An Empirical Study of Compound PCFGs

Yanpeng Zhao[ε]
[ε]ILCC, University of Edinburgh
yanp.zhao@ed.ac.uk

Ivan Titov[εæ]
[æ]ILLC, University of Amsterdam
ititov@inf.ed.ac.uk

Abstract

Compound probabilistic context-free grammars (C-PCFGs) have recently established a new state of the art for phrase-structure grammar induction. However, due to the high time-complexity of chart-based representation and inference, it is difficult to investigate them comprehensively. In this work, we rely on a fast implementation of C-PCFGs to conduct evaluation complementary to that of Kim et al. (2019). We highlight three key findings: (1) C-PCFGs are data-efficient, (2) C-PCFGs make the best use of global sentence-level information in preterminal rule probabilities, and (3) the best configurations of C-PCFGs on English do not always generalize to morphology-rich languages.

1 Introduction

Probabilistic context-free grammars (PCFGs) have been used for unsupervised constituency grammar learning since decades ago (Lari and Young, 1990), while learning PCFGs with the Expectation Maximization algorithm (Dempster et al., 1977) has been difficult as being involving non-convex optimization. Recently, Kim et al. (2019) propose compound PCFGs, an over-parameterized neural model that extends corpus-level PCFGs by defining a mixture of PCFGs per sentence. C-PCFGs have achieved the state-of-the-art performance on English and Chinese treebanks. They are also shown to be effective in a visually-grounded learning setting (Zhao and Titov, 2020). However, because of the high time-complexity of chart-based representation and inference, it is hard to inspect C-PCFGs comprehensively.

In this work, we rely on a fast implementation[1] of C-PCFGs to conduct a set of experiments complementary to those of Kim et al. (2019). Our first experiment concerns data efficiency and length generalization of C-PCFGs. We empirically find that though trained only on short sentences, e.g., shorter than 30 tokens, C-PCFGs can generalize to longer sentences while maintaining high performance (54.8% F1) at test time. We further investigate which factors contribute to the good performance of C-PCFGs. Since a major difference between C-PCFGs and vanilla PCFGs is that C-PCFGs define sentence-dependent rule probabilities by using global sentence-level information, we ablate C-PCFGs by removing it from start, preterminal, and nonterminal rules,[2] individually. Our experimental results show that sentence-level information is most effective for preterminal rules. Despite the impressive performance of C-PCFGs on English, it is still unclear whether they can generalize to other languages. We thus conduct multilingual evaluation of C-PCFGs on the SPMRL dataset (Seddah et al., 2014). The experimental results suggest that the best configurations of C-PCFGs on English do not necessarily generalize to morphology-rich languages.

2 Compound PCFGs

Compound PCFGs provide a novel parameterization of PCFGs. Unlike PCFGs, which assign each grammar rule r a non-negative *scalar* π_r such that $\sum_{r:A\rightarrow\gamma} \pi_r = 1$ for each given left-hand-side symbol A, C-PCFGs relax the strong context-free assumption of PCFGs by assuming that rule probabilities follow a compound distribution:

$$\pi_r = g_r(\mathbf{z};\theta), \quad \mathbf{z} \sim p(\mathbf{z}),$$

[1] https://github.com/zhaoyanpeng/cpcfg.

[2] Start rules generate a nonterminal symbol from the start symbol S (e.g., $S \rightarrow A$), preterminal rules generate a word from a nonterminal symbol (e.g., $A \rightarrow w$), and nonterminal rules are binary rules of the form $A \rightarrow BC$, which involve only nonterminal symbols.

Proceedings of the Second Workshop on Domain Adaptation for NLP, pages 166–171
April 20, 2021.

where $p(\mathbf{z})$ is a prior distribution and allows for capturing interdependencies between the rules; $g_r(\mathbf{z};\theta)$ takes a latent $\mathbf{z}$ as input and incorporates the interdependencies into rule probabilities. Typically, $g_r(\mathbf{z};\theta)$ is parameterized by flexible neural networks and is amenable to gradient-based optimization techniques (we refer interested readers to Kim et al. (2019) for the detailed parameterization).

