Rule Augmented Unsupervised Constituency Parsing

Atul Sahay [1*], **Anshul Nasery** [1*], **Ayush Maheshwari** [1], **Ganesh Ramakrishnan** [1], **Rishabh Iyer** [2]
[1]Department of CSE, IIT Bombay, India
[2] The University of Texas at Dallas
{atulsahay, anshuln, ayusham, ganesh}@cse.iitb.ac.in
rishabh.iyer@utdallas.edu

Abstract

Recently, unsupervised parsing of syntactic trees has gained considerable attention. A prototypical approach to such unsupervised parsing employs reinforcement learning and auto-encoders. However, no mechanism ensures that the learnt model leverages the well-understood language grammar. We propose an approach that utilizes very generic linguistic knowledge of the language present in the form of syntactic rules, thus inducing better syntactic structures. We introduce a novel formulation that takes advantage of the syntactic grammar rules and is independent of the base system. We achieve new state-of-the-art results on two benchmarks datasets, MNLI and WSJ.[1]

1 Introduction

Syntactic parse trees have demonstrated their importance in several downstream NLP applications such as machine translation (Eriguchi et al., 2017; Zaremoodi and Haffari, 2018), natural language inference (NLI) (Choi et al., 2018), relation extraction (Gamallo et al., 2012) and text classification (Tai et al., 2015). Based on linguistic theories that have promoted the usefulness of tree-based representation of natural language text, tree-based models such as Tree-LSTM have been proposed to learn sentence representations (Socher et al., 2011). Inspired by the Tree-LSTM based models, many approaches were proposed do not require parse tree supervision (Yogatama et al., 2017; Choi et al., 2018; Maillard et al., 2019; Drozdov et al., 2019). However, (Williams et al., 2018; Sahay et al., 2021) have shown that these methods cannot learn meaningful semantics (not even simple grammar), though they perform well on NLI tasks. Recently, there has been surge in approaches using weak supervision in the form of rules for various tasks such as sequence classification (Safranchik et al., 2020), text classification (Chatterjee et al., 2020; Maheshwari et al., 2020), *etc.* These approaches have demonstrated the importance of external knowledge in both unsupervised and supervised setup. To the best of our knowledge, previous works on syntactic parse tree has not utilized such external information. In this paper, we propose an approach that leverages linguistic (and potentially domain agnostic) knowledge in the form of explicit syntactic grammar rules while building upon a state of the art, deep and unsupervised inside-outside recursive autoencoder (DIORA; (Drozdov et al., 2019)). DIORA is an unsupervised model that uses inside-outside dynamic programming to compose latent representations from all possible binary trees. We extend DIORA and propose a framework that harness grammar rules to learn constituent parse trees. We use context free grammar (CFG) productions for English language (like NP $\rightarrow$VP NP, PP $\rightarrow$IN NP, *etc*) as rules. Note that the construction of such a rule set is a one time effort and our method is independent of any underlying dataset. The rule sets used are available in our github repository.
Summarily, our main contributions are : (a) a framework (*cf.*, Section 3) that uses (potentially domain agnostic), off-the-shelf CFG to learn to produce constituent parse trees (b) two rule-aware loss functions (*cf.*, Section 3.1) that maximize some form of agreement between the unsupervised model and the rule-based model (c) experimental analysis (*cf.*, Section 4), demonstrating improvements on unsupervised constituency parsing over previous state-of-the art by over **3**% on two benchmark datasets.

2 Background and Related Work

A brief survey of latent tree learning models is covered in (Williams et al., 2018). Several prior works have explored the unsupervised learning of

[*]Equal contribution
[1]The source code of the paper is available at https://github.com/anshuln/Diora_with_rules.

