Unsupervised Discontinuous Constituency Parsing with Mildly Context-Sensitive Grammars

Songlin Yang
ShanghaiTech University
yangsl66@mit.edu

Roger P. Levy
MIT BCS
rplevy@mit.edu

Yoon Kim
MIT CSAIL
yoonkim@mit.edu

Abstract

We study grammar induction with mildly context-sensitive grammars for unsupervised discontinuous parsing. Using the probabilistic linear context-free rewriting system (LCFRS) formalism, our approach fixes the rule structure in advance and focuses on parameter learning with maximum likelihood. To reduce the computational complexity of both parsing and parameter estimation, we restrict the grammar formalism to binary LCFRS with fan-out two and further discard rules that require $\mathcal{O}(\ell^6)$ time to parse, reducing inference to $\mathcal{O}(\ell^5)$. We find that using a large number of nonterminals is beneficial and thus make use of tensor decomposition-based rank-space dynamic programming with an embedding-based parameterization of rule probabilities to scale up the number of nonterminals. Experiments on German and Dutch show that our approach is able to induce linguistically meaningful trees with continuous and discontinuous structures.

1 Introduction

Unsupervised parsing aims to induce hierarchical linguistic structures given only the strings in a language. A classic approach to unsupervised parsing is through probabilistic grammar induction (Lari and Young, 1990), which learns a probabilistic grammar (i.e., a set of rewrite rules and their probabilities) from raw text. Recent work has shown that neural parameterizations of probabilistic context-free grammars (PCFG), wherein the grammar's rule probabilities are given by a neural network over shared symbol embeddings, can achieve promising results on unsupervised constituency parsing (Kim et al., 2019; Jin et al., 2019, 2021; Yang et al., 2021b, 2022).

However, context-free rules are not natural for modeling *discontinuous* language phenomena such as extrapositions, cross-serial dependencies, and wh-movements. *Mildly context-sensitive grammars* (Joshi, 1985), which sit between context-free and context-sensitive grammars in the classic Chomsky–Schützenberger hierarchy (Chomsky, 1959; Chomsky and Schützenberger, 1963),[1] are powerful enough to model richer aspects of natural language including discontinuous and non-local phenomena. And despite their expressivity they enjoy polynomial-time inference algorithms, making them attractive both as cognitively plausible models of human language processing and as targets for unsupervised learning.

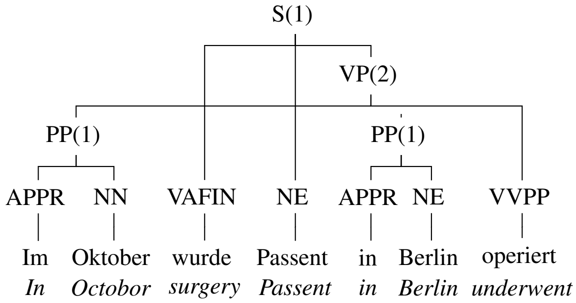


Figure 1: An example of a discontinuous parse tree in German. Each non-leaf node's fan-out is marked in brackets.

There are several weakly equivalent formalisms for generating the mildly context-sensitive languages which might serve as potential targets for grammar induction: tree adjoining grammars (Joshi, 1975), head grammars (Pollard, 1985), combinatory categorial grammars (Steedman, 1987), and linear indexed grammars (Gazdar, 1988). In this paper we work with linear context-free rewriting systems (LCFGS, Vijay-Shanker et al., 1987),

[1] This hierarchy does not necessarily extend to probabilistic grammars. For example Icard (2020) show that in a particular probabilistic version of the hierarchy in which a probabilistic grammar over a one-letter alphabet induces a distribution over the integers via its unary representation, the set of distributions that can be expressed by probabilistic mildly context-sensitive grammars (such as linear indexed grammars) is not a proper subset of the set of distributions that can be expressed by probabilistic context-sensitive grammars.

Code: https://github.com/sustcsonglin/TN-LCFRS.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 5747–5766
July 9-14, 2023 ©2023 Association for Computational Linguistics

which generalize the above formalisms and are weakly equivalent to multiple context-free grammars (Seki et al., 1991). Derivation trees in an LCFRS directly correspond to discontinuous constituency trees where each node can dominate a non-contiguous sequence of words in the yield, as shown in Fig. 1.

We focus on the LCFRS formalism as it has previously been successfully employed for supervised discontinuous constituency parsing (Levy, 2005; Maier, 2010; van Cranenburgh et al., 2016). The complexity of parsing in a LCFRS is $\mathcal{O}(\ell^{3k}|G|)$, where ℓ is the sentence length, k is the fan-out (the maximum number of contiguous blocks of text that can be dominated by a nonterminal), and $|G|$ is the grammar size. While polynomial, this is too high to be practical for unsupervised learning on real-world data. We thus restrict ourselves to LCFRS-2, i.e., binary LCFRS with fan-out two, which has been shown to have high coverage on discontinuous treebanks (Maier et al., 2012). Even with this restriction LCFRS-2 remains difficult to induce from raw text due to the $\mathcal{O}(\ell^6|G|)$ dynamic program for parsing and marginalization. However Corro (2020) observe that a $\mathcal{O}(\ell^5|G|)$ variant of the grammar that discards certain rules can still recover 98% of real world treebank constituents. Our approach uses with this restricted variant of LCFRS-2 (see Sec 2.2). Finally, following recent work which finds that that overparameterizing deep latent variable models is beneficial for unsupervised learning (Buhai et al., 2020; Yang et al., 2021b; Chiu and Rush, 2020; Chiu et al., 2021), we scale LCFRS-2 to a large number of nonterminals by adapting tensor-decomposition-based inference techniques—originally developed for PCFGs (Cohen et al., 2013; Yang et al., 2021b, 2022)—to the LCFRS case.

We conduct experiments German and Dutch—both of which have frequent discontinuous and non-local language phenomena and have available discontinuous treebanks—and observe that our approach is able to induce grammars with nontrivial performance on discontinuous constituents.

2 Approach

2.1 Background: Scaling PCFGs with low-rank neural parameterizations

Inference in PCFGs is cubic with respect to the number of nonterminals in the general case, which can make it difficult to scale up PCFGs to a large number (e.g., thousands) of nonterminals. However, under certain parameterizations it is possible to exploit low rank factorizations of the rule probability tensor to enable faster inference. For example, given a PCFG with m nonterminals Cohen et al. (2013) use canonical-polyadic decomposition (CPD, Rabanser et al., 2017) to decompose the 3D binary rule probability tensor $\mathsf{T} \in \mathbb{R}^{m \times m \times m}$ as,

$$\mathsf{T} = \sum_{q=1}^{r} u_q \otimes v_q \otimes w_q,$$

where $u_q, v_q, w_q \in \mathbb{R}^m$, r is the tensor rank (a hyperparameter), and $\otimes$ is the outer product. Letting $U, V, W \in \mathbb{R}^{r \times m}$ be the matrices resulting from stacking all u_q, v_q, w_q, Cohen et al. (2013) give the following recursive formula for calculating the inside tensor $\alpha \in \mathbb{R}^{(\ell+1)\times(\ell+1)\times m}$ for a sentence of length ℓ:

$$\alpha^L_{i,j} = V\alpha_{i,k}, \quad \alpha^R_{j,k} = W\alpha_{k,j},$$

$$\alpha_{i,j} = U^T \sum_{k=i+1}^{j-1} \alpha^L_{i,j} \circ \alpha^R_{j,k}.$$

Here $\alpha^L, \alpha^R \in \mathbb{R}^{(\ell+1)\times(\ell+1)\times r}$ are auxiliary tensors for storing intermediate values, and $\circ$ is the Hadamard product. The resulting complexity of this version of the inside algorithm is $\mathcal{O}(\ell^3 r + \ell^2 mr)$, which removes the cubic dependence on m. Based on this formula, Yang et al. (2021b) propose a low-rank *neural* parameterization which uses a neural network over shared symbol embeddings to produce unnormalized score matrices $\bar{U}, \bar{V}, \bar{W}$. Then, $\bar{U}$ is $\mathrm{softmax}$-ed across columns to obtain U, while $\bar{V}, \bar{W}$ are $\mathrm{softmax}$-ed across rows to obtain V, W. The difference between Cohen et al. (2013) and Yang et al. (2021b) is that the former performs CPD on an existing probability tensor T for faster (supervised) parsing, whereas the latter directly parameterizes and learns U, V, W from data without actually instantiating T.

Yang et al. (2022) build on Yang et al. (2021b) and further pre-compute matrices $J = VU^T, K = WU^T$ to rewrite the above recursive formula as:

$$\alpha^L_{i,j} = J\alpha'_{i,j} \quad ,\alpha^R_{i,j} = K\alpha'_{i,j}$$

$$\alpha'_{i,j} = \sum_{k=i+1}^{j-1} \alpha^L_{i,j} \circ \alpha^R_{j,k}$$

where $\alpha' \in \mathbb{R}^{(n+1)\times(n+1)\times r}$ is an auxiliary inside score tensor. The resulting complexity of this approach is $\mathcal{O}(\ell^3 r + \ell^2 r^2)$, which is smaller than $\mathcal{O}(\ell^3 r + \ell^2 mr)$ when $r \ll m$, i.e., in the setting with a large number of nonterminals whose probability tensor is of low rank. In this paper we adapt this low rank neural parameterization to the LCFRS case to scale to a large number of nonterminals.

