極小主義に動機づけられた統語的教示に基づく言語モデル

磯野真之介 * 梶川康平 * 吉田遼 * 大関洋平
東京大学
{isono-shinnosuke, kohei-kajikawa, yoshiryo0617, oseki}@g.ecc.u-tokyo.ac.jp

概要

近年、階層的な統語構造を明示的に扱う言語モデルである再帰的ニューラルネットワーク文法（RNNG）が、高い文法汎化能力を持ちうることが示されている。本研究は、RNNG をベースに、有力な言語理論である極小主義の見地からより妥当な統語構造を用いた新しい言語モデル（**極小主義 (Minimalism) に動機づけられた RNNG, M-RNNG**）を提案する。M-RNNG では、音形のない痕跡や機能範疇が加わる一方、非終端記号はなく、また木は全て二分木である。SyntaxGym による評価では、一部の文法タスクで M-RNNG の精度が RNNG を上回り、音形のない要素と極小主義的な木構造がそれぞれ精度に貢献し得ることが示唆された。

1 はじめに

理論言語学では、人間の文法知識は離散的で再帰的な階層構造によって特徴づけられると主張されてきた [1, 2]。近年主流の大規模な言語モデルにはそのような階層的な構造を明示的に扱わないものも多い（e.g., [3]）が、階層構造を明示的に扱う（統語的教示がある）ことの重要性は今日に至るまで盛んに議論されている [4, 5, 6, 7]。例えば、それら統語的教示のある言語モデルの代表の一つである再帰的ニューラルネットワーク文法 (Recurrent Neural Network Grammar, RNNG) [8] は統語的教示のない純粋な再帰的ニューラルネットワーク (Recurrent Neural Network, RNN) [9] に比べて高い文法汎化能力 [10, 11, 12] や心理言語学的評価 [13, 14] を達成できることが示されており、「人間らしい」言語モデルとして注目されている [15]。

しかし、RNNG をはじめとする統語的言語モデルが扱う統語構造と、人間の言語知識についての有力な理論である主流生成文法、特に極小主義 [2, 16] で想定されている統語構造との間には開きがある。主流生成文法では、移動の痕跡や機能範疇として音形を持たない要素が存在し、人間の言語知識の説明に重要であることが主張されている。しかし、こうした音形のない要素は、主流なツリーバンク [17, 18] のアノテーションに含まれているにも関わらず、統語的言語モデルに与えられる教示では一般に取り去られてしまっている [19]。そこで本研究では、RNNG をベースに、主流生成文法・極小主義の見地からより妥当な統語構造を用いた新しい言語モデル（**極小主義 (Minimalism) に動機づけられた RNNG, M-RNNG**）を提案する。M-RNNG は、音形のない要素の生成 `NULL(*x)` をアクションに含むことで、解析対象の文字列に含まれていない要素を推測できる。これを利用し、主流生成文法で広く受け入れられているいくつかの音形のない要素が、文法汎化能力に貢献するかを検証する。

また極小主義では、階層構造を生成する操作である併合 (Merge) は、常に 2 つの要素をとり、そのラベルはそれらの要素を元に決まるとされている [2]。これらを踏まえ、M-RNNG が扱う木は全て二分木とし、明示的な非終端記号を取り去った。これにより、RNNG などの既存の統語的言語モデルにおける非終端接点を開くアクション `NT(X)` は不要となり、併合に相当する `REDUCE` アクションが常にスタックの上位 2 個の要素を参照してそれらを合成することになる。

このように、M-RNNG は、理論言語学で有力視されている構造を統語的教示として用いた言語モデルであり、本研究は人間らしい言語モデルの開発へ向けて、人間の文法知識を探求してきた理論言語学の知見が生かせるかを検証するものである。

