Probability Distribution Collapse: A Critical Bottleneck to Compact Unsupervised Neural Grammar Induction

Jinwook Park and **Kangil Kim***
AI Graduate School
Gwangju Institute of Science and Technology
jinwookpark@gm.gist.ac.kr, kangil.kim.01@gmail.com

Abstract

Unsupervised neural grammar induction aims to learn interpretable hierarchical structures from language data. However, existing models face an expressiveness bottleneck, often resulting in unnecessarily large yet underperforming grammars. We identify a core issue, *probability distribution collapse*, as the underlying cause of this limitation. We analyze when and how the collapse emerges across key components of neural parameterization and introduce a targeted solution, *collapse-relaxing neural parameterization*, to mitigate it. Our approach substantially improves parsing performance while enabling the use of significantly more compact grammars across a wide range of languages, as demonstrated through extensive empirical analysis.

1 Introduction

Formal grammars, such as context-free grammars, represent the structure of languages by formalizing the hierarchical organization of components of natural languages into human-understandable rules. These rules enable logical interpretation of top-down or inclusion relationships within languages and support the control and use of hierarchical structures based on them. This property has a potential to enable neural models to understand the structures, and there have been steady efforts to utilize grammars in natural language processing and computer vision area (Zhao and Titov, 2020; Hong et al., 2021; Williams, 2023) via inducing high-quality grammars (Shen et al., 2019; Dyer et al., 2016; Kim et al., 2019b; Wang et al., 2019; Shen et al., 2021; Drozdov et al., 2019, 2020; Kim et al., 2019a). In particular, Neural PCFGs (N-PCFGs) (Kim et al., 2019a) have successfully learned probabilistic context-free grammars (PCFGs) in an unsupervised setting by leveraging neural parameterization to estimate rule probability distributions from vector representations of symbols.

Among various studies based on N-PCFGs (Yang et al., 2021b, 2022), some have achieved higher performance by enhancing the expressive power of N-PCFGs. This improvement is grounded in the findings of conventional studies that did not utilize neural networks (Petrov et al., 2006), which demonstrated high parsing performance through an increased number of symbols, as well as in experimental results indicating that a greater number of grammar parameters can improve model performance (Buhai et al., 2019).

However, before increasing grammar capacity for enhancing performance, can we be certain that current neural network architectures are truly designed to fully utilize the given capacity? This question is critical for interpretability, which requires both compact and accurate grammars. Yet, to date, the literature lacks analytical studies that directly address this important issue.

We introduce *Probability Distribution Collapse* (PDC) as a key bottleneck in constructing compact grammars. PDC implies indistinguishable probability distributions across many symbols, limiting grammar expressiveness even with a large number of symbols. We analyze this phenomenon in the neural parameterization process that maps symbol embeddings to rule probabilities, along with its training dynamics, and identify three underlying causes: 1) small dimension of embeddings and shallow neural network layers, 2) entangled scales of children embeddings in generating the probability, and 3) gradient explosion and dying ReLU collapsing symbol representations and corresponding probability projection. To this end, we propose *collapse-relaxing neural parameterization* (CRNP) equipped to a recent work, N-PCFGs with parse-focusing (Park and Kim, 2024), to address the causes via simple and comprehensive solutions. In empirical validation on constituent parsing, our ap-

*Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 33381–33392
November 4-9, 2025

proach improves the upper bound of accuracy while maintaining the same grammar size under structural supervision, demonstrating its effectiveness without interference from the implicit behaviors of unsupervised neural grammar induction (UNGI). Furthermore, in UNGI comparisons, our approach achieves strong performance even with highly compact grammars across multilingual benchmarks, including Penn TreeBank (PTB, English), Chinese TreeBank (CTB, Chinese), and SPMRL (Basque, French, German, Hebrew, Hungarian, Korean, Polish, Swedish). Our code is publicly available at https://github.com/GIST-IRR/CRNP.

Our contributions are summarized as follows:

- We introduce *probability distribution collapse* as a bottleneck to inducing more compact grammars in an unsupervised setting.
- We investigate its causes within neural parameterization and propose a simple, yet effective solution, termed *collapse-relaxing neural parameterization*.
- We provide extensive validation of our approach, demonstrating improvements in both upper bound and practical accuracy, as well as grammar compactness, on constituent parsing tasks for English and multilingual benchmarks.

2 Background

Notations of Probabilistic Context-Free Grammar In this paper, we use the following notation for a PCFG $G = (S, N, P, \Sigma, R, \Pi)$: the root symbol S, finite sets of nonterminals N, preterminals P, terminal words Σ, production rules R, and production rule probabilities Π. R consists of three types of rules:

$$\begin{aligned} S &\rightarrow A && \text{where } A \in N \\ A &\rightarrow B\ C && \text{where } B, C \in (N \cup P) \\ T &\rightarrow \omega && \text{where } T \in P \text{ and } \omega \in \Sigma \end{aligned}$$

Note that the left-hand side of the rules is distinguished between nonterminal and preterminal categories. In this paper, we refer to the symbol on left-hand side as the parent and the symbols on right-hand side as the children.

Neural Parameterization Neural parameterization is the process of using a neural network to parameterize the probabilities of production rules from symbol embeddings (Kim et al., 2019a). N-PCFGs parametrize root, nonterminal, and preterminal symbols with independent different neural networks. In more detail, the symbol embeddings used in parameterizations are not shared. For example, preterminal symbol embeddings in binary rule parameterizer and unary rule parameterizer are independent.

$$\pi_{S \rightarrow A} = \frac{\exp(\mathbf{u}_A^\top f_1(\mathbf{w}_S))}{\sum_{A' \in \mathcal{N}} \exp(\mathbf{u}_{A'}^\top f_1(\mathbf{w}_S))}, \quad (1)$$

$$\pi_{A \rightarrow BC} = \frac{\exp(\mathbf{u}_{BC}^\top \mathbf{w}_A)}{\sum_{B'C' \in \mathcal{M}} \exp(\mathbf{u}_{B'C'}^\top \mathbf{w}_A)}, \quad (2)$$

$$\pi_{T \rightarrow \omega} = \frac{\exp(\mathbf{u}_\omega^\top f_2(\mathbf{w}_T))}{\sum_{\omega' \in \Sigma} \exp(\mathbf{u}_{\omega'}^\top f_2(\mathbf{w}_T))} \quad (3)$$

Where, $\mathbf{w}_S, \mathbf{w}_A, \mathbf{w}_T$ represent root, nonterminal, preterminal symbol embeddings each, f_1, f_2 represent multi-layer perceptron composed by residual blocks, $\mathbf{u}_A^\top, \mathbf{u}_{BC}^\top, \mathbf{u}_\omega^\top$ represent weight matrix of each linear layer. The training objective of neural grammar induction is the maximization of sentence probabilities, and sentence probability is the sum of whole probabilities of derivable parse tree for given sentence. The parse tree probabilities are calculated by inside algorithm using probability distribution obtained by neural parameterization.

