大規模言語モデルを用いたシフト還元型 CCG 構文解析

東 洸介 [1]　平尾 努 [1]
[1] 金沢大学
{higashi@stu,hirao@se}.kanazawa-u.ac.jp

概要

CCG 構文解析において，文中の単語に CCG カテゴリを割り当てる Supertagging は，特に重要な基盤処理であり，さまざまな手法が提案されている．本研究では大規模言語モデル (LLM) を用いてシフト還元動作を予測して解析木を構築しつつ，シフト動作時に Supertagging を行う手法を提案する．提案手法の特徴は明示的にスタックとキューを与えず，文字列で解析状態を表し，1 つ前の状態を参照しつつ解析動作を予測することにある．CCGbank を用いて評価実験を行った結果，1 つ前の状態を参照することで解析性能が向上し，無効な動作の予測が減った．一方，提案手法は短い Supertag に対する正解率は高かったものの，長い Supertag に対する正解率は著しく低かった．

1　はじめに

自然言語処理における構文解析にはさまざまな文法があり，その 1 つに組み合わせ範疇文法 (Combinatory Categorial Grammar, CCG) [1] がある．CCG とは単語にカテゴリを割り当て，それらを組み合わせることで解析木を構築する語彙化文法の 1 つである．CCG で定義されるカテゴリは文を表す"S"や名詞句を表す"NP"などの原始的なカテゴリと"/"や "\" を組み合わせることで作られ，文法的な情報を含んでいる．図 1 に CCG 解析木の例を示す．各単語にはその親に CCG カテゴリが付与されている．"X/Y"は右に"Y"を組み合わせることで全体が"X"になることを意味し，"X\Y"は左に"Y"を組み合わせることで全体が"X"になることを意味している．図 1 の例では"NP[nb]/N"である *These* に対して，"N"である *stocks* を右から組み合わせることで"NP"である *These stocks* を導出している．このようにして CCG カテゴリを組み合わせながら，最終的に"S"を導出することで解析木を得る．

CCG 構文解析を行うにあたって，Supertagging と呼ばれる処理が最も重要である．これは単語に CCG カテゴリ (Supertag と呼ばれる) を割り当てる処理のことで，Supertag が文法的な情報を多く含んでいることから，その割り当てが CCG 構文解析の大部分を占める [2]．Supertagging は系列ラベリングと見なすことができるので，これまでに，Long Short-Term Memory(LSTM)[3] を用いた手法 [4]，大規模言語モデル (LLM) を用いた手法 [5] が提案されている．ただし，これらの手法は Supertagging を単なる系列ラベリングとして捉えているため，それに基づいて解析木を構築する際に CCG カテゴリを組み合わせることができず，解析できないことが起こりうる．

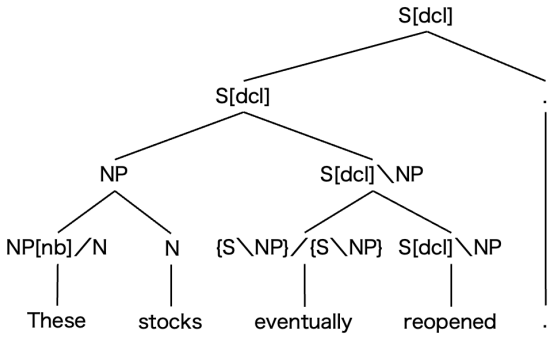

図 1　*These stocks eventually reopened .* の解析木 (CCGbank WSJ_2300 sentence33)．CCG カテゴリに含まれる"(", ")"を"{", "}"に変更している．

これに対し，本研究では Supertagging を系列ラベリングとして捉えるのではなく，シフト還元型解析を用いて解析木を構築しつつ，シフト動作時に Supertagging を行う手法を提案する．この手法は Zhang らの手法 [6] と同様であるが，本研究では中根らの手法 [7] に基づき，LLM を活用しつつ，さらに 1 つ前の解析状態を参照して解析動作を予測することで性能向上を図る．CCGbank [8] を用いて提案手法を評価した結果、Supertagging の正解率は 95.82，CCG 解析木の評価指標である Labeled F-score は 89.25 であった．両者とも現状での世界最高性能