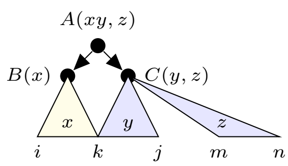
2.2 Restricted LCFRS

In an LCFRS, a single nonterminal node can dominate a tuple of strings that need not be adjacent in the yield. The tuple size is referred to as the *fan-out*. We mark the fan-out of each non-leaf node in Fig. 1. The fan-out of an LCFRS is defined as the maximal fan-out among all its nonterminals, and influences expressiveness and parsing complexity. For a binary LCFRS (i.e., LCFRS with derivation rules that have at most two nonterminals on the right hand side) with fan-out k, the parsing complexity for a sentence of length ℓ is $\mathcal{O}(\ell^{3k})$.[2] In this paper we work with binary LCFRS with fan-out 2 (Stanojević and Steedman, 2020, LCFRS-2), which is expressive enough to model discontinuous constituents but still efficient enough to enable practical grammar induction from natural language data. This choice is also motivated by Maier et al. (2012) who observe that restricting the fan-out to two suffices for capturing a large proportion of discontinuous constituents in standard treebanks.[3]

However, LCFRS-2's inference complexity of $\mathcal{O}(\ell^6|G|)$ is still too expensive for practical unsupervised learning. We thus follow Corro (2020) and discard all rules that require $\mathcal{O}(\ell^6)$ time to parse, which reduces parsing complexity to $\mathcal{O}(\ell^5|G|)$.[4] Formally, this restricted LCFRS-2 is a 6-tuple $\mathcal{G} = (S, \mathcal{N}^1, \mathcal{N}^2, \mathcal{P}, \Sigma, \mathcal{R})$ where: S is the start symbol; $\mathcal{N}^1, \mathcal{N}^2$ are a finite set of nonterminal symbols of fan-out one and two, respectively; $\mathcal{P}$ is a finite set of preterminal symbols; Σ is a finite set of terminal symbols; and $\mathcal{R}$ is a set of rules of the following form (where $\mathcal{M} \triangleq \mathcal{N}^1 \cup \mathcal{P}$):

$$
\begin{aligned}
&S(x) \to A(x) && A \in \mathcal{N}^1 \\
&A(xy) \to B(x)C(y) && A \in \mathcal{N}^1, B, C \in \mathcal{M} \\
&A(yxz) \to B(x)C(y,z) && A \in \mathcal{N}^1, B \in \mathcal{M}, C \in \mathcal{N}^2 \\
&A(x,y) \to B(x)C(y) && A \in \mathcal{N}^2, B, C \in \mathcal{M} \\
&A(xy,z) \to B(x)C(y,z) && A, C \in \mathcal{N}^2, B \in \mathcal{M} \\
&A(yx,z) \to B(x)C(y,z) && A, C \in \mathcal{N}^2, B \in \mathcal{M} \\
&A(y,xz) \to B(x)C(y,z) && A, C \in \mathcal{N}^2, B \in \mathcal{M} \\
&A(y,zx) \to B(x)C(y,z) && A, C \in \mathcal{N}^2, B \in \mathcal{M} \\
&T(w) \to w, && T \in \mathcal{P}, w \in \Sigma.
\end{aligned}
$$

Here $A(x)$ indicates that A has a fan-out 1; $A(x,y)$ indicates that A has a fan-out 2 and x and y are non-adjacent contiguous strings in the yield of A. Each nonterminal is annotated with lower-case letters that stand for strings, and xy denotes the concatenation of x and y, which are adjacent, into a single string $s \triangleq xy$.



Illustrative Example. As an example of how this LCFRS can model discontinuous spans, we depict the rule $A(xy,z) \to B(x)C(y,z)$ above. B is a fan-out-1 node whose yield is $x = w_i \cdots w_{k-1}$ and C is a fan-out-2 node whose first span is $y = w_k \cdots w_{j-1}$ and whose second span is $z = w_m \cdots w_{n-1}$. A is the parent node of B, C, and inherits the yields of B and C, where x is concatenated with y to form a contiguous span and z is a standalone span.

Parsing. Table 1 gives the parsing-as-deduction (Pereira and Warren, 1983) description of the CKY-style chart parsing algorithm of our restricted LCFRS-2.

Item form:	$[A,i,j]$: fan-out-1 node A spanning $[i,j)$ $[A,i,j,k,n]$: fan-out-2 node A spanning $[i,j),[k,n)$	
Axioms:	$[A,i,i+1], 0 \leq i < \ell+1, A \in \mathcal{N}^1$	
Goals:	$[S,0,n]$	
Deductive rules:		
$\dfrac{[B,i,k] \quad [C,k,j]}{[A,i,j]}$	$A(xy) \to B(x)C(y)$ $i<k<j$	1a
$\dfrac{[B,i,j] \quad [C,m,n]}{[A,i,j,m,n]}$	$A(x,y) \to B(x)C(y)$ $i<j<m<n$	1b
$\dfrac{[B,m,n] \quad [C,i,m,n,j]}{[A,i,j]}$	$A(yxz) \to B(x)C(y,z)$ $i<m<n<j$	2a
$\dfrac{[B,i,k] \quad [C,k,j,m,n]}{[A,i,j,m,n]}$	$A(xy,z) \to B(x)C(y,z)$ $i<k<j<m<n$	2b
$\dfrac{[B,k,j] \quad [C,i,k,m,n]}{[A,i,j,m,n]}$	$A(yx,z) \to B(x)C(y,z)$ $i<k<j<m<n$	2c
$\dfrac{[B,m,k] \quad [C,i,j,k,n]}{[A,i,j,m,n]}$	$A(y,xz) \to B(x)C(y,z)$ $i<j<m<k<n$	2d
$\dfrac{[B,m,k] \quad [C,i,j,k,n]}{[A,i,j,m,n]}$	$A(y,zx) \to B(x)C(y,z)$ $i<j<m<k<n$	2e

Table 1: Chart parsing algorithm described in the parsing-as-deduction framework. Here ℓ is the sentence length and we use interstice indices (not word indices) as in Corro (2020).

[2] A binary CFG is thus a special case of a binary LCFRS with fan-out one, and parsing in this case reduces to the classic CKY algorithm.

[3] For instance, Stanojević and Steedman (2020) report that LCFRS-2 can cover up to 87% of the gold discontinuous constituents in the NEGRA treebank. We refer readers to Table 1 of Corro (2020) for more details.

[4] These correspond to rules (d), (i), (j), (k), and (l) in Figure 3 of Corro (2020).

2.3 Tensor decomposition-based neural parameterization

We now describe a parameterization of LCFRS-2 that combines a neural parameterization with tensor decomposition, which makes it possible to scale LCFRS-2 to thousands of nonterminals. Let $m_1 = |\mathcal{N}^1|, m_2 = |\mathcal{N}^2|, p = |\mathcal{P}|$, and $m = m_1 + p$. The rules involving $A \in \mathcal{N}^1$ on the left hand side are 1a and 2a, whose probabilities can be represented by 3D tensors $C^1 \in \mathbb{R}^{m_1 \times m \times m}$ and $D^1 \in \mathbb{R}^{m_1 \times m \times m_2}$. For $A \in \mathcal{N}^2$, the relevant rules are 1b, 2b, 2c, 2d, 2e, whose probabilities can be represented by 3D tensors $C^2 \in \mathbb{R}^{m_2 \times m \times m}$ and $D^3, D^4, D^5, D^6 \in \mathbb{R}^{m_2 \times m \times m_2}$. We stack D^3, D^4, D^5, D^6 into a single 4D tensor $D^2 \in \mathbb{R}^{m_2 \times m \times m_2 \times 4}$ to leverage the structural similarity of these rules. Since these tensors are probabilities, we must have

$$\sum_{j,k} C^1_{ijk} + \sum_{j,k} D^1_{ijk} = 1, \quad \forall i, \tag{1}$$

$$\sum_{j,k} C^2_{ijk} + \sum_{j,k,d} D^2_{ijkd} = 1, \quad \forall i. \tag{2}$$

Tensor decomposition. To scale up the LCFRS-2 to a large number of nonterminals, we first apply CPD on all the binary rule probability tensors,

$$C^1 = \sum_{q=0}^{r_1-1} U^1_{:,q} \otimes V^1_{:,q} \otimes W^1_{:,q}$$

$$C^2 = \sum_{q=0}^{r_2-1} U^2_{:,q} \otimes V^2_{:,q} \otimes W^2_{:,q}$$

$$D^1 = \sum_{q=0}^{r_3-1} U^3_{:,q} \otimes V^3_{:,q} \otimes W^3_{:,q}$$

$$D^2 = \sum_{q=0}^{r_4-1} U^4_{:,q} \otimes V^4_{:,q} \otimes W^4_{:,q} \otimes P_{:,q}$$

where $U_{:,q}$ denotes the q-th column of U. The dimensions of these tensors are $U^1 \in \mathbb{R}^{m_1 \times r_1}$, $V^1, W^1 \in \mathbb{R}^{m \times r_1}$, $U^2 \in \mathbb{R}^{m_1 \times r_2}$, $V^2 \in \mathbb{R}^{m \times r_2}$, $W^2 \in \mathbb{R}^{m_2 \times r_2}$, $U^3, W^3 \in \mathbb{R}^{m_2 \times r_3}$, $U^4, W^4 \in \mathbb{R}^{m_2 \times r_4}$, $V^3 \in \mathbb{R}^{m \times r_3}$, $V^4 \in \mathbb{R}^{m \times r_4}$, and $P \in \mathbb{R}^{4 \times r_4}$. Here r_1, r_2, r_3, r_4 are the ranks of the tensors that control inference complexity. To ensure these factorizations lead to valid probability tensors, 1), we additionally impose the following restrictions: (1) all decomposed matrices are non-negative; (2) P, V^i, W^i are column-wise normalized where $i \in \{1,2,3,4\}$; (3) $\forall i, \sum_j U^1_{ij} + \sum_k U^2_{ik} = 1$; and (4) $\forall i, \sum_j U^3_{ij} + \sum_k U^4_{ik} = 1$. It is easy to verify that Eq. 1 and 2 are satisfied if the above requirements are satisfied.

Rank-space dynamic programming. For unsupervised learning, we need to compute the marginal likelihood of a sentence $p(w_1 w_2 \cdots w_\ell)$. We give the *rank-space* dynamic program (i.e., the inside algorithm) for computing $p(w_1 w_2 \cdots w_\ell)$ in this tensor decomposition-based LCFRS-2 in App. A. The resulting complexity is dominated by $\mathcal{O}(\ell^5 r_4 + \ell^4 (r_3 + r_4)(r_2 + r_4))$. We thus set r_4 to a very small value, which greatly improves runtime.

Parameterization. Following prior work on neural parameterizations of grammars (Jiang et al., 2016; Kim et al., 2019), we parameterize the component matrices to be the output of neural networks over shared embeddings.

The symbol embeddings are given by: $E^1 \in \mathbb{R}^{m \times d}$ where the first m_1 rows correspond to fan-out-1 nonterminal embeddings and the last p rows are the preterminal embeddings; $E^2 \in \mathbb{R}^{m_2 \times d}$ for the fan-out-2 nonterminal embedding matrix; $r \in \mathbb{R}^d$ for the start symbol embedding. We also have four sets of "rank embeddings" $R^1 \in \mathbb{R}^{r_1 \times d}$, $R^2 \in \mathbb{R}^{r_2 \times d}$, $R^3 \in \mathbb{R}^{r_3 \times d}$, and $R^4 \in \mathbb{R}^{r_4 \times d}$. Given this, the entries of the U, V, W matrices are given by,

$$\begin{aligned}
U^o_{ij} &\propto \exp\{(R^o_j)^\top f^o_U(E^1_i)\}, && o \in \{1,2\} \\
U^o_{ij} &\propto \exp\{(R^o_j)^\top f^o_U(E^2_i)\}, && o \in \{3,4\} \\
V^o_{ij} &\propto \exp\{(R^o_j)^\top f^o_V(E^1_i)\}, && o \in \{1,2,3,4\} \\
W^o_{ij} &\propto \exp\{(R^o_j)^\top f^o_W(E^1_i)\}, && o \in \{1,2\} \\
W^o_{ij} &\propto \exp\{(R^o_j)^\top f^o_W(E^2_i)\}, && o \in \{3,4\}
\end{aligned}$$

where f^o_U, f^o_V, f^o_W are one-layer ReLU MLPs with output size d. U^o, V^o, W^o are normalized according to the requirements described in the previous subsection. We share the parameters of the following MLP pairs: (f^1_U, f^2_U), (f^3_U, f^4_U), (f^1_V, f^3_V), (f^2_V, f^4_V), (f^1_W, f^3_W), (f^2_W, f^4_W) as they play similar roles (e.g., f^1_V and f^3_V are both applied to left children). For the D^2 tensor we also require the matrix $P \in \mathbb{R}^{4 \times r_4}$, and this is given by $P^\top = f_P(R^4)$, where f_P is a one-layer residual network with output size 4 that is normalized via a $\mathrm{softmax}$ across the last dimension.

