Grammar Induction with Neural Language Models: An Unusual Replication

Phu Mon Htut[1]
AdeptMind Scholar
pmh330@nyu.edu

Kyunghyun Cho[1,2]
CIFAR Global Scholar
kyunghyun.cho@nyu.edu

Samuel R. Bowman[1,2,3]
bowman@nyu.edu

[1]Center for Data Science
New York University
60 Fifth Avenue
New York, NY 10011

[2]Dept. of Computer Science
New York University
60 Fifth Avenue
New York, NY 10011

[3]Dept. of Linguistics
New York University
10 Washington Place
New York, NY 10003

Abstract

A substantial thread of recent work on *latent tree learning* has attempted to develop neural network models with parse-valued latent variables and train them on non-parsing tasks, in the hope of having them discover interpretable tree structure. In a recent paper, Shen et al. (2018) introduce such a model and report near-state-of-the-art results on the target task of language modeling, and the first strong latent tree learning result on constituency parsing. In an attempt to reproduce these results, we discover issues that make the original results hard to trust, including tuning and even training on what is effectively the test set. Here, we attempt to reproduce these results in a fair experiment and to extend them to two new datasets. We find that the results of this work are robust: All variants of the model under study outperform all latent tree learning baselines, and perform competitively with symbolic grammar induction systems. We find that this model represents the first empirical success for latent tree learning, and that neural network language modeling warrants further study as a setting for grammar induction.

1 Introduction and Background

Work on *grammar induction* attempts to find methods for syntactic parsing that do not require expensive and difficult-to-design expert-labeled treebanks for training (Charniak and Carroll, 1992; Klein and Manning, 2002; Smith and Eisner, 2005). Recent work on *latent tree learning* offers a new family of approaches to the problem (Yogatama et al., 2017; Maillard et al., 2017; Choi et al., 2018). Latent tree learning models attempt to induce syntactic structure using the supervision from a downstream NLP task such as textual entailment. Though these models tend to show good task performance, they are often not evaluated using standard parsing metrics, and Williams et al. (2018a) report that the parses they produce tend to be no better than random trees in a standard evaluation on the full Wall Street Journal section of the Penn Treebank (WSJ; Marcus et al., 1993).

This paper addresses the Parsing-Reading-Predict Network (PRPN; Shen et al., 2018), which was recently published at ICLR, and which reports near-state-of-the-art results on language modeling and strong results on grammar induction, a first for latent tree models (though they do not use that term). PRPN is built around a substantially novel architecture, and uses convolutional networks with a form of structured attention (Kim et al., 2017) rather than recursive neural networks (Goller and Kuchler, 1996; Socher et al., 2011) to evaluate and learn trees while performing straightforward backpropagation training on a language modeling objective. In this work, we aim to understand what the PRPN model learns that allows it to succeed, and to identify the conditions under which this success is possible.

Their experiments on language modeling and parsing are carried out using different configurations of the PRPN model, which were claimed to be optimized for the corresponding tasks. PRPN-LM is tuned for language modeling performance, and PRPN-UP for (unsupervised) parsing performance. In the parsing experiments, we also observe that the WSJ data is not split, such that the test data is used without parse information for training. This approach follows the previous works on grammar induction using non-neural models where the entire dataset is used for training (Klein and Manning, 2002). However, this implies that the parsing results of PRPN-UP may not be generalizable in the way usually expected of machine learning evaluation results. Additionally, it is not obvious that the model should be able to learn to parse reliably: (1) Since the parser is trained as part of a language model, it makes pars-

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4998–5003
Brussels, Belgium, October 31 - November 4, 2018. ©2018 Association for Computational Linguistics