には及ばない結果となったが，長い Supertag に対する正解率が著しく低いことが全体の性能低下を招いたことが明らかとなった．

2 関連研究

Supertagging は文脈を考慮した上で単語に CCG カテゴリを与えるタスクなので、いわゆる系列ラベリングとして定式化できる．よって，LSTM を用いた手法 [4] が提案された．Zhao ら [5] は，デコーダモデルである LLM を用いて Supertagging を行う手法を提案し，世界最高性能を達成した．これらの手法は入力単語列に対して，単語，Supertag の依存関係を考慮しつつ，一番尤もらしい Supertag 系列を予測する．しかし，解析木を構築することを前提としていないため，本来，組み合わせることのできない，つまり，解析木を構築できない Supertag を出力してしまう可能性がある．また，系列ラベリングではないが，Graph Convolutional Networks を用いた手法 [9] や Holographic Embeddings を用いることで単語ベクトルから，句，文の埋め込みベクトルを合成して Supertagging を行う手法 [10] も提案されており，非常に高い性能を示している．

Supertagging のみに注力するのではなく解析木を構築することを目標とした解析法も当然，提案されている．Zhang ら [6] は膨大な特徴量と線形モデルを用いたシフト還元型 CCG 構文解析を提案した．Ambati ら [11] はこれをニューラルネットワークを用いたエンコーダベースの手法に拡張し，性能向上を行った．しかし，この手法ではスタックやキューにある全ての単語や部分木を参照することができず，動作の予測に限界があると考えられる．これを解消する手法として，Xu [12] は LSTM を用いた手法を提案した．LSTM を用いることでスタックやキューにある全ての単語や部分木を参照することができるものの，明示的に木として扱ってないことから，動作を予測する性能に限界があると考えられる．

一方，中根ら [7] は，句構造解析において，スタックとキューの内容をすべて参照する，かつ，スタックに積まれた部分木は S 式として表現することで部分木を考慮するシフト還元型解析を LLM を活用することで実現した．LLM が S 式を扱えるという利点を活かしており、非常に高い解析性能を達成した．

3 提案手法

本研究では句構造解析のために提案された中根らの LLM を用いたシフト還元型句構造解析法 [7] を CCG 構文解析のために拡張する．

スタックとキューを明示的に与えず，文全体と<head>というタグを用いてシフト還元動作を予測する点は中根らの手法 [7] と同様であるが，シフト動作時に Supertagging を行う点，予測される動作にフィニッシュ動作が含まれる点，動作と CCG カテゴリの予測を異なる LLM で行う点，1 つ前の状態を参照して動作を予測する点が異なる．

3.1 シフト還元型解析

古くからある構文解析法の 1 つで，解析済みの部分木を格納するスタック，未解析の単語を格納するキューの 2 つの要素から構成されており，以下のシフト，還元動作を繰り返し実行することで解析木を構築する．

シフト キューの先頭の単語を取り出し，スタックに積む．

還元 (binary) スタックの上部 2 つの部分木を取り出し，それらの親となるラベルで 1 つの部分木を作り，スタックに積む．

還元 (unary) スタックの上部 1 つの部分木を取り出し，その親となるラベルで木を作り，スタックに積む．

本研究では，Zhang らの手法 [6] と同様に，従来のシフト還元型解析にあるシフト，還元動作に加えて，フィニッシュ動作を追加する．これはフィニッシュ動作が予測されたときに解析を終了するものである．この動作によって解析が終了したことを明確にすることができる．また，シフト動作時にその単語に対して Supertagging を行う．

3.2 プロンプト

本研究では 2 つの LLM を使用して解析を行う．1 つ目は動作を予測するモデル，2 つ目は予測された動作に対して CCG カテゴリを予測するモデルである．予備的に 1 つのモデルに動作と CCG カテゴリを同時に予測させる実験を行ったが，CCG カテゴリが長く，動作すらも正しく予測することができないことが多々あったため，本研究では動作予測と CCG カテゴリ予測の LLM を別々に用意した．