Finally, for the starting and the terminal distributions we have

$$s = f_s(r), \quad Q = f_Q(E^1_{m_1:}),$$

which results in $s \in \mathbb{R}^{m_1}$ (i.e., the probability vector for rules of the form $S \to A$) and $Q \in \mathbb{R}^{p \times v}$ (i.e., probability matrix for rules of the form $T(w) \rightarrow w$). Here $E^1_{m_1:}$ is the last p rows of E^1, and f_s and f_Q are residual MLPs with $\mathrm{softmax}$ applied in the last layer to ensure that s and Q are valid probabilities.

Decoding. While the rank-space inside algorithm enables efficient computation of sentence likelihoods, direct CKY-style argmax decoding in this grammar requires instantiating the full probability tensors and is thus computationally intractable. We follow Yang et al. (2021b) and use Minimal Bayes Risk (MBR) decoding (Goodman, 1996). This procedure first obtains the posterior probability of each span's being a constituent via the inside-outside algorithm (which has the same complexity as the inside algorithm). Then, these posterior probabilities are used as input into CKY in a grammar that only has a single nonterminal. The complexity of this approach is thus independent of the number of nonterminals in the original grammar, and takes $O(\ell^5)$. This strategy can be seen as finding the tree that has the largest number of expected constituents (Smith and Eisner, 2006). See App. A for details.

3 Empirical Study

Data. We conduct experiments with our **T**ensor decomposition-based **N**eural **LCFRS** (TN-LCFRS) on German and Dutch, where discontinuous phenomena are more common (than in English). For German we concatenate TIGER (Brants et al., 2001) and NEGRA (Skut et al., 1997) as our training set, while for Dutch we use the LASSY Small Corpus treebank (van Noord et al., 2013). The data split can be found in App. B.1. For processing we use `disco-dop`[5] (van Cranenburgh et al., 2016) and discard all punctuation marks. We further take the most frequent 10,000 words for each language as the vocabulary, similar to the standard setup in unsupervised constituency parsing on PTB (Shen et al., 2018, 2019; Kim et al., 2019).

Grammar size. To investigate the importance of using a large number of latent variables (which has previously been shown to be helpful for structure induction (Buhai et al., 2020; Yang et al., 2021b)), we train TN-LCFRSs of varying sizes. We first choose the number of preterminals $|\mathcal{P}| \in \{45, 450, 4500\}$ and set the number of fan-out one and fan-out two nonterminals to be $|\mathcal{N}^1| = |\mathcal{N}^2| = \frac{1}{3}|\mathcal{P}|$. The rank of the probability tensors are set to $r_1 = r_3 = 400, r_2 = r_4 = 4$, and the dimensionality of the embedding space is $d = 512$. Model parameters are initialized with Xavier uniform initialization. More training details and hyperparameters can be found in App. B.3 and App. B.4.

Baselines. Our baselines include: the neural PCFG (N-PCFG) and the compound PCFG (C-PCFG) (Kim et al., 2019), which cannot directly predict discontinuous constituents[6] but still serve as strong baselines for overall F1 since the majority of spans in these treebanks are continuous; and their direct extensions, neural LCFRS (N-LCFRS) and compound LCFRS (C-LCFRS), which do not employ the tensor-based low-rank factorization. These non-low-rank models have high computational complexity and hence we set $|\mathcal{P}| = 45$ for these models. When $|\mathcal{P}| = 4500$, we also compare against the tensor decompositional-based neural PCFG (TN-PCFG) from Yang et al. (2021b).

Evaluation. We use unlabeled corpus-level F1 to evaluate unsupervised parsing performance, reporting both overall F1 and discontinuous F1 (DF1). For all experiments, we report the mean results and standard deviations over four runs with different random seeds. See App. B.2 for further details.

3.1 Main results

Table 2 shows the main results. With smaller grammars ($|\mathcal{P}| = 45$), we find that both neural/compound LCFRSs have lower F1 than their PCFG counterparts, despite being able to predict discontinuous constituent spans. On the other hand, TN-LCFRS achieves better F1 than N-LCFRS even though it is a more restricted model (since it assumes that the rule probability tensors are of low rank), showing the benefits of parameter sharing through low rank factorizations. As we scale up TN-LCFRSs with $|\mathcal{P}| \in \{45, 450, 4500\}$ we observe continuous improvements in performance, with TN-LCFRS$_{4500}$ achieving the best F1 and DF1 on all three datasets. These results all outperform trivial (left branching, right branching, and random tree) baselines.

As an upper bound we also train a supervised model with TN-LCFRS$_{4500}$.[7] We also show the

[5] https://github.com/andreasvc/disco-dop

[6] But these models could implicitly model discontinuous constituents with a large number of nonterminals (in the neural PCFG case) and/or with a sentence-level random vector (in the compound PCFG case).

[7] For supervised training we use the optimal binarization from Gildea (2010) to binarize treebanks and remove all trees that are unrecognizable by our restricted LCFRS. We fixed the tree topology (provided by gold binarized tree) and used

Model	$\lvert\mathcal{P}\rvert$	NEGRA F1	NEGRA DF1	TIGER F1	TIGER DF1	LASSY F1	LASSY DF1
N-PCFG	45	40.8$_{\pm 0.5}$	–	39.5$_{\pm 0.4}$	–	40.1$_{\pm 3.9}$	–
C-PCFG	45	39.1$_{\pm 1.9}$	–	38.8$_{\pm 1.3}$	–	37.9$_{\pm 3.4}$	–
N-LCFRS	45	33.7$_{\pm 2.8}$	2.0$_{\pm 0.8}$	32.7$_{\pm 1.8}$	1.2$_{\pm 0.8}$	36.9$_{\pm 1.5}$	0.9$_{\pm 0.8}$
C-LCFRS	45	36.7$_{\pm 1.5}$	2.7$_{\pm 1.4}$	35.2$_{\pm 1.2}$	1.7$_{\pm 1.1}$	36.9$_{\pm 3.7}$	2.2$_{\pm 1.0}$
TN-LCFRS	45	41.1$_{\pm 1.2}$	3.1$_{\pm 2.8}$	40.2$_{\pm 1.1}$	2.3$_{\pm 2.3}$	41.6$_{\pm 3.0}$	2.3$_{\pm 2.3}$
TN-LCFRS	450	45.0$_{\pm 1.8}$	5.6$_{\pm 2.7}$	44.1$_{\pm 1.7}$	4.4$_{\pm 2.3}$	42.9$_{\pm 3.8}$	2.8$_{\pm 3.3}$
TN-PCFG	4500	45.4$_{\pm 0.5}$	–	44.7$_{\pm 0.6}$	–	44.3$_{\pm 6.4}$	–
TN-LCFRS	4500	**46.1**$_{\pm 1.1}$	**8.0**$_{\pm 1.1}$	**45.4**$_{\pm 0.9}$	**6.1**$_{\pm 0.8}$	**45.6**$_{\pm 2.3}$	**8.9**$_{\pm 1.5}$
Supervised	4500	54.4$_{\pm 0.3}$	38.1$_{\pm 1.1}$	50.7$_{\pm 0.2}$	32.1$_{\pm 1.0}$	–	–
Left branching	–	7.8	–	7.9	–	7.2	–
Right branching	–	12.9	–	14.5	–	24.1	–
Random trees	–	7.0$_{\pm 0.1}$	–	7.1$_{\pm 0.2}$	–	9.1$_{\pm 0.4}$	–
Oracle bound	–	64.3	88.5	65.0	86.2	73.7	68.0

Table 2: Results on test sets of German (NEGRA, TIGER) and Dutch (LASSY) treebanks for the various models. $|\mathcal{P}|$ indicates the number of preterminals, which also determines the number of nonterminals ($|\mathcal{N}^1| = |\mathcal{N}^1| = \frac{1}{3}|\mathcal{P}|$), and thus grammar size. F1 is the overall F1 for both continuous and discontinuous spans, while DF1 is the F1 on discontinuous spans only. These results are averaged across four seeds, and $\pm$ indicates standard deviation. Oracle bound shows the upper bound obtainable from binarized trees.

	NP	PP	VP	AP	PN
count	10236	8471	3312	1375	1249
N-PCFG$_{45}$	71.5	78.4	37.5	31.5	44.1
C-PCFG$_{45}$	67.3	79.4	31.1	29.0	51.2
N-LCFRS$_{45}$	60.9	70.5	25.8	29.9	40.8
C-LCFRS$_{45}$	58.6	72.6	28.6	33.0	24.0
TN-LCFRS$_{45}$	73.3	76.1	34.1	27.7	69.7
TN-LCFRS$_{450}$	77.6	**84.2**	30.6	42.8	72.1
TN-PCFG$_{4500}$	76.5	81.8	**51.4**	41.3	67.9
TN-LCFRS$_{4500}$	**78.7**	83.7	46.1	**55.8**	**73.6**
Supervised	78.8	86.1	60.9	74.3	79.0

Table 3: Recall (%) of the most five frequent constituent labels on the TIGER test set.

	VP	NP	PP	AP	AVP
count	1195	395	172	84	71
N-LCFRS$_{45}$	10.3	**4.8**	**1.9**	**2.4**	**2.1**
C-LCFRS$_{45}$	11.8	2.2	1.0	2.7	0.4
TN-LCFRS$_{45}$	6.0	3.0	1.2	0.3	1.1
TN-LCFRS$_{450}$	11.9	2.2	0.3	1.2	0.4
TN-LCFRS$_{4500}$	**19.9**	2.5	0.0	0.9	0.4
Supervised	23.7	14.1	31.7	18.5	25.4

Table 4: Recall (%) of the most five frequent discontinuou constituent labels on the TIGER test set.

maximum possible performance with oracle binary trees with this optimal binarization.

