言語モデリングに階層構造は必要か？ インクリメンタルな言語処理における記憶との関係について

石井太河　宮尾祐介
東京大学
{taigarana,yusuke}@is.s.u-tokyo.ac.jp

概要

自然言語は階層構造を持つとされるが、言語のモデル化において階層構造は本当に必要であるか、またどのような役割を持つかは自明ではない。本研究では、「階層構造はインクリメンタルな言語処理における記憶保持に寄与する」という心理言語学の仮説に着目し、形式言語を用いた検証を行う。具体的には、トークンと構造の両方を予測する統語的言語モデリングタスクにおいて、用いる構造の深さとモデルの記憶容量がトークン予測性能に与える影響を分析する。実験の結果、階層構造の利用により、テキストのみの場合と比較して予測性能の向上が確認され、モデルの記憶容量が小さいほど、より深い階層構造が必要になる傾向が示された。

1　はじめに

理論言語学、特に統語論において、「階層構造」は単語の組み合わせが句や文となる入れ子関係を表現する木構造であり、文法や言語現象のモデル化に重要な概念とされてきた [1]。しかし近年、大規模言語モデル（LLM） [2, 3] は、こうした階層構造を明示的に扱うことなく、極めて高い言語処理能力を達成している。この事実は、「そもそも言語のモデル化において、階層構造は本当に必要なのか？」という根本的な問いを生む。

仮に、言語のモデル化に階層構造が本質的に不要であるならば、テキストのみから構造を計算する教師なし構文解析タスク [4, 5] は、ある種の帰納バイアスなしには原理的に困難である可能性が示唆される。また、言語モデルの内部機序を明らかにするプロービング [6] 等において、正解構造として参照されるツリーバンク [7] の木構造そのものの妥当性が揺らぐ可能性もある。したがって、言語のモデル化における階層構造の必要性と役割を明らかにすることは、工学的な性能向上のみならず、言語の数理的性質を理解する上でも重要な課題である。

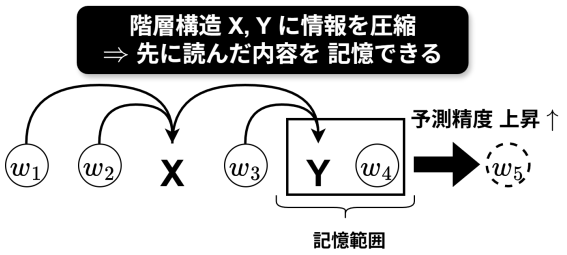

図 1　本研究の作業仮説

本研究では、形式言語を対象とし、インクリメンタルな次トークン予測という観点から、言語のモデル化に必要な構造を考察する。既存研究において、トークンと構造の両方を予測する統語的言語モデリングでは、ツリーバンクの木構造をモデル化することにより、トークン予測性能や統語的汎化能力が向上することが示されている [8, 9]。しかし、階層構造が「いつ」「どのような」役割を果たすことで言語のモデル化に寄与するのかは明らかになっていない。

この問題に対し、本研究では心理言語学における Now-or-Never bottleneck 仮説 [10] に着目する。Christiansen ら [10] は、人間が言語をインクリメンタルに処理する際、限られた記憶容量で情報を保持するために、入力を即座に情報圧縮し、その連鎖が階層構造となることを主張している（図 1）。本研究では、この考えを人間に限らず一般のモデルへと拡張し、「次トークン予測において、階層構造は必要な情報を記憶保持するために必要であり、かつ、具体的に必要な構造はモデルの記憶容量に依存する」という仮説を検証する。

この作業仮説に基づき、本研究ではモデルの記憶容量（コンテキストウィンドウサイズ）をパラメタとして「記憶を補完し次トークン予測を行うために必要な階層構造」を計算するアルゴリズムを提案する。そして、このアルゴリズムにより計算された深

さの異なる階層構造をモデルに与え、統語的言語モデリングにおける「モデルの記憶容量」と「必要な階層構造の深さ」の関係を分析する。実験の結果、階層構造を用いることで、テキストのみの場合と比較してトークン予測性能が向上することが確認された。さらに、次トークン予測に必要な構造はモデルの記憶容量によって異なり、記憶容量が小さいモデルほど、性能維持のためにより深い階層構造を必要とする傾向が示された。