2 提案手法

2.1 構文木の変換

主流生成文法、特に極小主義の見地からある程度妥当な構文木は、Penn Treebank (PTB) [17] 形式の構

* 共同第一著者

PTB形式（BLLIPより）

```
(S
 (NP-SBJ#1001
  (DT The)
  (NNP Nuclear)
  (NNP Regulatory)
  (NNP Commission))
 (VP
  (VBD gave)
  (NP
   (PRP$#1001 its) (NN staff))
  (NP
   (DT the)
   (NN authority)
   (S
    (NP-SBJ (-NONE- *))
       (VP
        (TO to)
        (VP (VB allow)
         (S ...)))))))
    (. .))
```

変換後

```
(
 (The
  (Nuclear
   (Regulatory Commission)))
 (*tns_pst
  (gave
   (
    (its staff)
    (the
     (authority
      (*t
       (to
        (allow
         (...)))))))))))
```

図 1　実際の木変換の例。三点リーダ部分は省略。

文木を一定のアルゴリズムに基づいて変換することで得られる。本研究で行った変換の例を図 1 に示す。変換は音形のない要素に関するものと、二分木化に関するものに大別される。音形のない要素のうち、PTB のアノテーションに存在するものについては、表記をアステリスク*から始まるように統一し、移動の痕跡*t、補文標識*c、削除の痕跡*e に整理した。句読点および単位記号の解釈位置（*U*）は削除した。さらに、時制の機能範疇を挿入した。具体的には、S に直接支配され、一般動詞の活用形を主要部とする VP があれば、それに被せる形で時制を示す要素（動詞の活用形に応じて、過去形*tns_pst、現在形 3 人称単数*tns_pres_3s、それ以外の現在形*tns_pres）を挿入した。時制の機能範疇を挿入したのは、その存在が主流生成文法において広く受け入れられており、かつ PTB のアノテーションを元に挿入することが容易だからである。

二分木化に際しては、PTB で三分木以上となっている木を、その品詞タグから判断できる限り言語学的に妥当になるように二分木に自動変換した。たとえば、図 1 の *the authority to ...* のように決定詞と名詞、それに後続する句を持つ三分木は、句を名詞の姉妹要素として付け替えることで二分木とした。また、*give* のような三項動詞については、動詞に後続する 2 つの項を 1 つの構成素にまとめた [20, 21]。

こうした変換を終えたあと、残った三分木以上の木を右枝分かれの二分木に変換し、枝分かれのない木を取り除き、非終端記号を取り去った。

2.2　極小主義に動機づけられた RNNG

提案モデルである極小主義 (Minimalism) に動機づけられた RNNG (M-RNNG) のアーキテクチャを図 2 に示す。スタックのみの RNNG [22, 10] をベースとしつつ、2.1 項の変換アルゴリズムにより得られた、主流生成文法・極小主義の見地からより妥当な統語構造を教示とすることが可能になっている。M-RNNG は、スタックデータ構造を用い、以下の 3 つのアクションによってその状態を更新していく：

- GEN(x)：単語 x を生成する。生成単語 x を表すベクトル e_x をスタックの先頭に追加する。
- NULL(*x)：音形のない要素*x を生成する。生成要素*x を表すベクトル e_{*x} をスタックの先頭に追加する。
- REDUCE：併合に相当する。スタックの上位 2 つの要素を双方向 LSTM [23] を用いて合成する。

非終端記号を取り去った二分木を扱うため、RNNG などの既存の統語的言語モデルにおける非終端記号を開くアクション NT(X) は不要となり、また、REDUCE アクションは常にスタックの上位 2 個の要素のみを参照すればよいことになる（cf. [24]）。各アクションの後には、スタックの状態がスタック LSTM [25] により単方向にエンコードされ、それに基づき次のアクションの確率分布が算出される。各アクションの後、スタック内に単一のベクトルが残された際のみ、生成の終了を表す STOP アクション [10] に対しても確率が付与できるようになる。

3　実験設定

3.1　音形のない要素の有無

M-RNNG の学習データには、PTB 形式のツリーバンクの一つである Brown Laboratory for Linguistic Information Processing 1987-89 Corpus Release 1 (BLLIP、LG、約 1.8M 文、約 42M トークン) [18] の各文に 2.1 項の変換アルゴリズムを適用したものを用いた。さらに本研究では、音形のない要素がモデルの文法汎化能力に貢献するかを検証するため、そうした要素を以下の 2 つのグループに分け、それぞれの有無だけが異なる 4 つの実験用のデータを用意した。