3 Probability Distribution Collapse

Implicit Bottleneck of Expressiveness Probabilistic grammars have been central in computational linguistics, offering tractable analyses of capacity utilization. In contrast, neural parameterizations lack such clarity due to the flexible and opaque mappings from symbol embeddings to probabilities. As a result, it is challenging to validate that the neural probabilistic grammars utilize its full representational capacity. Therefore, recent neural grammar induction methods frequently adopt overparameterized grammars to improve performance (Yang et al., 2021b, 2022; Liu et al., 2023), despite to the empirical availability of more compact grammars demonstrated in probabilistic learning frameworks (Stolcke and Omohundro, 1994). This discrepancy suggests that neural parameterization imposes implicit constraints on grammatical expressiveness, limiting the effective use of available capacity.

Probability Distribution Collapse We propose *Probability Distribution Collapse* (PDC) as the reason of the bottleneck. PDC refers to a phenomenon

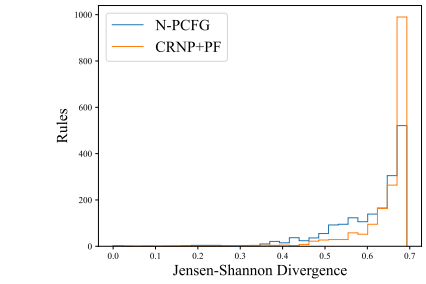
where embeddings for distinct symbols are mapped to similar probability distributions, resulting in a large portion of dominating rules being shared. We formalize PDC by using Jensen-Shannon Divergence (JSD) that measure the similarity between two probability distributions, where higher values indicate greater dissimilarity (See Appendix A.2). Therefore, we define PDC as:

Definition 1. Let G be a neural grammar. Let $z_{i,G}$ and $z_{j,G}$ be distinct symbol embeddings in G such that $z_{i,G} \neq z_{j,G}$. Let f be a function that maps each symbol embedding to a probability distribution, yielding $p_i = f(z_{i,G})$ and $p_j = f(z_{j,G})$. We say that *probability distribution collapse* occurs when the Jensen-Shannon Divergence between these distributions converges to zero, i.e., $\mathrm{JSD}(p_i \| p_j) \to 0$. where the JSD is defined as

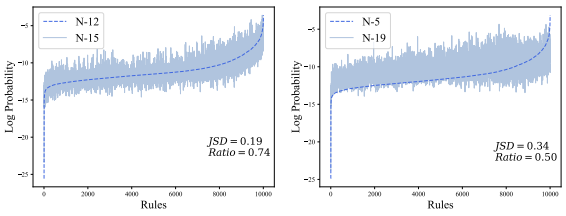
$$\mathrm{JSD}(P\|Q) = \frac{\mathrm{KL}(P\|\frac{P+Q}{2}) + \mathrm{KL}(Q\|\frac{P+Q}{2})}{2}$$

If a neural parameterization has an implicit bias that causes such PDC, a portion of symbols is inevitably wasted, and the grammar cannot represent the more diverse distributions available within its full capacity. This can degrade performance when structural diversity is required and limit even scalability in extending to more complex, real-world problems.

Empirical Evidence in N-PCFG Figure 1 illustrates evidence for PDC in N-PCFGs. In Figure 1(a), N-PCFGs reveal that while some rule distributions are distinct from each other, a large portion remains highly similar. In contrast, our proposed approach yields a distribution more concentrated around 0.7, indicating that most rule distributions are more distinct. Figure 1(b)–(d) show the sorted log probabilities for rules of two nonterminal symbols for each scale of JSD in N-PCFGs. To quantify the diversity of rule usage, we compute the ratio of overlapping rules within the top 90% of cumulative probability mass to the total number of rules contributing to that mass. The figure for low JSD (e.g., 0.2) shows a high overlap ratio of 0.71, indicating limited distributional variation. As JSD increases, this overlap ratio steadily declines, approaching zero near a JSD of 0.7, where rule probabilities differ substantially. Therefore, N-PCFGs with fewer distinct distributions suffer from PDC, which limits their ability to fully utilize model capacity, suggesting potential for further utilization gain.

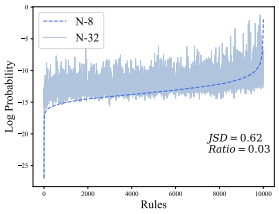


(a) Dissimilarity of rule probability distributions

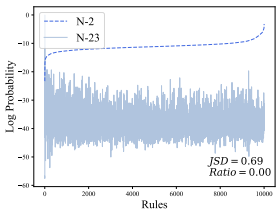


(b) JSD = 0.19 (similar)

(c) JSD = 0.34



(d) JSD = 0.62



(e) JSD = 0.69 (dissimilar)

Figure 1: Histogram that divided based on *Jensen-Shannon Divergence* in (a). The histogram shows how many pairs of rule probability distributions fall into each bin. Blue and orange represent N-PCFG and ours. (b)-(e) provide examples from each bin of the histogram on N-PCFGs, showing the overlap of their utilized rules. Each blue line represents a different symbol.

4 Collapse-Relaxing Neural Parameterization

In this section, we investigate the causes of PDC throughout all stages of neural parameterization. We discuss 1) symbol embeddings and their mapping to latent representations with limited expressiveness as a neural network in Section 4.1, 2) mapping latent representations to rule probability distributions entangled by embedding scale factors across symbols in Section 4.2, and 3) training dynamics causing representation collapse in Section 4.3. Then, we discuss the strategies to solve each issue, and propose an implementation on N-PCFG.

4.1 Symbol Embedding to Latent Representation

Less Expressive Embedding and Layers Existing models typically employ small embedding

sizes and shallow networks for neural parameterization. In particular, the parameterization of binary rules often relies on a single linear layer, which is a linear transformation of parent symbol embeddings to logits for rule probabilities, notated as: $T : \mathbf{R}^{emb} \rightarrow \mathbf{R}^{rules}$. When $emb < rules$ where $rules = \|N \cup P\|^2$, the mapping is surjective onto a subspace of the rule probability distribution space. As a result, certain rule probability distributions become unrepresentable given the limited dimensionality of the parent embeddings. When an optimal grammar requires those unrepresentable distributions, the trained grammar confined to the subspace becomes a projection of the optimum, exposed to losing distinction of the projected distributions, which is PDC effect.

