\text{rel})} \in \mathbb{R}^{r \times d_{rel}}$, and $b^{(\text{rel})} \in \mathbb{R}^r$.

We generally follow the hyperparameters chosen in Dozat and Manning (2017). Specifically, we use BiLSTMs layers with 400 units each. Input, layer-to-layer, and recurrent dropout rates are all 0.33. The depth of all MLPs is 1, and the MLPs for unlabeled attachment and those for labeling contain 500 (d_{arc}) and 100 (d_{rel}) units respectively. We train this model with the Adam algorithm to minimize the sum of the cross-entropy losses from head predictions (s_i from Eq. 1) and label predictions (ℓ_i from Eq. 2) with $\ell = 0.001$ and batch size 100 (Kingma and Ba, 2015). After each training epoch, we test the parser on the dev set. When labeled attachment score (LAS) does not improve on five consecutive epochs, training ends.

3 Dropout as Regularization

Dropout regularizes neural networks by randomly setting units to zero with some probability during training (Srivastava et al., 2014). In addition to usual dropout, we consider applying word dropout, a variant of dropout that targets entire words and therefore entire rows in the lexical embedding matrix (Iyyer et al., 2015). The intuition we follow here is that a trained network will be less dependent on lexical information, and more successful in a jabberwocky context, if lexical information is less reliably present during training. We consider a number of ways of word dropout.

Uniform Word Dropout Iyyer et al. (2015) introduced the regularization technique of word dropout in which lexical items are replaced by the "unknown" word with some fixed probability p and demonstrated that it improves performance for the task of text classification. Replacing words with the out-of-vocabulary word exposes the networks to out-of-vocabulary words that only occur in testing. In our experiments, we will use word dropout rates of 0.2, 0.4, 0.6, and 0.8.

Frequency-based Word Dropout Dropping words with the same probability across the vocabulary might not behave as an ideal regularizer. The network's dependence on frequent words or function words is less likely to lead to overfitting on the training data or corpus-specific properties, as the distribution of such words is less variable across different corpora. To avoid penalizing the networks for utilizing lexical information (in the form of word embeddings) for frequent words, Kiperwasser and Goldberg (2016) propose that word dropout should be applied to a word with a probability inversely proportional to the word's

frequency. Specifically, they drop out each word w that appears $\#(w)$ times in the training data with probability:

$$p_w = \frac{\alpha}{\#(w) + \alpha} \quad (3)$$

Kiperwasser and Goldberg (2016) set $\alpha = 0.25$, which leads to relatively little word dropout. In our WSJ training data, $\alpha = 0.25$ yields an expected word dropout rate of 0.009 in training, an order of magnitude less than commonly used rates in uniform word dropout. We experimented with $\alpha = 0.25, 1, 40, 352, 2536$ where the last three values yield expected word dropout rates of 0.2, 0.4, and 0.6 (the uniform dropout rates we consider). In fact, we will confirm that α needs to be much larger to significantly improve robustness to lexical noise.

Open Class Word Dropout The frequency-based word dropout scheme punishes the model less for relying upon frequent words in the training data. However, some words may occur frequently in the training data because of corpus-specific properties of the data. For instance, in the PTB WSJ training data, the word "company" is the 40th most frequent word. If our aim is to construct a parser that can perform well in different domains or across heterogeneous data, the networks should not depend upon such corpus-specific word senses. Hence, we propose to apply word dropout only on open class (non-function) words with a certain probability. We experimented with open class word dropout rates of 0.38 and 0.75 (where open class words are zeroed out 38% or 75% of the time), corresponding to the expected overall dropout rates of 0.2 and 0.4 respectively. To identify open class words in the data we used the following criteria. We consider a word as an open class word if and only if: 1) the gold UPOS is "NOUN", "PROPN", "NUM", "ADJ", or "ADV", or 2) the gold UPOS is "VERB" and the the gold XPOS (PTB POS) is not "MD" and the lemma is not "'be", "have", or "do".

4 Experiments

We test trained parsers on input that contains two types of lexical noise, designed to assess their ability to abstract away from idiosyncratic/collocational properties of lexical items: 1) *colorless green* noise and 2) *jabberwocky* noise. The former randomly exchanges words with PTB POS preserved, and the latter zeroes outs the embeddings for words (i.e. replacing words with an out-of-vocabulary word). In either case, we keep POS input to the parsers intact.