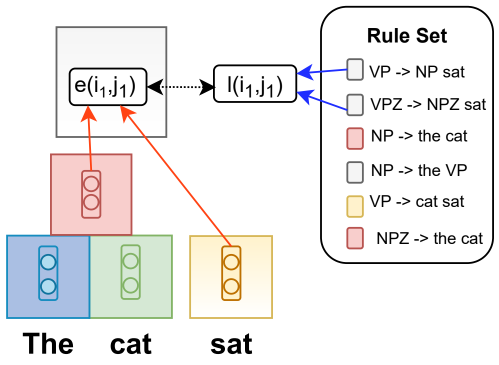


Figure 1: For the input 'The cat sat', DIORA computes $e(i_1, j_1)$ compatibility score for each pair of neighboring constituents. $l(i_1, j_1)$ is computed using triggered rules for each span and it interacts with the compatibility score in our loss function as explained in Section 3.1.

constituency trees (Brill et al., 1990; Ando and Lee, 2000) using dependency parsers (Klein and Manning, 2004) and inside-outside parsing algorithm (Drozdov et al., 2019). Recently, (Drozdov et al., 2019) proposed an unsupervised latent chart tree parsing algorithm, *viz.*, DIORA, that uses the inside-outside algorithm for parsing and has an autoencoder-based neural network trained to reconstruct the input sentence. DIORA is trained end to end using masked language model via word prediction. As of date, DIORA is the state-of-the-art approach to unsupervised constituency parsing.

Exploiting additional semantic and syntactic information that acts as a source of additional guidance rather than the primary objective function has been discussed since 1990s (Sun et al., 1993). Recently, Kim et al. (2019b) proposed to learn CFG rules and their probabilities by the parameterizing terminal or non-terminal symbols with neural networks. However, our approach leverages predefined language CFG rules and provisions for augmenting an existing (state-of-the-art) inside-outside algorithm with such external knowledge.

More specifically, we augment DIORA (Drozdov et al., 2019) with CFG rules to reconstruct the input by exploiting syntactic information of the language. We next provide some technical details of the inside-outside algorithm of DIORA.

2.1 DIORA

DIORA learns constituency trees from the raw input text using an unsupervised training procedure that operates like a masked language model or denoising autoencoder. It encodes the entire input sequence into a single vector analogous to the encoding step in an autoencoder. Thereafter, the decoder is trained to reconstruct and reproduce each input word. We next describe the inside and the outside pass of DIORA, respectively. *Inside Pass:* Given an input sentence with T tokens $x_0, x_1, x_2 \dots x_{T-1}$, DIORA computes a *compatibility* score e and a *composition* vector $\overline{a}$ for each pair of neighboring constituents i and j. It composes a vector $\overline{a}$ weighing over all possible pairs of constituents of i and j: $\overline{a}(k) = \sum_{i,j\epsilon\{k\}} e(i,j)a(i,j)$ & $\overline{e}(k) = \sum_{i,j\epsilon\{k\}} e(i,j)\hat{e}(i,j)$ The composition vector, $\overline{a}(k)$ is a weighted sum of all possible constituent pairs, k. Here, $\hat{e}$ is a bilinear function of the vectors from neighboring spans, $\overline{a}(i)$ and $\overline{a}(j)$. Composition vector $\overline{a}(k)$ is learnt using a TreeLSTM or multi-layer neural network (MLP).

Outside Pass: The outside pass of DIORA computes an outside vector $\overline{b}(k)$ representing the constituents not in $x_{i:j}$. It computes the values for a target space (i, j) recursively from its sibling $(j+1, k)$ and outside spans $(0, i-1)$ and $(k+1, T-1)$.

Training and Inference: DIORA is trained end to end using masked language model via word prediction. The missing token x_i is predicted from the outside vector $\overline{b}(k)$. The training objective uses reconstruction based max-margin loss to predict the original input x_i:

$$L_{rec} = \sum_{i=0}^{T-1} \sum_{i^*=0}^{N-1} \max(0, 1 - \overline{b}(i).\overline{a}(i) + \overline{b}(i).\overline{a}(i^*))$$

The chart filling procedure of DIORA is used to extract binary unlabeled parse trees. It uses the CYK algorithm to find the maximal scoring tree in a greedy manner. For each cell of the parse table, the algorithm computes the span (i, j) with the maximal net compatibility score, computed recursively by summing the maximum compatibility score $e(a, b)$ for each constituent of the span.