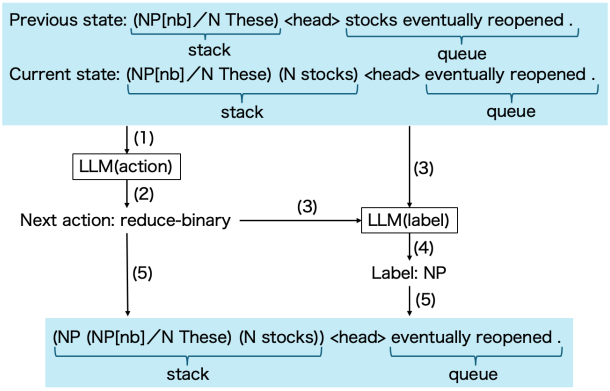

図 2　提案手法の動作例.

それぞれの LLM への入力は状態を表す文を以下のプロンプトとして与える.

`Previous state:`「スタック」<head>「キュー」

`Current state:`「スタック」<head>「キュー」

スタックにはシフト動作後の単語と還元動作後の部分木が格納され，キューには未解析の単語が格納されている．なお，部分木は S 式として表わす．`Previous state` が 1 つ前の状態であり，`Current state` が現在の状態である．1 つ前の状態に適用された動作や CCG カテゴリは明示しない．初期状態のみ，プロンプトを以下のようにする.

`Initial state:` <head>「単語列」

この時のスタックは空文字列で，キューは解析する文である.

入力されたプロンプトに対し，動作予測用の LLM は以下の動作のいずれかを出力する.

`Next action: shift or reduce-binary or reduce-unary or finish`

シフト，還元動作が予測された場合のみ，CCG カテゴリ予測用の LLM を用いて CCG カテゴリを出力する．図 2 に提案手法の動作例を示す．(1) では `Previous state` と `Current state` を，動作予測用の LLM に入力する．(2) では入力に対して予測された動作を出力する．(3) では予測された動作と `Previous state` と `Current state` を，CCG カテゴリ予測用の LLM に入力する．(4) では入力に対して予測された CCG カテゴリを出力する．(5) では予測された動作と CCG カテゴリを用いて状態に適用して変更する.

4　実験設定

4.1　データセットと比較した手法

解析器の訓練と評価には英語 CCGbank[8] を用いた．本研究では標準的な分割方法に従い，CCGbank を訓練データ（セクション 02-21），検証データ（セクション 00），テストデータ（セクション 23）に分割した．それぞれ 39604 文，1913 文，2407 文である.

提案手法の有効性を示すため，LSTM を用いた Vaswani らの手法 [4]，Graph Convolutional Networks を用いた Tian らの手法 [9]，LLM を用いて現在の世界最高性能を達成した Zhao らの手法 [5]，Holographic Embeddings を用いた Yamaki らの手法 [10] との比較評価を行った.

4.2　学習・評価の設定

本研究で用いた 2 つの LLM はそれぞれ Unsloth の 4bit 量子化された Qwen3-14B [13] であり，QLoRA [14] を用いて追加学習を行った．モデルの選択やハイパーパラメータの選択は検証データの損失を元に行った．モデル学習の各種ハイパーパラメータを表 1 に示す．なお，ハイパーパラメータの設定は中根らの実験設定 [7] を参考にしている．本研究では 2 つの LLM を使用するが，各種ハイパーパラメータは同じである.

解析器の評価指標には Supertagging の正解率 (**Acc**)，CCG 解析木の依存関係を示す Labeled F-score(**LF**) を用いた．なお、提案手法の Supertagging の結果に対して C&C parser[15] を用いて解析木を構築した際の Labeled F-score も算出した.

