Cross-Domain Generalization of Neural Constituency Parsers

Daniel Fried[*] **Nikita Kitaev**[*] **Dan Klein**
Computer Science Division
University of California, Berkeley
{dfried,kitaev,klein}@cs.berkeley.edu

Abstract

Neural parsers obtain state-of-the-art results on benchmark treebanks for constituency parsing—but to what degree do they generalize to other domains? We present three results about the generalization of neural parsers in a zero-shot setting: training on trees from one corpus and evaluating on out-of-domain corpora. First, neural and non-neural parsers generalize comparably to new domains. Second, incorporating pre-trained encoder representations into neural parsers substantially improves their performance across all domains, but does not give a larger relative improvement for out-of-domain treebanks. Finally, despite the rich input representations they learn, neural parsers still benefit from structured output prediction of output trees, yielding higher exact match accuracy and stronger generalization both to larger text spans and to out-of-domain corpora. We analyze generalization on English and Chinese corpora, and in the process obtain state-of-the-art parsing results for the Brown, Genia, and English Web treebanks.

1 Introduction

Neural constituency parsers have obtained increasingly high performance when measured by F1 scores on in-domain benchmarks, such as the Wall Street Journal (WSJ) (Marcus et al., 1993) and Penn Chinese Treebank (CTB) (Xue et al., 2005). However, in order to construct systems useful for cross-domain NLP, we seek parsers that generalize well to domains other than the ones they were trained on. While classical, non-neural parsers are known to perform better in their training domains than on out-of-domain corpora, their out-of-domain performance degrades in well-understood ways (Gildea, 2001; Petrov and Klein, 2007), and improvements in performance on in-domain treebanks still transfer to out-of-domain improvements (McClosky et al., 2006).

Is the success of neural constituency parsers (Henderson 2004; Vinyals et al. 2015; Dyer et al. 2016; Cross and Huang 2016; Choe and Charniak 2016; Stern et al. 2017; Liu and Zhang 2017; Kitaev and Klein 2018, *inter alia*) similarly transferable to out-of-domain treebanks? In this work, we focus on *zero-shot generalization*: training parsers on a single treebank (e.g. WSJ) and evaluating on a range of broad-coverage, out-of-domain treebanks (e.g. Brown (Francis and Kučera, 1979), Genia (Tateisi et al., 2005), the English Web Treebank (Petrov and McDonald, 2012)). We ask three questions about zero-shot generalization properties of state-of-the-art neural constituency parsers:

First, *do non-neural parsers have better out-of-domain generalization than neural parsers?* We might expect neural systems to generalize poorly because they are highly-parameterized, and may overfit to their training domain. We find that **neural and non-neural parsers generalize similarly**, and, encouragingly, improvements on in-domain treebanks still transfer to out-of-domain.

Second, *does pre-training particularly improve out-of-domain performance*, or does it just generally improve test accuracies? Neural parsers incorporate rich representations of language that can easily be pre-trained on large unlabeled corpora (Ling et al., 2015; Peters et al., 2018; Devlin et al., 2019) and improve accuracies in new domains (Joshi et al., 2018). Past work has shown that lexical supervision on an out-of-domain treebank can substantially improve parser performance (Rimell and Clark, 2009). Similarly, we might expect pre-trained language representations to give the largest improvements on out-of-domain treebanks, by providing representations of language disparate from the training domains. Surprisingly, however, we find that **pre-trained representations give similar error reductions across domains**.

[*] Equal contribution.

	Berkeley		BLLIP		In-Order		Chart	
	F1	Δ Err.	F1	Δ Err.	F1	Δ Err.	F1	Δ Err.
WSJ Test	90.06	+0.0%	91.48	+0.0%	91.47	+0.0%	93.27	+0.0%
Brown All	84.64	+54.5%	85.89	+65.6%	85.60	+68.9%	88.04	+77.7%
Genia All	79.11	+110.2%	79.63	+139.1%	80.31	+130.9%	82.68	+157.4%
EWT All	77.38	+127.6%	79.91	+135.8%	79.07	+145.4%	82.22	+164.2%

Table 1: Performance and relative increase in error (both given by F1) on English corpora as parsers are evaluated out-of-domain, relative to performance on the in-domain WSJ Test set. Improved performance on WSJ Test translates to improved performance out-of-domain. The two parsers with similar absolute performance on WSJ (BLLIP and In-Order) have comparable generalization out-of-domain, despite one being neural and one non-neural.