Learning C-PCFGs is formalized as maximizing the log likelihood of each observed sentence $\boldsymbol{w} = w_1 w_2 \dots w_n$:

$$\log p_\theta(\boldsymbol{w}) = \log \int_{\mathbf{z}} \sum_{t \in \mathcal{T}_\mathcal{G}(\boldsymbol{w})} p_\theta(t|\mathbf{z}) p(\mathbf{z})\, d\mathbf{z}\,,$$

where $\mathcal{T}_\mathcal{G}(\boldsymbol{w})$ consists of all parses of a sentence $\boldsymbol{w}$ under a PCFG $\mathcal{G}$. As standard in learning latent variable models, C-PCFGs resort to variational inference for tractable learning and instead maximize the evidence lower bound (ELBO):

$$\log p_\theta(\boldsymbol{w}) \geq \text{ELBO}(\boldsymbol{w};\phi,\theta) = \\ \mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{w})}[\log p_\theta(\boldsymbol{w}|\mathbf{z})] - \text{KL}[q_\phi(\mathbf{z}|\boldsymbol{w})||p(\mathbf{z})]\,,$$

where the first term computes the expected log likelihood under a variational posterior $q_\phi(\mathbf{z}|\boldsymbol{w})$; the KL term can be estimated analytically when $p(\mathbf{z})$ and $q_\phi(\mathbf{z}|\boldsymbol{w})$ are normally distributed. $q_\phi(\mathbf{z}|\boldsymbol{w})$ is parameterized by a neural network and defines a distribution over the latent $\mathbf{z}$.

C-PCFGs satisfy the context-free assumption conditioned on $\mathbf{z}$ and thus admit tractable inference for each given $\mathbf{z}$. Inference with C-PCFGs seeks the most probable parse t^* of $\boldsymbol{w}$:

$$t^* = \text{argmax} \int_{\mathbf{z}} p_\theta(t|\mathbf{w},\mathbf{z}) p_\theta(\mathbf{z}|\boldsymbol{w})\, d\mathbf{z}\,.$$

Though given $\mathbf{z}$, the maximum a posterior (MAP) inference over $p_\theta(t|\mathbf{w},\mathbf{z})$ can be exactly solved by using the CYK algorithm, the integral over $\mathbf{z}$ renders inference intractable. The MAP inference is instead approximated by:

$$t^* \approx \text{argmax} \int_{\mathbf{z}} p_\theta(t|\mathbf{w},\mathbf{z}) \delta(\mathbf{z} - \boldsymbol{\mu}_\phi(\boldsymbol{w}))\, d\mathbf{z}\,,$$

where $\delta(\cdot)$ is the Dirac delta function and $\boldsymbol{\mu}_\phi(\boldsymbol{w})$ is the mean vector of the variational posterior.

Similarly to C-PCFGs, neural PCFGs (N-PCFGs) also use neural networks to parameterize PCFGs, but their parameterization does not rely on the sentence-dependent $\mathbf{z}$. In the following discussion, we will refer to $\mathbf{z}$ as 'sentence embedding'.

3 Experimental setup

Datasets: We investigate the parsing performance of C-PCFGs across ten languages. Specifically, we conduct experiments on the Wall Street Journal (WSJ) corpus of the Penn Treebank (Marcus et al., 1994) for English, the Penn Chinese Treebank 5.1 (CTB) (Xue et al., 2005) for Chinese, and eight additional treebanks from the SPMRL 2014 shared task (Seddah et al., 2014) for the other eight languages (Basque, German, French, Hebrew, Hungarian, Korean, Polish, Swedish). We use the standard data splits for each treebank. Following Kim et al. (2019), punctuation is removed from all data; the top 10000 frequent tokens in the training data of each treebank are kept as the vocabulary.[3] Unless otherwise specified, we train C-PCFGs on sentences no longer than 40 tokens.

Model hyperparameters and evaluation: We re-implement C-PCFGs relying on Torch-Struct (Rush, 2020) and adopt the same hyperparameter settings as in Kim et al. (2019). We train C-PCFGs for each language separately. On each treebank we run C-PCFGs four times with different random seeds and for 30 epochs. The best model in each run is selected according to the perplexity on the validation data. At test time, trivial spans, such as single-word and sentence-level spans, are ignored. We report average corpus- and sentence-level F1 numbers as well as the unbiased standard deviations.

