Tree Planted Transformer: 統語的大規模言語モデルの構築に向けて

吉田遼　染谷大河　大関洋平
東京大学
{yoshiryo0617,taiga98-0809,oseki}@g.ecc.u-tokyo.ac.jp

概要

本研究では、小〜中規模ツリーバンクにより統語知識を導入したのち、その統語知識を足場かけとしてテキストコーパスで効率良く大規模化するという統語的大規模言語モデルのフレームワークを提案する。その先駆けとして、Transformer 言語モデルのアテンション重みがツリーバンク上の統語構造に基づく分布に近づくように学習を行う手法 (Tree Planting Training) を実装し、通常のテキストコーパスでの追加・継続学習が可能なアーキテクチャである Tree Planted Transformer (TPT) を構築する。評価の結果、TPT はトークン列の確率のみをモデル化しているにも関わらず統語的言語モデル相当の統語知識を獲得していることが確かめられ、統語的大規模言語モデルの基盤となりうることが示唆された。

1　はじめに

近年、Transformer [1] 言語モデルを基盤とした大規模言語モデル (Large Language Model, LLM) の進展が目覚ましい (e.g., [2])。LLM の成功からは、**言語モデルが広範な世界知識を獲得し様々な下流タスクを解くためには、大量のテキストコーパスにより大規模化することが必要不可欠**であることが示唆されている。しかしながら、その成功とは裏腹に、LLM の基盤となる Transformer 言語モデルはその**学習効率の低さ**がしばしば指摘される。例えば、GPT-3 [2] の学習には人間が 10 歳までに受ける言語刺激のおよそ 2000 倍のデータが用いられているなど [3]、高性能な LLM の構築には膨大な量のデータとそれを処理する計算リソースが必要になる。

一方で、理論言語学が仮定する自然言語の統語構造 [4] を言語モデルに明示的に扱わせることで、少量のデータで高い精度を達成しようとする統語的言語モデルのアプローチが存在する [5, 6, 7]。統語的言語モデルは数百倍規模のデータで学習した Transformer 言語モデルと同程度以上の統語知識を獲得できるとの報告もあり [6]、**言語モデルが高い学習効率を得るためには統語構造に関するバイアスが必要不可欠**であることが示唆されている。しかしながら、**統語的言語モデルの学習にはツリーバンクなどの統語解析コーパスが必要**であり、特に世界知識を必要とする下流タスクにおいては、数万倍以上のテキストコーパスで学習可能な LLM と同等の精度には到達し得ない。

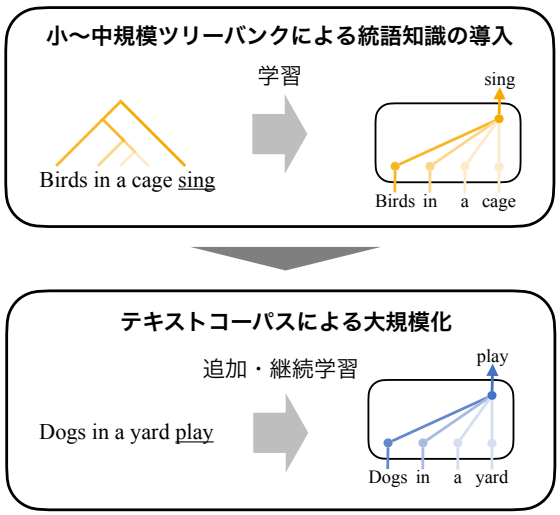

図 1: 統語的大規模言語モデルのフレームワーク。

そこで本研究では、**小〜中規模ツリーバンクでの学習により統語知識を導入したのち、その統語知識を足場かけとしてテキストコーパスで効率良く大規模化する**という**統語的大規模言語モデル**のフレームワークを提案する（図 1）。その先駆けとして、Transformer 言語モデルのアテンション重みがツリーバンク上の統語構造に基づく分布に近づくように学習を行う手法 (Tree Planting Training) を実装し、通常のテキストコーパスでの追加・継続学習が可能なアーキテクチャである Tree Planted Transformer (TPT) を構築する。評価の結果、TPT はトークン列の確率