3 Our Approach to Rule Augmentation

Our goal is to learn to produce constituency parse trees using input sentences alone and in the absence of ground truth parse trees. We introduce a rule-augmented unsupervised model that leverages generic (potentially domain agnostic) production rules of the language grammar to infer constituency trees. Since most grammar rules for constituency parsing are generic, designing them can be a one-time effort, while being able to leverage their benefits across domains as background knowledge (as we will see in our experiments in Section 4). As described in Section 2.1, the induction of latent trees in DIORA is based on a CYK-like parsing algorithm that uses the compatibility scores $e(i,j)$ at each cell to merge two constituents in the final tree. We impart supervision through the production rules of English language grammar.

For each sentence, we associate CFG production rules with constituents i and j in a CYK parse table format. We curate a set of domain-agnostic rules of the form $X \rightarrow YZ$ and a dictionary of the form $X \rightarrow x$, where X, Y, Z are non-terminals while x is a terminal. Concretely, X represents constituent tags such as S, NP, VP, etc., while x represents words in the vocabulary. Using the CYK parsing algorithm on our rule set and each sentence, we first determine which rules are triggered at each cell for a particular sentence. Whenever a rule r is triggered for a span (i,j), we weakly associate label $\delta_{(i,j)}(r) = 1$ otherwise 0. We use these weak labels to guide the rule scores $l(i,j)$ for the constituents. Compatibility score observed for the span (i,j) is defined as :

$$l(i,j) = \frac{\exp\left(\sum_{p=0}^{P} r_p \delta_{(i,j)}(p)\right)}{\sum_{(a,b)\in\{k\}} \exp\left(\sum_{p=0}^{P} r_p \delta_{(a,b)}(p)\right)}$$

where r_p are the *learned* weights associated with each of the production rules and P is the total number of rules. The score sums to 1 over all spans belonging to a particular cell in the CYK parse table. Intuitively, we aim to align $e(i,j)$ and $l(i,j)$ score to maximize the agreement between model and rules. We note that we use rules only to augment the training objective, and our inference procedure is identical to that of DIORA.

3.1 Training Objective

We learn a model that minimizes the overall loss L that is a composition of the reconstruction loss L_{rec} and the rule based agreement loss or L_{rule}: $L = L_{rec} + \lambda L_{rule}$. We propose two alternatives for the loss function L_{rule}.

Cross entropy (CE) - For each cell k in the CYK parse table, this loss (CE) tries *to match the distribution (score)* of $e(i,j)$ induced by DIORA with the distribution $l(i,j)$ induced by the background knowledge:

$$L_{ce} = \sum_{k} \sum_{(i,j)\in\{k\}} -l(i,j)\log(e(i,j))$$

Model	**F1**	δ
300D Gumbel Tree-LSTM	25.2	4.2
w/o Leaf GRU	29.0	4.7
300D RL-SPINN	19.0	8.6
w/o Leaf GRU	18.2	8.6
Structural Attentive(Gumble Tree LSTM)	31.3	4.7
w/o Leaf GRU	31.0	5.3
300D SPINN	**74.5**[†]	6.2
w/o Leaf GRU	65.7[†]	6.4
DIORA with PP	58.3	5.6
Ours: Rule augmented (HR) + RL + PP	60.5	5.7
Ours: Rule augmented (HR) + CE + PP	59.0	5.6
Ours: Rule augmented (AR) + RL + PP	60.3	5.7
Ours: Rule augmented (AR) + CE + PP	**61.7**	5.7

Table 1: F1-scores of trees wrt ground truth on the MultiNLI development set. The depth (δ) is the average tree height. All reported numbers are maximum F1-score. PP refers to post-processing heuristic. HR and AR refer to the training rule sets as per Sec 4.2. RL and CE refer to the losses from section 3.1. [†] indicates scores reported by (Williams et al., 2018) for a fully supervised model.