4 Results and discussion

4.1 Main results

We compare C-PCFGs against three trivial baselines (left- / right-branching model and random trees) and a neural PCFG model. In short, C-PCFGs beats all baselines in terms of corpus- and sentence-level F1 (see the second row of Table 1). Our re-implementation of C-PCFGs reaches the highest sentence-level F1, slightly outperforming the model of Kim et al. (2019) by 0.5% F1. To give an in-depth analysis of the model gains, we present recall numbers on six most frequent constituent labels in the test data (NP, VP, PP, SBAR, ADJP, ADVP). Unsurprisingly, C-PCFGs achieve the best recall for most labels (4 out of 6 constituent labels). However, on verb phrases (VPs) they fall far behind the right-branching baseline (-30.8%

[3] A unified data preprocessing pipeline is available at https://github.com/zhaoyanpeng/xcfg.

Model	NP	VP	PP	SBAR	ADJP	ADVP	C-F1	S-F1
Left Branching	10.4	0.5	5.0	5.3	2.5	8.0	6.0	8.7
Right Branching	24.1	**71.5**	42.4	**68.7**	27.7	38.1	36.1	39.5
Random Trees	$22.5_{\pm 0.3}$	$12.3_{\pm 0.3}$	$19.0_{\pm 0.5}$	$9.3_{\pm 0.6}$	$24.3_{\pm 1.7}$	$26.9_{\pm 1.3}$	$15.3_{\pm 0.1}$	$18.1_{\pm 0.1}$
†N-PCFG	71.2	33.8	58.8	52.5	32.5	45.5		50.8
N-PCFG	$72.2_{\pm 4.8}$	$31.4_{\pm 9.7}$	$66.8_{\pm 4.7}$	$50.2_{\pm 9.1}$	$46.3_{\pm 5.7}$	$55.2_{\pm 5.0}$	$49.0_{\pm 3.5}$	$50.8_{\pm 3.8}$
†C-PCFG	74.7	41.7	68.8	56.1	40.4	52.5		55.2
C-PCFG	$76.7_{\pm 2.0}$	$40.7_{\pm 5.5}$	$71.3_{\pm 2.1}$	$53.8_{\pm 3.1}$	$45.9_{\pm 2.8}$	$64.2_{\pm 2.8}$	$53.5_{\pm 1.4}$	$55.7_{\pm 1.3}$
L50C-PCFG	$\mathbf{76.9}_{\pm 3.6}$	$40.7_{\pm 3.7}$	$\mathbf{72.3}_{\pm 0.6}$	$60.1_{\pm 5.5}$	$\mathbf{46.9}_{\pm 5.8}$	$63.2_{\pm 5.0}$	$\mathbf{53.8}_{\pm 2.1}$	$\mathbf{55.9}_{\pm 1.9}$
L40C-PCFG	$76.7_{\pm 2.0}$	$40.7_{\pm 5.5}$	$71.3_{\pm 2.1}$	$53.8_{\pm 3.1}$	$45.9_{\pm 2.8}$	$64.2_{\pm 2.8}$	$53.5_{\pm 1.4}$	$55.7_{\pm 1.3}$
L30C-PCFG	$74.5_{\pm 2.8}$	$38.4_{\pm 1.7}$	$71.1_{\pm 1.2}$	$59.7_{\pm 4.8}$	$44.2_{\pm 4.1}$	$\mathbf{64.3}_{\pm 3.1}$	$52.5_{\pm 1.5}$	$54.8_{\pm 1.4}$
L20C-PCFG	$72.4_{\pm 2.3}$	$36.5_{\pm 1.1}$	$69.2_{\pm 1.7}$	$54.1_{\pm 3.2}$	$41.9_{\pm 2.3}$	$58.1_{\pm 7.1}$	$50.6_{\pm 0.9}$	$52.8_{\pm 0.7}$
L10C-PCFG	$67.1_{\pm 3.8}$	$31.0_{\pm 9.8}$	$61.3_{\pm 2.2}$	$45.9_{\pm 8.2}$	$36.7_{\pm 2.3}$	$41.3_{\pm 6.0}$	$45.5_{\pm 2.4}$	$48.2_{\pm 2.3}$