Colorless Green Experiments Gulordava et al. (2018) propose a framework to evaluate the generalization ability of LSTM language models that abstracts away from idiosyncratic properties of words or collocational information. In particular, they generate nonce sentences by randomly replacing words in the original sentences while preserving part-of-speech and morphological features. This can be thought of as a computational approach to producing sentences that are "grammatical" yet meaningless, exemplified by the famous example "colorless green ideas sleep furiously" (Chomsky, 1957). Concretely, for each PTB POS category, we pick the 50 most frequent words of that category in the training set and replace each word w in the test set by a word uniformly drawn from the 50 most frequent words for w's POS category. We consider three situations: 1) full *colorless green* experiments where all words are replaced by random words, 2) top 100 *colorless green* experiments where all words but the 100 most frequent words are replaced by random words, and 3) open class *colorless green* experiments where the input word is replaced by a random word if and only if the word is an open class word.[1]

Jabberwocky Experiments One potential shortcoming with the approach above is that it produces sentences which might violate constraints that are imposed by specific lexical items, but which are not represented by the POS category. For instance, this approach could generate a sentence like "it stays the shuttle" in which the intransitive verb "stay" takes an object (Gulordava et al., 2018).[2] Such a violation of argument structure constraints could mislead parsers (as well as language models studied in Gulordava et al. (2018)) and we will show that is indeed the

[1] We use the same criteria for open class words as in the open class word dropout.

[2] This shortcoming might be overcome by using lexical resources like PropBank (Palmer et al., 2005) or NomBank (Meyers et al., 2004) to guide word substitutions. In this paper, we do not do this, follow Gulordava et al.'s approach for the creation of colorless green sentences. We instead use the jabberwocky manipulation to avoid creating sentences that violate selectional constraints.

case.[3] To address this issue, we also experiment with *jabberwocky* noise, in which input word vectors are zeroed out. This noise is equivalent to replacing words with an out-of-vocabulary word by construction. Because fine-grained POS information is retained in the input to the parser, the parser is still able to benefit from the kind of morphological information present in Carroll's poem. We again consider three situations 1) full jabberwocky experiments where all word embeddings are zeroed out, 2) top 100 jabberwocky experiments where word embeddings for all but the most frequent 100 words are zeroed out, and 3) open class jabberwocky experiments where the input word vector is zeroed out if and only if the word is an open class word. Open class jabberwocky experiments are the closest to the situation when humans read Lewis Carroll's *Jabberwocky*.[4]

Out-of-domain experiments We also explore a practical aspect of our experiments with lexical noise. We apply our parsers that are trained on the WSJ corpus to the Brown corpus and observe how parsers with various configurations perform.[5] Prior work showed that parsers trained on WSJ yield degraded performance on the Brown corpus (Gildea, 2001) despite the fact that the average sentence length is shorter in the Brown corpus (23.85 tokens for WSJ; 20.56 for Brown). We show that robustness to lexical noise improves out-of-domain parsing.

Baseline Parsers Lexical information is clearly useful for certain parsing decisions, such as PP-attachment. As a result, a lexicalized parser clearly should make use of such information when it is available, and may well perform less well when it is not. In fact, in *jabberwocky* and *colorless green* settings, the absence of lexical information may lead to an underdetermination of the parse by the POS or word sequence, so that there is no non-arbitrary "gold standard" parse. As a result, simply observing a performance drop of a parser in the face of lexical noise does not help to establish an appropriate baseline with respect to how well a parser can be expected to perform in a lexically noisy setting. We propose three baseline parsers: 1) an unlexicalized parser where the network input is only POS tags, 2) a "top 100" parser where the network input is only POS tags and lexical information for the 100 most frequent words and 3) a "function word" parser where the network input is only POS tags and lexical information for function words. Each baseline parser can be thought of as specialized to the corresponding *colorless green* and *jabberwocky* experiments. For example, the unlexicalized parser gives us an upper bound for full *colorless green* and *jabberwocky* experiments because the parser is ideally adapted to the unlexicalized situation, as it has no dependence on lexical information.

Experimental Setup We use Universal Dependency representations obtained from converting the Penn Treebank (Marcus et al., 1993) using Stanford CoreNLP (ver. 3.8.0) (Manning et al., 2014). We follow the standard data split: sections 2-21, 22, and 23 for training, dev, and test sets respectively. For the out-of-domain experiments, we converted the Brown corpus in PTB again using Stanford CoreNLP into Universal Dependency representations.[6]

We only use gold POS tags in training for simplicity,[7] but we conduct experiments with both gold and predicted POS tags. Experiments with gold POS tags allow us to isolate the effect of lexical noise from POS tagging errors, while those with predicted POS tags simulate more practical situations where POS input is not fully reliable. Somewhat surprisingly, however, we find that relative performance patterns do not change even when using predicted POS tags. All pre-

[3] Prior work in psycholinguistics argued that verbs can in fact be used in novel argument structure constructions, and assigned coherent interpretations on the fly (Johnson and Goldberg, 2013). Our colorless green parsing experiments can be interpreted as a simulation for such situations.