Solution The solution to this problem is relatively straightforward: increasing the dimensionality of symbol embeddings and the number of network layers. While this can be achieved through simple hyperparameter tuning, the following two issues may hinder performance gains, which requires a method that addresses them simultaneously.

4.2 Latent Representation to Probability Distribution

Entangled Children Scale Across Rules The next step of neural parameterization also causes PDC due to the shared latent representations of children across different rules. This step involves computing the inner product between the parent and child latent representations, which is then passed to the penultimate layer to generate logits for the softmax function representing the rule probability distribution. The inner product is given by $\mathbf{u}_{BC}^{\top}\mathbf{w}_A$, where $\mathbf{w}_A$ and $\mathbf{u}_{BC}$ denote the latent representations of the parent and children, respectively, as shown in equation (2). The resulting rule probability can be rewritten in terms of the inner product as:

$$\pi_{A \rightarrow BC} = \text{softmax}(\|\mathbf{w_A}\|\|\mathbf{u_{BC}}\| \cos\theta_{\mathbf{w_A},\mathbf{u_{BC}}}) \tag{4}$$

A notable point is that the scale of the children representation is shared across many parent rules. For instance, the penultimate representations of rules $A \rightarrow BC$ and $A' \rightarrow BC$, involving different parent symbols, both include the shared children scale $\|\mathbf{u}_{BC}\|$. This shared scale entangles the distributions of different rules, so a substantial increase in the children scale during training for one rule induces a corresponding increase for other rules involving the same children. More critically, this scale entanglement manifests across all parent symbols that share the same children, occurring consistently for each children in the grammar. As a result, a greater reliance on the children scale leads to convergence toward more similar rule distributions, thus causing PDC.

Solution To mitigate PDC caused by entangled children scales, a simple yet effective solution is to normalize the scale. By fixing the magnitude as a constant, the rule probability distribution becomes invariant to scale, preventing unintended influence from unrelated rules while preserving the learnability of child representations. In contrast, normalizing parent representations is generally discouraged, as each parent defines a distinct probability distribution. Normalization may unnecessarily reduce variance, limiting expressiveness.

4.3 Training Dynamics

Bias to Specific Distributions During training with neural parameterization, two phenomena frequently occur: 1) gradient explosion that training is disrupted by excessively large gradients and 2) dying ReLU that activations passing through the ReLU function are mapped to zero. These effects push the probability distribution toward specific forms, causing PDC that limit the expressiveness for more diverse distributions. First, gradient explosion (GE) rapidly increases the scale of parent representations, resulting in an extremely sharp, one-hot-like distribution. This shift from a more diverse distribution to a near-deterministic one is a form of PDC. GE also easily increases the scale of children representations, which amplifies the previously discussed PDC, further concentrating probability mass on a few specific rules. To mitigate this, N-PCFGs often apply residual connections and gradient clipping. However, residual blocks still suffer from GE, and gradient clipping only partially alleviates the problem while potentially hindering the learning of other weights (Zhang et al., 2019). Thus, these methods offer limited effectiveness in resolving the core issue. Second, dying ReLU causes most dimensions of the parent representation to become zero. In this case, as shown in equation (4), the norm $|\mathbf{w}_A|$ approaches zero, causing the probabilities for all children representations to converge to equal values. Consequently, the grammar converges to a uniform probability distribution.

Solution There are various strategies to mitigate gradient exploding and the dying ReLU problem. In this work, we adopt two simple yet effective methods to demonstrate that resolving these issues alleviates PDC: 1) We replace the ReLU activation with GELU (Hendrycks and Gimpel, 2016), which offers two advantages. First, GELU allows activation in the negative domain, helping prevent the dying ReLU issue. Second, in the mean-field perspective, GELU exhibits a wider edge-of-chaos regime than ReLU, which helps stabilize gradients and mitigate explosion. 2) We remove residual connections for two reasons. First, residual networks without normalization still suffer from gradient explosion (Zhang et al., 2019). Second, even with normalization, residual blocks often suffer from scale imbalance between the main and residual paths (Hayou et al., 2020).

4.4 Implementation of Rule Parameterization

To use the proposed strategies, we present our redesign of the neural parameterization in N-PCFGs. As described, the neural parameterization consists of three independent neural networks, each modeling one of subsets of the rule probability distributions: root rules ($S \rightarrow A$), binary rules ($A \rightarrow BC$), and unary rules ($T \rightarrow \omega$). We apply distinguished strategies to each network according to the characteristics of the corresponding distribution set.

Root Rules The root rule probability distribution consists of a single distribution with a pseudo-root symbol as the parent. In other words, it does not conflict with any other probability distribution. Therefore, it does not require high expressive power, and we use a single fully connected layer to reduce computational cost.

$$\pi_{S \rightarrow A} = \frac{\exp(\mathbf{u}_A^\top \mathbf{w}_S)}{\sum_{A' \in \mathcal{N}} \exp(\mathbf{u}_{A'}^\top \mathbf{w}_S)} \tag{5}$$

Binary and Unary Rules Parameterizing binary and unary rule probability distributions is more complex than the root rule ($S \rightarrow NT$). Binary rules require $|N|$ distinct probability distributions over $|N \cup U|^2$ possible child combinations, and unary rules require $|P|$ distinct distributions over $|\Sigma|$ terminals. To learn distinct distributions for each parent symbol, the neural parameterization must have sufficient expressive power. Accordingly, we compute the binary and unary rule distributions using the following equations:

$$\pi_{A \rightarrow BC} = \frac{\exp(\|\mathbf{v}_A\| \cos \theta_{\mathbf{u}_{BC}, v_A})}{\sum_{B'C' \in \mathcal{M}} \exp(\|\mathbf{v}_A\| \cos \theta_{\mathbf{u}_{B'C'}, v_A})}, \tag{6}$$

$$\pi_{T \rightarrow \omega} = \frac{\exp(\|\mathbf{v}_T\| \cos \theta_{\mathbf{u}_\omega, \mathbf{v}_T})}{\sum_{\omega' \in \Sigma} \exp(\|\mathbf{v}_T\| \cos \theta_{\mathbf{u}_{\omega'}, \mathbf{v}_T})} \tag{7}$$

where $\mathbf{v}_A$, $\mathbf{v}_T$ indicates the representations for parent symbols calculated with the equation:

$$\mathbf{v}_A = \text{GELU}(\text{RMSNorm}(\mathbf{w}^\top \mathbf{w}_A)) \tag{8}$$

5 Experiments

5.1 Settings

Datasets We evaluate the performance of models for constituency parsing task using the Penn TreeBank (PTB) (Marcus et al., 1994) dataset for English, the Penn Chinese TreeBank (CTB) (Xue et al., 2005) and the SPMRL dataset (Seddah et al., 2014) for the other eight languages. We use the same preprocessing as Yang et al. (2022) and Park and Kim (2024). We use the same train / development / test split as Yang et al. (2022) for PTB, CTB and SPMRL dataset.