2 関連研究

ここでは、「階層構造の必要性や役割」に関連する研究について述べる。まず、心理言語学の分野では、人間の言語処理はヒューリスティックであり、ツリーバンクの完全な木構造よりも浅い部分構造を用いている可能性が示唆されている [11, 12]。同様に、能地ら [13] は、ツリーバンクの木構造から抽出した様々な部分構造を教師データとして用いて統語的言語モデルを学習・評価した。その結果、長距離依存関係の解決をはじめとした統語的汎化能力の向上には、中間程度の深さの部分構造を用いることが最適であることが報告されている。また、Lin と Tegmark [14] は、階層構造の存在により、系列上の単語間相互情報量が距離に対して冪減衰することを理論的に示しており、階層構造と長距離依存の間に密接な関係があることを示唆している。

一方で、言語学的な正解構造（ツリーバンク構造）を仮定せず、下流タスクの性能向上を目的関数として、強化学習により階層構造を探索するアプローチもある [15, 16]。しかしながら、これらの先行研究では、インクリメンタルな系列処理において、記憶容量といったモデルの性質に対し、階層構造がどのように機能しているのかまでは十分に解明されておらず、Now-or-Never bottleneck 仮説 [10] の計算論的な検証には至っていない。

本研究は、形式言語を対象とすることで問題を単純化し、モデルの記憶容量に対して相対的に必要な構造を分析する。これにより、構成的なアプローチから階層構造と記憶の相互関係の解明に取り組む。

3 タスクの定式化

本研究では、仮説検証のタスクとして、トークンと構造の両方を予測する統語的言語モデリングを用いる。既存の定式化を抽象化することで、記憶の保持に必要な構造の計算を簡潔にすることを目指す。

従来の統語的言語モデリングは、スタック構造を仮定し、NT(X)・SHIFT(w)・REDUCE といったスタック操作の列を予測することで定式化される [8, 9]。[1] 例えば、予測されるアクション系列は [..., SHIFT(w_1), NT(X_1), NT(X_2), SHIFT(w_2), REDUCE, SHIFT(w_3), ...] のようになる。しかし、この定式化ではトークン予測（SHIFT）の間に行われる構造予測（NT・REDUCE）の回数が不定であるため、推論時には word synchronous beam search [17] のような複雑なヒューリスティクスが必要となり、記憶の保持に必要な構造の計算が困難となる。

これに対し本研究では、スタック構造を仮定せず、トークン間の構造予測が高々一回となるように抽象化する。具体的には、語彙集合 $\mathcal{V}$ と予測対象の構造集合 $\mathcal{X}$ に対し、$z \in \mathcal{V} \mid (\mathcal{V} \mid \mathcal{X}\mathcal{V})^*$ で定まるトークンと構造の両方を含む系列 z（トークン構造系列と呼ぶ）を予測するタスクとして定式化する。

この定式化により、入力トークン列 $(x_1, \ldots, x_n) \in \mathcal{V}^+$ に対するトークン構造系列 z の予測は、各タイムステップ $t = 1, \ldots, n$ において、トークン x_t の予測と構造 $y_t \in \mathcal{X} \cup \{\mathrm{NOP}\}$ の予測をインクリメンタルに繰り返すことで行える。ここで、NOP は「構造を予測しない」ことを表現するものである。系列 x の i 番目から j 番目までの要素からなる部分系列を x_i^j と表記するとき、各ステップ t における推論は、その時点までのトークン構造系列 z_1^{j-1} を用いて以下の手順で行われる：

1. トークン x_t の予測確率 $p(x_t \mid x_1^{t-1}) = M^k_{\mathrm{tok}}(z_{j-k}^{j-1})|_{x_t}$ を計算し、z に追加する $(z_j \leftarrow x_t)$
2. 最後のトークン以外の場合 $(t < n)$、構造 $y_t \leftarrow \arg\max M^k_{\mathrm{struct}}(z_{j+1-k}^{j})|_{y_t}$ を貪欲に推論する。もし $y_t \neq \mathrm{NOP}$ ならば、z に追加する $(z_{j+1} \leftarrow y_t)$