[4] An anonymous reviewer notes that because of its greater complexity, human performance on a jabberwocky version of the WSJ corpus may not be at the level we find when reading the sentences of Lewis Carroll's poem or in the psycholinguistic work that has explored human ability to process jabberwocky-like sentences. We leave it for future work to explore whether human performance in such complex cases is indeed qualitatively different, and also whether the pattern of results changes if we restrict our focus to a syntactically simpler corpus, given a suitable notion of simplicity.

[5] We initially intended to apply our trained parsers to the Universal Dependency corpus (Nivre et al., 2015) as well for out-of-domain experiments, but we found annotation inconsistency and the problem of conversion from phrase structures to universal dependencies. We leave this problem for future.

[6] We exclude the domains of CL and CP in the Brown corpus because the Stanford CoreNLP converter encountered an error.

[7] One could achieve better results by training a parser on predicted POS tags obtained from jackknife training, but improving normal parsing performance is not the focus of our work.

dicted POS tags are obtained from a BiLSTM POS tagger with character CNNs, trained on the same training data (sections 2-21) with hyperparameters from Ma and Hovy (2016) and word embeddings initialized with GloVe vectors (Pennington et al., 2014). We train 5 parsing models for each training configuration with 5 different random initializations and report the mean and standard deviation.[8] We use the CoNLL 2017 official script for evaluation (Zeman et al., 2017).

5 Results and Discussions

Normal Parsing Results Table 1 shows normal parsing results on the dev set. In both gold and predicted POS experiments, we see a significant discrepancy between the performance in rows 2-4 and the rest, suggesting that lexicalization of a dependency parser greatly contributes to parsing performance; having access to the most frequent 100 words (row 3) or the function words (row 4) recovers part of the performance drop from unlexicalization (row 2), but the LAS differences from complete lexicalization (row 1, row 5 and below) are still significant. For each of the three word dropout schemes in gold POS experiments, we see a common pattern: performance improves up to a certain degree of word dropout (Uniform 0.2, Frquency-based 1-40, Open Class 0.38), and it drops after as word dropout becomes more aggressive. This suggests that word dropout also involves the bias-variance trade-off. Although performance generally degrades with predicted POS tags, the patterns of relative performance still hold. Again, for each of the three dropout schemes, there is a certain point in the spectrum of word dropout intensity that achieves the best performance, and such points are almost the same both in the models trained with gold and predicted POS tags. This is a little surprising because a higher word dropout rate encourages the model to rely more on POS input, and noisy POS information from the POS tagger can work against the model. Indeed, we observed this parallelism between experiments with gold and predicted POS tags consistently throughout the *colorless green* and *jabberwocky* experiments, and therefore we only report results with gold POS tags for the rest of the *colorless green* and *jabberwocky* experiments for simplicity.

[8] Our code is available at https://github.com/jungokasai/graph_parser for easy replication.

Model	Gold		Predicted	
	UAS	LAS	UAS	LAS
No Dropout	$93.6_{0.2}$	$92.3_{0.2}$	$92.7_{0.1}$	$90.6_{0.1}$
Unlexicalized	$88.0_{0.1}$	$85.4_{0.1}$	$87.1_{0.1}$	$83.8_{0.1}$
Top 100	$92.5_{0.1}$	$90.8_{0.1}$	$91.7_{0.1}$	$89.2_{0.1}$
Function	$90.7_{0.4}$	$88.1_{0.6}$	$90.0_{0.3}$	$86.8_{0.5}$
Uniform 0.2	$93.9_{0.1}$	$92.6_{0.1}$	$93.0_{0.1}$	$90.9_{0.2}$
Uniform 0.4	$94.0_{0.1}$	$92.5_{0.1}$	$93.0_{0.1}$	$90.8_{0.1}$
Uniform 0.6	$93.7_{0.1}$	$92.2_{0.1}$	$92.7_{0.1}$	$90.5_{0.1}$
Uniform 0.8	$93.0_{0.1}$	$91.4_{0.1}$	$92.1_{0.1}$	$89.7_{0.2}$
Freq 0.25	$93.7_{0.1}$	$92.4_{0.1}$	$92.9_{0.1}$	$90.8_{0.1}$
Freq 1	$93.9_{0.1}$	$92.6_{0.1}$	$93.0_{0.1}$	$91.0_{0.1}$
Freq 40	$94.0_{0.2}$	$92.6_{0.2}$	$93.0_{0.2}$	$90.9_{0.2}$
Freq 352	$93.6_{0.1}$	$92.2_{0.1}$	$92.7_{0.1}$	$90.5_{0.1}$
Freq 2536	$92.9_{0.1}$	$91.4_{0.1}$	$92.0_{0.1}$	$89.7_{0.1}$
Open Cl 0.38	$93.9_{0.1}$	$92.5_{0.2}$	$93.0_{0.1}$	$90.9_{0.2}$
Open Cl 0.75	$93.5_{0.1}$	$92.1_{0.1}$	$92.7_{0.1}$	$90.5_{0.1}$