- 痕跡（移動の痕跡*t および削除の痕跡*e）
- 機能範疇（時制*tns_pst、*tns_pres、*tns_pres_3s および補文標識*c）

各実験用データを用い、入力・隠れ層の次元数 256 の 2 層 M-RNNG を学習した。また、ベースラインとして、2.1 項の変換アルゴリズムを適用しないかつ従来通り音形のない要素を取り去った BLLIP で

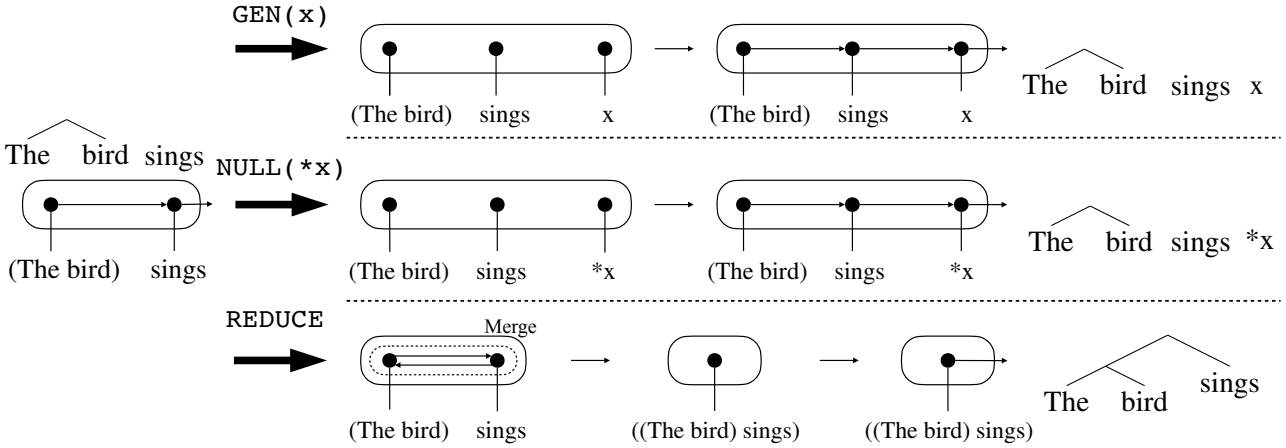

図 2 極小主義 (Minimalism) に動機づけられた RNNG (M-RNNG) のアーキテクチャ。GEN(x)、NULL(*x) アクションにより、それぞれ、単語、音形のない要素、を生成する。非終端記号を取り去った二分木を扱うため、非終端記号を開くアクション NT(X) は不要となり、併合に相当する REDUCE アクションは常にスタックの上位 2 個の要素を合成する。

学習した、入力・隠れ層の次元数 256 の 2 層 RNNG を学習した。M-RNNG と RNNG は共に並列化した実装 [4] を用いた。[1] 各 M-RNNG 及び RNNG について、ランダムシードの異なる 3 つのモデルを学習した。学習の詳細は付録 A に示した。

3.2 SyntaxGym

本研究では、音形のない要素の有無が異なる 4 つのデータで学習された M-RNNG 及び RNNG を、統語的汎化能力の観点から評価する。評価データとしては、SyntaxGym ベンチマーク [26] における 5 つのテストサーキット [12] を用いた。各サーキットは、以下のそれぞれの言語現象を対象とする：Agreement、Licensing、Garden-Path Effects、Center Embedding、Long-Distance Dependencies。ただし、Garden-Path Effects は NP/Z 及び主動詞/縮約関係節ガーデンパス文の扱いをテストするものであるが、このうち NP/Z ガーデンパス文は二分木化の際に取り除かれた「カンマ (,)」を手がかりとしないと解けないデザインとなっているため本研究では対象外とした。SyntaxGym における残り 1 つのサーキットである Gross Syntactic State を本研究で用いないのも同様の理由による。