While the discontinuous F1 of our unsupervised parsers are nontrivial, there is still a large gap between the unsupervised and supervised scores (and also between the supervised and the oracle scores), indicating opportunities for further work in this area.

3.2 Analysis

Recall by constituent label. Table 3 shows the recall by constituent tag for the different models averaged over four independent runs. Overall the unsupervised methods do well on noun phrases (NP), prepositional phrases (PP) and proper nouns (PN), with some of the models approach the supervised baselines. Verb phrases (VP) and adjective phrases (AP) remain challenging. Table 4 has recall by label for discontinuous constituents only, where we observe that most discontinuous constituents are VPs. In App. C , we also show F1/DF1 broken down by sentence length.

Approximation error. Approximation error in the context of unsupervised learning arises due to the mismatch between the EM objective (i.e., log marginal likelihood) and structure recovery (i.e., F1), and is related to model misspecification (Liang and Klein, 2008). Figure 2 (left column) plots the training/dev perplexity as well as the dev F1/DF1 as a function of the number of epochs. We find that larger grammars result in better performance in terms of both perplexity and structure recovery, which ostensibly indicates that the unsupervised objective is positively correlated with structure induction performance.

However, when we first perform supervised learning on the log joint likelihood and then switch to unsupervised learning with log marginal likelihood (Figure 2, right), we find that while perplexity improves when we switch to the unsupervised

dynamic programming to sum out all possible nonterminals for each node, resulting in the joint log probability of unlabeled binarized tree and sentence. This was then maximized during training. As for the oracle bound, we emphasize that the gold trees are nonbinary while our model can only predict binary trees.

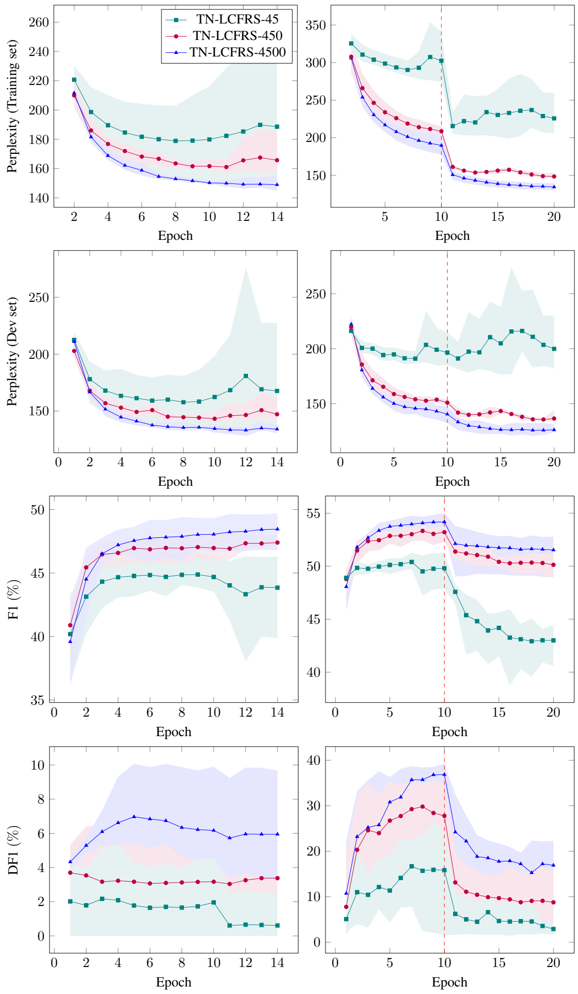


Figure 2: On the rows we have the German (TIGER) training set perplexities, dev set perplexities, overall F1, and discontinuous F1 (DF1) for TN-LCFRSs of various sizes as a function of training epochs. Colored regions indicate min/max values across four runs. Left column shows pure unsupervsed learning, while right column shows case where we train with supervised learning for 10 epochs and then switch to unsupervised learning (indicated by dashed lines).

objective, structure induction performance deteriorates.[8] Still, the difference in F1 before and after switching to the unsupervised objective is less for larger models, confirming the benefits of using larger grammars.

Even more restricted LCFRS formalisms. There are even more restricted versions of LCFRSs which have faster parsing (e.g. $\mathcal{O}(\ell^3)$, $\mathcal{O}(\ell^4)$) but

[8]It is worth noting that the phenomenon of mismatch between log marginal likelihood objective and parsing accuracy is quite common in unsupervised grammar induction (and latent variable modeling approaches to structured induction more generally). Many previous works have observed this phenomenon, e.g., Merialdo (1994) in the context of HMMs, and Johnson et al. (2007) and Liang and Klein (2008) in the context of PCFGs. This is partially attributed to the fact that generative grammars often make some unreasonable independence assumptions to make the training process tractable, which does not fully comply with the true generative process of human languages and their underlying structures.

Model	**NEGRA** F1	DF1	**TIGER** F1	DF1
TN-LCFRS$_{4500}$	46.1	8.0	45.4	6.1
w/o $\mathcal{O}(n^5)$ rules	46.4	4.0	45.3	3.0
w/o shared MLPs	44.4	6.7	43.6	5.3
w/o shared emb.	45.4	0.9	44.5	0.5

Table 5: Ablation studies on the German (TIGER) treebank.

can still model discontinuous constituents. In the supervised case, these restricted variants have been shown to perform almost as well as the more expressive $\mathcal{O}(\ell^5)$ and $\mathcal{O}(\ell^6)$ variants (Corro, 2020). In the unsupervised case however, we observe in Table 5 that disallowing $\mathcal{O}(\ell^5)$ rules (2b, 2c, 2d, 2e) significantly degrades discontinuous F1 scores. We posit that this phenomena is again related to empirical benefits of latent variable overparameterization—while in *theory* it is possible to model most discontinuous phenomena with more restricted rules, making the generative model more expressive via "overparameterizing" in rule expressivity space (i.e., using more flexible rules than is necessariy) empirically leads to better performance.

Parameter sharing. As shown in Table 5, it was important to share the symbol embeddings across the different rules. Sharing the parameters of the MLPs as described in Sec. 2.3 was also found to be helpful. This highlights the benefits of working with neural parameterizations of grammars which enable easy parameter sharing across rules that share symbols and/or have similar shapes.

Qualitative analysis. In Fig. 3, we show some examples trees in German. For each sentence, we show the gold, TN-LCFRS$_{4500}$, and TN-PCFG$_{4500}$ trees. In the first sentence, the crossing dependency occurs due to the initial adverb ("So")'s being analyzed as a dependent of the non-finite verb phrase at the end of the sentence which occurs due to German V2 word order. Our parser correctly predicts this dependency, although the subject NP (which itself is correctly identified) has the wrong internal structure. The second sentence highlights a case of partial success with right-extraposed relative clauses. While our model is able to correctly predict the top-level discontinuous constituent "[Für 15 200 Mark]–[Lampen einbauen lassen die mutwilligen Zerstörungen standhalten]", the parser does not adopt a discontinuous-constituency analysis of the right-extraposed relative clause itself ("[Lampen]–[die mutwilligen Zerstörungen standhalten]"). Instead it makes the

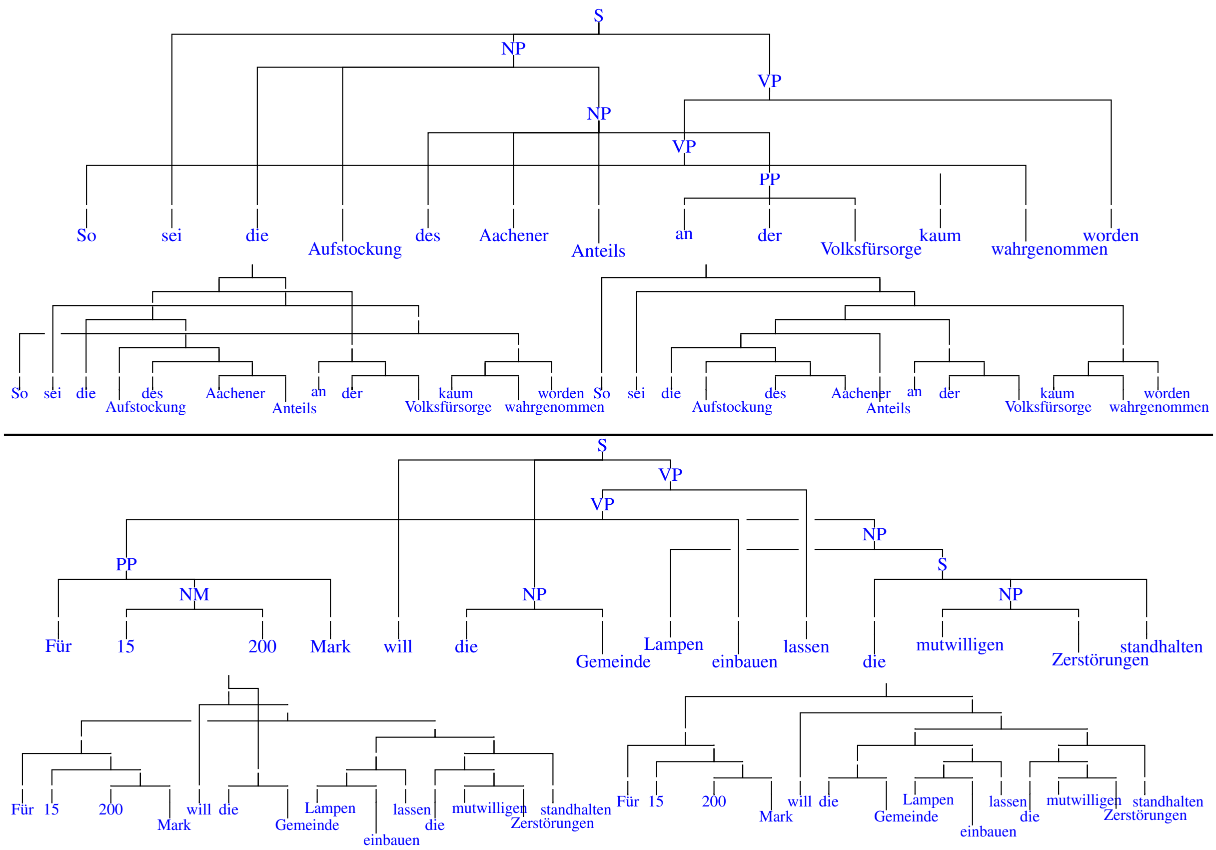


Figure 3: Examples of two sentences from the German NEGRA tree bank. In each example, the gold tree is shown at the top, the TN-LCFRS$_{4500}$ is shown bottom left, and the TN-PCFG$_{4500}$ tree is shown bottom right.

relative clause a part of the non-finite verb complex, which does not conform to the annotation guidelines but resembles an alternative analysis that has been proposed for extraposed relative clauses (Baltin, 1983).