Ranking Loss (RL) - We recall from Section 2.1 that the CYK algorithm finds the maximal scoring tree in a greedy manner based on the highest compatibility score $e(i,j)$ among all spans. Since the final parse tree output by DIORA relies only on the *relative order* of the $e(i,j)$ to decide which span to merge, we propose an alternative rule-based loss that aims *to match the relative order* induced by compatibility scores $e(i,j)$ of DIORA at each cell with the order induced by the scores of the rules $l(i,j)$ at that cell. We achieve this through a pairwise ranking loss defined as

$$L_{rank} = \sum_{\mathrm{k}} \sum_{\substack{(i,j)\\(i',j')}\in\{\mathrm{k}_{trig}\}} \left(\Delta^{l}_{\substack{(i,j),\\(i',j')}} - \Delta^{e}_{\substack{(i,j),\\(i',j')}}\right)^2$$

where $\{\mathrm{k}_{trig}\} = \{(i,j) \in \{k\} | \sum_{p=1}^{P} \delta_{i,j}(p) \neq 0\}$, $\Delta^{f}_{(i,j),(i',j')} = \tanh(f(i,j) - f(i',j'))$ and p is index into the rule set. The set $\{k_{trig}\}$ consists of all spans which have at least one rule triggered in its cell. In cases where our rule set is not extensive enough, we would like our model's compatibility score to rely more on the reconstruction loss, and $\{\mathrm{k}_{trig}\}$ ensures that a sparse rule set does not lead to bad performance.

4 Experiments

We evaluate our rule augmented model and compare it against baselines on the tasks of unsuper-

Model	F1	δ
LB	13.1	12.4
RB	16.5	12.4
Random	21.4	5.3
Balanced	21.3	4.6
RL-SPINN (Choi et al., 2018)	13.2	-
ST-Gumbel - GRU (Yogatama et al., 2017)	22.8 ±1.6	-
PRPN-UP (Shen et al., 2018a)	38.3	5.9
PRPN-LM	35.0	6.2
ON-LSTM (Shen et al., 2018b)	47.7	5.6
DIORA	48.9	8.0
PRPN-UP+PP	45.2	6.7
PRPN-LM+PP	42.4	6.3
DIORA+PP	55.7	8.5
Neural PCFG (Kim et al., 2019a) *	50.8	-
Compound PCFG (Kim et al., 2019b)*	55.2	-
300D SPINN	**59.6**†	-
Ours: Rule augmented (HR) + RL + PP	56.5	7.1
Ours: Rule augmented (HR) + CE + PP	55.3	7.2
Ours: Rule augmented (AR) + RL + PP	55.9	7.1
Ours: Rule augmented (AR) + CE + PP	**58.3**	7.3

Table 2: Performance on WSJ test set for binary constituency parsing including punctuation characters. HR and AR refers to handcrafted and automated rules respectively. RL and CE are rule loss and cross entropy loss respectively. (δ) is the average tree height. PP refers to post-processing heuristic. † indicates scores reported by (Williams et al., 2018) for a fully supervised model. * are reported by (Kim et al., 2019a).

vised parsing, unsupervised segment recall, and phrase similarity.

4.1 Data

We evaluate our model on two data sets: The Wall Street Journal (WSJ) and MultiNLI. WSJ is an extraction of PennTree Bank (Marcus et al., 2002) containing human-annotated constituency parse trees. MultiNLI consists of Stanford generated parse trees (Manning et al., 2014) as the ground truth. MultiNLI is originally designed for evaluating NLI tasks, but is often also utilized to evaluate constituency parse trees. We train on the complete NLI dataset, which is a composition of the MultiNLI and SNLI train sets. We evaluate model performance on the MultiNLI dev set and WSJ test set (split 23) following the experimental setting and evaluation metrics in (Drozdov et al., 2019). Further details are provided in the appendix. We initialize our model with the trained weights of DIORA and evaluate on unsupervised constituency parsing and segment recall. We also perform the post-processing (PP) of generated trees by attaching the trailing punctuation to the root node, exactly as carried out by (Drozdov et al., 2019).