Table 1: Normal Parsing Results on the Dev Set. The subscripts indicate the standard deviations.

Full Experiments Table 2 shows results for full *colorless green* and *jabberwocky* experiments. The models without word dropout yield extremely poor performance both in *colorless* and *jabberwocky* settings, suggesting that a graph-based parsing model learns to rely heavily on word information if word dropout is not performed. Here, unlike the normal parsing results, we see monotone increasing performance as word dropout is more aggressively applied, and the performance rises more dramatically. In particular, with uniform word dropout rate 0.2, full *jabberwocky* performance increases by more than 40 LAS points, suggesting the importance of the parser's exposure to unknown words to abstract away from lexical information. Frequency-based word dropout needs to be performed more aggressively ($\alpha \geq 40$) than has previously been done for dependency parsing (Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017) in order to achieve robustness to full lexical noise similar to that obtained with uniform word dropout with $p > 0.2$. Open class word dropout does not bring any benefit to parsers in the full *jabberwocky* and *colorless green* settings. This is probably because parsers trained with open class word dropout has consistent access to function words, and omitting the lexical representations of the function words is very harmful to such parsers. Interestingly, in some of the cases, *colorless green* outperforms *jabberwocky* performance, perhaps because noisy word information, even with argument constraint violations, is better than no word information.

Model	Colorless UAS	Colorless LAS	Jabberwocky UAS	Jabberwocky LAS
No Dropout	$62.6_{0.2}$	$56.3_{0.1}$	$51.9_{2.3}$	$39.1_{1.9}$
Unlexicalized	$88.0_{0.1}$	$85.4_{0.1}$	$88.0_{0.1}$	$85.4_{0.2}$
Top 100	$71.7_{0.4}$	$67.1_{0.3}$	$72.7_{0.3}$	$68.2_{0.5}$
Function	$69.2_{0.9}$	$62.1_{0.7}$	$58.8_{3.0}$	$39.8_{3.4}$
Uniform 0.2	$74.0_{0.2}$	$69.1_{0.2}$	$85.7_{0.3}$	$82.7_{0.3}$
Uniform 0.4	$76.9_{0.3}$	$72.3_{0.2}$	$87.1_{0.1}$	$84.3_{0.1}$
Uniform 0.6	$79.2_{0.2}$	$75.0_{0.2}$	$87.7_{0.1}$	$85.0_{0.1}$
Uniform 0.8	$82.0_{0.3}$	$78.5_{0.3}$	$88.0_{0.1}$	$85.4_{0.1}$
Freq 0.25	$62.9_{0.2}$	$56.4_{0.1}$	$55.0_{1.4}$	$43.4_{2.5}$
Freq 1	$63.6_{0.4}$	$57.1_{0.1}$	$60.1_{1.7}$	$48.8_{3.6}$
Freq 40	$67.5_{0.7}$	$61.6_{0.6}$	$76.4_{1.0}$	$72.0_{1.2}$
Freq 352	$74.5_{0.5}$	$69.7_{0.5}$	$82.9_{0.4}$	$79.5_{0.6}$
Freq 2536	$82.6_{0.2}$	$78.8_{0.3}$	$86.5_{0.4}$	$85.4_{0.2}$
Open Cl 0.38	$65.0_{0.5}$	$58.8_{0.4}$	$53.7_{2.6}$	$36.8_{3.6}$
Open Cl 0.75	$66.7_{0.2}$	$60.5_{0.3}$	$53.8_{1.0}$	$34.0_{1.3}$

Table 2: Full Colorless Green and Jabberwocky Experiments on the Dev Set.