Finally, *how much does structured prediction help neural parsers?* While neural models with rich modeling of syntactic structure have obtained strong performance on parsing (Dyer et al., 2016; Liu and Zhang, 2017) and a range of related tasks (Kuncoro et al., 2018; Hale et al., 2018), recent neural parsers obtain state-of-the-art F1 on benchmark datasets using rich input encoders without any explicit modeling of correlations in output structure (Shen et al., 2018; Kitaev and Klein, 2018). Does structural modeling still improve parsing performance even with these strong encoder representations? We find that, yes, while structured and unstructured neural models (using the same encoder representations) obtain similar F1 on in-domain datasets, **the structured model typically generalizes better to longer spans and out-of-domain treebanks, and has higher exact match accuracies in all domains**.

2 Experimental setup

We compare the generalization of strong non-neural parsers against recent state-of-the-art neural parsers on English and Chinese corpora.

Non-neural models We use publicly released code and models for the Berkeley Parser (Petrov and Klein, 2007) and BLLIP Parser (Charniak, 2000; Charniak and Johnson, 2005) for English; and ZPar (Zhang and Clark, 2011) for Chinese.

Neural models We use two state-of-the-art neural models: the Chart model of Kitaev and Klein (2018), and In-Order shift-reduce model of Liu and Zhang (2017). These parsers differ in their modeling both of input sentences and output structures. The Chart model uses a self-attentive encoder over the input sentence, and does not explicitly model output structure correlations, predicting tree span labels independently conditioned on the encoded input.[1] The In-Order shift-reduce model of Liu and Zhang (2017) uses a simpler LSTM-based encoding of the input sentence but a decoder that explicitly conditions on previously constructed structure of the output tree, obtaining the best performance among similarly structured models (Dyer et al., 2016; Kuncoro et al., 2017).

The In-Order model conditions on predicted part-of-speech tags; we use tags predicted by the Stanford tagger (following the setup of Cross and Huang (2016)). At test time, we use Viterbi decoding for the Chart model and beam search with beam size 10 for the In-Order model.

To control for randomness in the training procedure of the neural parsers, all scores reported in the remainder of the paper for the Chart and In-Order parsers are averaged across five copies of each model trained from separate random initializations.

Corpora The English parsers are trained on the WSJ training section of the Penn Treebank. We perform in-domain evaluation of these parsers on the WSJ test section, and out-of-domain evaluation using the Brown, Genia, and English Web Treebank (EWT) corpora. For analysis and comparisons within parsers, we evaluate on the entirety of each out-of-domain treebank; for final results and comparison to past work we use the same testing splits as the past work.

The Chinese parsers are trained on the training section of the Penn Chinese Treebank (CTB) v5.1 (Xue et al., 2005), consisting primarily of newswire. For out-of-domain evaluation on Chinese, we use treebank domains introduced in CTB versions 7 and 8: broadcast conversations (B. Conv), broadcast news (B. News), web discussion forums (Forums) and weblogs (Blogs).

[1] The only joint constraint on span predictions is to ensure they constitute a valid tree.

	ZPar		In-Order	
	F1	Δ Err.	F1	Δ Err.
CTB Test	83.01	+0.0%	83.67	+0.0%
B. News	77.22	+34.1%	77.83	+35.8%
Forums	74.31	+51.2%	75.71	+48.7%
Blogs	73.90	+53.6%	74.74	+54.7%
B. Conv.	66.70	+96.0%	67.69	+97.9%

Table 2: Performance on Chinese corpora and increase in error (relative to the CTB test set) as parsers are evaluated out-of-domain. The non-neural (ZPar) and neural (In-Order) parser generalize similarly.