のみをモデル化しているにも関わらず優れた統語知識を獲得していることが確かめられ、統語的大規模言語モデルの基盤となりうることが示唆された。

2 背景

2.1 Transformer 言語モデル

近年の LLM の基盤となるアーキテクチャが、Transformer [1] のデコーダのみを言語モデルとして利用する、Transformer 言語モデル (e.g., GPT-2; [8]) である。Transformer 言語モデルが LLM の基盤として用いられる理由の一つが、GPU 上での並列計算との親和性であり、その並列計算を可能にしているのが、Transformer 言語モデルのアテンションと呼ばれる機構である。アテンション機構では、文脈中の各トークンの重み付き和により、次トークン予測のための表現が得られ、その際の i 番目のトークンから j 番目のトークンに対する重み a_{ij} は以下のように計算される：

$$a_{ij} = \frac{\exp\left(\frac{Q_i K_j^T}{\sqrt{d_k}}\right)}{\sum_{k=1}^{i} \exp\left(\frac{Q_i K_k^T}{\sqrt{d_k}}\right)} \tag{1}$$

ここで、Q_i, K_j はそれぞれ i 番目のトークンのクエリベクトル、j 番目のトークンのキーベクトルを表し、d_k はキーベクトルの次元数を表す。

2.2 統語的言語モデル

Transformer 言語モデルをはじめとした、トークン列のみの確率をモデル化する通常の言語モデルとは異なり、統語的言語モデルはトークン列 X とその統語構造 Y の同時確率をモデル化する：

$$p(X, Y) = p(z_1, ..., z_n) = \prod_{i=1}^{n} p(z_t | z_{<t}) \tag{2}$$

ここで、z_t はトークン列とその統語構造を生成するアクションである。[1)]

近年、Transformer を基盤とした統語的言語モデルが複数提案され、データが制限された条件下においては、通常の Transformer 言語モデルよりも高い精度を達成できることが報告されている [5, 6, 7]。しかしながら、統語的言語モデルはトークン列と統語構造の同時確率をモデル化するため、**ツリーバンク以外での学習を行うことができない**。また、同時確率をモデル化することは、**推論時には統語構造に関してビームサーチを行う必要や外部パーサーで生成した統語構造を用いる必要がある**ことを意味し、この観点でも実用上での大きな妨げとなっている。

3 提案手法

本研究では、Transformer 言語モデルの **アテンション重みがツリーバンク上の統語構造に基づく分布に近づくように学習を行う手法** (Tree Planting Training) を実装し、通常のテキストコーパスでの追加・継続学習が可能なアーキテクチャである **Tree Planted Transformer (TPT)** を構築する（図 2）。

3.1 統語構造に基づく教示分布の算出

ツリーバンク上の統語構造に基づく教示分布の作成にあたり、統語構造上の距離に着目する。ここで、統語構造上での距離とは、各単語間の統語構造上でのパスにおけるエッジ数を意味する。本手法は句構造・依存構造の両者に対して適用可能であるが、依存構造においては、依存関係の方向は考慮せずに距離の測定を行う。

統語構造上での距離から教示分布 T への変換は以下の式による：

$$t_{ij} = \begin{cases} \frac{\exp(-d(w_{i+1}, w_j))}{\sum_{k=1}^{i} \exp(-d(w_{i+1}, w_k))} & (i \geq j) \\ 0 & (i < j) \end{cases} \tag{3}$$

ここで、t_{ij} は i 番目の単語 w_i から j 番目の単語 w_j に対する重みへの教示、$d(w_l, w_m)$ は l 番目の単語 w_l と m 番目の単語 w_m の間の統語構造上での距離を表す。すなわち、この教示は、**各単語のアテンション重みが、予測対象の単語との統語構造上の距離に関して、指数関数的に単調減少することを期待**して設計されている。[2)]

3.2 アテンション重みへの教示

3.1 節の教示分布はツリーバンク上の統語構造に基づき単語単位で作成されるが、一般的に、LLM の基盤となる Transformer 言語モデルのトークン単位はサブワードである。そこで、サブワード単位のアテンション重みを、以下のように単語単位のアテンション重み M に変換する：

$$m_{ij} = \frac{p_{ij}}{\sum_{k=1}^{i} p_{ij}} \tag{4}$$

1) 例えば、top-down かつ left-to-right な統語的言語モデルの場合、トークンの生成 または 構成素を開く または 構成素を閉じる のいずれかである。