4.2 Rule Set

We consider two rule-sets: (i) **Set of Handcrafted Rules (HR)** consists of 2500 human created CNF production rules ii) To assess robustness of the rule-augmented method to the preciseness of the rule set, we present comparison by instead using a **set of Automated Rules' (AR)** which consists of the 2500 most frequently occurring CNF production rules extracted from the trees of automatically (using the Stanford CoreNLP parser) parsed SNLI corpus. Further details about these rule sets can be found in the appendix. We also use a train-set specific dictionary containing the POS (part-of-speech) tags of words in the training vocabulary for the terminal CFG productions for CYK parsing.

4.3 Unsupervised Parsing

In Tables 1 and 2, we present comparison between different approaches on the MultiNLI dev set and WSJ test set. We observe that our rule augmented approach outperforms the state of the art with respect to the max-F1 score. registering a maximum increase of 3.4 and 3.1 F1 points over DIORA respectively. The HR trained models outperform DIORA on both datasets, demonstrating that rule creation is indeed a one-time process and independent of domain. We also report parsing scores of a fully supervised model SPINN from (Williams et al., 2018) as an upper bound, and RL-SPINN (Choi et al., 2018), a distantly supervised model.

4.4 Constituency Segment Recall

In Table 3, we present the breakdown of constituent recall across the 6 most common types. Our approach achieves the highest recall across all the types and is the only model to perform effectively on SBAR and NP. Unlike other approaches, our approach consistently close to or the best recall score. We observe that rule augmentation using HR is more beneficial than AR with respect to precise evaluation measures such as Constituency, Segment Recall and Phrase Recall but yields smaller improvements than AR with respect to looser evaluation measures such as max F1 of Unsupervised Parsing. This can be possibly attributed to our observation that the extracted (most frequent) rules from SNLI, have (around 25%) higher coverage on the training set than HR, but appear to be semantically less precise.

Model	SBAR	NP	VP	PP	ADJP	ADVP
LB †	5%	11%	0%	5%	2%	8%
RB †	68%	24%	71%	42%	27%	38%
Random †	8%	23%	12%	18%	23%	28%
Balanced †	7%	27%	8%	18%	27%	25%
PRPN-UP (Shen et al., 2018a)	55.4%	59.8%	31.6%	60.2%	36.0%	50%
PRPN-LM	40.3%	68.7%	39.3%	49.7%	34.2%	39.2%
DIORA	61.3%	76.7%	62.8%	59.5%	60.4%	69.3%
PRPN (tuned)†	50%	59%	46%	57%	44%	32%
ON (tuned) (Shen et al., 2018b)	51%	64%	41%	54%	38%	31%
Neural PCFG (Kim et al., 2019a)	52%	71%	33%	58%	32%	45%
Compound PCFG (Kim et al., 2019b)	56%	74%	41%	68%	40%	52%
Ours: Rule augmented (HR)+ RL	**71.1%**	77.2%	65.8%	59.4%	**62.9%**	69.5%
Ours: Rule augmented (HR)+ CE	68.3%	75.4%	66.5%	**60.5%**	61%	**70.8%**
Ours: Rule augmented (AR)+ RL	71%	76.4%	**69.1%**	58.6%	61%	64.8%
Ours: Rule augmented (AR)+ CE	70%	**77.5%**	67%	58.6%	**62.2%**	70%

Table 3: Segment recall from WSJ by phrase type; † are reported by Kim et al. (2019a).

Model	CONLL 2000			CONLL 2012	
	P@1	P@10	P@100	P@1	P@10
DIORA	0.974	0.969	0.943	0.815	0.759
Ours: Rule augmented(AR)+ RL	0.976	0.968	0.941	0.813	0.755
Ours: Rule augmented(HR)+ CE	**0.978**	**0.97**	0.941	0.786	0.717
Ours: Rule augmented(AR)+ CE	0.970	0.965	0.937	0.809	0.745
Ours: Rule augmented(HR)+ RL	0.976	0.970	**0.944**	**0.824**	**0.760**

Table 4: Phrase similarity scores on CoNLL2000 and CoNLL 2012 tasks.