3 How well do neural parsers generalize?

Table 1 compares the generalization performance of the English parsers, both non-neural (Berkeley, BLLIP) and neural (Chart, In-Order). None of these parsers use additional data beyond the WSJ training section of the PTB: we use the version of the BLLIP parser without self-training on unlabeled data, and use the In-Order parser without external pre-trained word embeddings. Across all parsers, higher performance on the WSJ Test set corresponds to higher performance on each out-of-domain corpus, showing that the findings of McClosky et al. (2006) extend to recent neural parsers. In particular, the Chart parser has highest performance in all four domains.

The Δ Err. column shows the *generalization gap* for each parser on each corpus: the parser's relative increase in error (with error defined by 100 – F1) from the WSJ Test set (lower values are better). Improved performance on the WSJ Test set corresponds to increased generalization gaps, indicating that to some extent parser improvements on WSJ have come at the expense of out-of-domain generalization. However, the two parsers with similar absolute performance on WSJ—the BLLIP parser and In-Order parser—have comparable generalization gaps, despite one being neural and one non-neural.

Table 2 shows results for ZPar and the In-Order parser on the Chinese treebanks, with Δ Err. computed relative to the in-domain CTB Test set. As with the English parsers and treebanks, increased performance on the in-domain test set corresponds to improvements on the out-of-domain treebanks (although these differences are small enough that this result is less conclusive than for English). In addition, as with English, we observe similar generalization performance of the non-neural and neural parsers across the out-of-domain treebanks.

	In-Order	+Embeddings		+BERT	
	F1	F1	Δ Err.	F1	Δ Err.
WSJ Test	91.47	92.13	-7.7%	95.71	-49.7%
Brown All	85.60	86.78	-8.2%	93.53	-55.0%
Genia All	80.31	81.64	-6.8%	87.75	-37.8%
EWT All	79.07	80.50	-6.8%	89.27	-48.7%
CTB Test	83.67	85.69	-12.4%	91.81	-49.9%
B. News	77.83	81.64	-17.2%	88.41	-47.7%
Forums	75.71	79.44	-15.4%	87.04	-46.6%
Blogs	74.74	78.21	-13.7%	84.29	-37.8%
B. Conv.	67.69	70.34	-8.2%	75.88	-25.3%

Table 3: Performance of the In-Order parser, comparing using no pre-trained representations (first column), word embeddings, and BERT, on English (top) and Chinese (bottom) corpora. Δ Err. shows change in F1 error relative to the base parser (without pretraining). For both pre-training methods, error reduction is not typically greater out-of-domain than in-domain.

4 How much do pretrained representations help out-of-domain?

Pre-trained word representations have been shown to increase in-domain parsing accuracies. Additionally, Joshi et al. (2018) showed that these representations (in their case, from ELMo, Peters et al. 2018) allow a parser to transfer well across domains. We analyze whether pre-trained representations provide a greater benefit in-domain or out-of-domain, by comparing relative performance improvements on in-domain and out-of-domain treebanks when augmenting the neural parsers with pre-trained language representations. We evaluate non-contextual word embeddings produced by structured skip-gram (Ling et al., 2015), as well as the current state-of-the-art contextual representations from BERT (Devlin et al., 2019).

4.1 Word embeddings

We use the same pre-trained word embeddings as the original In-Order English and Chinese parsers,[2] trained on English and Chinese Gigaword (Parker et al., 2011) respectively. Table 3 compares models without (In-Order column) to models with embeddings (+Embeddings), showing that embeddings give comparable error reductions both in-domain (the WSJ Test and CTB Test rows) and out-of-domain (the other rows).

4.2 BERT

For the Chart parser, we compare the base neural model (Sec. 2 and 3) to a model that uses a pre-

[2] https://github.com/LeonCrashCode/InOrderParser

	Chart F1	+BERT F1	 Δ Err.
WSJ Test	93.27	95.64	-35.2%
Brown All	88.04	93.10	-42.3%
Genia All	82.68	87.54	-28.1%
EWT All	82.22	88.72	-36.6%

Table 4: Performance of the Chart parser on English, comparing using no pretrained representations to using BERT. Δ Err. shows change in F1 error relative to the base parser. BERT does not generally provide a larger error reduction out-of-domain than in-domain.

trained BERT encoder (Kitaev et al., 2019), using the publicly-released code[3] to train and evaluate both models.