SyntaxGym において、言語モデルの統語的汎化能力は、統制された複数文の間での確率的予測の違いによって測られる。例えば、Agreement における、前置詞句を含む主語と述語の一致のテストでは、言語モデルは正文である (1a) の下線部に非文である (1b) の下線部よりも高い確率を付与していれば正解となる：

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

M-RNNG 及び RNNG を単語列に対する推論のモデルとして使うために、単語同期型ビームサーチ [27] を用いた。単語同期型ビームサーチは、各アクション毎の足切りとは別に、各単語の生成を行うアクション GEN(x) 毎にも足切りを行う。本研究で新たに導入した、音形のない要素の生成 NULL(*x) アクションを持つ M-RNNG にも直接適用可能である。先行研究 [4] に従い、アクションビーム幅を 100、単語ビーム幅を 10、ファストトラック幅を 1 に設定した。

4 結果と考察

4 種類のデータで学習した M-RNNG および RNNG の、サーキットごとの結果を、図 3 に示す。棒グラフはシードの異なる 3 つのモデルの結果の平均値を表し、それぞれの点は各シードの結果を表す。

まず、音形のない要素の有無による M-RNNG 間の精度の違いを見ると、5 つのサーキットのうち Licensing を除く 4 つで、機能範疇と痕跡の両方を含めたデータで学習したモデルの精度が最も高かった。これらの音形のない要素が、統語構造の適切な分析に貢献したと考えられる。たとえば、Agreement では、機能範疇と痕跡の両方を含めることで初めて高い精度が出ており、時制の機能範疇が一致素性を明示していること（tns_pres と

1) M-RNNG の実装は構文木の変換コードと共に後日公開予定である。RNNG の実装は以下を利用した：https://github.com/aistairc/rnng-pytorch。

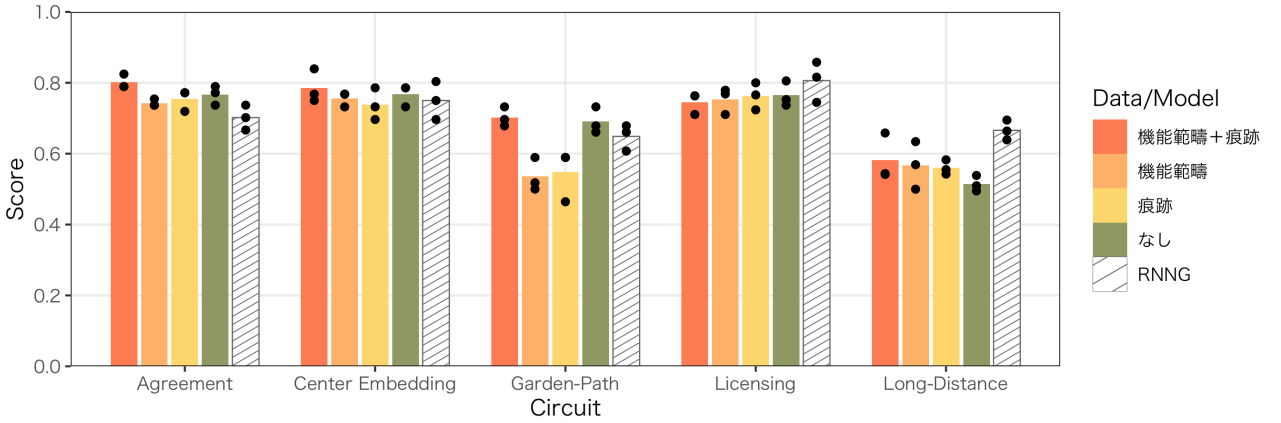

図 3　4 種類のデータで学習した M-RNNG および RNNG の、サーキットごとの結果。棒グラフはシードの異なる 3 つのモデルの結果の平均値を表し、それぞれの点は各シードの結果を表す。

tns_pres_3s の区別がある）と、移動の痕跡があることでサーキットに用いられている関係節が解析しやすくなっていることが寄与していると推測される。(2) の文に対する同一シードでの結果を比較すると、機能範疇と痕跡ありの M-RNNG が正文の構文解析を正しくできており、文法性も正しく判断している一方、これらのない M-RNNG は解析ができておらず文法性判断も誤っている。それぞれの解析結果を(3a, b) に示す。なお、RNNG も、(3c) に示すように解析を失敗している。

(2)　The consultants that the secretary doubted {are/*is} good.