4.5 Phrase Similarity

We also employed the phrase similarity strategy followed by (Drozdov et al., 2019). Phrase Similarity scores measures the models capability to learn meaningful representation for spans of the text. Generally, most models focus more on generating the tokens representation and then use some ad-hoc arithmetic operations to generate representation for the larger spans of text thus losing the essence of the context that ties the words of the span.

To evaluate on the phrase similarity task we consider two data sets of labeled phrases: 1) CoNLL 2000 (Tjong Kim Sang and Buchholz, 2000), which is a shallow parsed dataset and contains spans of verb phrases, noun phrases, preposition phrases *etc*., and 2) CoNLL 2012 (Pradhan et al., 2012) which is a named entity dataset containing 19 different entity types. For the evaluation routine, we first generated the phrase representation of labeled spans whose length is greater than one. Cosine similarity is then used to obtain the similarity score of it with respect to all other labeled spans. We then calculate if the label for that query span matches the labels for each of the K most similar other spans in the dataset.In Table 4 we report precision@K for both datasets and various values of K. The baseline numbers are reported using the weights of DIORA provided by the authors.

5 Conclusion

In this work, we leverage linguistically grounded and domain agnostic CFG rules for language to induce parse trees and representations of constituent spans. We show that our approach augmented with generic, linguistically grounded grammatical rules, is easily able to outperform previous methods on constituency parsing and obtain higher segment recall.

Acknowledgements

We thank anonymous reviewers for providing constructive feedback. Ayush Maheshwari is supported by a Fellowship from Ekal Foundation (www.ekal.org). We are also grateful to IBM Research, India (specifically the IBM AI Horizon Networks - IIT Bombay initiative) for their support and sponsorship.

References

Rie Kubota Ando and Lillian Lee. 2000. Mostly-unsupervised statistical segmentation of japanese: Applications to kanji. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 241–248. Citeseer.

Eric Brill, David Magerman, Mitchell Marcus, and Beatrice Santorini. 1990. Deducing linguistic structure from the statistics of large corpora. In *Proceedings of the 5th Jerusalem Conference on Information Technology, 1990.'Next Decade in Information Technology'*, pages 380–389. IEEE.

Oishik Chatterjee, Ganesh Ramakrishnan, and Sunita Sarawagi. 2020. Robust data programming with precision-guided labeling functions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3397–3404.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2018. Learning to compose task-specific tree structures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Andrew Drozdov, Pat Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. 2019. Unsupervised latent tree induction with deep inside-outside recursive autoencoders. In *North American Association for Computational Linguistics*.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 72–78.

Pablo Gamallo, Marcos Garcia, and Santiago Fernández-Lanza. 2012. Dependency-based open information extraction. In *Proceedings of the joint workshop on unsupervised and semi-supervised learning in NLP*, pages 10–18.

Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang-goo Lee. 2019a. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction. In *International Conference on Learning Representations*.

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

Dan Klein and Christopher D Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd annual meeting of the association for computational linguistics (ACL-04)*, pages 478–485.

Ayush Maheshwari, Oishik Chatterjee, KrishnaTeja Killamsetty, Rishabh K. Iyer, and Ganesh Ramakrishnan. 2020. Data programming using semi-supervision and subset selection. *CoRR*, abs/2008.09887.

Jean Maillard, Stephen Clark, and Dani Yogatama. 2019. Jointly learning sentence embeddings and syntax with unsupervised tree-lstms. *Natural Language Engineering*, 25(4).

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Mitchell Marcus, Mary Marcinkiewicz, and Beatrice Santorini. 2002. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19:313–330.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Esteban Safranchik, Shiying Luo, and Stephen Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5570–5578.

Atul Sahay, Ayush Maheshwari, Ritesh Kumar, Ganesh Ramakrishnan, Manjesh Kumar Hanawal, and Kavi Arya. 2021. Unsupervised learning of explainable parse trees for improved generalisation. *CoRR*, abs/2104.04998.

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

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2018b. Ordered neurons: Integrating tree structures into recurrent neural networks. In *International Conference on Learning Representations*.

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 151–161.

GZ Sun, CL Giles, HH Chen, and YC Lee. 1993. The neural network pushdown automation: model, stack and learning simulations.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566.