Model	Colorless UAS	Colorless LAS	Jabberwocky UAS	Jabberwocky LAS
No Dropout	$85.5_{0.1}$	$82.7_{0.1}$	$86.4_{0.8}$	$83.4_{0.9}$
Unlexicalized	$88.0_{0.1}$	$85.4_{0.1}$	$88.0_{0.1}$	$85.4_{0.2}$
Top 100	$92.5_{0.1}$	$90.8_{0.1}$	$92.5_{0.1}$	$90.8_{0.1}$
Function	$88.7_{0.4}$	$85.4_{0.7}$	$90.8_{0.3}$	$88.0_{0.6}$
Uniform 0.2	$87.5_{0.2}$	$84.9_{0.2}$	$90.2_{0.6}$	$88.2_{0.6}$
Uniform 0.4	$88.5_{0.2}$	$86.0_{0.2}$	$90.8_{0.4}$	$88.9_{0.4}$
Uniform 0.6	$89.2_{0.2}$	$86.8_{0.1}$	$91.0_{0.3}$	$89.1_{0.2}$
Uniform 0.8	$89.7_{0.1}$	$87.4_{0.2}$	$90.6_{0.2}$	$88.6_{0.2}$
Freq 0.25	$85.8_{0.2}$	$83.0_{0.2}$	$87.8_{0.6}$	$85.0_{0.6}$
Freq 1	$86.1_{0.1}$	$83.3_{0.1}$	$88.9_{0.4}$	$86.3_{0.4}$
Freq 40	$88.1_{0.2}$	$85.5_{0.1}$	$90.9_{0.2}$	$88.7_{0.4}$
Freq 352	$89.7_{0.1}$	$87.4_{0.1}$	$91.9_{0.2}$	$90.0_{0.2}$
Freq 2536	$90.7_{0.2}$	$88.6_{0.3}$	$91.3_{0.2}$	$89.3_{0.3}$
Open Cl 0.38	$88.6_{0.3}$	$86.2_{0.3}$	$90.6_{0.2}$	$88.3_{0.2}$
Open Cl 0.75	$89.6_{0.1}$	$87.4_{0.2}$	$90.8_{0.3}$	$88.0_{0.6}$

Table 3: Top 100 Colorless Green and Jabberwocky Experiments on the Dev Set.

Model	Colorless UAS	Colorless LAS	Jabberwocky UAS	Jabberwocky LAS
No Dropout	$84.1_{0.3}$	$81.3_{0.2}$	$84.9_{0.6}$	$82.1_{0.6}$
Unlexicalized	$88.0_{0.1}$	$85.4_{0.1}$	$88.0_{0.1}$	$85.4_{0.2}$
Top 100	$90.5_{0.2}$	$88.4_{0.2}$	$91.8_{0.1}$	$89.9_{0.1}$
Function	$90.7_{0.4}$	$88.1_{0.6}$	$90.7_{0.4}$	$88.1_{0.6}$
Uniform 0.2	$87.4_{0.2}$	$84.8_{0.2}$	$89.7_{0.7}$	$87.8_{0.7}$
Uniform 0.4	$88.3_{0.2}$	$85.9_{0.3}$	$90.6_{0.4}$	$88.7_{0.4}$
Uniform 0.6	$89.2_{0.2}$	$86.8_{0.2}$	$90.9_{0.3}$	$89.0_{0.3}$
Uniform 0.8	$89.9_{0.2}$	$87.7_{0.1}$	$90.5_{0.3}$	$88.6_{0.2}$
Freq 0.25	$84.7_{0.2}$	$82.0_{0.2}$	$86.7_{0.5}$	$84.3_{0.6}$
Freq 1	$85.2_{0.4}$	$82.4_{0.4}$	$88.1_{0.6}$	$85.9_{0.6}$
Freq 40	$87.7_{0.2}$	$85.2_{0.2}$	$91.2_{0.3}$	$89.2_{0.3}$
Freq 352	$89.5_{0.2}$	$87.3_{0.1}$	$92.0_{0.1}$	$90.2_{0.1}$
Freq 2536	$90.7_{0.2}$	$88.7_{0.2}$	$91.7_{0.2}$	$89.9_{0.1}$
Open Cl 0.38	$89.0_{0.2}$	$86.7_{0.3}$	$92.1_{0.2}$	$90.3_{0.2}$
Open Cl 0.75	$90.7_{0.1}$	$88.4_{0.2}$	$92.4_{0.1}$	$90.7_{0.1}$

Table 4: Open Class Colorless Green and Jabberwocky Experiments on the Dev Set.