Table 1: Recall on six frequent constituent labels (NP, VP, PP, SBAR, ADJP, ADVP) in the WSJ test data, corpus-level F1 (C-F1), and sentence-level F1 (S-F1) results. The best mean number in each column is in bold. † denotes results reported by Kim et al. (2019). L# indicates that the models are trained on sentences no longer than # tokens.

recall), presumably because VPs are longer and involve more complex linguistic structures. We further plot distributions of the six labels across constituent lengths (see Figure 1). We can see that VPs are nearly uniformly distributed over different constituent lengths. In contrast, noun phrases (NPs) account for 61% of short constituents that have less than 6 tokens and cover 51% of total constituents. It suggests that C-PCFGs can recognize local and short constituents with a high accuracy but struggles with long constituents; there is clearly a room for improvement on VPs.

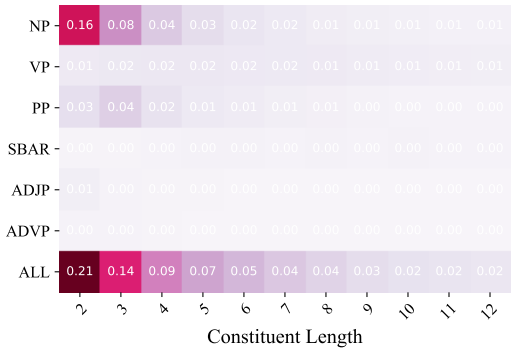


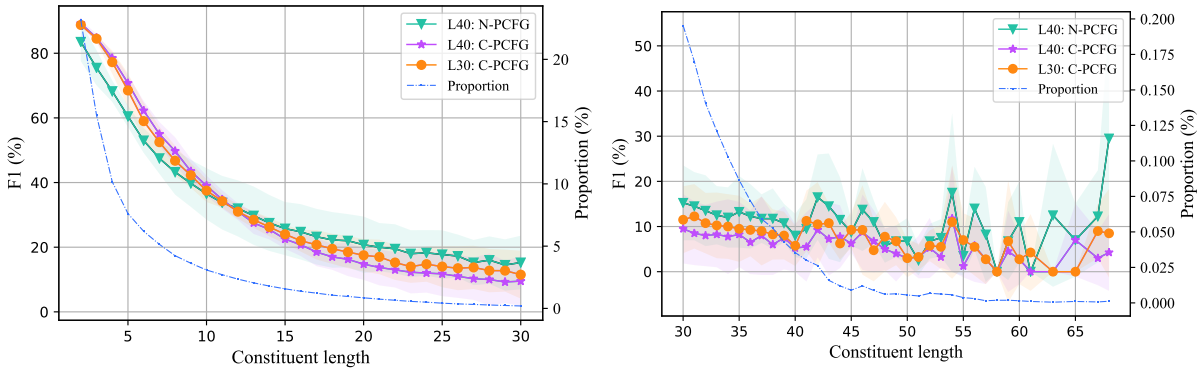
Figure 1: Label distribution over constituent lengths on the WSJ test data. *All* denotes frequencies of constituent lengths. Zero frequencies are due to the limited numerical precision.

4.2 Data efficiency and length generalization

A crucial aspect of human languages is their compositionality. Humans can derive grammar rules from a few sentences and combine the rules to generate new sentences compositionally. As C-PCFGs are backed by context-free grammar, we hypothesize that C-PCFGs are data-efficient and have a good generalizability to unseen sentences and constituents.

We design a length-generalization test to verify our hypothesis. Specifically, we train C-PCFGs using training sentences of length equal to or below a chosen sentence length. We choose five sentence lengths, 10, 20, 30, 40 and 50, indicated by L10, L20, L30, L40 and L50, respectively (see the third row of Table 1)). Figure 3 illustrates sentence-level F1 numbers on the test data of WSJ. Overall, training C-PCFGs on more / longer sentences results in higher F1 numbers. But using training sentences longer than 40 tokens only trivially improves the performance (+0.2% F1). Given that 97% test sentences are shorter than 40 tokens, we conjecture that training sentences shorter than 40 tokens can adequately cover lexical / structural characteristics in the test data. On the other hand, longer sentences have a larger tree space and probably make it harder for the model to learn. Notably, discarding training sentences longer than 30 tokens only decreases the model performance by 1.2% F1, suggesting that C-PCFGs are data-efficient.