Hyperparameters We train for 30 epochs with a batch size of 4 and use the same hyperparameters as N-PCFGs (Kim et al., 2019a), except that we do not apply curriculum learning and gradient clipping. For curriculum learning, TN-PCFGs (Yang et al., 2021b) reported no performance benefit, and our own preliminary tests similarly showed no impact. It is also not used in TN-PCFGs, Rank PCFGs, or SimplePCFGs. As for gradient clipping, we do not apply it because it hinders the learning of weights, as previously discussed in Section 4.3. Additionally, we use RMS Normalization (Zhang and Sennrich, 2019) after the activation function in CRNP to stabilize the forward path. We run only four runs for each model. We follow the ratio of 1:2 between nonterminals and preterminals. The details for the dataset and training are in Appendix A.1 We primarily evaluate parsing performance of grammars with MBR decoding (Yang et al., 2021b).

Parse-Focusing N-PCFGs already suffer from known potential issues, namely Structural Optimization Ambiguity (SOA) and Structural Simplicity Bias (SSB) (Park and Kim, 2024), which may overshadow the effects of mitigating PDC. To clearly analyze these effects, we apply Parse-Focusing (PF) (Park and Kim, 2024) to N-PCFGs

Type	Model	$\|N\|$	S-F1	
			Mean	Max
Original	N-PCFG*	30	$52.3_{\pm 2.3}$	55.8
	C-PCFG*	30	$56.3_{\pm 2.1}$	60.1
	TN-PCFG*	30	$51.4_{\pm 4.0}$	55.6
	TN-PCFG*	250	$57.7_{\pm 4.2}$	61.4
	Rank PCFG†	30	$51.2_{\pm 3.1}$	-
	Rank PCFG†	4500	64.1	-
	SN-PCFG‡	128	$51.1_{\pm 4.1}$	-
	SN-PCFG‡	4096	$65.1_{\pm 2.1}$	-
	CRNP (Ours)	30	$46.2_{\pm 5.2}$	52.7
Upper-bound	N-PCFG+PF$_g$	30	$72.6_{\pm 0.7}$	73.4
	CRNP+PF$_g$	30	$\mathbf{73.7}_{\pm 0.3}$	74.2
PF (fixed-NT size)	N-PCFG+PF	30	$68.3_{\pm 0.2}$	68.7
	Rank PCFG+PF	30	$67.4_{\pm 0.9}$	68.4
	CRNP+PF	30	$\mathbf{69.4}_{\pm 0.3}$	69.7
PF (no-limit)	Rank PCFG+PF	4500	$69.6_{\pm 0.6}$	70.3
	CRNP+PF	**90**	$\mathbf{70.2}_{\pm 0.5}$	70.9

Table 1: Performance measured by unlabeled sentence-F1 (S-F1) in English constituent parsing on PTB. $|N|$ represents the number of nonterminals. *, †, and ‡ indicate reported results from Yang et al. (2021b), Yang et al. (2022), and Liu et al. (2023), respectively. Parse-focusing PF$_g$ uses gold parse trees, while PF uses parse trees induced from pretrained models in an unsupervised setting without extra data.

and CRNP. The Parse-Focusing method was originally proposed to utilize the parse trees induced by pretrained unsupervised models (TN-PCFG (Yang et al., 2021b), NBL-PCFG (Yang et al., 2021a), and Structformer (Shen et al., 2021)) in the unsupervised setting, and we follow this setup. Additionally, we use the gold parse trees to reveal the experimental upper bound achievable in UNGI. In this setting, we assume that the gold parse trees provide the ideal focusing bias and thus yield the best performance. This corresponds to the supervised setting.

5.2 Performance Results

Performance in English Table 1 presents the unlabeled sentence-level F1 (S-F1) scores on the PTB dataset for various baselines and our model with the proposed parameterization (CRNP). CRNP without PF shows a lower mean and higher variance in S-F1 scores compared to conventional neural parameterization, due to the presence of SOA and SSB. When PF is applied using parse trees induced from pretrained models on the same dataset, CRNP+PF consistently outperforms N-PCFG+PF. CRNP+PF with 90 nonterminals demonstrates significant performance improvements across most baselines, achieving slightly higher performance than Rank PCFG+PF with 4500 nonterminals, a grammar that is 50 times larger. Notably, when both models use 30 nonterminals, CRNP+PF shows a substantial performance gain over Rank PCFG+PF. Furthermore, when gold parse trees are used for PF, CRNP+PF with just 30 nonterminals surpasses N-PCFG+PF, demonstrating a stronger upper bound.

Performance in Multilingual Table 2 presents the S-F1 scores for parsing performance on PTB, CTB, and SPMRL datasets. Overall, the proposed method, CRNP+PF, achieves substantial improvements across most languages, including Basque, Chinese, English, French, Hebrew, and Korean, and ranks second for German, Hungarian, and Swedish, resulting in the best average rank (1.7) among all models. These results highlight the effectiveness of CRNP across diverse language datasets. Under a fixed grammar size of 30 nonterminals, CRNP+PF consistently outperforms other models. Even under the unrestricted setting, it achieves the highest scores in most cases, while still using a highly compact grammar with only 90 nonterminals.

Grammar Compactness Table 3 compares performance across grammar sizes for N-PCFG, Rank PCFG, and CRNP. The results show that CRNP exhibits a steeper improvement and more accurate performance in S-F1 scores as grammar size increases, compared to the baseline models. This demonstrates that CRNP effectively leverages larger grammars by mitigating the bottleneck limitations of neural parameterization.

Ablation Studies For the ablation study, we examine the individual contributions of GELU activation and children embedding scale normalization, which are two core components of CRNP, to the upper bound without interference from unsupervised learning environments. Table 4 presents S-F1 scores under different configurations. The removal of either component consistently degrades performance, indicating that both are essential to the effectiveness of the proposed method. The ablation study on varying embedding dimensions, another core component, requires a more in-depth analysis. Therefore, we discuss it separately in Section 5.3.