5　結果と考察

結果を表 2 に示す．-history は 1 つ前の状態を参照しないモデル，+history は 1 つ前の状態を参照するモデルである．-history モデルでは解析できない文が 1 つあり，スコアの算出から除外している．また，キューが空で，スタックが 1 つの木になっているがフィニッシュ動作を予測できない文が 3 つあった．解析木を構築することはできているため，スコアの算出に含めている．一方，+history モデルでは全ての文を解析することができた．既存研究のスコアはそれぞれの論文に掲載されたものである.

表より，提案手法は，Vaswani らの手法には勝っ

表 1　モデル学習時のハイパーパラメータ.

Hyperparameter	Value
Training epochs	3(action), 3(label)
Batch size	16
Learning rate	1e-5
Learning rate scheduler	Linear warm-up and cosine annealing
Warm-up step	1 epoch
Weight decay	0.1
Optimizer	paged_adamw_32bit
LoRA r	32
LoRA α	16
LoRA dropout	0.0
LoRA target modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj

表 2　既存研究との評価結果.

Model	Parser	Acc	LF
Vaswani ら [4]	C&C	94.5	88.32
Tian ら [9]	EasyCCG[16]	96.25	90.58
Yamaki ら [10]	C&C	96.60	92.12
Zhao ら [5]	C&C	96.64	92.05
提案手法 -history	C&C	95.54	88.47
提案手法 -history	shift-reduce	-	87.81
提案手法 +history	C&C	95.82	89.25
提案手法 +history	shift-reduce	-	89.10

たものの，その他の手法には及ばない結果であった．Supertagging の正解率では Zhao らの手法に対して約 0.8 ポイント，Labeled F-score では 2.8 ポイントの差がある．また，シフト還元型解析により構築した解析木と C&C parser を用いて構築した解析木を比較すると C&C parser の方が良いスコアであった．これより，還元動作時に予測した CCG カテゴリが CCG の組み合わせ規則に則っていない可能性が示唆される．また，既存研究に対する Supertagging の正解率の差と比較して Labeled F-score の差の方が大きいことから，解析木を構築する過程で CCG カテゴリに致命的な誤りが生じていることも示唆される．

Supertagging の結果を詳細に分析するため，Supertag の長さが 1 以上 10 以下，11 以上 20 以下，21 以上の 3 つのグループに対して Supertagging の正解率を比較した．結果を図 3 に示す．図より，Supertag の長さが 10 を超えると急激に Supertagging の正解率が低下していることが分かる．長い Supertag に対して性能が劣化することはある程度予測されることであるが，6 ポイント以上の劣化はそれを考慮したとしても大きな差である．このような長い Supertag

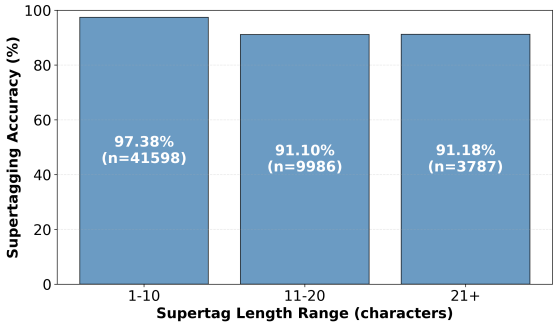

図 3　長さごとの Supertagging の正解率.

の間違いが Supertagging の正解率に対して顕著に低い Labeled F-score の要因ではないかと考える．

一方，-history モデルと+history モデルを比較すると 1 つ前の状態を参照することで性能が向上していることが分かる．-history モデルではキューが空であるのにシフト動作を予測したり，スタックが 1 つの木であるのに還元 (binary) 動作を予測したりするなどの無効な動作を予測する場合があった．こうした無効な動作は，状態に適用して変更することができないため，解析を続けることができなくなる．しかし，+history モデルでは無効な動作を予測した場合，状態を変更することはできないものの，1 つ前の状態と現在の状態を同じにすることでもう一度動作を予測することができ，解析を進めることができる．このようにすることで+history モデルでは一度無効な動作を予測したが，その後に予測される動作が変化することで解析を続けることができた．よって，1 つ前の状態を参照しつつ解析を行うことは有効であると考える．