We also conduct a constituent-length generalization test to study the generalizability of C-PCFGs on unseen long constituents. Since the test data of WSJ is too small to provide reliable statistics across constituent lengths, we test C-PCFGs on training sentences and report F1 numbers across *constituent lengths* (see Figure 2). In general, F1 numbers become lower as constituent length increases. This is reasonable because large constituents merge from



(a) F1 w.r.t. constituent length **below** 30

(b) F1 w.r.t. constituent length **above** 30

Figure 2: F1 numbers broken down by constituent lengths on the WSJ training data. During training, constituents (sentences) longer than 30 tokens (L30) are unseen to L30C-PCFG and are unseen to L40C-PCFG and L40N-PCFG when longer than 40 tokens (L40).

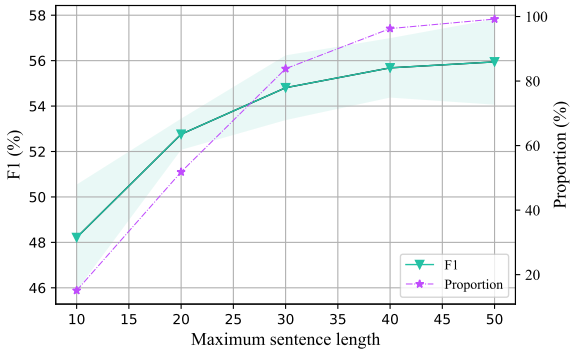


Figure 3: F1 numbers on the WSJ test data with varying maximum lengths of training sentences.

small constituents; errors in small constituents accumulate when composing larger constituents.

We further investigate the influence of training data on length generalization. We use sentence lengths 30 and 40 as an illustration. Compare with L40C-PCFG, when tested on constituents longer than 40 tokens, L30C-PCFG shows a slightly better generalizability (see Figure 2b). It consistently outperforms L40C-PCFG on sentences of length ranging from 30 to 40, though L40C-PCFG can access all sentences shorter than 40 tokens during training. This implies that C-PCFGs generalize well from short sentences; using additional long training sentences may hurt the generalizability. We also plots the proportions of constituent lengths. For example, there are about 6400 constituents of length from 30 to 40, which account for about 1.1% of total constituents, suggesting that the conclusion about the better generalizability of L30C-PCFGs is reliable.

Figure 2b visualizes the performance of an L40N-PCFG. Surprisingly, L40N-PCFG shows the best generalizability on long constituents. *Where does the F1 improvement of* C-PCFGs *over* N-PCFGs *come from?* Look at the F1 numbers on shorter constituents in Figure 2a, clearly C-PCFGs are better on constituents that are shorter than 11 tokens, while L40N-PCFGs consistently outperform C-PCFGs on constituents of length above 11. L30C-PCFGs fall in between L40C-PCFGs and L40N-PCFGs, once again showing that restricting training to short sentences can endow C-PCFGs good parsing performance as well as lead to improved generalization.

4.3 Model ablation

C-PCFGs demonstrate a significant improvement over N-PCFGs. Compare with N-PCFGs, C-PCFGs use an additional sentence embedding (i.e., the latent variable $\mathbf{z}$, see Section 2) to parameterize sentence-specific PCFGs. Concretely, the sentence embedding is used to parameterize three types of rules: preterminal rules (P), nonterminal rules (N), and start rules (R). We would like to know *which type of rules makes the best use of the sentence embedding?* To this end, we let a C-PCFG use corpus-level parameters for each of the three types of rules, individually, i.e., parameters for a rule type are shared among sentences. Interestingly, C-PCFGs degenerate into N-PCFGs when using corpus-level parameters for preterminal rules (see Figure 4). It implies that the sentence embedding is most crucial for the parameterization of preterminal rules, presumably because the sentence embedding helps preterminal rules derive the knowledge of