Model	$\|N\|$	Basque	Chinese	English	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	Mean rank
N-PCFG*	30	$35.1_{\pm 2.0}$	$26.3_{\pm 2.5}$	$52.3_{\pm 2.3}$	$45.0_{\pm 2.0}$	$42.3_{\pm 1.6}$	$45.7_{\pm 2.2}$	$43.5_{\pm 1.2}$	$28.4_{\pm 6.5}$	$43.2_{\pm 0.8}$	$17.0_{\pm 9.9}$	6.2
C-PCFG*	30	$36.0_{\pm 1.2}$	$38.7_{\pm 6.6}$	$56.3_{\pm 2.1}$	$45.0_{\pm 1.1}$	$43.5_{\pm 1.2}$	$45.2_{\pm 0.5}$	**$44.9_{\pm 1.5}$**	$30.5_{\pm 4.2}$	$43.8_{\pm 1.3}$	$33.0_{\pm 15.4}$	5.0
CRNP+PF (Ours)	30	$46.1_{\pm 0.2}$	$45.8_{\pm 1.7}$	$69.4_{\pm 0.3}$	$50.4_{\pm 0.1}$	$47.9_{\pm 0.2}$	**$49.5_{\pm 0.3}$**	$43.9_{\pm 0.1}$	$41.9_{\pm 0.8}$	$45.8_{\pm 0.9}$	$34.0_{\pm 0.9}$	2.9
TN-PCFG*	250	$36.0_{\pm 3.0}$	$39.2_{\pm 5.0}$	$57.7_{\pm 4.2}$	$39.1_{\pm 4.1}$	$47.1_{\pm 1.7}$	$39.2_{\pm 10.7}$	$43.1_{\pm 1.1}$	$35.4_{\pm 2.8}$	**$48.6_{\pm 3.1}$**	**$40.0_{\pm 4.8}$**	4.5
Rank PCFG†	4500	$38.4_{\pm 7.3}$	$31.0_{\pm 8.4}$	$59.6_{\pm 7.7}$	$43.9_{\pm 3.1}$	$48.0_{\pm 1.4}$	$46.2_{\pm 4.1}$	$42.2_{\pm 0.7}$	$31.5_{\pm 4.0}$	$41.6_{\pm 4.3}$	**$40.0_{\pm 0.6}$**	4.7
Rank PCFG+PF†	4500	$45.9_{\pm 0.3}$	$46.1_{\pm 0.9}$	$69.7_{\pm 0.9}$	$50.5_{\pm 0.5}$	**$49.1_{\pm 0.3}$**	**$49.5_{\pm 0.2}$**	$43.7_{\pm 0.2}$	$42.1_{\pm 0.3}$	$47.9_{\pm 0.3}$	$33.4_{\pm 1.2}$	2.4
CRNP+PF (Ours)	90	**$46.3_{\pm 0.6}$**	**$46.6_{\pm 1.0}$**	**$70.2_{\pm 0.5}$**	**$51.4_{\pm 0.3}$**	$48.7_{\pm 0.2}$	**$49.5_{\pm 0.5}$**	$44.1_{\pm 0.2}$	**$42.3_{\pm 0.3}$**	$45.7_{\pm 0.4}$	$34.4_{\pm 0.6}$	**1.7**

Table 2: Performance (S-F1 score and mean rank) in multilingual parsing on PTB, CTB, and SPMRL datasets. (*,† :reported results in Yang et al. (2021b), Park and Kim (2024), respectively, $|N|$: number of nonterminals). The bold represent the best performance and the underline represent the second performance. Mean rank represents the average rank of each model across all languages. Blue text indicates the best result in each division, either with NT fixed to 30 or with NT unrestricted.

$\|N\|$	S-F1		
	Rank PCFG	Rank PCFG+PF	CRNP+PF
1	$39.7_{\pm 0.0}$	$44.3_{\pm 0.1}$	$45.0_{\pm 0.1}$
5	$41.6_{\pm 9.6}$	$60.4_{\pm 1.9}$	$59.4_{\pm 3.0}$
15	$43.2_{\pm 0.8}$	$65.5_{\pm 0.7}$	$67.5_{\pm 0.7}$
30	$51.2_{\pm 3.1}$	$67.4_{\pm 0.9}$	$69.4_{\pm 0.3}$
60	-	-	$70.0_{\pm 0.2}$
90	-	-	$70.2_{\pm 0.5}$
250	$54.3_{\pm 3.9}$	$69.7_{\pm 0.9}$	-
4500	$57.4_{\pm 6.0}$	$69.6_{\pm 0.7}$	-

Table 3: Performance (S-F1) by grammar size to show grammar compactness.

Model	S-F1
CRNP+PF$_{-\text{GELU}-\text{NormalizedEmb}}$	$72.4_{\pm 0.4}$
CRNP+PF$_{-\text{GELU}+\text{ReLU}}$	$72.9_{\pm 0.2}$
CRNP+PF$_{-\text{NormalizedEmb}}$	$73.5_{\pm 0.2}$
CRNP+PF	**73.7**$_{\pm 0.3}$

Table 4: Ablation studies on upper bound performance ($_{-\text{GELU}+\text{ReLU}}$: using ReLU instead of GELU, $_{\text{-NormaliedEmb}}$: no normalization)

5.3 In-Depth Analysis

Rule Type	Model	GPJ with varying $\|N\|$		
		30	60	90
Binary	N-PCFG	0.661	0.661	0.657
	N-PCFG+PF	0.647	0.645	0.634
	CRNP+PF	0.647	0.657	0.662
Unary	N-PCFG	0.561	0.591	0.590
	TN-PCFG	0.481	-	-
	Rank PCFG	0.587	-	-
	N-PCFG+PF	0.615	0.634	0.636
	CRNP+PF	0.633	0.646	0.649

Table 5: Probability distribution collapse (measured by geometric mean of pairwise JSD (GPJ)) by the grammar size (nonterminal and preterminal size).

Mitigation of Probability Distribution Collapse Beyond performance and compactness improvement, we validate that PDC is effectively reduced by our approach. To evaluate the impact, we utilize the Jensen-Shannon Divergence (JSD), a symmetric and finite metric that measures the similarity between probability distributions. The range of JSD is $(0, \ln 2) \approx (0.00, 0.69)$. A high JSD indicates a greater difference between two probability distributions, while a value of zero indicates identical distributions. To verify the overall tendency of PDC in grammars, we employ the geometric mean of pairwise JSD (GPJ) for rule probability distributions. The GPJ reflects the diversity of the distributions, thereby implying the degree of capacity utilization. We address the detailed definition of GPJ in Appendix A.2. In Table 5, CRNP exhibits high GPJ, indicating that the probability distributions in the grammar are distinct from one another. In contrast, N-PCFGs show low GPJ for unary probability distributions while maintaining a comparable GPJ for binary distributions. Similarly, TN-PCFGs and Rank PCFGs exhibit low GPJ for unary rules, which means they also suffer from PDC. These results demonstrate the effectiveness of our approach in mitigating PDC and improving capacity utilization.