For the In-Order parser, we introduce a novel integration of a BERT encoder with the parser's structured tree decoder. These architectures represent the best-performing types of encoder and decoder, respectively, from past work on constituency parsing, but have not been previously combined. We replace the word embeddings and predicted part-of-speech tags in the In-Order parser's stack and buffer representations with BERT's contextual embeddings. See Appendix A.1 for details on the architecture. Code and trained models for this system are publicly available.[4]

Both the Chart and In-Order parsers are trained in the same way: the parameters of the BERT encoder ($\text{BERT}_{\text{LARGE, Uncased}}$ English or $\text{BERT}_{\text{BASE}}$ Chinese) are fine-tuned during training on the treebank data, along with the parameters of the parser's decoder. See Appendix A.2 for details.

Results for the In-Order parser are shown in the +BERT section of Table 3, and results for the chart parser are shown in Table 4. BERT is effective across domains, providing between 25% and 55% error reduction over the base neural parsers. However, as for word embeddings, the pre-trained BERT representations do not generally provide a larger error reduction in out-of-domain settings than in in-domain (although a possible confound is that the BERT model is fine-tuned on the relatively small amount of in-domain treebank data, along with the other parser parameters).

For English, error reduction from BERT is comparable between WSJ and EWT, largest on Brown, and smallest on Genia, which may indicate a dependence on the similarity between the out-of-domain treebank and the pre-training corpus.[5] For Chinese, the relative error reduction from BERT is largest on the in-domain CTB Test corpus.

	F1 Chart +BERT	 In-Order +BERT	Exact Match Chart +BERT	 In-Order +BERT
WSJ Test	95.64	95.71	55.11	57.05
Brown All	93.10	93.54	49.23	51.98
EWT All	88.72	89.27	41.83	43.98
Genia All	87.54	87.75	17.46	18.03
CTB Test	92.14	91.81	44.42	44.94
B. News	88.21	88.41	15.91	17.29
Forums	86.72	87.04	20.00	21.95
Blogs	84.28	84.29	17.14	18.85
B. Conv.	76.35	75.88	17.24	18.99

Table 5: F1 and exact match accuracies comparing the Chart (unstructured) and In-Order (structured) parsers with BERT pretraining on English (top) and Chinese (bottom) corpora.

5 Can structure improve performance?

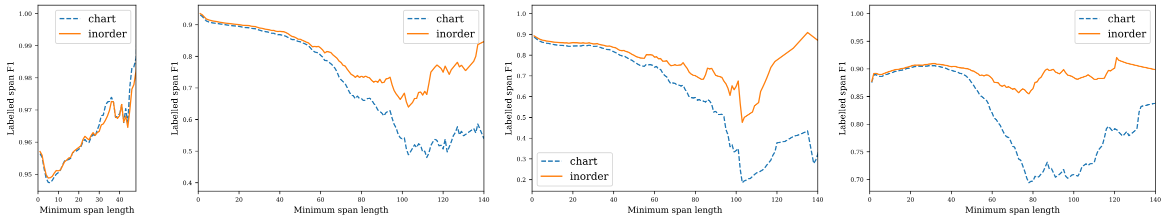
When using BERT encoder representations, the Chart parser (with its unstructured decoder) and In-Order parser (with its conditioning on a representation of previously-constructed structure) obtain roughly comparable F1 (shown in the first two columns of Table 5), with In-Order better on seven out of nine corpora but often by slight margins. However, these aggregate F1 scores decompose along the structure of the tree, and are dominated by the short spans which make up the bulk of any treebank. Structured-conditional prediction may plausibly be most useful for predicting larger portions of the tree, measurable in exact match accuracies and in F1 on longer-length spans (containing more substructure).