Model	Chinese	Basque	German	French	Hebrew	Hungarian	Korean	Polish	Swedish	Mean
Left Branching	7.2	17.9	10.0	5.7	8.5	13.3	18.5	10.9	8.4	11.2
Right Branching	25.5	15.4	14.7	26.4	30.0	12.7	19.2	**34.2**	**30.4**	23.2
Random Trees	15.2	19.5	13.9	16.2	19.7	14.1	22.2	21.4	16.4	17.6
N-PCFG	30.1$_{\pm 4.6}$	**30.2**$_{\pm 0.9}$	**37.8**$_{\pm 1.7}$	**42.2**$_{\pm 1.4}$	**41.0**$_{\pm 0.6}$	37.9$_{\pm 0.8}$	25.7$_{\pm 2.8}$	31.7$_{\pm 1.8}$	14.5$_{\pm 12.7}$	32.3
C-PCFG	**35.1**$_{\pm 6.1}$	27.9$_{\pm 2.0}$	37.3$_{\pm 1.8}$	40.5$_{\pm 0.8}$	39.2$_{\pm 1.2}$	**38.3**$_{\pm 0.7}$	**27.7**$_{\pm 2.8}$	32.4$_{\pm 1.1}$	23.7$_{\pm 14.3}$	**33.6**

Table 2: Sentence-level F1 numbers on multilingual treebanks. Similarly to Kim et al. (2019), we observe that C-PCFGs suffer a huge variance, e.g., on the Chinese and Swedish treebanks.

part-of-speech tags, which is beneficial for parsing.

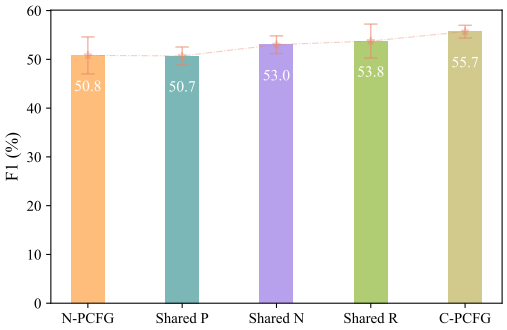


Figure 4: F1 numbers on the WSJ test data. Shared P / N / R indicates C-PCFGs that use corpus-level parameters for preterminal / nonterminal / start rules (see Section 4.3).

4.4 Multilingual evaluation

Despite the surprisingly good performance on English, it is still unclear whether C-PCFGs can generalize to languages beyond English. We thus conduct multilingual evaluation of C-PCFGs on nine additional languages (see Tablel 2). When training C-PCFGs on the nine languages, we use the hyperparameters of the best-performing C-PCFG on English, i.e., we tune C-PCFGs only on WSJ and use the best configurations on the other treebanks. We can see that C-PCFGs achieve the highest overall mean F1 (average F1 number over all treebanks), though they have two fewer winning treebanks than N-PCFGs. Notably, both C-PCFGs and N-PCFGs outperform the trivial baselines by a large margin, suggesting their nice generalizability on languages beyond English. However, they are worse than the right-branching baseline on the Polish and Swedish treebanks. As these languages have rich morphologies, we anticipate an improvement from encoding the knowledge of morphologies into the sentence embedding.

5 Conclusion

We have presented an in-depth analysis of C-PCFGs from a quantitative perspective. The analysis concerns three aspects of C-PCFGs: data efficiency and length generalization, the role of the latent sentence embedding, and multilingual performance. Our experimental results show that C-PCFGs can learn well only from short sentences and maintain good performance at test time. The latent sentence embedding is crucial for the good performance of C-PCFGs. Among the three rule types, preterminal rules make the most of it. However, the configurations of the best-performing C-PCFGs on English do not always generalize to morphology-rich languages.

Acknowledgments

We would like to thank anonymous reviewers for their suggestions and comments. The project was supported by the European Research Council (ERC Starting Grant BroadSem 678254) and the Dutch National Science Foundation (NWO VIDI 639.022.518).

References

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Yoon Kim, Chris Dyer, and Alexander Rush. 2019. Compound probabilistic context-free grammars for grammar induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2369–2385, Florence, Italy. Association for Computational Linguistics.

K. Lari and S.J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4(1):35 – 56.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger.

1994. The Penn Treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Alexander Rush. 2020. Torch-struct: Deep structured prediction library. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 335–342, Online. Association for Computational Linguistics.

Djamé Seddah, Sandra Kübler, and Reut Tsarfaty. 2014. Introducing the SPMRL 2014 shared task on parsing morphologically-rich languages. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 103–109, Dublin, Ireland. Dublin City University.

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.

Yanpeng Zhao and Ivan Titov. 2020. Visually grounded compound PCFGs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4369–4379, Online. Association for Computational Linguistics.