Embedding Size	S-F1		
	N-PCFG	Rank PCFG+PF	CRNP+PF
64	54.3	65.9	68.6
128	54.8	67.2	68.8
256	52.8	68.4	69.4
512	51.8	66.5	69.2
1024	50.6	66.5	69.2
2048	45.4	67.3	69.3

Table 6: Impact (S-F1) of the size of input embeddings and hidden states.

Embedding Size We investigate the impact of model expressiveness by evaluating performance across varying symbol embedding and hidden state sizes, as shown in Table 6. For N-PCFGs, S-F1 scores increase with embedding size up to 128 but decline beyond that, likely due to training dynamics that exacerbate PDC in larger networks. In contrast, CRNP performance continues to improve with larger embeddings until saturation, which occurs once the expressive power matches the complexity of the target grammar. Beyond this point, further improvement would require increasing grammar size.

Layers	1	2	3	4
S-F1	$68.3_{\pm 0.6}$	$69.2_{\pm 0.4}$	$69.4_{\pm 0.3}$	$69.2_{\pm 0.3}$

Table 7: Impact of (S-F1) the number of layers by the depth of neural parameterization layers.

The Number of Layers Another important factor in controlling network capacity is the number of layers. Table 7 reports performance changes as network depth increases in the neural parameterization. Unlike N-PCFGs, our approach enhances model capacity by increasing the number of layers, thereby improving nonlinearity. Consistent with the previous results from varying embedding size, performance saturates once sufficient expressiveness is achieved at two layers, with no further gains observed from additional depth.

Model	cosine similarity	
	binary	unary
N-PCFG	0.85	0.71
N-PCFG+PF	0.91	0.78
CRNP+PF	0.75	0.30

Table 8: Average of cosine similarity among children representations used as inputs of the penultimate layer

Children Scale To examine the effect of normalizing the scale of children representations, we evaluate their cosine similarity, as shown in Table 8. Normalization in CRNP significantly reduces similarity for both binary and unary rules, indicating more diverse distributions. This outcome is expected, as removing scale from the learnable parameters places greater emphasis on cosine similarity to capture distributional differences. These results suggest normalizing the children scale enhances representational diversity and encourages more varied rule probability distributions.

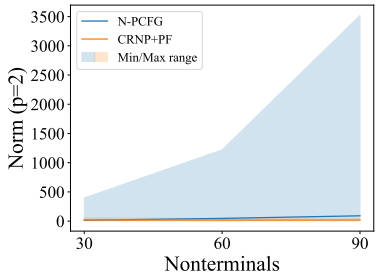


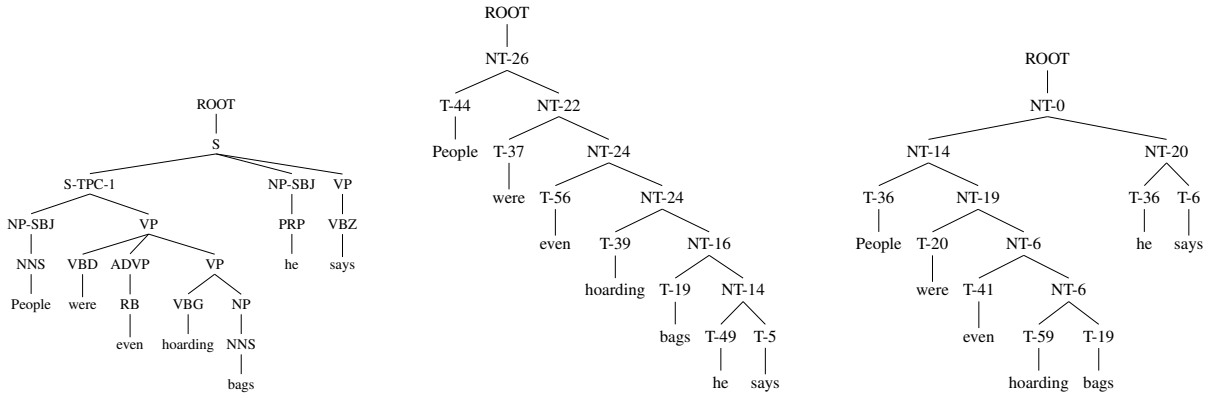
Figure 2: Gradient explosion evaluation reflected by embedding scales by nonterminals.

Gradient Explosion We evaluate the scale of children representations to demonstrate the presence of gradient explosion and the effectiveness of our proposed method. Figure 2 shows the scale values before and after applying our approach. When the number of non-terminals is 30, both models exhibit comparable scales. However, as the number of non-terminals increases, the N-PCFG exhibits an exponential increase in scale values by the large maximum, indicating the onset of severe gradient explosion. In contrast, our method maintains consistently low scales and a narrow range, even with larger grammars. These results confirm that our approach effectively stabilizes the representation scale and mitigates critical PDC caused by sharing the scale factor in children representations.

Rule Type	Model	$\|N\|$	Entropy	Ratio of zeros
Binary	N-PCFG	30	5.168	0.990
		60	4.996	0.994
		90	4.506	0.993
	CRNP+PF	30	0.808	0.000
		60	0.873	0.000
		90	0.720	0.000
Unary	N-PCFG	30	2.596	0.000
		60	5.148	0.000
		90	4.718	0.000
	CRNP+PF	30	0.743	0.000
		60	0.886	0.000
		90	0.794	0.000

Table 9: Evaluation of the dying ReLU phenomenon. Entropy is computed from each nonterminal's rule probability distribution, and the ratio from ReLU-generated representations. Both are averaged over all rules. Higher entropy indicates greater uniformity; the zero ratio reflects dying elements in the representations.

Dying ReLU In Table 9, N-PCFG consistently exhibits higher entropy, indicating that its rule probability distributions are, on average, more uniform. However, high entropy does not necessarily imply the absence of inactive (dying) components in the representations. To assess this, we also measure



(a) Gold parse tree. (b) Parse tree induced by N-PCFG+PF. (c) Parse tree induced by CRNP+PF.