ここで、k はモデルのコンテキストウィンドウサイズであり、$M^k_{\mathrm{tok}}(\cdot)$ は与えられたコンテキストを元に計算した $\mathcal{V}$ 上の次トークンの確率分布である。同様に、$M^k_{\mathrm{struct}}(\cdot)$ は $\mathcal{X} \cup \{\mathrm{NOP}\}$ 上の確率分布である。以上より、与えられたトークン列 $(x_1, \ldots, x_n)$ の確率は、$\prod_{t=1}^{n} p(x_t \mid x_1^{t-1})$ として計算される。[2] [3]

1) NT(X) はスタック上に句の左端を開くアクション、SHIFT(w) はトークン w を予測しスタックにプッシュするアクション、REDUCE はスタック上の開いた句を閉じて一つの要素に合成するアクションである。

2) 厳密には BOS・EOS トークンも扱う必要があるが、表記の簡潔さのため省略する。

3) 統語的言語モデルは一般的にトークンと構造の同時確率を計算するが [8, 9]、ここでは PRPN [18] モデルと同様に構造

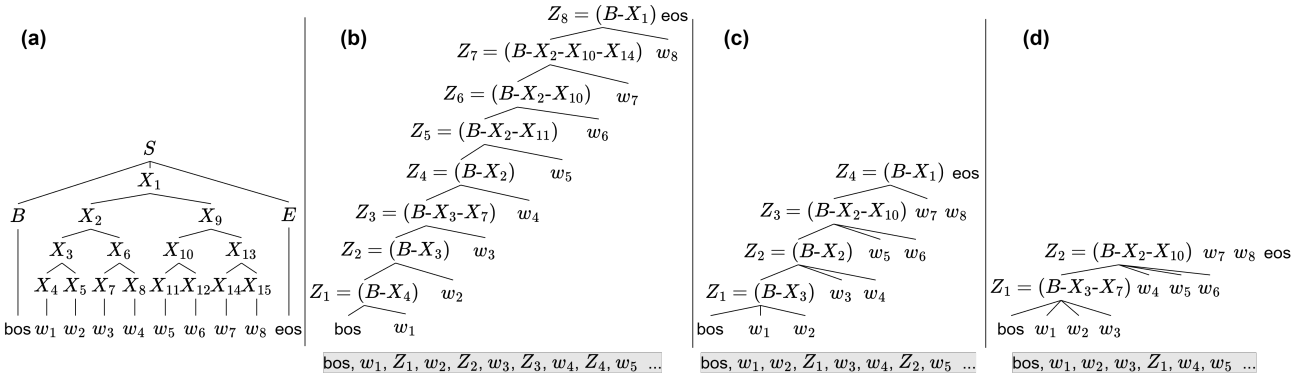

図 2　(a) 元の木構造; (b, c, d) mem_m（$m = 2, 3, 4$）をそれぞれ適用した結果のトークン構造系列とそれ対応する階層構造

4　記憶を保持するための階層構造

ここでは、前節で定式化した「トークン構造系列」を具体的に計算する方法について述べる。本研究では、曖昧性のない確率文脈自由文法 G が与えられていると仮定し、G から生成されたトークン列 x に対して、記憶の保持に必要な構造を導出する。この計算において、特に「文脈自由性」が重要となる。

まず、系列 x_1^{t-1} をボトムアップ構文解析した際に得られる、部分木の根ノードのタプルを $\mathrm{BUNodes}_G(x_1^{t-1})$ とする。[4] 確率文脈自由文法の性質上、G から生成されたトークン列 x の t 番目のトークン x_t を予測するために必要な情報は、高々 $\mathrm{BUNodes}_G(x_1^{t-1})$ に集約される。

例えば、図 2 (a) においてトークン w_3 を予測するには、それまでのトークン列 bos, w_1, w_2 に関する情報が保持されていればよい。ここで、bos は非終端記号 B から、(w_1, w_2) は X_3 からそれぞれ文脈自由に生成されるとすれば、保持すべき情報は「B および X_3 から生成された」という事実のみで十分である。これは、bos, w_1, w_2 をボトムアップ構文解析した際の部分木の根ノードのタプル (B, X_3) に相当する。このように、表層のトークン列ではなく非終端記号（抽象化された構造）を必要な情報として保持することは、Now-or-Never bottleneck 仮説 [10] における「情報圧縮」のプロセスとして解釈できる。