Model	Gold UAS	Gold LAS	Predicted UAS	Predicted LAS
No Dropout	$89.7_{0.1}$	$87.5_{0.1}$	$88.8_{0.1}$	$85.7_{0.1}$
Unlexicalized	$83.0_{0.1}$	$79.3_{0.2}$	$81.3_{0.1}$	$76.7_{0.1}$
Top 100	$89.4_{0.1}$	$86.8_{0.1}$	$88.5_{0.1}$	$84.9_{0.1}$
Function	$88.3_{0.3}$	$84.8_{0.7}$	$87.4_{0.3}$	$83.1_{0.7}$
Uniform 0.2	$90.0_{0.1}$	$87.8_{0.2}$	$89.1_{0.1}$	$86.0_{0.2}$
Uniform 0.4	$90.1_{0.1}$	$87.9_{0.1}$	$89.3_{0.1}$	$86.1_{0.1}$
Uniform 0.6	$90.1_{0.1}$	$87.7_{0.1}$	$89.1_{0.1}$	$85.8_{0.1}$
Uniform 0.8	$89.4_{0.1}$	$86.8_{0.1}$	$88.5_{0.1}$	$85.0_{0.1}$
Freq 0.25	$89.9_{0.2}$	$87.7_{0.2}$	$89.1_{0.2}$	$86.0_{0.2}$
Freq 1	$90.2_{0.2}$	$88.1_{0.2}$	$89.3_{0.1}$	$86.3_{0.1}$
Freq 40	$90.7_{0.2}$	$\mathbf{88.5_{0.3}}$	$89.8_{0.2}$	$\mathbf{86.7_{0.2}}$
Freq 352	$90.3_{0.1}$	$88.0_{0.1}$	$89.3_{0.1}$	$86.1_{0.1}$
Freq 2536	$89.3_{0.2}$	$86.8_{0.2}$	$88.4_{0.1}$	$85.0_{0.2}$
Open Cl 0.38	$90.3_{0.2}$	$88.1_{0.2}$	$89.4_{0.1}$	$86.2_{0.2}$
Open Cl 0.75	$90.3_{0.1}$	$88.0_{0.1}$	$87.4_{0.3}$	$86.1_{0.1}$

Table 5: Brown CF Results.

Top 100 Experiments Table 3 shows the results of top 100 *colorless green* and *jabberwocky* experiments. Performance by parsers trained without word dropout is substantially better than what is found in the full *colorless green* or *jabberwocky* settings. However, the performance is still much lower than the unlexicalized parsers (2.7 LAS points for *colorless green* and 2.0 LAS points for *jabberwocky*), meaning that the parser without word dropout has significant dependence on less frequent words. On the other hand, parsers trained with a high enough word dropout rate outperform the unlexicalized parser (e.g., uniform 0.4, frequency-based $\alpha = 40$, and open class 0.38). Frequency-based word dropout is very effective. Recall that $\alpha = 352, 2536$ correspond to the expected word dropout rates of 0.4 and 0.6. The two configurations yield better results than uniform 0.4 and 0.6.

Open Class Experiments Table 4 gives the results of open class *colorless green* and *jabberwocky* experiments. We see similar patterns to the top 100 *jabberwocky* experiments except that open class word dropout performs better and frequency-based word dropout performs worse, which naturally follows from the way we train the parsers.

Out-of-domain Parsing Table 5 reports parsing performance on the CF domain in the Brown corpus.[9] POS tagging accuracy was 96.4%, relatively low as compared to in-domain performance in WSJ. Gildea (2001) demonstrated that removing lexical information from lexicalized PCFG does not deteriorate out-of-domain parsing perfor-

[9]We found similar patterns in relative performance when applied to the other domains of the Brown corpus.

	Colorless			Jabberwocky		
relation	Pre	Rec	F1	Prec	Rec	F1
nsubj	87.8	86.2	87.0	91.0	90.0	90.5
dobj	79.2	81.4	80.3	83.0	90.7	86.7
iobj	29.4	27.8	28.6	71.4	55.6	62.5
csubj	30.4	46.7	36.8	33.3	60.0	42.9
ccomp	75.3	75.3	75.3	77.2	78.2	77.7
xcomp	69.0	69.2	69.1	78.7	64.3	70.8
others	83.6	83.6	83.6	86.7	86.6	86.6

Table 6: Performance Breakdown by Dependency Relation.

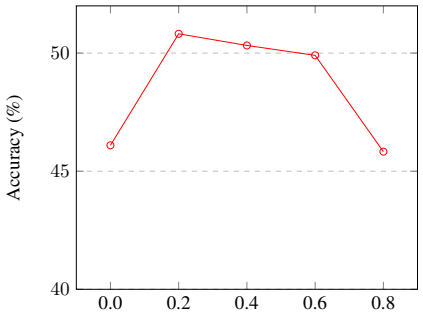

Figure 2: POS Prediction Results from the Word Embedding Induced in Varying Uniform Dropout Rates.

mance in the Brown corpus. In contrast, our results show that lexicalization of the neural network dependency parsers via distributed representations facilitates parsing performance. Most of the gains from lexicalization stem from the 100 most frequent words (row 3) and function words (row 4), but we get a significant improvement by performing relatively aggressive word dropout ($\alpha = 40$). This confirms the practical significance of *jabberwocky* parsing experiments.