6　おわりに

本研究では CCG 構文解析のために LLM にシフト還元動作を予測させる手法を提案した．同様の手法は中根らによって提案されているが，シフト動作時に Supertagging を行う点，予測される動作にフィニッシュ動作が含まれる点，動作と CCG カテゴリを予測する LLM が異なる点，1 つ前の状態を参照する点が異なる．提案手法を CCGbank を用いて評価したところ，Supertagging の正解率は 95.82，CCG 解析木の評価指標である Labeled F-score は 89.25 であり，特に Labeled F-score については，現在の世界最高性能には遠く及ばなかった．この原因は，提案手法では長さが 11 以上の Supertag に対する正解率が著しく低いことにあると考える．今後，こうした長い Supertag に対する正解率の向上に取り組みたい．

謝辞

本研究の一部は JSPS 科研費 25K03191 の助成を受けたものです．また，Labeled F-score の算出方法についてご教示いただいた立命館大学の山木良輔氏に深く感謝申し上げます．本研究では，東京大学，情報基盤センターのスーパーコンピュータ Miyabi および産総研及び AIST Solutions が提供する ABCI 3.0 を利用した．

参考文献

[1] Mark Steedman. **The Syntactic Process**, Vol. 24. MIT Press, Cambridge, MA, 2000.

[2] Srinivas Bangalore and Aravind K. Joshi. Supertagging: An approach to almost parsing. **Computational Linguistics**, Vol. 25, No. 2, pp. 237–265, 1999.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. **Neural Computation**, Vol. 9, No. 8, pp. 1735–1780, 11 1997.

[4] Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. Supertagging with LSTMs. 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. 232–237, San Diego, California, June 2016. Association for Computational Linguistics.

[5] Jinman Zhao and Gerald Penn. LLM-supertagger: Categorial grammar supertagging via large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, **Findings of the Association for Computational Linguistics: EMNLP 2024**, pp. 697–705, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[6] Yue Zhang and Stephen Clark. Shift-reduce CCG parsing. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, **Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies**, pp. 683–692, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[7] 中根稜介, 前川在, 上垣外英剛, 平尾努, 奥村学. 大規模言語モデルを用いたシフト還元型句構造解析. 言語処理学会第 31 回年次大会 発表論文集, pp. 3788–393, 2025.

[8] Julia Hockenmaier and Mark Steedman. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. **Computational Linguistics**, Vol. 33, No. 3, pp. 355–396, 2007.

[9] Yuanhe Tian, Yan Song, and Fei Xia. Supertagging Combinatory Categorial Grammar with attentive graph convolutional networks. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 6037–6044, Online, November 2020. Association for Computational Linguistics.

[10] Ryosuke Yamaki, Tadahiro Taniguchi, and Daichi Mochihashi. Holographic CCG parsing. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, **Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 262–276, Toronto, Canada, July 2023. Association for Computational Linguistics.

[11] Bharat Ram Ambati, Tejaswini Deoskar, and Mark Steedman. Shift-reduce CCG parsing using neural network models. 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. 447–453, San Diego, California, June 2016. Association for Computational Linguistics.

[12] Wenduan Xu. LSTM shift-reduce CCG parsing. In Jian Su, Kevin Duh, and Xavier Carreras, editors, **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing**, pp. 1754–1764, Austin, Texas, November 2016. Association for Computational Linguistics.

[13] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

[14] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, **Advances in Neural Information Processing Systems**, Vol. 36, pp. 10088–10115. Curran Associates, Inc., 2023.

[15] Stephen Clark, Darren Foong, Luana Bulat, and Wenduan Xu. The Java Version of the C&C Parser: Version 0.95. Technical report, University of Cambridge Computer Laboratory, August 2015.

[16] Mike Lewis and Mark Steedman. A* CCG parsing with a supertag-factored model. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 990–1000, Doha, Qatar, October 2014. Association for Computational Linguistics.