2) Lin and Tegmark [9] が、単語間の相互情報量は統語構造上の距離に関して指数的関数的に減衰する可能性を報告しているため、単調減少関数として指数関数を採用した。

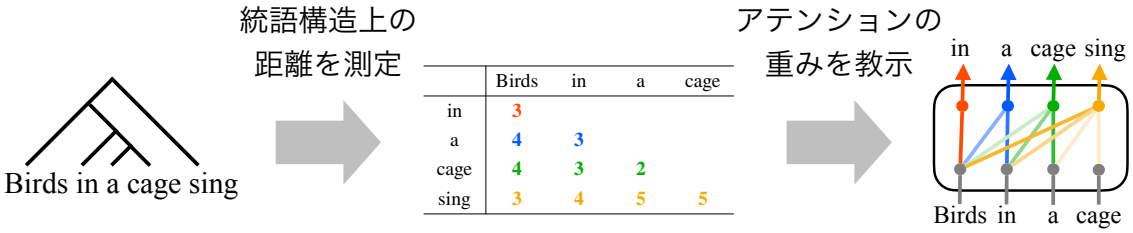

図 2: 提案手法である Tree Planting Training 及びその結果得られる Tree Planted Transformer (TPT) の概略。

ここで、m_{ij} は i 番目の単語 w_i から j 番目の単語 w_j に対する重みを表し、

$$p_{ij} = \sum_{k=\mathrm{START}(w_{i+1})}^{\mathrm{END}(w_{i+1})} \sum_{l=\mathrm{START}(w_j)}^{\mathrm{END}(w_j)} a_{kl} \tag{5}$$

である。ただし、a_{kl} は k 番目のサブワードから l 番目のサブワードに対する重みを表し、[3] $\mathrm{START}(w_n)$、$\mathrm{END}(w_n)$ はそれぞれ n 番目の単語 w_n 内部のサブワードのインデックスの開始位置、終了位置を表す。つまり、p_{ij} は w_{i+1} 内部のサブワードが予測される際の、w_j 内部のサブワードのアテンション重みの和である。単語単位のアテンション重み M を教示分布 T に近づけることを目的として、Ma et al. [10] を踏襲し、カルバック・ライブラー距離 (D_{KL}) による損失を導入する：

$$\mathcal{L}_{\mathrm{ATTN}} = \frac{\sum_{i=1}^{n-1} D_{\mathrm{KL}}(\boldsymbol{t}_i || \boldsymbol{m}_i)}{n-1} \tag{6}$$

ここで、$\boldsymbol{t}_i$、$\boldsymbol{m}_i$ はそれぞれ、T, M の i 行目のベクトル、n は単語列 $\boldsymbol{w}$ の長さを表す。つまり、$\mathcal{L}_{\mathrm{ATTN}}$ は $\boldsymbol{w}$ の先頭を除く各単語を予測する際の分布についての損失を平均したものである。学習時には、次単語予測の損失 $\mathcal{L}_{\mathrm{NWP}}$ と $\mathcal{L}_{\mathrm{ATTN}}$ との重み付き和を最小化する：

$$\mathcal{L} = \mathcal{L}_{\mathrm{NWP}} + \lambda \mathcal{L}_{\mathrm{ATTN}} \tag{7}$$

4 実験・結果

提案手法である Tree Planting Training により TPT に優れた統語知識が導入されることを確認するため、ツリーバンクによる学習および統語知識の観点での評価を行う。

4.1 実験設定

学習データ BLLIP コーパス [11]（XL、約 42M トークン）を用いる。統語構造としては、(i) 句構造、(ii) 二分木化した句構造、(iii) 依存構造、の 3 つを用いる。(i) の句構造は、Hu et al. [11] により Berkeley Neural Parser [12] で再解析されたもの、(ii) の二分木は、句構造を `nltk` [13] ライブラリにより二分木化したもの、(iii) の依存構造は、`spacy` [14] ライブラリの `en_core_web_sm` モデルを用いて解析したものを用いる。