Sentence initial adverbs in the context of auxiliary verb constructions and right-extraposed relative clauses describe two common instances of discontinuous phenomena in German. Wh- questions constitute another potential class of discontinuous phenomena; however, these are not treated as discontinuous in TIGER/NEGRA. See App. D for more examples trees (including on Dutch).

4 Related work

Mildly context-sensitive grammars. Given the evidence against the context-freeness of natural language (Shieber, 1985), mildly context-sensitive grammars such as tree adjoining grammars were thought to be just flexible (but still constrained) enough to model natural language (Joshi, 1985). Prior work on inducing mildly context-sensitive grammars has generally focused on combinatory categorial grammars (Bisk and Hockenmaier, 2012, 2013), and we are unaware of any work on inducing LCFRSs from observed yields alone. Our work is also related to the rich line of work on supervised discontinuous parsing (Kallmeyer and Maier, 2010; Maier et al., 2012; Maier, 2015; Corro, 2020; Vilares and Gómez-Rodríguez, 2020; Fernández-González and Gómez-Rodríguez, 2020, 2021, 2023), though we are unaware of any prior work on unsupervised discontinuous parsing.

Neural grammars. Early work on probabilistic approaches to grammar induction was largely negative (Lari and Young, 1990; Carroll and Charniak, 1992). However, recent work has shown that neural parameterizations of classic grammars can greatly improve structure induction. Our work adds to the line of work on neural parameterizations of dependency models (Jiang et al., 2016; Han et al., 2017; He et al., 2018; Yang et al., 2020), context-free grammars (Kim et al., 2019; Jin et al., 2019; Zhu et al., 2020; Yang et al., 2021a), and synchronous grammars (Kim, 2021; Wang et al., 2022; Friedman et al., 2022). Neural parameterizations make it easy to share parameters and condition on additional side information (images/audio/video) which has shown to be particularly useful for multimodal grammar induction (Zhao and Titov, 2020; Jin and

Schuler, 2020; Su et al., 2021; Hong et al., 2021; Zhang et al., 2021).

Scaling latent variable models. Buhai et al. (2020) study the empirical benefits of overparameterization in learning latent variable models. Other works have explored parameterizations of latent variable models that make it especially amenable to scaling (Chiu and Rush, 2020; Chiu et al., 2021; Yang et al., 2021b, 2022). Relatedly, Peharz et al. (2020) and Liu et al. (2022) show the benefits of scaling probabilistic circuits (Choi et al., 2020).

5 Conclusion

This work studied unsupervised discontinuous constituency parsing with mildly context-sensitive grammars, focusing on the formalism of linear context-free rewriting systems. By using a tensor decomposition-based neural parameterization of linear context-free rewriting systems, our approach was able to induce grammars that had nontrivial discontinuous parsing performance on German and Dutch. Whether even more expressive grammars will eventually lead to models learn linguistically meaningful structures and are at the same time competitive with pure neural language models (as a language model) remains an open question.

Limitations

There are several limitations of our work. We tried training the TN-LCFRS on the discontinuous version of the English Penn Treebank (DPTB, Evang and Kallmeyer, 2011) but failed to induce any meaningful discontinuous structures. This is possibly because discontinuous phenomena in English are much less common than in German and Dutch. For example, while 5.67% of the gold constituents are discontinuous in NEGRA, only 1.84% gold constituents are discontinuous in DPTB (Corro, 2020).

The neural LCFRS was also quite sensitive to hyperparameters and parameterization. The instability of unsupervised structure induction is widely acknowledged and could potentially be mitigated by a large amount of training data, as suggested by Liang and Klein (2008) and Pate and Johnson (2016). Due to this sensitivity, we rely on dev sets for some modeling choices (e.g., rank of the probability tensors). Hence, our approach is arguably not fully unsupervised in the strictest sense of the term, although this is a common setup in unsupervised parsing due to the mismatch between the unsupervised learning objective and structure recovery. (However see Shi et al. (2020) for a critical discussion of this approach.)

Finally, while we observed significant increases in performance as we scaled up the number of nonterminals, we also observed diminishing returns. Further scaling up the grammar is thus unlikely to close the (large) gap that still exists between the unsupervised and supervised parsing results.

Ethics Statement

We foresee no ethical concerns with this work.

Acknowledgment

SY was supported by the National Natural Science Foundation of China (61976139). This study was also supported by funds from an MIT-IBM Watson AI Lab grant.

References

Mark R. Baltin. 1983. Extraposition: Bounding versus government-binding. *Linguistic Inquiry*, 14(1):155–162.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML '09*.

Yonatan Bisk and Julia Hockenmaier. 2012. Simple robust grammar induction with combinatory categorial grammars. In *Proceedings of AAAI*.

Yonatan Bisk and Julia Hockenmaier. 2013. An HDP model for inducing Combinatory Categorial Grammars. *Transactions of the Association for Computational Linguistics*, 1:75–88.

Gosse Bouma and Gertjan van Noord. 2017. Increasing return on annotation investment: The automatic construction of a Universal Dependency treebank for Dutch. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 19–26, Gothenburg, Sweden. Association for Computational Linguistics.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2001. The tiger treebank.

Rares-Darius Buhai, Yoni Halpern, Yoon Kim, Andrej Risteski, and David A. Sontag. 2020. Empirical study of the benefits of overparameterization in learning latent variable models. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1211–1219. PMLR.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora.

David Chiang and Darcey Riley. 2020. Factor graph grammars. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Justin Chiu and Alexander Rush. 2020. Scaling hidden Markov language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1341–1349, Online. Association for Computational Linguistics.

Justin T. Chiu, Yuntian Deng, and Alexander M. Rush. 2021. Low-rank constraints for fast inference in structured models. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 2887–2898.

YooJung Choi, Antonio Vergari, and Guy Van den Broeck. 2020. Probabilistic circuits: A unifying framework for tractable probabilistic models.

N. Chomsky and M. P. Schützenberger. 1963. The algebraic theory of context-free languages. *Studies in Logic and the Foundations of Mathematics*, 35(C):118–161.

Noam Chomsky. 1959. On certain formal properties of grammars. *Inf. Control.*, 2(2):137–167.

Shay B. Cohen, Giorgio Satta, and Michael Collins. 2013. Approximate PCFG parsing using tensor decomposition. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 487–496, Atlanta, Georgia. Association for Computational Linguistics.

Caio Corro. 2020. Span-based discontinuous constituency parsing: a family of exact chart-based algorithms with time complexities from O(n^6) down to O(n^3). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2753–2764, Online. Association for Computational Linguistics.

Amit Dubey and Frank Keller. 2003. Probabilistic parsing for German using sister-head dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo, Japan. Association for Computational Linguistics.

Jason Eisner. 2016. Inside-outside and forward-backward algorithms are just backprop (tutorial paper). In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 1–17, Austin, TX. Association for Computational Linguistics.

Kilian Evang and Laura Kallmeyer. 2011. PLCFRS parsing of English discontinuous constituents. In *Proceedings of the 12th International Conference on Parsing Technologies*, pages 104–116, Dublin, Ireland. Association for Computational Linguistics.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2020. Discontinuous constituent parsing with pointer networks. In *Proceedings of AAAI*.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2021. Reducing discontinuous to continuous parsing with pointer network reordering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2023. Discontinuous grammar as a foreign language. *Neurocomputing*, 524:43–58.

Dan Friedman, Alexander Wettig, and Danqi Chen. 2022. Finding Dataset Shortcuts with Grammar Induction. In *Proceedings of EMNLP*.

Gerald Gazdar. 1988. Applicability of Indexed Grammars to Natural Languages. In *Natural Language Parsing and Linguistic Theories*, pages 69–94.

Daniel Gildea. 2010. Optimal parsing strategies for linear context-free rewriting systems. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 769–776, Los Angeles, California. Association for Computational Linguistics.

Joshua Goodman. 1996. Parsing algorithms and metrics. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 177–183, Santa Cruz, California, USA. Association for Computational Linguistics.

Wenjuan Han, Yong Jiang, and Kewei Tu. 2017. Dependency grammar induction with neural lexicalization and big training data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1683–1688, Copenhagen, Denmark. Association for Computational Linguistics.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Unsupervised learning of syntactic structure with invertible neural projections. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, Brussels, Belgium. Association for Computational Linguistics.

Yining Hong, Qing Li, Song-Chun Zhu, and Siyuan Huang. 2021. VLGrammar: Grounded Grammar Induction of Vision and Language. *arXiv:2103.12975*.

Thomas Icard. 2020. Calibrating generative models: The probabilistic chomsky-schützenberger hierarchy. *Journal of Mathematical Psychology*, 95.

Yong Jiang, Wenjuan Han, and Kewei Tu. 2016. Unsupervised neural dependency parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 763–771, Austin, Texas. Association for Computational Linguistics.

Lifeng Jin, Finale Doshi-Velez, Timothy Miller, Lane Schwartz, and William Schuler. 2019. Unsupervised learning of PCFGs with normalizing flow. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2442–2452, Florence, Italy. Association for Computational Linguistics.

Lifeng Jin, Byung-Doh Oh, and William Schuler. 2021. Character-based PCFG induction for modeling the syntactic acquisition of morphologically rich languages. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4367–4378, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lifeng Jin and William Schuler. 2020. Grounded PCFG induction with images. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 396–408, Suzhou, China. Association for Computational Linguistics.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 139–146, Rochester, New York. Association for Computational Linguistics.

Aravind K. Joshi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163.

Aravind K Joshi. 1985. How much context sensitivity is necessary for characterizing structural descriptions: Tree adjoining grammars. *Natural language parsing: Psychological, computational and theoretical perspectives*, pages 206–250.

Laura Kallmeyer and Wolfgang Maier. 2010. Data-driven parsing with probabilistic linear context-free rewriting systems. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 537–545, Beijing, China. Coling 2010 Organizing Committee.

Yoon Kim. 2021. Sequence-to-sequence learning with latent neural grammars. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 26302–26317.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Dan Klein and Christopher D. Manning. 2001. Parsing and hypergraphs. In *Proceedings of the Seventh International Workshop on Parsing Technologies (IWPT-2001), 17-19 October 2001, Beijing, China*. Tsinghua University Press.

Karim Lari and Steve J Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer speech & language*, 4(1):35–56.

Roger Levy. 2005. *Probabilistic Models of Word Order and Syntactic Discontinuity*. Ph.D. thesis, Stanford University.