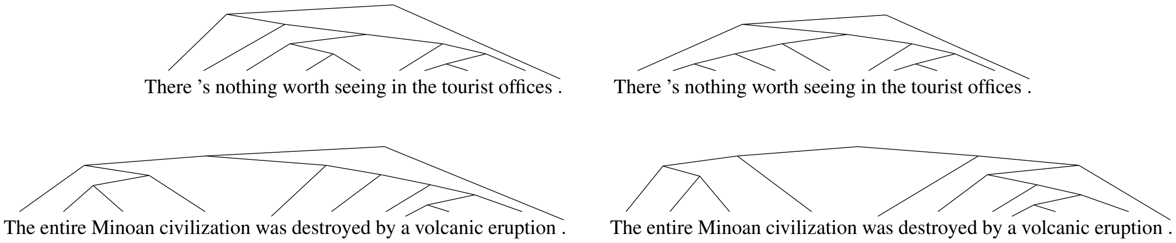
There 's nothing worth seeing in the tourist offices .
There 's nothing worth seeing in the tourist offices .
The entire Minoan civilization was destroyed by a volcanic eruption .
The entire Minoan civilization was destroyed by a volcanic eruption .

Figure 1: *Left* Parses from PRPN-LM trained on AllNLI. *Right* Parses from PRPN-UP trained on AllNLI (stopping criterion: parsing). We can observe that both sets of parses tend to have roughly reasonable high-level structure and tend to identify noun phrases correctly.

ing decisions greedily and with *no* access to any words to the right of the point where each parsing decision must be made (Collins and Roark, 2004); (2) As RNN language models are known to be insufficient for capturing syntax-sensitive dependencies (Linzen et al., 2016), language modeling as the downstream task may not be well-suited to latent tree learning.

In this replication we train PRPN on two corpora: The full WSJ, a staple in work on grammar induction, and AllNLI, the concatenation of the Stanford Natural Language Inference Corpus (SNLI; Bowman et al., 2015) and the Multi-Genre NLI Corpus (MultiNLI; Williams et al., 2018b), which is used in other latent tree learning work for its non-syntactic classification labels for the task of textual entailment, and which we include for comparison. We then evaluate the constituency trees produced by these models on the WSJ test set, full WSJ10,[1] and the MultiNLI development set.

Our results indicate that PRPN-LM achieves better parsing performance than PRPN-UP on both WSJ and WSJ10 even though PRPN-UP was tuned—at least to some extent—for parsing. Surprisingly, a PRPN-LM model trained on the large out-of-domain AllNLI dataset achieves the best parsing performance on WSJ despite not being tuned for parsing. We also notice that vocabulary size affects the language modeling significantly—the perplexity gets higher as the vocabulary size increases.

Overall, despite the relatively uninformative experimental design used in Shen et al. (2018), we find that PRPN is an effective model. It outperforms all latent tree learning baselines by large margins on both WSJ and MultiNLI, and performs competitively with symbolic grammar induction systems on WSJ10, suggesting that PRPN in particular and language modeling in general are a viable setting for latent tree learning.

2 Methods

PRPN consists of three components: (i) a *parsing network* that uses a two-layer convolution kernel to calculate the *syntactic distance* between successive pairs of words, which can form an indirect representation of the constituency structure of the sentence, (ii) a recurrent *reading network* that summarizes the current memory state based on all previous memory states and the implicit constituent structure, and (iii) a *predict network* that uses the memory state to predict the next token. We refer readers to the appendix and the original work for details.

We do not re-implement or re-tune PRPN, but rather attempt to replicate and understand the results of the work using the author's publicly available code.[2] The experiments on language modeling and parsing are carried out using different configurations of the model, with substantially different hyperparameter values including the size of the word embeddings, the maximum sentence length, the vocabulary size, and the sizes of hidden layers. PRPN-LM is larger than PRPN-UP, with embedding layer that is 4 times larger and the number of units per layer that is 3 times larger. We use both versions of the model in all our experiments.

We use the 49k-sentence WSJ corpus in two settings. To replicate the original results, we re-run an experiment with no train/test split, and for a clearer picture of the model's performance, we run it again with the train (Section 0-21 of WSJ), validation (Section 22 of WSJ), and test (Section 23

[1] A standard processed subset of WSJ used in grammar induction in which the sentences contain no punctuation and no more than 10 words.