次に、トークン列 x に対して、モデルのコンテキストウィンドウサイズ m を考慮したトークン構造列 $\mathrm{mem}_m(x)$ を構成する。基本的なアイデアは、「必要な情報が圧縮された構造が、常にコンテキストウィンドウ内に収まるようにする」というものである。

	0	1
S	S → NP VP	S → VP NP
VP	VP → NP VP	VP → VP NP
	NP → VP NP	NP → NP VP
PP	PP → NP PP	PP → PP NP
	NP → PP NP	NP → NP PP

表 1　生成規則をコントロールするパラメタ

これに基づき、$\mathrm{mem}_m(x)$ を以下のように定める：

$$\mathrm{mem}_m(x) = x_1^m \cdot \mathrm{BUNodes}_G(x_1^m) \cdots \cdot x_{m\cdot i+1}^{m\cdot(i+1)-1} \cdot \mathrm{BUNodes}_G(x_1^{m\cdot(i+1)-1}) \cdots x_{|x|}$$

この構成では、最初を除き、$m-1$ 個のトークンごとに、その時点までの系列を要約した構造が予測される。図 2 (b-d) に、異なる m で mem_m を適用した結果を示す。情報圧縮の連鎖によって階層構造が形成されており、m が大きいほど、次トークン予測に用いられる階層構造は浅くなる傾向が確認できる。

5　実験設定

実験では、トークン構造系列を生成する際のコンテキストウィンドウサイズ m（次トークン予測に用いる階層構造の深さ）と、モデルのコンテキストウィンドウサイズ k（モデルの記憶容量）の二つの変数を操作し、トークン予測性能への影響を分析する。

データ　実験に用いるコーパス D は、曖昧性のない確率文脈自由文法 G からランダムサンプリングにより生成する。本研究では問題を単純化するため、導出木が完全二分木となり、かつ BOS・EOS トークンを除いた葉数が常に 32 となるような文法を用いる。生成規則の設計は White ら [19] にならい、表 1に示すパラメタ **S**・**VP**・**PP** を変更することで、パターンの異なる合計 8 つの文法を用いる。[5]

が貪欲に推論されるため、推論された構造は常に確率 1 で予測されるものとして考えられることに注意されたい。

4)　G は曖昧性がないため $\mathrm{BUNodes}_G$ の出力は一意に定まる。

5)　生成規則などの詳細については第 A 章に記す。

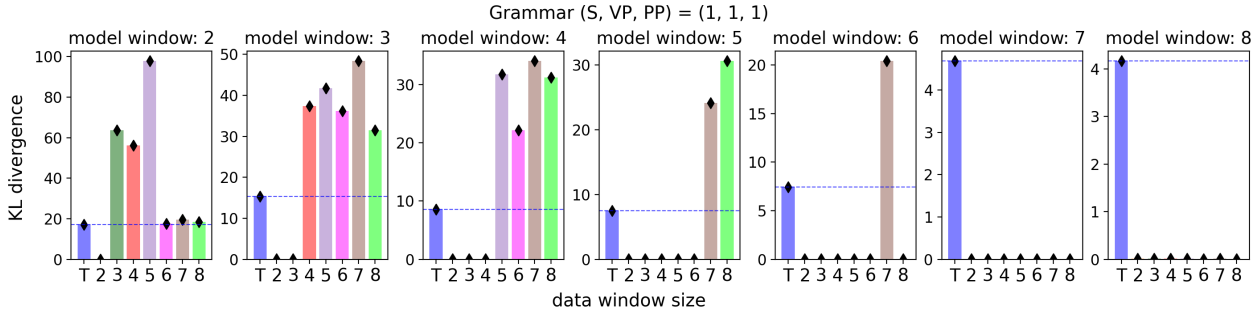

図 3　文法 (S, VP, PP) = (1, 1, 1) における各モデルコンテキストウィンドウサイズ k での実験結果。横軸はトークン構造系列を生成する際のコンテキストウィンドウサイズ m を表す。T はテキストのみで学習した結果を表す。