6 Analysis of Results

Colorless Green vs. Jabberwocky Earlier we hypothesized that *colorless green* lexical noise could mislead the parser by violating argument structure constraints. And in fact, we saw in the previous section that *jabberwocky* parsing results are generally better than *colorless green* parsing results. In order to further test our hypothesis, we provide a breakdown of parsing performance broken down by dependency type (Nivre et al., 2015). Table 6 provides the open class performance breakdown for a parser trained with uniform word dropout rate 0.2. We observe that in the *colorless green* situation, the parser particularly suffers in "iobj" and "dobj", consistent with what we would expect for parsing sentences with argument structure violations. For example, the *colorless green* scheme does not prevent us from replacing a ditransitive verb with a non-ditransitive verb. This result validates our hypothesis, and the *colorless green* scheme is limited as a framework to purely assess parser's generalization ability.

Trained Word Embeddings The logic of our *jabberwocky* experiments and the role of word dropout involves the assumption that the parser will succeed in the absence of lexical information if it depends more heavily on the grammatical category information present in the POS embeddings. However, this fails to allow for the possibility that the word embeddings themselves might also represent POS information. It might instead be the case that what word dropout does is merely discouraging the model from constructing grammatical abstractions in the word embeddings, depending instead on the information present in the POS embeddings.

To test this possibility, we attempt to use a simple softmax regression on top of the induced word embeddings to detect the presence of grammatical category information. Concretely, we find the set of words that occur 100 times or more in the training data whose unigram probability for the most frequent POS tag is greater than 0.5. In this way, we obtain a set of pairs of a word and its most frequent POS tag, and we then randomly split the set to run 5-fold cross validation. Figure 2 shows POS prediction accuracy over varying uniform word dropout rates. These results do not support the idea that word dropout reduces the representation of grammatical category information in word embedding. On the contrary, word dropout rates of 0.2 through 0.6 leads to word embeddings that *better* represent POS.

Parser Weight Sizes What then is the effect of word dropout on the network? Figure 3 shows L2 norms of the trained POS embeddings (46 by 25) and the input/output gate weights for POS input in the first BiLSTM layer (400 by 25) across varying uniform word dropout rates. We can observe that L2 norm for each becomes larger as the word dropout rate increases. This suggests that what word dropout does is to encourage the model to make greater use of POS information.

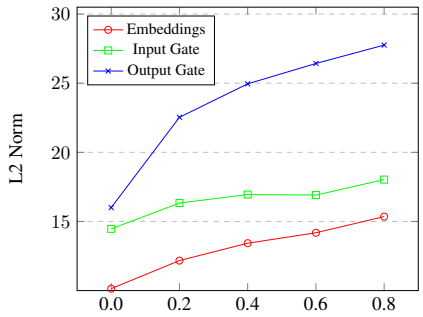

Figure 3: POS Embeddings and Weight Sizes over Uniform Word Dropout Rates.

7 Conclusion and Future Work

We conducted extensive analysis on the robustness of a state-of-the-art graph-based dependency parser against lexical noise. Our experiments showed that parsers trained under usual regimens performed poorly in the face of lexical noise. However, we demonstrated that the technique of word dropout, when applied aggressively, remedies this problem without sacrificing parsing performance. Word dropout is commonly used in literature, but our results provide further guidance about how it should be used in the future. In future work, we would like to compare our results with different parsing architectures such as transition-based parsers.

References

Bharat Ram Ambati, Tejaswini Deoskar, and Mark Steedman. 2016. Shift-reduce ccg parsing using neural network models. In *NAACL*.

Riyaz Ahmad Bhat, Irshad Ahmad Bhat, and Dipti Misra Sharma. 2017. Leveraging newswire treebanks for parsing conversational data with argument scrambling. In *IWPT*.

Lewis Carroll. 1883. *Through the Looking-Glass*. Macmillan and Co., New York.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *ANLP*.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton & Co., Berlin, Germany.

Michael Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *ACL*.

Michael Collins. 1999. . *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Timothy Dozat and Christopher Manning. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR*.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *NAACL*.

Sandiway Fong and Robert C. Berwick. 2008. Treebank parsing and knowledge of language : A cognitive perspective. In *CogSci*.