[2] https://github.com/yikangshen/PRPN

Model	**Training Data**	**Stopping Criterion**	**Vocab Size**	**Parsing F1**				**Depth**	**Accuracy on WSJ by Tag**			
				WSJ10		**WSJ**		**WSJ**	**ADJP**	**NP**	**PP**	**INTJ**
				μ (σ)	**max**	μ (σ)	**max**					
PRPN-UP	AllNLI Train	UP	76k	67.5 (0.6)	68.6	36.9 (0.6)	38.0	5.8	29.3	62.0	31.6	*0.0*
PRPN-UP	AllNLI Train	LM	76k	66.3 (0.8)	68.5	38.3 (0.5)	39.8	5.8	28.7	65.5	32.7	*0.0*
PRPN-LM	AllNLI Train	LM	76k	52.4 (4.9)	58.1	35.0 (5.4)	**42.8**	6.1	**37.8**	59.7	**61.5**	**100.0**
PRPN-UP	WSJ Full	UP	15.8k	64.7 (3.2)	70.9	26.4 (1.7)	31.1	5.8	22.5	47.2	17.9	*0.0*
PRPN-UP	WSJ Full	LM	15.8k	64.3 (3.3)	70.8	26.3 (1.8)	30.8	5.8	22.7	46.6	17.8	*0.0*
PRPN-UP	WSJ Train	UP	15.8k	63.5 (3.5)	70.7	26.2 (2.3)	33.0	5.8	24.8	55.2	18.0	*0.0*
PRPN-UP	WSJ Train	LM	15.8k	62.2 (3.9)	70.3	26.0 (2.3)	32.8	5.8	24.8	54.4	17.8	*0.0*
PRPN-LM	WSJ Train	LM	10k	70.5 (0.4)	71.3	37.4 (0.3)	38.1	5.9	26.2	**63.9**	24.4	*0.0*
PRPN-LM	WSJ Train	UP	10k	66.1 (0.5)	67.2	33.4 (0.8)	35.6	5.9	33.0	57.1	18.3	*0.0*
300D ST-Gumbel	AllNLI Train	NLI	–	–	–	*19.0 (1.0)*	*20.1*	–	*15.6*	*18.8*	*9.9*	59.4
w/o Leaf GRU	AllNLI Train	NLI	–	–	–	22.8 (1.6)	25.0	–	18.9	24.1	*14.2*	51.8
300D RL-SPINN	AllNLI Train	NLI	–	–	–	*13.2 (0.0)*	*13.2*	–	*1.7*	*10.8*	*4.6*	50.6
w/o Leaf GRU	AllNLI Train	NLI	–	–	–	*13.1 (0.1)*	*13.2*	–	*1.6*	*10.9*	*4.6*	50.0
CCM	WSJ10 Full	–	–	–	71.9	–	–	–	–	–	–	–
DMV+CCM	WSJ10 Full	–	–	–	77.6	–	–	–	–	–	–	–
UML-DOP	WSJ10 Full	–	–	–	**82.9**	–	–	–	–	–	–	–
Random Trees	–	–	–	–	34.7	21.3 (0.0)	21.4	5.3	17.4	22.3	16.0	40.4
Balanced Trees	–	–	–	–	–	21.3 (0.0)	21.3	4.6	22.1	*20.2*	*9.3*	55.9
Left Branching	–	–	–	*28.7*	28.7	13.1 (0.0)	13.1	12.4	–	–	–	–
Right Branching	–	–	–	61.7	61.7	16.5 (0.0)	16.5	12.4	–	–	–	–