Percy Liang and Dan Klein. 2008. Analyzing the errors of unsupervised learning. In *Proceedings of ACL-08: HLT*, pages 879–887, Columbus, Ohio. Association for Computational Linguistics.

Anji Liu, Honghua Zhang, and Guy Van den Broeck. 2022. Scaling up probabilistic circuits by latent variable distillation. *CoRR*, abs/2210.04398.

Wolfgang Maier. 2010. Direct parsing of discontinuous constituents in German. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 58–66, Los Angeles, CA, USA. Association for Computational Linguistics.

Wolfgang Maier. 2015. Discontinuous incremental shift-reduce parsing. 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 1202–1212, Beijing, China. Association for Computational Linguistics.

Wolfgang Maier, Miriam Kaeshammer, and Laura Kallmeyer. 2012. PLCFRS parsing revisited: Restricting the fan-out to two. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11)*, pages 126–134, Paris, France.

Bernard Merialdo. 1994. Tagging English Text with a Probabilistic Model. *Computational Linguistics*, 20(2):155–171.

John K Pate and Mark Johnson. 2016. Grammar induction from (lots of) words alone. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 23–32, Osaka, Japan. The COLING 2016 Organizing Committee.

Robert Peharz, Steven Lang, Antonio Vergari, Karl Stelzner, Alejandro Molina, Martin Trapp, Guy Van den Broeck, Kristian Kersting, and Zoubin Ghahramani. 2020. Einsum networks: Fast and scalable learning of tractable probabilistic circuits. In *Proceedings of ICML*.

Fernando C. N. Pereira and David H. D. Warren. 1983. Parsing as deduction. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 137–144, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Carl Pollard. 1985. *Generalized phrase structure grammars, head grammars and natural language*. Ph.D. thesis, Stanford University.

Stephan Rabanser, Oleksandr Shchur, and Stephan Günnemann. 2017. Introduction to tensor decompositions and their applications in machine learning. *CoRR*, abs/1711.10781.

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.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theor. Comput. Sci.*, 88:191–229.

Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron Courville. 2018. Neural Language Modeling by Jointly Learning Syntax and Lexicon. In *Proceedings of ICLR*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered Neurons: Integrating Tree Structures into Recurrent Neural Networks. In *Proceedings of ICLR*.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2020. On the role of supervision in unsupervised constituency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7611–7621, Online. Association for Computational Linguistics.

Stuart Shieber. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–43.

Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997. An annotation scheme for free word order languages. In *Fifth Conference on Applied Natural Language Processing*, pages 88–95, Washington, DC, USA. Association for Computational Linguistics.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 787–794, Sydney, Australia. Association for Computational Linguistics.

Miloš Stanojević and Mark Steedman. 2020. Span-based LCFRS-2 parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 111–121, Online. Association for Computational Linguistics.

Mark Steedman. 1987. Combinatory Grammars and Parasitic Gaps. *Natural Language and Linguistic Theory*, 5:403–439.

Ruisi Su, Shruti Rijhwani, Hao Zhu, Junxian He, Xinyu Wang, Yonatan Bisk, and Graham Neubig. 2021. Dependency induction through the lens of visual perception. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 17–26, Online. Association for Computational Linguistics.

Andreas van Cranenburgh, Remko J. H. Scha, and Rens Bod. 2016. Data-oriented parsing with discontinuous constituents and function tags. *J. Lang. Model.*, 4:57–111.

Leonoor van der Beek, Gosse Bouma, Robert Malouf, and Gertjan van Noord. 2001. The alpino dependency treebank. In *CLIN*.

Gertjan van Noord, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste. 2013. Large scale syntactic annotation of written dutch: Lassy. In Peter Spyns and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch, Results by the STEVIN-programme*, Theory and Applications of Natural Language Processing, pages 147–164. Springer.

K. Vijay-Shanker, David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *25th Annual Meeting of the Association for Computational Linguistics*, pages 104–111, Stanford, California, USA. Association for Computational Linguistics.

David Vilares and Carlos Gómez-Rodríguez. 2020. Discontinuous constituent parsing as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2771–2785, Online. Association for Computational Linguistics.

Bailin Wang, Ivan Titov, Jacob Andreas, and Yoon Kim. 2022. Hierarchical Phrase-based Sequence-to-Sequence Learning. In *Proceedings of EMNLP*.

Songlin Yang, Yong Jiang, Wenjuan Han, and Kewei Tu. 2020. Second-order unsupervised neural dependency parsing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3911–3924, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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.

Songyang Zhang, Linfeng Song, Lifeng Jin, Kun Xu, Dong Yu, and Jiebo Luo. 2021. Video-aided unsupervised grammar induction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1513–1524, Online. Association for Computational Linguistics.

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

Hao Zhu, Yonatan Bisk, and Graham Neubig. 2020. The return of lexical dependencies: Neural lexicalized PCFGs. *Transactions of the Association for Computational Linguistics*, 8:647–661.

A Fast LCFRS Inference with CPD

Yang et al. (2022) propose a family of CPD-based algorithms for fast inference in B-FGGs which combine B-graphs (Klein and Manning, 2001) and factor graph grammars (FGG, Chiang and Riley, 2020). Inference in LCFRS is subsumed by B-FGG because for each rule, the number of variables in the left-hand side is always one. As such, we can adopt the method of Yang et al. (2022) to perform fast dynamic programming inference in "rank space" for our restricted LCFRS-2.

Concretely, for a length-ℓ sentence $x_0, \dots x_{n-1}$ (x_j is the index in the terminal vocabulary), let $N = n + 1$. The inside scores defined in the rank-space (similar to Sec. 2.1) are,

- $\alpha^{A_1}, \alpha^{B_1}, \alpha^{C_1} \in \mathbb{R}^{N \times N \times r_1}$: corresponding to A, B, C in rule 1a.
- $\alpha^{A_2} \in \mathbb{R}^{N \times N \times r_2}$, $\alpha^{B_2} \in \mathbb{R}^{N \times N \times N \times N \times r_2}$, $\alpha^{C_2} \in \mathbb{R}^{N \times N \times r_2}$: corresponding to A, B, C in rule 2a.
- $\alpha^{A_3} \in \mathbb{R}^{N \times N \times N \times N \times r_3}$, $\alpha^{B_3}, \alpha^{C_3} \in \mathbb{R}^{N \times N \times r_3}$: corresponding to A, B, C in rule 1b.
- $\alpha^{A_4} \in \mathbb{R}^{N \times N \times N \times N \times r_4}$, $\alpha^{B_4} \in \mathbb{R}^{N \times N \times r_4}$, $\alpha^{C_4} \in \mathbb{R}^{N \times N \times N \times N \times r_4}$: corresponding to A, B, C in rule 2b, 2c, 2d, 2e.

The base cases are,

$$\alpha^{B_o}_{i,i+1} = (Q_{:,x_i})^T V^o_{m_1:} \qquad o \in \{1,2,3,4\}$$
$$\alpha^{C_o}_{i,i+1} = (Q_{:,x_i})^T W^o_{m_1:} \qquad o \in \{1,3\}$$

where $Q_{:,x_i}$ is the x_i-th column of Q. The recursive DP computation formulas are,

$$\begin{aligned}
\alpha^{A_1}_{ij} &= \sum_{i<k<j} \alpha^{B_1}_{ik} \circ \alpha^{C_1}_{kj} \\
\alpha^{A_2}_{ij} &= \sum_{i<m<n<j} \alpha^{B_2}_{mn} \circ \alpha^{C_2}_{imnj} \\
\alpha^{A_3}_{imnj} &= \alpha^{B_3}_{im} \circ \alpha^{C_3}_{nj}
\end{aligned} \tag{3}$$

$$\begin{aligned}
\alpha^{A_4}_{imnj} &= \sum_{i<k<m} \alpha^{B_4}_{ik} \circ \alpha^{C_4}_{kmnj} \circ P_0 \\
&+ \sum_{i<k<m} \alpha^{B_4}_{km} \circ \alpha^{C_4}_{iknj} \circ P_1 \\
&+ \sum_{n<k<j} \alpha^{B_4}_{nk} \circ \alpha^{C_4}_{imkj} \circ P_2 \\
&+ \sum_{n<k<j} \alpha^{B_4}_{kj} \circ \alpha^{C_4}_{imnk} \circ P_3
\end{aligned} \tag{4}$$

Items:

I $[i, j]$: accumulated scores for continuous spans.

II $[i, j, k, n]$: accumulated scores for discontinuous spans.

Deductive rules:

$$\frac{[i,k] \quad [k,j]}{[i,j]} X_{ij}$$
$$\frac{[i,j] \quad m,n]}{[i,j,m,n]} Y_{ijmn}$$
$$\frac{[m,n] \quad [i,m,n,j]}{[i,j]} X_{ij}$$
$$\frac{[i,k] \quad [k,j,n,j]}{[i,j,m,n]} Y_{ijmn}$$
$$\frac{[k,j] \quad [i,k,m,n]}{[i,j,m,n]} Y_{ijmn}$$
$$\frac{[m,k] \quad [i,j,k,n]}{[i,j,m,n]} Y_{ijmn}$$
$$\frac{[m,k] \quad [i,j,k,n]}{[i,j,m,n]} Y_{ijmn}$$

Table 6: CKY-style parsing with span marginals.

$$\begin{aligned}
\alpha^{B_o}_{ij} &= F^o \alpha^{A_1}_{ij} + G^o \alpha^{A_2}_{ij} & o \in \{1,2,3,4\} \\
\alpha^{C_o}_{ij} &= H^o \alpha^{A_1}_{ij} + I^o \alpha^{A_2}_{ij} & o \in \{1,3\} \\
\alpha^{C_o}_{imnj} &= J^o \alpha^{A_3}_{imnj} + K^o \alpha^{A_4}_{imnj} & o \in \{2,4\}
\end{aligned} \tag{5}$$

where

$$\begin{aligned}
F^o &= V^o_{:m_1} (U^1)^T & o \in \{1,2,3,4\} \\
G^o &= V^o_{:m_1} (U^2)^T & o \in \{1,2,3,4\} \\
H^o &= W^o_{:m_1} (U^1)^T & o \in \{1,3\} \\
I^o &= W^o_{:m_1} (U^2)^T & o \in \{1,3\} \\
J^o &= W^o (U^3)^T & o \in \{2,4\} \\
K^o &= W^o (U^4)^T & o \in \{2,4\}
\end{aligned}$$

can pre-computed before inference. The partition function Z (i.e., the sentence likelihood) is then given by,

$$Z = R_1 \alpha^{A_1}_{0n} + R_2 \alpha^{A_2}_{0n}$$

where $R_1 = s^T U^1$ and $R_2 = s^T U^2$.