モデルと評価　帰納バイアスの影響を排除し、純粋にコンテキストウィンドウサイズの影響を評価するため、本実験ではデータ内の (context, target) ペアの出現頻度に基づく単純なカウントベースのモデルを用いる。[6] モデルのトークン予測性能の評価には、コーパス D を用いてモデル M^k と真の分布 G の間で KL ダイバージェンスを計算する。

6　実験結果・考察

実験の結果、文法間で全体の傾向に顕著な差異は見られなかった。ここでは代表として文法 (S, VP, PP) = (1, 1, 1) の結果について議論する。図 3は、各モデルのコンテキストウィンドウサイズ $k = 2, \ldots, 8$ でのトークン予測性能（KL ダイバージェンス）を示す。ここで、横軸はトークン構造系列を生成する際のコンテキストウィンドウサイズ m であり、T はテキストのみで学習した結果を表す。

6.1　確率文脈自由文法で Now-or-Never bottleneck 仮説は成り立つ

図 3では、少なくとも $m \leq k$ である場合において、階層構造を用いたモデルの KL ダイバージェンスは、テキストのみの場合よりも小さく、0 に近い値となっている。この結果は、例えば $k = 2$ のようにモデルの記憶容量（コンテキストウィンドウサイズ）が限られる場合でも、適切な階層構造を用いることで、真の分布をほぼ再現可能であることを示しており、確率文脈自由文法において Now-or-Never bottleneck 仮説が成り立つことを示唆する。

6.2　モデルの記憶容量が小さいほど、深い階層構造が必要になる

例えば、モデルのコンテキストウィンドウサイズが $k = 2$ の場合、性能向上が見られるのは $m = 2$ のときのみであるのに対し、$k = 4$ の場合は $m = 4$ でも性能向上が見られる。この結果から、m が小さいほど構造予測の頻度が増え、階層構造は深くなることを考慮すると（図 2）、モデルの記憶容量 k が小さいほどより深い階層構造が必要になる一方、k が十分に大きい場合は、深い階層構造は必要なく、浅い階層構造でも十分に言語をモデル化できると言える。以上より、モデルの記憶容量に依存して必要な階層構造が変化することが示唆される。

6.3　自然言語の統語構造は記憶の保持に寄与するか？

本実験の結果から、もし自然言語の統語構造が文脈自由であるならば、インクリメンタル処理において記憶の保持に寄与しうると考えられる。しかし、ツリーバンクの木構造が文脈自由ではない可能性が示唆されており [20]、実際に記憶の保持に寄与するかは明らかではない。一方で、十分大きな確率文脈自由文法であれば自然言語を近似できるため [21]、本研究の手法を用いて、言語学的な正解構造と文脈自由に近似した構造の間でトークン予測性能の差を分析することが可能となる。これにより、自然言語の統語構造が記憶の保持にどの程度寄与するかの定量的な分析が可能になると考えられる。

7　結論と今後の展望

本研究では、確率文脈自由文法を用い、次トークン予測において階層構造は記憶の保持に必要であり、かつ、モデルの記憶容量に依存して必要な階層構造の深さが変化することを示した。具体的には、統語的言語モデリングの抽象化を行い、記憶の保持に必要な階層構造の計算手法を提案した。なお、記憶の保持に必要な階層構造は一意ではない。そうした構造間の関係性の解明により、言語が持つ階層構造の数理的性質の理解に繋がることが期待される。

6)　推論時にはバックオフスムージングを用いる。

謝辞

本研究は、JST 次世代研究者挑戦的研究プログラム JPMJSP2108 および JSPS 科研費 JP24KJ0666 の支援を受けたものです。また、実験の実装について助言をいただいた上田亮氏に感謝いたします。

参考文献

[1] Noam Chomsky. **Syntactic Structures**. Mouton de Gruyter, 1957.

[2] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901, 2020.

[4] Lifeng Jin, Finale Doshi-Velez, Timothy Miller, William Schuler, and Lane Schwartz. Unsupervised grammar induction with depth-bounded PCFG. **Transactions of the Association for Computational Linguistics**, Vol. 6, pp. 211–224, 2018.