Erik F Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 127–132.

Adina Williams, Andrew Drozdov*, and Samuel R Bowman. 2018. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association for Computational Linguistics*, 6:253–267.

D Yogatama, P Blunsom, C Dyer, E Grefenstette, and W Ling. 2017. Learning to compose words into sentences with reinforcement learning. In *5th International Conference on Learning Representations (ICLR 2017)*. International Conference on Learning Representations.

Poorya Zaremoodi and Gholamreza Haffari. 2018. Incorporating syntactic uncertainty in neural machine translation with a forest-to-sequence model. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1421–1429.

Appendix

A Training and evaluation details

We use the multi-layered neural network model provided by the authors of (Drozdov et al., 2019) to initialize our model's weights. We have a learning rate of 10^{-4} and a batch size of 32. Using these settings we train our models for 500000 steps. We use weights $\lambda_{ranking} = 10^{-1}$ and $\lambda_{ce} = 1.0$ for the rule based losses. All other model parameters are same as the ones set in (Drozdov et al., 2019). We run all our experiments on Nvidia RTX 2080Ti GPUs 12 GB RAM over Intel Xeon Gold 5120 CPU having 56 cores and 256 GB RAM. It takes about 2 days to train the model on NLI data.

For evaluation, we have reported the tree F-1 score for MNLI dev and WSJ test set. The metric computes the F-1 score for each tree based on the constituent spans induced in the predicted tree against the constituent spans in the ground truth. We further binarize the WSJ test set using the Stanford CoreNLP Parser and report scores on unlabelled binary trees.

We find that training with AR helps us achieve better results on both MNLI as well as on WSJ. This could be because extracted rules from SNLI have wider coverage on the training set than HR resulting in a stronger training signal and better performance. Further, our ranking loss performs better for HR extracted rules, indicating its efficacy with non-extensive rule sets, i.e. in the cases where the training signal is not rich. In such cases when some cells may not have any triggering rules, the ranking loss ensures that the model's decision is guided by the reconstruction loss.

We also find that the generic background knowledge of English grammar (HR) helps the model to better chunk constituents that are rarer (e.g. SBAR), while dataset-specific rules (AR) might benefit its overall tree structures more, leading to higher unlabelled F1 scores.

B Rule sets

In this work we utilise two distinct rule sets - (i) The first rule set (HR) consists of 2500 human created CNF production rules ii) the other set (AR) consisted of 2500 most frequently occurring CNF production rules extracted from the trees of automatically parsed SNLI corpus. All rules in both these sets consist only of non-terminals. The rules in (HR) come from observing human annotated parse trees from the PTB train set and consists of 2500 rules in the Chomsky Normal Form. The rules in (AR) are programatically extracted from the parse trees generated by running the Stanford Parser on the SNLI train set. We only retain the 2500 most frequently occurring productions from the set to match the size of the HR set. We note however that these rules have a higher coverage on the train data. We also provides `rules-AR.txt` and `rules-HR.txt` in the github repository.

In our training procedure, we aim to learn a weight r_p for each production rule p in our train set. Table 5 shows the top 10 most important (i.e. the ones with the highest r_p) rules from the grammar as determined by our models.

C Learnt Trees

In this section we present examples of trees induced by DIORA and our model. The first row of fig. 1 shows an example where our tree matches the ground truth exactly while DIORA does not, and the second row of fig. 1 shows an example where both models do not provide exact matches, but our model is able to capture the syntax better.