トークナイザ・モデル トークナイザ・モデルは、それぞれ GPT-2 [8]（small、約 124M パラメータ）と同様のものを用いる。[4] モデルのパラメータは全てランダムに初期化する。Ma et al. [10] を踏襲し、$\mathcal{L}_{\mathrm{ATTN}}$ の算出には上部 3 層を用いる。[5] また、ベースラインとして $\mathcal{L}_{\mathrm{ATTN}}$ の重み λ が 0 のモデル（Transformer 言語モデルと等価）の学習も行う。その他のハイパーパラメータは付録 A に記載する。

評価 統語知識評価のベンチマークである SyntaxGym [11, 16] を用いる。SyntaxGym は、様々な文法現象に関する統語知識の評価を目的として設計されているが、例えば `Agreement` の統語的サーキットでは主に主語と動詞を一致させる能力が問われ、以下の (1) のような文が与えられた時に、正しい動詞（a の下線部）に誤った動詞（b の下線部）よりも高い確率を推定した時に正解となる：

(1) a. The author next to the senators is good.
b. *The author next to the senators are good.

言語モデルのより一般的な評価指標である perplexity（単語単位）についてもあわせて評価する。

また、全てのモデルについて、異なる 2 つのランダムシードで学習を行い、その平均値を報告する。

3) a_{kl} はマルチヘッドアテンションの全ヘッドの平均として計算する。層についての扱いは 4.1 節で後述する。

4) `transformers` [15] ライブラリの `GPT2Tokenizer`、`GPT2LMHeadModel` を用いる。

5) ただし、事前探索の結果を踏まえ、句構造と二分木は層平均のアテンションに対する損失、依存構造は各層のアテンションに対する損失の合計、を用いる。同じく事前探索の結果を踏まえ、$\mathcal{L}_{\mathrm{ATTN}}$ の重み λ についても、句構造と二分木は 0.1、依存構造は 0.05 を用いる。

4.2 結果

表 1 に、SyntaxGym 全体の正答率 (SG) と perplexity (PPL) を示した。先行研究で提案されている統語的言語モデルである PLM、PLM-mask [5] 及び Transformer Grammar (TG) [6] の結果もあわせて報告する。これらは提案手法と同じく BLLIP コーパスで学習されたものである。また、Transformer 言語モデルである GPT-2 [8]、Gopher [17]、Chinchilla [18] の結果も報告する。これらは提案手法より大規模なテキストコーパスで学習されたもの（¶ で表す）であり、参考値として先行研究 [6] より引用した。TPT（ゼロ）は、$\mathcal{L}_{\text{ATTN}}$ の重み λ がゼロのモデル（Transformer 言語モデルと等価）を表す。PLM・PLM-mask（無印）は、TPT と推論時のコストを揃えるため、統語構造に関するビームサーチを行わずに測定した精度を表す。[6] また、PLM・PLM-mask† 及び TG‡ は、提案手法よりハイコストな条件[7] で評価された結果であり、参考値として先行研究より引用した。

SyntaxGym 全体の正答率について、重要な結果を 3 つ観察することができる。まず、TPT（句構造、依存構造）が、TPT（ゼロ）を上回っていることから、提案手法の有効性が読み取れる。次に、TPT（二分木）は TPT（ゼロ）と同程度の正答率であることから、二分木化により統語構造が深くなりすぎると、適切なバイアスにならないことが示唆される (cf. [20])。最後に、TPT（句構造、依存構造）は、推論時のコストが等しい統語的言語モデル（PLM・PLM-mask（無印））を大きく上回り、よりハイコストな条件で評価された統語的言語モデルのうちの一部（PLM・PLM-mask†）とも同等の精度を達成していることから、**TPT（句構造、依存構造）はトークン列の確率のみをモデル化しているにも関わらず統語的言語モデル相当の統語知識を獲得している**ことが分かる。SyntaxGym の各統語的サーキットにおける正答率は付録 B に示した。

また、perplexity については、句構造解析器と依存構造解析器で単語分割単位が異なるため直接比較不可な値を含むものの、[8] 先行研究 [6] で報告されている、一部の統語的言語モデルに見られる perplexity の顕著な悪化が、提案手法では見られないことが示唆されている（cf. TPT（ゼロ）・TPT（依存構造））。