First, we compare the tree-level exact match accuracies of the two parsers. In the last two columns of Table 5, we see that the In-Order parser consistently achieves higher exact match than the Chart parser across domains (including the in-domain WSJ and CTB Test sets), with improvements ranging from 0.5 to 2.8 percentage absolute. In fact, for several corpora (Blogs and B. Conv) the In-Order parser outperforms the Chart parser on exact match despite having the same or lower F1. This suggests that conditioning on structure in the model induces a correlation between span-level decisions that becomes most apparent when using a metric defined on the entire structure.

[3] https://github.com/nikitakit/self-attentive-parser

[4] https://github.com/dpfried/rnng-bert

[5] BERT is pre-trained on books and Wikipedia; Genia consists of biomedical text.



(a) WSJ Test

(b) Brown All

(c) EWT All

(d) Genia All

Figure 1: Labelled bracketing F1 versus minimum span length for the English corpora. F1 scores for the In-Order parser with BERT (orange) and the Chart parser with BERT (cyan) start to diverge for longer spans.

	prior work	Chart +BERT	In-Order +BERT
Brown Test	87.7 (C+'15)	93.16	93.66
Genia Test	79.4 (C+'15)	86.11	86.45
EWT Test	83.5 (L+'12)	89.13	89.62

Table 6: Comparison of F1 scores for neural models with BERT pretraining to past state-of-the art results on transfer to the out-of-domain treebanks: (C+'15: Choe et al. 2015, L+'12: Le Roux et al. 2012).[6] EWT scores are averaged across the 3 SANCL'12 test sets, as reported by Petrov and McDonald (2012).

Second, we compare the performance of the two parsers on longer spans of text. Figure 1 plots F1 by minimum span length for the In-Order and Chart parsers with BERT encoders on the English treebanks. Across datasets, the improvement of the In-Order parser is slight when computing F1 across all spans in the dataset ($x = 0$), but becomes pronounced when considering longer spans. This effect is not observed in the WSJ test set, which may be attributable to its lack of sufficiently many long spans for us to observe a similar effect there. The curves start to diverge at span lengths of around 30–40 words, longer than the median length of a sentence in the WSJ (23 words).

6 Discussion

Neural parsers generalize surprisingly well, and are able to draw benefits both from pre-trained language representations and structured output prediction. These properties allow single-model parsers to surpass previous state-of-the-art systems on out-of-domain generalization (Table 6). We note that these systems from prior work (Choe et al., 2015; Petrov and McDonald, 2012; Le Roux et al., 2012) use additional ensembling or self-training techniques, which have also been shown to be compatible with neural constituency parsers (Dyer et al., 2016; Choe and Charniak, 2016; Fried et al., 2017; Kitaev et al., 2019) and may provide benefits orthogonal to the pre-trained representations and structured models we analyze here. Encouragingly, parser improvements on the WSJ and CTB treebanks still transfer out-of-domain, indicating that improving results on these benchmarks may still continue to yield benefits in broader domains.

Acknowledgements

This research was supported by DARPA through the XAI program, as well as by a Tencent AI Lab fellowship to the first author. This research used the Savio computational cluster provided by the Berkeley Research Computing program at the University of California, Berkeley.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan. Association for Computational Linguistics.

[6] Although the F1 scores obtained here are higher than the zero-shot transfer results of Joshi et al. (2018) on the Brown and Genia corpora due to the use of improved encoder (BERT) and decoder (self-attentive Chart and In-Order) models, we note the results are not directly comparable due to the use of different sections of the corpora for evaluation.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336, Austin, Texas. Association for Computational Linguistics.

Do Kook Choe, David McClosky, and Eugene Charniak. 2015. Syntactic parse fusion. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1360–1366, Lisbon, Portugal. Association for Computational Linguistics.

James Cross and Liang Huang. 2016. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1–11. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

Winthrop Nelson Francis and Henry Kučera. 1979. *Manual of information to accompany a standard corpus of present-day edited American English, for use with digital computers*. Brown University, Department of Linguistics.

Daniel Fried, Mitchell Stern, and Dan Klein. 2017. Improving neural parsing by disentangling model combination and reranking effects. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 161–166, Vancouver, Canada. Association for Computational Linguistics.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*.