Table 1: Unlabeled parsing F1 results evaluated on full WSJ10 and WSJ test set broken down by training data and by early stopping criterion. The *Accuracy* columns represent the fraction of ground truth constituents of a given type that correspond to constituents in the model parses. Italics mark results that are worse than the random baseline. Underlining marks the best results from our runs. Results with RL-SPINN and ST-Gumbel are from Williams et al. (2018a), and are evaluated on the full WSJ. We run the model with 5 different random seeds to calculate the average F1. We use the model with the best F1 score to report ADJP, NP, PP, and INTJ. WSJ10 baselines are from Klein and Manning (2002, CCM), Klein and Manning (2005, DMV+CCM), and Bod (2006, UML-DOP). As the WSJ10 baselines are trained using additional information such as POS tags and dependency parser, they are not strictly comparable with the latent tree learning results.

of WSJ) splits. To compare PRPN to the models studied in Williams et al. (2018a), we also retrain it on AllNLI. As the MultiNLI test set is not publicly available, we follow Williams et al. (2018a) and use the development set for testing. The parsing evaluation code in the original codebase does not support PRPN-LM, and we modify it in our experiments only to add this support.

For early stopping, we remove 10k random sentences from the MultiNLI training set and combine them with the SNLI development set to create a validation set. Our AllNLI training set contains 280.5K unique sentences (1.8M sentences in total including duplicate premise sentences), and covers six distinct genres of spoken and written English. We do not remove the duplicate sentences. We train the model for 100 epochs for WSJ and 15 epochs for AllNLI. We run the model five times with random initializations and average the results from the five runs. The generated parses from the trained models with the best F1 scores and the pre-trained model that provides the highest F1 are available online.[3]

3 Experimental Results

Table 2 shows our results for language modeling. PRPN-UP, configured as-is with parsing criterion and language modeling criterion, performs dramatically worse than the standard PRPN-LM (a vs. d and e). However, this is not a fair comparison as the larger vocabulary gives PRPN-UP a harder task to solve. Adjusting the vocabulary of PRPN-UP down to 10k to make a fairer comparison possible, the PPL of PRPN-UP improves significantly (c vs. d), but not enough to match PRPN-LM (a vs. c). We also observe that early stopping on

[3] https://github.com/nyu-mll/PRPN-Analysis

	Model	Training Data	Stopping Criterion	Vocab Size	PPL Median
(a)	PRPN-LM	WSJ Train	LM	10k	**61.4**
(b)	PRPN-LM	WSJ Train	UP	10k	81.6
(c)	PRPN-UP	WSJ Train	LM	10k	92.8
(d)	PRPN-UP	WSJ Train	LM	15.8k	112.1
(e)	PRPN-UP	WSJ Train	UP	15.8k	112.8
(f)	PRPN-UP	AllNLI Train	LM	76k	797.5
(g)	PRPN-UP	AllNLI Train	UP	76k	848.9

Table 2: Language modeling performance (perplexity) on the WSJ test set, broken down by training data used and by whether early stopping is done using the parsing objective (UP) or the language modeling objective (LM).

parsing leads to incomplete training and a substantial decrease in perplexity (a vs. b and d vs. e). The models stop training at around the 13th epoch when we early-stop on parsing objective, while they stop training around the 65th epoch when we early-stop on language modeling objective. Both PRPN models trained on AllNLI do even worse (f and g), though the mismatch in vocabulary and domain may explain this effect. In addition, since it takes much longer to train PRPN on the larger AllNLI dataset, we train PRPN on AllNLI for only 15 epochs while we train the PRPN on WSJ for 100 epochs. Although the parsing objective converges within 15 epochs, we notice that language modeling perplexity is still improving. We expect that the perplexity of the PRPN models trained on AllNLI could be lower if we increase the number of training epochs.