(3)　a.　((The (consultants (that ((the secretary) (*tns_pst (doubted *t)))))) (are good))
　　b.　(The (consultants (that (((the secretary) doubted) (are good)))))
　　c.　(S (NP (NP The consultants) (SBAR (WHNP that) (S (NP the secretary) (VP doubted (SBAR (S (VP are (ADJP good)))))))))

また、M-RNNG と RNNG を比較すると、Agreement、Center embedding、Garden-Path の 3 つのサーキットで、機能範疇と痕跡を含めた M-RNNG、およびどちらも含めない M-RNNG が RNNG の精度を上回った。これらのサーキットでは二分木ないし非終端記号の除去という M-RNNG の極小主義的な設計自体が精度を高めたと考えられる。

一方、Licensing と Long-Distance Dependency では全ての M-RNNG の精度が RNNG を下回った。Licensing では、M-RNNG が RNNG に比べて特に苦手とするのが (4) のような *herself* を含む問題であった。非終端記号の除去により、主語名詞句を構造的に抽象化する効果が薄れ、主語に対するジェンダーバイアス [28] が RNNG より残存しやすかった可能性が考えられる。

(4)　The pilot that the teachers met injured {herself/*themselves}.

最後に、本研究の中心的課題ではないが、M-RNNG は RNNG よりも計算効率の面で優れている。32GB の NVIDIA V100 GPU × 1 を用いた BLLIP（LG、約 1.8M 文、約 42M トークン）の 15 エポックの学習に、RNNG は約 13 時間を要したが、M-RNNG は最も学習時間がかかった機能範疇と痕跡ありの条件でも約 6 時間の学習時間しか要さなかった。

5　おわりに

本研究は、有力な言語理論である極小主義に動機づけられた統語構造を用いた新しい言語モデルとして M-RNNG を提案した。SyntaxGym による評価では、一部の文法タスクで、M-RNNG の精度が RNNG を上回った。本実験の結果は、音形のない要素の存在と、全て二分木で非終端記号のない極小主義的な木構造がいずれも言語モデルの文法汎化能力に寄与し得ることを示唆している。提案手法はその他の統語的言語モデル（e.g., Transformer Grammar [6]）にも応用可能であり、理論言語学の知見を生かした人間らしい言語モデルの開発を後押しすると考えられる。

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] Noam Chomsky. **Aspects of the theory of syntax**. MIT Press, Cambridge, MA, 1965.

[2] Noam Chomsky. **The Minimalist Program**. MIT Press, Cambridge, MA, 1995.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In **NAACL 2019**, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Hiroshi Noji and Yohei Oseki. Effective batching for recurrent neural network grammars. In **Findings of ACL 2021**, pp. 4340–4352, Online, August 2021. Association for Computational Linguistics.

[5] Peng Qian, Tahira Naseem, Roger Levy, and Ramón Fernandez Astudillo. Structural guidance for transformer language models. In **ACL-IJCNLP 2021**, 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. **TACL 2022**, Vol. 10, pp. 1423–1439, 12 2022.

[7] Ryo Yoshida and Yohei Oseki. Composition, attention, or both? In **Findings of EMNLP 2022**, pp. 5851–5863, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[8] Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. Recurrent neural network grammars. In **NAACL 2016**, pp. 199–209, San Diego, California, June 2016. Association for Computational Linguistics.

[9] Jeffrey L. Elman. Finding structure in time. **Cognitive Science**, Vol. 14, No. 2, pp. 179–211, 1990.