Figure 3: Comparison between the gold parse tree and the parse trees induced by the models (N-PCFG+PF and CRNP+PF) on the example sentence *"People were even hoarding bags he says"*.

the ratio of zero values. While N-PCFG shows a near-one ratio in binary rules, CRNP+PF maintains the zero rate. This suggests that N-PCFG suffers from the dying ReLU problem, whereas the proposed method effectively mitigates it. There is no dying ReLU phenomenon for binary rules, because N-PCFGs do not use an activation function to parameterize binary rules.

The Difference in Tree Structure For qualitative analysis, we provide example parse trees in Figure 3 to examine the influence of CRNP on parse structures. N-PCFG+PF generates a right-binarized parse tree that differs from the gold by relying heavily on a few nonterminal symbols such as NT-22 and NT-24, whereas CRNP+PF reproduces the same structure as the gold by utilizing a wider variety of symbols. From the perspective of symbol roles, CRNP+PF utilizes a broader range of symbols for S and VP than N-PCFG+PF, indicating a more diverse representation of intermediate sentence structures. We further present additional examples in Appendix A.3.

6 Related Works

Overparameterization of UNGI Yang et al. (2021b) proposed the neural parameterization based on tensor decomposition, and demonstrated that overparameterization leads to better grammars with many symbols. Moreover, Yang et al. (2022) improved it based on factor graph grammars (FGGs). Recently, Liu et al. (2023) introduced the SimplePCFG formalism. These show the performance improvements based on a large grammar size. Conversely, we highlight the inefficiency in learning UNGIs caused by an implicit bottleneck, and demonstrate resolving this issue leads to compact and accurate grammars.

Collapse Problem In self-supervised learning (SSL), the collapse problem refers to representations with different semantics being mapped to very close positions. Hua et al. (2021) verified complete collapse that representations converge into a trivial solution, and identified dimensional collapse that representations concentrate into sparse dimensions. In another line of work, Papyan et al. (2020) proposed neural collapse in which representations within the same class converge to their class means in classification tasks. These studies focus on representations themselves. Whereas we focus on the collapse of probability distributions generated from representations and their negative effect, which directly limits capacity utilization of grammars.

7 Conclusion

In this paper, we address the issue of *probability distribution collapse*, which creates a bottleneck in neural parameterization and limits the effective utilization of model capacity in unsupervised neural grammar induction. We analyze the underlying causes of this collapse, tracing training dynamics from symbol embeddings to the probability distributions, and propose a *collapse-relaxing neural parameterization* to mitigate it. Our method improves the upper bound and overall accuracy of constituency parsing across English and multiple languages, while achieving significantly more compact grammars. This work underscores the often overlooked limitations of neural parameterization and reveals that large-capacity models are not strictly necessary for effective grammar learning.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2022R1A2C2012054, Development of AI for Canonicalized Expression of Trained Hypotheses by Resolving Ambiguity in Various Relation Levels of Representation Learning).

Limitations

Unsupervised neural grammar induction using N-PCFGs is still constrained by expensive computational cost. Therefore, extending our idea to a decomposed tensor-based method is important to scale up. However, applying our method to models such as TN-PCFGs or Rank PCFGs is constrained by tensor decomposition. While PDC may affect these models as well, further analysis is required to assess its impact and applicability. In addition, we observed that CRNP+PF with 90 nonterminals performs slightly less effectively in a few languages (e.g., German and Polish). However, language-specific differences in resolving PDC are out of the main scope of this work. Therefore, such issues should be further investigated in future work.

References

Rares-Darius Buhai, Yoni Halpern, Yoon Kim, Andrej Risteski, and David A. Sontag. 2019. Empirical study of the benefits of overparameterization in learning latent variable models. In *International Conference on Machine Learning*.

Andrew Drozdov, Subendhu Rongali, Yi-Pei Chen, Tim O'Gorman, Mohit Iyyer, and Andrew McCallum. 2020. Unsupervised parsing with S-DIORA: Single tree encoding for deep inside-outside recursive autoencoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4832–4845, Online. Association for Computational Linguistics.

Andrew Drozdov, Pat Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. 2019. Unsupervised latent tree induction with deep inside-outside recursive autoencoders.

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.

Soufiane Hayou, Eugenio Clerico, Bo He, George Deligiannidis, A. Doucet, and J. Rousseau. 2020. Stable resnet. In *International Conference on Artificial Intelligence and Statistics*.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv: Learning*.

Yining Hong, Qing Li, Song-Chun Zhu, and Siyuan Huang. 2021. Vlgrammar: Grounded grammar induction of vision and language. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1645–1654.

Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. 2021. On feature decorrelation in self-supervised learning. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9578–9588.

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

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019b. Unsupervised recurrent neural network grammars. 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 1105–1117, Minneapolis, Minnesota. Association for Computational Linguistics.

Wei Liu, Songlin Yang, Yoon Kim, and Kewei Tu. 2023. Simple Hardware-Efficient PCFGs with Independent Left and Right Productions. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1662–1669, Singapore. Association for Computational Linguistics.

Mitch 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*.

Vardan Papyan, Xuemei Han, and David L. Donoho. 2020. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences of the United States of America*, 117:24652 – 24663.

Jinwook Park and Kangil Kim. 2024. Structural optimization ambiguity and simplicity bias in unsupervised neural grammar induction. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15124–15139, Bangkok, Thailand. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca

Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440.

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.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks.

Yikang Shen, Yi Tay, Che Zheng, Dara Bahri, Donald Metzler, and Aaron Courville. 2021. Structformer: Joint unsupervised induction of dependency and constituency structure from masked language modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7196–7209.

Andreas Stolcke and Stephen Omohundro. 1994. *Inducing probabilistic grammars by Bayesian model merging*, page 106–118. Springer Berlin Heidelberg.

Yau-Shian Wang, Hung-Yi Lee, and Yun-Nung Chen. 2019. Tree transformer: Integrating tree structures into self-attention.

Christopher K. I. Williams. 2023. Structured generative models for scene understanding. *Int. J. Comput. Vis.*, 133:2845–2867.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11:207–238.

Songlin Yang, Wei Liu, and Kewei Tu. 2022. Dynamic programming in rank space: Scaling structured inference with low-rank HMMs and PCFGs. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4797–4809, Seattle, United States. Association for Computational Linguistics.

Songlin Yang, Yanpeng Zhao, and Kewei Tu. 2021a. Neural bi-lexicalized PCFG induction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2688–2699, Online. Association for Computational Linguistics.