Turning toward parsing performance, Table 1 shows results with all the models under study, plus several baselines, on WSJ test set and full WSJ10. On full WSJ10, we reproduce the main parsing result of Shen et al. (2018) with their UP model trained on WSJ without a data split. We also find the choice of parse quality as an early stopping criterion does not have a substantial effect and that training on the (unlabeled) test set does not give a significant improvement in performance. In addition and unexpectedly, we observe that PRPN-LM models achieve *higher* parsing performance than PRPN-UP. This shows that any tuning done to separate PRPN-UP from PRPN-LM was not necessary, and more importantly, that the results described in the paper can be largely reproduced by a unified model in a fair setting. Moreover, the PRPN models trained on WSJ achieves comparable results with CCM (Klein and Manning, 2002). The PRPN models are outperformed by DMV+CCM(Klein and Manning, 2005), and UML-DOP(Bod, 2006). However, these models use additional information such as POS and dependency parser so they are not strictly comparable with the PRPN models.

Model	Stopping Criterion	F1 wrt. LB	RB	SP	Depth
300D SPINN	NLI	19.3	36.9	70.2	6.2
w/o Leaf GRU	NLI	21.2	39.0	63.5	6.4
300D SPINN-NC	NLI	19.2	36.2	70.5	6.1
w/o Leaf GRU	NLI	20.6	38.9	64.1	6.3
300D ST-Gumbel	NLI	32.6	37.5	23.7	4.1
w/o Leaf GRU	NLI	30.8	35.6	27.5	4.6
300D RL-SPINN	NLI	95.0	13.5	18.8	8.6
w/o Leaf GRU	NLI	99.1	10.7	18.1	8.6
PRPN-LM	LM	25.6	26.9	45.7	4.9
PRPN-UP	UP	19.4	41.0	46.3	4.9
PRPN-UP	LM	19.9	37.4	**48.6**	4.9
Random Trees	–	27.9	28.0	27.0	4.4
Balanced Trees	–	21.7	36.8	21.3	3.9

Table 3: Unlabeled parsing F1 on the MultiNLI development set for models trained on AllNLI. *F1 wrt.* shows F1 with respect to strictly right- and left-branching (LB/RB) trees and with respect to the Stanford Parser (SP) trees supplied with the corpus; The evaluations of SPINN, RL-SPINN, and ST-Gumbel are from Williams et al. (2018a). SPINN is a supervised parsing model, and the others are latent tree models. Median F1 of each model trained with 5 different random seeds is reported.

Turning to the WSJ test set, the results look somewhat different: Although the differences in WSJ10 performance across models are small, the same is not true for the WSJ in terms of average F1. PRPN-LM outperforms all the other models on WSJ test set, even the potentially-overfit PRPN-UP model. Moreover, the PRPN models trained on the larger, out-of-domain AllNLI perform better than those trained on WSJ. Surprisingly, PRPN-LM tained on out-of-domain AllNLI achieves the best F1 score on WSJ test set among all the models we experimented, even though its performance on WSJ10 is the lowest of all. This mean that PRPN-LM trained on AllNLI is strikingly good at parsing longer sentences though its performance on shorter sentences is worse than other models. Under all the configurations we tested, the PRPN model yields much better performance than the baselines from Yogatama et al. (2017, called RL-SPINN) and Choi et al. (2018, called ST-Gumbel), despite the fact that the model was tuned exclusively for WSJ10 parsing. This suggests that PRPN is consistently effective at latent tree learning.

We also show detailed results for several specific constituent types, following Williams et al. (2018a). We observe that the accuracy for NP (noun phrases) on the WSJ test set is above 46% (Table 1) for all PRPN models, much higher than any of the baseline models. These runs also perform substantially better than the random baseline in the two other categories Williams et al. (2018a) report: ADJP (adjective phrases) and PP (prepositional phrases). However, as WSJ test set contains only one INTJ (interjection phrases), the results on INTJ are either 0.0% or 100%.

In addition, Table 3 shows that the PRPN-UP models achieve the median parsing F1 scores of 46.3 and 48.6 respectively on the MultiNLI dev set while PRPN-LM performs the median F1 of 45.7; setting the state of the art in parsing performance on this dataset among latent tree models by a large margin. We conclude that PRPN does acquire some substantial knowledge of syntax, and that this knowledge agrees with Penn Treebank (PTB) grammar significantly better than chance.