[10] Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. In **ACL 2018**, pp. 1426–1436, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[11] Ethan Wilcox, Peng Qian, Richard Futrell, Miguel Ballesteros, and Roger Levy. Structural supervision improves learning of non-local grammatical dependencies. In **NAACL 2019**, pp. 3302–3312, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[12] Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger Levy. A systematic assessment of syntactic generalization in neural language models. In **ACL 2020**, pp. 1725–1744, Online, July 2020. Association for Computational Linguistics.

[13] John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan Brennan. Finding syntax in human encephalography with beam search. In **ACL 2018**, pp. 2727–2736, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[14] Ryo Yoshida, Hiroshi Noji, and Yohei Oseki. Modeling human sentence processing with left-corner recurrent neural network grammars. In **EMNLP 2021**, pp. 2964–2973, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[15] Tal Linzen. How can we accelerate progress towards human-like linguistic generalization? In **ACL 2020**, pp. 5210–5217, Online, July 2020. Association for Computational Linguistics.

[16] David Adger. **Core Syntax: A Minimalist Approach**. Oxford University Press, 2003.

[17] Mitchell P. 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.

[18] Eugene Charniak, Don Blaheta, Niyu Ge, Keith Hall, John Hale, and Mark Johnson. BLLIP 1987-89 WSJ Corpus Release 1 LDC2000T43, 2000.

[19] C.D. Manning and H. Schutze. **Foundations of Statistical Natural Language Processing**. MIT Press, Cambridge, MA, 1999.

[20] Richard K. Larson. On the double object construction. **Linguistic Inquiry**, Vol. 19, No. 3, pp. 335–391, 1988.

[21] Kenneth Hale and Samuel Jay Keyser. On argument structure and the lexical expression of syntactic relations. In Kenneth Hale and Samuel Jay Keyser, editors, **The view from Building 20**, pp. 53–109. MIT Press, Cambridge, MA, 1993.

[22] Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. What do recurrent neural network grammars learn about syntax? In **EACL 2017**, pp. 1249–1258, Valencia, Spain, April 2017. Association for Computational Linguistics.

[23] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. **IEEE 1997**, Vol. 45, pp. 2673–2681, November 1997.

[24] Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. Unsupervised recurrent neural network grammars. In **NAACL 2019**, pp. 1105–1117, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[25] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. Transition-based dependency parsing with stack long short-term memory. In **ACL 2015**, pp. 334–343, Beijing, China, July 2015. Association for Computational Linguistics.

[26] Jon Gauthier, Jennifer Hu, Ethan Wilcox, Peng Qian, and Roger Levy. SyntaxGym: An online platform for targeted evaluation of language models. In **ACL 2020**, pp. 70–76, Online, July 2020. Association for Computational Linguistics.

[27] Mitchell Stern, Daniel Fried, and Dan Klein. Effective inference for generative neural parsing. In **EMNLP 2017**, pp. 1695–1700, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[28] Rebecca Marvin and Tal Linzen. Targeted syntactic evaluation of language models. In **EMNLP 2018**, pp. 1192–1202, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[29] Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In **COLING-ACL 2006**, pp. 433–440, Sydney, Australia, July 2006. Association for Computational Linguistics.

[30] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In **ICLR 2015**, San Diego, CA, USA, 2015. Conference Track Proceedings.

A　学習の詳細

前処理　学習、開発、およびテストセットの分割は先行研究 [12] に従った。先行研究 [4] を踏襲し、学習データの頻出語上位 50,000 語を語彙に含め、残りは Berkeley parser's surface feature rule [29] に則って未知語化した。ただし、M-RNNG が REDUCE の際に常にスタックの上位 2 個の要素を対象として合成できるよう、サブワード化は行わなかった。

ハイパーパラメータ　M-RNNG 及び RNNG の学習時のハイパーパラメータは先行研究 [4] を踏襲した。最適化には学習率を 0.001 に設定した Adam [30] を用いた。ドロップアウト率は 0.1、バッチサイズは 512 に設定した。また、15 エポックの学習を行い、各モデルの評価の際には、BLLIP の開発セットで最も損失が小さかったチェックポイントを用いた。