Time complexity. From the above we can see that Eq. 3 takes $\mathcal{O}(\ell^4 r_3)$, Eq. 4 takes $\mathcal{O}(\ell^5 r_4)$, and Eq. 5 takes $\mathcal{O}(\ell^4 (r_2 + r_4)(r_3 + r_4))$. Therefore the total time complexity is dominated by $\mathcal{O}(\ell^5 r_4 + \ell^4 (r_2 + r_4)(r_3 + r_4))$.

MBR decoding. MBR decoding aims to find the best parse with maximum expected number of constituent spans, which can be decomposed into two steps: i) span marginal estimation, and ii)

Model	$\vert\mathcal{P}\vert$	NEGRA		TIGER		LASSY	
		F1	DF1	F1	DF1	F1	DF1
N-PCFG	45	41.3	–	40.0	–	45.5	–
C-PCFG	45	40.2	–	39.8	–	40.9	–
N-LCFRS	45	37.0	3.4	35.6	2.0	39.4	1.7
C-LCFRS	45	38.2	4.3	36.4	3.0	42.4	3.7
TN-LCFRS	45	42.5	5.5	41.3	4.4	44.4	4.6
TN-LCFRS	450	47.1	8.4	45.9	6.4	47.0	8.1
TN-LCFRS	4500	**47.2**	**9.7**	**46.6**	**7.3**	48.0	**10.2**
TN-PCFG	4500	46.2	–	45.5	–	**50.0**	–
Supervised	4500	54.8	39.2	50.9	33.3	–	–

Table 7: Maximum F1 results across four random seeds on the German (NEGRA, TIGER) and Dutch (LASSY) test sets.

CKY-style parsing with marginals. Denote continuous and discontinuous span marginals as $X \in \mathbb{R}^{N\times N}$ and $Y \in \mathbb{R}^{N\times N\times N\times N}$ with $\sum_{ij} X_{ij} + \sum_{ijmn} Y_{ijmn} = 2\ell - 1$. Span marginals can be estimated via inside-outside, or equivalently, backprogation on the inside algorithm (Eisner, 2016, Sec. 6.2), i.e.,

$$X_{ij} = \sum_{r} \sum_{o\in\{1,2\}} \frac{\partial \log Z}{\partial \log \alpha^{A_o}_{ijr}},$$

$$Y_{imnj} = \sum_{r} \sum_{o\in\{3,4\}} \frac{\partial \log Z}{\partial \log \alpha^{A_o}_{imnjr}}.$$

The second-stage CKY-style parsing is similar to the description in Table 1, except that the grammar rule probabilities are replaced with span marginals, as described in Table 6. The total time complexity is dominated by the first stage of marginal estimation, whose complexity is the same as that of the inside algorithm (Eisner, 2016).

B Experimental Details

B.1 Data split

For German, we follow Corro (2020) and use the NEGRA treebank (Skut et al., 1997) with the split proposed by Dubey and Keller (2003), and the TIGER treebank (Brants et al., 2001) with the split provided by the SPRML 2014 shared task (Seddah et al., 2014). For Dutch, there is no standard split in the discontinuous parsing literature. We follow `UD-Dutch-Alpino` (Bouma and van Noord, 2017) and use a hybrid training dataset that comprises the whole Alpino treebank (van der Beek et al., 2001) and a subset of LASSY Small Corpus (van Noord et al., 2013). We further use the whole `WR-P-P-H` section and `WR-P-P-L` section as the development and test sets, respectively.

B.2 Evaluation metric details

Following standard practice in unsupervised parsing evaluation, we ignore all trivial continuous spans, i.e., whole-sentence spans and single-word spans. In addition, we ignore all discontinuous spans of fan-out greater than two. Finally, we evaluate only on sentences of length up to 40 due to computational considerations.

B.3 Training details

For training, we use a curriculum training strategy (Bengio et al., 2009) where we train only on sentences of length up to 30 in the first epoch, and increase the maximum length by five for each epoch until we reach the maximum sentence length (60 for Dutch and 40 for German). We use the Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = 0.75, \beta_2 = 0.999$, learning rate 0.002, batch size 20, and a maximum gradient norm limit of 3. We train for 20 epochs and perform early stopping strategy based on the performance of development set with maximum patience 5.

B.4 Choice of hyperparameters

We assumed a 1:1 ratio between the numbers of fan-out one and fan-out two nonterminals and tuned the ratio of the number of fan-out one nonterminals to preterminals from $\{\frac{1}{2}, \frac{1}{3}, \frac{1}{4}\}$. Since C-LCFRS and N-LCFRS are computationally expensive without tensor decomposition, we could only use up to 45 preterminals and 15 fan-out one/two nonterminals. We then scaled up our approach by a factor of 10 and 1000 to study the benefits of overparameterization, which resulted in our final choice.

Regarding rank size, we used as much as we could while keeping the ratio of $\frac{r_1}{r_3} = \frac{r_2}{r_4} = 100$. To save tuning time, we assumed $r_1 = r_2$ and

Model	$\lvert\mathcal{P}\rvert$	**TIGER-10**		**TIGER-20**		**TIGER-30**	
		F1	DF1	F1	DF1	F1	DF1
N-PCFG	45	$47.7_{\pm 0.9}$	–	$42.5_{\pm 0.2}$	–	$40.5_{\pm 0.2}$	–
C-PCFG	45	$48.1_{\pm 1.1}$	–	$41.7_{\pm 1.3}$	–	$39.7_{\pm 1.2}$	–
N-LCFRS	45	$41.7_{\pm 2.4}$	$3.2_{\pm 1.4}$	$36.3_{\pm 2.4}$	$2.7_{\pm 1.0}$	$34.5_{\pm 2.5}$	$2.2_{\pm 0.8}$
C-LCFRS	45	$42.5_{\pm 1.6}$	$2.7_{\pm 1.6}$	$37.7_{\pm 1.2}$	$2.3_{\pm 1.3}$	$36.0_{\pm 1.1}$	$1.9_{\pm 1.0}$
TN-LCFRS	45	$48.3_{\pm 1.4}$	$1.9_{\pm 2.3}$	$42.8_{\pm 0.9}$	$1.6_{\pm 1.9}$	$41.0_{\pm 1.0}$	$1.4_{\pm 1.6}$
TN-LCFRS	450	$51.4_{\pm 1.8}$	$6.1_{\pm 1.7}$	$46.1_{\pm 1.7}$	$5.5_{\pm 1.9}$	$44.5_{\pm 1.7}$	$4.8_{\pm 1.8}$
TN-PCFG	4500	$52.4_{\pm 0.4}$	$0.0_{\pm 0.0}$	$47.6_{\pm 0.5}$	–	$45.8_{\pm 0.5}$	–
TN-LCFRS	4500	$\mathbf{52.9}_{\pm 1.3}$	$\mathbf{8.2}_{\pm 2.0}$	$\mathbf{47.9}_{\pm 1.1}$	$\mathbf{7.4}_{\pm 1.1}$	$\mathbf{46.3}_{\pm 0.9}$	$\mathbf{6.4}_{\pm 1.0}$
Oracle bound		64.3	88.5	65.0	86.2	73.7	68.0

Table 8: Results on TIGER test set by broken down by sentence length.

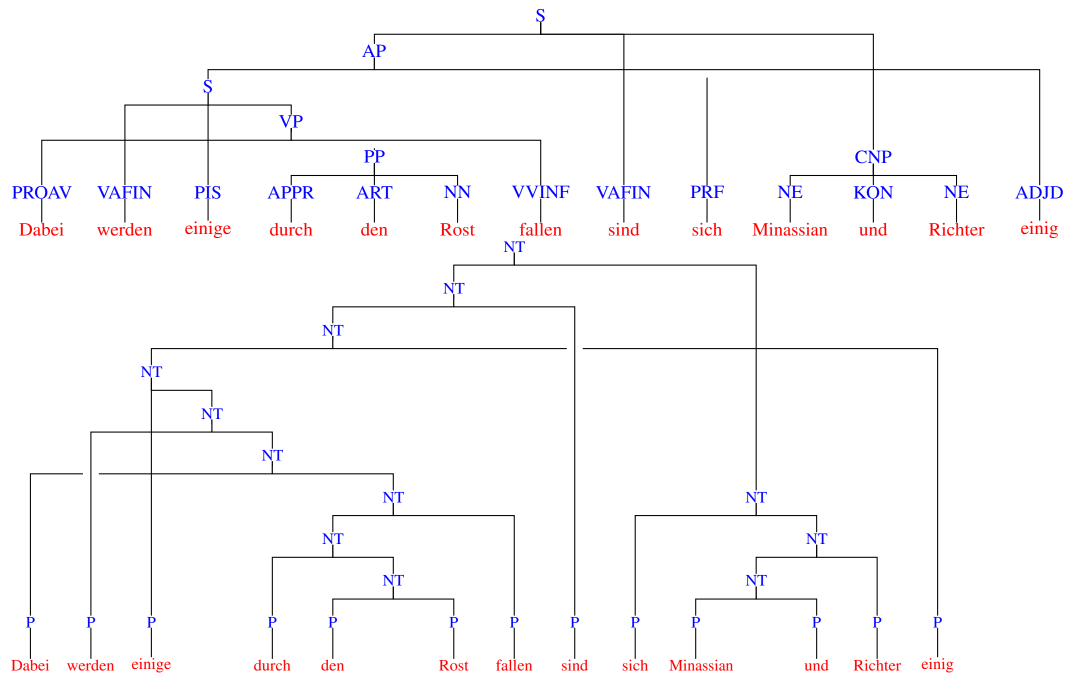
$r_3 = r_4$. Due to the high computational complexity, we used r_1 up to 400. It is important to note that we cannot use a ratio of $\frac{r_1}{r_3}$ or $\frac{r_2}{r_4}$ arbitrarily, such as 80:20 or 50:50. We observed much lower total F1 scores (much more discontinuous spans would be predicted) when using such ratios in our experiments. This is because $\frac{r_3}{r_1+r_3}$ can be regarded as the prior probability (when the network is randomly initialized) of having a discontinuous child for a fan-out-1 parent node. If the ratio of $\frac{r_3}{r_1+r_3}$ is too high, the model will predict many discontinuous spans from the beginning. Unsupervised learning will use the expected counts from the start for feedback self-supervised learning, resulting in the grammar learned at the end predicting many more discontinuous spans.