Qualitatively, the parses produced by most of the best performing PRPN models are relatively balanced (F1 score of 36.5 w.r.t balanced trees) and tend toward right branching (F1 score of 42.0 with respect to balanced trees). They are also shallower than average ground truth PTB parsed trees. These models can parse short sentences relatively well, as shown by their high WSJ10 performance.

For a large proportion of long sentences, most of the best performing models can produce reasonable constituents (Table 1). The best performing model, PRPN-LM trained on AllNLI, achieves the best accuracy at identifying ADJP (adjective phrases), PP (prepositional phrases), and INTJ (interjection phrases) constituents, and a high accuracy on NP (noun phrases). In a more informal inspection, we also observe that our best PRPN-LM and PRPN-UP runs are fairly good at pairing determiners with NPs as we can observe in Figure 1). Although lower level tree constituents appear random in many cases for both PRPN-LM and PRPN-UP, the intermediate and higher-level constituents are generally reasonable. For example, in Figure 1, although the parse for lower level constituents like *The entire Minoan* seem random, the higher-level constituents, such as *The entire Minoan civilization* and *nothing worth seeing in the tourist offices*, are reasonable.

4 Conclusion

In our attempt to replicate the grammar induction results reported in Shen et al. (2018), we find several experimental design problems that make the results difficult to interpret. However, in experiments and analyses going well beyond the scope of the original paper, we find that the PRPN model presented in that work is nonetheless robust. It represents a viable method for grammar induction and the first clear success for latent tree learning with neural networks, and we expect that it heralds further work on language modeling as a tool for grammar induction research.

Acknowledgments

This project has benefited from financial support to SB by Google and Tencent Holdings, and was partly supported by Samsung Electronics (Improving Deep Learning using Latent Structure). We thank Adina Williams, Katharina Kann, Ryan Cotterell, and the anonymous reviewers for their helpful comments and suggestions, and NVIDIA for their support.

References

Rens Bod. 2006. An All-Subtrees Approach to Unsupervised Parsing. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 865–872.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Eugene Charniak and Glen Carroll. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In *Proceedings of the AAAI Workshop on Statistically-Based NLP Techniques*, page 113.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2018. Learning to compose task-specific tree structures. In *Proceedings of the Thirty-Second Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI-18)*, volume 2.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain.*, pages 111–118.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of International Conference on Neural Networks (ICNN'96).*

Sepp Hochreiter and Jürgen Schmidhuber. 1996. Long Short Term Memory. *Memory*, (1993):1–28.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks.

Dan Klein and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, page 128.

Dan Klein and Christopher D. Manning. 2005. Natural language grammar induction with a generative constituent-context model. *Pattern Recognition*, 38(9):1407–1419.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *TACL*, 4:521–535.

Jean Maillard, Stephen Clark, and Dani Yogatama. 2017. Jointly learning sentence embeddings and syntax with unsupervised Tree-LSTMs. arXiv preprint 1705.09189.

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.

Yikang Shen, Zhouhan Lin, Chin wei Huang, and Aaron Courville. 2018. Neural language modeling by jointly learning syntax and lexicon. In *International Conference on Learning Representations*.

Noah A. Smith and Jason Eisner. 2005. Guiding unsupervised grammar induction using contrastive estimation. In *Proceedings of IJCAI Workshop on Grammatical Inference Applications*, pages 73–82.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Ng, and Chris Manning. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 28th International Conference on Machine Learning*, pages 129–136.

Adina Williams, Andrew Drozdov, and Samuel R. Bowman. 2018a. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association for Computational Linguistics (TACL).*

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018b. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL).*

Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. 2017. Learning to Compose Words into Setences with Reinforcement Learning. *Proceedings of the International Conference on Learning Representations*, pages 1–17.