John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan Brennan. 2018. Finding syntax in human encephalography with beam search. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2727–2736, Melbourne, Australia. Association for Computational Linguistics.

James Henderson. 2004. Discriminative training of a neural network statistical parser. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 95–102, Barcelona, Spain.

Vidur Joshi, Matthew Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1199, Melbourne, Australia. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686. Association for Computational Linguistics.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain. Association for Computational Linguistics.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Melbourne, Australia. Association for Computational Linguistics.

Joseph Le Roux, Jennifer Foster, Joachim Wagner, Rasul Kaljahi, and Anton Bryl. 2012. Dcu-paris13 systems for the sancl 2012 shared task. In *SANCL Shared Task*.

Wang Ling, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Two/too simple adaptations of word2vec for syntax problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1299–1304. Association for Computational Linguistics.

Jiangming Liu and Yue Zhang. 2017. In-order transition-based constituent parsing. *Transactions of the Association for Computational Linguistics*, 5:413–424.

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

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 337–344. Association for Computational Linguistics.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. *ArXiv*.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword. *Linguistic Data Consortium*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411, Rochester, New York. Association for Computational Linguistics.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*.

Laura Rimell and Stephen Clark. 2009. Porting a lexicalized-grammar parser to the biomedical domain. *Journal of biomedical informatics*, 42(5):852–865.

Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordoni, Aaron Courville, and Yoshua Bengio. 2018. Straight to the tree: Constituency parsing with neural syntactic distance. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1180, Melbourne, Australia. Association for Computational Linguistics.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 818–827. Association for Computational Linguistics.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax annotation for the genia corpus. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in neural information processing systems*, pages 2773–2781.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(2):207–238.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1).

A Appendix

A.1 Integrating BERT into the In-Order Parser

In this section we describe our integration of the BERT encoder into the In-Order parser decoder. We refer to the original In-Order (Liu and Zhang, 2017) and BERT (Devlin et al., 2019) papers for full details about the model architectures, only describing the modifications we make at the interface between the two. Code and pre-trained models for this integrated parser are publicly available.[7]

BERT divides each word in an input sentence into one or more subword units and produces a contextual representation for each subword unit using a self-attentive architecture (Devlin et al., 2019). Following the implementation of Kitaev et al. (2019) for the Chart parser, we take the contextual representation vector for the last subword unit in each word w_i as the word's representation, e_{w_i}, replacing the (non-contextual) word and POS tag vectors used in the original In-Order parser. We use a learned linear projection to scale e_{w_i} to a vector x_i of size 128 (compare with section 4.1 of Liu and Zhang (2017)).

These contextual word representations x_i enter into the In-Order parser's decoder in two positions: the stack (representing the parse tree as constructed so far) and the buffer (representing the remainder of the sentence to be parsed). We retain the stack representation, but omit the LSTM which the original In-Order work uses to summarize the words remaining on the buffer. We instead use the representation x_i as the buffer summary for the word i when i is word at the front of the buffer (the next word in the sentence to be processed). In early experiments we found that removing the LSTM summary of the buffer in this manner had no consistent effect on performance, indicating that the BERT contextual vectors already sufficiently aggregate information about the input sentence so that an additional LSTM provides no further benefit.

We pass values and gradients between the DyNet (Neubig et al., 2017) implementation of the In-Order parser and the Tensorflow (Abadi et al., 2016) implementation of BERT using the Tensorflow C++ API.

A.2 BERT Optimization Settings

We train the In-Order parser with BERT following the optimization procedure used in Kitaev et al. (2019)'s publicly-released implementation of the BERT Chart parser: training with mini-batches of size 32 using the Adam optimizer (Kingma and Ba, 2015); halving the base learning rates for Adam whenever 2 epochs of training pass without improved F1 on the development set, and using a warmup period for the BERT learning rate. For the In-Order parser, we use initial Adam learning rates of 2×10^{-5} for the BERT encoder parameters and 1×10^{-3} for the In-Order decoder parameters, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a BERT learning rate warmup period of 160 updates.

[7] https://github.com/dpfried/rnng-bert