	SG (↑)	PPL (↓)
TPT（ゼロ）	71.7 ± 0.3	47.5 ± 0.1♠
TPT（句構造）	74.1 ± 2.1	41.6 ± 0.0♡
TPT（二分木）	71.7 ± 1.7	**41.5** ± **0.1**♡
TPT（依存構造）	**74.7** ± **1.6**	47.7 ± 0.0♠
PLM [5]	42.2 ± 1.2	-
PLM-mask [5]	42.5 ± 1.5	-
PLM [5]†	73.2 ± 0.6	49.3 ± 0.3♡
PLM-mask [5]†	74.6 ± 1.0	49.1 ± 0.3♡
TG [6]‡	82.5 ± 1.6	30.3 ± 0.5♡
GPT-2 [8] ¶	78.4	-
Gopher [17] ¶	79.5	-
Chinchilla [18] ¶	79.7	-

表 1: SyntaxGym 全体の正答率 (SG) と perplexity (PPL)。TPT（ゼロ）は、$\mathcal{L}_{\text{ATTN}}$ の重み λ がゼロのモデルを表す。† 及び ‡ は、提案手法よりハイコストな条件で評価された統語的言語モデルの参考値である。¶ は、提案手法より大規模なテキストコーパスで学習された Transformer 言語モデルの参考値である。単語分割単位が異なるため、perplexity は ♠ 内のみ、♡ 内のみでの比較が可能である。

5 おわりに

本研究では、小〜中規模ツリーバンクにより統語知識を導入したのち、その統語知識を足場かけとしてテキストコーパスで効率良く大規模化するという統語的大規模言語モデルのフレームワークを提案した。その先駆けとして、Transformer 言語モデルのアテンション重みがツリーバンク上の統語構造に基づく分布に近づくように学習を行う手法 (Tree Planting Training) を実装し、通常のテキストコーパスでの追加・継続学習が可能なアーキテクチャである Tree Planted Transformer (TPT) を構築した。評価の結果、TPT はトークン列の確率のみをモデル化しているにも関わらず統語的言語モデル相当の統語知識を獲得していることが確かめられ、統語的大規模言語モデルの基盤となりうることが示唆された。今後は、文章単位への拡張、及び、導入された統語知識を消失させることなく、むしろ統語知識を足場かけとして TPT を効率よくテキストコーパスで大規模化する手法の開発に取り組む。

6) TG については、学習済みのパラメータが公開されていないため測定していない。
7) それぞれ、統語構造に関してビームサーチ [19] を行う設定（アクションビーム幅 100、単語ビーム幅 10、fast track 幅 5）、外部パーサーで生成した統語構造を用いる設定。
8) TPT（ゼロ）は、TPT（依存構造）の λ を 0 にしたものである。

謝辞

本研究は、JST さきがけ JPMJPR21C2 の支援を受けたものです。

参考文献

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In **Advances in Neural Information Processing Systems**, Vol. 30. Curran Associates, Inc., 2017.

[2] 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. In **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020.

[3] Alex Warstadt and Samuel Bowman. What Artificial Neural Networks Can Tell Us about Human Language Acquisition. In **Algebraic Structures in Natural Language**. CRC Press, 2022.

[4] Noam Chomsky. **Syntactic Structures**. Mouton, The Hague, 1957.

[5] Peng Qian, Tahira Naseem, Roger Levy, and Ramón Fernandez Astudillo. Structural Guidance for Transformer Language Models. 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)**, pp. 3735–3745, Online, August 2021. Association for Computational Linguistics.

[6] Laurent Sartran, Samuel Barrett, Adhiguna Kuncoro, Miloš Stanojević, Phil Blunsom, and Chris Dyer. Transformer Grammars: Augmenting Transformer Language Models with Syntactic Inductive Biases at Scale, March 2022.

[7] Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher Manning. Pushdown Layers: Encoding Recursive Structure in Transformer Language Models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing**, pp. 3233–3247, Singapore, February 2023. Association for Computational Linguistics.

[8] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. p. 24, 2019.

[9] Henry W. Lin and Max Tegmark. Critical Behavior in Physics and Probabilistic Formal Languages. **Entropy**, Vol. 19, No. 7, p. 299, July 2017.