Angela D. Friederici, Mia Viktoria Meyer, and D. Yves von Cramon. 2000. Auditory language comprehension: an event-related fmri study on the processing of syntactic and lexical information. *Brain and language*, 75 3:289–300.

Daniel Gildea. 2001. Corpus variation and parser performance. In *EMNLP*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *NAACL*. Association for Computational Linguistics.

Mohit Iyyer, Varun Manjunatha, Jordan L. Boyd-Graber, and Hal Daumé. 2015. Deep unordered composition rivals syntactic methods for text classification. In *ACL*.

Tao Ji, Yuanbin Wu, and Man Lan. 2017. A fast and lightweight system for multilingual dependency parsing. In *CoNLL Shared Task*.

Matt A. Johnson and Adele E. Goldberg. 2013. Evidence for automatic accessing of constructional meaning: Jabberwocky sentences prime associated verbs. *Language and Cognitive Processes*, 28(10):14391452.

Jungo Kasai, Robert Frank, Pauli Xu, William Merrill, and Owen Rambow. 2018. End-to-end graph-based TAG parsing with neural networks. In *NAACL*. Association for Computational Linguistics.

Gaurav Kharkwal. 2014. *Taming the Jabberwocky: Examining Sentence Processing with Novel Words*. Ph.D. thesis, Rutgers University.

Diederik P. Kingma and Jimmy Lei Ba. 2015. ADAM: A Method for Stochastic Optimization. In *ICLR*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 4:313–327.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*.

Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. 2017. From raw text to universal dependencies - look, no tags! In *CoNLL Shared Task*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *ACL*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Maryellen C. MacDonald, Neal J. Pearlmutter, and Mark S. Seidenberg. 1994. Lexical nature of syntactic ambiguity resolution. *Psychological Review*, 101:676–703.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proceedings of the Workshop on Frontiers in Corpus Annotation at HLT-NAACL*.

Dat Quoc Nguyen, Mark Dras, and Mark Johnson. 2017. A novel neural network model for joint pos tagging and graph-based dependency parsing. In *CoNLL Shared Task*.

Joakim Nivre, Željko Agić, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Cristina Bosco, Sam Bowman, Giuseppe G. A. Celano, Miriam Connor, Marie-Catherine de Marneffe, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Tomaž Erjavec, Richárd Farkas, Jennifer Foster, Daniel Galbraith, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Berta Gonzales, Bruno Guillaume, Jan Hajič, Dag Haug, Radu Ion, Elena Irimia, Anders Johannsen, Hiroshi Kanayama, Jenna Kanerva, Simon Krek, Veronika Laippala, Alessandro Lenci, Nikola Ljubešić, Teresa Lynn, Christopher Manning, Cătălina Mărănduc, David Mareček, Héctor Martínez Alonso, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Shunsuke Mori, Hanna Nurmi, Petya Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Slav Petrov, Jussi Piitulainen, Barbara Plank, Martin Popel, Prokopis Prokopidis, Sampo Pyysalo, Loganathan Ramasamy, Rudolf Rosa, Shadi Saleh, Sebastian Schuster, Wolfgang Seeker, Mojgan Seraji, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Kiril Simov, Aaron Smith, Jan Štěpánek, Alane Suhr, Zsolt Szántó, Takaaki Tanaka, Reut Tsarfaty, Sumire Uematsu, Larraitz Uria, Viktor Varga, Veronika Vincze, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. 2015. Universal dependencies 1.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The proposition bank: A corpus annotated with semantic roles. *Computational Linguistics*, 31(1).

Hao Peng, Sam Thomson, and Noah A. Smith. 2017. Deep multitask learning for semantic dependency parsing. In *ACL*.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018. Learning joint semantic parsers from disjoint data. In *NAACL*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Tianze Shi and Lillian Lee. 2018. Valency-augmented dependency parsing. In *EMNLP*. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Karen Stromswold, David O. Caplan, Nathaniel M. Alpert, and Scott L. Rauch. 1996. Localization of syntactic comprehension by positron emission tomography. *Brain and language*, 52 3:452–73.

John C. Trueswell and Michael K. Tanenhaus. 1994. Toward a lexicalist framework for constraint-based syntactic ambiguity resolution. In Charles Clifton, Jr., Lyn Frazier, and Keith Rayner, editors, *Perspectives on Sentence Processing*, pages 155–179. Lawrence Erlbaum Associates, Hillsdale, NJ.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinkova, Jan Hajic jr., Jaroslava Hlavacova, Václava Kettnerová, Zdenka Uresova, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria dePaiva, Kira Droganova, Héctor Martínez Alonso, Çağr Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo

Mendonca, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.