C Additional results

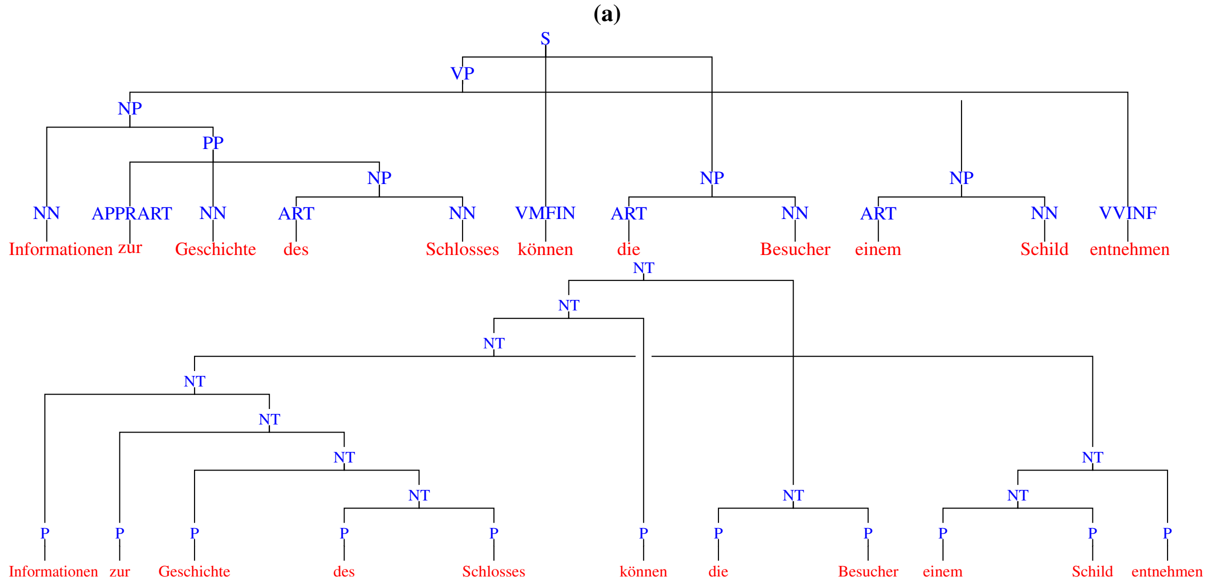
Table 7 shows the maximum performance across four seeds, while Table 8 gives the F1 broken down by sentence length on TIGER.

D Additional example trees

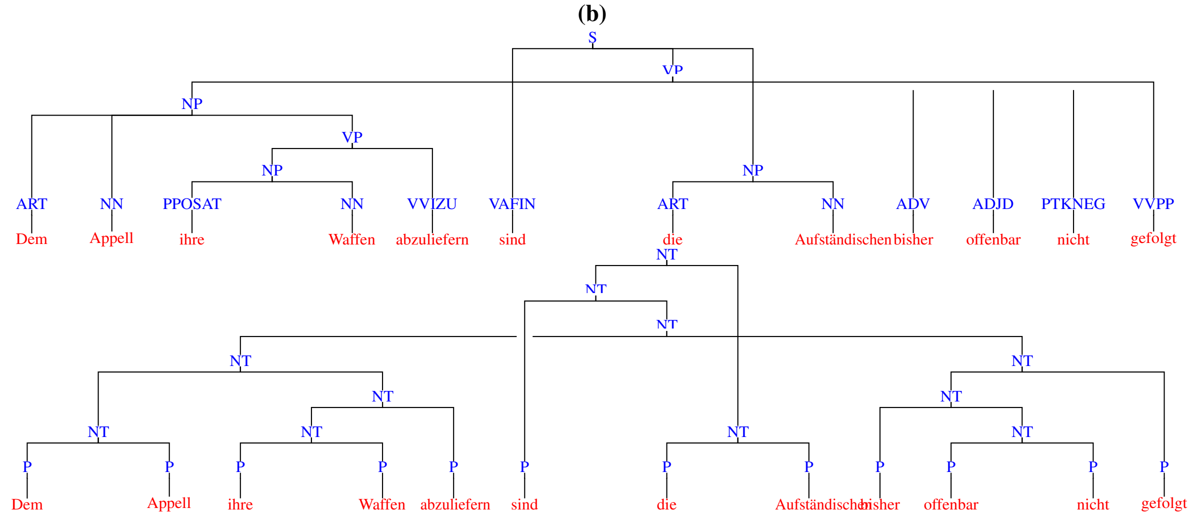
We show some additional trees on German in Fig. 4 and on Dutch in Fig. 5.



(a)

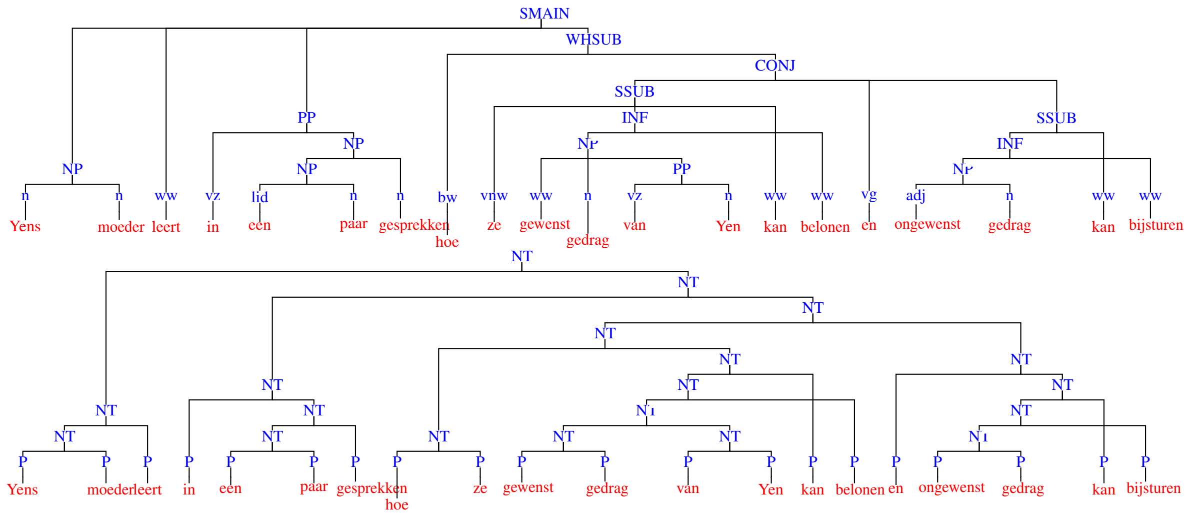


(b)

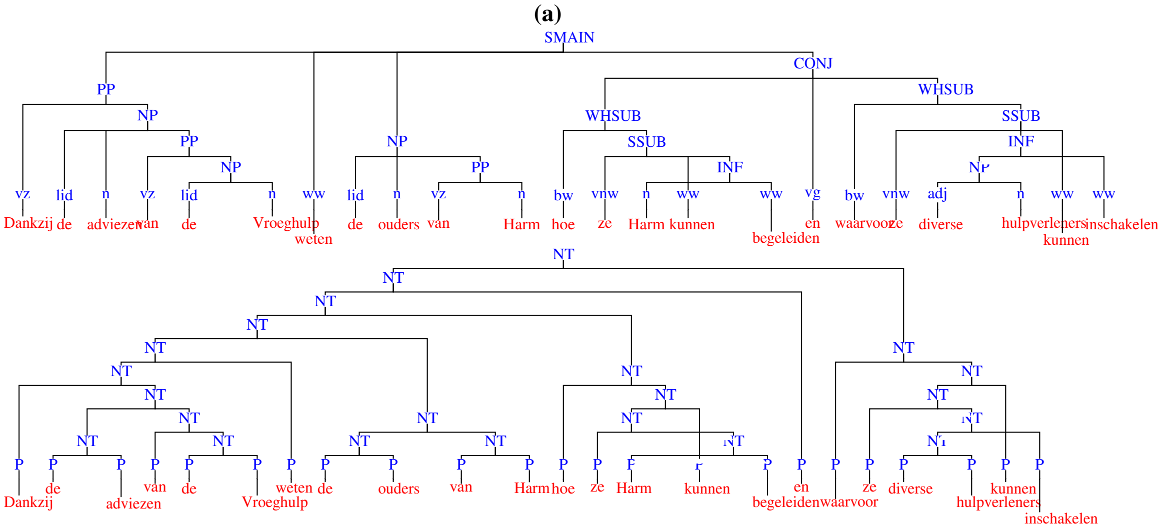


(c)

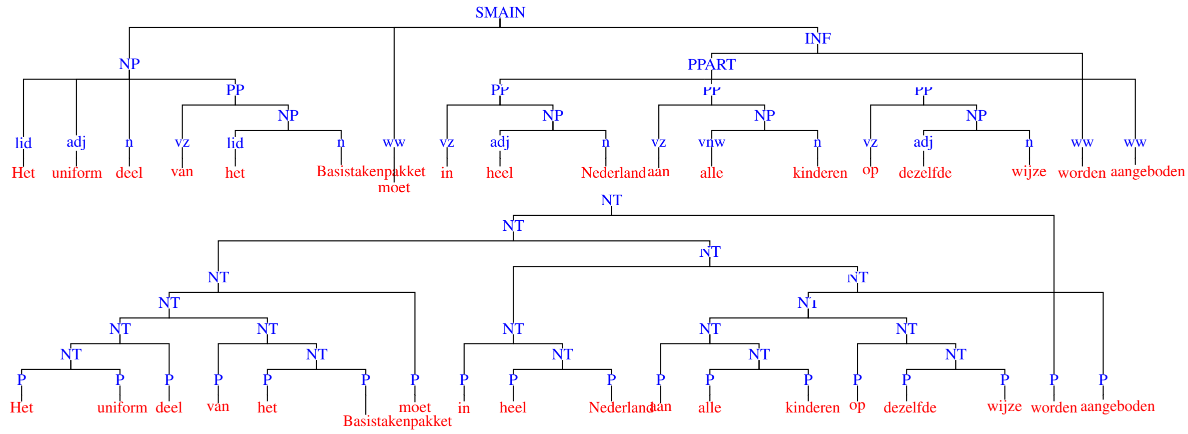
Figure 4: Examples of gold (top) and predicted (bottom) trees in Germain. NT and P denote predicted nonterminals and preterminals.



(a)



(b)



(c)

Figure 5: Examples of gold (top) and predicted (bottom) trees in Dutch. NT and P denote predicted nonterminals and preterminals.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
after conclusion selection, page 9

☒ A2. Did you discuss any potential risks of your work?
we do not see any ethical implications or risks of our work

☑ A3. Do the abstract and introduction summarize the paper's main claims?
abstract and section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

sectiion 3, Appendix B

☑ B1. Did you cite the creators of artifacts you used?
section 3, Appendix B

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Left blank.

☒ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☑ Did you run computational experiments?

section 3.1

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
we show the grammar size and embedding size in section 3

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
section 3, appendix b.3

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
section 3

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
section 3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.