[10] Youmi Ma, An Wang, and Naoaki Okazaki. DREEAM: Guiding Attention with Evidence for Improving Document-Level Relation Extraction. In **Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics**, pp. 1971–1983, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics.

[11] Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger Levy. A Systematic Assessment of Syntactic Generalization in Neural Language Models. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 1725–1744, Online, July 2020. Association for Computational Linguistics.

[12] Nikita Kitaev and Dan Klein. Constituency Parsing with a Self-Attentive Encoder. In **Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 2676–2686, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[13] Steven Bird, Ewan Klein, and Edward Loper. **Natural language processing with Python: analyzing text with the natural language toolkit**. ” O’Reilly Media, Inc.”, 2009.

[14] Ines Montani, Matthew Honnibal, Matthew Honnibal, Adriane Boyd, Sofie Van Landeghem, and Henning Peters. Explosion/spaCy: V3.7.2: Fixes for APIs and requirements. Zenodo, October 2023.

[15] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-Art Natural Language Processing. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, pp. 38–45, Online, October 2020. Association for Computational Linguistics.

[16] Jon Gauthier, Jennifer Hu, Ethan Wilcox, Peng Qian, and Roger Levy. SyntaxGym: An Online Platform for Targeted Evaluation of Language Models. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations**, pp. 70–76, Online, July 2020. Association for Computational Linguistics.

[17] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d’Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling Language Models: Methods, Analysis & Insights from Training Gopher, January 2022.

[18] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training Compute-Optimal Large Language Models, March 2022.

[19] Mitchell Stern, Daniel Fried, and Dan Klein. Effective Inference for Generative Neural Parsing. In **Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing**, pp. 1695–1700, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[20] 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.

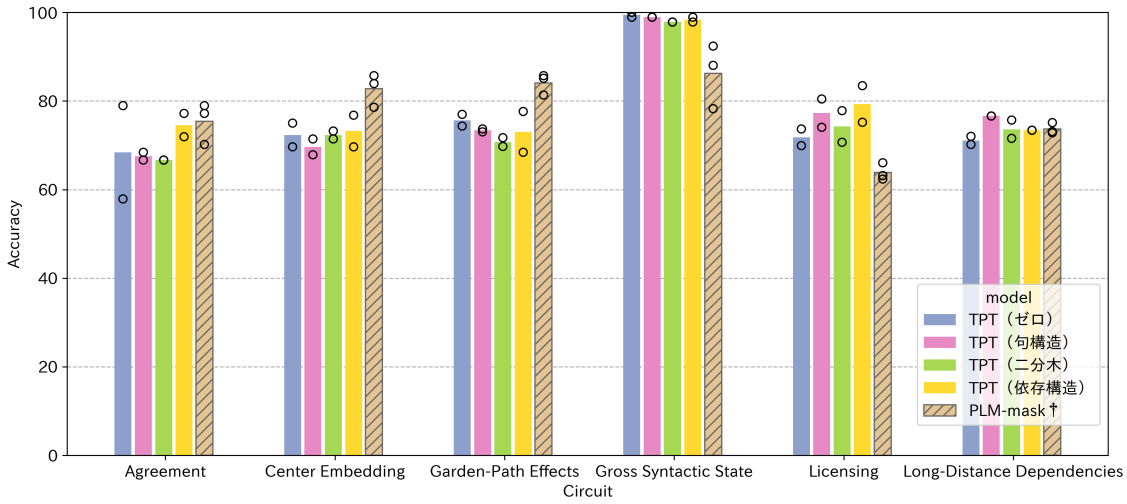

図 3: SyntaxGym の各統語的サーキットにおける正答率。

A　ハイパーパラメータ

TPT 学習時のハイパーパラメータを表 2 に示す。

最適化手法	AdamW
学習率	5e-5
エポック数	10
ドロップアウト率	0.1
バッチサイズ	256

表 2: TPT の全モデルで共通のハイパーパラメータ。

B　SyntaxGym の各統語的サーキットにおける正答率

提案手法の、SyntaxGym の各統語的サーキットにおける正答率を、図 3 に示す。SyntaxGym 全体で TPT（句構造、依存構造）と同等の正答率である PLM-mask† の結果もあわせて報告する。