Songlin Yang, Yanpeng Zhao, and Kewei Tu. 2021b. PCFGs can do better: Inducing probabilistic context-free grammars with many symbols. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1487–1498, Online. Association for Computational Linguistics.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *ArXiv*, abs/1910.07467.

Hongyi Zhang, Yann Dauphin, and Tengyu Ma. 2019. Fixup initialization: Residual learning without normalization. In *International Conference on Learning Representations*.

Yanpeng Zhao and Ivan Titov. 2020. Visually grounded compound pcfgs. In *Conference on Empirical Methods in Natural Language Processing*.

A Appendices

A.1 Experiment Details

Dataset Detail We adopt the standard setup and preprocessing for PTB,[1] using sections 02–21 for training, section 22 for validation, and section 23 for testing, with punctuation and trivial constituents removed. The vocabulary consists of the 10,000 most frequent words, while all other words are replaced with <unk>. For data processing, we follow the pipeline employed by prior models that utilize base models and parsers, as described in (Kim et al., 2019a; Yang et al., 2022; Shen et al., 2021; Yang et al., 2021a; Drozdov et al., 2019). We also utilize the CTB[2] and the Basque, French, German, Hebrew, Hungarian, Korean, Polish, and Swedish datasets from SPMRL,[3] following the standard setup used in prior work.

Implementation Detail To implement our methodology on top of the base model FGG-TNPCFGs, we utilized PyTorch version 2.2 (Paszke et al., 2019). For smooth processing and analysis of tree structures, we employed NLTK[4].

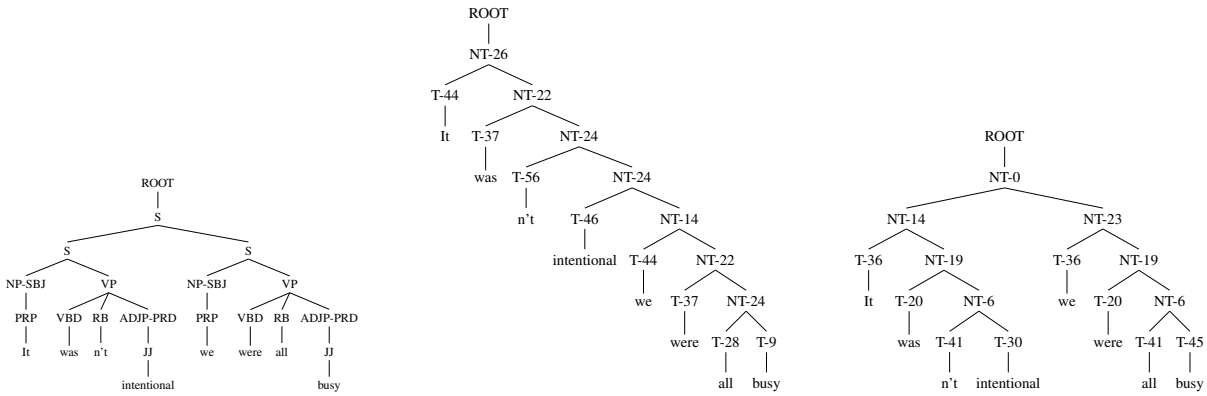
Training Detail The hyperparameter settings follow those reported in Yang et al. (2022). The ratio of nonterminal to preterminal symbols is primarily fixed at 1:2. To analyze the effect of the number of symbols, we conduct experiments varying the number of nonterminal symbols from 1 to

[1]The license of PTB is LDC User Agreement for Non-Members. https://catalog.ldc.upenn.edu/LDC99T42

[2]The license of CTB is LDC User Agreement for Non-Members. https://catalog.ldc.upenn.edu/LDC2005T01

[3]The license of SPMRL is Creative Commons Attribution 4.0 International License. https://www.spmrl.org/spmrl2013-sharedtask.html

[4]https://www.nltk.org/



(a) Gold parse tree. (b) Parse tree induced by N-PCFG+PF. (c) Parse tree induced by CRNP+PF.

Figure 4: Comparison between the gold parse tree and the parse trees induced by the models (N-PCFG+PF and CRNP+PF) on the example sentence *"It wasn't intentional, we were all busy"*.

4500. Training proceeds for up to 10 epochs, with early stopping based on validation likelihood. Optimization is performed using the Adam optimizer with a learning rate of 2×10^{-3}, $\beta_1 = 0.75$, and $\beta_2 = 0.999$. The model is configured with a rank size of 1000, a symbol embedding size of 256, and a word embedding size of 200. These hyperparameters are held constant across all languages without task-specific tuning. All experiments are run on an NVIDIA RTX 2080Ti. To measure variance, we repeat each experiment 32 times for both the original FGG-TNPCFGs and our method, as reported in Table 1, while all other experiments are repeated four times. Each run takes approximately 3 hours.

A.2 Measures for probability distribution collapse

Local Perplexity represents the number of valid rules in single probability distribution. In other words, local perplexity (PPL) represent sparsity of single probability distribution. Therefore, probability distribution that have low local PPL have few utilized rules in distribution and is sparse.

$$\text{PPL}_{\text{local}} = \frac{\sum_{p \in P} \exp H(p)}{|P|}$$

Global PPL represent the number of valid categories of the mean distribution of whole probability distributions in grammar, which represent how many various rules are used without duplication for parsing. This is related with the expressive power of grammar. In other words, higher performance, higher variety of the rules that utilized in grammar.

$$q(C) = \frac{\sum_{P' \in S} p(P' \to C)}{|S|}$$

$$\text{PPL}_{\text{global}} = \exp H(q)$$

Geometric Mean of Pairwise Jensen-Shannon Divergence shows how different the probability distributions in grammar with each other. In this paper, we use JSD with base-e logarithm for computational convenience. JSD is represented by following equation:

$$\text{JSD}(P \| Q) = \frac{1}{2}\text{KL}(P \| M) + \frac{1}{2}\text{KL}(Q \| M)$$

$$\text{where } M = \frac{1}{2}(P + Q)$$

We use JSD to compare similarity for several distributions, then we get the values for $\frac{N(N-1)}{2}$ pairs of distributions, where N represent the number of distributions. We use geometric mean to average on these values of pairs. The reason why we use geometric mean is arithmetic mean is not sensitive on outlier, which make that the duplication of pairs is not appeared in values in arithmetic mean. Therefore, the GPJ is defined as:

$$\text{GPJ} = \sqrt[|\text{pairs}|]{\prod_{P,Q \in S} JSD(P \| Q)}.$$

A.3 Additional Examples for parse trees

We provide additional examples of parse trees to compare the quality by methods, N-PCFG+PF and CRNP+PF in Figure 4.