[5] Jun Li, Yifan Cao, Jiong Cai, Yong Jiang, and Kewei Tu. An empirical comparison of unsupervised constituency parsing methods. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 3278–3283, Online, July 2020. Association for Computational Linguistics.

[6] David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. Probing for constituency structure in neural language models. In **Findings of the Association for Computational Linguistics: EMNLP 2022**, pp. 6738–6757, Stroudsburg, PA, USA, December 2022. Association for Computational Linguistics.

[7] Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. **Computational Linguistics**, Vol. 19, No. 2, pp. 313–330, 1993.

[8] Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. Recurrent neural network grammars. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, **Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 199–209, San Diego, California, June 2016. Association for Computational Linguistics.

[9] Do Kook Choe and Eugene Charniak. Parsing as language modeling. In Jian Su, Kevin Duh, and Xavier Carreras, editors, **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing**, pp. 2331–2336, Austin, Texas, November 2016. Association for Computational Linguistics.

[10] Morten H Christiansen and Nick Chater. The now-or-never bottleneck: A fundamental constraint on language. **Behav. Brain Sci.**, Vol. 39, No. e62, p. e62, January 2016.

[11] Fernanda Ferreira, Karl G D Bailey, and Vittoria Ferraro. Good-enough representations in language comprehension. **Curr. Dir. Psychol. Sci.**, Vol. 11, No. 1, pp. 11–15, February 2002.

[12] Fernanda Ferreira and Nikole D Patson. The 'good enough' approach to language comprehension. **Lang. Linguist. Compass**, Vol. 1, No. 1-2, pp. 71–83, March 2007.

[13] Hiroshi Noji and Yohei Oseki. How much syntactic supervision is "good enough"? In Andreas Vlachos and Isabelle Augenstein, editors, **Findings of the Association for Computational Linguistics: EACL 2023**, pp. 2300–2305, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

[14] Henry W Lin and Max Tegmark. Critical behavior in physics and probabilistic formal languages. **Entropy**, Vol. 19, No. 7, p. 299, June 2017.

[15] Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. Learning to compose words into sentences with reinforcement learning. In **International Conference on Learning Representations**, February 2017.

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

[17] Mitchell Stern, Daniel Fried, and Dan Klein. Effective inference for generative neural parsing. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, **Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing**, pp. 1695–1700, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

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

[19] Jennifer C White and Ryan Cotterell. Examining the inductive bias of neural language models with artificial languages. 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)**, Stroudsburg, PA, USA, 2021. Association for Computational Linguistics.

[20] 中石海, 吉田遼, 梶川康平, 福島孝治, 大関洋平. 自然言語における冪則と統語構造の関係の再考. 言語処理学会第 31 回年次大会, 2025.

[21] Songlin Yang, Yanpeng Zhao, and Kewei Tu. 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**, pp. 1487–1498, Stroudsburg, PA, USA, 2021. Association for Computational Linguistics.

A　データの詳細

本研究では、導出木が完全二分木となるように文法を構成する。具体的には、導出木の最大の高さを H とし、表 1にある各生成規則を、各高さ $h = 1, \ldots, H$ において $A_h \to B_{h-1} C_{h-1}$ の形式を持つ生成規則として具体化して用いる。これにより、特定の高さには特定の非終端記号のみが出現するように制限を加えることができる。終端記号については、$\mathrm{NP}_1 \to \mathrm{n}, \mathrm{VP}_1 \to \mathrm{v}, \mathrm{PP}_1 \to \mathrm{p}$ のように、各非終端記号がそれぞれ 1 種類の単語を生成するように設定する。なお、各非終端記号を左辺とする生成規則には、常に一様な確率を割り当てる。最後に、BOS・EOS トークンを導入するため、新たな開始記号 S を定義し、$\mathrm{S} \to B\ \mathrm{S}_H\ E$ および $B \to \mathrm{bos}, E \to \mathrm{eos}$ という生成規則を追加する。

また、このように設定した各文脈自由文法 G において、30000 個の系列をランダムサンプルすることにより実験に用いるコーパス D を生成する。