Model	Top Rules
AR ranking loss	NP —>PRP NP|CD-JJ-VBN-NNS>
	NP —>PRP NP|<NNP-NNP-NN>
	S —>S S|<CC-SINV-.>
	ADVP —>IN ADVP|<CC-JJ>
	VP —>VBN VP|<“-NP-”-PP>
	NP —>NP NP|<NN-NN-”>
	PP —>IN ,
	NP —>NP NP|<ADJP-PP-SBAR>
	NP —>NP NP|<NNS-S>
	PP —>“ PP|<IN-NP>
HR cross entropy	NP —>CD NNS
	S-CLR —>VP
	PP-PRP —>IN NP
	QP|<CD-TO-CD-CD>—>CD QP|<TO-CD-CD>
	S-2 —>NP S-2|<VP-.>
	VP —>VB VP|<NP-ADVP-S>
	NP|<,-”-SBAR>—>, NP|<”-SBAR>
	S —>NP S|<NP-VP-.>
	NP|<JJS-NNS>—>JJS NNS
	S-2|<VP-.>—>VP .
HR ranking loss	NP —>PRP NP|<NNP-CD-NN>
	NP —>DT NP|<JJ-NN-NN>
	S|<NP-VP-.—>—>NP S|<VP-.—>
	VP|<NP-S>—>NP S
	NP|<NNP-NNP-NNP-NN-NN>—>NNP NP|<NNP-NNP-NN-NN>
	NP —>JJ NP|<NN-POS>
	VP|<CC-,-VP>—>CC VP|<,-VP>
	NP —>NNP NP|<CC-NNS>
	NP|<:-SBAR>—>: SBAR
	ADJP —>$ CD
AR cross entropy	NP —>DT NP|<JJ-NNP-NNP-POS>
	VP —>VB VP|<NP-PRT>
	S|<S-:>—>S :
	VP —>VBZ VP
	PRN —>, PRN|<CC-PP>
	VP|<CC-VBG-NP-PP>—>CC VP|<VBG-NP-PP>
	NP|<,-NP-,-VP-.>—>, NP|<NP-,-VP-.>
	S|<PP-,-VP-.>—>PP S|<,-VP-.>
	PP —>RB PP|<CC-RB-NP>
	NP —>JJ NP|<NNP-NNP>

Table 5: Rules with the highest weights as learnt by our models

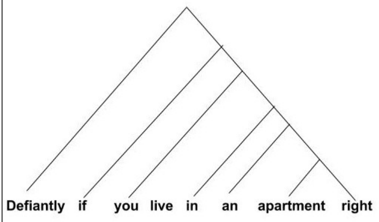
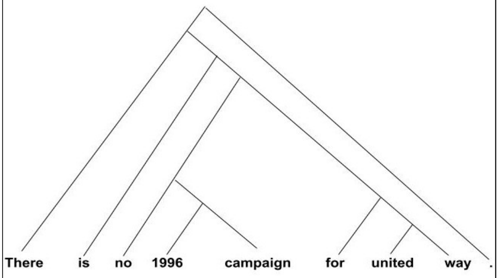
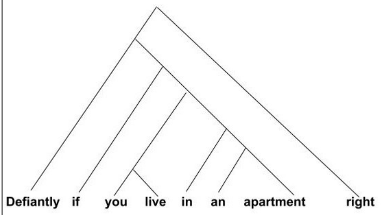
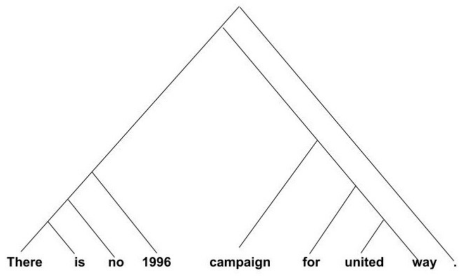
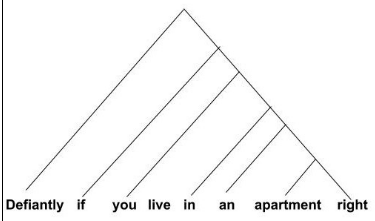
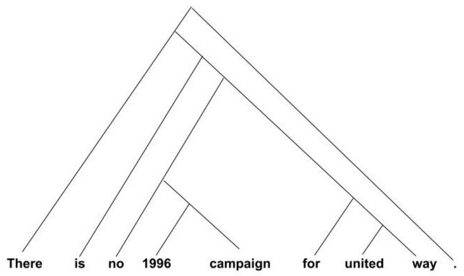
Our Results		
DIORA's Results		
Ground Truth		

Figure 2: Comparison of induced trees by our model and DIORA with the ground truth trees